



OpenAir

XML API & SOAP API

Copyright © 2013, 2024, Oracle and/or its affiliates.

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, then the following notice is applicable:

U.S. GOVERNMENT END USERS: Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs) and Oracle computer documentation or other Oracle data delivered to or accessed by U.S. Government end users are "commercial computer software" or "commercial computer software documentation" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, reproduction, duplication, release, display, disclosure, modification, preparation of derivative works, and/or adaptation of i) Oracle programs (including any operating system, integrated software, any programs embedded, installed or activated on delivered hardware, and modifications of such programs), ii) Oracle computer documentation and/or iii) other Oracle data, is subject to the rights and limitations specified in the license contained in the applicable contract. The terms governing the U.S. Government's use of Oracle cloud services are defined by the applicable contract for such services. No other rights are granted to the U.S. Government.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Inside are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Epyc, and the AMD logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information about content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services unless otherwise set forth in an applicable agreement between you and Oracle. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services, except as set forth in an applicable agreement between you and Oracle.

If this document is in public or private pre-General Availability status:

This documentation is in pre-General Availability status and is intended for demonstration and preliminary use only. It may not be specific to the hardware on which you are using the software. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to this documentation and will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

If this document is in private pre-General Availability status:

The information contained in this document is for informational sharing purposes only and should be considered in your capacity as a customer advisory board member or pursuant to your pre-General Availability trial agreement only. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described in this document may change and remains at the sole discretion of Oracle.

This document in any form, software or printed matter, contains proprietary information that is the exclusive property of Oracle. Your access to and use of this confidential material is subject to the terms and conditions of your Oracle Master Agreement, Oracle License and Services Agreement, Oracle PartnerNetwork Agreement, Oracle distribution agreement, or other license agreement which has been executed by you and Oracle and with which you agree to comply. This document and information contained herein may not be disclosed, copied, reproduced, or distributed to anyone outside Oracle without prior written consent of Oracle. This document is not part of your license agreement nor can it be incorporated into any contractual agreement with Oracle or its subsidiaries or affiliates.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc>

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info> or visit <http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs> if you are hearing impaired.

Sample Code

Oracle may provide sample code in SuiteAnswers, the Help Center, User Guides, or elsewhere through help links. All such sample code is provided "as is" and "as available", for use only with an authorized NetSuite Service account, and is made available as a SuiteCloud Technology subject to the SuiteCloud Terms of Service at www.netsuite.com/tos, where the term "Service" shall mean the OpenAir Service.

Oracle may modify or remove sample code at any time without notice.

No Excessive Use of the Service

As the Service is a multi-tenant service offering on shared databases, Customer may not use the Service in excess of limits or thresholds that Oracle considers commercially reasonable for the Service. If Oracle reasonably concludes that a Customer's use is excessive and/or will cause immediate or ongoing performance issues for one or more of Oracle's other customers, Oracle may slow down or throttle Customer's excess use until such time that Customer's use stays within reasonable limits. If Customer's particular usage pattern requires a higher limit or threshold, then the Customer should procure a subscription to the Service that accommodates a higher limit and/or threshold that more effectively aligns with the Customer's actual usage pattern.

Table of Contents

OpenAir XML and SOAP API Overview	1
OpenAir API Best Practice Guidelines	4
Build the API Integration	4
Optimize the API Integration	5
Maintain the API Integration	6
Getting Started with OpenAir XML API and SOAP API	8
XML Schema and WSDL Definition Documents	11
XML API Call and Response	12
Building SOAP API Client Applications with Apache Axis	17
Java Sample Code – Authentication (SOAP API)	20
Building SOAP API Client Applications with Microsoft Visual Studio IDE	22
C# Sample Code — Read (SOAP API)	23
Authentication	27
SessionHeader	27
OAuth 2.0 Authorization	30
Managing API Integration Applications in OpenAir	30
Auditing and Managing OAuth 2.0 Authorizations	38
OAuth 2.0 for Integration Applications Developers	40
Authorizing Applications to Access OpenAir on Your Behalf	49
Reading Objects	52
Read Methods	52
Read Attributes	54
Filtering	57
Pagination	59
Sorting	59
Adding, Updating and Upserting Objects	61
Add, Update and Upsert Attributes	61
Related Object Lookup Using the XML API	62
Related Object Lookup Using the SOAP API	63
Deleting Objects	65
Approval-Related Operations	66
Approval	67
Utility Operations	68
OpenAir Pages Supported by the Make URL Operation	68
Handling Errors	72
Error	73
Error Codes	73
API Limits	98
RateLimit	99
Web Services Logs	101
XML API Commands	102
Add	102
Approve	104
Auth	105
Login	106
CreateUser	106
Delete	108
MakeURL	108
Modify	109
ModifyOnCondition	111
Read	113
Reject	117
RemoteAuth	118

Report	118
Submit	119
Time	120
Unapprove	120
Version	121
Whoami	122
SOAP API Commands	123
Common Object Types Used With SOAP API Commands	123
oaBase	123
Attribute	124
UpdateResult	124
add()	124
approve()	125
ApproveRequest	126
ApproveResult	127
createUser()	127
delete()	129
login()	130
LoginParams	131
LoginResult	131
logout()	132
makeURL()	132
MakeURLRequest	133
MakeURLResult	134
modify()	134
Sample Codes — C#	135
Sample Codes — Java	137
read()	138
ReadRequest	139
ReadResult	139
Sample Codes — C#	139
Sample Codes — Java	143
reject()	146
RejectRequest	147
RejectResult	147
servertime()	147
submit()	148
SubmitRequest	149
SubmitResult	150
unapprove()	150
UnapproveRequest	151
UnapproveResult	151
upsert()	152
version()	153
VersionResult	154
whoami()	154
Business Object Properties Overview	156
System Fields	156
Calculated Fields	157
Required Fields	157
Reference Fields	158
External ID Fields	159
Date Fields	159
Date	159
Address Fields	160

Company and User Settings	161
Flag	162
oaSwitch	162
Money Fields	162
Custom Fields	163
CustomField	167
XML and SOAP API Business Object Reference	168
AccountingPeriod	174
Actualcost	175
Address	176
Agreement	177
Agreement_to_project	179
ApprovalLine	179
ApprovalProcess	182
Attachment	182
Attribute	185
AttributeDescription	185
Attributeset	186
BillingSplit	187
Booking	187
BookingByDay	190
BookingType	191
Booking_request	192
Budget	194
BudgetAllocation	194
Category	195
Category_<N>	196
Ccrate	197
Company	198
Contact	200
Costcategory	202
Costcenter	202
Costtype	203
Currency	204
Currencyrate	205
CustField	206
Customer	208
CustomerLocation	213
Customerpo	214
Customerpo_to_project	215
CustomerProspect	216
Deal	216
Dealcontact	217
Dealschedule	217
Department	218
Entitytag	218
Envelope	219
Estimate	223
Estimateadjustment	223
Estimateexpense	224
Estimatelabor	225
Estimatemarkup	225
Estimatephase	226
Event	226
ExpensePolicy	227

ExpensePolicyItem	228
Filter	229
Filterset	230
ForexInput	231
Fulfillment	232
Hierarchy	233
HierarchyNode	234
History	235
HistoryNotes	236
ImportExport	237
Invoice	238
InvoiceLayout	241
Issue	241
IssueCategory	242
IssueSeverity	243
IssueSource	243
IssueStage	244
IssueStatus	244
Item	245
ItemToUserLocation	246
Jobcode	247
JobCodeUsed	248
Leave_accrual_rule	248
Leave_accrual_rule_to_user	249
Leave_accrual_transaction	250
LoadedCost	251
Module	252
Newsfeed	252
NewsfeedMessage	253
Payment	254
Paymentterms	254
Paymenttype	255
Payrolltype	256
PendingBooking	257
Preference	259
Product	259
Project	260
Projectassign	270
ProjectAssignmentProfile	271
Projectbillingrule	272
Projectbillingtransaction	277
ProjectBudgetGroup	279
ProjectBudgetRule	281
ProjectBudgetTransaction	283
Projectgroup	285
Projectlocation	286
ProjectPricing	286
ProjectStage	287
Projecttask	288
Projecttaskassign	293
ProjecttaskEstimate	294
Projecttask_type	295
Proposal	296
Proposalblock	297
Proxy	298

Purchase_item	299
Purchaseorder	302
Purchaser	304
Purchaserequest	305
Ratecard	307
RateCardItem	307
Reimbursement	308
Repeat	309
Report	310
Request_item	310
ResourceAttachment	312
Resourceprofile	313
Resourceprofile_type	314
ResourceRequest	315
ResourceRequestQueue	316
ResourceSearch	317
RevenueContainer	320
RevenueProjection	321
Revenue_recognition_rule	325
Revenue_recognition_rule_amount	328
Revenue_recognition_transaction	329
RevenueStage	331
Role	332
Schedulebyday	332
Scheduleexception	333
Schedulerequest	335
Schedulerequest_item	336
Slip	337
SlipProjection	341
Slipstage	343
SummaryView	344
TagGroup	345
TagGroupAttribute	345
TargetUtilization	346
Task	346
TaskAdjustment	351
TaskTimecard	351
TaxLocation	353
TaxRate	354
Term	355
Ticket	355
Timecard	359
Timesheet	360
Timetype	362
Todo	363
Uprate	364
User	365
UserLocation	375
UserWorkschedule	376
Vendor	378
Viewfilter	379
Viewfilterrule	380
WorkscheduleWorkhour	381
Workspace	381
Workspacelink	382

Workspaceuser	383
Release History	384

OpenAir XML and SOAP API Overview

The OpenAir XML API and SOAP API provide programmatic access to your OpenAir account data and business processes without using the OpenAir UI. You can use the OpenAir XML API and SOAP API to perform operations and integrate OpenAir with other applications.

OpenAir XML API interfaces with the data layer in OpenAir and provides the most comprehensive programmatic access to your OpenAir data.

OpenAir SOAP API uses SOAP-based web services, and serves as a wrapper around OpenAir XML API, providing the same or similar functionality.

This guide provides a reference for using OpenAir XML API and SOAP API.

- Review [OpenAir API Best Practice Guidelines](#) before you start using OpenAir XML API or SOAP API.
- Review [Getting Started with OpenAir XML API and SOAP API](#) for guidance about setting up and using OpenAir XML API and SOAP API. The section also includes information about:
 - [XML Schema and WSDL Definition Documents](#)
 - [XML API Call and Response](#)
 - [Building SOAP API Client Applications with Apache Axis.](#)
 - [Building SOAP API Client Applications with Microsoft Visual Studio IDE](#)
- The following help topics describe specific type of operations:
 - [Authentication](#)
 - [Reading Objects](#)
 - [Adding, Updating and Upserting Objects](#)
 - [Deleting Objects](#)
 - [Approval-Related Operations](#)
 - [Utility Operations](#)
 - [Handling Errors](#)
- The final section gives reference information about API commands, business objects and object properties.
 - [XML API Commands](#)
 - [SOAP API Commands](#)
 - [Business Object Properties Overview](#)
 - [XML and SOAP API Business Object Reference](#)

Key Concepts

OpenAir XML API and SOAP API are application programming interfaces (API) – a set of functions and procedures that let application developers access OpenAir functionality and data within their application. The information is exchanged across the internet in a consistent format.

Note: The OpenAir XML API and SOAP API follow the same security best practice as OpenAir. All data is encrypted in transport using the industry standard transport layer security (TLS) protocol.

XML

XML stands for eXtensible Markup Language. It was designed to store and transport information wrapped in tags. Unlike HTML, XML does not use tags that are predefined in an XML standard. The XML tags and document can be defined for the specific purpose of an application. XML is self-descriptive – each XML start tag and end tag pair provide a context for the information contained within. Each XML start tag and end tag pair and everything within these tags constitutes an XML element. XML documents are formed as element trees. An XML tree structure starts at a root element and branches from the root to child elements. All elements can have sub elements (child elements) or text content. All elements can have attributes providing information related to that element.

The OpenAir XML API uses XML syntax for object information (information contained in the OpenAir records and fields), commands (operations performed on the OpenAir records and fields) or the outcome of these commands in the request sent by the client or the response returned by the API. It uses predefined tags for commands, object types and object properties, as well as attributes providing additional information related to an operation.

SOAP

SOAP stands for Simple Object Access Protocol. It is an XML-based application communication protocol for accessing web services that can be used by client applications to perform operations such as retrieving, adding, updating and deleting data from OpenAir. OpenAir SOAP web services (SOAP API) serve as a wrapper around OpenAir XML API, providing the same or similar functionality.

XSD

XSD stands for XML Schema Definition. It refers to the XML Schema language which is used to describe the structure of an XML document, including the elements and attributes that can appear in an XML document, and information related to these elements and attributes such as their data types. OpenAir provides an XML schema for each business object type supported by OpenAir XML API and SOAP API.

WSDL

WSDL stands for Web Service Description Language. A WSDL document uses an XML-based language to describe web services. OpenAir provides a generic WSDL definition and an account-specific WSDL definition. These WSDL definitions describe the OpenAir SOAP API and tells the client how to compose a web service request.

For more information about WSDL, see <https://www.w3.org/TR/wsdl/>.

Standards Compliance

OpenAir XML API is implemented in accordance with the following specifications

Standard Name	Website
Extensible Markup Language (XML) 1.0 (Fourth Edition)	http://www.w3.org/TR/xml/

OpenAir SOAP API observes the following specifications

Standard Name	Website
Simple Object Access Protocol (SOAP) 1.1	https://www.w3.org/TR/2000/NOTE-SOAP-20000508/
Web Service Description Language (WSDL) 1.1	https://www.w3.org/TR/2001/NOTE-wsdl-20010315

HTTPS Transport – Connection to OpenAir API must be made over a secure layer using the HTTPS protocol. Ensure connections from any integration tools have supported cipher suites enabled. See the help topic [TLS Protocol and Cipher Suites](#).

OpenAir API Best Practice Guidelines

Before you start using OpenAir API, make sure that your OpenAir account is fully configured and in production. There are many ways to customize your OpenAir account to meet your company's unique business requirements. Account customization allows you to maximize the effectiveness of OpenAir for your company. However, it is best to start using OpenAir API to access or modify your account data after the account is deployed and in use.

- [Build the API Integration](#)
- [Optimize the API Integration](#)
- [Maintain the API Integration](#)

Note: You should work with your OpenAir Professional Services consultant to design integrations leveraging the OpenAir API. The knowledge you gain about how tables and data fields are used in your business processes will save development time on the front end and help you optimize your integration on an ongoing basis.

Build the API Integration

The OpenAir API provides tools for building a powerful integration. Take some time to plan what you want to do, design your integration and document the process, develop your integration, and test it extensively in a sandbox account. A sandbox account is a safe environment you can use to test new integrations and processes without impacting your production account data. See the help topic [OpenAir Sandbox](#).

Step 1: Plan What You Want To Do

Think about what you are trying to achieve in your OpenAir implementation and how the OpenAir API can increase your ability to do that. Ask and answer the following questions:

- What are your critical processes? How can the API integration help you streamline them?
- What are your repetitive tasks? How can the API integration help automate those tasks?
- What will the API integration be able to do that can help your employees save time?

Step 2: Design Your API Integration

Take time to develop a document that describes your API integration. Your OpenAir Professional Services consultant can expedite this effort and help reduce development time. Gain an understanding of what you are trying to achieve so that key players in your organization can provide valuable feedback before you the development starts.

Step 3: Develop Your Integration

Read the OpenAir API documentation in its entirety, talk to your OpenAir Professional Services consultant, and learn about the OpenAir data model and how it is used. Links to key information are provided in [OpenAir XML and SOAP API Overview](#) and [Getting Started with OpenAir XML API and SOAP API](#).

- Develop the API integration with the help of your OpenAir Professional Services consultant. Incorporate labels and terms that will both reduce confusion and enhance the integration you develop.

- Test the API integration in your sandbox account. It is crucial that you use a non-production environment until you can be sure that the integration runs smoothly without error and does not damage vital production data.

Optimize the API Integration

The following suggestions will help you get the most out of your API integration. Discuss them with your OpenAir Professional Services consultant to ensure you understand why they improve the efficiency and effectiveness of your integration.

Make Batch Calls

When making calls in your API integration, request and update data records in batches. Typically, you should group records into batches of 500, but the specifics vary depending on the context and the expected volume of data to be transacted. Even when requesting data based on filtering criteria, multiple read operations can be specified within one read request. You should run batch operations during off-peak hours to minimize impact on integration performance.

Make Fewer Calls

Reducing the number of calls you make to the OpenAir API improves the performance of your integration. Because API calls require a call and response over the public Internet, they can consume both time and resources. Minimizing the number of calls you make increases the speed at which your API integration operates. Running batch operations during off-peak hours also minimizes impact on performance.

You should read and update custom fields inline with standard object properties instead of using the legacy `custom_equal` method. If you work with the OpenAir SOAP API, use the account-specific WSDL, which includes custom fields. If you work with OpenAir XML API, set the `enable_custom` attribute to 1. For more information about working with custom fields, see [Custom Fields](#).

You should not run the API from multiple clients simultaneously. Although they are, even though the API technically allows concurrent connections. Concurrent connections are technically possible but they may cause performance deterioration due to contention on Web and database servers' resources and can affect the performance of both the API integration and user interaction.

 **Note:** Throttling controls apply to OpenAir API usage. Batching multiple API operations into one request and caching data locally are the best methods to avoid our servers ever triggering throttling controls.

For information about throttling controls, see [API Limits](#).

Cache Locally

Transactional records in OpenAir contain as many as a dozen foreign keys that refer to other records in the system. When retrieving a batch of transactional records, you will often be retrieving many records with the same foreign key value. You could retrieve many charges for the same project (with the same `projectid`) or many time entries for the same user (with the same `userid`), for example.

To optimize performance, after retrieving a batch of charges, you should construct a message to retrieve all the project records associated with those charges and hold those project records locally, either in memory or using persistent storage, to use with the next batch. To ensure the persistent cache is up to date, the client application can retrieve data using a `newer-than` filter. You should retrieve list data, cache it, and then keep it in synchronization with OpenAir by retrieving records that have been modified since

the previous update. The OpenAir XML API and SOAP API lets you read records that have been deleted since the last request, which is another way to ensure your local list data cache is up-to-date.

Another way of optimizing performance is paying attention to the range of possible foreign key values for an attribute. This range of values could be small. Even a large OpenAir account may have only 3 or 4 time types, for example, and every time entry record will then have one of those 3 or 4 values. After you retrieve the time type records, you can hold them locally for an indefinite period as time type values change infrequently and the same small set can be referenced on every time entry transaction.

Use External IDs

In addition to caching locally, you can use an external ID [externalid] saved in OpenAir in place of an OpenAir internal ID when related data is being referenced on a OpenAir record during a modify or add operation. This often avoids the need to request the OpenAir list data in the first place. The integration logic should properly manage possible errors if the externalid was not found in OpenAir system. See [Update Using External ID as Foreign Key Lookup — C#](#) and [Modify with Foreign Key Lookup — Java](#).

Use Date Filters to Limit Amount of Data Processed

Make sure you are only requesting data that is new, modified, or deleted. When requesting list elements like projects, as mentioned previously, you should keep a local cache of records. See [Cache Locally](#). Issuing a read method call that requests records that have been added, modified or deleted since the previous integration run allows the integration logic to process only a small data set of changed records. By default, the newer-than filter uses the updated date on each record, which is the timestamp appropriate for such use. For code examples, see [Read Not Exported Expense Reports with all Method and Date Filter — C#](#) and [Read with all Method and Date Filters — Java](#).

Use not-exported Filters to Limit Amount of Data Processed

Make sure you are only requesting data that has not previously been exported. For transactional exports, you should export approved entries and mark these records in the OpenAir system as having been exported. You can configure OpenAir to lock exported data so that it cannot be modified after the export. You can also configure OpenAir reports and lists to show records as having been exported to another system. Export child elements and mark these child elements as being exported. For example:

- Use the not-exported filter when you export invoices [Invoice] and their charges [Slip]. Since charges are the child elements of an invoice, you can mark each charge as exported. Subsequent integration runs issue a read request and the not-exported filter only returns qualifying transactions, that is transactions not previously processed.
- Use the not-exported filter to export Task records for timesheet information.
- Use the not-exported filter to export Ticket records for export expense information.

For code examples, see [Read Not Exported Charges with all Method — C#](#) and [Read Not Exported Charges with all Method — Java](#).

Maintain the API Integration

Before you use your API integration, there are two additional tasks to perform: set up the storage of communication logs and determine a process for upgrading the OpenAir system. Each is explained as follows.

Store Communication Logs

In the event of an API integration error, your OpenAir Professional Services consultant or OpenAir Customer Support can help you troubleshoot the error. To do so, you need to be able to provide them with both the request code and associated response. Store a log of recent API communications as well as the exact timestamps of API requests to OpenAir servers. You should create a communication log that stores a minimum of the last seven days transactions. For information about contacting OpenAir Customer Support, see the help topic [Creating a Support Case](#).

Upgrade With Caution

After you have tested your API integration and deployed it in production, you need to determine a process for upgrading or making changes to your OpenAir account. Before you make any changes to your OpenAir production account, you should always test these changes extensively against the API integration in your sandbox account. In particular, use care when you need to modify an object or application setting related to data or functionality that is tied to your API integration. Always test changes in your sandbox account prior to implementing them your production account.

Getting Started with OpenAir XML API and SOAP API

You can use the following steps to set up and get familiar with OpenAir XML API and SOAP API before using the API in your integration applications:

- [Step 1: Enabling OpenAir API Access](#)
- [Step 2 — Read the Relevant Documentation](#)
- [Step 3 — Register an API Integration Application](#)
- [Step 4 — Test and Familiarize Yourself with the XML API and SOAP API on a Sandbox Account](#)
- [Step 5 \(SOAP API\) — Set Up Your Development Environment and Build a Sample Client Application](#)
- [Step 6 — Connect your Application to OpenAir API](#)

Step 1: Enabling OpenAir API Access

OpenAir API Access must be enabled for your account before you can start using the OpenAir XML API or SOAP API in your integration applications. OpenAir API is a licensed add-on. To enable OpenAir API Access, contact your OpenAir account manager.

The following information is issued when OpenAir API Access is enabled:

- **API Namespace**
- **API Key**

This information is required in addition to valid authentication details (user credentials or OAuth 2.0 access token). The API namespace and API key are used to verify that the request is coming from a valid partner that has permission to use our API.

Note: OpenAir Mobile and other add-on services provided by OpenAir use OpenAir API to access your OpenAir data. You do not need OpenAir API Access to use OpenAir Mobile, OpenAir Exchange Manager, OpenAir Integration Manager, OpenAir Projects Manager or OpenAir OffLine.

Step 2 — Read the Relevant Documentation

This guide provides a reference for using OpenAir XML API and SOAP API.

- Review [OpenAir API Best Practice Guidelines](#) before you start using OpenAir XML API or SOAP API.
- Review [Getting Started with OpenAir XML API and SOAP API](#) for guidance about setting up and using OpenAir XML API and SOAP API. The section also includes information about:
 - [XML Schema and WSDL Definition Documents](#)
 - [XML API Call and Response](#)
 - [Building SOAP API Client Applications with Apache Axis.](#)
 - [Building SOAP API Client Applications with Microsoft Visual Studio IDE](#)
- The following help topics describe specific type of operations:
 - [Authentication](#)
 - [Reading Objects](#)

- [Adding, Updating and Upserting Objects](#)
- [Deleting Objects](#)
- [Approval-Related Operations](#)
- [Utility Operations](#)
- [Handling Errors](#)
- The final section gives reference information about API commands, business objects and object properties.
 - [XML API Commands](#)
 - [SOAP API Commands](#)
 - [Business Object Properties Overview](#)
 - [XML and SOAP API Business Object Reference](#)

For reference documentation about the OpenAir database, see the help topics [Database](#) and the [OpenAir Data Dictionary](#).

The business logic configured for your account may impact API requests and responses. Make sure you consult the relevant documentation, for information about OpenAir business rules. For example:

- For a description of account global and application settings and access control mechanisms, see the help topics [Administrator Guide](#) and [Security](#).
- For a description of optional features that may impact behavior, see the help topic [Optional Features](#).
- For a description of user scripting, see the help topic [User Scripting](#).

Step 3 — Register an API Integration Application

Even though OpenAir XML API and SOAP API support other authentication methods, you should use OAuth 2.0 access token authentication to access OpenAir API. This eliminates the need to store OpenAir user login details outside of OpenAir or to prompt users for credentials with every request. To use OAuth 2.0 access token authentication, an account administrator must register an application in OpenAir and enable it before you can use OpenAir API.

For more information about registering an API integration application, see [Managing API Integration Applications in OpenAir](#).

For more information about OAuth 2.0 access token authentication, see [OAuth 2.0 Authorization](#).

Step 4 — Test and Familiarize Yourself with the XML API and SOAP API on a Sandbox Account



Important: It is crucial that you test integration applications leveraging OpenAir API extensively on a Sandbox account. Make sure your integration applications run smoothly without error on a non-production account before you implement it on your production account.

You can use a GUI API client to test and familiarize yourself with OpenAir XML API and SOAP API. A GUI API client lets you:

- Execute HTTP requests from a user-friendly interface instead of using a command-line utility such as cURL.

- Save your requests and other important information and reuse them again later.
- Get a new access token using the built-in GUI API client functionality.
- Enter request information more easily and in the right format.
- See the response in a prettified XML view or a raw format.

You can use most XML API sample codes in this guide with a GUI API client. For information about how to form a XML API request and read a response, see [XML API Call and Response](#).

This guide does not currently describe the XML syntax for OpenAir SOAP web services that could be used for testing with a GUI API client.

Step 5 (SOAP API) — Set Up Your Development Environment and Build a Sample Client Application

You need to generate the OpenAir WSDL definition and import it into your development platform so that your development environment can generate the necessary objects required to build applications that consume OpenAir SOAP web services (SOAP API).

For information about generating and saving the OpenAir WSDL definition, see [XML Schema and WSDL Definition Documents](#).

The following topic provides steps for setting up Apache Axis web service framework and Microsoft Visual Studio IDE. These steps are provided for illustration purposes only. For detailed instructions about setting up these or other development platforms, refer to the vendor documentation.

- [Building SOAP API Client Applications with Apache Axis](#).
- [Building SOAP API Client Applications with Microsoft Visual Studio IDE](#)

After you have imported the OpenAir WSDL definition into your development platform, you can walk through the following sample codes to create simple client applications.

- [Java Sample Code – Authentication \(SOAP API\)](#)
- [C# Sample Code — Read \(SOAP API\)](#)

 **Note:** This guide provides examples in C# (.NET) and Java.

Development platforms vary in their SOAP implementations. Implementation differences in certain development platforms might prevent access to some or all features of the API.

Step 6 — Connect your Application to OpenAir API

 **Important:** It is crucial that you test integration applications leveraging OpenAir API extensively on a Sandbox account. Make sure your integration applications run smoothly without error on a non-production account before you implement it on your production account.

Configure your client application to connect to the relevant endpoint:

- OpenAir XML API endpoint – `https://<account-domain>/api.pl`
- OpenAir SOAP web services (SOAP API) endpoint – `https://<account-domain>/soap`

Note: The URL for OpenAir services includes the domain name for your OpenAir account <account-domain>. For more information about your account-specific domain name, see the help topic [Use Account-Specific Domain](#).

XML Schema and WSDL Definition Documents

This topic provides steps to generate and download the following reference documents:

- [XML Schema Definitions](#)
- [Generic OpenAir WSDL Definition](#)
- [Account-Specific OpenAir WSDL Definition](#)

Note: The OpenAir WSDL definition is available with the following binding styles:

- `rpc-encoded` – Remote procedure call (rpc) style, encoded use.
- `wrapped document-literal` – Document style, literal use, wrapped. One advantage of the `wrapped document-literal` binding style is that everything in the SOAP body is described by the XML schema and can therefore be validated against XML schema definitions.

The `wrapped document-literal` WSDL supports the `login()`, `read()`, `add()`, `createUser()`, `modify()`, and `delete()` methods only.

XML Schema Definitions

OpenAir provides an XML schema for each business object type supported by OpenAir XML API.

Account administrators and users with the *Export data* role permission can use the following steps to download the account-specific OpenAir WSDL.

To download the XML schema definitions:

1. Go to Administration Global Settings > Account > Integration: Import/Export.
You must be an account administrator or have the *Export data* role permission to access this page.
2. Click **XSD schema files**.
OpenAir generates a ZIP archive containing XML schema definitions for each business object type supported by OpenAir XML API. A confirmation screen appears on completion.
3. Click the **Click here** link to download the ZIP archive to your computer

Generic OpenAir WSDL Definition

To view the generic OpenAir WSDL definition in your browser, enter one of the following URLs in the address bar. The URL you use depends on the OpenAir account type and the required WSDL binding style. You can then save the page to your computer to retain a local copy of the WSDL definition.

Account type	WSDL – <code>rpc-encoded</code> binding	WSDL – <code>wrapped document-literal</code> binding
Production	https://app.openair.com/wSDL.pl	https://app.openair.com/wSDL.pl?style=document

Account type	WSDL – <code>rpc-encoded</code> binding	WSDL – <code>wrapped document-literal</code> binding
Sandbox	https://app.sandbox.openair.com/wsdl.pl	https://app.sandbox.openair.com/wsdl.pl?style=document

Account-Specific OpenAir WSDL Definition

The account-specific OpenAir WSDL definition includes custom fields defined for your account

Account administrators and users with the *Export data* role permission can use the following steps to generate the account-specific OpenAir WSDL.

To generate and save the account-specific OpenAir WSDL definition:

1. Go to Administration > Global Settings > Account > Integration: Import/Export.
2. Click **Account specific WSDL**.
The `rpc-encoded` account-specific WSDL appears in a new browser tab.
3. (Optional) To generate the `wrapped document-literal` account-specific WSDL, append a semi-colon (;) and `style=document` to the URL in the address bar of your browser and press Enter.
4. Save the page to your computer to retain a local copy of the WSDL definition.

XML API Call and Response

Client applications must make an HTTP call to send a XML API request that includes a series of commands to the OpenAir XML API endpoint. OpenAir XML API processes the commands in the XML API request and sends a response back to the client application.

This topic describes the [HTTP Call Syntax](#), [XML API Request Syntax](#), [XML Data Object Syntax](#) and [XML API Response](#). You can also walk through an example of XML API request and response – see [XML API Request and Response Sample Codes](#).

Note: This guide provides formatted XML syntax and code samples with line returns and indentation throughout. Client applications typically send and receive XML content in its raw form (minified).

HTTP Call Syntax

```

1 PUT /api.pl HTTP 1.1
2 Host: company-id.app.openair.com
3 Content-Type: application/xml
4
5 <!-- XML API request -->
```

- **HTTP method** – Use either PUT or POST. Many libraries support either or both of these methods. From OpenAir XML API point of view, the PUT and POST are almost identical.
- **OpenAir XML API endpoint** – `https://<account-domain>/api.pl`
- **HTTP call headers:**
 - **Host** – The account-specific domain name for your OpenAir account is typically sent in the `Host` header. For more information about your account-specific domain name, see the help topic [Use Account-Specific Domain](#).
 - **Content-Type** – The XML API request in the body of the call uses XML syntax (`application/xml`).

- **HTTP Call Body** – The XML API request. See [XML API Request Syntax](#).

XML API Request Syntax

All requests sent to the OpenAir XML API endpoint must use the following syntax.

```

1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <request API_version="1.0" client="example client" client_ver="1.1" namespace="example" key="0123456789">
3   <Auth>
4     <Login>
5       <access_token>0123456789-ABCDEFGHIJKLMNIOQRSTUVWXYZ0123456789ABCDEF-ABCDEFGHIJKLMNIOQRSTUVWXYZ01234567</I
access_token>
6     </Login>
7   </Auth>
8
9   <!-- XML API commands -->
10
11 </request>

```

Each request includes a XML prolog followed by a request element that contains the XML API commands.

- **XML prolog** – OpenAir uses UTF-8 encoding to store and display characters. Specify the encoding in the XML prolog to ensure that non-Latin characters are handled and stored correctly.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

- **Root element** – The request element is the root element and the immediate parent to all command elements in the XML API request.
 - The request element has the following attributes:

Attribute	Description
API_version	The API_version attribute value is always 1.0
client	The name of your client application
client_ver	The version number of your client application
namespace	The API namespace assigned to your OpenAir account. See Step 1: Enabling OpenAir API Access .
key	The API key assigned to your OpenAir account. See Step 1: Enabling OpenAir API Access .

- All command elements are siblings and direct descendants of the request element.
 - The first command in the request must be the [Auth](#) command. Successful authentication is required for all other commands except [Time](#). See also [Authentication](#).
 - Subsequent commands can be used to interact with your OpenAir data. Depending on the operation, each command element may have a child business data object element. See the following topics for information about the type of information you can perform, and for information about the syntax and usage of each command.
 - [Reading Objects](#)
 - [Adding, Updating and Upserting Objects](#)
 - [Deleting Objects](#)
 - [Approval-Related Operations](#)
 - [Utility Operations](#)
 - [XML API Commands](#)



Important: The XML syntax is case sensitive. Element tags for XML API commands and objects always start with an uppercase character. Element tags for object properties always start with a lowercase character.

XML Data Object Syntax

An object element is defined by an XML start and end tag pair specifying the object type, and all the children property elements within this tags.

Each property element is defined by an XML start and end tag pair specifying the property name, and the property value within this tags.

```

1      <ObjectType>
2          <property_name_1>property_value_1</property_name_1>
3          <property_name_2>property_value_2</property_name_2>
4          <property_name_3>property_value_3</property_name_3>
5
6          <!-- Additional property elements -->
7
8      </ObjectType>

```

Elements for properties referencing related object(s) by internal ID may include the attribute external or name. You can use this to look up the internal ID of a related object using its external ID or name as foreign key. See [Related Object Lookup Using the XML API](#).

Property elements with empty values can be represented using an XML start and end tag pair with no content (such as the property_name_1 element in the following example), or with a self-closing XML tag, if you want to set the property to an empty value (such as the property_name_2 element in the following example).

```

1      <property_name_1></property_name_1>
2      <property_name_2/>

```

Property elements can be omitted if you do not want to set a value.

Property values can either be text or an XML substructure. Properties containing address information, date or time information, and account or user settings use have an XML substructure as property value. These XML substructures use the same syntax as for XML data objects. The following example shows the XML syntax for a property including address information:

```

1      <addr>
2          <Address>
3              <first>John</first>
4              <last>Smith</last>
5              <phone>123-345-6789</phone>
6          </Address>
7      </addr>

```

For more information about supported business object types and their properties, see [XML and SOAP API Business Object Reference](#).

XML API Response

The XML API response uses a syntax similar to the request sent to the OpenAir XML API endpoint.

```

1      <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2      <response>
3          <Auth status = "0"></Auth >
4
5          <!-- XML API command responses -->

```

```
6 |
7 | </response>
```

Each response includes a XML prolog followed by a response element that contains the responses to each XML API command sent in the request.

- **XML prolog** – OpenAir uses UTF-8 encoding to store and display characters.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

- **Root element** – The response element is the root element and the immediate parent to all command response elements in the XML API response.
 - The response element has no attributes unless the XML request is malformed, in which case it has a status attribute set to 1.

Attribute	Description
status	Only included and has the value 1 if the XML request is malformed.

- All command response elements are siblings and direct descendants of the response element.
 - Command response elements in the response follow the same sequence as the command elements in the request.
 - The names of command response elements in the response are the same as the names of the corresponding command elements in the request.

The first child element is an Auth element returned in response to the [Auth](#) command in the request.

Subsequent elements show the outcome of each subsequent commands in the request. Depending on the operation, each of these element may have children business data object elements. For example, if the request included a [Read](#) command element and a [CreateUser](#) command element, the response includes a Read element and a CreateUser element in the same order. The Read element has up to 1,000 children (objects of the requested type), the CreateUser has one child ([User](#) object).
 - Each command response element includes a status attribute which indicates the success or failure of the operation. For more information about the command response status, see [Error Codes](#).

XML API Request and Response Sample Codes

The following XML API request sample code includes the following operations:

1. Auth element – Authentication using OAuth 2.0 access token.
2. Time element – Retrieves the time from the OpenAir API server.
3. Read element – Retrieves up to 100 invoices from OpenAir starting with index 0.
4. CreateUser element – Add the user John Smith to the account. In this example, the value specified for the Company ID includes a typographic error.

```
1 | <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 | <request API_ver="1.0" client="sample code" client_ver="1.1" namespace="example" key="0123456789">
3 |   <Auth>
4 |     <Login>
5 |       <access_token>0123456789-ABCDEFGHIJKLMNIOQRSTUVWXYZ0123456789ABCDEF-ABCDEFGHIJKLMNIOQRSTUVWXYZ01234567</
6 |     </Login>
```

```

7   </Auth>
8   <Time/>
9   <Read type="Invoice" limit="0,100" method="all"/>
10  <CreateUser>
11    <Company>
12      <nickname>company_id_with_typo_error</nickname>
13    </Company>
14    <User>
15      <name>jsmith</name>
16      <password>new_user_password</password>
17      <taapprover>16</taapprover>
18      <teapprover>16</teapprover>
19      <addr>
20        <Address>
21          <first>John</first>
22          <last>Smith</last>
23          <phone>123-345-6789</phone>
24        </Address>
25      </addr>
26    </User>
27  </CreateUser>
28 </request>

```

The XML API response returns:

1. Auth element – A successful authentication status.
2. Time element – The time on the OpenAir API server as a [Date](#) object.
3. Read element – Up to 100 [Invoice](#) objects. There are only 2 invoice records in the example OpenAir account.
4. CreateUser element – An error status with error code 201. A subsequent request can be sent to retrieve the error description corresponding to error code 201, if required. See [Error Codes](#).

```

1 <response>
2 <Auth status="0"></Auth>
3 <Time status="0">
4   <Date>
5     <day>13</day>
6     <month>02</month>
7     <year>2024</year>
8     <hour>23</hour>
9     <minute>59</minute>
10    <second>01</second>
11  </Date>
12 </Time>
13 <Read status="0">
14   <Invoice>
15     <id>1</id>
16     <number>234</number>
17     <customerid>204</customerid>
18     <total>99.00</total>
19     <tax>0.00</tax>
20     <balance>80.00<balance>
21     <draw>1</draw>
22     <credit/>
23     <credit_reason/>
24     <terms/>
25     <emailed>
26       <Date>
27         <day>13</day>
28         <month>2</month>
29         <year>2024</year>
30         <hour>0</hour>
31         <minute>0</minute>
32         <second>0</second>
33       </Date>
34     </emailed>
35   </Invoice>
36   <Invoice>
37     <id>4</id>

```

```

38     <number>983</number>
39     <customerid>204</customerid>
40     <total>12.00</total>
41     <tax>0.00</tax>
42     <balance>50.00<balance>
43     <draw>1</draw>
44     <credit/>
45     <credit_reason/>
46     <terms/>
47     <emailed/>
48   </Invoice>
49 </Read>
50 <CreateUser status="201"></CreateUser>
51 </response>

```

Building SOAP API Client Applications with Apache Axis

This section provides steps to build a SOAP web services application using Java with the Apache Axis framework (versions 1.3 and 1.4).

You must generate the Java client binding objects required to consume OpenAir SOAP web services (SOAP API) from the OpenAir WSDL before you can build client applications in a Java environment. These objects serve as proxies for their server-side counterparts.

After you have imported the OpenAir WSDL definition into your development platform, you can walk through a following sample code to create a simple client application. See [Java Sample Code – Authentication \(SOAP API\)](#).

Note: These steps are provided for illustration purposes only. For detailed instructions about setting up Apache Axis or other development platforms, refer to the Apache Axis documentation.

OpenAir SOAP web Services are compatible with Apache Axis 1.3 and 1.4. It is not compatible with the Apache Axis2 libraries.

To use the Apache Axis framework with OpenAir SOAP web services:

1. Install Java JDK 8.
Download and install Java JDK 8. Ensure that the executables are available through the system path. This is required because all inbound and outbound secure communication must use TLS 1.2 or TLS 1.3.
2. Download and install Apache Axis from <http://ws.apache.org/axis/>. The following steps are provided for Axis version 1.4 installed in C:\axis-1_4.
3. Set up Apache Axis to consume OpenAir SOAP web services.
 - a. Go to the directory where you want to generate the Java source code files and folders.
 - b. Run the following command to:
 - Ensure that your CLASSPATH includes the required elements from the Axis binary distribution.
 - Use the WSDL2Java utility to generate the proxy classes from the OpenAir WSDL definition.

```

1 java -cp c:\axis-1_4\lib\axis.jar;
2 c:\axis-1_4\lib\axis-ant.jar;
3 c:\axis-1_4\lib\commons-logging-1.0.4.jar;
4 c:\axis-1_4\lib\commons-discovery-0.2.jar;
5 c:\axis-1_4\lib\jaxrpc.jar;
6 c:\axis-1_4\lib\log4j-1.2.8.jar;

```

```

7 | c:\axis-1_4\lib\wsdl4j-1.5.1.jar;
8 | org.apache.axis.wsdl.WSDL2Java -a <wsdl-file-url>

```

Where <wsdl-file-url> is the URL for the OpenAir WSDL. For example, if you are using the generic WSDL for production accounts, the URL is <https://app.openair.com/wsdl.pl>.

Note: Use the WSDL2Java utility -a option to generate code for all elements, even unreferenced ones. By default, WSDL2Java only generates code for those elements in the WSDL file that are referenced.

(Optional) Use the -p option to specify a package namespace instead of the default com.soaplite.namespaces.per1 namespace.

For more information about the WSDL2Java utility, see [WSDL2Java: Building stubs, skeletons, and data types from WSDL \(External link to Apache Axis website\)](#) and [WSDL2Java Reference \(External link to Apache Axis website\)](#).

- c. Compile the generated source code.

Note: You can use Apache Ant or wizard-based tools instead of the command line to build the stub objects and proxy classes in most Java development environments.

4. Implement workaround to handle SAXExceptions caused by WSDL updates.

Important: New objects and object properties may be added to the OpenAir WSDL definition from time to time. These changes may not be automatically reflected in stubs generated using the WSDL2Java utility and may cause exceptions in your client application code.

- a. Create a subclass of org.apache.axis.encoding.ser.BeanDeserializer and override the onStartChild method to handle any SAXExceptions that may occur.

```

1 | public class MyDeserializer extends BeanDeserializer {
2 |     public MyDeserializer(java.lang.Class javaType, javax.xml.namespace.QName xmlType, org.apache.axis.description.TypeDesc typeDesc)
3 |     {
4 |         super(javaType, xmlType, typeDesc);
5 |     }
6 |     public SOAPHandler onStartChild(String arg0, String arg1, String arg2, Attributes arg3, DeserializationContext arg4) throws SAXException
7 |     {
8 |         // TODO Auto-generated method stub
9 |         try
10 |         {
11 |             __return super.onStartChild(arg0, arg1, arg2, arg3, arg4);
12 |         } catch (SAXException e) {
13 |             __return null;
14 |         }
15 |     }
16 | }

```

- b. For each generated client class (oaEnvelope, oaTimesheet, etc.), change the getDeserializer method to use the new subclass of BeanDeserializer instead of the default implementation.

```

1 | /**
2 |  * Get Custom Deserializer
3 |  */

```

```

4 public static org.apache.axis.encoding.Deserializer getDeserializer(
5     java.lang.String mechType,
6     java.lang.Class _javaType,
7     javax.xml.namespace.QName _xmlType)
8     {
9         return new MyDeserializer( _javaType, _xmlTpe, typeDesc);
10    }

```

5. Implement your application by writing your business logic using the generated Axis proxy classes.

a. Locate the OpenAir service.

```
1 | OAIRServiceHandlerServiceLocator locator = new OAIRServiceHandlerServiceLocator();
```

b. Get an instance of the OpenAir service.

```
1 | m_svc = (OAIRServiceSoapBindingStub)locator.getOAIRService();
```

c. Authenticate by populating the LoginParams object and then invoking the login() operation.

```

1 | LoginParams lp = new LoginParams();
2 | lp.setApi_namespace("<Your_API_Namespace>");
3 | lp.setApi_key("<Your_API_Key>");
4 | lp.setCompany("<Your_Company_ID>");
5 | lp.setUser("<Your_User_ID>");
6 | lp.setPassword("<Your_Password>");
7 | LoginResult loginResult = m_svc.login(lp);

```

Replace *<Your_API_Namespace>*, *<Your_API_Key>*, *<Your_Company_ID>*, *<Your_User_ID>*, and *<Your_Password>* with your API namespace, API key, and OpenAir company ID, user ID and password.

The login() command creates a session and returns a session ID which you can use in the SOAP header to provide authentication for API calls implementing your business logic.

d. Set up the SOAP header to include the session ID.

```

1 | SOAPHeaderElement header = new SOAPHeaderElement("https://company-id.app.openair.com/OAIRService", "SessionHeader");
2 | SOAPElement node = header.addChildElement("sessionId");
3 | node.addTextNode(loginResult.getSessionId());
4 | m_svc.setHeader(header);

```

e. Implement your business logic. For example, create a new customer in OpenAir.

```

1 | oaCustomer customer = new oaCustomer();
2 | customer.setName("XYZ Inc");
3 | try
4 | {
5 |     // Add the customer and output any errors encountered
6 |     UpdateResult result = m_svc.add(customer);
7 |     if (result.getErrors() != null)
8 |     {
9 |         for (oaBase base : result.getErrors())
10 |         {
11 |             oaError err = (oaError)base;
12 |             System.out.println("Error: " + err.getCode() + "\t" + err.getComment() + "\t" + err.getText());
13 |         }
14 |     }
15 |     if (result.getStatus() == "A") System.out.println("Customer added");
16 | }

```

f. Log out to invalidate the current session.

```
1 | m_svc.logout();
```

Java Sample Code – Authentication (SOAP API)

This section walks through a sample Java client application. Web service client access helper classes and stubs for this example were generated using the Apache Axis WSDL2Java tool. For more information on this tool, see [Getting Started with OpenAir XML API and SOAP API](#). The example demonstrates the following functions:

1. Log in to OpenAir web services with user credentials entered by the user at the console prompt.
2. Add several user records to the OpenAir account using the information entered by the user at the console prompt.
3. Log out of OpenAir web services.

```

1 import java.rmi.RemoteException;
2 import javax.xml.soap.SOAPElement;
3 import org.apache.axis.message.SOAPHeaderElement;
4 import java.io.*;
5
6 class Program
7 {
8     // Instance of OpenAir web services proxy object
9     private static OAIRServiceSoapBindingStub m_svc;
10
11     // Company ID for the OpenAir account new users will be added to
12     private static String m_strCompany;
13
14     // Console prompt for user credentials
15     private static String GetUserInput(String prompt)
16     {
17         {
18             BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
19             return reader.readLine();
20         }
21         catch (java.io.IOException e)
22         {
23             return null;
24         }
25     }
26
27     // Log in to OpenAir web services using with user credentials.
28     // Returns true if successful, false if not
29     private static boolean Login() throws javax.xml.soap.SOAPException, javax.xml.rpc.ServiceException
30     {
31         // Set up login information
32         LoginParams lp = new LoginParams();
33         lp.setApi_key("*****");
34         lp.setApi_namespace("company_namespace");
35         lp.setUser( GetUserInput("Enter username: ") );
36         lp.setPassword( GetUserInput("Enter password: ") );
37         lp.setCompany( GetUserInput("Enter company: ") );
38         m_strCompany = lp.getCompany();
39
40         try
41         {
42             // Get an instance of OpenAir web services and login
43             OAIRServiceHandlerServiceLocator locator = new OAIRServiceHandlerServiceLocator();
44             m_svc = (OAIRServiceSoapBindingStub)locator.getOAIRService();
45
46             LoginResult loginResult = m_svc.login(lp);
47             System.out.println("Logged in, session ID = " + loginResult.getSessionId()+"\n");
48
49             // Set up session header to include returned session ID to perform further operations
50             SOAPHeaderElement header = new SOAPHeaderElement("https://my-account-domain.app.openair.com/OAIRService","Session
Header");
51             SOAPElement node = header.addChildElement("sessionId");
52             node.addTextNode(loginResult.getSessionId());
53             m_svc.setHeader(header);
54         }
55         catch (java.rmi.RemoteException e)
56         {

```

```

57         // Catch any login problems and return
58         System.out.println(e.toString());
59         return false;
60     }
61     return true;
62 }
63 // Prompt for information about user to be added and Add a new user record using the information supplied
64 private static void CreateUser()
65 {
66     System.out.println("-----");
67     System.out.println("Enter new user information\n");
68
69     // Create the company object that the new user will be associated with
70     oaCompany company = new oaCompany();
71     company.setNickname(m_strCompany);
72
73     // Get the new user information
74     oaUser user = new oaUser();
75     user.setNickname( GetUserInput("Enter username: ") );
76     user.setRole_id( GetUserInput("Enter role ID: ") );
77     user.setAddr_first( GetUserInput("Enter first name: ") );
78     user.setAddr_last( GetUserInput("Enter last name: ") );
79     user.setAddr_email( GetUserInput("Enter email: ") );
80     user.setPassword( GetUserInput("Enter password: ") );
81     try
82     {
83         // Add the user and output any errors encountered
84         UpdateResult result = m_svc.createUser(user, company);
85         if (result.getErrors() != null)
86         {
87             for (oaBase base : result.getErrors())
88             {
89                 oaError err = (oaError)base;
90                 System.out.println("Error: " + err.getCode() + "\t" + err.getComment() + "\t" + err.getText());
91             }
92         }
93         if (result.getStatus() == "A")
94             System.out.println("User successfully added");
95     }
96     catch (Exception e)
97     {
98         System.out.println("Error while adding user:\n"+e.toString());
99     }
100 }
101 // Application entry point
102 public static void main(String[] args)
103 {
104     try
105     {
106         // Log in to OpenAir web services and add users
107         if (Login())
108         {
109             {
110                 do
111                 {
112                     CreateUser();
113                 } while (GetUserInput("\nAdd another (y/n)?
114 ") .toLowerCase().startsWith("y"));
115             }
116             m_svc.logout();
117         }
118     }
119     catch (Exception e)
120     {
121         System.out.println(e.toString());
122     }
123     System.out.println("\nDone");
124 }
125 }

```

Building SOAP API Client Applications with Microsoft Visual Studio IDE

This section provides steps to build a SOAP web services application using Microsoft Visual Studio IDE.

You must generate the classes required to consume OpenAir SOAP web services (SOAP API) from the OpenAir WSDL before you can build client applications using Visual Studio languages. These classes serve as proxies for their server-side counterparts.

After you have imported the OpenAir WSDL definition into your development platform, you can walk through the following sample code to create a simple client application. See [C# Sample Code — Read \(SOAP API\)](#).

Note: These steps are provided for illustration purposes only. For detailed instructions about setting up, and building applications using Visual Studio IDE or other development platforms, refer to the Visual Studio documentation.

To use Microsoft Visual Studio IDE with OpenAir SOAP web services:

1. In Microsoft Visual Studio, create a new project and give it a name, for example SoapClientApplication.
2. Go to Project > Add Service Reference > Advanced > Add Web Reference.
The Add Web Reference window appears.
3. In the **URL** box, enter the URL for the OpenAir WSDL. For example, if you are using the generic WSDL for production accounts, the URL is <https://app.openair.com/wsd1.pl>.
4. Click **Go** to retrieve information about OpenAir SOAP Web Services.
5. In the **Web reference name** box, rename the web reference as OpenAir, for example.
6. Click **Add Reference**.

Visual Studio retrieves the service description and generates the proxy classes to interface between your application and OpenAir SOAP web services.

7. Implement your application by writing your business logic using the generated proxy classes.
 - a. Add a **using** directive to the OpenAir SOAP web service reference. Replace SoapClientApplication.OpenAir with the name of your project and the web reference name for OpenAir SOAP web services MyProjectName.MyReferenceName.

```

1 using System;
2 using SoapClientApplication.OpenAir;
3
4 namespace SoapClientApplication

```

- b. Get an instance of the OpenAir SOAP web services proxy class.

```

1 {
2     class GettingStartedWithOaSoapApi
3     {
4         private static OAIServiceHandlerService m_svc;
5         m_svc = new OAIServiceHandlerService();

```

- c. Authenticate by populating the [LoginParams](#) object and then invoking the [login\(\)](#) operation.

```

1     private static bool Login()
2     {
3         OpenAir.LoginParams lp = new OpenAir.LoginParams();
4         lp.api_namespace = "<Your_API_Namespace>";

```

```

5 | lp.api_key = "<Your_API_Key>";
6 | lp.company = "<Your_Company_ID>";
7 | lp.user = "<Your_User_ID>";
8 | lp.password = "<Your_Password>";
9 |
10 | LoginResult loginResult;
11 | loginResult = m_svc.login(lp);

```

Replace `<Your_API_Namespace>`, `<Your_API_Key>`, `<Your_Company_ID>`, `<Your_User_ID>`, and `<Your_Password>` with your API namespace, API key, and OpenAir company ID, user ID and password.

The `login()` command creates a session and returns a session ID which you can use in the SOAP header to provide authentication for API calls implementing your business logic.

- d. Set up the SOAP header to include the session ID.

```

1 | SessionHeader header = new SessionHeader();
2 | header.sessionId = loginResult.sessionId;
3 | m_svc.SessionHeaderValue = header;
4 | }

```

- e. Implement your business logic. For example, create a new customer in OpenAir.

```

1 | private static void AddCustomer()
2 | {
3 |     oaCustomer customer = new oaCustomer();
4 |     customer.name = "XYZ Inc";
5 |     try
6 |     {
7 |         // Add the customer and output any errors encountered
8 |         UpdateResult result = m_svc.add(customer);
9 |         if (result.errors != null)
10 |         {
11 |             foreach (oaError err in result.errors)
12 |             {
13 |                 oaError err = (oaError)base;
14 |                 Console.WriteLine("Error "+err.code + ": " + err.comment + "\t" + err.text);
15 |             }
16 |         }
17 |         if (result.status == "A") Console.WriteLine("Customer added");
18 |     }
19 | }

```

- f. Add an application entry point, perform operations if the authentication is successful and log out to invalidate the current session.

```

1 | static void Main(string[] args)
2 | {
3 |     if (Login())
4 |     {
5 |         AddCustomer();
6 |         // end the session
7 |         m_svc.logout();
8 |     }
9 | }
10 | }
11 | }

```

C# Sample Code — Read (SOAP API)

This section walks through a sample C# application. Web service client access helper classes and stubs were generated using Microsoft Visual Studio 2005. For more information about generating these classes, see [Getting Started with OpenAir XML API and SOAP API](#). This example demonstrates the following functions:

1. Log in to OpenAir web services with user credentials entered by the user at the console prompt.
2. Retrieve all expense reports (`Envelope` objects) in the OpenAir account. This operation uses the `read()` command and `all` method.
3. Retrieve the expense report envelope (`Envelope` object) with the internal ID entered by the user at the console prompt. This operation uses the `read()` command and `equal` to method.
4. Log out of OpenAir web services.

```

1  using System;
2  using SoapApplication.OpenAir;
3
4  namespace SoapApplication
5  {
6      class Program
7      {
8          /// <summary>
9          /// Instance of OpenAir web services proxy object
10         /// </summary>
11         private static OAIServiceHandlerService m_svc;
12
13         /// <summary>
14         /// Console prompt for user credentials
15         /// </summary>
16         private static string GetUserInput(string prompt)
17         {
18             Console.Write(prompt);
19             try
20             {
21                 return Console.ReadLine();
22             }
23             catch (System.IO.IOException e)
24             {
25                 return null;
26             }
27         }
28
29         /// <summary>
30         /// Log in to OpenAir web services using with user credentials.
31         /// </summary>
32         /// <returns>True if successful, false if not</returns>
33         private static bool Login()
34         {
35             // Set up login information
36             OpenAir.LoginParams lp = new OpenAir.LoginParams();
37             lp.api_key = "*****";
38             lp.api_namespace = "company_namespace";
39             lp.user = GetUserInput("Enter username: ");
40             lp.password = GetUserInput("Enter password: ");
41             lp.company = GetUserInput("Enter company: ");
42
43             // Get an instance of OpenAir web services and login
44             m_svc = new OAIServiceHandlerService();
45             LoginResult loginResult;
46             try
47             {
48                 loginResult = m_svc.login(lp);
49                 Console.WriteLine("Logged in, session ID = " + loginResult.sessionId);
50
51                 // Set up session header to include returned session ID to perform further operations
52                 SessionHeader header = new SessionHeader();
53                 header.sessionId = loginResult.sessionId;
54                 m_svc.SessionHeaderValue = header;
55             }
56             catch (System.Web.Services.Protocols.SoapException e)
57             {
58                 // Catch any login problems and return
59                 Console.WriteLine(e.Message);
60                 return false;
61             }
62             return true;
63         }
64     }

```

```

64
65     /// <summary>
66     /// Output the specified Envelope object.
67     /// </summary>
68     private static void PrintEnvelope(oaEnvelope env)
69     {
70         Console.WriteLine("-----");
71         Console.WriteLine("ID:\t" + env.id);
72         Console.WriteLine("Name:\t" + env.name);
73         Console.WriteLine("Date:\t" + env.date);
74         Console.WriteLine("Total:\t" + env.total);
75         Console.WriteLine("# receipts:\t" + env.tottickets);
76         // Add any other envelope properties required
77     }
78
79     /// <summary>
80     /// Read all envelopes from OpenAir web services
81     /// </summary>
82     static void ReadAllEnvelopes()
83     {
84         // Do the read operation
85         Console.WriteLine("\nPerforming read of ALL envelopes\n");
86         ReadRequest req = new ReadRequest();
87         req.type = "Envelope";
88         req.method = "all";
89
90         try
91         {
92             ReadResult[] results = m_svc.read(new ReadRequest[] { req });
93             // iterate through our results and output them to console
94             foreach (ReadResult result in results)
95             {
96                 // Output any errors
97                 if (result.errors != null)
98                 {
99                     foreach (oaError err in result.errors)
100                     {
101                         Console.WriteLine("Error "+err.code + ": " + err.comment + "\t" + err.text);
102                     }
103                 }
104
105                 // Output the envelope read results
106                 if (result.objects != null)
107                 {
108                     Console.WriteLine("Received "+result.objects.Length+" envelope(s) from OpenAir");
109                     foreach (oaEnvelope env in result.objects)
110                     {
111                         PrintEnvelope(env);
112                     }
113                 }
114             }
115         }
116         catch (Exception e)
117         {
118             Console.WriteLine("Error while reading envelopes:\n" + e);
119         }
120     }
121
122     /// <summary>
123     /// Read the envelope with the internal ID entered by the user at the console prompt
124     /// </summary>
125     private static void ReadSingleEnvelope()
126     {
127         Console.WriteLine("\n\nPerforming read using \"equal to\" method");
128         oaEnvelope envelope = new oaEnvelope();
129         envelope.id = GetUserInput("Enter an envelope id: ");
130         ReadRequest req = new ReadRequest();
131         req.objects = new oaBase[] { envelope };
132         req.type = "Envelope";
133         req.method = "equal to";
134         try
135         {
136             ReadResult[] results = m_svc.read(new ReadRequest[] { req });

```

```

137
138 // Iterate through read results and output them in the console
139 foreach (ReadResult result in results)
140 {
141     // output any errors
142     if (result.errors != null)
143     {
144         foreach (oaError err in result.errors)
145         {
146             Console.WriteLine("Error " + err.code + ": " + err.comment + "\t" + err.text);
147         }
148     }
149
150     // Output the envelope read results
151     if (result.objects != null)
152     {
153         Console.WriteLine("Received " + result.objects.Length + " envelope(s) from OpenAir");
154         foreach (oaEnvelope env in result.objects)
155         {
156             PrintEnvelope(env);
157         }
158     }
159 }
160
161 catch (Exception e)
162 {
163     Console.WriteLine("Error while reading envelopes:\n" + e);
164 }
165
166
167 /// <summary>
168 /// Application entry point
169 /// </summary>
170 static void Main(string[] args)
171 {
172     if (Login())
173     {
174         ReadAllEnvelopes();
175         ReadSingleEnvelope();
176
177         // end our session
178         m_svc.logout();
179     }
180     Console.WriteLine("\nPress enter to exit");
181     Console.ReadLine();
182 }
183 }
184 }

```

Authentication

OpenAir XML API and SOAP API support the following authentication methods:

- OAuth 2.0 access token – You can use the [Auth](#) (XML API) or use a session header [[SessionHeader](#)] (SOAP API) to send the OAuth 2.0 access token [access_token] with each request. See [OAuth 2.0 Access Token Authentication](#).

Note: You should use authentication by OAuth 2.0 bearer token instead of password or client session ID where possible. OAuth 2.0 access token authentication is a more secure and reliable way to access data than other supported authentication methods. Integration applications must be registered in OpenAir to use OpenAir REST API to access OpenAir data, and users must give the application explicit permission to access OpenAir on their behalf.

For more information about OAuth 2.0, see [OAuth 2.0 for Integration Applications Developers](#).

- Password – You can use the [Auth](#) (XML API) or [login\(\)](#) (SOAP API) command and pass user credentials (Company ID, User ID and Password). The [login\(\)](#) (SOAP API) command starts a client session and returns a unique client session identifier that can be used to make subsequent calls. See [Password Authentication](#).
- Client session ID – (**SOAP API only**) The [login\(\)](#) (SOAP API) command starts a client session and returns a unique client session identifier [sessionId]. You can use a session header [[SessionHeader](#)] (SOAP API) to send this client session ID for subsequent calls to OpenAir SOAP API until the session expires or is ended for the authenticated user by a [logout\(\)](#) call.

Sessions expire automatically after a predetermined length of inactivity, which can be configured in the OpenAir UI. See the help topic [Session Timeout](#).

The [login\(\)](#) (SOAP API) command also returns a URL for the active OpenAir UI session for the authenticated user. With the XML API, you can use the [RemoteAuth](#) command to obtain a URL for the active OpenAir UI session for the authenticated user.

SessionHeader

The SessionHeader holds the client session information.

A SessionHeader object has the following properties:

Name	Type	Description
sessionId	string	Valid client session ID for the authenticated user. Used with session based password authentication.
accessToken	string	Valid access token. Used with OAuth 2.0 access token authentication.

- For examples using the SessionHeader complex type to hold the sessionId for **session based password authentication**, see the [login\(\)](#), [makeURL\(\)](#) and [logout\(\)](#) methods, as well as the following sample codes: [Java Sample Code – Authentication \(SOAP API\)](#) and [C# Sample Code — Read \(SOAP API\)](#).
- For examples using the SessionHeader complex type to hold the accessToken for **OAuth 2.0 token based authentication**, see [Using SessionHeader for OAuth2.0 Token Based Authentication](#).



Important: An invalid OAuth2 access token authorization has priority over a valid session based password authentication. You cannot use a valid Session ID as a fallback for an invalid access token. See [Using OAuth 2.0 Access Tokens in Your API Requests](#).

Using SessionHeader for Client Session ID Authentication

When using client session ID based authentication, the SessionHeader web services method complex type is used to hold the client session ID [sessionId].

Use the syntax given in the following examples:

Sample Code — XML

```
1 <SessionHeader xsi:type="perl:SessionHeader" xmlns:perl="http://namespaces.soaplite.com/perl">
2   <sessionId xsi:type="xsd:string">ABCDEFGHIJKlmnopqrstuv</sessionId>
3 </SessionHeader>
```

Sample Code — C#

```
1 // Create service stub
2 OAIRServiceHandlerService _svc = new OAIRServiceHandlerService();
3
4 // create LoginParam object
5 LoginParams loginParams = new LoginParams();
6 loginParams.api_namespace = "my namespace";
7 loginParams.api_key = "*****";
8 loginParams.company = "company name";
9 loginParams.user = "username";
10 loginParams.password = "password";
11 loginParams.client = "my client name";
12 loginParams.version = "1.0";
13 LoginResult loginResult = _svc.login(loginParams);
14
15 // Create a new session header object
16 // Add the session ID returned from the login
17 _svc.SessionHeaderValue = new SessionHeader();
18 _svc.SessionHeaderValue.sessionId = loginResult.sessionId;
```

Sample Code — Java

```
1 // create our login parameters
2 LoginParams lp = new LoginParams();
3 lp.setUser("username");
4 lp.setPassword("password");
5 lp.setCompany("company name");
6 lp.setApi_namespace("my namespace");
7 lp.setApi_key("*****");
8 lp.setClient("my client name");
9 lp.setVersion("1.0");
10
11 // set the service URL from our arguments
12 OAIRServiceHandlerServiceLocator locator = new OAIRServiceHandlerServiceLocator();
13 locator.setOAIRServiceAddress("https://my-account-domain.app.openair.com/soap");
14
15 // now login
16 OAIRServiceSoapBindingStub binding =
17 (OAIRServiceSoapBindingStub)locator.getOAIRService();
18 LoginResult loginResult = binding.login(lp);
19
20 // Create a new session header object
21 // Add the session ID returned from the login
22 SOAPHeaderElement header = new SOAPHeaderElement("https://my-account-domain.app.openair.com/OAIRService", "SessionHeader");
23 SOAPElement node = header.addChildElement("sessionId");
24 node.addTextNode(loginResult.getSessionId());
```

```
25 | binding.setHeader(header);
```

Using SessionHeader for OAuth2.0 Token Based Authentication

When using OAuth 2.0 token based authentication, the SessionHeader web services method complex type is used to hold the OAuth 2.0 access token (accessToken) instead of the logged in user Session ID (sessionId).

Use the syntax given in the following examples:

Sample Code — XML

```
1 | <SessionHeader xsi:type="perl:SessionHeader" xmlns:perl="http://namespaces.soaplite.com/perl">
2 |   <accessToken xsi:type="xsd:string">eNNJ1GX025-6IUy1F6RZT33HqhoqSAAK53F0kxT62fBoKreDoc8Y_-Gnk21UIqNbhWguHxDtxUsJMY6NirDoiBnd</
   |   accessToken>
3 | </SessionHeader>
```

Sample Code — C#

```
1 | // Create service stub
2 | OAIRServiceHandlerService _svc = new OAIRServiceHandlerService();
3 |
4 | // POST Request to get access_token
5 |
6 | // Create a new session header object and add the access token
7 | _svc.SessionHeaderValue = new SessionHeader();
8 | _svc.SessionHeaderValue.accessToken = response.access_token;
```

Sample Code — Java

```
1 | // Set the service URL
2 | OAIRServiceHandlerServiceLocator locator = new OAIRServiceHandlerServiceLocator();
3 | locator.setOAIRServiceAddress("https://company-id.app.openair.com/soap");
4 |
5 | // POST Request to get access_token
6 |
7 | // Create a new session header object and add the access token
8 | SOAPHeaderElement header = new SOAPHeaderElement("https://company-id.app.openair.com/OAIRService", "SessionHeader");
9 | SOAPElement node = header.addChildElement("accessToken");
10 | node.addTextNode(access_token);
11 | binding.setHeader(header);
```

For more information about OAuth 2.0, see [OAuth 2.0 for Integration Applications Developers](#).

OAuth 2.0 Authorization

OpenAir supports OAuth 2.0, a robust authorization framework. This authorization framework enables client applications to use a token to access OpenAir through the OpenAir XML, SOAP, or REST API. The application accesses the protected resources on behalf of a user who gave an explicit permission for the access. This method eliminates the need for API integrations to store user credentials.

This feature is available if OpenAir API access is enabled for your account. It includes the following elements:

- **Administrators** can register up to 20 integration applications with OpenAir and enable or disable these applications in the Administration module. For more information, see [Managing API Integration Applications in OpenAir](#).
- **Administrators** can use web services reports to audit and revoke authorizations granted by OpenAir users to integration applications. For more information, see [Auditing and Managing OAuth 2.0 Authorizations](#).
- **Application Developers** can use the OAuth 2.0 authorization code flow to get an access token then use the access token to access your OpenAir data using the OpenAir XML or SOAP API. For more information, see [OAuth 2.0 for Integration Applications Developers](#).

 **Note:** OpenAir only supports the OAuth 2.0 authorization code grant type.

- **End-users** can give applications explicit permission to access OpenAir on their behalf and they can revoke this permission at any time. For more information, see [Authorizing Applications to Access OpenAir on Your Behalf](#).

 **Note:** The first time a registered application attempts to access OpenAir on their behalf, users must sign in using the same trusted login form they normally use to log in to OpenAir then give the application explicit permission. The OAuth 2.0 feature supports the following user authentication mechanisms:

- Password authentication by OpenAir — Users enter their Company ID, User ID and Password on the OpenAir login form.
- SAML authentication:
 - Service Provider initiated Single Sign-on — Users enter their login details on your company Single Sign-on form.
 - Identity Provider initiated Single Sign-on — Users must log in using their Identity Provider Single Sign-on form before the application attempts to access OpenAir on their behalf. When the application attempts to access OpenAir, the authorization screen appears automatically. Users do not need to enter their login details again if the Single Sign-on session has not expired.

Managing API Integration Applications in OpenAir

Integration applications using OAuth 2.0 to obtain access to your OpenAir data must be registered and enabled by an account administrator. To register and manage your integration applications, go to Administration > Global Settings > Account > API Integration Applications.

 **Note:** OpenAir API access must be enabled for your account to connect tools and services to OpenAir using OpenAir APIs. The API Integration Application screen is not available if OpenAir API access is not enabled. To enable OpenAir API access for your account, contact OpenAir Customer Support or your OpenAir account manager.

1. All your registered applications are listed in a grid. Details include the name of the application and the date and time when it was last updated.
2. To register a new application, click **ADD NEW APP**. This button is disabled if you reach the limit of 20 registered applications. See [Adding a New Application](#).
3. To enable or disable an application, click **ENABLE** or **DISABLE** in the top right corner of the corresponding box. See [Enabling, Disabling, or Removing Registered Applications](#)
4. To edit an application configuration, click the edit icon  in the bottom right corner of the corresponding box. See [Application Configuration](#).
5. To remove an application configuration from the list of registered applications, click the delete icon  in the bottom right corner of the corresponding box. See [Enabling, Disabling, or Removing Registered Applications](#).
6. To select one or more applications, check the box next to each application you want to select. You can only select multiple applications if they are either all enabled, or all disabled. You can then enable, disable or remove the selected applications. See [Enabling, Disabling, or Removing Registered Applications](#).

Note: All times are given as Eastern Standard Time (EST).

The screenshot displays the 'API Integration Applications' page in the OpenAir Administration console. The page includes a sidebar with navigation options like Home, Opportunities, Projects, Resources, Invoices, Timesheets, Expenses, and Purchases. The main content area is titled 'API Integration Applications' and features an 'ADD NEW APP' button (callout 2). Below the title, there is explanatory text and a list of conditions for application use. A section titled 'Your applications' (callout 1) contains a table of four applications:

Application Name	Status	Updated (EST)	Action
CRM Integration (C)	DISABLE	2020-03-20 04:19:15	EDIT (callout 4), REMOVE
Example Application (E)	ENABLE	2020-03-20 05:29:16	EDIT (callout 4), REMOVE
OAuth2 Connector (O)	ENABLE	2020-09-12 18:23:28	EDIT (callout 5), REMOVE
Some New Integration (S)	DISABLE	2020-09-12 18:23:00	EDIT (callout 5), REMOVE

Additional callouts: 3 points to the status indicator, 4 points to the 'Edit configuration' button, and 6 points to the application selection checkbox. The footer includes 'Copyright © Oracle 1999-2020 All rights reserved.', 'OpenAir Powered by ORACLE NETSUITE', and 'Filter set: All Access'.

Adding a New Application

You can register up to 20 applications. Each application needs a Client ID and Client Secret to obtain access to OpenAir using OAuth 2.0. The Client ID and Client Secret are generated by OpenAir as part of the registration process and are unique to each application.



Important: The Client Secret is a **private** key the application uses to request an authorization code from OpenAir. It should not be shared or stored in public code repositories.

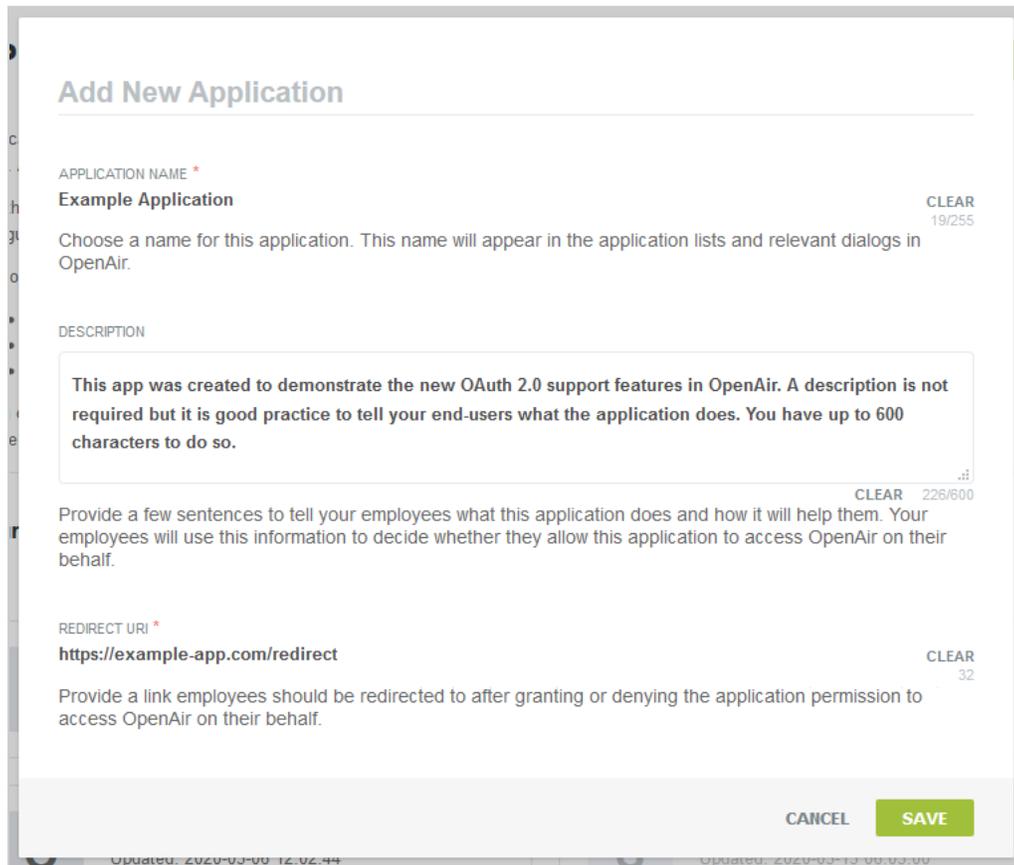
The Client Secret is displayed only once. You will not be able to retrieve it after you close the Application Credentials dialog.

If you misplace the Client Secret, you can edit the application configuration and generate a new Client Secret for the application.

To register a new application with OpenAir:

1. Do one of the following:
 - Go to Administration > Global settings > Account > API Integration Applications, and click **ADD NEW APP**.
 - From any screen in OpenAir, click the Create button and click API integration application.The Add New Application dialog box appears.
2. Enter the following information:
 - **Application name** (Required) — Enter a display name for your application in OpenAir. The name must be unique to the application. You will not be able to use a name already used by another registered application.
 - **Description** — Enter a few sentences to tell your employees what the application and how it will help them. Your employees will use this information to decide whether they allow this application to access OpenAir on their behalf.
 - **Redirect URI** (Required) — Enter a link users should be redirected to after granting or denying the application permission to access OpenAir on their behalf.

Important: Client applications use the redirect URI when requesting access to OpenAir. Ensure you enter the redirect URI supplied by the application developers.



Add New Application

APPLICATION NAME *
Example Application CLEAR 19/255

Choose a name for this application. This name will appear in the application lists and relevant dialogs in OpenAir.

DESCRIPTION
CLEAR 226/600

This app was created to demonstrate the new OAuth 2.0 support features in OpenAir. A description is not required but it is good practice to tell your end-users what the application does. You have up to 600 characters to do so.

Provide a few sentences to tell your employees what this application does and how it will help them. Your employees will use this information to decide whether they allow this application to access OpenAir on their behalf.

REDIRECT URI *
https://example-app.com/redirect CLEAR 32

Provide a link employees should be redirected to after granting or denying the application permission to access OpenAir on their behalf.

CANCEL SAVE

3. Click **Save**. The Application Credentials dialog box appears.
4. Copy the **Client Secret** and store it in a safe place. The Client Secret is displayed only once. You will not be able to retrieve it after you close this window.

Application Credentials

The application will need the following credentials to gain access to OpenAir Web API using OAuth 2.0. The Client ID and Client Secret are generated by OpenAir and are unique to each application.

CLIENT ID COPY TO CLIPBOARD

174_h1FiXfWsJtLJG0DG

CLIENT SECRET COPY TO CLIPBOARD

vcFTaNE3nUXRuJ29jhEIXSH7LpXwbxTGdmJHoQMvo_jz4vldOPITFJ-ZFToYR2G6gnl0dqXeeiFUloKnRGSfQ

- The client secret should not be shared or stored in public code repositories.
- The client secret is displayed only once. You will not be able to retrieve it after you close this window. You can generate a new client secret at any time.

Copy the client secret and store it in a safe place.
To close this window, check the box to confirm you have stored the client secret in a safe place and click Close.

I have stored the client secret in a safe place
 CLOSE

5. Check the box to confirm you have copied and stored the Client Secret in a safe place then Click **Close**.

Enabling, Disabling, or Removing Registered Applications

You must enable an application to allow this application to obtain access to OpenAir using OAuth 2.0.

You can disable an application to prevent this application from obtaining access to OpenAir using OAuth 2.0. If you disable an application OpenAir automatically revokes all permissions given by users for the application to access OpenAir on their behalf. Employees will not be able to use the disabled application.

You can remove a disabled application from the list of registered applications. All permissions, authorizations and application credentials associated with the application configuration will be deleted. This action cannot be undone.

To enable or disable a registered application:

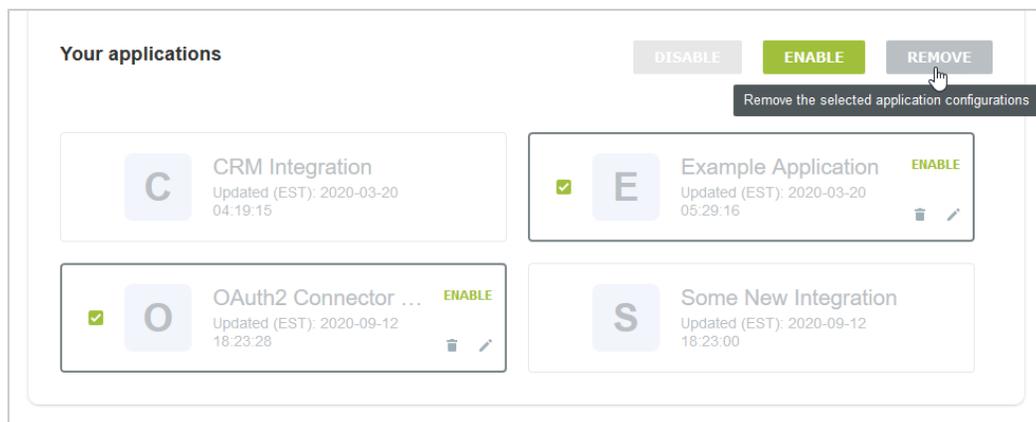
1. Go to Administration > Global settings > Account > API Integration Applications.
2. Click **ENABLE** or **DISABLE** in the top right corner of the corresponding box. A confirmation dialog box appears.
3. Click **ENABLE** or **DISABLE** to enable or disable the application. Click **Cancel** to cancel the operation and return to the API Integration Applications screen.

To remove a registered application:

1. Go to Administration > Global settings > Account > API Integration Applications.
2. Click the delete icon  in the bottom right corner of the corresponding box. A confirmation dialog box appears.
3. Click **REMOVE** to remove the application. Click **Cancel** to cancel the operation and return to the API Integration Applications screen.

To enable, disable, or remove multiple applications at the same time:

1. Go to Administration > Global settings > Account > API Integration Applications.
2. Check the box for each application you want to enable, disable, or remove. Notice that you can only select multiple applications if they are either all enabled, or all disabled. After you select the first application, the application that are not available for selection appear in light gray color. Notice also that some of the buttons in the top right corner of the list of registered applications become available and change from light gray color to dark gray or green.



3. Click **ENABLE**, **DISABLE**, or **REMOVE** to perform the corresponding action on all selected applications. A confirmation dialog box appears. Click **ENABLE**, **DISABLE**, or **REMOVE** to confirm. Click **Cancel** to cancel the operation and return to the API Integration Applications screen.

Application Configuration

You can view the configuration details of your registered applications, including their unique Client ID from the Application Configuration form. You can change the application name, description or Redirect URI or generate a new Client Secret for the application.

To open the Application Configuration screen for a registered application, go to Administration > Global Settings > Account > API Integration Applications and click the edit icon  in the bottom right corner of the corresponding box.

1. The **General** section of the form lists the main application detail:
 - You can change the **Application name**, **Description** and **Redirect URI**.

 **Important:** Client applications use the redirect URI when requesting access to OpenAir. Ensure you enter the redirect URI supplied by the application developers. If you need to change the redirect URI, disable the application, change the redirect URI and enable the application again.

- You can view when the application was registered under **Created**.

Note: All times are given as Eastern Standard Time (EST).

2. You can view the **Client ID** — the unique identifier a client application needs to send to OpenAir along with a client secret as part of the OAuth 2.0 authorization code flow.
3. Use the **Tokens Lifetime** section to configure the validity period of the access and refresh tokens:
 - **Access token lifetime** — Select the expiration time of access tokens. Available values go from 5 to 60 minutes in 5-minute increments. The default access token lifetime is 15 minutes.

Note: The validity period of access tokens cannot be greater than the session timeout set for your account. If the **Access token lifetime** value is greater than the session timeout value, the session timeout value is used for the access token validity period. The application configuration form shows the current values for the session timeout and access token validity period for reference.

To change the session timeout value, go to Administration > Global settings > Account > Security.

- **Refresh token lifetime** — Select the expiration time of refresh tokens. Available values go from 1 to 31 days in one-day increments. The default access token lifetime is 1 day.

Note: Before the October 2021 OpenAir release, you could set the refresh token lifetime to values from 1 to 24 hours in one-hour increments. Values for the refresh token lifetime set before the October 2021 OpenAir release show in days (decimal values) instead of hours

As part of the OAuth 2.0 authorization code flow, authorized applications need to exchange an authorization code for an access token and refresh token to obtain access to OpenAir. The access token has a short expiration time. When the access token expires, the client application can use the refresh token to obtain a new access token without user interaction until the refresh token expires or the authorization is revoked.

Note: Access tokens normally remain valid for their entire lifetime. However, the access token becomes invalid before it is due to expire if any of the OpenAir business rules have changed and the access token is refreshed. Business rule changes may include any changes to the OpenAir configuration, or to the access privileges or role permissions of the employee who authorized the client application.

Each refresh token can be used one time only. Refresh in an access token generates a new access token and a new refresh token.

4. To generate a new Client Secret, click **Regenerate Secret** — You may need to generate a new client secret if you misplace or delete the client secret accidentally or if your client secret becomes compromised.

The new client secret will be valid immediately. The old client secret will continue to be valid for 24 hours after you generate a new one. This allows time to update any enabled application with the new client secret.

5. If you made any changes to the configuration details in the General section, the **Save** button is enabled. Click **Save** to save changes and return to the API Integration Applications screen or click **Cancel** to close the configuration form without saving.

API Integration Applications

CANCEL
SAVE
5

Example Application

General 1

APPLICATION NAME *
Example Application CLEAR 13/255

Choose a name for this application. This name will appear in the application lists and relevant dialogs in OpenAir.

CREATED (EST)
2020-03-06 12:03:40

DESCRIPTION CLIENT ID

This app was created to demonstrate the new OAuth 2.0 support features in OpenAir. A description is not required but it is good practice to tell your end-users what the application

Provide a few sentences to tell your employees what this application does and how it will help them. Your employees will use this information to decide whether they allow this application to access OpenAir on their behalf.

147_64j8bj8YMB7wL9 2

REDIRECT URI *
https://example-app.com/redirect CLEAR 32

Provide a link employees should be redirected to after granting or denying the application permission to access OpenAir on their behalf.

Tokens Lifetime 3

After the employee authenticates successfully and authorizes access, the application receives an Access tokens and a Refresh token. Access tokens have a short expiration time. When the Access token expires, the application can use the Refresh token to obtain a new Access token without employee interaction until the Refresh token expires or the authorization is revoked.

ACCESS TOKEN LIFETIME
45 minutes

Select the validity period of the Access token in minutes.

i The validity period of access tokens cannot be greater than the session timeout set for your account. If the access token lifetime value is greater than the session timeout value, the session timeout value is used for the access token validity period.

The current session timeout value is: 30 minutes
The current access token validity period is: 30 minutes

To change the session timeout value, go to Administration > Global settings > Account > Security.

REFRESH TOKEN LIFETIME
31 days

Select the validity period of the Refresh token in days.

Client Secret

You can generate a new Client Secret at any time by clicking **Regenerate Secret**. Copy the new Client Secret and store it in a safe place. It should not be shared or stored in public code repositories. This Client Secret is displayed only once. You will not be able to retrieve it after you close this window.

REGENERATE SECRET
4

Last generated (EST): 2020-03-06 12:03:40

Auditing and Managing OAuth 2.0 Authorizations

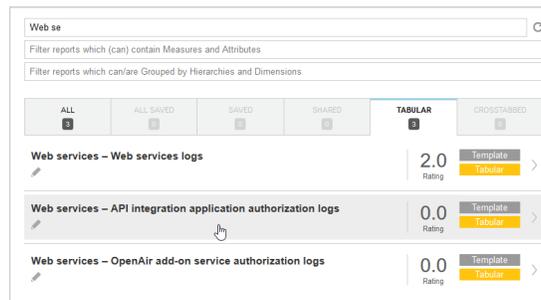
Account administrators can use web services reports to audit and revoke authorizations granted by OpenAir users to integration applications utilizing OAuth 2.0 to connect to OpenAir data.

- **API integration application authorization logs** — User authorizations granted to custom or third party applications registered with your OpenAir account in Administration > Account > API integration applications.
- **OpenAir add-on service authorization logs** — User authorizations granted to OpenAir add-on services (OpenAir Mobile and other add-on service applications).

The reports include information about which integration applications were authorized, when, and by which users. The reports also include a link to revoke the authorization given for an integration application by a user.

To access the OAuth 2.0 authorizations logs (if the Report Management feature is enabled):

1. In OpenAir, go to Reports > Management.
2. Enter “web services” in the **Search saved reports by name** box. The Report Management UI shows the list of web-services tabular reports.
3. Click the report name, then click **New** to create a new report.
4. Add columns and define filters as required.
5. (Optional) Click Untitled in the top bar and enter a name for your report.
6. (Optional) Click **Save** to save the report you created for later use. The Report Management UI will list the report under on Saved reports tab.
7. Click **Run** to run the report.



User	Application	Application type	Authorization granted	Action	Audit trail
Favre, Brett	Example Application	API Integration Application	04-Apr-2020 10:33 AM	Revoke	Created at 04-Apr-2020 10:33 AM
Admin, Jim	My first REST API integration	API Integration Application	12-Aug-2020 09:07 AM	Revoke	Created at 12-Aug-2020 09:07 AM
Admin, Jim	Example Application	API Integration Application	05-Apr-2020 07:51 AM	Revoke	Created at 05-Apr-2020 07:51 AM
Admin, Jim	OAuth Connector Application	API Integration Application	05-Apr-2020 10:10 AM	Revoke	Created at 05-Apr-2020 10:10 AM
Admin, Jim	Some New Integration	API Integration Application	05-Apr-2020 10:49 AM	Revoke	Created at 05-Apr-2020 10:49 AM
Garcia, Don	My first REST API integration	API Integration Application	10-Feb-2021 02:54 PM	Revoke	Created at 10-Feb-2021 02:54 PM

To access the OAuth 2.0 authorizations logs (if the Report Management feature is not enabled):

1. In OpenAir, go to Reports > Detail.
2. Click the report name under the Web services heading. The report options form appears.
3. (Optional) Set a date range for the **Authorization granted** filter. Defaults to All.
4. Click **Report layout** and select the columns to include, or keep the default layout.
5. (Optional) Click **Employee** and select the employees to include in the report.

6. (Optional) Click **API integration application** and select the applications to include in the report.
7. (Optional) Check the **Save this report as** box and enter a name for the report
8. (Optional) Click **Save** to save the report. The report will be accessible in Reports > Saved reports.
9. Click **Run** to run the report.

OAuth 2.0 for Integration Applications Developers

OpenAir supports two methods to access OpenAir data using OpenAir XML or SOAP API requests:

- Using user credentials (Company ID, User ID, password) and, in the case of OpenAir SOAP web services, a session ID.
- Using OAuth 2.0 access tokens.

OAuth 2.0 bearer token authentication is the only supported method to access OpenAir data using OpenAir REST API.

In the OAuth 2.0 scenario, client applications use one of the OAuth 2.0 grant types to get an access token after the user authorizes the application. The user's identity is verified by an authentication service, which issues the access token. The access token can then be used to gain authenticated access to OpenAir through the XML API, SOAP API or REST API.

This section describes how to get an access token using the OAuth 2.0 authorization code grant type in your applications, and how to use the access token in your API calls.

 **Note:** OpenAir only supports the OAuth 2.0 authorization code grant type, which defines a particular workflow client applications can use to obtain the access token.

OAuth 2.0 Authorization Code Flow

Application developers can use the OAuth 2.0 redirection-based authorization code grant type to obtain an access token. This method eliminates the need for client applications to collect and store user credentials.

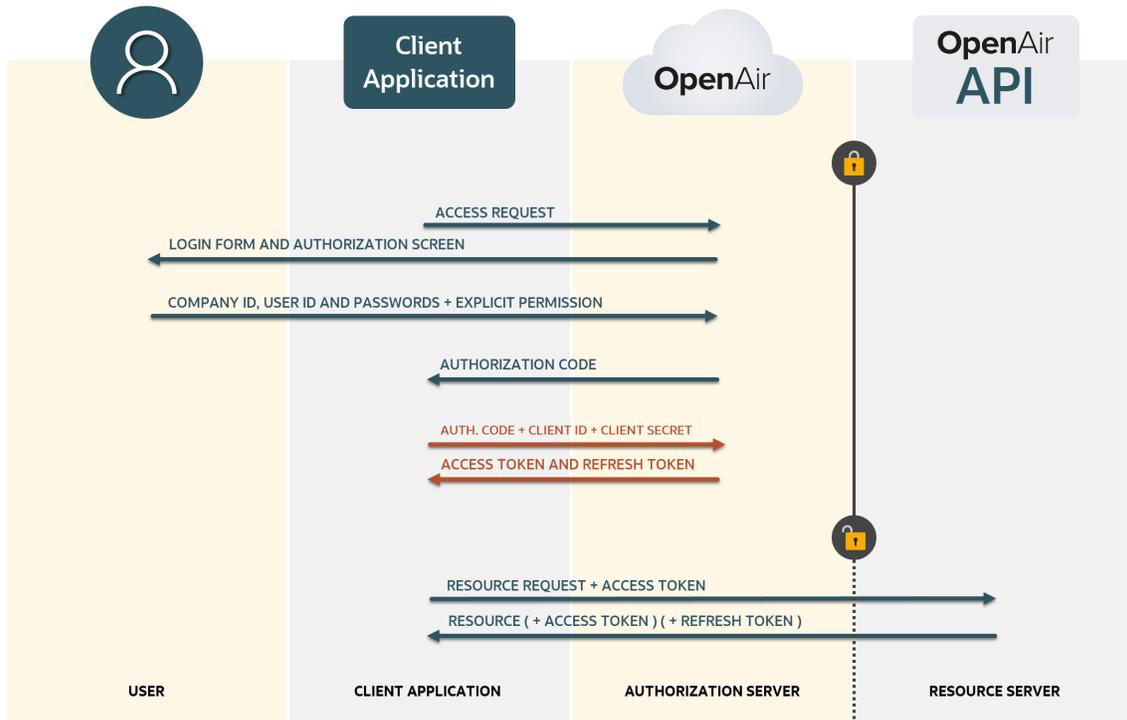
The authorization code flow includes the following steps:

1. **Getting the user's explicit permission** to access OpenAir on their behalf. See [Getting the User's Permission](#).
 - a. The client application opens a browser and directs the user to the OpenAir identity authentication service with the necessary URL query string parameters.
 - b. The user enters user credentials in the OpenAir login form or in a third-party identity provider Single Sign-on login form. The authenticated user is then prompted to authorize the application's access request.
2. **Receiving the authorization code** — OpenAir issues an authorization code. The user is redirected back to the client application with the authorization code in the query string. See [Receiving the Authorization Code](#).
3. **Exchanging the authorization code for an access token** — The client application must exchange the authorization code for an access token and a refresh token. See [Exchanging the Authorization Code for an Access Token](#).

An additional step — **Refreshing an access token** — is required to get a new access token after the previously issued access token has expired. See [Refreshing an Access Token](#).

Note: You must send a request to one of the OpenAir OAuth 2.0 endpoints for each of these steps. For information about OpenAir OAuth 2.0 URLs, see [OAuth 2.0 Endpoints URL Schema and Account-Specific URLs](#).

You can then use OAuth 2.0 token based authentication for your OpenAir API calls. See [Using OAuth 2.0 Access Tokens in Your API Requests](#).



OAuth 2.0 Endpoints URL Schema and Account-Specific URLs

For each step of the OAuth 2.0 authorization code flow, you must send requests to the authorization server using URLs specific to each type of request.

The following URL shows how you construct a request URL:

`https://<account-domain>/login/oauth2/v1/<endpoint><query-string>`

- The first part of the URL must include your account-specific domain <account-domain> and the service path for OAuth 2.0. For more information about your account-specific domain name, see the help topic [Use Account-Specific Domain](#).
- The second part of the URL depends on the endpoint you want to access

<endpoint>	Description
authorize	Use the authorization endpoint to get the user's explicit permission and receive an authorization code in response if the user authorizes the app to access OpenAir on their behalf. The request URL includes a query string with request parameter. <code>https://<account-domain>/login/oauth2/v1/authorize?<query-string></code> See Getting the User's Permission and Receiving the Authorization Code .
token	Use the token endpoint to exchange the authorization code for an access token or to refresh an access token. Request parameters are passed in the request headers and body.

<endpoint>	Description
	https://<account-domain>/login/oauth2/v1/token See Exchanging the Authorization Code for an Access Token and Refreshing an Access Token .

Getting the User's Permission

To begin the OAuth 2.0 authorization code flow, the client application must direct the user to the authorization server — OpenAir — using a GET request.

Send a GET request to the authorization endpoint using a URL like the following example:

```
https://company-id.app.openair.com/login/oauth2/v1/authorize?
response_type=code&redirect_uri=https://example-app.com/
redirect&client_id=174_h1FiXfWsJtLJG0DG&scope=xml+soap+rest&state=ryjp37y2qa28hdseck1gat
```

The GET request URL includes the authorization endpoint for the OpenAir account followed by a query string: https://<account-domain>/login/oauth2/v1/authorize?<query-string>.

The request parameters are described in the following table.

Request parameter	Description
response_type	The value of the response_type parameter is always code. It tells the authorization server that the client application is initiating the OAuth 2.0 authorization code flow.
redirect_uri	The valid redirect URI where the application will process the authorization code. The user should be redirected to this URI after allowing or denying the access request. The redirect URI must match the redirect URI specified on the application configuration form in OpenAir.
client_id	The public identifier for the client application. The Client ID is generated by OpenAir when an administrator registers the client application.
scope	One or more plus-separated scope values indicating the access requested by the application. The scope determines which OpenAir APIs the application will be able to access. <ul style="list-style-type: none"> ■ OpenAir currently supports the following scope values: xml, soap, rest. ■ OpenAir accepts multiple scope values. The scope values are case insensitive. ■ Authorized applications have the same permissions and data access privileges as the user authorizing the application within the selected scope.
state	A random string generated by the client application, which is used to prevent cross-site request forgery (CSRF) attacks. For more information see the OAuth 2.0 specification RFC6749 Section 10.12 .

After the application sends the GET request, OpenAir redirects the user to the OpenAir login form. OpenAir may redirect the user to a third-party Identity provider Single Sign-on form, if SAML SSO is enabled for the account and the user. After successful authentication, OpenAir displays an authorization screen prompting the user to approve the application's access request.

Receiving the Authorization Code

After obtaining the user's explicit permission, OpenAir initiates a redirect to the redirect URI specified in the GET request with the authorization code and the state as query parameters.

The redirect query parameters are described in the following table.

Redirect parameter	Description
state	The client application should check that the state in the redirect matches the state set in the GET request initiating the OAuth 2.0 authorization code flow. Validating the state sent to and returned from the authorization server can be used to prevent cross-site request forgery (CSRF) attacks.
code	<p>The authorization code issued by OpenAir.</p> <ul style="list-style-type: none"> It is a unique single use code issued only for the client application requesting access. The authorization code is valid for 10 minutes. The client application must exchange the authorization code for an access token before the authorization expires.

The following sample redirects illustrate successful and unsuccessful authorization.

- Application successfully authorized.

```
https://example-app.com/redirect?state=ryjp37y2qa28hdseck1gat&code=JTlQ43UvYDKbI_SpEwsIE_bTp bou2-kYeeLtkiMur1iqZ3W3roqM4rmRC8fFCOJtBI6a85AnJPefx2szw9g4jCY
```
- Application not authorized.

```
https://example-app.com/redirect?error_description=The+resource+owner+or+authorization+server+denied+the+request&error=access_denied&state=ryjp37y2qa28hdseck1gat
```

Exchanging the Authorization Code for an Access Token

The application can use the authorization code to obtain an access token and a refresh token using a POST request.

Send a POST request to the token endpoint.

- The POST request URL is `https://<account-domain>/login/oauth2/v1/token`
- The request must include the client ID and the client secret in the HTTP authorization request header.
 - The client authentication method used in a header of the request follows the HTTP Basic authentication scheme. For more information, see [RFC 7617](#).
 - The format is `client_id:clientsecret`.
 - The string value is Base64 encoded.
- The request must include the parameters `grant_type`, `code` and `redirect_uri` in the request body.
 - Request parameters must be encoded based on the HTML specification for `application/x-www-form-urlencoded` media type. For more information, see [URL Specification 5.1](#).

The request parameters are described in the following table.

Request parameter	Description
grant_type	The value of the <code>grant_type</code> parameter is <code>authorization_code</code> . It tells the token endpoint that the client application is using the OAuth 2.0 authorization code grant type.
code	The authorization code issued by OpenAir and received by the client application in the redirect.
redirect_uri	The valid redirect URI. The redirect URI must match the redirect URI specified on the application configuration form in OpenAir and when requesting the authorization code.

Request parameter	Description
client_id	The public identifier for the client application. The Client ID is generated by OpenAir when an administrator registers the client application.
client_secret	The client secret for the application. This ensures that the request to get the access token is made only from the application, and not from a potential attacker that may have intercepted the authorization code.

Example POST request:

```

1 | POST /login/oauth2/v1/token HTTP/1.1
2 | Host: company-id.app.openair.com
3 | Authorization: Basic MTc0X2gxRmlyZlZlZnRMSkcwREc6dmNGVGFORTNuVvHsDUoyOWpoRwxYU0g3THBYd2J4VEdkbUpIb1FNdm9fano0dmxkT0ZQSVRGSi1aR1RvWVIyRzZnbmwwZHFYZwVpRlVJb0tuUkdTZlEK
4 | Content-Type: application/x-www-form-urlencoded
5 | code=JTlQ43UvYDKbhI_SpEwsIE_bTpbou2-kYeeLtkiMuR1iqZ3W3roqM4zmRC8fFC0JtBI6a85AnJPefx2szW9g4jCY&redirect_uri=https%3A%2F%2Fexample-app.com%2Fredirect&grant_type=authorization_code
    
```

The token endpoint will verify all the parameters in the request to ensure that the authorization code is valid and that the client ID and client secret match. If all the request headers and parameters are valid, the token endpoint generates an access token and refresh token and includes them in the response.

The token endpoint returns the response as a JSON object with the properties described in the following table.

JSON object properties	Description
access_token	The access token in JSON Web Token (JWT) format. The access token is valid for the period configured in OpenAir for the application. See Application Configuration .
refresh_token	The refresh token in JSON JWT format. The refresh token is valid for the period configured in OpenAir for the application. See Application Configuration .
expires_in	The access token expiration time in seconds. The access token is valid for the period configured in OpenAir for the application. See Application Configuration .
type	The value of the type property is always bearer. For more information, see the OAuth 2.0 specification — RFC 6750 .

Example response (successful token request):

```

1 | HTTP/1.1 200 OK
2 | Content-Type: application/json
3 | Cache-Control: no-store
4 | Pragma: no-cache
5 | {
6 |   "refresh_token": "WGxpeGNVTm1mNGhaS2E1djFQQ2twV1pKcWpOU0pXUkhZUU5oMTR1MFU5OUtLY3N1NkZKOG9SMHp4UnNumjYyRTJGcm9NVUo50WxENDFzcw5WSDFsUEhoSF8xNzQ",
7 |   "expires_in": 900,
8 |   "access_token": "eNNJ1GX025-6IUy1F6RZT33HqhoqSAAK53F0kxT62fBoKreDoc8Y_-Gnk21UIqNbhgwuHnxDtxUsJMY6NrDoiBnd",
9 |   "token_type": "bearer"
10 | }
    
```

If the request fails, the token endpoint returns a JSON object with the error and error_description properties. See [Token Request Errors](#).

The client application can now use the access token to make API requests. This completes the authorization flow.

The access token is only valid for a short period of time and within the scope it was issued for. The client application will need to refresh the access token to continue making API requests after it expires.

Refreshing an Access Token

The access token has a short expiration time (15 minutes). When the access token expires, the client application can use the refresh token to obtain a new access token using a POST request.

- You can use the expiration time value (`expires_in`) to refresh access tokens before it is due to expire.
- You can refresh access tokens if an API request returns an authentication failed error.
- Each refresh token can be used one time only. Refreshing an access token generates a new access token and a new refresh token.

Send a POST request to the OpenAir token endpoint. The POST request is similar to that used to exchange an authorization code for an access token except it now uses the parameters set in the following table.

Request parameter	Description
<code>grant_type</code>	The value of the <code>grant_type</code> parameter is <code>refresh_token</code> . It tells the token endpoint that the client application is requesting to refresh an access token.
<code>refresh_token</code>	A valid <code>refresh_token</code> . Refresh tokens are valid for the period configured in OpenAir for the application. See Application Configuration .
<code>scope</code>	(Optional) The requested scope must be within the scope the original access token was issued for. If omitted, the new access token will be issued for the same scope as the original access token.
<code>redirect_uri</code>	The valid redirect URI.
<code>client_id</code>	The public identifier for the client application.
<code>client_secret</code>	The client secret for the application.

Example POST request:

```

1 | POST /login/oauth2/v1/token HTTP/1.1
2 | Host: company-id.app.openair.com
3 | Authorization: Basic MTC0X2gxRmlyZlZlZS5nRm5kcwREc6dmNGVGFORTNuVHsduoyOWpoRwxYU0g3THBYd2J4VEEdkbUpIb1FNdm9fano0dmxkT0ZQSVRG5i1aRlRvWVIyRzZnbmwwZHFYZWVpRlVJb0tuUkdTZlEK
4 | Content-Type: application/x-www-form-urlencoded
5 | refresh_token=WGxpeGNVTm1mNGhaS2E1djFQQ2twV1pKcWpOU0pXUkhZUU5oMTR1MFU5OUtLY3N1NkZKOG9SMHp4UnNuMjYyRTJGcm9NVUo50WxENDFzcw5WSDFsUEhoSF8xNzQ&redirect_uri=https%3A%2F%2Fexample-app.com%2Fredirect&grant_type=refresh_token
    
```

The token endpoint will verify all the parameters in the request to ensure that the refresh token is valid and that the client ID and client secret match. If all the request headers and parameters are valid, the token endpoint generates an access token and refresh token and includes them in the response.

The token endpoint returns the response as a JSON object with the properties described in the following table. The response includes both a new access token and a new refresh token.

JSON object properties	Description
<code>access_token</code>	The access token in JSON Web Token (JWT) format. The access token is valid for the period configured in OpenAir for the application. See Application Configuration .
<code>refresh_token</code>	The refresh token in JSON JWT format. The refresh token is valid for the period configured in OpenAir for the application. See Application Configuration .
<code>expires_in</code>	The access token expiration time in seconds. The access token is valid for the period configured in OpenAir for the application. See Application Configuration .
<code>type</code>	The value of the <code>type</code> property is always <code>bearer</code> . For more information, see the OAuth 2.0 specification — RFC 6750 .

Note: Access tokens normally remain valid for their entire lifetime. However, the access token becomes invalid before it is due to expire if there are any changes in the OpenAir configuration, or in the access privileges or role permissions of the employee who authorized the client application, and the application uses the access token is refreshed.

Example response (successful token request):

```

1 HTTP/1.1 200 OK
2 Content-Type: application/json
3 Cache-Control: no-store
4 Pragma: no-cache
5 {
6   "refresh_token": "N25RdE82N0FIMnBDRTQ1d3hITHN6dXRwQ3dNdzVHWHMydWd3WwFqUXMwMwGrcTB3UHBFQ3VLaUZQT1RuR0J3U09LaWcvWWJ2UHhpPS2hWUUtX
  Q1JCMzBHVf8xNzQ",
7   "expires_in":900,
8   "access_token": "pWprqUHKgaOWai7fSjNaN5ywpDr7a7W6hLiq-b1vBBAT48zxAPTyy-wpTNxyfwJieQzZ5E1HE0sG1hNtwJpILR5",
9   "token_type": "bearer"
10 }

```

If the request fails, the token endpoint returns a JSON object with the error and error_description properties. See [Token Request Errors](#).

Token Request Errors

Your client application needs to handle the following cases when the request to exchange an authorization code for an access token or to refresh a token fails. The token endpoint may return one of the errors listed in the following table if the token request fails.

- Errors are listed in descending priority order. Only the first error is returned if there are more than one.
- Some of the errors are specific to one of the two possible types of request (grant_type).

error	error_description	grant_type	Reason
unsupported_grant_type	The authorization grant type is not supported by the authorization server		The grant_type must be either authorization_code or refresh_token.
invalid_request	Authorization header not sent		Request headers must include a Basic Authorization header.
invalid_request	No credentials provided		The Client Id and Client Secret must be sent in the request Authorization header.
access_denied	Authorization code is not valid	authorization_code	<p>The authorization code must be valid. Possible reasons include:</p> <ul style="list-style-type: none"> Expired authorization code — The authorization code is valid for 10 minutes. Authorization revoked — Users can revoke an application at any time. Disabled application — Account administrators can disable an application at any time. Application removed — Account administrators can

error	error_description	grant_type	Reason
			remove an application at any time.
invalid_request	redirect_uri or client_id is not valid	authorization_code	The redirect URI and Client ID must match the information specified on the application configuration form in OpenAir.
access_denied	Refresh token is not valid	refresh_token	<p>The refresh token sent in the request is not valid. Possible reasons include:</p> <ul style="list-style-type: none"> ■ Expired refresh token — Refresh tokens are valid for 24 hours. ■ Authorization revoked — Users can revoke an application at any time. ■ Disabled application — Account administrators can disable an application at any time. ■ Application removed — Account administrators can remove an application at any time.
invalid_scope	Changing scopes is not supported	refresh_token	<p>Tokens are issued for a specific scope. The scope of the token requested must be within the scope of the token sent in the request. For example, if the scope of the original token includes both <code>xml</code> and <code>rest</code>, the scope of the access token requested can be <code>rest</code> only.</p>
access_denied	Authorization failed		<p>The Client Id and Client Secret pair sent in the request is not valid. Note that account administrators may generate a new Client Secret for the application in OpenAir.</p>
access_denied	API access via OAuth2 is disabled		<p>OpenAir API access is not enabled for the OpenAir account.</p>

Note: If applicable, the client application can initiate the OAuth 2.0 authorization code flow again to obtain a new authorization code and exchange it for an access token. The end user will be directed to the login form and required to enter valid user credentials. If the user revoked the application the authorization screen will appear. If the application is still authorized, the authorization endpoint will return a new authorization code immediately after the successful user authentication.

Using OAuth 2.0 Access Tokens in Your API Requests

You can use OAuth 2.0 access token authorization instead of password authentication or Session ID in your API requests. OpenAir XML API, OpenAir SOAP API and OpenAir REST API support authorization using access tokens. OAuth 2.0 access token authorization is the only supported authentication method with OpenAir REST API.

- In your **XML API** requests, use the [Auth XML](#) command. See [OAuth 2.0 Access Token Authentication](#).
- In your **SOAP API** requests, use the [SessionHeader](#) web services method complex type to hold the access token. See [Using SessionHeader for OAuth2.0 Token Based Authentication](#).
- In your **REST API** requests, send the access token as a bearer token in the HTTP authorization header. See the help topic [Authentication](#).

Note: For REST API requests, the access token lifetime will either be the **Access token lifetime** set on the application configuration form in OpenAir, or the **Session timeout** set in Administration > Global settings > Account > Security options, whichever is the lower.

Note: Both OpenAir XML API and OpenAir SOAP API continue to support password authentication. OpenAir SOAP API continues to support SessionID.

Authorization Errors

OpenAir API access must be enabled and API requests must use a valid access token with a valid scope.

- The access token must exist.
- The access token must not be expired.
- The user who gave the application permission to access OpenAir must be active. The same access token can be used if the user is set to active again.
- The application must be enabled for the OpenAir account. The same access token can be used if the application is disabled and then enabled again.
- The access token must be used within the scope it was issued for. For example, if the authorization code was obtained for the scope `xml+rest`, the client application cannot use the access token in a SOAP API request.

The error code and message returned depend on the API:

- OpenAir XML API returns error code 420 and message Authentication failed.
- OpenAir SOAP API returns error code 9 and message Logged out.
- OpenAir REST API returns HTTP status 401 Unauthorized and the response includes a WWW-Authenticate header `error="invalid_token", error_description="The access token is invalid"`.

An invalid OAuth 2.0 access token authorization has priority over a valid password authentication. You cannot use password authentication (Company ID, User ID, password) — or a valid Session ID in the SOAP session header — as a fallback for an invalid access token.

Authorizing Applications to Access OpenAir on Your Behalf

Integration applications let you connect OpenAir with other applications and they extend what you can do with OpenAir. Integration applications may use the OAuth 2.0 authorization protocol to gain access to your OpenAir account.

The first time an application using the OAuth 2.0 protocol attempts to access OpenAir on your behalf, you will need to give this application your explicit permission.

To authorize an application, you will typically use the following steps:

1. The application opens a browser and directs you to the same trusted login form you normally use to log into OpenAir — the OpenAir login form or your company Single Sign-on form appears.
2. Enter your login details and click **Log in**.

An authorization screen will appear indicating that the application <application name> would like to access your OpenAir data.

3. Read the content of the authorization screen attentively. It should describe what the application does and how it will help you. It should also say what the application can do, for example:
 - The application will be able to access all data you have access to.
 - The application will be able to perform all actions permitted by your role and user privileges.



Important: For Administrators — Business rules configured for your OpenAir account are applied when an integration application interacts with your OpenAir data through OpenAir REST API. However, they are not applied when an integration application interacts with your OpenAir data through OpenAir SOAP API or XML API — application developers must enforce business rules within their integration application if required. Business rules include OpenAir account configuration settings and access control mechanisms, as well as any user scripts deployed on your OpenAir account.

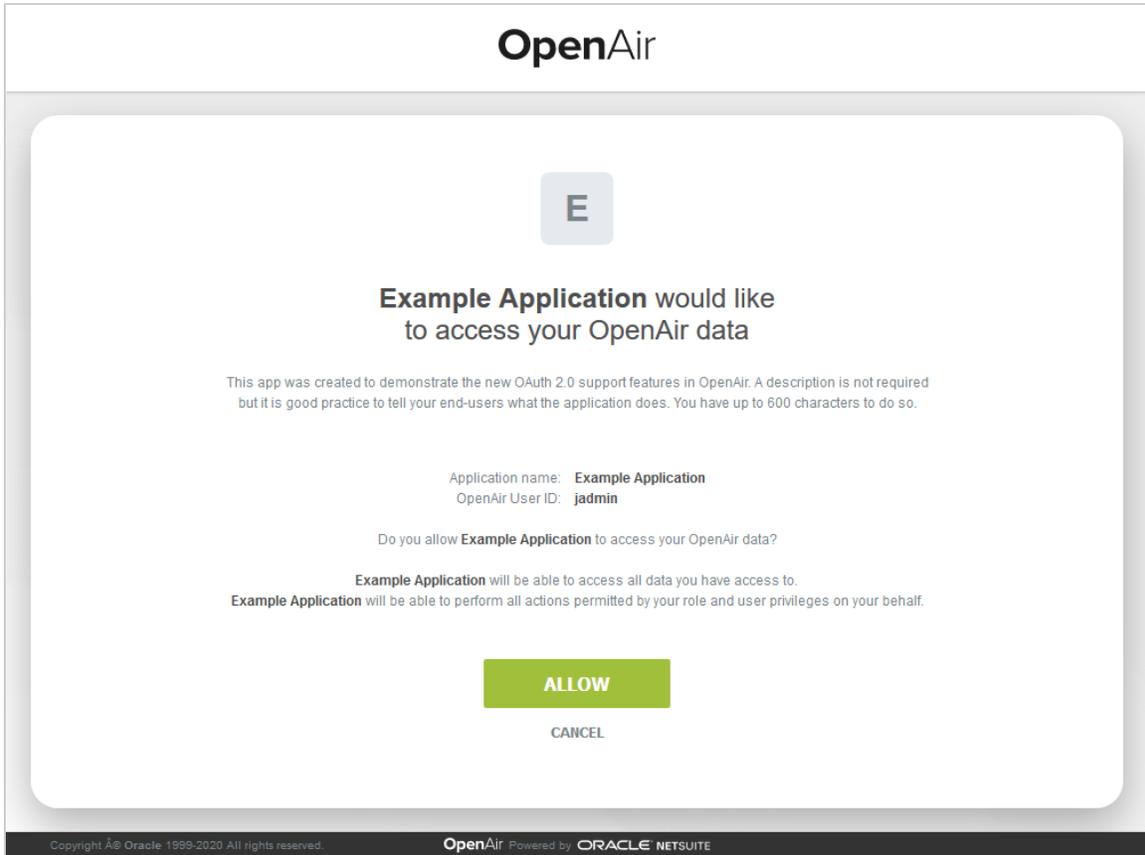
4. Click **ALLOW** to authorize the application or click **CANCEL** if you do not want the application to access OpenAir on your behalf.



Note: The steps may vary depending on the method you use to log in to OpenAir:

- If you normally enter your company ID, user ID and password in OpenAir or if you enter your company ID and user ID in OpenAir and then your password on your company Single Sign-on page, the above steps apply.
- If you normally need to enter all login details then select OpenAir from your company Single Sign-on solution to access OpenAir without needing to enter any login details on the OpenAir login page (Identity Provider initiated Single Sign-on), you must log in and select to open OpenAir before the application attempts to access OpenAir on your behalf. The authorization screen appears automatically. Follow steps 3 and 4 above. You do not need to re-enter your login details.

Integration applications are registered and managed by your account administrator. They need to be enabled on your account before they can attempt to connect to OpenAir and request your permission.



Note: Integration applications are registered and managed by your account administrator. They need to be enabled on your OpenAir account before they can attempt to connect to OpenAir and request your permission.

Account administrators can disable an application at any time.

- If you have authorized an application and this application is disabled by an administrator, the application will no longer be able to interact with OpenAir.
- If an administrator enables this application again, you will need to give this application your explicit permission again before you can continue to work with it in connection with OpenAir.

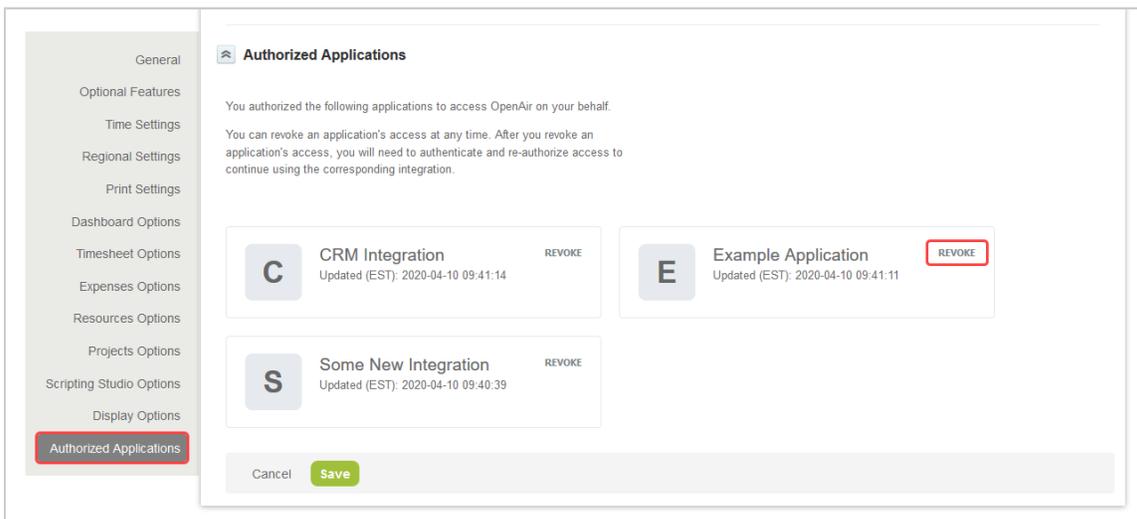
After you authorize an application, it will be able to interact with OpenAir on your behalf until you revoke the authorization.

To view the application you have authorized, go to User Center > Personal Settings > Authorized Applications. All your authorized applications are listed in a grid. Details include the name of the application and the date and time when it was last updated.

Note: All times are given as Eastern Standard Time (EST).

To revoke an application, click **REVOKE** in the top right corner of the corresponding box, then click **REVOKE** in the confirmation message. The application no longer shows in the authorized applications

list. If a revoked application attempts to access OpenAir on your behalf, you will be prompted to give this application your explicit permission again.



Reading Objects

Use the [Read](#) (XML API) or [read\(\)](#) (SOAP API) command to get an object or a list of objects of a specific type.

A read command passes the following parameters and arguments:

- **type** — You must specify the type of object you want to retrieve. For information about supported object types, see [List of Supported Business Object Types](#).
- **method** — You must specify the read method you want to use. You can use a read command to return all objects, objects matching or not matching a search query object, custom field values for each object, or objects associated with a specific user or project. See [Read Methods](#).
- **Attributes** — You must set a `limit` attribute to control the number of objects returned. You can specify other attributes to modify the response when reading objects using the OpenAir XML API or SOAP API. Some attributes control API features available for most supported object types, other attributes can only be used when reading a specific object type. See [Read Attributes](#).
- **Fields** — (Optional) By default, the response includes all object properties. You can specify the object properties included in the response.

Note: By default, the XML API includes all standard object properties in the response. To include custom fields for the object type, set the `enable_custom` attribute to 1. To exclude the `flags` property for User or Company objects, set the `exclude_flags` attribute to 1.

By default, the SOAP API includes all standard object properties in the response except for the `flags` property for User or Company objects. It also includes all custom fields for the object type in the response. To include the `flags` property for User or Company objects, set the `include_flags` attribute to 1.

- **Objects** — Depending on the read method you are using, or whether you are using date filters, you may need to include objects as arguments. See [Read Methods](#) and [Date Filters Usage](#).

Read Methods

The following table lists the supported methods for reading objects using the OpenAir XML API or SOAP API. For each supported read method, the table includes a description of the API response.

Note: Most available business object types support the `all`, `equal to` and `not equal to` methods except where otherwise noted in the reference material for each business object type. See [XML and SOAP API Business Object Reference](#).

Method	Response
all	Returns all objects. Sample codes: <ul style="list-style-type: none"> ■ XML API – Read with all Method and Date Filters ■ SOAP API – Read Not Exported Charges with all Method — C# and Read with all Method and Date Filters — Java
equal to	Returns objects that are matching the search query object specified as argument in the read command.

Method	Response
	<p>The read command must include a search query object as argument. The search query object must be of the same type as the objects you want to retrieve. Include all the standard object property values and custom field values that the returned objects must match. When using OpenAir XML API, set the attribute <code>enable_custom</code> to 1 if you include custom field values in the search query object.</p> <div data-bbox="402 361 1377 451" style="border: 1px solid #ccc; background-color: #fff9c4; padding: 5px;">  Important: Calculated fields are not supported. You cannot include properties corresponding to calculated fields in your search query object. </div> <p>The OpenAir SOAP API lets you specify multiple <code>equal to</code> or <code>not equal to</code> conditions combined with <code>and</code> or <code>or</code> logical operators. See Combining Relational Methods Using Logical Operators (SOAP API).</p> <p>Sample codes:</p> <ul style="list-style-type: none"> ■ XML API – Read with equal to Method and Date Filter and Read Expense Reports Pending Reimbursement with equal to Method ■ SOAP API – Read with equal to Method — C# and Read with equal to Method — Java
not equal to	<p>Returns objects that are not matching the search query object specified as argument in the read command.</p> <p>The read command must include a search query object as argument. The search query object must be of the same type as the objects you want to retrieve. Include all the standard object property values and custom field values that the returned objects must not match.</p> <div data-bbox="402 877 1377 968" style="border: 1px solid #ccc; background-color: #fff9c4; padding: 5px;">  Important: Calculated fields are not supported. You cannot include properties corresponding to calculated fields in your search query object. </div> <p>The OpenAir SOAP API lets you specify multiple <code>equal to</code> or <code>not equal to</code> conditions combined with <code>and</code> or <code>or</code> logical operators. See Combining Relational Methods Using Logical Operators (SOAP API).</p> <p>Sample codes:</p> <ul style="list-style-type: none"> ■ XML API – Read with not equal to Method Matching a Custom Field ■ SOAP API – Read with not equal to Method — C# and Read with not equal to Method — Java
custom equal to	<p>Returns the custom field values for the object with the internal ID in the search query object. Custom field values are returned as <code>CustomField</code> objects (See CustomField).</p> <p>The read command must include a search query object as argument. The search query object must be of the same type as the objects you want to retrieve and must include the internal ID [<code>id</code>] of the object for which you want to retrieve custom field values.</p> <div data-bbox="402 1369 1377 1602" style="border: 1px solid #ccc; background-color: #e1f5fe; padding: 5px;">  Note: All business object types that support custom fields support the <code>custom equal to</code> method. Returned objects include custom field values along with standard object properties by default when using OpenAir SOAP API. When using OpenAir XML API, you can set the <code>enable_custom</code> attribute to 1 to include custom field values in the returned object properties along with standard object properties. </div> <p>Sample codes:</p> <ul style="list-style-type: none"> ■ XML API – Read Custom Field Values with custom equal to Method ■ SOAP API – Read Custom Field Values with custom equal to Method — C#

Method	Response
user	<p> Note: All business objects with a <code>userid</code> reference fields support the <code>project</code> method.</p> <p>Returns objects associated with the user matching the internal ID specified in the <code>User</code> search query object. When reading <code>Projecttask</code> objects, returns project tasks assigned to the user.</p>
project	<p> Note: All business objects with a <code>projectid</code> reference fields and <code>Projecttaskassign</code> support the <code>project</code> method.</p> <p>Returns objects associated with the project matching the internal ID specified in the <code>Project</code> search query object. When reading <code>Projecttaskassign</code> objects, returns project task assignments for tasks in the project.</p>

Combining Relational Methods Using Logical Operators (SOAP API)

The OpenAir SOAP API lets you combine `equal` to or `not equal` to methods using `and` or `or` logical operators.

To do so:

- Set the method parameter to a list of `equal` to and `not equal` to relational operator methods, each separated by a comma followed by an `and` or `or` logical operator. An `and` logical operator is used by default if not specified. The use of grouping or parentheses to modify precedence is not supported.
- Pass the same number of search query objects as arguments. There must be one search query object for each listed method.

Examples:

- If method is set to `equal` to, `not equal` to, which is the same as `equal` to, and `not equal` to, the response returns object that match the first search query object but do not match the second search query object.
- If method is set to `equal` to, `or equal` to, and `not equal` to, the response returns object that do not match the third search query object and match either the first search query object or the second search query object.

Sample code: [Read with Multiple equal to Methods Combined with Explicit OR Condition — C#](#).

Read Attributes

You can use attributes to modify the response when reading objects using the OpenAir XML API or SOAP API. Some attributes control API features available for most supported object types, other attributes can only be used when reading a specific object type.

The following table lists the supported attributes, and for each attribute, the object types and methods that supports the attribute if it can only be used when reading a specific object type or with a specific method, and a description of its usage.

Attribute	Object Type	Supported Methods	Usage
<code>base_currency</code>	<code>Currencyrate</code>	all, <code>equal</code> to	Set the <code>base_currency</code> attribute to a three-letter currency code to get the exchange rates against the specified base currency.

Attribute	Object Type	Supported Methods	Usage
between_date	Entitytag	equal to, not equal to	<p>XML API Only — Use the between_date to set the exact date when the entity tag applied. The response includes only entity tags with start_date before and end_date after the date specified. The value must be a date in the following format yyyy-mm-dd (2023-11-26, for example).</p> <p>Set between_date to 0000-00-00 to return all entries with no start and end dates.</p>
calculate_hours	Timesheet	all, equal to, not equal to, user	Set the calculate_hours attribute to 1 to return the minimum number of hours required [min_hour] and the maximum number of hours allowed [max_hour] on the timesheet as per applicable timesheet rules (Administration > Application Settings > Timesheets > Timesheet Rules).
deleted	—	all, custom equal to, equal to, not equal to, project, user	<p>Set the deleted attribute to 1 to return objects that are marked as deleted. Can be used together with newer-than filter to return objects deleted after a specific date.</p> <p>Set the include_nondeleted attribute to 1 to include objects that are not marked as deleted as well as objects that are marked as deleted.</p>
enable_custom	—	all, equal to, not equal to, project, user	<p>XML API Only — Set the enable_custom attribute to 1 to include custom field values in the returned object properties inline with standard object properties.</p> <p>Custom field values are included in the returned object properties by default when using OpenAir SOAP API.</p>
end_date	Schedulebyday	all	The end_date attribute must be set when retrieving Schedulebyday objects using the read_all method. The value must be a date in the following format yyyy-mm-dd (2023-11-26, for example). You must specify a date range (start date and end date) for which you want to retrieve employee work schedules per date. The OpenAir XML API or SOAP API returns error code 10 with the message "Invalid parameters, specify start_date, end_date, and user_filter attributes" otherwise.
exclude_flags	<ul style="list-style-type: none"> ■ Company ■ User 	equal to, not equal to	XML API Only — Set the exclude_flags attribute to 1 to exclude account or user settings from the response.
field	—	all, custom equal to, equal to, not equal to, project, user	Use the field attribute in conjunction with the filter attribute to set date filters and compare any date fields other than updated. See Filtering .
field_names	—	custom equal to	Use the field_names attribute to list the custom field to be returned. The value must be a

Attribute	Object Type	Supported Methods	Usage
			comma-separated list of custom field names. Listed custom fields must be for the requested object type.
filter	—	all, custom equal to, equal to, not equal to, project, user	Use the filter attribute to return only objects that follow one or more filter criteria. By default, date filters compare the updated field to the date specified. Use the field attribute to compare any date fields other than updated. For more information about the Filtering feature and filter criteria, see Filtering .
generic	User	all	Set the generic attribute to 1 to return generic resources (users) only. By default the API returns named resources (users) only.
include_flags	<ul style="list-style-type: none"> ■ Company ■ User 	all, equal to, not equal to	SOAP API Only — Set the include_flags attribute to 1 to include account or user settings from the response. The flag value is a oaSwitch object.
include_nondeleted	—	all, custom equal to, equal to, not equal to, project, user	Use the include_nondeleted attribute in conjunction with the deleted attribute to include objects that are not marked as deleted as well as objects that are marked as deleted.
limit	—	all, custom equal to, equal to, not equal to, project, user	[Required] Use the limit attribute to control the response pagination with an offset (the number of objects to skip) and the maximum number of objects to be returned per page. See Pagination .
order	—	all, custom equal to, equal to, not equal to, project, user	XML API Only — Use the order attribute to sort the returned list by the specified property in ascending or descending order.
precision	Currencyrate	all, equal to	Use the precision attribute to set the decimal precision (maximum number of digits to the right of the decimal point) for foreign currency exchange rate values.
start_date	Schedulebyday	all	The start_date attribute must be set when retrieving Schedulebyday objects using the read_all method. The value must be a date in the following format yyyy-mm-dd (2023-10-26, for example). You must specify a date range (start date and end date) for which you want to retrieve employees work schedule per date. The OpenAir XML API or SOAP API returns error code 10 with the message "Invalid parameters, specify start_date, end_date, and user_filter attributes" otherwise.
tag_with_name	Entitytag	equal to, not equal to	Set the tag_with_name attribute to 1 to include the name of the associated tag group attribute [tag_group_attribute_name] as property the response. By default, only the internal ID of

Attribute	Object Type	Supported Methods	Usage
			the tag group attribute is returned but not the name.
user_filter	Schedulebyday	all	The user_filter attribute must be set when retrieving Schedulebyday objects using the read_all method. The value must be a comma-separated list of user internal IDs. You must specify the employees for whom you want to retrieve the work schedules per date. The OpenAir XML API or SOAP API returns error code 10 with the message "Invalid parameters, specify start_date, end_date, and user_filter attributes" otherwise.
with_project_only	Customer	all, equal to, not equal to, user	Set the with_project_only attribute to 1 to return only customers associated to one or more project.

Filtering

You can use the filter attribute to specify search criteria when retrieving a list of objects using the [Read](#) (XML API) or [read\(\)](#) (SOAP API) command. The response will only include the objects matching the conditions specified in the request.

The following table lists the supported values for the filter attribute. To apply multiple filters or search criteria, use a single filter attribute and a comma-separated list of filter values. For example, you can set the filter attribute to newer-than,older-than,approved-timesheets to return only approved timesheets or objects associated with approved timesheets updated within a certain date range.

 **Important:** Make sure there are no spaces between the commas and the filter values.

 **Note:** In the following table "objects associated with a(n) <associated object type>" means that one of the properties of the object type requested references the ID of the <associated object type> object. For example, an object associated with an expense report (envelope) has the property envelope_id or envelopeid. You should not use a filter if the requested object type does not reference the associated object on which the filter is based. For example, you should not use the filter open-envelopes when reading Project objects.

Filter Value	Response
open-envelopes	Returns only open expense reports (envelopes) or objects associated with an open expense report (envelope).
approved-envelopes	Returns only approved expense reports (envelopes) or objects associated with an approved expense report (envelope).
rejected-envelopes	
submitted-envelopes	Returns only submitted expense reports (envelopes) or objects associated with a submitted expense report (envelope).
nonreimbursed-envelopes	Returns envelopes that have a non-zero balance attribute.

Filter Value	Response
	Returns only expense reports (envelopes) that have a non-zero balance <i>or</i> objects associated with expense reports (envelopes) that have a non-zero balance.
reimbursable-envelope	Returns only reimbursable expense reports (envelopes) <i>or</i> objects associated with a reimbursable expense report (envelope).
open-slips	Returns only open charges (slips) <i>or</i> objects associated with an open charge (slip).
approved-slips	Returns only approved charges (slips) <i>or</i> objects associated with an approved charge (slip).
open-timesheets	Returns only open timesheets <i>or</i> objects associated with an open timesheet.
approved-timesheets	Returns only approved timesheets <i>or</i> objects associated with an approved timesheet.
rejected-timesheets	Returns only rejected timesheets <i>or</i> objects associated with a rejected timesheet.
submitted-timesheets	Returns only submitted timesheets <i>or</i> objects associated with a submitted timesheet.
approved-revenue-recognition-transactions	Returns only revenue recognition transactions belonging to approved revenue containers.
not-exported	Returns only objects that have not been marked as exported. Include an <code>ImportExport</code> argument object to specify the application the objects should be exported to. See Read Not Exported Charges with all Method — C# and Read Not Exported Charges with all Method — Java .
newer-than	Date filter. Returns only objects that have a value in the updated field (or in the field specified using the field attribute) that is newer than the date specified. See Date Filters Usage .
older-than	Date filter. Returns only objects that have a value in the updated field (or in the field specified using the field attribute) that is older-than the date specified. See Date Filters Usage .
date-equal-to	Date filter. Returns only objects that have a value in the updated field (or in the field specified using the field attribute) that is equal to the date specified. See Date Filters Usage .
date-not-equal-to	Date filter. Returns only objects that have a value in the updated field (or in the field specified using the field attribute) that is not equal to the date specified. See Date Filters Usage .

Date Filters Usage

You can use date filters listed in the above table to specify search criteria when retrieving a list of objects using the [Read](#) (XML API) or [read\(\)](#) (SOAP API) command.

By default, date filters compare the updated field to the date specified. Use the `field` attribute in conjunction with the `filter` attribute to set date filters and compare any date fields other than updated.

To apply multiple date filters or search criteria:

- Use a comma-separated list to specify the date fields for multiple date filters in the same order as the filters those fields apply to.



Important: Make sure there are no spaces between the commas and the values.

- Include `Date` argument objects in the same order as the date filters those arguments apply to as part of the objects collection in your request.

Sample codes:

- XML API – [Read with all Method and Date Filters](#)
- SOAP API – [Read Not Exported Expense Reports with all Method and Date Filter — C#](#) and [Read with all Method and Date Filters — Java](#).

Pagination

The maximum number of object you can read in a single API request depends on your account configuration. For OpenAir production and sandbox accounts, the maximum number of object you can read per request is typically 1,000. It may be set to a lower value but cannot exceed 1,000.

You must set the `limit` attribute when retrieving a list of objects using the `Read` (XML API) or `read()` (SOAP API) command to control the number of objects returned. The OpenAir XML API or SOAP API returns error code 605 if the `limit` attribute is not set or exceeds the maximum number of read objects configured for your account. You set the `limit` attribute to a lower value to optimize performance.

The `limit` attribute lets you read objects in batches by setting a batch size, and an optional offset. Use the `limit` attribute and increment the offset to control the response pagination and read objects in consecutive pages of results.

The `limit` attribute value must have one of the following formats:

- One integer number between 1 and 1000 – The batch size or maximum number of objects to be returned.
- Two integer numbers separated by a comma `<offset>, <batch_size>`, where:
 - `<offset>` – A cursor for use in pagination. The response will skip the number of matching objects (or rows) specified using the offset parameter and return the page of objects starting with the next object (or row) in the list. The `<offset>` must be a positive number.
 - `<batch_size>` – A limit on the length of the page. The `<batch_size>` must be between 1 and 1000.

To read objects in consecutive batches, you should set the `limit` attribute in the following format `<offset>, <batch_size>` where `<max_number_of_objects>` is the same for each request and `<offset>` is incremented by `<batch_size>` starting from 0 with each request, until the request returns no objects or less than `<batch_size>` objects. For example, you can use the following values to read object in pages of 500: "0,500", "500,500", "1000,500", "1500,500", and so on until the response is empty or contains less than 500 objects.

Sorting

You can use the `order` attribute when retrieving a list of objects using the `Read` (XML API) or `read()` (SOAP API) command to sort the returned list by the specified property in ascending or descending order.

The order attribute value must have one of the following format:

- `<property>, <direction>` where:
 - `<property>` is the name of the object property used to sort the objects.
 - `<direction>` is the sort direction which can be either of the following:
 - `asc` – ascending order (lowest values presented first).
 - `desc` – descending order (highest values presented first).
- `<property>` – An ascending sort order is used if the sort direction is not specified.
- You can specify an ascending sort order or descending sort order using a + sign (ascending) or - sign (descending) before the attribute name. An ascending sort order is used if the sort order is not specified.

Examples:

- The Sorting feature supports single level sorting only. Entering a comma-separated list of attributes will return an error. A secondary sorting level by internal ID is applied by default. When reading a list of receipts sorted by date, for example, if two receipts have the same receipt date, the response lists the receipt with the lower internal ID (id) first.
- The Sorting feature supports standard indexed fields only. See the help topic [Sortable fields](#).
- Sorting by a hidden sortable field is allowed. The returned list of resources is sorted in the order specified even if the sortable field is hidden due to form permissions, form permission rules or other account configuration setting.

Examples:

- If you set the order attribute to `created, desc`, the response lists objects and sort them in descending order of created date (most recent first).
- If you set the order attribute to `name, asc`, the response lists objects and sort them in ascending order of name.
- If you set the order attribute to `name`, the response lists objects and sort them in ascending order of name.

Adding, Updating and Upserting Objects

Use the [Add](#) (XML API) or [add\(\)](#) (SOAP API) command to **add** objects of a supported type except [User](#) objects.

Use the [CreateUser](#) (XML API) or [createUser\(\)](#) (SOAP API) command to **add User** objects.

Use the [Modify](#) (XML API), [ModifyOnCondition](#) (XML API) or [modify\(\)](#) (SOAP API) command to **update** objects of a supported type including [User](#) objects.

Use the [Add](#) (XML API) or [upsert\(\)](#) (SOAP API) command to **upsert** objects of a supported type.

Note: Upserts use a lookup field to determine whether it adds or updates an object:

- If the lookup field is not matched in any existing objects, a new object is added.
- If the lookup field is matched one time, the existing object is updated.

These commands pass the following parameters and arguments:

- **type** — (XML API only) You must specify the type of object you want to add, update or upsert. For information about supported object types, see [List of Supported Business Object Types](#).
- **Attributes** — You can use attributes to set custom field values or to look up objects with a matching object property value when adding, updating and upserting objects using the OpenAir XML API or SOAP API. See [Add, Update and Upsert Attributes](#).
- **Objects** — All command must include the object(s) to be added, updated or upserted as argument. The [CreateUser](#) (XML API) or [createUser\(\)](#) (SOAP API) command also requires the [Company](#) object as argument. See the command descriptions for details.

In the argument object properties, you can use a related object lookup to set the internal ID of any related object indirectly when you add, update or upsert an object. The API looks up the internal ID of the related object using the external ID [externalid] or name [name] as foreign key and set the internal ID if a unique object matching the foreign key can be found. See [Related Object Lookup Using the XML API](#) and [Related Object Lookup Using the SOAP API](#).

Add, Update and Upsert Attributes

You can use attributes to set or look up custom field values when adding, updating and upserting objects using the OpenAir XML API or SOAP API.

The following table lists the supported attributes, and for each attribute, the XML API commands and SOAP API commands that support the attribute if it can only be used for specific operations.

Attribute	XML API Commands	SOAP API Commands	Usage
enable_custom	Add CreateUser Modify ModifyOnCondition	—	<p>Set the enable_custom attribute to 1 to include custom field values in the argument object properties inline with standard object properties.</p> <p>Custom field values can be included in the argument object properties by default when using OpenAir SOAP API. Use the account-specific WSDL.</p>

Attribute	XML API Commands	SOAP API Commands	Usage
lookup	Add CreateUser	upsert()	<p>Use the lookup to upsert an object and designate the lookup field. The lookup field determines whether it adds or updates an object:</p> <ul style="list-style-type: none"> ■ If the lookup field is not matched in any existing objects, a new object is added. ■ If the lookup field is matched one time, the existing object is updated. <p>Sample codes:</p> <ul style="list-style-type: none"> ■ XML API - Sample Code — Upsert ■ SOAP API - Sample Code — C# and Sample Code — Java
lookup_custom	—	modify()	<p>Set the lookup_custom attribute to the custom field name with a suffix of two underscores immediately followed by a lowercase c character to look up and modify objects matching a custom field value.</p> <p>Sample code: Update Using Custom Field as Foreign Key Lookup — C#</p>
method	Modify ModifyOnCondition	modify()	<p>Set the method attribute to custom equal to to update a custom field value.</p> <div style="border: 1px solid #00a0e3; padding: 5px; margin: 10px 0;"> <p> Note: You should modify custom field values inline with standard object properties in the argument object instead of using the custom equal to method. See the description for the enable_custom attribute in this table.</p> </div> <p>SOAP sample codes: Modify Custom Field Values with custom equal to Method — C# and Modify Custom Field Values with custom equal to Method — Java</p>
update_custom	—	modify() upsert()	<p>Set the update_custom attribute to 1 to set custom field values inline when modifying objects.</p> <p>SOAP sample code: Update Custom Field Value as Object Property — C#</p>

Related Object Lookup Using the XML API

You can use a related object lookup to set the internal ID of any related object indirectly when you add, update or upsert an object if you know the external ID or the name of this related object. To do so:

1. Add the attribute external or name to each object property referencing the related object by internal ID. The attribute name determines which foreign key is used for looking up the related object.

2. For each object property referencing related object(s) by internal ID:
 - a. Set the attribute value to the related object type.
 - b. Pass the external ID or name instead of the internal ID as property value.

To look up the internal ID of a related object using an external ID as foreign key with the XML API, use the following syntax.

```

1 <Modify type="ObjectType">
2   <ObjectType>
3     <id>internalID</id>
4     <related_objectid external="RelatedObjectType">relatedObjectExternalID</related_objectid>
5   </ObjectType>
6 </Modify>

```

To look up the internal ID of a related object using an external ID as foreign key with the XML API, use the following syntax.

```

1 <Modify type="ObjectType">
2   <ObjectType>
3     <id>internalID</id>
4     <related_objectid name="RelatedObjectType">relatedObjectName</related_objectid>
5   </ObjectType>
6 </Modify>

```

Sample Codes

The following example updates the customer associated with a project and looks up the customer with external ID 805-25664 instead of passing the customer internal ID.

```

1 <Modify type="Project">
2   <Project>
3     <id>200</id>
4     <customerid external="Customer">805-25664</customerid>
5   </Project>
6 </Modify>

```

The following example updates the cost center associated with a service (category) and looks up the cost center with name Maintenance instead of passing the cost center internal ID.

```

1 <Modify type="Category">
2   <Category>
3     <id>200</id>
4     <cost_centerid name="Costcenter">Maintenance</cost_centerid>
5   </Category>
6 </Modify>

```

Related Object Lookup Using the SOAP API

You can use a related object lookup to set the internal ID of any related object indirectly when you add, update or upsert an object if you know the external ID or the name of this related object. To do so:

1. Use the following argument object property.

Name	Type	Description
attributes	oaFieldAttribute[]	A collection of <code>oaFieldAttribute</code> objects.

Note: Most business object types except those used as substructures such as Address, Date, and Module support the attributes property.

2. Create a collection of `oaFieldAttribute` objects.
3. For each object property referencing related object(s) by internal ID you want to set indirectly using a lookup.
 - a. Add one object to the collection of `oaFieldAttribute` objects.
 - b. Pass the external ID or name instead of the internal ID, or a comma-separated list of external IDs or names instead of Internal IDs as property value.

Sample codes: [Update Using External ID as Foreign Key Lookup — C#](#) and [Modify with Foreign Key Lookup — Java](#).

oaFieldAttribute

The OpenAir SOAP API uses a `oaFieldAttribute` object as a substructure to define a related object lookup.

The `oaFieldAttribute` object has the following properties:

Property	Description
name	<p>Specifies the foreign key.</p> <ul style="list-style-type: none"> ■ Set name to external to use the object's external ID [externalid] as foreign key. ■ Set name to name to use the object's name [name] as foreign key.
value	<p>Specifies the object property referencing related object(s) by internal ID, the related object type, an optional flag indicating whether object property value referencing related object(s) by internal ID is a comma-separated list.</p> <p>Format (single internal ID):</p> <pre><i>related_objectid:RelatedObjectType</i></pre> <p>Format (multiple internal IDs):</p> <pre><i>related_objectids:RelatedObjectType:1</i></pre> <p>where <i>related_objectid</i> or <i>related_objectids</i> is the object property referencing related object(s) by internal ID, <i>RelatedObjectType</i> is the related object type.</p>

Deleting Objects

Use the [Delete](#) (XML API) or [delete\(\)](#) (SOAP API) command to **delete** objects.

A delete command passes the following parameters and arguments:

- **type** — (XML API only) You must specify the type of object you want to delete. For information about supported object types, see [List of Supported Business Object Types](#).
- **Objects** — All command must include the object(s) to be deleted as argument. Object properties must include the object internal ID [*id*].

Approval-Related Operations

Use the [Submit](#) (XML API) or [submit\(\)](#) (SOAP API) command to **submit** transactions of a supported type for approval.

Use the [Approve](#) (XML API) or [approve\(\)](#) (SOAP API) command to **approve** transactions that were submitted for approval.

Use the [Reject](#) (XML API) or [reject\(\)](#) (SOAP API) command to **reject** transactions that were submitted for approval.

Use the [Unapprove](#) (XML API) or [unapprove\(\)](#) (SOAP API) command to **unapprove** transactions that were previously approved.

A submit, approve, reject or unapprove command passes the following parameters and arguments:

- **type** — (XML API only) You must specify the type of object you want to submit, approve, reject or unapprove. For information about supported object types, see [Object Types Supporting Approval-Related Operations](#).
- **Objects** — All command must include the object(s) to be submitted, approved, rejected or unapproved as argument and an approval action information [[Approval](#)] object. Object properties must include the object internal ID [`id`].
- **Attributes** — You can use the following attribute to submit the transaction for approval even if there are warnings concerning the transaction.

Attribute	XML API Commands	SOAP API Commands	Usage
submit_warning	Submit	submit()	Set the submit_warning attribute to 1 to submit the transaction for approval even if there are some warnings concerning the transaction.

Object Types Supporting Approval-Related Operations

The following table lists the object types for which the OpenAir XML API and SOAP API support approval-related operations.

 **Note:** Approval routing is available for other transaction object types in the OpenAir UI. Some transaction approvals are available as standard if the transaction type is available, other transaction approvals are optional. For example, account administrators optional transaction approvals for invoices in the OpenAir UI (Administration > Global Settings > Organization > Approval Options). See the help topic [Approval Routing](#).

Transaction Type	Object Type	Notes	Submit	Approve	Reject	Unapprove
Booking	Booking	Mutually exclusive with booking request approvals	✓	✓	✓	✓
Expense report	Envelope	—	✓	✓	✓	✓

Transaction Type	Object Type	Notes	Submit	Approve	Reject	Unapprove
Invoice Credit invoice Rebill invoice	Invoice	Optional	✓	✓	✓	✓
Receipt	Ticket	Reject operation only. Other operations are supported indirectly as part of an Envelope .	✓	✓	✓	✓
Time entry	Task	Reject operation only. Other operations are supported indirectly as part of an Timesheet .	✓	✓	✓	✓
Time off request	Schedulerequest	—	—	—	—	✓
Timesheet	Timesheet	—	✓	✓	✓	✓

Approval

An Approval object provides additional information, typically recipients to copy in the system notification and notes for the recipients when submitting, approving, rejecting or unapproving a transaction.

Note: To read the approval table, use the [ApprovalLine](#) object.

—	XML	SOAP
Object	Approval	oaApproval

The approval action information object has the following properties:

XML / SOAP	Description
cc	Email cc field
notes	Notes

Utility Operations

The following utility operations are available:

- You can use the [MakeURL](#) (XML API) or [makeURL\(\)](#) (SOAP API) command to obtain a URL for a specific page in the active OpenAir UI session for the authenticated user. See [OpenAir Pages Supported by the Make URL Operation](#).
- You can use the [Report](#) (XML API) command to run a report and email a timesheet [[Timesheet](#)], expense report [[Envelope](#)] or saved report [[Report](#)] as PDF.
- You can use the [Time](#) (XML API) or [servertime\(\)](#) (SOAP API) command to retrieve the current system timestamp from the OpenAir servers. No attributes or arguments are required.
- You can use the [Version](#) (XML API) or [version\(\)](#) (SOAP API) command to retrieve the version of a thin client application supported by OpenAir. A version command passes the name [[name](#)] and version number [[number](#)] of the client application as arguments.
- You can use the [Whoami](#) (XML API) or [whoami\(\)](#) (SOAP API) command to retrieve information about the authenticated user. No attributes or arguments are required.

OpenAir Pages Supported by the Make URL Operation

You can use the [MakeURL](#) (XML API) or [makeURL\(\)](#) (SOAP API) command to obtain a URL for a specific page in the active OpenAir UI session for the authenticated user. A Make URL passes the arguments page, app and arg to identify the page.

The following table lists the supported pages in the OpenAir UI (identified by a navigation path) and the corresponding page, app and arg combinations.

 **Note:** The information in the following table is subject to change without prior notice.

Navigation Path	page	app	arg
Application landing page for the authenticated user.	default-url	<ul style="list-style-type: none"> ■ km – Workspaces ■ ma – Administration, Home, Reports ■ pm – Projects ■ po – Purchases ■ rm – Resources ■ ta – Timesheets ■ te – Expenses ■ tb – Invoices 	—
Administration > Global Settings	company-settings	ma	—
Administration > Global Settings > Organization > Currencies	currency-rates	ma	—
Administration > Global Settings > Account > Integration: Import/Export	import-export	ma	—

Navigation Path	page	app	arg
Administration > Global Settings > Custom Fields	custom-fields	ma	—
Reports > last page accessed	list-reports	ma	—
Administration > Global Settings > Customers > Customers	list-customers	ma	—
Projects > Projects	list-projects	pm	—
Opportunities > Prospects	list-prospects	om	—
Resources > Resources	list-resources	rm	—
Timesheets > Timesheets > Open	list-timesheets	ta	—
Timesheets > Create Timesheet	create-timesheet	ta	—
Invoices > Charges	list-timebills	tb	—
Invoices > Invoices > Open	list-invoices	tb	—
Invoices > Invoices > Create Invoice	create-invoice	tb	—
Expenses > Envelopes > Receipts	list-envelope-receipts	te	SOAP API: <pre>oaEnvelope envelope = new oaEnvelope(); envelope.id = internalID;</pre> XML API: <pre><Envelope> <id>internalID</id> </Envelope></pre>
Expenses > Expense Reports > Open	list-envelopes	te	—
Expenses > Expense Reports > Create Envelope	create-envelope	te	—
Expenses > Expense Reports > Create Receipt	create-envelope-receipt	te	—
Home > Dashboard	dashboard	ma	—
Purchases > Purchase Requests	list-purchase-requests	po	—
Resources > Quick Search	quick-search-resources	rm	—

Navigation Path	page	app	arg
Resources > Custom Search	custom-search-resources	rm	—
Invoices > Invoices > [Select an invoice]	view-invoice	tb	<p>SOAP API:</p> <pre>oaInvoice invoice = new oaInvoice(); invoice.id = internalID;</pre> <p>XML API:</p> <pre><Invoice> <id>internalID</id> </Invoice></pre>
Projects > Projects > [Select a project] > Dashboard	dashboard-project	pm	<p>SOAP API:</p> <pre>oaProject project = new oaProject(); project.id = internalID;</pre> <p>XML API:</p> <pre><Project> <id>internalID</id> </Project></pre>
Timesheets > Timesheets > [Select a timesheet] > Edit	grid-timesheet	ta	<p>SOAP API:</p> <pre>oaTimesheet timesheet = new oaTimesheet(); timesheet.id = internalID;</pre> <p>XML API:</p> <pre><Timesheet> <id>internalID</id> </Timesheet></pre>
Timesheets > Timesheets > [Select a timesheet] > Report	report-timesheet	ta	<p>SOAP API:</p> <pre>oaTimesheet timesheet = new oaTimesheet(); timesheet.id = internalID;</pre> <p>XML API:</p> <pre><Timesheet> <id>internalID</id> </Timesheet></pre>
Home > Calendar	calendar-user	ma	<ul style="list-style-type: none"> ■ period_view = <daily weekly monthly> (monthly view by default) ■ user_view = <user internal ID> (authenticated user by default) ■ department_view = <department internal ID> ■ start_date = <yyyy-mm-dd> (today's date by default)

Navigation Path	page	app	arg
			<ul style="list-style-type: none">transaction = [booking, schedule_request, assignment, workschedule] (all transaction types by default)

Handling Errors

OpenAir API returns error data when a request fails. Your client application needs to identify and handle these errors.

OpenAir XML API uses the status attribute for the response element and each command response element to indicate success or failure.

- The response element includes a status attribute with the value 1 if the XML request is badly formed or could not be parsed. It has no attributes otherwise.

```
<response status="1">unclosed token at line 1, column 322</response>
```

- If the request was well formed, each command response element includes a status attribute. The status attribute value is 0 if the operation succeeded or a different value (error code) if the operation failed.

In some cases of failed operation, the command response element may include additional error information using the errors child element `<errors>`.

```
1 <response>
2   <Add status="1106">
3     <errors>Some error description</errors>
4   </Add>
5 </response>
```

OpenAir SOAP API uses two types of errors:

- **SOAP Faults** — OpenAir SOAP API uses the standard SOAP Fault element in case of incorrect usage, badly formed SOAP messages, failed authentication, or similar problems. The SOAP Fault includes a `faultcode` element with the error code, a `faultstring` element with a brief description of the error, and a `detail` element with the error code.

The following example shows the Fault element returned when sending an expired or invalid OAuth 2.0 access token [access_token] or Session ID [sessionId] in a session header [SessionHeader] with your request.

```
1 <SOAP-ENV:Fault>
2   <faultcode>SOAP-ENV:9</faultcode>
3   <faultstring>Logged out</faultstring>
4   <faultactor>http://www.soaplite.com/custom</faultactor>
5   <detail>
6     <Error xsi:type="namespace7:Error">
7       <code xsi:type="xsd:int">9</code>
8     </Error>
9   </detail>
10 </SOAP-ENV:Fault>
```

- **SOAP Command Errors** — The response objects for each command have an `errors` property. If errors occur because of a problem specific to the operation, the `errors` property is an array of one or more `oaError` objects (see [Error](#)). The `oaError` object contains the error code and may also contain the error description [text].

To obtain more information about the error returned, you can use the [Read](#) (XML) or [read\(\)](#) (SOAP) command to retrieve the [Error](#) object which includes a short description and additional information about the error.

A list of error codes and their descriptions is also included in this guide for reference. See [Error Codes](#).

Error

An error is a message that OpenAir API returns when a request fails.

—	XML	SOAP	REST	Database table
Object	Error	oaError	—	—
Supported Commands	Read (all, equal to)	read()	—	—

Note: The Error object supports the `all` and `equal` to read methods only.

The error object include the following properties:

XML / SOAP	Database	Description
code	—	Error code returned by the API.
comment	—	Additional comments.
text	—	Text of the error.

Error Codes

The OpenAir XML API and SOAP API return error codes to indicate problems with your request or specific to a query or action on a particular object. The following tables list the error codes, the commands or methods and the type of objects that can result in the error, and the error descriptions.

[0 - 199](#) | [200 - 399](#) | [400 - 599](#) | [600 - 799](#) | [800 - 899](#) | [900 - 999](#) | [1000 - 1199](#) | [1200 - 1999](#) | [2000 - 2999](#)

0 - 199

Error Code	Command	Object Type	Short Message	More Information
0	—	—	Success	The operation completed successfully.
1	—	—	Unknown Error	An unknown error occurred. Try again and if the problem persists, contact OpenAir Customer Support.
2	—	—	not logged in	The operation requires authentication but the authentication command was not successful or was omitted in the API request.
3	—	—	too many arguments	The API request included too many arguments for one of the commands. Verify the commands and the arguments for each command included in your request.
4	—	—	too few arguments	The API request did not include enough arguments for one of the commands. Verify the commands and the arguments for each command included in your request.

Error Code	Command	Object Type	Short Message	More Information
5	—	—	Unknown Command	One of the commands in your API request uses an unknown method. Verify the commands and ensure only supported methods are used in your request.
6	—	—	Access from an invalid URL	The URL used to access OpenAir API is not valid. Verify the URL.
7	—	—	Invalid OffLine version	The version of OpenAir OffLine is not supported. Install the latest version of the OpenAir OffLine add-on on your computer.
8	—	—	Failure + Dynamic Message	The operation was unsuccessful. Review the Error record included in the response. This code is reserved for dynamically generated error codes.
9	—	—	Logged out	The sessionId (session-based authentication) or accessToken (OAuth 2.0 access token authorization) is not valid or has expired. Use the login method to obtain a sessionId if using session-based authentication. Refresh the OAuth2.0 access token, if using OAuth 2.0 access token authorization.
10	—	—	Invalid parameters	Invalid parameters used. Verify your API request.

200 – 399

Error Code	Command	Object Type	Short Message	More Information
201	CreateUser createUser()	User	invalid company	The account with the Company ID [nickname] specified does not exist.
202	CreateUser createUser()	User	duplicate user nick	There is already a user with the same User ID [nickname]. Use a different value.
203	CreateUser createUser()	User	too few arguments	Both a company object and a user object must be specified.
204	CreateUser createUser()	User	Namespace error	Users must be created in the same namespace as the company.
205	CreateUser createUser()	User	Workschedule error	Invalid account workschedule specified
206	CreateUser createUser()	User	Role error	Invalid role specified
303	CreateUser createUser()	User	please pick a different password	The password specified does not meet the minimum password policy requirements configured for the OpenAir account. Review the password policy and use a password that meets the policy requirements.

Error Code	Command	Object Type	Short Message	More Information
304	CreateAccount	—	Not enabled	Reserved usage. The CreateAccount method is not available to OpenAir customers.
305	CreateUser createUser()	User	Not enabled to edit password	The password cannot be modified or the user is set up to use SAML single sign-on for authentication.

400 – 599

Error Code	Command	Object Type	Short Message	More Information
401	Authentication	—	Auth failed	Access denied. The combination of company ID [company], user ID [user] and password [password] specified is not valid. This is a generic error message used for all authentication issues other than those specified in this table.
409	Authentication	—	Account Canceled	The account with company ID company was canceled and cannot be accessed.
411	Authentication	—	Account conflict, contact customer service	There is a problem with the account. Contact OpenAir Customer Support.
413	Authentication	—	Account not privileged to access API	API access is not permitted for this OpenAir account or user.
414	Authentication	—	Temporarily unavailable	The service is temporarily unavailable. Try again in a few minutes and if the problem persists, contact OpenAir Customer Support.
415	Authentication	—	Account archived	The account with company ID company was archived and cannot be accessed.
417	Authentication	—	Restricted IP address	Access is not allowed from the client IP address. The IP address is not in the IP address allowlist for the authenticating or authenticated user. See the help topic IP Restriction .
418	Authentication	—	Invalid uid session	The uid specified is not valid. Repeat authentication.
419	Authentication	—	Authentication failed, please retry	Unexpected authentication failure. Try again later. If you are using session-based authentication or OAuth2.0 access token authorization, renew the sessionId or accessToken.
420	—	—	Authentication failed	SAML single sign-on authentication error. If the problem persists, contact the identity provider services vendor.

Error Code	Command	Object Type	Short Message	More Information
421	—	—	Account misconfiguration or invalid assertion	SAML single sign-on authentication error. Verify your identity provider configuration and the SAML integration configuration in OpenAir. If the problem persists, contact the identity provider services vendor.
423	—	—	No permissions to read ServerStatus data	Reserved usage. Server status information is not accessible to OpenAir customers.
424	Add, Modify, Delete add(), modify(), delete()	— <ul style="list-style-type: none"> ■ Envelope ■ Timesheet 	No permissions to modify data	The authenticated user is not an account administrator or the owner of the object and the account is configured to prevent the creation, modification or deletion of an object owned by a different user.
425	Add, Modify, Delete add(), modify(), delete()	—	Functionality not available	The functionality is not available for your OpenAir account. For example, OpenAir API returns this error if you attempt to modify proxy user information and your account is not configured to allow proxy modification using OpenAir API, or if you attempt to add or modify project task estimate objects and the Hours Remaining on Tasks feature is not enabled for your account.
426	Authentication	—	You must use account-specific domain	All connections to your OpenAir account must use the account-specific domain <account-domain>. Use the following URLs for OpenAir API requests. For more information, see Getting Started with OpenAir XML API and SOAP API . <ul style="list-style-type: none"> ■ XML API — https://<account-domain>/api.pl ■ SOAP API — https://<account-domain>/soap
501	—	—	API authentication required	API access must first be authenticated.
503	—	—	Invalid or missing key attribute	A valid API key must be specified.
504	—	—	Invalid or missing namespace attribute	A valid API namespace must be specified.
505	—	—	The namespace and key do not match	The API namespace and API key combination is not valid.
506	—	—	Authentication key disabled	This API key is disabled. For more information, contact OpenAir Customer Support
555	—	—	You have exceeded the limit set for the account for input objects	The maximum number of object you can add or modify in a single request is 1,000. Above this number object must be processed in batches.

Error Code	Command	Object Type	Short Message	More Information
556	—	—	API rate limit exceeded	The maximum number of requests allowed for your OpenAir account in a 24-hour window or in a 60-seconds window has been reached. Try again later.

600 – 799

Error Code	Command	Object Type	Short Message	More Information
601	Add , CreateUser , Read , Modify , ModifyOnCondition , Delete , Submit , Approve , Reject , Unapprove add() , createUser() , read() , modify() , upsert() , delete() , submit() , approve() , reject() , unapprove()	All	Invalid id/code	Could not find an object matching the internal ID or the criteria specified in your request.
602	Read read()	All	Invalid field	One or more of the property names listed for inclusion in the response are not valid. Verify the fields listed in your API request.
603	Read read()	All	Invalid type or method	Either the object type or the read method specified is not valid.
604	Add add()	Attachment	Attachment size exceeds space available	The OpenAir account does not have enough storage space available to add the attachment. Contact your account administrator to increase the storage space for your account or free some of the account storage space.
605	Read read()	All	Limit clause must be specified and be less than the account limit for output data	The maximum number of object you can read in a single request is 1,000. Use the <code>limit</code> attribute to process objects in batches.
606	Read read()	RevenueProjection	Projections are running, please try again in a few minutes	Revenue projection objects can only be read using the OpenAir API if there are no charge projection runs in progress. This is because the data may be incomplete until the charge projection job completes.
701	Delete delete()	All	Cannot delete, failed dependency check	All dependencies (objects of the same or other types directly or indirectly referencing this object by internal ID) must be

Error Code	Command	Object Type	Short Message	More Information
				deleted before this object can be deleted.
702	Delete delete()	Preference	Invalid note	The preference could not be deleted.

800 – 899

Error Code	Command	Object Type	Short Message	More Information
800	CreateUser, Modify createUser(), modify()	User	Synchronization failed	Failed to propagate the changes to the identity service. Please try again later.
801	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Agreement ■ Booking ■ Contact ■ Customerpo ■ Invoice ■ Issue ■ Payment ■ Project ■ ProjectBudget Group ■ ResourceRequest 	Not a valid Customer ID	The customer with internal ID <code>customerid</code> does not exist or is marked as deleted. Verify the customer internal ID.
802	Add, Modify add(), modify()	Envelope	This Envelope number is already taken	An expense report with tracking number <code>number</code> already exists. Use a different tracking number, or do not specify a tracking number to use auto-numbering.
803	Add, CreateUser, Modify add(), createUser(), modify()	<ul style="list-style-type: none"> ■ Company ■ HierarchyNode ■ User 	This user does not have permission to modify the record	The authenticated user must be an account administrator or have sufficient role and access permissions to modify this record.
804	Add, Modify add(), modify()	Item	Not a valid Item type	The expense item type must be either R (regular) or M (mileage).
805	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Ticket ■ ResourceRequest ■ ResourceRequest Queue 	Reference number in use	<p>A receipt with tracking number <code>reference_number</code> already exists within the associated expense report. Use a different tracking number.</p> <p>A resource request (queue) with tracking number <code>number</code> already exists. Use a different tracking number.</p>
806	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Ticket ■ Task 	Already accepted by signer	The receipt or time entry was accepted by a signer and cannot be modified.

Error Code	Command	Object Type	Short Message	More Information
807	Add, Modify add(), modify()	—	Invalid payment type	The payment type passed is not valid (possibly inactive, or deleted).
808	Add, Modify add(), modify()	Preference	Invalid preference	To modify a preference you must set a valid name and group_name.
809	Add, Modify add(), modify()	Task	Invalid Timesheet	The timesheet with internal ID timesheetid does not exist or is marked as deleted. Verify the timesheet internal ID.
810	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Projecttask ■ Projecttaskassign 	Invalid index	The index you specified does not exist in that table.
811	Add, Modify add(), modify()	Projecttask	Invalid predecessor	One or more internal IDs in the predecessors list could not be found.
812	Add, Modify add(), modify()	Projecttask	Invalid parentid	The internal ID of the immediate ancestor [parentid] is not valid. Verify the internal ID for the immediate ancestor.
813	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Agreement_to_project ■ Booking ■ Customerpo_to_project ■ ExpensePolicy ■ Issue ■ Projectassign ■ ProjectAssignment Profile ■ ProjectBudget Group ■ ProjectPricing ■ Projecttask ■ Projecttaskassign ■ Purchase_item ■ Ticket ■ ResourceRequest ■ ResourceRequest Queue 	Invalid projectid	The project with internal ID projectid does not exist or is marked as deleted. Verify the project internal ID.
814	Add, Modify add(), modify()	Projecttask	duplicate id_number	This user-defined task number id_number is already used for another task in the same project.
815	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Booking ■ Issue ■ Projecttaskassign ■ ProjecttaskEstimate 	Projecttask does not exist	The project task with internal ID project_taskid does not exist or is marked as deleted. Verify the project task internal ID.

Error Code	Command	Object Type	Short Message	More Information
816	Add, CreateUser, Modify add(), createUser(), modify()	<ul style="list-style-type: none"> ■ Proxy ■ User 	User role/type does not exist	The role_id or type is not valid.
817	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Ticket ■ Reimbursement 	Invalid envelope	The expense report with internal ID envelopeid does not exist or is marked as deleted. Verify the expense report internal ID.
818	CreateUser, Modify createUser(), modify()	User	duplicate user nick	A user with the same nickname already exists. Use a different value for nickname.
819	Add, Modify add(), modify()	Slip	Slip cannot be deleted	This slip cannot be deleted because it is part of an invoice.
820	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Attachment ■ Envelope ■ Ticket ■ Reimbursement 	Envelope not open	The expense report cannot be modified because it is has been submitted for approval (no longer open).
821	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Attachment ■ ProjecttaskEstimate ■ Task ■ Timesheet 	Timesheet not open	<p>This error is returned under the following conditions:</p> <ul style="list-style-type: none"> ■ The timesheet cannot be modified because it has been submitted for approval. ■ The timesheet status is A (Approved) or X (Archived) and the account configuration does not allow the modification of approved and archived timesheets using the OpenAir API. ■ The timesheet is not in an open period and the authenticated user's role does not allow the modification of timesheets outside of open periods. ■ The timesheet status is S (Submitted), the authenticated user is the submitter, and the account configuration does not allow the modification of submitted timesheet by the timesheet owner ■ The timesheet has been exported and the account configuration does not allow the modification of exported timesheets. ■ The authenticated user is neither the timesheet owner nor an account administrator.
822	Add, Modify	Slip	Slip cannot be modified	The charge cannot be edited.

Error Code	Command	Object Type	Short Message	More Information
	add(), modify()			
823	Add, Modify add(), modify()	Slip	Slip, bad invoice id	The charge is already part of an invoice and cannot be moved to a different invoice.
824	Add, Modify add(), modify()	Customer	Must specify name or company	The customer or prospect must have a valid name or company value.
825	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Slip ■ Invoice ■ Payment 	Invalid invoice	The invoice with internal ID invoiceid or original_invoiceid does not exist. Verify the invoice internal ID.
826	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Actualcost ■ Agreement ■ Customerpo ■ Envelope ■ Fulfillment ■ ImportExport ■ Payment ■ Ticket ■ Reimbursement 	Date is required	A date must be specified, or in the case of the ImportExport object, either a last_import or last_export daet must be specified.
827	Add, Modify add(), modify()	Reimbursement	Reimbursements can only be applied after the envelope is approved	The status of the expense report with internal ID envelopeid must be A (Approved).
828	Add, Modify add(), modify()	Invoice	This Invoice number is already taken	This user-defined invoice number is already used for another invoice. Use a different value or do not specify a number to use auto-numbering.
829	Add, CreateUser, Modify add(), createUser(), modify()	<ul style="list-style-type: none"> ■ Actualcost ■ Booking ■ Entitytag ■ Issue ■ Leave_accrual_rule_to_user ■ Leave_accrual_transaction ■ LoadedCost ■ Projectassign ■ Projecttaskassign ■ Proxy ■ Resource Attachment ■ Resourceprofile ■ ResourceRequest ■ Revenue_recognition_rule ■ TargetUtilization ■ Task 	Not a valid user	The user with internal ID userid, ownerid, assigned_user or id does not exist or is not a valid user. Verify the user internal ID.

Error Code	Command	Object Type	Short Message	More Information
		<ul style="list-style-type: none"> ■ Timesheet ■ User 		
830	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Booking ■ ResourceRequest 	Not a valid booking type	booking_type_id is required and must be a valid booking type internal ID.
831	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ AccountingPeriod ■ Booking ■ Entitytag ■ Leave_accrual_rule_to_user ■ Leave_accrual_transaction ■ Project ■ ResourceRequest ■ ResourceRequest Queue ■ Timesheet ■ UserWorkschedule 	No startdate or enddate specified	A start date and an end date are both required.
832	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ AccountingPeriod ■ Booking ■ Entitytag ■ ResourceRequest ■ ResourceRequest Queue ■ Timesheet 	Illegal date range	The start date must be before the end date.
833	Add, Modify add(), modify()	Booking	Percentage not specified	A percentage must be specified if as_percentage is set.
834	Add, Modify add(), modify()	Booking	Hours not specified	hours must be specified if as_percentage is not set.
835	Add, Modify add(), modify()	Project	Only owner can edit this project	The authenticated user must be the project's owner to modify this project or project task.
836	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Project ■ Schedulerequest 	Not allowed to add entity	The authenticated user does not have sufficient permission to add a record.
837	Add, CreateUser, Modify add(), createUser(), modify()	<i>All objects with currency property</i>	Not a valid account currency	The currency must be a valid currency for the account as set in Administration > Account > Currencies > Multi-currency
838	Add, Modify add(), modify()	LoadedCost	Not allowed to have more than one current costs per user	Only one loaded cost can be the current loaded cost for the same user.
839	Add, Modify add(), modify()	Attachment	base64_data must be set to add an attachment	The base 64 encoded binary data of the attachment file [base64_data] must be set.

Error Code	Command	Object Type	Short Message	More Information
840	CreateUser, Modify createUser(), modify()	User	Not a valid primary filter set	A valid primary filter set [primary_filter_set] must be specified.
841	CreateUser, Modify createUser(), modify()	User	Invalid email	email must be specified.
842	Add, Modify add(), modify()	—	Invalid period	Period is a required field.
843	Add, Modify add(), modify()	—	Invalid timing	Timing is a required field.
844	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Leave_accrual_rule_to_user ■ Leave_accrual_transaction 	Invalid leave accrual rule	A leave_accrual_ruleid must be specified.
845	Add, Modify add(), modify()	Leave_accrual_transaction	Invalid task	A taskid must be specified.
846	Add, Modify add(), modify()	Purchase_item	Cannot create non-po purchase items	The Quick PO functionality must be enabled for your OpenAir account and you must have sufficient permission to add or modify a "quick PO" or "non-po purchase item".
847	Add, Modify add(), modify()	Purchase_item	Purchaseorderid must be blank	purchaseorderid must be empty to add or modify a "quick PO" or "non-po purchase item".
848	Add, Modify add(), modify()	Purchase_item	Only non_po purchase items can be added/modified	non_po must be set to 1 to add or modify a "quick PO" or "non-po purchase item".
849	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Entitytag ■ Envelope ■ Timesheet 	Another record with the same date range already exists	The period between start and end dates overlap with that of another record, and period overlaps are not allowed. Verify the start_date and end_date.
850	Add, Modify add(), modify()	Entitytag	Another record already exists as a default for this user and group	There is already one default entity tag record associated with the user with internal ID userid and the entity tag group with internal ID tag_group_id, and only one default entity tag record is allowed.
851	Add, Modify add(), modify()	Entitytag	Not a valid tag_group_attribute	The tag group attribute with internal ID tag_group_attributeid is not valid.
852	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Contact ■ Customer 	Duplicate external_id	Another record with the same external_id already exists.
853	Add, Modify	LoadedCost	Invalid Loaded Cost parameters	When current is set to 1, start and end dates must be empty, and

Error Code	Command	Object Type	Short Message	More Information
	<code>add(), modify()</code>			inversely if start and end dates are set, current must be set to 0.
854	Read <code>read()</code>	<i>All object types</i>	Too many records requested	— The number of records requested is larger than the maximum allowed. Limit the data returned using filter parameters.
856	Add, Modify <code>add(), modify()</code>	<ul style="list-style-type: none"> ■ AccountingPeriod ■ Leave_accrual_rule_to_user 	Date overlaps with existing record	The period between start and end dates overlap with that of another record. Verify the <code>start_date</code> and <code>end_date</code> .
857	Add, Modify <code>add(), modify()</code>	<ul style="list-style-type: none"> ■ Booking ■ Entitytag ■ ForexInput 	Date range exceeded maximum	The period between start and end dates exceeds the maximum length allowed.
858	Add, Modify <code>add(), modify()</code>	ForexInput	ForexInput error	Update error. Record could not be saved.
859	Add, Modify <code>add(), modify()</code>	<ul style="list-style-type: none"> ■ ExpensePolicy ■ Ticket 	Invalid customer id	The customer with internal ID <code>customerid</code> does not exist or is marked as deleted. Verify the customer internal ID.
860	Add, Modify <code>add(), modify()</code>	Entitytag	<code>default_for_entity</code> and start and end dates are mutually exclusive	Make sure either <code>default_for_entity</code> or start and end dates are set but not both.
861	Add, Modify <code>add(), modify()</code>	Slip	Invalid customer id	The <code>customerid</code> on the charge must be the same as the internal ID of the customer associated with the invoice this charge is part of (invoice with internal ID <code>invoiceid</code>).
862	Add, Modify <code>add(), modify()</code>	Slip	Invalid project id	The customer associated with the project with internal ID <code>projectid</code> must be the same as the customer associated with the invoice this charge is part of (invoice with internal ID <code>invoiceid</code>).
863	Add, Modify <code>add(), modify()</code>	Slip	Only one project per invoice	The invoice this charge is part of (invoice with internal ID <code>invoiceid</code>) is associated with a different project and the account is configured to allow only one project per invoice.
864	Add, CreateUser, Modify <code>add(), createUser(), modify()</code>	<ul style="list-style-type: none"> ■ User ■ UserWorkschedule 	Error while saving user workschedule	Error saving the user workschedule. Record could not be saved.
865	CreateUser, Modify <code>createUser(), modify()</code>	User	Invalid workdays	Workdays must be a comma-separated list of values, containing digits between 0 (Monday) and 6 (Sunday).
866	Add, CreateUser, Modify	<ul style="list-style-type: none"> ■ User ■ UserWorkschedule 	Invalid workdays or workshours	Both workdays and workhours must be set with valid values when setting a user work schedule.

Error Code	Command	Object Type	Short Message	More Information
	<code>add(), createUser(), modify()</code>			
867	CreateUser, Modify <code>createUser(), modify()</code>	User	Distinct workhours not enabled	workhours must include only one value and not a comma-separated list of values if your account configuration does not allow for different work hours per day.
868	Add, Modify <code>add(), modify()</code>	Hierarchy	Invalid type specified	The hierarchy must be associated with a record type. The record type can be either customer, project, or user.
869	Add, Modify <code>add(), modify()</code>	Hierarchy	Invalid value for primary_user_filter	primary_user_filterset can be set to 1 only for one hierarchy and only if type is set to project.
870	Add, Modify <code>add(), modify()</code>	Hierarchy	Invalid value for primary_dropdown_filter	primary_dropdown_filter can be set to 1 only for one hierarchy and only if type is set to project.
871	Read <code>read()</code>	—	Invalid number of read arguments supplied	[Read error] — The number of argument objects must be equal to the number of filter clauses.
872	Add, Modify <code>add(), modify()</code>	Actualcost	Invalid cost type	The cost type with internal ID <code>cost_typeid</code> does not exist or is marked as deleted. Verify the cost type internal ID.
873	Add, Modify <code>add(), modify()</code>	Actualcost	Invalid period	A period must be specified.
874	Add, Modify <code>add(), modify()</code>	Schedulerequest	Schedule request error	Error saving the record. Record could not be saved.
875	Add, Modify <code>add(), modify()</code>	Repeat	Repeat error	Error saving the record. Record could not be saved.
876	Add, Modify <code>add(), modify()</code>	Attachment	Attachment too small	Attachment size is too small. Returned when attempting to save an empty backup file for OpenAir Integration Manager.
877	Add, Modify <code>add(), modify()</code>	—	Invalid project group	The project assignment group with internal ID <code>project_groupid</code> does not exist or is marked as deleted. Verify the project assignment group internal ID.
878	Add, Modify <code>add(), modify()</code>	—	Purchaseorder not open	The purchase order cannot be modified because it is no longer open.
879	Add, Modify <code>add(), modify()</code>	Fulfillment	Invalid purchase order	The purchase order with internal ID <code>purchaseorder_id</code> does not exist or is marked as deleted. Verify the purchase order internal ID.

Error Code	Command	Object Type	Short Message	More Information
880	Add, Modify add(), modify()	Purchase_item	Invalid purchase item	Quick POs or "non-PO" purchase items must have a positive quantity.
881	Add, Modify add(), modify()	Attachment	Invalid attachment	The immediate ancestor attachment with internal ID parentid does not exist or is not in workspace with internal ID workspaceid.
882	Add, Modify add(), modify()	Slip	Invalid reference slip ID	The original charge with internal ID ref_slipid associated with this credit or rebill charge does not exist or is marked as deleted. Verify the original charge internal ID.
883	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Slip ■ Project ■ Revenue_recognition_transaction 	Invalid portfolio project ID	The project with internal ID portfolio_projectid does not exist, is marked as deleted, is not associated with the same customers as this record, or is not a portfolio a project. Verify the portfolio project internal ID.
884	Add, Modify add(), modify()	Project	Invalid portfolio link	A portfolio project cannot be subordinated to another portfolio project. portfolio_projectid cannot be set if is_portfolio_project is set to 1.
885	Add, Modify add(), modify()	Purchase_item	Invalid purchase item	The date must be specified.
886	Add, Modify add(), modify()	Ticket	Project task type mismatch	The project task type with internal ID project_task_typeid does not exist or is marked as deleted, or project_taskid is not specified.
887	Add, Modify add(), modify()	ProjectAssignment Profile	Wrong project assignment profile name	A project assignment profile with the same name and associated with the same project already exists.
888	Add, Modify add(), modify()	Task	Timesheet task invalid date	The time entry date is not within the project task assignment date range. Requirement set a the project task level when the Time Entries Match Task Assignments feature is in use.
889	Add, Modify add(), modify()	Attachment	Ticket cannot be modified	The receipt associated with this attachment cannot be edited.
890	Add, Modify add(), modify()	Attachment	User cannot be modified	The user associated with this attachment cannot be edited
891	Add, CreateUser, Modify add(), createUser(), modify()	<ul style="list-style-type: none"> ■ Attachment ■ NewsfeedMessage ■ User 	Invalid user	The associated user does not exist or is marked as deleted. Verify the user internal ID.
892	Add, Modify	Attachment	Invalid envelope	The associated expense report does not exist or is marked as

Error Code	Command	Object Type	Short Message	More Information
	<code>add(), modify()</code>			deleted. Verify the expense report internal ID.
893	Add, Modify <code>add(), modify()</code>	Attachment	Invalid receipt	The associated receipt does not exist or is marked as deleted. Verify the receipt internal ID.
894	Add, Modify <code>add(), modify()</code>	<ul style="list-style-type: none"> ■ Attachment ■ ProjecttaskEstimate 	Invalid timesheet	The associated timesheet does not exist or is marked as deleted. Verify the timesheet internal ID.
895	Add, Modify <code>add(), modify()</code>	Attachment	Invalid customerpo	The associated customer PO does not exist or is marked as deleted. Verify the customer PO internal ID.
896	Add, Modify <code>add(), modify()</code>	Attachment	Agreement cannot be modified	The agreement associated with this attachment cannot be edited.
897	Add, Modify <code>add(), modify()</code>	Attachment	Customerpo cannot be modified	The customer PO associated with this attachment cannot be edited.
898	Add, Modify <code>add(), modify()</code>	Attachment	Invalid workspace	The workspace with internal ID <code>workspaceid</code> does not exist or is marked as deleted. Verify the workspace internal ID.
899	Add, Modify <code>add(), modify()</code>	ExpensePolicyItem	Invalid expense policy	The expense policy with internal ID <code>expense_policyid</code> does not exist or is marked as deleted. Verify the expense policy internal ID.

900 – 999

Error Code	Command	Object Type	Short Message	More Information
900	Add, Modify <code>add(), modify()</code>	ExpensePolicyItem	Invalid item	The expense item with internal ID <code>itemid</code> does not exist or is marked as deleted. Verify the expense item internal ID.
901	MakeURL <code>makeURL ()</code>	—	The combination of <code>uid</code> , <code>app</code> , <code>arg</code> , and <code>page</code> is not valid	[makeURL error] — The values passed don't combine to represent a valid page, check the values and try again
902	MakeURL <code>makeURL ()</code>	—	A valid record could not be created from the <code>arg</code> passed	[makeURL error] — Check to make sure the required fields are being passed in the <code>arg</code> record
903	MakeURL <code>makeURL ()</code>	—	The user does not have access to that page	[makeURL error] — That combination of <code>app</code> , <code>arg</code> , and <code>page</code> is not valid for this user
904	Add, Modify <code>add(), modify()</code>	<ul style="list-style-type: none"> ■ Purchaseorder ■ Schedulerequest 	This number is already taken	The user-defined number <code>[number]</code> is already used for another record. Use a different value or do not specify a number to use auto-numbering.
905	Add, Modify <code>add(), modify()</code>	<ul style="list-style-type: none"> ■ Purchase_item ■ Schedulerequest_item 	Invalid purchaseorder	The purchase order with internal ID <code>purchaseorderid</code> or the time off request with internal ID <code>schedule_requestid</code> does not exist or is marked as deleted. Verify the internal ID.

Error Code	Command	Object Type	Short Message	More Information
906	Add, CreateUser, Modify add(), createUser(), modify()	<ul style="list-style-type: none"> ■ Category ■ Item ■ Project ■ Projectbillingrule ■ Projecttask ■ Ticket ■ Revenue_recognition_rule_amount ■ Task ■ Timetype ■ User 	Invalid Cost Center	The cost center with internal ID cost_centerid does not exist or is not an active cost center. Verify the internal ID.
907	Add, Modify add(), modify()	Contact	Invalid Contact	First name, Last name and email are required fields.
908	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Agreement ■ Attachment ■ Category ■ Category_<N> ■ CustomerLocation ■ Customerpo ■ Hierarchy ■ Leave_accrual_rule ■ Project ■ Projectgroup ■ Projecttask ■ UserWorkschedule 	Invalid Name	A name [name] must be specified.
909	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Customer ■ Project 	Invalid Contact	One of the associated contacts does not exist or is not associated with the customer. Verify all contact internal IDs.
910	Modify modify()	—	Lookup record not located	[Lookup error] — There is no record of the specified type matching one or more lookup field values.
911	Add, Modify add(), modify()	Task	No Timesheet specified	The internal ID of the associated timesheet [timesheetid] must be specified when modifying a time entry.
912	Add, Modify add(), modify()	Projectbillingrule	Invalid type specified	A valid type must be specified.
913	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Issue ■ Projectassign ■ Projecttaskassign 	Invalid project task specified for a project	<p>The project task with internal ID project_task_id must be part of the project with internal ID project_id (issue).</p> <p>A projectid must be specified (project assignment).</p> <p>A projecttaskid must be specified (project task assignment).</p>
914	Add, Modify add(), modify()	Resourceprofile	Invalid resourceprofile_type_id specified	A valid resourceprofile_typeid must be specified.
915	Add, Modify add(), modify()	Resourceprofile	Invalid type specified	Both type and resourceprofile_typeid must be specified and type must have the same value as type for the resource

Error Code	Command	Object Type	Short Message	More Information
				profile type record with internal ID resourceprofile_typeid.
916	Modify modify()	—	Table specified does not have external_id field	[Lookup error] — Either the record type (table) does not exist or the lookup field does not exist for the record type.
917	Add, Modify add(), modify()	Issue	This Issue number is already taken	The user-defined number [number] is already used for another record. Use a different value or do not specify a number to use auto-numbering.
918	Add, Modify add(), modify()	Issue	No description specified	The issue must have a description.
919	Add, Modify add(), modify()	IssueStage	Only one default issue stage is permitted	There is already one issue stage record with default_for_new set to 1, and only one default issue stage is allowed.
920	Add, Modify add(), modify()	RateCardItem	No rate card ID specified	A rate_cardid must be specified.
921	Add, Modify add(), modify()	RateCardItem	Job code in use for rate card	There is already one rate card item with the same job_codeid in the associated rate card.
922	Add, Modify add(), modify()	RateCardItem	Invalid job code specified	The job code with internal ID job_codeid does not exist or is not an active job code. Verify the internal ID.
923	Add, Modify add(), modify()	RateCardItem	Invalid rate card specified	The rate card with internal ID rate_cardid does not exist or is not an active rate card. Verify the internal ID.
924	Add, Modify add(), modify()	RateCardItem	No job code ID specified	A job_codeid must be specified.
925	Add, Modify add(), modify()	Project	Invalid template project ID specified	The project with internal ID template_project_id does not exist or is marked as deleted. Verify the internal ID.
926	CreateUser, Modify add(), modify()	User	Invalid value for user cost	cost must be set to a positive value.
927	CreateUser, Modify createUser(), modify()	User	Invalid user cost start date	cost_start_date must not be before any existing historical cost start date for the user.
928	Add, Modify add(), modify()	Workspaceuser	Invalid project group ID for workspace user	The project assignment group with internal ID projectgroupid does not exist or is marked as deleted. Verify the internal ID.
929	Add, Modify add(), modify()	Workspaceuser	Workspace user cannot contain both project group ID and user ID	Either projectgroupid or userid can be set but not both.
930	CreateUser, Modify createUser(), modify()	User	Generic flag cannot be modified	is_generic cannot be modified. A generic resource cannot become a named resource and inversely.
931	Add, Modify add(), modify()	Projectassign	Duplicate project assignment	The user with internal ID userid is already assigned to the project with internal ID projectid.

Error Code	Command	Object Type	Short Message	More Information
932	Add, Modify add(), modify()	Proxy	Only admin users may update proxies	The authenticated user must be an account administrator to add, modify or delete proxy information.
933	Add, Modify add(), modify()	Proxy	Not a valid proxy user	The user with internal ID proxy_id does not exist or is marked as deleted. Verify the internal ID.
934	Add, Modify add(), modify()	Project	Error while creating project from template	Error creating a project from template. Project could not be added.
935	CreateUser, Modify createUser(), modify()	User	Invalid user tag start date	tag_start_date must not be before any existing historical tag start date for the user.
936	Add, Modify add(), modify()	Projectgroup	Error while creating project group assignments	Error creating project assignment group. The project assignment group could not be added.
937	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Agreement_to_project ■ Attachment 	Invalid agreement ID specified	An agreementid (agreement – project link) or ownerid (attachment) must be specified. The agreement with internal ID agreementid or ownerid does not exist or is marked as deleted. Verify the internal ID.
938	Add, Modify add(), modify()	Agreement_to_project	Duplicate agreement_to_project	The projectid and agreementid pair must be unique. There is already one agreement – project link between the same agreement and the same project.
939	MakeURL makeURL()	—	View is not allowed for this user	[MakeURL error] — The authenticated user does not have sufficient permission (role or filter set) to view this dashboard, invoice, or timesheet.
940	MakeURL makeURL()	—	Dashboard view is not allowed for this project	[MakeURL error] — The dashboard is not enabled for projects in this stage. Verify the project stage configuration and the project stage for the project.
941	CreateUser, Modify createUser(), modify()	User	Invalid timezone specified for user	The timezone, when specified, must be a string containing a + or – sign, followed by a four digit offset, and optionally a single letter. Examples: -0500, +0330, +1300a.
942	Add, Modify add(), modify()	LoadedCost	Loaded costs not allowed for generic resources	The user with internal ID userid is a generic resource and project_taskid is set. Loaded cost override information at the task assignment level is not allowed if your account is configured to allow multiple generic resource assignments on the same task.
943	Add, Modify add(), modify()	Project	Project names must be unique by customer	A project with the same name already exists for the customer with internal ID customerid. Use a different value for name.
944	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Envelope ■ Timesheet 	Invalid date	Either the start or end date is not valid. Verify the start and end date.
945	Add, Modify add(), modify()	ProjectBudgetRule	Invalid Project budget group ID specified	A project_budget_groupid must be specified. The project budget group with internal ID project_budget_groupid does not exist or is marked as deleted. Verify the internal ID.

Error Code	Command	Object Type	Short Message	More Information
946	Add, Modify add(), modify()	ProjectBudgetTransaction	Invalid project budget rule ID specified	A project_budget_ruleid must be specified. The project budget rule with internal ID project_budget_ruleid does not exist or is marked as deleted. Verify the internal ID.
947	Add, Modify add(), modify()	ExpensePolicy	Project already has an expense policy	There is already one expense policy associated with the project with internal ID projectid, and there can only be one expense policy per project.
948	Add, Modify add(), modify()	ExpensePolicyItem	Duplicate itemid for expense policy	There is already one expense item with the same itemid in the associated expense policy. The expense_policyid and itemid combination must be unique.
949	Add, Modify add(), modify()	AttributeDescription	Invalid Resourceprofile_type ID specified	The resource profile type with internal ID resourceprofile_typeid does not exist or is marked as deleted. Verify the internal ID.
950	Add, Modify add(), modify()	AttributeDescription	Invalid Attribute ID specified	The attribute with internal ID attributeid does not exist or is marked as deleted. Verify the internal ID.
951	Add, Modify add(), modify()	AttributeDescription	Duplicate Attribute for Resourceprofile_type	There is already one attribute with the same attributeid associated with the resource profile type with internal ID resourceprofile_typeid. The attributeid and resourceprofile_typeid combination must be unique.
952	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ ProjecttaskEstimate ■ Proxy 	Duplicate entry for user	<ul style="list-style-type: none"> ■ There is already a project task estimate with the same project_taskid, userid, and timesheetid. The project_taskid, userid, and timesheetid combination must be unique. ■ There is already one proxy record with the same proxyid and the same userid. The proxyid and userid combination must be unique.
953	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ ProjectBudgetRule ■ ProjectBudget Transaction 	Missing labor subcategory	A labor_subcategory must be set for the associated project budget group when category is set to 1 (labor). See ProjectBudgetGroup .
954	Add, Modify add(), modify()	Revenue_recognition_rule	Invalid Project billing rule ID specified	A project_billing_ruleid must be specified. The project billing rule with internal ID project_billing_ruleid does not exist or is marked as deleted. Verify the internal ID.
960	Add, Modify add(), modify()	ResourceAttachment	Invalid Resource attachment type	The resource attachment type must be either CV or AVATAR.
961	Add, Modify add(), modify()	ResourceAttachment	Duplicate entry for user	There is already one resource attachment record with the same type and the same userid. A user can only have one resource attachment of the same type.
962	Add, Modify add(), modify()	ResourceAttachment	ResourceAttachment cannot be modified	The authenticated user does not have sufficient permission to add or modify a resource attachment, or the resource attachment cannot be edited.
963	Add, Modify	ResourceAttachment	Invalid attachment id	The attachment with internal ID attachment_id does not exist, is marked

Error Code	Command	Object Type	Short Message	More Information
	<code>add(), modify()</code>			as deleted, or is not associated with the user with internal ID <code>userid</code> . Verify the internal ID.
964	Add, Modify <code>add(), modify()</code>	Attachment	Invalid ResourceAttachment id	The resource attachment with internal ID <code>owner_id</code> does not exist or is marked as deleted. Verify the internal ID.
965	Add, Modify <code>add(), modify()</code>	Attachment	File could not be saved	The attachment record was created but marked as deleted because the file could not be saved to disk. Try adding the attachment again and if the error persists, contact OpenAir Customer Support.

1000 – 1199

Error Code	Command	Object Type	Short Message	More Information
1001	Submit, Approve, Reject, Unapprove <code>submit(), approve(), reject(), unapprove()</code>	All approvable	Invalid state	Submit, Approve or Reject operation could not be completed. Transaction objects can only be submitted if the approval status is O (Open) or R (Rejected), can only be approved or rejected if the approval status is P (Pending approval), and can only be unapproved if the approval status is A (Approved).
1002	Submit, Approve, Reject, Unapprove <code>submit(), approve(), reject(), unapprove()</code>	All approvable	Submit/Approve/Reject error	The transaction object could not be submitted, approved, or rejected. Some of the possible reasons for this error include but are not limited to the following: <ul style="list-style-type: none"> ■ [Submit error] There's a problem with the transaction record, or the authenticated user does not sufficient permission to submit the transaction. ■ [Approve/Reject error] The Approval argument was not specified in the API request. ■ [Approve/Reject error] The authenticated user cannot approve or reject the transaction object. ■ [Reject error] Rejection reasons [notes] must be specified. ■ [Unapprove error] The account configuration does not allow the unapprove action, the authenticated user does not have sufficient permission to unapprove the transaction object, there are downstream transactions blocking the unapproval, or the approved transaction was exported.
1003	Submit, Approve, Reject, Unapprove <code>submit(), approve(), reject(), unapprove()</code>	All approvable	Submit/Approve/Reject warning	The transaction object could not be submitted, approved, or rejected because there are warnings associated with this transaction and it is not allowed to submit a transaction with warnings.

Error Code	Command	Object Type	Short Message	More Information
1050	Add, CreateUser, Modify add(), createUser(), modify()	<ul style="list-style-type: none"> ■ HierarchyNode ■ User 	Invalid hierarchy node specified	<p>The hierarchy node specified is not valid.</p> <p>If the object is a hierarchy node, verify the following conditions</p> <ul style="list-style-type: none"> ■ The hierarchy node must be part of an existing and active hierarchy. hierarchyid must be specified, and the hierarchy with this internal ID must exist and be active. ■ If an immediate ancestor is specified, the hierarchy node with internal ID parentid must exist and must be part of the same hierarchy. ■ If isalevel is set to 1 then levelid, isanode and recordid must be set to 0, and name must be specified. ■ If isanode is set to 1 then name must be specified and: <ul style="list-style-type: none"> □ isalevel must be set to 0. □ If parentid is set then there must be a hierarchy level under the hierarchy level for the immediate ancestor. □ levelid must be specified, must be the internal ID of a hierarchy level, and must be part of the same hierarchy. ■ If isalevel and isanode are both set to 0 then recordid must be specified, id must not be specified (record – hierarchy node association can be added but not modified), parentid must be the internal ID of a valid hierarchy node, and the record with internal ID recordid must not be associated already with a hierarchy node in the same hierarchy. <p>If the object is a user, verify that all hierarchy nodes with internal IDs specified in project_access_nodes exist, are project hierarchy nodes (isanode set to 1 and hierarchy with internal ID hierarchyid has type set to project).</p>
1051	CreateUser, Modify createUser(), modify()	User	You cannot assign multiple nodes within one hierarchy	The same object cannot be associated with more than one hierarchy node in the same hierarchy. If the object is a user, verify that all hierarchy nodes with internal IDs specified in hierarchy_node_ids belong to different hierarchies.
1100	Add, CreateUser, Modify add(), createUser(), modify()	All object types with custom fields	Invalid value specified for a checkbox custom field	The custom field type is a checkbox and the value specified for a checkbox custom field must be either 0, 1, or empty.

Error Code	Command	Object Type	Short Message	More Information
1101	Add, CreateUser, Modify add(), createUser(), modify()	All object types with custom fields	Value specified is not on the list of values for this custom field	The value specified for this custom field must be one of the values defined in the custom field properties. Value options can be defined for custom fields of the following types: allocation grid, dropdown, dropdown and text, multiple selection, radio group, tag.
1102	Add, Modify add(), modify()	CustField	Custom field could not be saved	The custom field properties could not be saved. Check returned error description for details. This error is also returned if you specify valuelist and the custom field is not of one of the following types: allocation grid, dropdown, dropdown and text, multiple selection, radio group, tag.
1103	Modify	CustField	Modification of the field specified is not supported	Custom field property other than valuelist cannot be modified.
1104	Add, CreateUser, Modify add(), createUser(), modify()	All object types with custom fields	This custom field value is not unique	The custom field value must be unique and the value specified is already used for another object of this type. Use a different value. Date, text and URL custom fields can be set up so that each record must have a unique value held in the custom field.
1105	Add, CreateUser, Modify add(), createUser(), modify()	All object types with custom fields	Value specified is not on the list of values in the source pick list defined for this custom field	The custom field type is a pick list, and the value specified must be an integer and the internal ID of an existing record of the type selected in the custom field List source property, or in the case of a pick list sourced from projects, the value specified must follow the format <customerid>:<projectid>, where <customerid> and <projectid> are the internal ID of an existing customer, and an existing project for this customer.
1106	Add, CreateUser, Modify add(), createUser(), modify()	All object types with custom fields	One or more inline custom fields failed to be updated	One or more custom field values could not be saved. Check returned error description for details.

1200 – 1999

Error Code	Command	Object Type	Short Message	More Information
1200	ModifyOnCondition	All	Condition not met	The condition was not met and the object was not modified. The API response includes the read and unmodified object.

Error Code	Command	Object Type	Short Message	More Information
1300	Add, CreateUser, Modify add(), createUser(), modify()	—	Invalid filter set specified	One or more of the filter set internal IDs specified is not valid. Verify that all filter sets with internal IDs specified in filterset_ids exist.
1400	Add, Modify add(), modify()	Timesheet	Missing start_end_month_ts flag	start_end_month_ts, when set, must be either S (start) or E (end). start_end_month_ts must be specified if associated_tmId is specified.
1401	Add, Modify add(), modify()	Timesheet	Invalid associated_tmId	The timesheet with internal ID associated_tmId is not the valid split timesheet counterpart for this timesheet. Linked split timesheets at month end must have the same userid, starts and ends date. One must have start_end_month_ts set to S and the other must have start_end_month_ts set to E.
1402	Add, Modify add(), modify()	Timesheet	Non-overlapping timesheet	To specify associated_tmId or start_end_month_ts, your account must be configured to split timesheets automatically at month end, and the linked timesheets must be across two consecutive months.
1403	Add, Modify add(), modify()	Timesheet	Cannot modify timesheet with associated_tmId	The following properties cannot be modified for either part of a split timesheet at month end (that is, if associated_tmId is not empty): userid, starts, ends.
1404	Add, Modify add(), modify()	Task	Invalid time	Either start_time or end_time value is invalid. The format for start_time and end_time must be hh:mm:ss.
1405	Add, Modify add(), modify()	Task	Illegal time range	The start_time value must be before the end_time value.
1406	Add, Modify add(), modify()	Task	No permission to edit time data	To set start_time or end_time, start and end time entry on timesheets must be enabled for your account. The Enable start and end time entry on timesheets box must be checked on the Administration > Application Settings > Timesheets > Other settings form.
1407	Add, Modify add(), modify()	Task	Invalid hours	The duration set using either decimal_hours or hours and minutes does not match the period between start_time and end_time
1408	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Newsfeed Message ■ Project 	Invalid newsfeed	The newsfeed with internal ID newsfeedid does not exist or is marked as deleted. Verify the internal ID.

Error Code	Command	Object Type	Short Message	More Information
1409	Add, Modify add(), modify()	NewsfeedMessage	Both author or editor not set	Either authorid or editorid must be specified.
1410	CreateUser, Modify createUser(), modify()	User	Deactivate ns integration user	The user is a NetSuite integration administrator and cannot be deactivated (active must be set to 1).
1411	CreateUser, Modify createUser(), modify()	User	Change admin role of ns integration user	The user is a NetSuite integration administrator and must be an account administrator (role cannot be changed).
1412	Add, Modify add(), modify()	Ticket	Invalid quantity	The quantity must not be 0 (zero).
1413	Add, Modify add(), modify()	Invoice	Invalid payment terms	The internal ID of the associated payment terms payment_termsid must be valid and correspond to the payment terms (terms) specified for the invoice. To specify payment_termsid, the Save Payment Terms Internal ID on Invoice Records optional feature must be enabled for the OpenAir account.
1414	Add, Modify add(), modify()	Schedulerequest	Invalid approval status	The approval_status must be specified to one of the following values: O (Open), P (Pending approval), A (Approved), R (Rejected).
1415	Add, Modify add(), modify()	<ul style="list-style-type: none"> ■ Projecttask ■ Projecttaskassign 	Phase cannot be assigned	A project phase cannot have project task assignments. A project task assignment cannot be associated to the project task with internal ID projecttaskid if that project task is a phase (is_a_phase set to 1). A project task cannot be designated as a phase (is_a_phase set to 1) if there are project task assignments associated with the project task. assign_user_names cannot be set if is_a_phase is set to 1 for a project task.
1416	Add, Modify add(), modify()	Projectbillingrule	Invalid cap by customer PO	You can only set or modify the cap_by_customerpo property if all the following conditions are met: <ul style="list-style-type: none"> ■ The project associated to the project billing rule is a portfolio project. ■ The project associated to the project billing rule is associated with at least one customer PO. ■ The billing rule is associated with a customer PO.

Error Code	Command	Object Type	Short Message	More Information
				<ul style="list-style-type: none"> ■ The billing rule type is one of the following: Expense item, Purchase item, Time. ■ The following features are enabled for your account: <ul style="list-style-type: none"> □ Portfolio Projects and Subordinate Projects. □ Single Billing Cap across Multiple Subprojects Within a Portfolio Project.
1417	Add, Modify add(), modify()	Projectbillingrule	Invalid project task id	<p>You can only set or modify the project_task_id property if all the following conditions are met:</p> <ul style="list-style-type: none"> ■ project_task_id must be the internal ID of a top level phase in the project schedule. ■ The account must be configured to require either a Service or Service 1-5 line on top level phases. ■ The billing rule type must be 'F' (fixed fee billing rule).
1418	Add, Modify add(), modify()	Preference	Invalid preference settings format	The preference settings format is not valid.
1419	CreateUser, Modify createUser(), modify()	User	No full user licenses available	User cannot be granted access to modules other than Account, Expenses and Timesheets.
1420	CreateUser, Modify createUser(), modify()	User	No T&E or full user licenses available	Cannot add T&E user or mark T&E user as active.
1421	CreateUser, Modify createUser(), modify()	User	No guest or full user licenses available	Cannot add guest user or mark guest user as active.

2000 – 2999

Error Code	Command	Object Type	Short Message	More Information
2001	—	—	Invalid argument passed	There is a problem with one or more arguments in your API request. Verify the arguments.
2002	—	—	Invalid format passed	There is a problem with the format of your API request. Verify the format of your API request.

API Limits

OpenAir enforces some limits to control API consumption and manage the demands on OpenAir application and database servers. These limits apply across all OpenAir API components — SOAP API, XML API, and REST API.

There are two types of usage limits

■ Number of objects

- The API returns a maximum of 1,000 objects per request. Use pagination to retrieve a list of more than 1,000 objects. The maximum page length is 1000.
 - REST API – You can set a `limit` parameter to control the page length. The page length defaults to 100 if not specified. See the help topic [Pagination](#).
 - XML and SOAP API – You must set a `limit` attribute to control the page length. See [Read Attributes](#) and [Pagination](#).
- The API accepts a maximum of 1,000 objects as arguments per request. To process more than 1,000 objects, do so in batches. You can add, modify, delete, submit, approve, reject or unapprove a maximum of 1,000 objects using one XML API call or SOAP API command. You can add or modify only one object using one REST API request, You can delete a maximum of 100 or 1,000 objects, depending on the object type, using one REST API request.

■ Frequency limits

- Maximum number of requests allowed within a 24-hour window for your company's OpenAir account.
- Maximum number of requests allowed within a 60-second window for your company's OpenAir account.

API Frequency Limit Error

If the number of API requests made in the last 60 seconds or in the last 24 hours reaches the maximum allowed, OpenAir API returns an error.

- The XML API returns the error code 556 to the authentication operation [\[Auth\]](#).
- The SOAP API returns a 403 Access denied.
- The REST API returns a 429 Too Many Requests error for any request sent within the 24-hour or 60-second window.

OpenAir sends a warning email when you approach your API frequency limits.



Tip: To work within your API frequency limits and avoid frequency limit errors, batch operations into each API call and avoid making API calls within a loop. See [Optimize the API Integration](#).

Tracking API Usage Against Frequency Limits

This screen shows the API requests limits that are currently set for the account and the number of requests remaining within the current 24-hour period. It is a useful reference to track your usage level when using the OpenAir SOAP API and XML API. This is for reference purposes only, account administrators cannot change these settings.

To track your API usage against frequency limits, do one of the following:

- In OpenAir, go to Administration > Global Settings > Account > API Limits. The page screen shows the API frequency limits for your company's OpenAir account, the thresholds when OpenAir sends a warning email for each frequency limit, and the number of requests remaining within the current 24-hour window. You can use this page to track API usage but you cannot change the frequency limit settings. It is a read-only page for all users.

- In OpenAir, review web services logs to identify requests contributing toward any usage limit overages. See [Web Services Logs](#).
- Use the XML API to read the number of requests remaining within the current 24-hour window. To do so, use the [Read XML API](#) command and the [RateLimit](#) object.

 **Tip:** Query the remaining number of requests at various points in your integration application to identify where your application is sending the highest volume of requests. Follow [OpenAir API Best Practice Guidelines](#) to see how you can improve your application.

If you have any questions about frequency limits, contact OpenAir Customer Support. For assistance with your integration applications and to help reduce the number of API requests in your integration applications, contact OpenAir Professional Services.

RateLimit

The rate limit [RateLimit] is the number of OpenAir API requests remaining in the current 24-hour window.

Review [Usage Guidelines](#) for the RateLimit object.

 **Note:** OpenAir enforces some limits to control API consumption and manage the demands on OpenAir application and database servers. One such usage limit applies to the number of requests in any 24-hour window. See [API Limits](#).

—	XML	SOAP	REST
Object	RateLimit	oaRateLimit	—
Supported Commands	Read (all)	—	—

 **Note:** The RateLimit object supports the all read method only.

The RateLimit object has the following properties:

Field Name	Description
remain_24h_error	Number of calls remaining in a 24 hour window

Usage Guidelines

You can only query the remaining number of requests using OpenAir XML API. The OpenAir WSDL lists the `oaRateLimit` but OpenAir SOAP API does not support reading this object. OpenAir REST API does not have an equivalent method to query the remaining number of requests.

The following XML API sample code queries the remaining number of requests within the current 24-hour window:

```

1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <request API_version="1.0" client="example client" client_ver="1.1" namespace="example" key="0123456789">
3   <Auth>
4     <Login>
5       <access_token>0123456789-ABCDEFGHIJKLMNIOQRSTUVWXYZ0123456789ABCDEF-ABCDEFGHIJKLMNIOQRSTUVWXYZ01234567</access_token>
6     </Login>
7   </Auth>
8   <Read type="RateLimit" method="all" limit="1">
9     </Read>
10 </request>

```

The following XML API sample response indicates that there are 99949 requests remaining within the current 24-hour window.

```

1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <response>
3   <Auth status = "0"></Auth >
4   <Read status = "0">
5     <RateLimit>
6       <remain_24h_error>99949</remain_24h_error>
7     </RateLimit>
8   </Read>
9 </response>

```



Note: This request call counts uses up one request in your API frequency limit.

Web Services Logs

An optional feature lets you access web services logs in the Reports application in OpenAir. If the feature is enabled for your account, go to Reports > Detail > Web services > Web services logs, or go to Reports > Management and search for “Web services logs”. Configure and run the report for auditing purposes or to help troubleshooting API requests in your integration applications.

- Each row in the Web services log represents **one** request and response pair. For the request part, OpenAir logs the method used, the request URL, the Content-Type header and the request body. For the response part, OpenAir logs the HTTP response code and the response body.
- Records in the Web services log are only available for seven days after they are created.

 **Note:** This feature includes an optional component, which may be enabled to help troubleshoot any issues with the add-on services provided by OpenAir.

If you are using Web services log reports to track your API usage limits, note that API requests made by OpenAir Mobile apps, OpenAir Integration Manager and other OpenAir add-on services do not count toward your usage limits.

 **Important:** The Web services log report feature has the following limitations:

- If you do not use this feature for more than 30 days, the feature is disabled and the log entries are deleted.
- Log entries are retained for 7 days only, then they are purged from the database.

XML API Commands

The following table lists the commands supported in the OpenAir XML API in alphabetical order. For each command, it provides a brief description and a link to a help topic with syntax and usage information and code samples.

Command	Description
Add	Adds a new object or updates an existing object.
Approve	Approves a transaction that was submitted for approval.
Auth	Authenticates the user.
CreateUser	Adds or Updates a User object.
Delete	Deletes an object.
MakeURL	Creates a URL to a specific OpenAir page.
Modify	Updates an object.
ModifyOnCondition	Updates an object only if the condition is satisfied.
Read	Retrieves one or more objects of the type specified based on the method, attributes and other parameters passed.
Reject	Rejects a transaction that was submitted for approval.
RemoteAuth	Authenticates the user and returns a URL to access the OpenAir UI as the authenticated user.
Report	Runs a report and emails a timesheet, expense report or saved report as PDF.
Submit	Submits a transaction for approval.
Time	Retrieves the current system timestamp from the OpenAir servers.
Unapprove	Unapproves a transaction that was previously approved.
Version	Retrieves the version of a thin client application supported by OpenAir.
Whoami	Retrieves information about the authenticated user.

Add

Adds a new object or updates an existing object.

Syntax

```

1 <Add type="ObjectType" attr_name1="attr_value1" attr_name2="attr_value2">
2   <ObjectType>
3     <property1>value1</property1>
4     <property2>value2</property2>
5     ...
6   </ObjectType>
7 </Add>
```

Usage

Use the Add command to add or update an object. The maximum number of objects you can add or update with one XML API call is 1,000.

The Add command cannot be used to add User objects. Use [CreateUser](#) instead.

You can set custom field values as well as standard field values when adding objects. See [Reading or Setting Custom Field Values Inline](#).

Attributes

Attribute	Usage
enable_custom	XML API Only — Set the enable_custom attribute to 1 to include custom field values in the argument object properties along with standard object properties.
lookup	Use the lookup to upsert an object and designate the lookup field. The lookup field determines whether it adds or updates an object: <ul style="list-style-type: none"> ■ If the lookup field is not matched in any existing objects, a new object is added. ■ If the lookup field is matched one time, the existing object is updated.

Arguments

Name	Type	Description
<i>ObjectType</i>	Object	The object to add or update.

Response

ObjectType — The added or updated object with all properties including the object internal ID [id], date created timestamp [created], and date updated timestamp [updated].

Sample Code — Upsert

The following example looks up a Category object with name="XML-created category 1", updates the external ID of the matching Category object, if it exists, or adds the object otherwise.

```

1 <Add type="Category" lookup="name">
2   <Category>
3     <name>XML-created category 1</name>
4     <externalid>111-2222</externalid>
5   </Category>
6 </Add>
```

Sample Code — Adding CV as Attachment to a Resource Profile

The following example adds a CV as attachment to an employee's resource profile. This is a two step process:

1. Add a ResourceAttachment object.
2. Add the attachment file as base64 encoded data. Set the ownerid property of the Attachment object to the internal ID of the ResourceAttachment object added in the previous step.

```

1 // Step 1 - Add a ResourceAttachment object
2 <Add type="ResourceAttachment">
3   <ResourceAttachment>
4     <type>CV</type>
5     <userid>123</userid>
6   </ResourceAttachment>
7 </Add>
8
9 // Step 2 - Upload the CV as Attachment with ownerid set to the
10 // internal ID of the ResourceAttachment object added in Step 1
11 <Add type="Attachment">
12   <Attachment>
13     <base64_data>U251emth</base64_data>
14     <file_name>Collins_Marc_CV.txt</file_name>
15     <owner_type>ResourceAttachment</owner_type>
16     <ownerid>98765</ownerid>
17   </Attachment>
18 </Add>

```

Approve

Approves a transaction that was submitted for approval.

Syntax

```

1 <Approve type="ObjectType">
2   <ObjectType>
3     <id>objectID</id>
4   </ObjectType>
5   <Approval>
6     <cc>name@example.com</cc>
7     <notes>An example of approval notes.</notes>
8   </Approval>
9 </Approve>

```

Usage

Use the Approve command to approve a transaction which was submitted for approval. The maximum number of objects you can approve in the same API call is 1,000.

For information about object types for which the OpenAir XML API and SOAP API support approval-related operations, see [Approval-Related Operations](#).

Arguments

Name	Type	Description
<i>ObjectType</i>	Object	An object of type <i>ObjectType</i> . The <i>ObjectType</i> must support approvals. The object property passed must include the object internal ID [id]
Approval	Approval	An Approval object.

Response

A success or fail status.

Auth

Authenticates the user.

Syntax

Password Authentication

```

1 <Auth>
2   <Login>
3     <company>example</company>
4     <user>mcollins</user>
5     <password>openair_password</password>
6   </Login>
7 </Auth>

```

OAuth 2.0 Access Token Authentication

```

1 <Auth>
2   <Login>
3     <access_token>0123456789-ABCDEFGHIJKLMNIOQRSTUVWXYZ0123456789ABCDEF-ABCDEFGHIJKLMNIOQRSTUVWXYZ01234567</access_token>
4   </Login>
5 </Auth>

```

Usage

Use the Auth command to authenticate the user. The Auth XML command is required at the top of every XML API call for other commands to succeed.

OpenAir XML API supports the following authentication methods:

- Password – User credentials (Company ID, User ID and Password) can be passed in the Auth command. See [Password Authentication](#).
- OAuth 2.0 access token – The access token (access_token) is passed instead of user credentials in the Auth XML command. See [OAuth 2.0 Access Token Authentication](#).

For more information about OAuth 2.0, see [OAuth 2.0 for Integration Applications Developers](#).



Important: An invalid OAuth2 access token authorization has priority over a valid password or client session ID authentication. You cannot use password or client session ID authentication as a fallback for an invalid access token. See [Using OAuth 2.0 Access Tokens in Your API Requests](#).

Arguments

Name	Type	Description
Login	Login	A Login object.

Response

A success or fail status. Success is required for other commands to complete successfully.

Login

The Login object defines the authentication parameters for the Auth command.

A Login object has the following properties:

Name	Type	Description
access_token	string	An OAuth 2.0 access token can be used instead of user credentials [company, user, and password] or client session ID [session_id]. See OAuth 2.0 Access Token Authentication . All other properties are ignored if access_token is not empty. For more information about OAuth 2.0, see OAuth 2.0 for Integration Applications Developers .
company	string	Your company ID.
password	string	The authenticating user's password.
user	string	The authenticating user's user ID.

CreateUser

Adds or Updates a [User](#) object.

Syntax

The following syntax includes all required properties. You can set any other properties for the Company and User objects.

```

1  <CreateAccount>
2    <Company>
3      <nickname>Example</nickname>
4    </Company>
5    <User>
6      <nickname>mcollins</nickname>
7      <password>openair_password</password>
8      <addr>
9        <Address>
10       <email>mcollins@example.com</email>
11     </Address>
12   </addr>
13 </User>
14 </CreateAccount>

```

Usage

Use the CreateUser command to add a new user or update an existing user. The maximum number of objects you can add or update with one XML API call is 1,000.

To add objects of types other than User, use the [Add](#) command instead.

You can set custom field values as well as standard field values when adding User objects. See [Reading or Setting Custom Field Values Inline](#).

You can set the employee work schedules when adding User objects. See [User](#).



Important: Review the following guidelines:

- You must be authenticated as an account administrator to use the CreateUser command.
- You must set a password when using the CreateUser command **to create a new user record** except for generic user records (generic set to 1). This is also true when using the CreateUser command with a foreign key lookup that results in inserting a new user record.
- If you are using SAML for authentication in to your OpenAir account:
 - You can set a password and enable SAML authentication for the user (setting the Boolean custom field saml_auth__c to true) when using the CreateUser command **to create a new user record**.
 - You cannot set a password if SAML authentication is enabled for the user (saml_auth__c set to true) when using the CreateUser command with a foreign key lookup **to update an existing user record**. The API will return the following error: "System.Exception: Not enabled to edit password: Edit of passwords is not allowed".
- Limits are enforced to prevent you from creating or activating users if doing so would exceed the number of user licenses purchased for your account. If no user licenses of the appropriate type are available, the CreateUser command creates a new user record, but sets it as inactive (clears the **Active** box on the employee record), or to activate a user record (to check the **Active** box on the employee record). For more information about OpenAir licensing and compliance, see .

Attributes

Attribute	Usage
enable_custom	XML API Only — Set the enable_custom attribute to 1 to include custom field values in the argument object properties along with standard object properties.
exclude_flags	XML API Only — Set the exclude_flags attribute to 1 to exclude user settings from the response.
lookup	Use the lookup to upsert an object and designate the lookup field. The lookup field determines whether it adds or updates an object: <ul style="list-style-type: none"> ■ If the lookup field is not matched in any existing objects, a new object is added. ■ If the lookup field is matched one time, the existing object is updated.

Arguments

Name	Type	Description
Company	Object	A Company object

Name	Type	Description
User	Object	A User object

Response

A User object.

Delete

Deletes an object.

Syntax

```

1 <Delete type="ObjectType">
2   <ObjectType>
3     <id>internalID</id>
4   </ObjectType>
5 </Delete>

```

Usage

Use the Delete command to delete an object based with a specific internal ID. The maximum number of objects you can delete with one XML API call is 1,000.

Arguments

Name	Type	Description
ObjectType	Object	The object to be deleted. Object properties must include the object internal ID [id].

Response

- Success if the object with internal ID id was found and deleted.
- Failure if the object with internal ID id was not found.
- A list of objects if the object with internal ID id was found but has dependencies. All dependent objects (objects referencing the ObjectType object by internal ID) must be deleted before you can delete the ObjectType object. Properties for the listed objects include only the internal ID [id].

MakeURL

Creates a URL to a specific OpenAir page.

Syntax

```

1 <MakeURL>
2   <uid>userID</uid>
3   <page>pageName</page>
4   <app>xy</app>
5   <arg>
6     <ObjectType>
7       <id>internalID</id>
8     </ObjectType>
9   </arg>
10 </MakeURL>

```

Usage

Use the MakeURL command to create a URL to a specific OpenAir page.

Arguments

Name	Type	Description
uid	string	A valid client session ID.
page	string	Code representing the page.
app	string	Two-letter code representing the application.
arg	string	Argument required depending on the page.

For information about the supported pages in the OpenAir UI (identified by a navigation path) and the corresponding page, app and arg combinations, see [OpenAir Pages Supported by the Make URL Operation](#).

Response

The page URL for the authenticated user.

Modify

Updates an object.

Syntax

```

1 <Modify type="ObjectType" attr_name1="attr_value1" attr_name2="attr_value2">
2   <ObjectType>
3     <property1>value1</property1>
4     <property2>value2</property2>
5     ...
6   </ObjectType>
7 </Modify>

```

Usage

Use the Modify command to update an object. The maximum number of objects you can add or update with one XML API call is 1,000.

You can use the Modify command to update a [User](#) object. See the object reference for usage limitations specific to the [User](#) object.

Determine the internal ID of each object you want to update and pass the object containing the object internal ID and the object properties you want to update.

You can set custom field values as well as standard field values when updating objects.

You can look up associated objects using an external ID or name as foreign key instead of passing the associated object internal ID. See [Related Object Lookup Using the XML API](#).

Attributes

Attribute	Usage
enable_custom	— Set the enable_custom attribute to 1 to include custom field values in the argument object properties inline with standard object properties. Object property names for custom fields include the custom field name as defined in OpenAir followed by two underscores and the letter c: <i>custom_field_name__c</i> .
method	Set the method attribute to <code>custom equal to</code> to update a custom field value. <pre> 1 <Modify type="ObjectType" method="custom equal to"> 2 <ObjectType> 3 <id>X</id> 4 <custom_field>custom_field_name</custom_field> 5 <value>custom_field_value</value> 6 </ObjectType> 7 </Modify> </pre> <div style="border: 1px solid #0070C0; padding: 5px; margin-top: 10px;"> <p> Note: You should use the enable_custom to modify custom field values inline with standard object properties in the argument object instead of using the custom equal to method.</p> </div>

Arguments

Name	Type	Description
<i>ObjectType</i>	Object	The object to add or update.

Response

ObjectType — The updated object with all properties including the object internal ID [id], and date updated timestamp [updated].

Sample Codes

The following example marks a charge as exported:

```

1 <Modify type="ImportExport">
2   <ImportExport>
3     <application>MyAppName</application>
4     <type>Slip</type>
5     <id>10158</id>
6     <exported>
7       <Date>
8         <year>2023</year>
9         <month>03</month>
10        <day>14</day>
11        <hour>11</hour>
12        <minute>25</minute>
13        <second>15</second>
14      </Date>
15    </exported>
16  </ImportExport>
17 </Modify>

```

The following example updates the name and the value of custom field `my_custom_field` for a customer:

```

1 <Modify type="Customer" enable_custom="1">
2   <ImportExport>
3     <name>Example Customer</name>
4     <my_custom_field__c>123456789</my_custom_field>
5   </Customer>
6 </Modify>

```

Updating Company Logos

Note: The following information is subject to change without prior notice.

You can use the Modify command to update the company logos. This is equivalent to changing the company logos on the Administration > Global Settings > Display > Logos: UI: Documents in the OpenAir UI – see the help topic [Logos: UI, documents](#). To do so, use the object type Logo. Attributes, if passed, have no impact.

A successful request returns the updated Logo object.

A Logo object has the following properties:

Name	Description
name	The name determines which logo will be updated. Possible values: <ul style="list-style-type: none"> banner_logo – Logo displayed on OpenAir HTML documents. html_logo – Logo displayed on OpenAir PDF documents. pdf_logo – Company pictogram displayed in the upper left corner of the OpenAir UI .
type	[Read-only] – The file type calculated from the binary property value. A successful command returns a Logo object and includes the calculated file type.
filename	The filename. Used for display purposes only.
binary	Base 64 encoded binary data of the company logo file.

ModifyOnCondition

Updates an object only if the condition is satisfied.

Syntax

```

1 <ModifyOnCondition type="ObjectType" condition="if-not-updated" attr_name1="attr_value1" attr_name2="attr_value2">
2   <ObjectType>
3     <property1>value1</property1>
4     <property2>value2</property2>
5     ...
6   </ObjectType>
7   <Date>
8     <day>DD</day>
9     <month>MM</month>
10    <year>YYYY</year>
11    <hour>hh</hour>
12    <minute>mm</minute>
13    <second>ss</second>
14  </Date>
15 </ModifyOnCondition>

```

Usage

Use the `ModifyOnCondition` command in the same way as the `Modify` command to update an object but only when the specified condition is satisfied.

The only supported condition is `if-not-updated` – Use this condition to update an object only if it was last updated before or on the date specified in the `Date` argument.

- If the updated time stamp is before or equal to the date specified in the `Date` argument, the object is updated and the response status is 0 (Success).
- If the updated time stamp is after the date specified in the `Date` argument, the object is not updated and the response status is 1200 (Condition not met)

For other usage notes, see [Modify](#).

Attributes

Same attributes as [Modify](#) and one additional attribute.

Attribute	Usage
condition	[Required] Set the condition attribute to <code>if-not-updated</code> to only if it was not updated after the date specified in the <code>Date</code> argument.

Arguments

Name	Type	Description
<i>ObjectType</i>	Object	The object to add or update.
Date	Date	A Date object.

Response

- *ObjectType* — The object (updated or not) with all properties including the object internal ID [`id`], and date updated timestamp [`updated`]

- A status (error code):
 - 0 if the condition was met and the object was updated.
 - 1200 if the object was not updated because the condition was not met.

Sample Code

The following example modifies the booking's external ID, Notes, and booked resource if the booking was last updated before or on January 1, 2023 at 20:05:09 Eastern Time (UTC-5).

```

1 <ModifyOnCondition condition="if-not-updated" type="Booking">
2   <Booking>
3     <id>2</id>
4     <externalid>123456789</externalid>
5     <notes>New notes</notes>
6     <userid>123</userid>
7   </Booking>
8   <Date>
9     <day>01</day>
10    <month>01</month>
11    <year>2023</year>
12    <hour>20</hour>
13    <minute>05</minute>
14    <second>09</second>
15  </Date>
16 </ModifyOnCondition>

```

Read

Retrieves one or more objects of the type specified based on the method, attributes and other parameters passed.

Syntax

```

1 <Read type="ObjectType" method="method name" attr_name1="attr_value1" attr_name2="attr_value2">
2 </Read>

```

Usage

Use the Read command to retrieve one or more objects of a specific type.

Define the type of objects to retrieve, the read method, the maximum number of objects to return (limit) and any optional attributes. See [ReadRequest](#).

Attributes

Name	Description
type	[Required] The type of object to return. For information about supported object types, see List of Supported Business Object Types .
method	[Required] The name of the read method to be used. The read method lets you specify whether to return all objects, only the objects matching or not matching a search query object, or the custom field values for a specific object. See Read Methods .

Name	Description
limit	[Required] Use the <code>limit</code> attribute to control the response pagination with an offset (the number of objects to skip) and the maximum number of objects to be returned per page. See Pagination .
<i>optional attributes</i>	The attributes lets you modify the response when reading objects using the OpenAir XML API or SOAP API. Some attributes control API features available for most supported object types, other attributes can only be used when reading a specific object type. See Read Attributes .

Arguments

Name	Description
<i>ObjectType</i>	Used with the <code>equal</code> to, <code>not equal</code> to, and <code>custom equal</code> to methods. An object of the same type as the objects to return with the object property values that returned objects should match or not match.
Date objects	Used with date filters. A Date object for each date filter defined by the <code>filter</code> attribute, if any. See Date Filters Usage .
ImportExport	Used with <code>not-exported</code> filter. An ImportExport object specifies the application the object was not exported to. <pre> 1 <ImportExport> 2 <application>MyApp</application> 3 </ImportExport> </pre>
<code>_Return</code>	An object listing the object properties to return. <pre> 1 <_Return> 2 <property1/> 3 <property2/> 4 <property3/> 5 </_Return> </pre> <p>The <code>_Return</code> object, must be the last argument passed (immediately before the <code>Read</code> command closing tag).</p> <p>If omitted or empty, the response includes all standard object properties for each returned object. Unless the attribute <code>exclude_flags</code> is set to 1, the response also includes the <code>flags</code> property for <code>User</code> or <code>Company</code> objects. If the attribute <code>enable_custom</code> is set to 1, the response also includes all custom fields for the object type.</p>

Response

ObjectType objects.

Sample Codes

Read with all Method and Date Filters

The following example returns up to a thousand time entries for dates between July 1, 2023 and August 1, 2023 using `newer-than` and `older-than` date filters. The only properties included in the response are

the time entry internal ID, the number of hours and minutes, the associated project internal ID, and the associated project task internal ID.

```

1 <Read type="Task" filter="newer-than,older-than" field="date,date" method="all" limit="1000">
2   <Date>
3     <year>2023</year>
4     <month>07</month>
5     <day>01</day>
6   </Date>
7   <Date>
8     <year>2023</year>
9     <month>08</month>
10    <day>01</day>
11  </Date>
12  <_Return>
13    <id/>
14    <hours/>
15    <minutes/>
16    <projectid/>
17    <projecttaskid/>
18  </_Return>
19 </Read>

```

Read with equal to Method and Date Filter

The following example returns up to a thousand time entries for dates after January 1, 2023 that are associated with the customer with internal ID 5 and that are not associated with any charges (charge internal ID is null). The only properties included in the response are the time entry internal ID, the number of hours and minutes, the associated project internal ID, and the associated project task internal ID.

```

1 <Read type="Task" method="equal to" filter="newer-than" field="date" limit="1000">
2   <Date>
3     <year>2023</year>
4     <month>01</month>
5     <day>01</day>
6   </Date>
7   <Task>
8     <slipid/>
9     <customerid>5</customerid>
10  </Task>
11  <_Return>
12    <id/>
13    <hours/>
14    <minutes/>
15    <projectid/>
16    <projecttaskid/>
17  </_Return>
18 </Read>

```

Read Expense Reports Pending Reimbursement with equal to Method

The following example returns a list of up to a thousand approved expense reports pending reimbursement.

```

1 <Read type="Envelope" method="equal to" filter="nonreimbursed-envelopes" limit="1000">
2   <Envelope>
3     <status>A</status>
4   </Envelope>
5 </Read>

```

Read with not equal to Method Matching a Custom Field

The following example returns a list of up to a thousand projects that do not match the specified values for both custom field `my_custom_field` and standard object property `tax_locationid`.

```

1 <Read type="Project" enable_custom="1" method="not equal to" limit="1000">
2   <Project>
3     <my_custom_field_c>756</my_custom_field_c>
4     <tax_locationid>5</tax_locationid>
5   </Project>
6 </Read>

```

Read the Foreign Exchange Rate for a Currency on a Specific Date

The following example returns the foreign exchange rate for the counter currency (EUR) against the base currency (USD) with a 10 decimal point precision.

 **Tip:** Read the `Currencyrate` object with `equal to` method to get a foreign exchange rate for a counter currency (or quote currency) against a base currency on a specific day.

```

1 <Read type="Currencyrate" method="equal to" limit="1000" filter="date-equal-to" field="date" base_currency="USD" precision="10">
2   <Currencyrate>
3     <csymbol>EUR</csymbol>
4     <type/>
5   </Currencyrate>
6   <Date>
7     <year>2016</year>
8     <month>09</month>
9     <day>01</day>
10  </Date>
11 </Read>

```

Read Custom Field Values with custom equal to Method

The following example returns the values of custom fields `netsuite_customer_id` and `export_customer_to_ns` for the customer with internal Id 22 as `CustomField` objects.

```

1 <Read type="Customer" method="custom equal to" field_names="netsuite_customer_id,export_customer_to_ns" limit="1000" >
2   <Customer>
3     <id>22</id>
4   </Customer>
5 </Read>

```

 **Note:** If you omit the `field_names` attribute, the values of all custom fields for the object are returned as `CustomField` objects..

Read Objects Associated with a Specific User with user Method

The following example returns the values of custom fields `netsuite_customer_id` and `export_customer_to_ns` for the customer with internal Id 22 as `CustomField` objects.

```

1 <Read type="Uprate" method="user" limit="1000" >
2   <User>
3     <id>22</id>

```

```

4 | </User>
5 | </Read>

```

Read Not Exported Charges with all Method

The following example returns the list of up to a thousand Slip objects that were not exported to the application MyApp.

```

1 | <Read type="Slip" filter="not-exported" method="all" limit="1000" >
2 |   <ImportExport>
3 |     <application>MyApp</application>
4 |   </ImportExport>
5 | </Read>

```

Reject

Rejects a transaction that was submitted for approval.

Syntax

```

1 | <Reject type="ObjectType">
2 |   <ObjectType>
3 |     <id>objectID</id>
4 |   </ObjectType>
5 |   <Approval>
6 |     <cc>name@example.com</cc>
7 |     <notes>An example of approval notes.</notes>
8 |   </Approval>
9 | </Reject>

```

Usage

Use the Reject command to reject a transaction which was submitted for approval. The maximum number of objects you can Reject in the same API call is 1,000.

For information about object types for which the OpenAir XML API and SOAP API support approval-related operations, see [Approval-Related Operations](#).

Arguments

Name	Type	Description
<i>ObjectType</i>	Object	An object of type <i>ObjectType</i> . The <i>ObjectType</i> must support approvals. The object property passed must include the object internal ID [id]
Approval	Approval	An Approval object.

Response

A success or fail status.

RemoteAuth

Authenticates the user and returns a URL to access the OpenAir UI as the authenticated user.

Syntax

```

1 <RemoteAuth>
2   <Login>
3     <company>example</company>
4     <user>mcollins</user>
5     <password>openair_password</password>
6   </Login>
7 </RemoteAuth>

```

Usage

Use the RemoteAuth command to authenticate the user and create an authenticated user session URL to access the OpenAir UI in a browser.

Both Auth and RemoteAuth authenticate the user to perform other commands in the API call. However RemoteAuth returns a URL that can be used to access OpenAir as the authenticated user in a browser without having to enter credentials on the OpenAir login page.



Note: RemoteAuth should not be used as a substitute for Auth. RemoteAuth can be used to perform single sign-on and redirecting users to an active OpenAir UI user session.

Arguments

Name	Type	Description
Login	Login	A Login object.

Response

A success or fail status. If successful, a URL to access the OpenAir UI as the authenticated user.

Report

Runs a report and emails a timesheet, expense report or saved report as PDF.

Syntax

```

1 <Report type="ObjectType">
2   <Report>
3     <relatedid>objectInternalID</relatedid>
4     <email_report>1</email_report>
5   </Report>
6 </Report>

```

Usage

Use the Report command to run a report and email the timesheet, expense report or report as PDF.

The object type can be either Timesheet, Envelope (expense report) or Report (saved report).

Arguments

Name	Type	Description
Report	Report	A Report object with the relatedid and email_report object properties.

Response

A success status (if the report exists) or fail status.

Submit

Submits a transaction for approval.

Syntax

```

1 <Submit type="ObjectType">
2   <ObjectType>
3     <id>objectID</id>
4   </ObjectType>
5   <Approval>
6     <cc>name@example.com</cc>
7     <notes>An example of approval notes.</notes>
8   </Approval>
9 </Submit>

```

Usage

Use the Submit command to submit a transaction for approval. The maximum number of objects you can submit in the same API call is 1,000.

For information about object types for which the OpenAir XML API and SOAP API support approval-related operations, see [Approval-Related Operations](#).

Arguments

Name	Type	Description
<i>ObjectType</i>	Object	An object of type <i>ObjectType</i> . The <i>ObjectType</i> must support approvals. The object property passed must include the object internal ID [id]
Approval	Approval	An Approval object.

Response

A success or fail status.

Time

Retrieves the current system timestamp from the OpenAir servers.

Syntax

```
1 | <Time/>
```

Usage

Use the Time command to retrieve the current system timestamp from as a Date object. No arguments are required.

Arguments

None

Response

Date — A [Date](#) object.

Unapprove

Unapproves a transaction that was previously approved.

Syntax

```
1 | <Unapprove type="ObjectType">
2 |   <ObjectType>
3 |     <id>objectID</id>
4 |   </ObjectType>
5 |   <Approval>
6 |     <cc>name@example.com</cc>
7 |     <notes>An example of approval notes.</notes>
8 |   </Approval>
9 | </Unapprove>
```

Usage

Use the Unapprove command to unapprove a transaction which was previously approved. The maximum number of objects you can unapprove in the same API call is 1,000.

Transactions that were approved and subsequently billed or archived cannot be unapproved.

For information about object types for which the OpenAir XML API and SOAP API support approval-related operations, see [Approval-Related Operations](#).

Arguments

Name	Type	Description
<i>ObjectType</i>	Object	An object of type <i>ObjectType</i> . The <i>ObjectType</i> must support approvals. The object property passed must include the object internal ID [id]
Approval	Approval	An Approval object.

Response

A success or fail status.

Version

Retrieves the version of a thin client application supported by OpenAir.

Syntax

```

1 <Version>
2   <name>clientApplicationName</name>
3   <number>clientVersionNumber</number>
4 </Version>

```

Usage

The Version command is used to get the current version of a thin client application supported by OpenAir.

Arguments

Name	Description
name	The name of the client application.
number	The version number.

Response

The following properties:

Name	Description
number	Current version number.
size	Size of the file to download.
url	URL of the current version of the client installation.

Whoami

Retrieves information about the authenticated user.

Syntax

```
1 | <Whoami>  
2 | </Whoami>
```

Usage

Use the `Whoami` command to retrieve the [User](#) object for the authenticated user. No arguments are required.

Arguments

None

Response

A [User](#) object.

SOAP API Commands

The following table lists the calls supported in the OpenAir SOAP API in alphabetical order. For each call, it provides a brief description and a link to a help topic with syntax and usage information and code samples.

Command	Description
add()	Adds one or more new objects.
approve()	Approves one or more transactions that were submitted for approval.
createUser()	Adds or Updates a User object.
delete()	Deletes one or more objects.
login()	Authenticates the user and starts a client session.
logout()	Ends the client session of the authenticated user.
makeURL()	Creates one or more URL to specific OpenAir pages.
modify()	Updates one or more objects.
read()	Retrieves one or more objects of the type specified based on the method, attributes and other parameters passed.
reject()	Rejects one or more transactions that were submitted for approval.
servertime()	Retrieves the current system timestamp from the OpenAir servers.
submit()	Submits one or more transactions for approval.
unapprove()	Unapproves one or more transactions that were previously approved.
upsert()	Adds or updates one or more objects.
version()	Retrieves the version of a thin client application supported by OpenAir.
whoami()	Retrieves information about the authenticated user.

Common Object Types Used With SOAP API Commands

Many commands in OpenAir SOAP API use the [oaBase](#) and [Attribute](#) complex types as arguments.

The [add\(\)](#), [createUser\(\)](#), [modify\(\)](#), [upsert\(\)](#), and [delete\(\)](#) commands return an array of [UpdateResult](#) [UpdateResult](#) objects.

oaBase

When using OpenAir SOAP API, [oaBase](#) is the base object for all other object types. The [oaBase](#) object represents a business object. Collections are passed as arrays of [oaBase](#) objects and all business object types are derived from [oaBase](#).

For the list of business object types supported in OpenAir SOAP API, see [List of Supported Business Object Types](#).

The `oaBase` object does not have any properties.

Attribute

A `Attribute` object defines the parameters for the multiple commands either as a command argument or as a property of an argument object.

A `Attribute` object has the following properties:

Name	Type	Description
name	string	The name of the attribute.
value	string	The attribute value.

UpdateResult

The `add()`, `createUser()`, `modify()`, `upsert()`, and `delete()` commands return an array of `UpdateResult` objects.

An `UpdateResult` object has the following properties:

Name	Type	Description
errors	<code>oaError[]</code>	An array of <code>oaError</code> objects (see Error). If an error occurred, the <code>add()</code> , <code>createUser()</code> , <code>modify()</code> , <code>upsert()</code> , or <code>delete()</code> command returns an array of one or more <code>oaError</code> objects providing the error code and description.
id	string	The internal ID of the record created, deleted, or updated
objects	<code>oaBase[]</code>	An array of <code>oaBase</code> objects.
status	string	If the operation is successful, a one-letter code indicating that the record was added [A], deleted [D], or updated [U]. Otherwise, -1 if one or more errors occurred.

add()

Adds one or more new objects.

Syntax

```
1 | UpdateResult[] addResults = stub.add(new oaBase[] objects);
```

Usage

Use the `add()` command to add one or more new objects. The maximum number of objects you can add with one single call is 1,000.

The `add()` command cannot be used to add `User` objects. Use `createUser()` instead.

You can set custom field values as well as standard field values when adding objects. See [Reading or Setting Custom Field Values Inline](#).

Arguments

Name	Type	Description
objects	oaBase[]	Array of oaBase objects

Response

UpdateResult[] — Array of [UpdateResult](#) objects.

Sample Code — C#

```

1 //Define a category object to create in OpenAir
2 oaCategory category = new oaCategory();
3 category.name = "New Category";
4 category.cost_centerid = "123";
5 category.currency = "USD";
6
7 //Invoke the add call
8 UpdateResult[] results = _svc.add(new oaBase[] { category });
9
10 //Get the new ID
11 string newID = results[0].id;
```

Sample Code — Java

```

1 // Create a category object to send to the service
2 oaCategory category = new oaCategory();
3
4 // Set several properties
5 category.setName("my new category");
6
7 // Add the category to an array of oaBase objects
8 oaBase[] records = new oaBase[] { category };
9
10 // Invoke the add call
11 UpdateResult[] results = stub.add(records);
12
13 // Get the new ID
14 String newID = results[0].getId();
```

approve()

Approves one or more transactions that were submitted for approval.

Syntax

```

1 | ApproveResult[] approveResults = stub.approve(new ApproveRequest[] approveRequests);
```

Usage

Use the `approve()` command to approve transactions which were submitted for approval. The maximum number of objects you can approve with one single call is 1,000.

For information about object types for which the OpenAir XML API and SOAP API support approval-related operations, see [Approval-Related Operations](#).

Arguments

Name	Type	Description
approveRequests	ApproveRequest[]	Array of ApproveRequest objects

Response

ApproveResult[] — Array of [ApproveResult](#) objects.

Sample Code — C#

```

1 // approve an envelope
2 oaEnvelope env = new oaEnvelope();
3 env.id = "122";
4
5 oaApproval appr = new oaApproval();
6 appr.cc = "help@ddd.com"; // cc approval email to additional contacts
7 appr.notes = "Approval notes";
8
9 ApproveRequest ar = new ApproveRequest();
10 ar.approve = env;
11 ar.approval = appr;
12
13 ApproveResult[] results = _svc.approve(new ApproveRequest[] { ar });

```

Sample Code — Java

```

1 // approve an envelope which was submitted for approval
2 oaEnvelope env = new oaEnvelope();
3 env.setId("122");
4 oaApproval appr = new oaApproval();
5 appr.setCc("help@ddd.com"); // cc approval email to additional contacts
6 appr.setNotes("approval notes");
7 ApproveRequest ar = new ApproveRequest();
8 ar.setApproval(appr);
9 ar.setApprove( env );
10
11 ApproveResult[] results = stub.approve(new ApproveRequest[] { ar });

```

ApproveRequest

A `ApproveRequest` object defines the parameters for the `approve()` command.

A `ApproveRequest` object has the following properties:

Name	Type	Description
approval	oaApproval[]	An array of oaApproval objects (see Approval).
approve	oaBase[]	An array of oaBase objects. See Object Types Supporting Approval-Related Operations .
attributes	Attribute[]	An array of Attribute objects.

ApproveResult

The `approve()` command returns an array of `ApproveResult` objects.

A `ApproveResult` object has the following properties:

Name	Type	Description
approval_errors	string	A string describing any approval errors.
approval_warnings	string	A string describing any approval warnings.
id	string	The internal ID of the record submitted for approval or approved.
errors	oaError[]	An array of oaError objects (see Error). If an error occurred the <code>approve()</code> command returns an array of one or more oaError objects providing the error code and description.
status	string	-1 if the operation is unsuccessful (one or more errors occurred).

createUser()

Adds a [User](#) object.

Syntax

```
1 | UpdateResult[] createUserResults = stub.createUser(oaUser[] users, oaCompany[] company);
```

Usage

Use the `createUser()` command to add one or more new [User](#) objects. The maximum number of objects you can add with one single call is 1,000.

To add objects of types other than [User](#), use the `add()` command instead.

You can set custom field values as well as standard field values when adding [User](#) objects. See [Reading or Setting Custom Field Values Inline](#).

You can set the employee work schedules when adding [User](#) objects.



Important: Review the following guidelines:

- You must be authenticated as an account administrator to use the `createUser()` command.
- The `createUser()` command does not support foreign key lookup.
- You must set a password when using the `createUser` command **to create a new user record** except for generic user records (`generic` set to 1).
- The following applies if you are using SAML for authentication in to your OpenAir account. You can set a password and enable SAML authentication for the user (setting the Boolean custom field `saml_auth__c` to true) when using the `createUser()` command **to create a new user record**.
- Limits are enforced to prevent you from creating or activating users if doing so would exceed the number of user licenses purchased for your account. If no user licenses of the appropriate type are available, the `createUser()` command creates a new user record, but sets it as inactive (clears the **Active** box on the employee record). For more information about OpenAir licensing and compliance, see .

Arguments

Name	Type	Description
users	oaUser[]	Array of oaUser objects (see User)
company	oaCompany[1]	Array of oaCompany objects (see Company)

Response

UpdateResult[] — Array of [UpdateResult](#) objects.

Sample Code — C#

```

1 | oaCompany comp = new oaCompany();
2 | comp.nickname = "New Account"; // specify nickname of the account the
3 | user is being added to.
4 |
5 | oaUser newUser = new oaUser();
6 | newUser.nickname = "userA";
7 | newUser.role_id = "1"; // role of administrator.
8 | newUser.password = "*****";
9 | newUser.addr_email = "sss@sss.com";
10 | newUser.account_workscheduleid = "1"; // Associate a valid workschedule for the user.
11 |
12 | UpdateResult result = _svc.createUser(newUser, comp);

```

Sample Code — Java

```

1 | oaCompany comp = new oaCompany();

```

```

2 | oaUser cuser = new oaUser();
3 | comp.setNickname("openair"); // specify nickname of the account
4 | the user is being added to.
5 | cuser.setNickname("userA");
6 | cuser.setRole_id("1"); // role of administrator.
7 | cuser.setPassword("*****");
8 | cuser.setAddr_email("sss@sss.com");
9 | cuser.setAccount_workscheduleid("1"); // Associate a valid workschedule for the user.
10 |
11 | UpdateResult result = stub.createUser(cuser, comp);

```

delete()

Deletes one or more objects.

Syntax

```

1 | UpdateResult[] deleteResults = stub.delete(new oaBase[] objects);

```

Usage

Use the `delete()` command to delete one or more objects based on their internal IDs. The maximum number of objects you can delete with one single call is 1,000.

Arguments

Name	Type	Description
objects	oaBase[]	Array of oaBase objects

Response

UpdateResult[] — Array of UpdateResult objects.

Sample Code — C#

```

1 | // delete customer with internal ID 66.
2 | oaCustomer customer = new oaCustomer();
3 | customer.id = "66";
4 |
5 | UpdateResult[] deleteResults = stub.delete(new oaBase[1] { customer });

```

Sample Code — Java

```

1 | // delete customer with internal ID 66.
2 | oaCustomer customer = new oaCustomer();

```

```

3 | customer.setId("66");
4 |
5 | UpdateResult[] deleteResults = stub.delete(new oaBase[] { customer });

```

login()

Authenticates the user and starts a client session.

Syntax

```

1 | LoginResult loginResults = stub.login(LoginParams loginParameters);

```

Usage

Use the `login()` command to authenticate the user and start a client session. The call returns a unique client session identifier [sessionId]. This client session ID can be used to make subsequent calls to OpenAir SOAP API until the session expires or is ended for the authenticated user by a `logout()` call. See [Using SessionHeader for Client Session ID Authentication](#).

Arguments

Name	Type	Description
loginParameters	LoginParams	A LoginParams object.

Response

LoginResult — A [LoginResult](#) object.

Sample Code — C#

```

1 | // Create service stub
2 | OAIRServiceHandlerService _svc = new OAIRServiceHandlerService();
3 |
4 | // create LoginParam object
5 | LoginParams loginParams = new LoginParams();
6 | loginParams.api_namespace = "my namespace";
7 | loginParams.api_key = "*****";
8 | loginParams.company = "company name";
9 | loginParams.user = "username";
10 | loginParams.password = "password";
11 | loginParams.client = "my client name";
12 | loginParams.version = "1.0";
13 | LoginResult loginResult = _svc.login(loginParams);
14 |
15 | // Create a new session header object
16 | // Add the session ID returned from the login
17 | _svc.SessionHeaderValue = new SessionHeader();
18 | _svc.SessionHeaderValue.sessionId = loginResult.sessionId;

```

Sample Code — Java

```

1 // create our login parameters
2 LoginParams lp = new LoginParams();
3 lp.setUser("username");
4 lp.setPassword("password");
5 lp.setCompany("company name");
6 lp.setApi_namespace("my namespace");
7 lp.setApi_key("*****");
8 lp.setClient("my client name");
9 lp.setVersion("1.0");
10
11 // set the service URL from our arguments
12 OAIRServiceHandlerServiceLocator locator = new OAIRServiceHandlerServiceLocator();
13 locator.setOAIRServiceAddress("https://my-account-domain.app.openair.com/soap");
14
15 // now login
16 OAIRServiceSoapBindingStub binding =
17 (OAIRServiceSoapBindingStub)locator.getOAIRService();
18 LoginResult loginResult = binding.login(lp);
19
20 // Create a new session header object
21 // Add the session ID returned from the login
22 SOAPHeaderElement header = new SOAPHeaderElement("https://my-account-domain.app.openair.com/OAIRService", "SessionHeader");
23 SOAPElement node = header.addChildElement("sessionId");
24 node.addTextNode(loginResult.getSessionId());
25 binding.setHeader(header);

```

LoginParams

The `LoginParams` object defines the authentication parameters for the `login()` command.

A `LoginParams` object has the following properties:

Name	Type	Description
api_key	string	The namespace assigned to your account.
api_namespace	string	The OpenAir API key assigned to your account.
client	string	The client name.
company	string	Your company ID.
password	string	The authenticating user's password.
user	string	The authenticating user's user ID.
version	string	The client version.

LoginResult

The `login()` command returns a `LoginResult` object.

A `LoginResult` object has the following properties:

Name	Type	Description
sessionId	string	The unique client session ID.

Name	Type	Description
URL	string	Session URL for the authenticated user.

logout()

Ends the client session of the authenticated user.

Syntax

```
1 stub.logout();
```

Usage

Use `logout()` command to end the client session for the authenticated user issuing the call. No arguments are required.

Arguments

None — This command ends the session for the authenticated user issuing the call, so no arguments are needed. The authenticated user is identified by the specified in the `sessionId` in the `SessionHeader` for this call.

Response

The client session has ended. Failure of the call means that the session was logged out prior to issuing the call, so no results are needed. Any unexpected error should be handled by your client application – see [Error Codes](#).

Sample Code — C#

```
1 // This invalidates sessionId and user needs to login to make any calls to the API again.
2 stub.logout();
```

Sample Code — Java

```
1 // This invalidates sessionId and user needs to login to make any calls to the API again.
2 stub.logout();
```

makeURL()

Creates one or more URL to specific OpenAir pages.

Syntax

```
1 | MakeURLResult[] makeURLResults = stub.makeURL(MakeURLRequest[] makeURLRequests);
```

Usage

Use the `makeURL()` command to create URLs to specific OpenAir page.

Arguments

Name	Type	Description
makeURLRequests	MakeURLRequest[]	An array of MakeURLRequest objects.

Response

`makeURLResult[]` — Array of [MakeURLResult](#) objects.

Sample Code — C#

```
1 | oaEnvelope envelope = new oaEnvelope();
2 | envelope.id = "1";
3 |
4 | MakeURLRequest mur = new MakeURLRequest();
5 | mur.uid = _svc.SessionHeaderValue.sessionId;
6 | mur.app = "te";
7 | mur.page = "list-envelope-receipts";
8 | mur.arg = envelope;
9 |
10 | MakeURLResult[] results = _svc.makeURL(new MakeURLRequest[] { mur });
```

Sample Code — Java

```
1 | String sessionId = loginResult.getSessionId();
2 | MakeURLRequest make = new MakeURLRequest();
3 |
4 | make.setUid(sessionId);
5 | make.setApp("te");
6 | make.setPage( "list-envelope-receipts" );
7 | oaEnvelope envelope = new oaEnvelope();
8 | envelope.setId("1");
9 | make.setArg( envelope );
10 |
11 | // make url
12 | MakeURLResult[] mkresults = stub.makeURL(new MakeURLRequest[] { make });
```

MakeURLRequest

A `MakeURLRequest` object defines the parameters for the `makeURL()` command.

A `MakeURLRequest` object has the following properties:

Name	Type	Description
uid	string	A valid client session ID.
page	string	Code representing the page.
app	string	Two-letter code representing the application.
arg	string	Any argument.

For information about the supported pages in the OpenAir UI (identified by a navigation path) and the corresponding page, app and arg combinations, see [OpenAir Pages Supported by the Make URL Operation](#).

MakeURLResult

The `makeURL()` command returns an array of `MakeURLResult` objects.

A `MakeURLResult` object has the following properties:

Name	Type	Description
errors	oaError[]	An array of oaError objects (see Error). If an error occurred, the <code>makeURL()</code> command returns an array of one or more oaError objects (see Error) providing the error code and description.
status	string	Two-letter code representing the application.
url	string	The page URL for the authenticated user.

modify()

Updates one or more objects.

Syntax

```
1 | UpdateResult[] modifyResults = stub.modify(attributes, objects);
```

Usage

Use the `modify()` command to update one or more objects. The maximum number of objects you can add with one single call is 1,000.

You can use the `modify()` command to update a [User](#) object. See the object reference for usage limitations specific to the [User](#) object.

Determine the internal ID of each object you want to update and construct an array of objects, each containing the object internal ID and the object properties you want to update.

You can also use an external ID property as a foreign key and update an object without querying the object internal ID first. See [Related Object Lookup Using the SOAP API](#).

You can set custom field values as well as standard field values when updating objects. See [Reading or Setting Custom Field Values Inline](#).

Arguments

Name	Type	Description
attributes	Attribute[]	Array of Attribute objects. See Add, Update and Upsert Attributes .
objects	oaBase[]	Array of oaBase objects

Response

UpdateResult[] — Array of [UpdateResult](#) objects.

Sample Codes — C#

Modify — C#

Modify a customer's email address.

```

1 | oaCustomer customer = new oaCustomer();
2 | customer.id = "37";
3 | customer.addr_email = "newest@example.com";
4 |
5 | UpdateResult[] res = _svc.modify(new OA.Attribute[] {}, new oaBase[] {customer});

```

Update Using External ID as Foreign Key Lookup — C#

This modify request updates the `filterset_ids` property of user (id = 12). The API looks up the internal IDs of Filtersets which have external_ids "extrn1", "extrn2" and "extrn3" and assigns the `filterset_ids` property of the target user with the list of corresponding internal IDs, like "12,3,24".

```

1 | oaFieldAttribute lookupAttr = new oaFieldAttribute();
2 | lookupAttr.name = "external";
3 | lookupAttr.value = "filterset_ids:Filterset:1";
4 |
5 | oaUser user = new oaUser();
6 | user.id = "12";
7 | user.filterset_ids = "extrn1,extrn2,extrn3";
8 | user.attributes = new oaBase[] { lookupAttr };
9 |
10 | UpdateResult[] results = _svc.modify(new OA.Attribute[] {}, new oaBase[] { user });

```

Update Using Custom Field as Foreign Key Lookup — C#

To use custom field as a lookup field in place of internal ID, use the following syntax. The API will find the customer(s) which have `CustField12` value set to "somevalue" and update the name on matching records to "John Carr".

```

1 | oaCustomer customer = new oaCustomer();
2 | customer.name = "John Carr";

```

```

3 customer.CustField12__c = "somevalue";
4
5 //this attribute specifies which custom field should be used for lookup
6 OA.Attribute lookupAttr = new OA.Attribute();
7 lookupAttr.name = "lookup_custom";
8 lookupAttr.value = "CustField12__c";
9
10 UpdateResult[] results = _svc.modify(new OA.Attribute[] { lookupAttr }, new oaBase[] { customer });

```

Update Custom Field Value as Object Property — C#

```

1 oaProject project = new oaProject();
2 project.id = "123"; //id of record to modify
3 project.ProjectStatus__c = "Yellow"; //new custom field value
4
5 //Define attribute that directs API to update a custom field.
6 OA.Attribute updateCustom = new OA.Attribute();
7 updateCustom.name = "update_custom";
8 updateCustom.value = "1";
9
10 UpdateResult[] res = _svc.modify(new OA.Attribute[] { updateCustom },
11 new oaBase[] { project });

```

Modify Custom Field Values with custom equal to Method — C#

```

1 OA.Attribute lookupAttr = new OA.Attribute();
2 lookupAttr.name = "method";
3 lookupAttr.value = "custom equal to";
4
5 oaCustomField customField = new oaCustomField();
6 customField.type = "User"; //name of the object the field is associated with
7 customField.id = "12"; //internal ID of the user record.
8 customField.name = "cust_field_name"; //internal name of the custom field.
9 customField.value = "My new value"; // new value
10
11 UpdateResult[] updateResults = _svc.modify(new OA.Attribute[] { lookupAttr }, new oaBase[] { customField });

```

Modify ImportExport Objects and Read Not Exported — C#

Mark the envelope with ID = 4 as exported by the application MY_APP on 4/1/2008. By doing this, we can later use "not-exported" filter to read only records that have not yet been exported by MY_APP.

```

1 oaImportExport exportRecord = new oaImportExport();
2 exportRecord.application = "MY_APP";
3 exportRecord.type = "Envelope";
4 exportRecord.id = "4";
5 exportRecord.exported = "2008-04-01 00:00:00";
6 UpdateResult[] ur = _svc.upsert(new OA.Attribute[] {}, new oaBase[] { exportRecord });
7
8 //Define read parameters
9 ReadRequest rr = new ReadRequest();
10 rr.method = "all";
11 rr.type = "Envelope";
12
13 //Export only records that have not yet been exported by MY_APP
14 OA.Attribute notExportedAttr = new OA.Attribute();
15 notExportedAttr.name = "filter";
16 notExportedAttr.value = "not-exported";
17 rr.attributes = new OA.Attribute[] { notExportedAttr };
18
19 //Direct the API to filter out records exported by MY_APP

```

```

20 oaImportExport importExport = new oaImportExport();
21 importExport.application = "MY_APP";
22 rr.objects = new oaBase[] { importExport };
23
24 ReadResult[] results = _svc.read(new ReadRequest[] { rr });

```

Sample Codes — Java

Modify — Java

```

1 // Modify customer's email address
2 oaCustomer customer1 = new oaCustomer();
3 customer1.setAddr_email("new@example.com");
4 customer1.setId("66");
5
6 // Attribute not used in this case but needs to be passed in.
7 Attribute dummy = new Attribute();
8 UpdateResult[] results = binding.modify(new Attribute[] { dummy },
9 new oaBase[] { customer1 });

```

Modify with Foreign Key Lookup — Java

```

1 // For each xxxid field that needs to be looked up, set the ID field (filtersetids) to the externalid field value.
2 // Create an oaFieldAttribute object and set its members. In this example, filtersetids accepts a list of OpenAir internal IDs separated by a comma. In most
3 // cases just a single ID values is used.
4 oaFieldAttribute attr = new oaFieldAttribute();
5 attr.setName("external"); // type of lookup (external will lookup the field using external_id field)
6 attr.setValue("filtersetids:Filterset:1"); // colon separated values: field name (as it exists in the object, matching record type to process lookup
7 // for, 1 is needed to process comma separated values (it's not required for regular fields)
8
9 // Set the user field (filtersetids in this case, to the value of the externalid (instead of the internalid used normally)
10 oaUser user = new oaUser();
11 user.setId("10119");
12 user.setFiltersetids("external1,external2,external3,external4");
13
14 // notice that the field used (filtersetids) matches the first part of the attributes value.
15 // Set the attributes collection for user object. Set the attribute as part of collection.
16 user.setAttributes(new oaBase[] { attr});
17
18 // process modify
19 UpdateResult[] results = service.modify(new Attribute[] { null}, new oaBase[] {user});

```

Modify Custom Field Values with custom equal to Method — Java

```

1 // Attributes can be used to specify non-default method to update custom fields for a given object:
2 // create the attribute to specify method.
3 Attribute custom = new Attribute();
4 custom.setName("method");
5 custom.setValue("custom equal to");
6
7 // create custom field object
8 oaCustomField customf = new oaCustomField();
9 customf.setType("User"); // name of the object custom field is associated with. In this example, it is User. See table of custom equal to objects.
10
11 customf.setName("userc"); // internal name of the custom field.
12 customf.setId("1"); // internal ID of the user record
13 customf.setValue("My new value"); // custom value
14
15 UpdateResult[] updateResults = stub.modify(new Attribute[] { custom }, new oaBase[] { customf });

```

Modify Not Exported Expense Reports — Java

```

1 // To modify and read not-yet exported envelopes
2 // mark the envelope with ID = 4 as exported by the application
3 // MY_APP on 4/1/2008. By doing this, we can add a filter attribute to
4 // our read call that will filter out records exported by MY_APP.
5 oaImportExport exportRecord = new oaImportExport();
6 exportRecord.setApplication( "MY_APP" );
7 exportRecord.setType( "Envelope" );
8 exportRecord.setId( "4" );
9 exportRecord.setExported( "2008-04-01 00:00:00" );
10 UpdateResult[] ur = binding.upsert( new Attribute[] {}, new oaBase[] {
11
12     exportRecord } );
13
14 // Now do a read of all envelopes, but add a filter
15 // so we only retrieve not-yet-exported records
16 Attribute attr = new Attribute();
17 attr.setName( "filter" );
18 attr.setValue( "not-exported" );
19
20 ReadRequest read = new ReadRequest();
21 read.setMethod( "all" );
22 read.setType( "Envelope" );
23 read.setAttributes( new Attribute[] {attr} );
24
25 // tell the server we're filtering on import_export records created by MY_APP
26 oaImportExport importExport = new oaImportExport();
27 importExport.setApplication( "MY_APP" );
28 read.setObjects( new oaBase[] { importExport } );
29
30 ReadResult[] results = binding.read( new ReadRequest[] { read } );

```

read()

Retrieves one or more objects of the type specified based on the method, attributes and other parameters passed.

Syntax

```

1 | ReadResult[] ReadResults = stub.read( new ReadRequest[] ReadRequests );

```

Usage

Use the `read()` command to retrieve one or more objects of specific types.

Define the types of objects to retrieve and other request parameters using an array of [ReadRequest](#) objects.

The `read()` command returns an array of [ReadResult](#) objects.

Arguments

Name	Type	Description
ReadRequests	ReadRequest[]	Array of ReadRequest objects

Response

ReadResult[] — Array of [ReadResult](#) objects.

ReadRequest

A ReadRequest object defines the parameters for the [read\(\)](#) command.

A ReadRequest object has the following properties:

Name	Type	Description
type	string	The type of object to return. For information about supported object types, see List of Supported Business Object Types .
method	string	The name of the read method to be used. The read method lets you specify whether to return all objects, only the objects matching or not matching a search query object, or the custom field values for a specific object. See Read Methods .
objects	oaBase[]	An array of oaBase objects as arguments.
attributes	Attribute[]	An array of Attribute objects. The attributes lets you modify the response when reading objects using OpenAir SOAP API. Some attributes control API features available for most supported object types, other attributes can only be used when reading a specific object type. See Read Attributes .
fields	string	Comma-separated list of object properties to return. If empty, the response includes all standard object properties for each returned object except for the flags property for oaUser or oaCompany objects (see User and Company). It also includes all custom fields for the object type. To include the flags property for oaUser or oaCompany objects, set the include_flags attribute to 1.

ReadResult

The [read\(\)](#) command returns an array of ReadResult objects.

A ReadResult object has the following properties:

Name	Type	Description
objects	oaBase[]	An array of oaBase objects.
errors	oaError[]	An array of oaError objects (see Error). If an error occurred the read() command returns an array of one or more oaError objects (see Error) providing the error code and description.

Sample Codes — C#

Note that “limit” attribute is **always required**, but for the sake of saving space, only the first example shows the loop which correctly gets multiple batches of data.

Read with equal to Method — C#

Read the fields ID, nickname, updated for users with nickname 'jsmith'.

```

1 ReadResult[] results;
2 ReadRequest rr = new ReadRequest();
3 rr.type = "User";
4 rr.method = "equal to"; //return only records that match search
5 criteria
6 rr.fields = "id, nickname, updated"; //specify fields to be returned. //Specify search criteria
7 oaUser user = new oaUser();
8 user.nickname = "jsmith";
9 rr.objects = new oaBase[1] { user }; //pass in one object with search
10 criteria int index = 0; //Starting index
11 const int LIMIT = 1000; //Return maximum of 1000 records per request
12 do {
13     // Limit attribute is required.
14     OA.Attribute attr = new OA.Attribute();
15     attr.name = "limit";
16     attr.value = String.Format("{0}, {1}", index, LIMIT);
17     rr.attributes = new OA.Attribute[] { attr };
18     results = _svc.read(new ReadRequest[] { rr });
19     if (results != null && results.Length > 0 && results[0].errors != null) {
20         foreach (oaError err in results[0].errors) {
21             Debug.WriteLine(string.Format("Error {0} - {1}", err.code, err.text));
22         }
23     }
24     // get next 1000 records
25     index += LIMIT;
26 } while (results[0].objects != null && results[0].objects.Length > 0);

```

Read with not equal to Method — C#

Read ID, nickname, and updated fields for users that do not match certain search criteria.

```

1 ReadRequest rr = new ReadRequest();
2 rr.type = "User";
3 rr.method = "not equal to"; //return only records that do not match search criteria
4 rr.fields = "id, nickname, updated"; //specify fields to be returned
5
6 oaUser user = new oaUser();
7 user.nickname = "jsmith";
8 rr.objects = new oaBase[1] { user }; //pass in one object with search criteria
9
10 // Limit attribute is required.
11 OA.Attribute attr = new OA.Attribute();
12 attr.name = "limit";
13 attr.value = "500";
14 rr.attributes = new OA.Attribute[] { attr };
15
16 ReadResult[] results = _svc.read(new ReadRequest[1] { rr });

```

Read Custom Field Definitions with equal to Method — C#

Read all custom fields associated with project record type.

```

1 ReadRequest rr = new ReadRequest();
2 rr.method = "equal to";
3 rr.type = "CustField";
4
5 oaCustField cf = new oaCustField();
6 cf.association = "project"; // custom field association
7
8 // Limit attribute is required.
9 OA.Attribute attr = new OA.Attribute();
10 attr.name = "limit";
11 attr.value = "500";
12 rr.attributes = new OA.Attribute[] { attr };
13

```

```

14 rr.objects = new oaBase[] { cf };
15 ReadResult[] results = _svc.read(new ReadRequest[] { rr });

```

Read Custom Field Values with custom equal to Method — C#

Read custom field values for a given project.

```

1 ReadRequest rr = new ReadRequest();
2 rr.method = "custom equal to"; //all custom fields associated with the
3 project are returned
4 rr.type = "Project"; //See table of objects that have custom field
5 support
6
7 oaCustomField cf = new oaCustomField();
8 cf.id = "238"; //ID of the project
9 rr.objects = new oaBase[] { cf };
10
11 OA.Attribute attr = new OA.Attribute();
12 attr.name = "limit";
13 attr.value = "500";
14 rr.attributes = new OA.Attribute[] { attr };
15
16 ReadResult[] results = _svc.read(new ReadRequest[] { rr }); //
17 returns name/value pairs.

```

 **Note:** Review the chapter that addresses [Custom Fields](#).

Read Not Exported Charges with all Method — C#

Read all slips and add a filter to retrieve not yet reported records only.

```

1 ReadRequest rr = new ReadRequest();
2 rr.method = "all";
3 rr.type = "Slip";
4
5 OA.Attribute attrLimit = new OA.Attribute();
6 attrLimit.name = "limit";
7 attrLimit.value = "500";
8
9 OA.Attribute attrFilter = new OA.Attribute();
10 attrFilter.name = "filter";
11 attrFilter.value = "not-exported";
12
13 rr.attributes = new OA.Attribute[] { attrLimit, attrFilter };
14
15 // Tell the server we are filtering on import_export records created
16 by MY_APP
17 oaImportExport importExport = new oaImportExport();
18 importExport.application = "MY_APP";
19 rr.objects = new oaBase[] { importExport };
20
21 ReadResult[] results = _svc.read(new ReadRequest[] { rr });
22
23 // Mark the slip with ID = 4 as exported by the application MY_APP on 4/1/2011.
24 // After doing this, the next read call with not-exported filter
25 attribute will not export this record.
26 oaImportExport exportRecord = new oaImportExport();
27 exportRecord.application = "MY_APP";
28 exportRecord.type = "Slip";
29 exportRecord.id = "4";
30 exportRecord.exported = "2011-04-01 00:00:00";
31
32 UpdateResult[] ur = _svc.upsert(new OA.Attribute[] {}, new oaBase[] { exportRecord });

```

Read Not Exported Expense Reports with all Method and Date Filter — C#

Request envelope records that were approved in a certain date range and were not exported yet.

Note: Multiple filters can be used. They should be CSV concatenated in one single filter attribute. For example, to retrieve all timesheet entries in a certain date range for approved timesheets only, `attrFilter.value` should be “newer-than, older-than, approved-timesheets”.

```

1 ReadRequest rr = new ReadRequest();
2 rr.method = "all";
3 rr.type = "Envelope";
4
5 //Filter by date range and by the special not-exported flag
6 OA.Attribute attrFilter = new OA.Attribute();
7 attrFilter.name = "filter";
8 attrFilter.value = "newer-than,older-than,not-exported";
9
10 //Name of the field to apply date filter to
11 OA.Attribute attrField = new OA.Attribute();
12 attrField.name = "field";
13 attrField.value = "date_approved,date_approved";
14
15 OA.Attribute attrLimit = new OA.Attribute();
16 attrLimit.name = "limit";
17 attrLimit.value = "500";
18
19 rr.attributes = new OA.Attribute[] { attrFilter, attrField, attrLimit};
20
21 // set newer-than filter date
22 oaDate dateNewer = new oaDate();
23 dateNewer.year = "2008";
24 dateNewer.month = "10";
25 dateNewer.day = "17";
26
27 // set older-than filter date
28 oaDate dateOlder = new oaDate();
29 dateOlder.year = "2008";
30 dateOlder.month = "10";
31 dateOlder.day = "17";
32
33 rr.objects = new oaBase[] { dateNewer, dateOlder };
34 ReadResult[] results = _svc.read(new ReadRequest[] { rr });

```

Read with equal to Method and Lookup by Custom Field Value — C#

```

1 ReadRequest rr = new ReadRequest();
2 rr.method = "equal to";
3 rr.type = "Project";
4
5 //Get the first 100 records only
6 OA.Attribute attrLimit = new OA.Attribute();
7 attrLimit.name = "limit";
8 attrLimit.value = "100";
9 rr.attributes = new OA.Attribute[] { attrLimit };
10
11 //Get only records with custom field ProjectStatus set to "Yellow"
12 //Specify additional values in comma delimited list, e.g.
13 "Yellow,Green,Red"
14 //Provide empty string to get records that don't have any value set,
15 //e.g. filter.ProjectStatus_c = "";
16 //If the custom field type is Date provide default date to get records
17 that

```

```

18 //have no value, e.g. filter.ProjectStatus__c = "0000-00-00";
19 oaProject filter = new oaProject();
20 filter.ProjectStatus__c = "Yellow";
21 rr.objects = new oaBase[] { filter };
22
23 ReadResult[] results = _svc.read(new ReadRequest[] { rr });

```

Read with Multiple equal to Methods Combined with Explicit OR Condition — C#

Search for all Bookings for users with ID 5 or 6.

```

1 ReadRequest rr = new ReadRequest();
2 rr.method = "equal to, or equal to";
3 rr.type = "Booking";
4
5 OA.Attribute attrLimit = new OA.Attribute();
6 attrLimit.name = "limit";
7 attrLimit.value = "100";
8 rr.attributes = new OA.Attribute[] { attrLimit };
9
10 oaBooking book1 = new oaBooking();
11 book1.userid = "3";
12
13 oaBooking book2 = new oaBooking();
14 book2.userid = "10";
15
16 rr.objects = new oaBase[] { book1, book2 };
17 ReadResult[] results = _svc.read(new ReadRequest[] { rr });

```

Single Read() Call with Multiple Read Requests — C#

In many cases, It is not possible to retrieve all objects required in a single read request.

You should batch multiple read requests in a single read() call when the integration logic allows it.

The following code samples executes multiple read requests in a single call to the server, and counts as one single transaction against your daily account limit.

```

1 int[]idsList = new int[] { 1, 22, 1633, 32, 9, 28, 39 };
2 ReadRequest[] readRequestsList = new ReadRequest[ IDs.Length];
3 for (int i = 0; i < 1000 && i <idsList.Length; i++) {
4     ReadRequest rr = new ReadRequest();
5     rr.type = "Slip";
6     rr.method = "equal to";
7     OA.Attribute attrLimit = new OA.Attribute();
8     attrLimit.name = "limit";
9     attrLimit.value = "1";
10    rr.attributes = new OA.Attribute[] { attrLimit };
11    oaSlip slipToRead = new oaSlip();
12    slipToRead.id =idsList[i].ToString();
13    rr.objects = new oaBase[] { slipToRead };
14    readRequestsList[i] = rr;
15 }
16 ReadResult[] results = _svc.read(readRequestsList);

```

Sample Codes — Java

Note that "limit" attribute is **always required**, but for the sake of saving space, only the first example shows the loop which correctly gets multiple batches of data.

Read with equal to Method — Java

```

1 // Create a read request for an envelope with internal ID 211
2 ReadRequest[] reads = new ReadRequest[1];
3 reads[0] = new ReadRequest();
4 reads[0].setType("Envelope"); // we are requesting Envelope type
5 reads[0].setMethod("equal to"); // method to return a specific
6 envelope
7 oaEnvelope env = new oaEnvelope();
8 env.setId("211");
9 reads[0].setObjects(new oaBase[]{env});
10 int limit = 1000; // only read 1000 records at a time
11 int index = 0; // record index to start the read from
12 // add an attribute to our read, specifying the base record # (index)
13 // and the max # of records to be returned (limit)
14 Attribute attr = new Attribute();
15 attr.setName("limit");
16 attr.setValue(String.format("%1$d", limit));
17 reads[0].setAttributes(new Attribute[]{attr}); // perform the read
18 System.out.print("Fetching envelopes...");
19 ReadResult[] results = binding.read(reads); // output the results
20 while(true) {
21     int numRead = 0;
22     for (int i = 0; i < results.length; ++i) {
23         ReadResult r = results[i];
24         if (r.getObjects() != null) {
25             System.out.println("Read " + r.getObjects().length + " envelopes\n");
26             oaBase[] objs = r.getObjects();
27             for (numRead = 0; numRead < objs.length; ++numRead) {
28                 oaEnvelope envelope = (oaEnvelope)objs[numRead];
29                 System.out.println("Envelope name: " + envelope.getName());
30                 System.out.println("Envelope number: " + envelope.getNumber());
31                 System.out.println("Envelope total: " + envelope.getTotal());
32                 System.out.println();
33                 // ..etc.
34                 index++;
35             }
36             System.out.println("Read "+numRead+" envelopes");
37         } else {
38             System.out.println("Read 0 envelopes\n");
39         }
40     }
41     // if we've read up to the limit, do another read using index as our base
42     if ( numRead == limit ) {
43         System.out.println("Fetching "+ limit + " more envelopes");
44         attr.setValue(String.format("%1$d, %2$d", index, limit));
45         results = binding.read(reads);
46     } else {
47         // no more to read
48         break;
49     }
50 }

```

Read with not equal to Method — Java

Note that the “limit” attribute is required as illustrated in [Read with equal to Method — Java](#).

```

1 // Create a read request for envelopes with a non-zero total.
2 ReadRequest[] reads = new ReadRequest[1];
3 reads[0] = new ReadRequest();
4 reads[0].setType("Envelope"); // we are requesting Envelope type
5 reads[0].setMethod("not equal to"); // method to return a subset of
6 data based on search criteria. // We are searching for an envelope with total not equal to 0.
7 oaEnvelope obj = new oaEnvelope();
8 obj.setTotal("0.00");
9 reads[0].setObjects(new oaEnvelope[] { obj }); // perform the read
10 System.out.print("Fetching envelopes...");
11 ReadResult[] results = binding.read(reads); // output the results

```

```

12 for (int i = 0; i < results.length; ++i)
13 { ReadResult r = results[i]; if (r.getObjects() != null) { System.out.println("Read " + r.getObjects().length + " envelopes\n");
    oaBase[] objs = r.getObjects(); for (int j = 0; j < objs.length; ++j) { oaEnvelope env = (oaEnvelope)objs[j]; System.out.println
    n("Envelope name: " + nv.getName()); System.out.println("Envelope number: " + env.getNumber()); System.out.println("Envelope total:
    " + nv.getTotal()); System.out.println(); // ..etc. } } else { System.out.println("Read 0 envelopes\n");
14 }

```

Read Not Exported Charges with all Method — Java

Note that the “limit” attribute is required as illustrated in [Read with equal to Method — Java](#).

```

1 // Read all slips, but add a filter so we only retrieve not-yet-exported
2 records.
3 // Below is an example on how to mark items as being exported.
4 Attribute attr = new Attribute();
5 attr.setName( "filter" );
6 attr.setValue( "not-exported" );
7 ReadRequest read = new ReadRequest();
8 read.setMethod( "all" );
9 read.setType( "Slip" );
10 read.setAttributes( new Attribute[]{attr} );
11
12 // Tell the server we're filtering on import_export records created by
13 MY_APP
14 oaImportExport importExport = new oaImportExport();
15 importExport.setApplication( "MY_APP" );
16 read.setObjects( new oaBase[] { importExport } );
17
18 ReadResult[] results = binding.read(new ReadRequest[] { read });
19
20 // To modify and read not-yet exported slips
21 // Mark the slip with ID = 4 as exported by the application
22 // MY_APP on 4/1/2011. By doing this, we can add a filter attribute to
23 // our read call that will filter out records exported by MY_APP.
24 oaImportExport exportRecord = new oaImportExport();
25 exportRecord.setApplication( "MY_APP" );
26 exportRecord.setType( "Slip" );
27 exportRecord.setId( "4" );
28 exportRecord.setExported( "2011-04-01 00:00:00" );
29 UpdateResult[] ur = binding.upsert( new Attribute[] {}, new oaBase[] {
30 exportRecord });
31

```

Read with all Method and Date Filters — Java

Note that the “limit” attribute is required as illustrated in [Read with equal to Method — Java](#).

```

1 // Request envelope records updated based on a certain date.
2 ReadRequest read = new ReadRequest();
3 Attribute attr = new Attribute();
4 attr.setName( "filter" );
5 attr.setValue( "newer-than,older-than" ); // filter for records in
6 date range separated by a comma
7 Attribute field = new Attribute();
8 field.setName( "field" );
9 field.setValue( = "updated,updated" ); //one for each date object
10 separated by a comma
11 read.setMethod( "all" );
12 read.setType( "Envelope" );
13 read.setAttributes( new Attribute[]{attr});
14
15 oaDate dateNewer = new oaDate();
16 oaDate dateOlder = new oaDate();
17
18 // set newer than date.
19 dateNewer.setYear( "2008" );

```

```

20 dateNewer.setMonth( "10" );
21 dateNewer.setDay( "16" );
22
23 // set older than date.
24 dateOlder.setYear( "2008" );
25 dateOlder.setMonth( "10" );
26 dateOlder.setDay( "17" );
27
28 read.setObjects(new oaBase[] { dateNewer,dateOlder });
29 ReadResult[] results = stub.read(new ReadRequest[] { read });

```

reject()

Rejects a transaction that was submitted for approvals.

Syntax

```
1 | RejectResult[] rejectResults = stub.reject(new RejectRequest[] rejectRequests);
```

Usage

Use the `reject()` command to reject transactions such as bookings, expense reports, invoices, or timesheets which were submitted for approval. The maximum number of objects you can reject with one single call is 1,000.

For information about object types for which the OpenAir XML API and SOAP API support approval-related operations, see [Approval-Related Operations](#).

Arguments

Name	Type	Description
rejectRequests	RejectRequest[]	Array of RejectRequest objects

Response

RejectResult[] — Array of [RejectResult](#) objects.

Sample Code — C#

```

1 | // reject an envelope which was submitted for approval
2 | oaEnvelope env = new oaEnvelope();
3 | env.id = "122";
4 |
5 | oaApproval appr = new oaApproval();
6 | appr.cc = "help@ddd.com"; // cc approval email to additional contacts
7 | appr.notes = "Approval notes";
8 |
9 | RejectRequest rr = new RejectRequest();
10 | rr.reject = env;
11 | rr.approval = appr;
12 |

```

```
13 | RejectResult[] results = _svc.reject(new RejectRequest[] { rr });
```

Sample Code — Java

```
1 | // reject an envelope which was submitted for approval
2 | oaEnvelope env = new oaEnvelope();
3 | env.setId("122");
4 | oaApproval appr = new oaApproval();
5 | appr.setCc("help@ddd.com"); // cc approval email to additional
6 | contacts
7 | appr.setNotes("approval notes");
8 | RejectRequest rr = new RejectRequest();
9 | rr.setApproval(appr);
10 | rr.setReject( env );
11 |
12 | RejectResult[] results = stub.reject(new RejectRequest[] { rr });
```

RejectRequest

A `RejectRequest` object defines the parameters for the `reject()` command.

A `RejectRequest` object has the following properties:

Name	Type	Description
approval	oaApproval[]	An array of <code>oaApproval</code> objects (see Approval).
reject	oaBase[]	An array of <code>oaBase</code> objects. See Object Types Supporting Approval-Related Operations .
attributes	Attribute[]	An array of <code>Attribute</code> objects.

RejectResult

The `reject()` command returns an array of `RejectResult` objects.

A `RejectResult` object has the following properties:

Name	Type	Description
approval_errors	string	A string describing any approval errors.
approval_warnings	string	A string describing any approval warnings.
id	string	The internal ID of the record submitted for approval or approved.
errors	oaError[]	An array of <code>oaError</code> objects (see Error). If an error occurred, the <code>reject()</code> command returns an array of one or more <code>oaError</code> objects (see Error) providing the error code and description.
status	string	-1 if the operation is unsuccessful (one or more errors occurred).

servertime()

Retrieves the current system timestamp from the OpenAir servers.

Syntax

```
1 | oaDate dateNow = stub.servertime();
```

Usage

Use the `servertime()` command to get the current time on the OpenAir servers as an `oaDate` object (see [Date](#)). No arguments are required.

Arguments

None

Response

`oaDate` — A `oaDate` object (see [Date](#)).

Sample Code — C#

```
1 | oaDate dateNow = stub.servertime();
```

Sample Code - Java

```
1 | oaDate dateNow = stub.servertime();
```

submit()

Submits a transaction for approval.

Syntax

```
1 | SubmitResult[] submitResults = stub.submit(new SubmitRequest[] submitRequests);
```

Usage

Use the `submit()` command to submit transactions such as bookings, expense reports, invoices, or timesheets for approval. The maximum number of objects you can submit with one single call is 1,000.

For information about object types for which the OpenAir XML API and SOAP API support approval-related operations, see [Approval-Related Operations](#).

Arguments

Name	Type	Description
submitRequests	SubmitRequest[]	Array of SubmitRequest objects

Response

SubmitResult[] — Array of [SubmitResult](#) objects.

Sample Code — C#

```

1 // submit an envelope for approval
2 oaEnvelope env = new oaEnvelope();
3 env.id = "122";
4
5 oaApproval appr = new oaApproval();
6 appr.cc = "help@ddd.com"; // cc approval email to additional
7 appr.notes = "Approval notes";
8
9 SubmitRequest sr = new SubmitRequest();
10 sr.submit = env;
11 sr.approval = appr;
12
13 SubmitResult[] results = _svc.submit(new SubmitRequest[] { sr });

```

Sample Code — Java

```

1 // submit an envelope for approval
2 oaEnvelope env = new oaEnvelope();
3 env.setId("122");
4 oaApproval appr = new oaApproval();
5 appr.setCc("help@ddd.com"); // cc approval email to additional
6 contacts
7 appr.setNotes("approval notes");
8 SubmitRequest sub = new SubmitRequest();
9 sub.setApproval(appr);
10 sub.setSubmit( env );
11
12 SubmitResult[] results = stub.submit(new SubmitRequest[] { sub });

```

SubmitRequest

A [SubmitRequest](#) object defines the parameters for the [submit\(\)](#) command.

A [SubmitRequest](#) object has the following properties:

Name	Type	Description
approval	oaApproval[]	An array of oaApproval objects (see Approval).
submit	oaBase[]	An array of oaBase objects. See Object Types Supporting Approval-Related Operations .
attributes	Attribute[]	An array of Attribute objects. See Approval-Related Operations .

SubmitResult

The `submit()` command returns an array of `SubmitResult` objects.

A `SubmitResult` object has the following properties:

Name	Type	Description
approval_errors	string	A string describing any approval errors.
approval_warnings	string	A string describing any approval warnings.
id	string	The internal ID of the record submitted for approval.
errors	oaError[]	An array of oaError objects (see Error). If an error occurred, the <code>submit()</code> command returns an array of one or more oaError objects (see Error) providing the error code and description.
status	string	-1 if the operation is unsuccessful (one or more errors occurred).

unapprove()

Unapproves one or more transactions that were previously approved.

Syntax

```
1 | UnapproveResult[] unapproveResults = stub.unapprove(new UnapproveRequest[] unapproveRequests);
```

Usage

Use the `unapprove()` command to unapprove transactions which were previously approved. The maximum number of objects you can unapprove with one single call is 1,000.

Transactions that were approved and subsequently billed or archived cannot be unapproved.

For information about object types for which the OpenAir XML API and SOAP API support approval-related operations, see [Approval-Related Operations](#).

Arguments

Name	Type	Description
unapproveRequests	UnapproveRequest[]	Array of UnapproveRequest objects

Response

`UnapproveResult[]` — Array of [UnapproveResult](#) objects.

Sample Code — C#

```

1 // unapprove an envelope which has already been approved
2 oaEnvelope env = new oaEnvelope();
3 env.id = "122";
4
5 oaApproval appr = new oaApproval();
6 appr.cc = "help@ddd.com"; // cc approval email to additional contacts
7 appr.notes = "Approval notes";
8
9 UnapproveRequest ur = new UnapproveRequest();
10 ur.unapprove = env;
11 ur.approval = appr;
12
13 UnapproveResult[] results = _svc.unapprove(new UnapproveRequest[] { ur });

```

Sample Code — Java

```

1 // Unapprove an envelope which has already been approved
2 oaEnvelope env = new oaEnvelope();
3 env.setId("122");
4 oaApproval appr = new oaApproval();
5 appr.setCc("help@ddd.com"); // cc approval email to additional
6 contacts
7 appr.setNotes("approval notes");
8 UnapproveRequest ur = new UnapproveRequest();
9 ur.setApproval(appr);
10 ur.setUnapprove( env );
11
12 UnapproveResult[] results = stub.unapprove(new UnapproveRequest[] { ur });

```

UnapproveRequest

A `UnapproveRequest` object defines the parameters for the `unapprove()` command.

A `UnapproveRequest` object has the following properties:

Name	Type	Description
approval	oaApproval[]	An array of <code>oaApproval</code> objects (see Approval).
unapprove	oaBase[]	An array of <code>oaBase</code> objects. See Object Types Supporting Approval-Related Operations .
attributes	Attribute[]	An array of <code>Attribute</code> objects.

UnapproveResult

The `unapprove()` command returns an array of `UnapproveResult` objects.

A `UnapproveResult` object has the following properties:

Name	Type	Description
approval_errors	string	A string describing any approval errors.
approval_warnings	string	A string describing any approval warnings.

Name	Type	Description
id	string	The internal ID of the record unapproved.
errors	oaError[]	An array of oaError objects (see Error). If an error occurred, the <code>unapprove()</code> command returns an array of one or more oaError objects (see Error) providing the error code and description.
status	string	-1 if the operation is unsuccessful (one or more errors occurred).

upsert()

Adds or updates one or more objects.

Syntax

```
1 | UpdateResult[] upsertResults = stub.upsert(attributes, objects);
```

Usage

Use the `upsert()` command to add or update one or more objects. The maximum number of objects you can upsert with one single call is 1,000.

Upserts use a lookup field to determine whether it adds or updates an object:

- If the lookup field is not matched in any existing objects, a new object is added.
- If the lookup field is matched one time, the existing object is updated.

The default lookup field is the object internal ID. You can also use an external ID property as a foreign key and update an object without querying the object internal ID first. See [Related Object Lookup Using the SOAP API](#).

You can set custom field values as well as standard field values when updating objects. See [Reading or Setting Custom Field Values Inline](#).

Arguments

Name	Type	Description
attributes	Attribute[]	Array of Attribute objects. See Add, Update and Upsert Attributes .
objects	oaBase[]	Array of oaBase objects

Response

UpdateResult[] — Array of [UpdateResult](#) objects.

Sample Code — C#

```
1 | //Define a category object to create/update in OpenAir
```

```

2 | oaCategory category = new oaCategory();
3 | category.name = "Updated Category";
4 | category.externalid = "555";
5 |
6 | // Specify that the lookup is done by external_id and not by (default) internal ID
7 | OA.Attribute attrLookup = new OA.Attribute();
8 | attrLookup.name = "lookup";
9 | attrLookup.value = "externalid";
10 |
11 | // Invoke the upsert call, passing and saving the results in an UpdateResult object
12 | UpdateResult[] results = _svc.upsert(new OA.Attribute[] { attrLookup }, new oaBase[] { category });

```

Sample Code — Java

```

1 | // upsert call
2 | // Create a category object to send to the service
3 | oaCategory category = new oaCategory();
4 |
5 | // Set several properties
6 | category.setName("SOAP created category 12/4");
7 | category.setExternalid("555");
8 |
9 | // Specify lookup attribute
10 | Attribute lookup = new Attribute();
11 | lookup.setName("lookup");
12 | lookup.setValue("externalid");
13 |
14 | // Add the account to an array of oaBase objects
15 | oaBase[] records = new oaBase[] { category };
16 |
17 | // Invoke the upsert call, passing
18 | // and saving the results in a UpdateResult object
19 | UpdateResult[] results = stub.upsert(new Attribute[] { lookup }, records);

```

version()

Gets the version of a thin client application supported by OpenAir

Syntax

```
1 | VersionResult version = stub.version(name, number);
```

Usage

The version() command is used to get the current version of a thin client application supported by OpenAir.

Arguments

Name	Type	Description
name	string	The name of the client application.
number	string	The version number.

Response

VersionResult — A [VersionResult](#) object.

Sample Code — C#

```
1 | VersionResult version = stub.version("My app", "1.1");
```

Sample Code — Java

```
1 | VersionResult version = binding.version("My app", "1.1");
```

VersionResult

The [version\(\)](#) command return a VersionResult object.

A VersionResult object has the following properties:

Name	Type	Description
number	string	Current version number.
size	string	Size of the file to download.
url	string	URL of the current version of the client installation.

whoami()

Retrieves information about the authenticated user.

Syntax

```
1 | oaUser user = stub.whoami();
```

Usage

Use the [whoami\(\)](#) command to get the authenticated user. No arguments are required.

Arguments

None

Response

oaUser — An oaUser object. See [User](#).

Sample Code — C#

```
1 | oaUser user = stub.whoami();
```

Sample Code — Java

```
1 | oaUser user = stub.whoami();
```

Business Object Properties Overview

Most object properties in the OpenAir XML API and SOAP API are simple type properties. Exceptions include [Date Fields](#), [Address Fields](#), and [Company and User Settings](#).

The OpenAir WSDL and XSD define every simple type property as a string.

Account administrators can define custom fields for supported record types in the OpenAir UI. For information about record that support custom fields, see the help topic [Record Types and Forms Supporting Custom Fields](#). For information about working with custom fields in OpenAir XML API or SOAP API calls, see [Custom Fields](#).

All SOAP business objects (complex types) except [Address](#), [Date](#), [oaFieldAttribute](#) and [Module](#) have an additional attributes property – A collection of additional attributes for this complex type.

Refer to the following help topics for more information about different type of object properties:

- [System Fields](#)
- [Calculated Fields](#)
- [Required Fields](#)
- [Reference Fields](#)
- [External ID Fields](#)
- [Date Fields](#)
- [Address Fields](#)
- [Company and User Settings](#)
- [Money Fields](#)

System Fields

Most object types have the following system-generated properties. These properties are always read-only. These fields are automatically updated during API operations and cannot be modified. For example, the internal ID [id] property is automatically generated when you add an object and the last modified date [updated] property is automatically updated when you modify an object.

Property	Database Field Type	Description
id	INTEGER	<p>With rare exceptions, all object types in the API have an internal ID property. The property value is a unique identifier for each object of that type. It is analogous to a primary key in relational databases.</p> <p>The internal ID value is generated automatically when a new object is added. It is an integer number incremented by 1 for each object of the same type. The value is unique and stays the same for the lifetime of the object. It cannot be modified.</p> <p>You can identify any specific object by its internal ID value in subsequent API calls. All API commands reading, updating, or deleting business data, can or in some cases must include an argument object that includes the id property, which references the unique object to retrieve, update, or delete.</p> <p>Many objects have one or more properties referencing a related object by its internal ID value. See Reference Fields.</p>

Property	Database Field Type	Description
created	DATETIME	Eastern Time (UTC - 5) date and time when the object was created.
updated	TIMESTAMP	Eastern Time (UTC - 5) date and time when the object was last modified.
deleted	CHAR(1)	<p>A 0 or 1 flag that indicates whether the object has been deleted. The value is 1 if the object was deleted.</p> <p>Objects marked as deleted and last updated 180 or more days ago are removed permanently from the database according to a routine schedule. See the help topic Data Deletion.</p> <p>Only a few objects in the OpenAir API have this property. However, the Read (XML API) and read() (SOAP API) commands with the <code>deleted</code> attribute can be used to read objects that are flagged as deleted. See Read Attributes.</p>
audit	MEDIUMTEXT	Audit trail information. Object types in the OpenAir API with the exception of Reimbursement do not include this property. Object audit trail information in OpenAir include information about the interface (such as OpenAir UI, XML API, SOAP API, REST API, for example) used to make the changes.

Calculated Fields

Calculated fields are read-only properties. OpenAir calculates the value for these properties automatically based on other information in OpenAir and updates the values when this information changes.

OpenAir XML API and SOAP API do not support doing a query on calculated fields. You cannot retrieve objects matching or not matching specific calculated field values using the [Read](#) (XML API) and [read\(\)](#) (SOAP API) with the `equal` or `not equal` methods.

Required Fields

Required fields must have a non-null value.

- OpenAir sets the data for certain required fields automatically such as the internal ID and other system fields when creating or updating objects.

Similarly, if a required field has a default value and a value is not passed for this property when creating an object, OpenAir implicitly sets the default value for this field.

For all other required fields, the client application must explicitly assign a non-null value when creating the object.

- When updating an object, a required field cannot be set to null, and some required fields cannot be changed.

Some fields are required as standard. For most objects, the name or nickname is a required field, for example. Other required fields depend on the business logic configured for your company's OpenAir account.

Note: Access to certain object types and object properties depend on the business logic configured for the OpenAir account. It may vary depending on the role and access privileges associated with the authenticated user.

Required and read-only properties also depend on the business logic configured for each specific OpenAir account. Some properties such as `id`, `created`, and `updated` are system-generated and always read-only.

Reference Fields

Many objects have one or more properties referencing a related object of a certain type by its internal ID value.

- Typically, the property name for a reference field ends with the letters `id` and references the object type of the related object, noted as per the name of the database table corresponding to this object type. For `Envelope` objects, for example, the `userid` property references the employee [`user`] who created and submitted the expense report [`Envelope`].

Note: In many cases, the column name for a reference field in the database ends with an underscore (`_`) and the letters `id`, and the property name for that reference field in the OpenAir XML API and SOAP API ends with the letters `id` without underscore. There are exceptions to this rule. Refer to the [XML and SOAP API Business Object Reference](#), WSDL and XSD to validate your syntax.

- Some object types allow for parent-child relationships between objects of the same types. An object can reference another object of the same type, and the property name typically indicates this relationship (`parentid` or `portfolio_projectid` for example).
- Some object types have compound reference fields that allow to specify the object type as well as the internal ID of the related object. An `Attachment` object, for example, has an `owner_type` property specifying the type of object the file is attached to and an `ownerid` referencing one object of the specified object type by its internal ID.
- Other properties with names that do not end with the letters `id` can reference a related object by internal ID. Typically, the names of these other properties indicate the type of relationship. If the property holds the internal ID of an employee approving a certain type of transactions, for example, the property name includes a two-letter code representing the transaction type, such as `ta` for timesheets, and the word `approver` (`ta_approver`).

Whereas the internal ID field [`id`] is analogous to a primary key in relational databases and cannot be modified, reference fields are analogous to foreign keys and can be modified using a command performing a create or update operation.

The reference field value must either be:

- A valid internal ID (the internal ID of an existing object of the appropriate object type in your company's account data).
- In some instances such as the `approver` properties, for example, a negative integer representing a metavalue. See the property descriptions for accepted values.
- An empty value, which indicates an empty reference, if the reference field is not a required field.

After you read an object including reference field properties, you can read each objects referenced by internal ID to get additional information about related objects.

When you add, update or upsert an object, you can use a related object lookup to set the internal ID of any related object indirectly if you know the external ID or the name of this related object. See [Adding, Updating and Upserting Objects](#).

For some object types, you cannot delete an object if this object is referenced by another object. For more information about dependencies preventing deletion, see [XML and SOAP API Business Object Reference](#) and the usage guidelines for individual business object type.

Note: Access control rules for the authenticated users may restrict viewing or editing the referenced objects.

External ID Fields

Most objects also have an external ID property [externalid] that can be used to store the internal ID of the matching object in an external system.

Note: Typically, the column name for an external ID field in the database is external_id (with an underscore) and the property name in the OpenAir API is externalid (without underscore).

Date Fields

Date properties are defined as complex type properties in the XSD and as simple type properties in the WSDL.

OpenAir XML API handles dates as compound fields using the [Date](#) element.

OpenAir SOAP API handles every date field as a string. Most date values use the format YYYY-MM-DD HH:MM:SS.

Date

Use the date [Date] object to add or update date or datetime information.

The OpenAir XML API uses the Date object as a substructure to pass date and time information when reading, adding or updating any type of records including such information.

The OpenAir XML API and SOAP API use the Date object to specify a comparison date for date filters. See [Filtering](#) and [Date Filters Usage](#).

The XML API command [Time](#) and SOAP API command [servertime\(\)](#) return the current time on the OpenAir servers as a Date object.

—	XML	SOAP
Object	Date	oaDate

The Date object has the following properties:

XML / SOAP	Database	Description
year	—	Year (YYYY). Must be a four-digit number.

XML / SOAP	Database	Description
month	—	Month (MM). Must be a two-digit number.
day	—	Day (DD). Must be a two-digit number.
hour	—	Hour (HH). Must be a two-digit number.
minute	—	Minute (MM). Must be a two-digit number.
second	—	Second (SS). Must be a two-digit number.

Address Fields

Address properties are defined as complex type properties in the XSD and OpenAir XML API handles addresses as compound fields using the [Address](#) element.

- OpenAir XML API also uses the Address object as a substructure to pass address and other contact information when reading, adding or updating any type of records including such information.
- The following example shows the XML structure to modify the city element of a contact's address

```

1 | <Contact>
2 |   <addr>
3 |     <Address>
4 |       <city>Boston</city>
5 |     </Address>
6 |   </addr>
7 | </Contact>

```

OpenAir SOAP API handles the different elements forming an address as individual component fields, which are defined as simple type properties in the WSDL.

Usage Guidelines

Review the following guidelines:

- When reading [Company](#), [Contact](#), [Customer](#), [CustomerProspect](#), [User](#) or [Vendor](#) objects you can list the specific address information to be returned.
 - XML — List the address information required between the address object property tags as per the following example.

```

1 | <_Return>
2 |   <addr>
3 |     <city/>
4 |   </addr>
5 |   <contactaddr>
6 |     <email/>
7 |     <mobile/>
8 |   </contactaddr>
9 |   <name/>
10 |   <id/>
11 | </_Return>

```

Address object properties include addr for [Company](#), [Contact](#), [Customer](#), [CustomerProspect](#), [User](#) or [Vendor](#) objects, as well as billingaddr and contactaddr for [Customer](#) and [CustomerProspect](#).

- SOAP — List the address information required in the fields of the [ReadRequest](#) complex type.
- When adding or modifying address information using the XML API, the address value between object property tags must be an [Address](#) object.

Company and User Settings

Properties containing company and user settings [flags] are defined as complex type properties in both WSDL and XSD.

OpenAir XML API handles the flags property using a series of [Flag](#) object elements.

OpenAir SOAP API handles the flags property using an array of [oaSwitch](#) objects.

Company settings are the account configuration settings controlled by account administrators in the OpenAir Administration module. They include settings that impact the company's OpenAir account overall functionality (global settings) and settings that impact the functionality of a specific module (application settings) for all users of the company's OpenAir account.

User settings are the settings controlled by account administrators on the employee record in OpenAir. Users may also control some of these settings in their personal settings. These settings control individual access privileges and other options for individual users of your company's OpenAir account.

The setting name is used to reference the setting in the OpenAir software and contains only alphanumeric or underscore characters. To verify the name of a setting in the [Flag](#) or [oaSwitch](#) object, go to the page for that setting in OpenAir then use the developer tools in your browser to inspect the label for the setting. The Inspector pane or window shows the HTML code for the page you are viewing with the element you are inspecting highlighted. The element should read as follows, with the setting name showing in between quotation marks.

```
<label for="setting_name">Setting label</label>
```

You can choose to include or exclude company or user settings (flags property) when reading [Company](#) or [User](#) objects. See [Read Attributes](#).

- When using the **XML API**, the company or user settings (flags property) are included by default in the response. Set the `exclude_flags` attribute to 1 to exclude account or user settings from the response when reading [Company](#) or [User](#) objects using the `equal` to or `not equal` to method.
- When using the **SOAP API**, the company or user settings (flags property) are excluded by default from the response. Set the `include_flags` attribute to 1 to include account or user settings in the response.

Note: Not all user settings can be controlled using the flags property. Some user settings are controlled using the [Preference](#) object.

OpenAir XML API also uses the [Flag](#) object for returning role permissions ([Role](#) object and permissions object property).

For more information about company and user settings, see the help topic [Administrator Guide](#).

For more information about user personal settings, see the help topic [Personal Settings](#).

Flag

Use the `Flag` (XML API) or `oaSwitch` (SOAP API) object to read, add or update company or user settings.

The OpenAir XML API and SOAP use the `Flag` (XML API) or `oaSwitch` (SOAP API) object as a substructure to pass company or user setting information when reading, adding or updating the [Company](#) record or [User](#) records.

—	XML	SOAP	REST	Database table
Object	Flag	oaSwitch	—	switch

The `Flag` (XML API) or `oaSwitch` (SOAP API) object has the following properties:

XML	SOAP	Database	Description
id	—	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	name	The name of the switch setting.
setting	setting	setting	The contents of the switch setting. A zero length field is considered 'undef'.

oaSwitch

See [Flag](#)

Money Fields

Money fields for an object contain monetary values in a given currency. The currency property defines the currency used for all the money fields for this object.

If a currency is not set for the object, money fields for an object contain monetary values in the default currency for your company's OpenAir account.

For [Ticket](#) objects, the currency can be set to the three-letter code of any currencies supported by OpenAir. All OpenAir accounts include the capability to create expense receipts in any of the supported

currencies even if all your business transactions are done in the default currency for your company's OpenAir account.

For other objects of other types:

- If the Multicurrency feature is enabled for your company's OpenAir account, the currency can be set to the three-letter code of any of the transaction currencies set up for your company's account. See the help topic [Enabling Multiple Currencies in OpenAir](#).
- Otherwise, the currency is always the default currency for your company's OpenAir account.

Note: There are three object types you can use in the OpenAir XML API and SOAP API to interact with foreign currency exchange information:

- Use the [Currency](#) object to set or read custom exchange rates. This is the equivalent of reading or setting custom exchange rates in Administration > Global Settings > Organization > Currencies > Set Exchange Rate when using the OpenAir UI. The exchange rates in the Currency object and in the [currency](#) table are quoted against the default currency for the account. The default currency for the company's OpenAir account is defined by the `base_currency` property for the Company object.
- Use the [Currencyrate](#) to read current or historical exchange rates used for the account.
- Use the [ForexInput](#) object to add or modify entries in the exchange cross rate table when the Multicurrency feature is enabled for your company's account. Make sure you review the [Usage Guidelines](#) for the ForexInput object. This is the equivalent of editing exchange cross rates using the [Edit Exchange Cross Rates](#) in the OpenAir UI.

Custom Fields

Account administrators can define custom fields for supported record types in the OpenAir UI. For information about record that support custom fields, see the help topic [Record Types and Forms Supporting Custom Fields](#). For more information about defining and using custom fields in the OpenAir UI, see the help topic [Custom Fields](#).

Custom Field Definitions

You can use the OpenAir XML API or SOAP API to:

- Read custom field definitions. To read the definitions of all custom fields for a specific object type, for example, read [CustField](#) objects with `equal` to method and an argument [CustField](#) object including only the association property.
- Update the **Value list** [`valuelist`] in the custom field definition ([CustField](#) object), depending on the custom field type.

Note: Client applications cannot use the OpenAir XML API or SOAP API to:

- Add new custom fields.
- Update custom field definitions except for **Value list** [valuelist].
- Delete custom fields.

Custom Field Values

You can use the OpenAir XML API or SOAP API to:

- Read custom field values inline with standard object property values when reading objects. See [Reading or Setting Custom Field Values Inline](#).
- Set custom field values inline with standard object property values when adding or updating objects. See [Reading or Setting Custom Field Values Inline](#).
- Look up and update objects matching a custom field value. See [Look Up and Update Objects Matching a Custom Field Value](#)
- Read only custom field values (`CustomField` objects) for the object referenced by internal ID using the `custom equal to` method. See [Read Methods](#).
- Update only custom field values (`CustomField` objects) for the object referenced by internal ID using the `custom equal to` method.

Reading or Setting Custom Field Values Inline

Note: When reading or setting custom field values inline, custom fields are referenced by the custom field name with a suffix of two underscores followed by a lowercase c character. See [Naming Conventions for Custom Fields](#).

Review the instructions in the following tables to read or set custom field values inline when reading, adding or updating an object using the **XML API**:

Command	Instructions
Read	<p>Set the <code>enable_custom</code> attribute to 1.</p> <p>List both standard object properties and custom fields to be returned in the <code>_Return</code> argument, if passed. If omitted or empty, the response includes all standard object properties and custom field values for each returned object.</p> <p>List both standard object property values and custom field values to match or not to match in the search query argument object when using the <code>equal to</code> or <code>not equal to</code> methods.</p>
Add	Set the <code>enable_custom</code> attribute to 1.
CreateUser	Include all standard object properties and custom fields you want to set in the argument object.
Modify	

Review the instructions in the following tables to read or set custom field values inline when reading, adding or updating objects using the **SOAP API**:

Command	Instructions
<code>read()</code>	Use your account-specific WSDL. List both standard object properties and custom fields to be returned in the <code>fields</code> property of each <code>ReadRequest</code> object passed. If omitted or empty, the response includes all standard object properties and custom field values for each returned object. List both standard object property values and custom field values to match or not to match in the search query argument object when using the <code>equal</code> to or <code>not equal</code> to methods.
<code>add()</code> <code>createUser()</code>	Use your account-specific WSDL. Include all standard object properties and custom fields you want to set in each argument object.
<code>modify()</code>	Use your account-specific WSDL. Set the <code>update_custom</code> attribute to 1. Include all standard object properties and custom fields you want to set in each argument object.

Note: Setting custom fields inline is not possible when using the `upsert()` call.

Look Up and Update Objects Matching a Custom Field Value

Review the instructions in the following tables to look up and modify objects matching a custom field value using the **SOAP API**:

Command	Instructions
<code>modify()</code>	Set the <code>lookup_custom</code> attribute to the custom field name with a suffix of two underscores immediately followed by a lowercase <code>c</code> character. See Naming Conventions for Custom Fields . Include the custom field you want to match instead of the internal ID.

Note: The `lookup_custom` attribute cannot be used with an `upsert()` call.

Naming Conventions for Custom Fields

Account administrators define the name of most custom fields. Each custom field must have a unique name. Some OpenAir features use custom fields that are either added automatically when enabling the feature or that needs to be created by an account administrator before the feature can be used.

When reading or updating `CustField` or `CustomField` objects, the custom field name is the same as the **Name** on the custom field properties form in the OpenAir UI.

When reading or setting custom fields values inline with standard object properties, the OpenAir API references custom fields by the name with a suffix of two underscores immediately followed by a lowercase `c` character. For example, a custom field with the name `my_custom_field` is listed as `my_custom_field__c` in the account-specific WSDL.

Note: XML naming rules apply to custom field names.

- The name must start with a letter or an underscore.
- The name cannot start with the letters xml in any letter case combination.

The custom field names such as 1MyCustomField and xmlMyCustomField, for example, cannot be read or set inline with standard object properties and are not included in the account-specific WSDL.

Uniqueness for Custom Fields

Custom fields for an object type can be set up to accept unique values across all the objects of this type in the OpenAir UI, depending on the type of custom field. Uniqueness is useful for a custom field holding a foreign key such as an object ID in a system outside of OpenAir, for example. To find out whether the custom field values must be unique, read the custom field definition ([CustField](#) object type) and inspect the value of the unique property. If the unique property is set to 1, custom field values must be unique. If the unique property is set to 0, the custom field can have the same value in different objects.

If you try to set a duplicate value when adding or updating an object and the custom field values must be unique, the object is saved and all standard object properties and custom fields are saved except those failing the uniqueness validation. The API response includes error status 1106 and an error code 1104 for each custom field that failed the uniqueness validation.

Default Values in Custom Fields

Custom fields can be set up to have a default value, depending on the type of custom field. When adding objects using the OpenAir API, a custom field will be set to its default value unless a valid value is passed for the custom field.

Value Lists for Custom Fields

Custom fields can be set up to have a list of possible values, depending on the type of custom field. The custom field must be set to one of the listed value except in the case of Dropdown and Text custom fields. If you try to set the custom field to a value that is not in the value list when adding or updating an object, the object is saved and all standard object properties and custom fields are saved except those failing the accepted values validation. The API response includes error status 1106 and an error code 1105 for each custom field for which the value passed is not in the value list for this custom field.

Note: The **Value list** [valueList] is the only custom field definition ([CustField](#)) property that can be modified using the OpenAir XML API or SOAP API.

Custom Field Value Format

Some custom field types only accept values in a specific format. For example, allocation grid custom field values are set as a string with each value-number pairs on a separate line.

Sample Code (SOAP API) – C#

```
myProject.my_allocation_grid_c = "Marc Collins,0\nBill Carter,0\nMarie Porter,50\nEd Ellis,50";
```

Sample Code (XML API)

```
1 | <my_allocation_grid_c>"Marc Collins",0
2 | "Bill Carter",0
3 | "Marie Porter",50
4 | "Ed Ellis",50</my_allocation_grid_c>
```

CustomField

The API returns CustomField (XML API) or oaCustomField (SOAP API) objects when you use the [Read](#) (XML API) or `read()` (SOAP API) with the `custom equal` to method to read custom field values for an object.

You can use the [Modify](#) (XML API) or `modify()` (SOAP API) with the `custom equal` to method and CustomField (XML API) or oaCustomField (SOAP API) objects as arguments to set or update custom field values for an object.

The CustomField or oaCustomField object has the following properties:

XML / SOAP	Database	Description
id	—	Unique ID. Automatically assigned by OpenAir.
name	—	The name of the custom field.
type	—	Association of the custom field. See the help topic Record Types and Forms Supporting Custom Fields for a list of associations.
value	—	Value of the field for a specific record.

XML and SOAP API Business Object Reference

This section provides reference information for business objects supported by the OpenAir XML API and SOAP API.

Note: This section provides reference information for supported **business objects** only. These are the objects that are used to pass your OpenAir data.

Except where indicated otherwise, all listed business object types are supported in both the XML API and SOAP API. Business object type names in the SOAP API include the prefix oa.

OpenAir SOAP API also uses **method objects** (complex types) to hold parameters for SOAP requests and return values in SOAP responses. For reference information about the SOAP API commands and method objects, see [SOAP API Commands](#).

The term "object" is equivalent to the term "record" and describes a particular occurrence of an object type. Depending on the object type, an object represents an entity –such as a customer, a project, or an employee, for example–, a transaction –such as a receipt, expense report, or an invoice, for example–, or a relationship between other objects. The term "object property" is equivalent to the term "field". OpenAir object types and object properties are analogous to database tables and the table columns. An object is analogous to a row in a database table.

For a list of supported business object types, see [List of Supported Business Object Types](#).

Reference information for each business object types include:

- The name of the object types in OpenAir XML API and SOAP API, the equivalent REST API object, where supported, and the corresponding table in the OpenAir database as documented in the [OpenAir Data Dictionary](#).
- A short description of what an object of that type represents.
- The operations you can perform on objects of that type. Most of the objects accessible through the API are read-write objects. However, there are a few objects that are read-only.
- A table listing the object properties in OpenAir XML API, SOAP API and REST API, and the corresponding column in the OpenAir database table as documented in the [OpenAir Data Dictionary](#). For information and guidelines about the main types of object properties, including custom fields, see [Business Object Properties Overview](#).
- Usage guidelines for the business object type, where relevant. These usage guidelines are not exhaustive.

Important: Your applications work with only the objects that you are authorized to access. Programmatic access to objects is determined by your company's OpenAir account configuration, the user role permissions and access settings configured by your account administrator, and other factors related specifically to the object.

List of Supported Business Object Types

The following table lists the business object types available in OpenAir XML API, and SOAP API with a link to additional information about each object.

For each business object type, the table also shows:

- The equivalent REST API object, where supported.
- The corresponding table in the OpenAir database as documented in the [OpenAir Data Dictionary](#).
- A summary of supported CRUD operations (Create, Read, Update, Delete). For information about each operation type

 **Note:** Review the usage guidelines in the help topic for the business object where provided. Some operations may be subject to limitations. Asterisks (*) in CRUD operation columns indicate that the operation is supported in special cases only.

For a list of objects supporting approval operations (submit, approve, reject, unapprove), see [Object Types Supporting Approval-Related Operations](#).

XML	SOAP	REST	Database	Create	Read	Update	Delete
AccountingPeriod	oaAccountingPeriod	—	accounting_period	✓	✓	✓	✓
Actualcost	oaActualcost	—	actual_cost	✓	✓	✓	—
Address	oaAddress	—	address	✓	✓	✓	✓
Agreement	oaAgreement	—	agreement	✓	✓	✓	✓
Agreement_to_project	oaAgreement_to_project	—	agreement_to_project	✓	✓	✓	✓
ApprovalLine	oaApprovalLine	—	approval	—	✓	—	—
ApprovalProcess	oaApprovalProcess	—	approvalprocess	✓	✓	✓	✓
Attachment	oaAttachment	Attachment <i>Must be accessed using endpoints for the parent object — See the help topic Attachments</i>	attachment	✓	✓	✓*	✓
Attribute	oaAttribute	—	attribute	—	✓	—	—
Attribute Description	oaAttribute Description	—	attribute_description	✓	✓	✓	✓
Attributeset	oaAttributeset	—	attribute_set	—	✓	—	—
BillingSplit	oaBillingSplit	—	billing_split	—	✓	—	—
Booking	oaBooking	—	booking	✓	✓	✓	✓
BookingByDay	oaBookingByDay	—	booking_by_day	—	✓	—	—
BookingType	oaBookingType	—	booking_type	✓	✓	✓	—
Booking_request	oaBooking_request	—	booking_request	—	✓	—	—
Budget	oaBudget	—	budget	✓	✓	✓	—
BudgetAllocation	oaBudgetAllocation	—	budget_allocation	✓	✓	✓	—
Category	oaCategory	—	category	✓	✓	✓	✓
Category_<N>	oaCategory_<N>	—	category_<N> (category_1)	✓	✓	✓	✓
Ccrate	oaCcrate	—	cc_rate	—	✓	—	—
Company	oaCompany	—	account	—	✓	✓	—

XML	SOAP	REST	Database	Create	Read	Update	Delete
Contact	oaContact	Contact See the help topic Contacts	contact	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Costcategory	oaCostcategory	—	cost_category	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Costcenter	oaCostcenter	—	cost_center	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Costtype	oaCosttype	—	cost_type	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Currency	oaCurrency	—	currency	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Currencyrate	oaCurrencyrate	—	—	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
CustField	oaCustField	—	cust_field	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Customer	oaCustomer	—	customer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
CustomerLocation	oaCustomerLocation	—	customer_location	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Customerpo	oaCustomerpo	—	customerpo	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Customerpo_to_project	oaCustomerpo_to_project	—	customerpo_to_project	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
CustomerProspect	oaCustomer	—	customer	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Deal	oaDeal	—	deal	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Dealcontact	oaDealcontact	—	deal_contact	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Dealschedule	oaDealschedule	—	deal_schedule	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Department	oaDepartment	—	department	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Entitytag	oaEntitytag	—	entity_tag	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Envelope	oaEnvelope	ExpenseReport See the help topic Expense Reports	envelope	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Estimate	oaEstimate	—	estimate	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Estimateadjustment	oaEstimateadjustment	—	estimate_adjustment	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Estimateexpense	oaEstimateexpense	—	estimate_expense	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Estimatelabor	oaEstimatelabor	—	estimate_labor	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Estimatemarkup	oaEstimatemarkup	—	estimate_markup	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Estimatephase	oaEstimatephase	—	estimate_phase	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Event	oaEvent	—	event	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
ExpensePolicy	oaExpensePolicy	—	expense_policy	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
ExpensePolicyItem	oaExpensePolicyItem	—	expense_policy_item	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Filter	—	—	filter	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Filterset	oaFilterset	—	filter_set	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
ForexInput	oaForexInput	—	—	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Fulfillment	oaFulfillment	—	fulfillment	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Hierarchy	oaHierarchy	—	hierarchy	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>

XML	SOAP	REST	Database	Create	Read	Update	Delete
HierarchyNode	oaHierarchyNode	—	hierarchy_node	✓	✓	✓	✓
History	oaHistory	—	approval	—	✓	—	—
HistoryNotes	oaHistoryNotes	—	approval	—	✓	—	—
ImportExport	oaImportExport	—	import_export	✓	✓	✓	✓
Invoice	oaInvoice	—	invoice	✓	✓	✓	✓
InvoiceLayout	oaInvoiceLayout	—	invoice_layout	—	✓	—	—
Issue	oaIssue	—	issue	✓	✓	✓	✓
IssueCategory	oaIssueCategory	—	issue_category	✓	✓	✓	—
IssueSeverity	oaIssueSeverity	—	issue_severity	✓	✓	✓	—
IssueSource	oaIssueSource	—	issue_source	✓	✓	✓	—
IssueStage	oaIssueStage	—	issue_stage	✓	✓	✓	—
IssueStatus	oaIssueStatus	—	issue_status	—	✓	—	—
Item	oaItem	—	item	✓	✓	✓	✓
ItemToUserLocation	oaItemToUserLocation	—	item_to_user_location	✓	✓	✓	✓
Jobcode	oaJobcode	JobCode See the help topic Job Codes	job_code	✓	✓	✓	—
JobCodeUsed	oaJobCodeUsed	—	job_code_used	—	✓	—	—
Leave_accrual_rule	oaLeave_accrual_rule	—	leave_accrual_rule	✓	✓	✓	—
Leave_accrual_rule_to_user	oaLeave_accrual_rule_to_user	—	leave_accrual_rule_to_user	✓	✓	✓	—
Leave_accrual_transaction	oaLeave_accrual_transaction	—	leave_accrual_transaction	✓	✓	✓	—
LoadedCost	oaLoadedCost	—	loaded_cost	✓	✓	✓	—
Module	oaModule	—	—	—	✓	—	—
Newsfeed	oaNewsfeed	—	newsfeed	✓	✓	✓	✓
NewsfeedMessage	oaNewsfeedMessage	—	newsfeed_message	✓	✓	✓	✓
Payment	oaPayment	—	payment	✓	✓	✓	✓
Paymentterms	oaPaymenttems	—	payment_tems	✓	✓	✓	—
Paymenttype	oaPaymenttype	—	payment_type	✓	✓	✓	✓
Payrolltype	oaPayrolltype	—	payroll_type	✓	✓	✓	✓
PendingBooking	oaPendingBooking	—	pending_booking	—	✓	—	—
Preference	oaPreference	—	preference	✓	✓	✓	—
Product	oaProduct	—	product	✓	✓	✓	—
Project	oaProject	Project See the help topic Projects	project	✓	✓	✓	✓
Projectassign	oaProjectassign	—	project_assign	✓	✓	✓	✓

XML	SOAP	REST	Database	Create	Read	Update	Delete
ProjectAssignment Profile	oaProjectAssignment Profile	—	project_assignment_profile	✓	✓	✓	✓
Projectbillingrule	oaProjectbillingrule	—	project_billing_rule	✓	✓	✓	—
Projectbillingtransaction	oaProjectbillingtransaction	—	project_billing_transaction	✓*	✓	✓*	✓*
ProjectBudgetGroup	oaProjectBudgetGroup	—	project_budget_group	✓	✓	✓	✓
ProjectBudgetRule	oaProjectBudgetRule	—	project_budget_rule	✓	✓	✓	✓
ProjectBudget Transaction	oaProjectBudget Transaction	—	project_budget_transaction	✓	✓	✓	✓
Projectgroup	oaProjectgroup	—	project_group	✓	✓	✓	—
Projectlocation	oaProjectlocation	—	project_location	—	✓	—	—
ProjectPricing	oaProjectPricing	—	project_pricing	✓	✓	✓	✓
ProjectStage	oaProjectstage	—	project_stage	✓	✓	✓	✓
Projecttask	oaProjecttask	ProjectMilestone – See the help topic Project Milestones ProjectPhase – See the help topic Project Phases ProjectTask – See the help topic Project Tasks	project_task classification set to M for milestones, P for phases, or T for tasks	✓	✓	✓	✓
Projecttaskassign	oaProjecttaskassign	—	project_task_assign	✓	✓	✓	✓
ProjecttaskEstimate	oaProjecttask Estimate	—	project_task_estimate	✓	✓	✓	✓
Projecttask_type	oaProjecttask_type	—	project_task_type	✓	✓	✓	—
Proposal	oaProposal	—	proposal	—	✓	—	—
Proposalblock	oaProposalblock	—	proposal_block	—	✓	—	—
Proxy	oaProxy	—	proxy	✓	✓	✓	✓
Purchase_item	oaPurchase_item	—	purchase_item	✓*	✓	✓*	✓
Purchaseorder	oaPurchaseorder	—	purchaseorder	✓	✓	✓	✓
Purchaser	oaPurchaser	—	purchaser	—	✓	—	—
Purchaserequest	oaPurchaserequest	—	purchase_request	—	✓	—	—
Ratecard	oaRatecard	—	rate_card	✓	✓	✓	—
RateCardItem	oaRateCardItem	—	rate_card_item	✓	✓	✓	—
Reimbursement	oaReimbursement	—	reimbursement	✓	✓	✓	✓
Repeat	oaRepeat	—	repeat	✓	✓	✓	—
Report	oaReport	—	report	—	✓	—	—
Request_item	oaRequest_item	—	request_item	—	✓	—	✓
ResourceAttachment	oaResourceAttachment	—	resource_attachment	✓	✓	✓	✓
Resourceprofile	oaResourceprofile	—	resourceprofile	✓	✓	✓	✓
Resourceprofile_type	oaResourceprofile_type	—	resourceprofile_type	✓	✓	✓	✓

XML	SOAP	REST	Database	Create	Read	Update	Delete
ResourceRequest	oaResourceRequest	—	resource_request	✓	✓	✓	✓
ResourceRequest Queue	oaResourceRequest Queue	—	resource_request_queue	✓	✓	✓	✓
ResourceSearch	oaResourceSearch	—	resourcesearch	✓	✓	✓	✓
RevenueContainer	oaRevenueContainer	—	revenue_container	—	✓	✓*	—
RevenueProjection	oaRevenueProjection	—	revenue_projection	—	✓	—	—
Revenue_recognition_rule	oaRevenue_recognition_rule	—	revenue_recognition_rule	✓	✓	✓	—
Revenue_recognition_rule_amount	oaRevenue_recognition_rule_amount	—	revenue_recognition_rule_amount	✓	✓	✓	—
Revenue_recognition_transaction	oaRevenue_recognition_transaction	—	revenue_recognition_transaction	✓	✓	✓	—
RevenueStage	oaRevenueStage	—	revenue_stage	—	✓	—	—
Role	oaRole	—	role	—	✓	—	—
Schedulebyday	oaSchedulebyday	—	schedule_by_day	—	✓	—	—
Scheduleexception	oaScheduleexception	—	scheduleexception	✓	✓	✓	✓
Schedulerequest	oaSchedulerequest	—	schedule_request	✓	✓	✓	✓
Schedulerequest_item	oaSchedulerequest_item	—	schedule_request_item	—	✓	✓	✓
Slip	oaSlip	—	slip	✓	✓	✓	✓
SlipProjection	oaSlipProjection	—	slip_projection, slip	—	✓	—	—
Slipstage	oaSlipstage	—	slip_stage	✓	✓	✓	—
SummaryView	oaSummaryView	—	—	—	✓	—	—
TagGroup	oaTagGroup	—	tag_group	✓	✓	✓	✓
TagGroupAttribute	oaTagGroupAttribute	—	tag_group_attribute	✓	✓	✓	✓
TargetUtilization	oaTargetUtilization	—	target_utilization	✓	✓	✓	✓
Task	oaTask	— TimeEntry – See the help topic Time Entries	task	✓	✓	✓	✓
TaskAdjustment	oaTaskAdjustment	—	task_adjustment	—	✓	—	—
TaskTimecard	oaTaskTimecard	—	—	—	✓	—	—
TaxLocation	oaTaxLocation	—	tax_location	✓	✓	✓	—
TaxRate	oaTaxRate	—	tax_rate	✓	✓	✓	—
Term	oaTerm	—	term	—	✓	—	—
Ticket	oaTicket	Receipt See the help topic Receipts	ticket	✓	✓	✓	✓
Timecard	oaTimecard	—	time_card	—	✓	—	—
Timesheet	oaTimesheet	—	timesheet	✓	✓	✓	✓

XML	SOAP	REST	Database	Create	Read	Update	Delete
Timetype	oaTimetype	—	time_type	✓	✓	✓	✓
Todo	oaTodo	—	todo	—	✓	—	—
Uprate	oaUprate	—	up_rate	✓	✓	✓	✓
User	oaUser	—	user	✓	✓	✓	✓
UserLocation	oaUserLocation	—	user_location	✓	✓	✓	✓
UserWorkschedule	oaUserWorkschedule	—	workschedule	✓	✓	✓	✓
Vendor	oaVendor	—	vendor	✓	✓	✓	✓
Viewfilter	oaViewfilter	—	viewfilter	—	✓	—	—
Viewfilterrule	oaViewfilterrule	—	viewfilter_rule	—	✓	—	—
Workschedule Workhour	oaWorkschedule Workhour	—	workschedule_workhour	—	✓	—	—
Workspace	oaWorkspace	—	workspace	✓	✓	✓	✓
Workspacelink	oaWorkspacelink	—	workspace_link	✓	✓	✓	—
Workspaceuser	oaWorkspaceuser	—	workspace_user	✓	✓	✓	—

Note: The following object types are listed in the OpenAir WSDL but are not available for use by client applications: FormPermissionField, ServerStatus.

The following object types are described in other sections of this guide: [Approval](#), [CustomField](#), [Date](#), [Error](#), [oaFieldAttribute](#), [Flag](#), [oaSwitch](#).

AccountingPeriod

An accounting period [AccountingPeriod] defines a custom accounting date for transactions which can then be used instead of the transaction date for reporting purposes.

Review the [Usage Guidelines](#) for the AccountingPeriod object.

For more information about accounting periods, see the help topic [Accounting Periods](#).

—	XML	SOAP	REST	Database table
Object	AccountingPeriod	oaAccountingPeriod	—	accounting_period
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The AccountingPeriod object has the following standard properties:

XML / SOAP	Database	Description
active	active	A "1/0" field indicating whether this period is open or closed.
created	created	[Read-only] The date the accounting period record was created. See Date Fields

XML / SOAP	Database	Description
current_period	current_period	A "1/0" field indicating whether this is the current period.
default_period	default_period	A "1/0" field indicating whether this is the default period.
end_date	end_date	[Required] The ending date of the period. See Date Fields
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the accounting period.
notes	notes	Notes.
period_date	period_date	The custom date to use for this period. See Date Fields
period_date_how	period_date_how	What date should be used when marking transactions to this period: <ul style="list-style-type: none"> ■ S – Start date ■ E – End date ■ P – Period date
start_date	start_date	[Required] The starting date of the period. See Date Fields
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields

Usage Guidelines

Review the following guidelines:

- The date range defined by start_date and end_date must be valid. start_date must be before end_date.
- The date range must not be overlapping with that of another

Actualcost

An actual cost [Actualcost] defines a cost for a specific date range across projects based on payroll records.

Note: This object type is used for a complex feature that is not enabled for most company's OpenAir account.

—	XML	SOAP	REST	Database table
Object	Actualcost	oaActualcost	—	actual_cost
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Actualcost object has the following standard properties:

XML / SOAP	Database	Description
cost	cost	The cost.
cost_typeid	cost_type_id	The ID of the cost_type.
created	created	[Read-only] Time the record was created. See Date Fields
currency	currency	Currency of the cost field.
date	date	[Required] Date for the actual cost. See Date Fields
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
is_accrual	is_accrual	A 1/0 field indicating whether this actual cost is an accrual.
name	name	The name of the actual cost. This field is never populated. It is used only to satisfy subtotalling by actual cost.
notes	notes	Notes.
period	period	[Required] The time period of the actual cost: Daily, Weekly, Monthly, Quarterly, Annually.
updated	updated	[Read-only] Time the record was last modified. See Date Fields
userid	user_id	[Required] The ID of the user.

Address

Use the Address object to add or update address information.

OpenAir XML API also uses the Address object as a substructure to pass address and other contact information when reading, adding or updating any type of records including such information. See [Address Fields](#).

—	XML	SOAP	REST	Database table
Object	Address	oaAddress	—	address
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Address object has the following standard properties:

XML	SOAP	Database	Description
addr1	addr1	address1	Address line 1

XML	SOAP	Database	Description
addr2	addr2	address2	Address line 2
addr3	addr3	address3	Address line 3
addr4	addr4	address4	Address line 4
city	city	city	City
contact_id	contact_id	contact_id	The internal ID of the associated contact
country	country	country	Country
customer_only	—	—	Used for backward compatibility when modifying a customer record. Set to yes to update only the customer's billing address [billingaddr] of the customer, and not its associated contact. See Customer .
email	email	email	Email address
fax	fax	—	Fax number
first	first	—	First name
id	id	id	Unique ID. Automatically assigned by OpenAir.
last	last	—	Last name
middle	middle	—	Middle name
mobile	mobile	—	Mobile phone number
phone	phone	—	Phone number
salutation	salutation	—	Contact's salutation
state	state	state	State
zip	zip	zip	Zip code

Agreement

An agreement [Agreement] is a contract or a statement of work (SOW) that is used to track budget balance for a specific customer.

Review the [Usage Guidelines](#) for the Agreement object.

For more information about agreements, see the help topic [Agreements](#).

—	XML	SOAP	REST	Database table
Object	Agreement	oaAgreement	—	agreement
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Agreement object has the following standard properties:

Note: Agreement object properties may also include custom fields. The object type supports the `custom_equal_to_read` method and the `enable_custom_read` attribute.

XML / SOAP	Database	Description
acct_date	acct_date	The accounting period date of the agreement. See Date Fields
active	active	A 1/0 field indicating whether this is an active agreement. Defaults to 1 if not set when adding an agreement.
code	acct_code	Optional accounting system code for integration with external accounting systems.
created	created	[Read-only] Time the record was created. See Date Fields
currency	currency	Currency for the money fields in the record.
customerid	customerid	[Required] Customer ID.
date	date	[Required] The date of the agreement. See Date Fields
externalid	externalid	External ID.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the agreement.
notes	notes	Notes.
number	number	The agreement number.
picklist_label	—	Label as shown on form picklist.
total	total	The agreement total. Dated by the date field.
updated	updated	[Read-only] Time the record was last modified. See Date Fields

Usage Guidelines

You cannot delete an Agreement object if this object is referenced by an object of any of the following type. Delete any dependent objects first before you delete an Agreement object.

- [Agreement_to_project](#)
- [Projectbillingrule](#)
- [Projectbillingtransaction](#)
- [Revenue_recognition_rule](#)
- [Revenue_recognition_rule_amount](#)
- [Revenue_recognition_transaction](#)
- [Slip](#)

- SlipProjection

Agreement_to_project

Use the agreement-project link [Agreement_to_project]object to read or create many-to-many links between projects and agreements.

Review the [Usage Guidelines](#) for the Agreement_to_project object.

—	XML	SOAP	REST	Database table
Object	Agreement_to_project	oaAgreement_to_project	—	agreement_to_project
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Agreement_to_project object has the following standard properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this is an active agreement-project link. Defaults to 1 if not set when adding an object agreement-project link.
agreementid	agreement_id	[Required] The ID of the associated agreement.
created	created	[Read-only] Time the record was created. See Date Fields
customerid	customer_id	The ID of the associated customer. Does not need to be set as it can be derived inline from project_id.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
projectid	project_id	[Required] The ID of the associated project.
updated	updated	[Read-only] Time the record was last modified. See Date Fields

Usage Guidelines

Each agreement-project link must be unique. There cannot be two Agreement_to_project objects with the same agreementid and projectid combination. An error is returned if the operation would result in a duplicate agreementid and projectid combination.

ApprovalLine

An approval line [ApprovalLine]is an action in the chain of approval for transactions that require approval, such as timesheets or expense reports for example.

For more information about approval routing, see the help topic [Approval Routing](#).

—	XML	SOAP	REST	Database table
Object	ApprovalLine	oaApprovalLine	—	approval

—	XML	SOAP	REST	Database table
Supported Commands	Read	read()	—	—

The ApprovalLine object has the following standard properties:

XML / SOAP	Database	Description
action	action	[Read-only] The approval action. <ul style="list-style-type: none"> ■ S - Initial submission for approval ■ P - Pending approval request ■ A - Acceptance of approval request ■ R - Rejection of approval request ■ U - Unapproval action
approvalid	approval_id	[Read-only] ID of the associated approval. Represents a meta-approval, or an 'approval confirmation'.
approvalprocess_ruleid	approvalprocess_rule_id	[Read-only] ID of the approval process rule if this is associated with an approval process.
approvalprocessid	approvalprocess_id	[Read-only] ID of the approval process if this is associated with an approval process.
audit	audit	[Read-only] Audit trail of changes
authorizationid	authorization_id	[Read-only] ID of the associated authorization
booking_requestid	booking_request_id	[Read-only] ID of the associated booking request
bookingid	bookingid	[Read-only] ID of the associated booking
created	created	[Read-only] Time the record was created. See Date Fields
customerid	customer_id	[Read-only] ID of the associated customer
date	date	[Read-only] Date and time of the action. See Date Fields
deal_booking_requestid	deal_booking_request_id	[Read-only] ID of the associated deal booking request
delay_action	delay_action	[Read-only] Delayed action
delay_to	delay_to	[Read-only] Delay action until this time
envelopeid	envelope_id	[Read-only] ID of the associated envelope (expense report)
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
invoiceid	invoice_id	[Read-only] ID of the associated invoice
notes	notes	[Read-only] Notes, reasons, etc.

XML / SOAP	Database	Description
pending_done	pending_done	[Read-only] If the action is 'P'ending, this flag is set to 1 once an 'A' or 'R' action record is created.
project_budget_groupid	project_budget_group_id	[Read-only] ID of the associated project budget group
project_total	project_total	[Read-only] If this is a project-based approval this holds the total amount (money or hours) that was approved.
projectid	project_id	[Read-only] ID of the associated project if this is a project approval
proposalid	proposal_id	[Read-only] ID of the associated proposal
purchaseorderid	purchaseorder_id	[Read-only] ID of the associated purchase order
purchaserequestid	purchaserequest_id	[Read-only] ID of the associated purchase request
revenue_containerid	revenue_container_id	[Read-only] ID of the associated revenue container
schedule_requestid	schedule_request_id	[Read-only] ID of the associated schedule request
seq_number	seq_number	[Read-only] If this is associated with an approval process, this is the sequence number associated with it.
status	status	[Read-only] The status of the child meta-approval. Only assigned a value if the record has a meta-approval. <ul style="list-style-type: none"> ■ S - Submitted ■ A - Approved ■ R - Rejected
submitter	submitter	[Read-only] ID of the user submitting the approval. Only valid for a submittal record (action = 'S').
timesheetid	timesheet_id	[Read-only] ID of the associated timesheet.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields
userid	user_id	[Read-only] ID of the user. A submittal record has the ID of the user whose approvals are to be followed, this is usually the user who submitted the request, but for booking requests, it may be either the submitter or the user for whom the booking request is for depending on setting. All other records have the ID of the approver.

ApprovalProcess

An approval process [ApprovalProcess] is a multilevel chain of approval required before a transaction can be processed. It includes the header information for a group of approval process rules but does not include information about the approval process rules themselves. For more information about approval processes, see the help topic [Approval Processes](#).

—	XML	SOAP	REST	Database table
Object	ApprovalProcess	oaApprovalProcess	—	approvalprocess
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The ApprovalProcess object has the following standard properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields
externalid	external_id	The unique external record ID if the record was imported from an external system
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name used for display in popups and lists.
updated	updated	[Read-only] Time the record was last modified. See Date Fields

Attachment

An Attachment is either:

- A file uploaded and stored in OpenAir and associated with a record. Each attachment includes file metadata as well as the file itself.

 **Note:** Workspace documents are attachments associated with a workspace record.

- A folder that can contain attachments.

Review the [Usage Guidelines](#) for the Attachment object.

—	XML	SOAP	REST	Database table
Object	Attachment	oaAttachment	Attachment	attachment
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	To work with attachments, use the endpoints and methods specific to the object the attachments are associated with. See the help topic OpenAir REST API Endpoint Reference .	—

The Attachment object has the following standard properties:

Note: Attachment object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom` read attribute.

XML / SOAP	REST	Database	Description
—	attachmentCategoryId	attachment_category_id	The internal ID of the attachment category.
attachmentid	—	attachmentid	If non-zero, the attachment record associated with this attachment.
base64_data	—	base64_data	Base 64 encoded binary data of the attachment file.
created	created	created	[Read-only] The date and time the attachment was created. See Date Fields
file_name	fileName	file_name	The filename of the attached file uploaded by the user.
—	fileType	—	[Read-only] The file type. This is used for tracking purposes.
hash_name	—	hash_name	[Read-only] The relative path and filename of the attached file stored on OpenAir servers.
—	hasThumbnail	—	[Read-only] A 1/0 field indicating if a thumbnail image is available for the attachment. Returned only if the Attachment Thumbnail and Attachment Viewer feature is enabled for the OpenAir account.
id	id	id	[Read-only] The unique internal identifier of the attachment. Assigned by OpenAir
—	isExcludedFromAlert	exclude_alert	A 1/0 field indicating if the attachment is excluded from the alert system.
is_a_folder	isFolder	is_a_folder	A 1/0 field indicating if other attachment records have this attachment record as a parent.
—	isLocked	is_locked	A 1/0 field indicating if the attachment is locked for editing.
—	isViewable	—	A 1/0 field indicating if the attachment is viewable using the Attachment Viewer feature. Returned only if the Attachment Thumbnail and Attachment Viewer feature is enabled for the OpenAir account.
locked_by	lockedBy	locked_by	[Read-only] The internal ID of the employee who uploaded the file. The value is 0 (zero) if unlocked.

XML / SOAP	REST	Database	Description
name	name	name	The display name of the attachment.
notes	notes	notes	Notes about the attachment. This attribute is used for keyword search.
owner_type	—	association	The type of record this attachment is attached to. For example: user, envelope, ticket, timesheet, agreement, or customerpo.
ownerid	—	record	The internal ID of the record linking to this attachment.
parentid	—	parentid	The internal ID of the immediate ancestor attachment. If 0 or Null, the attachment is a top-level document or folder.
size	size	size	[Read-only] The file size in bytes.
updated	updated	updated	[Read-only] Time the record was last modified. See Date Fields
—	uploadedBy	uploaded_by	[Read-only] The internal ID of the employee who uploaded the file.
workspaceid	workspaceId	workspaceid	The internal ID of the associated workspace.

Usage Guidelines

Review the following guidelines:

- Some object properties are required depending on the `is_a_folder` property value:
 - If `is_a_folder` is set to 1 then following object properties are required:
 - name
 - workspaceid
 - Otherwise, the following object properties are required:
 - base64_data
 - file_name
 - owner_type
 - ownerid
- It is not possible to add, modify, or delete an attachment file if the object that this file is attached to (object referenced by `owner_type` and `ownerid`) cannot be edited. For expense or timesheet attachments, for example, the associated Envelope or Timesheet must have a status equal to 0 (Open).
- When adding an attachment file, the size of the attachment file (calculated from `base64_data`) must be smaller than the storage space available in your company's OpenAir account.
- When modifying an Attachment, only the following properties can be updated:
 - attachmentid
 - name

- notes
- parentid
- If the Attachment Thumbnail and Attachment Viewer feature is enabled for your OpenAir account, a thumbnail is generated automatically when you add an attachment of a supported format. The `file_name` property value must include a supported file extension. For more information about the Attachment Thumbnail feature, including supported file format and filename extensions, see  [OpenAir Optional Features Book](#).

Attribute

An attribute [Attribute] is a measurement level for a skill or competency. See also [AttributeDescription](#) and [Attributeset](#).

For more information about resource profiles and attributes, see the help topics [Resource Profile](#) and [Attribute Sets](#).

—	XML	SOAP	REST	Database table
Object	Attribute	oaAttribute	—	attribute
Supported Commands	Read	read()	—	—

The Attribute object has the following properties:

XML / SOAP	Database	Description
attribute_setid	attribute_set_id	[Read-only] ID of the associated attribute set.
created	created	[Read-only] Time the record was created. See Date Fields
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] Name of the attribute.
notes	notes	[Read-only] Notes.
updated	updated	[Read-only] Time the record was last modified. See Date Fields

AttributeDescription

An attribute description [AttributeDescription] is a definition of a measurement level for a skill or competency in resource profiles. For example, attribute descriptions can be used to provide detailed characteristics for each language skill level (beginner, intermediate, advanced) or technical competencies. See also [Attribute](#) and [Attributeset](#).

Review the [Usage Guidelines](#) for the AttributeDescription object.

For more information about resource profiles and attributes, see the help topics [Resource Profile](#) and [Attribute Sets](#).

—	XML	SOAP	REST	Database table
Object	AttributeDescription	oaAttributeDescription	—	attribute_description

—	XML	SOAP	REST	Database table
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The AttributeDescription object has the following properties:

XML / SOAP	XML / SOAP	Description
attributeid	attribute_id	[Required] ID of the attribute.
created	created	[Read-only] The time the record was created. See Date Fields
deleted	deleted	[Read-only] A "1/0" field indicating if the record was deleted.
description	description	[Required] Information about the attribute in context of specific resourceprofile_type.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
resourceprofile_typeid	resourceprofile_type_id	[Required] ID of the resourceprofile_type.
updated	updated	[Read-only] The time the record was last modified. See Date Fields

Usage Guidelines

There cannot be two AttributeDescription objects with the same attributeid and resourceprofile_typeid combination. An error is returned if the operation would result in a duplicate attributeid and resourceprofile_typeid combination.

Attributeset

An attribute set [Attributeset] is a measurement scale for skills and competencies. See also [Attribute](#) and [AttributeDescription](#).

For more information about resource profiles and attributes, see the help topics [Resource Profile](#) and [Attribute Sets](#).

—	XML	SOAP	REST	Database table
Object	Attributeset	oaAttributeset	—	attribute_set
Supported Commands	Read	read()	—	—

The Attributeset object has the following properties:

XML / SOAP	XML / SOAP	Description
created	created	[Read-only] Time the record was created. See Date Fields
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.

XML / SOAP	XML / SOAP	Description
name	name	[Read-only] Name of the attribute set.
notes	notes	[Read-only] Attribute set notes
updated	updated	[Read-only] Time the record was updated. See Date Fields

BillingSplit

When there is not a one-to-one relationship between a billed charge (slip) and record of another type, billing split [BillingSplit] objects are used to record to many-to-one or one-to-many lookup. The one-to-one relationship is still modeled with the embedded `slip_id` field.

—	XML	SOAP	REST	Database table
Object	BillingSplit	oaBillingSplit	—	billing_split
Supported Commands	Read	read()	—	—

The BillingSplit object has the following properties:

XML / SOAP	XML / SOAP	Description
created	created	[Read-only] Time the record was created. See Date Fields
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
project_billing_transactionid	project_billing_transaction_id	[Read-only] The ID of the associated project billing transaction.
slipid	slip_id	[Read-only] The ID of the slip that was created.
taskid	task_id	[Read-only] The ID of the associated task.
updated	updated	[Read-only] Time the record was updated. See Date Fields

Booking

A booking [Booking] is the allocation of an employee's (or resource's) time to a work package (project or project task).

Review the [Usage Guidelines](#) for the Booking object.

—	XML	SOAP	REST	Database table
Object	Booking	oaBooking	—	booking

—	XML	SOAP	REST	Database table
Supported Commands	Add, Read, Modify, ModifyOnCondition, Delete, Submit, Approve, Reject, Unapprove	add(), read(), modify(), upsert(), delete(), submit(), approve(), reject(), unapprove()	—	—

The Booking object has the following standard properties:

Note: Booking object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
approval_status	approval_status	A one-character string indicating the approval status of the booking request. Possible values: <ul style="list-style-type: none"> ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected
as_percentage	as_percentage	A 1/0 field indicating which of the fields (hours or percentage) are actual, and which is derived. <ul style="list-style-type: none"> ■ 1 = percentage is actual and hours is derived. ■ 0 = hours in actual and percentage is derived.
booking_typeid	booking_type_id	The ID of the associated booking_type.
created	created	[Read-only] Time the record was created. See Date Fields
customerid	customer_id	[Required] The ID of the associated customer.
date_approved	date_approved	The date the booking request was approved. See Date Fields
date_submitted	date_submitted	The date the booking_request was submitted. See Date Fields
enddate	enddate	[Required] The end date of the booking. See Date Fields
endtime	endtime	End time. See Date Fields

XML / SOAP	Database	Description
externalid	external_id	If the record was imported from an external system you store the unique external record ID here.
hours	hours	The number of hours booked to this project during this date range. This is either the actual booked hours or derived from the percentage.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
job_codeid	job_code_id	The ID of the associated job code.
locationid	location_id	The location ID for this booking.
notes	notes	Booking notes.
notify_owner	notify_owner	A 1/0 field indicating whether to send email to the requestor when the booking is modified.
ownerid	owner_id	The ID of the associated user creating the booking. Can be specified when adding an object.
percentage	percentage	The percentage of time booked to this project during this date range. This is either the actual booked percentage or derived from the hours.
project_assignment_profileid	project_assignment_profile_id	The ID of the associated project assignment profile.
project_taskid	project_task_id	The ID of the task within the associated project.
projectid	project_id	[Required] The ID of the associated project.
repeatid	repeat_id	The ID of the associated repeating event.
resource_request_queue_id	resource_request_queue_id	The ID of the associated resource request queue.
source_booking_id	source_booking_id	ID of the booking used to create this booking.
startdate	startdate	[Required] The start date of the booking. See Date Fields
starttime	starttime	Start time. See Date Fields
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields

XML / SOAP	Database	Description
userid	user_id	[Required] The ID of the associated user.

Usage Guidelines

Review the following guidelines:

- Some object properties are required depending on the `as_percentage` property value:
 - `percentage` is required if `as_percentage` is set to 1.
 - `hours` is required otherwise.
- The date range defined by `start_date` and `end_date` must be valid. `start_date` must be before `end_date`.
- `approval_status` is set automatically, if empty or not already set when adding or modifying a Booking object. It is set to 0 if booking approvals are enabled for your company's OpenAir account, or to A otherwise.
- A `booking_typeid` is required if the **Require booking type when booking resources** [`rm_booking_require_type`] box is checked in Administration > Application Settings > Resources > Other Settings for your company's account.
- Depending on your company's account configuration, `booking_typeid` may be set automatically, if empty or not already set when adding or modifying a Booking object. It is set to the default booking type associated with the `approval_status`, if one is set. See the help topic [Booking Types](#).
- A booking cannot be deleted if it is a source booking [`source_booking_id`] for another booking. Delete any other bookings referencing the booking as source booking before you can delete a Booking object.
- Deleting a booking deletes the booking and all its attachments, all references to this booking in project task assignments, all related approvals, and all related alerts.

BookingByDay

Booking by day [BookingByDay] objects give a day-by-day representation of bookings.

—	XML	SOAP	REST	Database table
Object	BookingByDay	oaBookingByDay	—	booking_by_day
Supported Commands	Read	read()	—	—

The BookingByDay object has the following properties:

XML / SOAP	XML / SOAP	Description
booking_id	booking_id	[Read-only] The ID of the associated booking.
booking_type_id	booking_type_id	[Read-only] The ID of the associated booking type.
created	created	[Read-only] Time the record was created. See Date Fields
customer_id	customer_id	[Read-only] The ID of the associated customer.

XML / SOAP	XML / SOAP	Description
date	date	[Read-only] The date of the booking. See Date Fields
hours	hours	[Read-only] The number of booked hours on this date for this customer/project/user/booking type. High precision to reduce effect of rounding.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
job_code_id	job_code_id	[Read-only] The ID of the associated job code.
project_id	project_id	[Read-only] The ID of the associated project.
project_task_id	project_task_id	[Read-only] The ID of the task within the associated project.
updated	updated	[Read-only] Time the record was last modified. See Date Fields
userid	user_id	[Read-only] The ID of the associated user.

BookingType

Booking types [BookingType] are used for classification of bookings. Examples of booking types may include billable, non-billable, or business development.

—	XML	SOAP	REST	Database table
Object	BookingType	oaBookingType	—	booking_type
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The BookingType object has the following standard properties:

Note: BookingType object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	XML / SOAP	Description
active	active	A 1/0 field specifying if the type is active.
approval_status	approval_status	A one-character string indicating the approval status of the booking request. Possible values: <ul style="list-style-type: none"> ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected
created	created	[Read-only] Time the record was created. See Date Fields

XML / SOAP	XML / SOAP	Description
default_for_approval_status	default_for_approval_status	A "1/0" field indicating whether this is the default booking type used when the approval status changes.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The name of the booking type.
notes	notes	Booking notes.
picklist_label	—	Label as shown on form picklist.
priority	priority	The priority of the booking type (1 - 9).
updated	updated	[Read-only] Time the record was last modified. See Date Fields

Booking_request

A booking request [Booking_request] is a request for allocation of an employee's (or resource's) time to a project or project task.

—	XML	SOAP	REST	Database table
Object	Booking_request	oaBooking_request	—	booking_request
Supported Commands	Read	read()	—	—

The Booking_request object has the following properties:

XML / SOAP	Database	Description
approval_status	approval_status	[Read-only] A one-character string indicating the approval status of the booking request. Possible values: <ul style="list-style-type: none"> ■ O - Open ■ P - Pending approval ■ A - Approved ■ R - Rejected
as_percentage	as_percentage	[Read-only] A 1/0 field indicating which of the fields...hours or percentage are actual, and which is therefore derived. Only one value can be actual. If 1 then percentage is the actual, hours is derived. If 0 then percentage is derived, hours is actual.
attachment_id	attachment_id	[Read-only] If non-zero, the attachment record associated with this booking_request.
booking_type_id	booking_type_id	[Read-only] The ID of the associated booking_type.
created	created	[Read-only] Time the record was created. See Date Fields
customer_id	customer_id	[Read-only] The ID of the associated customer.

XML / SOAP	Database	Description
date_approved	date_approved	[Read-only] The date the booking_request was approved. See Date Fields
date_submitted	date_submitted	[Read-only] The date the booking_request was submitted. See Date Fields
description	description	[Read-only] The description or purpose for the booking_request.
enddate	enddate	[Read-only] The end date of the booking_request. See Date Fields
external_id	external_id	[Read-only] If the record was imported from an external system you store the unique external record ID here.
hours	hours	[Read-only] The number of hours booked to this project during this date range. This is either the actual booked hours or derived from the percentage.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
job_code_id	job_code_id	[Read-only] The ID of the associated job code.
name	name	[Read-only] The name of the booking_request (Prefix + number).
notes	notes	[Read-only] Booking notes
notify_owner	notify_owner	[Read-only] A 1/0 field indicating whether to send email to the booking request owner changes occur to the resulting bookings.
number	number	[Read-only] The booking_request number that increments by 1.
owner_id	owner_id	[Read-only] The ID of the associated user creating the booking request.
percentage	percentage	[Read-only] The percentage of time booked to this project during this date range. This is either the actual booked percentage or derived from the hours.
prefix	prefix	[Read-only] A static alphanumeric booking_request number prefix.
project_id	project_id	[Read-only] The ID of the associated project.
project_task_id	project_task_id	[Read-only] The ID of the task within the associated project.
repeat_id	repeat_id	[Read-only] The ID of the associated repeating eventnote exception to associated object ID rule
startdate	startdate	[Read-only] The start date of the booking_request. See Date Fields
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields
user_id	user_id	[Read-only] The ID of the associated user.

Budget

A transactional budget [Budget] lets you track project funding over time. For more information, see the help topic [Transactional Budget](#)

Review the [Usage Guidelines](#) for the Budget object.

—	XML	SOAP	REST	Database table
Object	Budget	oaBudget	—	budget
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Budget object has the following properties:

XML / SOAP	Database	Description
budgetcategory_id	budget_category_id	The ID of the budget category.
categoryid	category_id	The ID of the associated category.
created	created	[Read-only] Time the record was created. See Date Fields
currency	currency	Currency for the money fields in the record.
customerid	customer_id	The ID of the associated customer.
date	date	The date of the budget entry. See Date Fields
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The name.
notes	notes	Budget notes.
projectid	project_id	The ID of the associated project.
total	total	The total value of budget entry. Dated by the date field.
updated	updated	[Read-only] Time the record was last modified. See Date Fields

Usage Guidelines

Depending on your company's account configuration, adding or modifying a budget transaction triggers the budget calculation for the project.

BudgetAllocation

A budget allocation [BudgetAllocation] designates an employee responsible for an activity performed to obtain funding for a project.

—	XML	SOAP	REST	Database table
Object	BudgetAllocation	oaBudgetAllocation	—	budget_allocation
Supported Commands	Add, Read, Modify	add(), read(), modify(), upsert()	—	—

The BudgetAllocation object has the following properties:

XML / SOAP	Database	Description
allocation	allocation	The percentage of the budget entry that this user was allocated to.
budgetactivity_id	budget_activity_id	The ID of the budget activity.
budgetcategory_id	budget_category_id	The ID of the budget category.
budgetid	budget_id	The ID of the associated budget.
created	created	[Read-only] Time the record was created. See Date Fields
currency	currency	Currency for the money fields in the record.
customerid	customer_id	The ID of the associated customer.
date	date	The date of the budget entry. See Date Fields
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
projectid	project_id	The ID of the associated project.
total	total	The total value of budget entry. Dated by the date field.
updated	updated	[Read-only] Time the record was last modified. See Date Fields
userid	userid	The ID of the associated user.

Category

A service (category, activity, or time type) [Category] is a specific economic activity offered to your customers and used on invoices to customers. A service can be set up to bill at an hourly rate, at a set amount per day, week, or month, or at a flat rate. Services are available in the Timesheets, Projects (tasks, billing rules, and recognition rules), Invoices, and Opportunity applications.

Review the [Usage Guidelines](#) for the Category object

—	XML	SOAP	REST	Database table
Object	Category	oaCategory	—	category
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The Category object has the following standard properties:

Note: Category object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this is designated as an active category. Set to 1 if omitted when adding an object.
code	acct_code	Optional accounting system code for integration with external accounting systems.
cost_centerid	cost_center_id	The ID of the associated cost center.
cost_rate	cost_rate	The hourly cost rate
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
externalid	external_id	If the record was imported from an external system you store the unique external record ID here.
fixed_fee	fixed_fee	The fixed fee value of this service.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The category name.
notes	notes	Category notes.
other_rate	other_rate	The rate for another time billing metric.
other_rate_type	other_rate_type	The time the <code>other_rate</code> field applies to. Valid entries are Day, Week, Month, Quarter, Year and Session.
picklist_label	—	Label as shown on form picklist.
rate	rate	The hourly billing rate.
taxable	taxable	A 1/0 field indicating whether this item is taxable, vat-taxable, etc.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Usage Guidelines

You cannot delete a Category object if this object is referenced by an object of any of the following type. Delete any dependent objects first before you delete a Category object.

- [Proposalblock](#)
- [Slip](#)
- [Task](#)

Category_<N>

Service line <N> [Category_<N>] can be used to categorize transactions in the same way as [Category](#). Transactions in OpenAir can be split across service lines for reporting purposes.

—	XML	SOAP	REST	Database table
Object	Category_<N>	oaCategory_<N>	—	category_<N> (category_1)
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Category_<N> object has the following standard properties:

 **Note:** Category_<N> object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this is designated as an active category. Set to 1 if omitted when adding an object.
code	acct_code	Optional accounting system code for integration with external accounting systems.
created	created	[Read-only] Time the record was created. See Date Fields .
externalid	external_id	If the record was imported from an external system you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The category name.
notes	notes	Notes
picklist_label	—	Label as shown on form picklist.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Ccrate

A service rate per customer [Ccrate] is the hourly rate used to bill for a particular service to a particular customer. Used when your OpenAir account is configured to get the billing rates from Service / Customer [`account.rate_from = cc`].

—	XML	SOAP	REST	Database table
Object	Ccrate	oaCcrate	—	cc_rate
Supported Commands	Read	read()	—	—

The Ccrate object has the following properties:

XML / SOAP	Database	Description
categoryid	category_id	[Read-only] The ID of the category this rate is associated with.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	[Read-only] The currency these rates are quoted in.
customerid	customer_id	[Read-only] The ID of the customer this rate is associated with.

XML / SOAP	Database	Description
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
notes	notes	[Read-only] Notes about the table.
rate	rate	[Read-only] The hourly billing rate.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Company

The company [Company] object holds your company information and generic account information including account-wide OpenAir settings.

—	XML	SOAP	REST	Database table
Object	Company	oaCompany	—	account
Supported Commands	Read , Modify	read() , modify()	—	—

The Company object has the following properties:

XML	SOAP	Database	Description
addr	—	—	An Address object with the company's address details. See Address Fields .
—	addr_addr1	address1	First line of the address.
—	addr_addr2	address2	Second line of the address.
—	addr_addr3	address3	Third line of the address.
—	addr_addr4	address4	Fourth line of the address.
—	addr_city	city	City name.
—	addr_contact_id	—	The associated contact ID.
—	addr_country	country	Country name.
—	addr_email	—	Email address.
—	addr_fax	fax	Fax number.
—	addr_first	—	First name.
—	addr_id	—	The ID of the associated address.
—	addr_last	—	Last name.
—	addr_middle	—	Middle name.
—	addr_mobile	—	Mobile phone number.
—	addr_phone	phone	Phone number.
—	addr_salutation	—	Salutation.
—	addr_state	state	State name.

XML	SOAP	Database	Description
—	addr_zip	zip	Zip code of the address.
base_currency	base_currency	base_currency	Three-letter code for the base currency.
businesstype	businesstype	business_type	General business category.
company	company	company	The company name, as it should be printed on invoices and other documents.
created	created	created	[Read-only] The date the company record was created. Must be a Date object for XML API and SOAP API. See Date Fields
currencies	currencies	currencies	Comma-separated list of three-letter codes for the currencies used for business transactions in a multicurrency account.
flags	flags	—	Company-specific settings. See Company and User Settings .
hide_rate	hide_rate	—	A 1/0 field indicating whether to hide hourly rate from normal user types in the company.
id	id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
is_multicurrency	is_multicurrency	is_multicurrency	A 1/0 field indicating whether the company's account is a multicurrency account.
nickname	nickname	nickname	The company nickname.
rate_from	rate_from	rate_from	Where is the hourly billing rate pulled from: <ul style="list-style-type: none"> ■ ca - Category billing rate ■ cc - Category billing rate with a customer override ■ us - User billing rate ■ cp - Customer/project billing rate ■ up - User billing rate over-ridden on a project basis ■ pr - Project based billing rules
updated	updated	updated	[Read-only] The date the company record was last updated or modified. Must be a Date object for XML API and SOAP API. See Date Fields
VAT_registration_number	VAT_registration_number	vat_registration_number	VAT registration number.
workscheduleid	workscheduleid	workschedule_id	[Read-only] The ID of the associated primary company workschedule.

Contact

A contact [Contact] is a person working for, or associated with a customer. For more information about contacts, see the help topic [Contacts](#).

—	XML	SOAP	REST	Database table
Object	Contact	oaContact	Contact	contact
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	See the help topic Contacts	—

The Contact object has the following standard properties:

 **Note:** Contact object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML	SOAP	REST	Database	Description
active	active	isActive	active	A 1/0 field indicating if the contact is active. Defaults to 1 if not set when adding a contact.
addr	—	—	—	[Required] An Address object with the contact's address details. The Address object must include at least a last name [last]. See Address Fields
—	addr_addr1	address1	address .address1	First line of the contact's address.
—	addr_addr2	address2	address .address2	Second line of the contact's address.
—	addr_addr3	address3	address .address3	Third line of the contact's address.
—	addr_addr4	address4	address .address4	Fourth line of the contact's address.
—	addr_city	city	address .city	The contact's city.
—	addr_contact_id	—	—	The associated contact ID.
—	addr_country	country	address .country	The contact's country.
—	addr_email	email	addr_email	The contact's Email address.
—	addr_fax	fax	addr_fax	The contact's fax number.
—	addr_first	firstName	addr_first	The contact's first name.
—	addr_id	—	address .id	The ID of the associated address.
—	addr_last	lastName	addr_last	[Required] The contact's last name.
—	addr_middle	—	—	Middle name.
—	addr_mobile	mobile	addr_mobile	The contact's mobile phone number.
—	addr_phone	phone	addr_phone	The contact's phone number.

XML	SOAP	REST	Database	Description
—	addr_salutation	title	addr_salutation	The contact's title.
—	addr_state	state	address.state	The contact's State or Region.
—	addr_zip	zip	address.zip	The contact's ZIP code or postal code.
can_bill_to	can_bill_to	canBill	can_bill_to	A 1/0 field indicating if the contact can be used for billing.
can_ship_to	can_ship_to	canShip	can_ship_to	A 1/0 field indicating if the contact can be used for shipping.
can_sold_to	can_sold_to	canSell	can_sold_to	A 1/0 field indicating if the contact can be used for selling.
code	code	accountingCode	acct_code	Optional accounting code that can be used for integration with external accounting systems.
created	created	created	created	[Read-only] The date the contact record was created. See Date Fields
customer_company	customer_company	—	customer_company	Import-only field to specify customer by company name.
customerid	customerid	customerId	customer_id	[Required] The internal ID of the associated customer.
exported	exported	—	exported	[Read-only] Date and time the record was marked as exported. See Date Fields
externalid	externalid	externalId	external_id	The unique external ID of the contact, if the record was imported from an external system. Uniqueness may be enforced depending on your company's OpenAir account configuration.
id	id	id	id	[Read-only] The unique internal identifier of the contact. Assigned by OpenAir.
job_title	job_title	jobTitle	job_title	The contact's job title.
name	name	nickname	name	The nickname used to identify the contact. Automatically generated if not supplied.
notes	notes	notes	notes	Notes about the contact.
picklist_label	picklist_label	—	—	Label as shown on form picklist.
updated	updated	updated	updated	[Read-only] The date the contact record was last updated or modified. See Date Fields

Costcategory

Cost categories [Costcategory] can be used for classification of cost types. See also [Costtype](#).

—	XML	SOAP	REST	Database table
Object	Costcategory	oaCostcategory	—	cost_category
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Costcategory object has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating if this cost category is active.
created	created	[Read-only] Time the record was created. See Date Fields
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the actual cost. This field is never populated. It is used only to satisfy subtotalling by actual cost.
notes	notes	Notes.
updated	updated	[Read-only] Time the record was last modified. See Date Fields

Costcenter

Cost centers [Costcenter] can be used for classification of costs from expenses and employee's time for reporting and accounting purposes.

—	XML	SOAP	REST	Database table
Object	Costcenter	oaCostcenter	—	cost_center
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Costcenter object has the following properties:

Note: Costcenter object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this is active.

XML / SOAP	Database	Description
code	acct_code	Optional accounting system code for integration with external accounting systems.
created	created	[Read-only] Time the record was created. See Date Fields
externalid	external_id	If the record was imported from an external system you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the cost center.
notes	notes	Cost center notes.
picklist_label	—	Label as shown on form picklist.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields

Costtype

Cost types [Costtype] can be used for classification of actual costs. See also [Actualcost](#).

—	XML	SOAP	REST	Database table
Object	Costtype	oaCosttype	—	cost_type
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Costtype object has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating if this cost category is active.
cost_categoryid	cost_category_id	The ID of the associated cost category.
created	created	[Read-only] Time the record was created. See Date Fields
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the actual cost. This field is never populated. It is used only to satisfy subtotalling by actual cost.
notes	notes	Notes.
updated	updated	[Read-only] Time the record was last modified. See Date Fields

Currency

A currency exchange rate override [Currency] is a custom exchange rate against the base currency, which overrides the market exchange rate for your company's account.

Review the [Usage Guidelines](#) for the Currency object.

—	XML	SOAP	REST	Database table
Object	Currency	oaCurrency	—	currency
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Currency object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
rate	rate	The custom foreign exchange rate indicating how much of the counter currency (or quote currency) is needed to purchase one unit of the base currency.
symbol	symbol	[Required] The symbol for the counter currency (or quote currency). Must be the three-letter code for one of the standard currencies supported by OpenAir.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Usage Guidelines

Use the Currency object to set or read custom exchange rates. This is the equivalent of reading or setting custom exchange rates in Administration > Global Settings > Organization > Currencies > Set Exchange Rate when using the OpenAir UI.

The exchange rates in the Currency object and in the [currency](#) table are quoted against the default currency for the account. The default currency for the company's OpenAir account is defined by the `base_currency` property for the Company object.

Note: There are three object types you can use in the OpenAir XML API and SOAP API to interact with foreign currency exchange information:

- Use the [Currency](#) object to set or read custom exchange rates. This is the equivalent of reading or setting custom exchange rates in Administration > Global Settings > Organization > Currencies > Set Exchange Rate when using the OpenAir UI. The exchange rates in the Currency object and in the [currency](#) table are quoted against the default currency for the account. The default currency for the company's OpenAir account is defined by the `base_currency` property for the Company object.
- Use the [Currencyrate](#) to read current or historical exchange rates used for the account.
- Use the [ForexInput](#) object to add or modify entries in the exchange cross rate table when the Multicurrency feature is enabled for your company's account. Make sure you review the [Usage Guidelines](#) for the ForexInput object. This is the equivalent of editing exchange cross rates using the [Edit Exchange Cross Rates](#) in the OpenAir UI.

Currencyrate

A currency exchange rate [Currencyrate] is the historical (dated, past or future) exchange rate used in the account for the currency.

Review the [Usage Guidelines](#) for the Currencyrate object.

—	XML	SOAP	REST	Database table
Object	Currencyrate	oaCurrencyrate	—	—
Supported Commands	Read	read()	—	—

The Currencyrate object has the following properties:

XML / SOAP	Database	Description
cname	—	[Read-only] The name of the counter currency (or quote currency).
crate	—	[Read-only] The foreign exchange rate indicating how much of the counter currency (or quote currency) is needed to purchase one unit of the base currency on either the date specified, dates after the last date or dates prior to the first date in the exchange cross rate table.
csymbol	—	[Read-only] The symbol for the counter currency (or quote currency).
date	—	[Read-only] The date the quoted exchange rate applied, if dated. See Date Fields .
type	—	[Read-only] An Empty value if the quoted exchange rate is dated. Otherwise: <ul style="list-style-type: none"> ■ PAST – If the quoted exchange rate applied on dates prior to exchange rates for dates prior to the first date in the exchange cross rate table. ■ FUTURE – If the quoted exchange rate applied on dates prior to exchange rates for dates after the last date in the exchange cross rate table.

Usage Guidelines

Use the Currencyrate to read current or historical exchange rates used for the account.

When reading Currencyrate objects:

- Set the `base_currency` attribute to a three-letter currency code to get the exchange rates against the specified base currency. The foreign currency exchange rate is quoted against base currency for your company's OpenAir account otherwise. The base currency must be one of the currencies defined for the account in Administration > Global Settings > Organization > Currencies > Multi-currency.
- Use the `precision` attribute to set the decimal precision (maximum number of digits to the right of the decimal point) for foreign currency exchange rate values.
- To read the foreign exchange rates on a specific date or for a specific date range, use the `filter` and `field` attributes to specify a criteria, such as `date=equal-to` for example, based on the date property value.

For more information about using read attributes and filtering, see [Read Attributes](#) and [Filtering](#).

Note: There are three object types you can use in the OpenAir XML API and SOAP API to interact with foreign currency exchange information:

- Use the [Currency](#) object to set or read custom exchange rates. This is the equivalent of reading or setting custom exchange rates in Administration > Global Settings > Organization > Currencies > Set Exchange Rate when using the OpenAir UI. The exchange rates in the Currency object and in the `currency` table are quoted against the default currency for the account. The default currency for the company's OpenAir account is defined by the `base_currency` property for the Company object.
- Use the [Currencyrate](#) to read current or historical exchange rates used for the account.
- Use the [ForexInput](#) object to add or modify entries in the exchange cross rate table when the Multicurrency feature is enabled for your company's account. Make sure you review the [Usage Guidelines](#) for the ForexInput object. This is the equivalent of editing exchange cross rates using the [Edit Exchange Cross Rates](#) in the OpenAir UI.

CustField

A custom field [CustField] is a field defined for the OpenAir account for a specific record type to suit the company business requirements. Custom field may also be created automatically or manually to support optional features in OpenAir.

—	XML	SOAP	REST	Database table
Object	CustField	oaCustField	—	<code>cust_field</code>
Supported Commands	Read, Modify	<code>read(), modify()</code>	—	—

The CustField object has the following properties:

XML / SOAP	Database	Description
active	active	[Read-only] A 1/0 field indicating if this alert is active.

XML / SOAP	Database	Description
association	association	[Read-only] The table or object type this field is associated with. See the help topic Record Types and Forms Supporting Custom Fields for a list of associations.
checked	checked	[Read-only] A "1/0" field indicating if checkbox field is on by default
created	created	[Read-only] Time the record was created. See Date Fields .
decpos	decpos	[Read-only] The decimal size of the field.
defnow	defnow	[Read-only] A 1/0 field indicating if date fields default to today.
description	description	[Read-only] The description of the custom field.
divider	divider	[Read-only] A 1/0 field indicating whether to paint a divider.
divider_text	divider_text	[Read-only] Optional divider text.
force_unique	force_unique	[Read-only] A 1/0 field indicating if this field is unique.
hidden_data_entry	hidden_data_entry	[Read-only] A 1/0 field indicating whether the custom field should be hidden on the data entry UI.
hint	hint	[Read-only] The hint used on forms.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
maxlength	maxlength	[Read-only] The maximum length of data in the field.
mover	mover	[Read-only] A 1/0 field indicating if the selector should have mover controls.
name	name	[Read-only] The name of the custom field.
never_copy	never_copy	[Read-only] A 1/0 field indicating if the field can be cloned.
next_seq	next_seq	[Read-only] Next sequence number to use.
pick_source	pick_source	[Read-only] Picker source. For example: user, customer, etc.
picker	picker	[Read-only] The type of field for on screen representation: numeric, currency, date, text, textarea, check, radio, drop down, drop text, selector, or alloc_gr.
required	required	[Read-only] A 1/0 field indicating if this field is required.
rows	rows	[Read-only] The number of display rows for text area fields
seq	seq	[Read-only] The sequence number of the field.
size	size	[Read-only] The display size of the field on forms.
title	title	[Read-only] The title used on forms with this custom field.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
url	url	[Read-only] Default url for url custom fields
userid	user_id	[Read-only] The ID of the user who created or owns this custom field.

XML / SOAP	Database	Description
valuelist	valuelist	A list of values for radio groups and popup menu fields in csv format.

Customer

A customer [Customer] is an individual or company that is billed or expensed.

Review the [Usage Guidelines](#) for the Customer object.

—	XML	SOAP	REST	Database table
Object	Customer	oaCustomer	—	customer
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The Customer object has the following standard properties:

Note: Customer object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML	SOAP	Database	Description
active	active	active	A 1/0 field indicating whether this is designated as an active customer. Defaults to 1 if not set when adding a customer.
addr	—	—	The customer's address. An Address object with the company's address details. See Address Fields .
—	addr_addr1	address1	First line of the customer's address.
—	addr_addr2	address2	Second line of the customer's address.
—	addr_addr3	address3	Third line of the customer's address.
—	addr_addr4	address4	Fourth line of the customer's address.
—	addr_city	city	Customer's city.
—	addr_contact_id	—	The associated contact ID.
—	addr_country	country	Customer's country.
—	addr_email	email	Customer's email address.
—	addr_fax	fax	Customer's fax number.
—	addr_first	—	Customer's first name.
—	addr_id	—	The ID of the associated address.
—	addr_last	—	Customer's last name.
—	addr_middle	—	Customer's middle name.
—	addr_mobile	—	Customer's mobile phone number.

XML	SOAP	Database	Description
—	addr_phone	phone	Customer's phone number.
—	addr_salutation	—	Customer's salutation.
—	addr_state	state	Customer's state.
—	addr_zip	zip	Customer's zip code.
billingaddr	—	—	<p>The billing address. An Address object with the company's address details. See Address Fields.</p> <p>The billingaddr property is redundant but continues to be supported for backward compatibility.</p> <ul style="list-style-type: none"> You can use the <code>billing_contact_id</code> to designate a Contact object as the billing contact for the customer. If a billingaddr is specified when you add a customer object, a contact object is added with the information in billingaddr. If a billingaddr is specified when you modify a customer object, the contact object is also modified by default. To modify the billingaddr but not the contact object, set the <code>customer_only="yes"</code> attribute for the Address object as per the following example. <pre> 1 <Modify type="customer"> 2 <Customer> 3 <billingaddr> 4 <Address customer_on 5 ly="yes"> 6 <addr1>1234 Main 7 St</addr1> 8 </Address> 9 </billingaddr> 10 </Customer> 11 </Modify> </pre>
—	billing_addr_addr1 or billingaddr_addr1	—	First line on the billing address.
—	billing_addr_addr2 or billingaddr_addr2	—	Second line on the billing address.
—	billing_addr_addr3 or billingaddr_addr3	—	Third line on the billing address.
—	billing_addr_addr4 or billingaddr_addr4	—	Fourth line on the billing address.
—	billing_addr_city or billingaddr_city	—	City on the billing address.
—	billing_addr_contact_id or billingaddr_contact_id	primary_contact_id	The internal ID of the contact associated with the billing address.
—	billing_addr_country or billingaddr_country	—	Country on the billing address.

XML	SOAP	Database	Description
—	billing_addr_email or billingaddr_email	—	Email address on the billing address.
—	billing_addr_fax or billingaddr_fax	—	Fax number on the billing address.
—	billing_addr_first or billingaddr_first	—	First name on the billing address.
—	billing_addr_id or billingaddr_id	—	The ID of the associated billing address.
—	billing_addr_last or billingaddr_last	—	Last name on the billing address.
—	billing_addr_middle or billingaddr_middle	—	Middle name on the billing address.
—	billing_addr_ or billingaddr_	—	Mobile phone number on the billing address.
—	billing_addr_mobile or billingaddr_mobile	—	Phone number on the billing address.
—	billing_addr_salutation or billingaddr_salutation	—	Salutation on the billing address.
—	billing_addr_state or billingaddr_state	—	State on the billing address.
—	billing_addr_zip or billingaddr_zip	—	Zip code on the billing address.
billing_code	billing_code	billing_code	The customer billing code. Used in bulk invoicing.
billing_contact_id	billing_contact_id	billing_contact_id	The billing contact ID.
bus_typeid	bus_typeid	bus_type_id	Type of business this customer is in.
code	code	acct_code	Optional user-defined code.
company	company	company	[Required] The company name.
company_sizeid	company_sizeid	company_size_id	This customer's company size.
contactaddr	—	—	The primary contact's address. An Address object with the company's address details. See Address Fields .
—	contact_addr_addr1 or contactaddr_addr1	—	First line of the contact's address.
—	contact_addr_addr2 or contactaddr_addr2	—	Second line of the contact's address.
—	contact_addr_addr3 or contactaddr_addr3	—	Third line of the contact's address.
—	contact_addr_addr4 or contactaddr_addr4	—	Fourth line of the contact's address.
—	contact_addr_city or contactaddr_city	—	Contact's city.

XML	SOAP	Database	Description
—	contact_addr_contact_id or contactaddr_contact_id	billing_contact_id	The internal ID of the contact associated with the primary contact's address.
—	contact_addr_country or contactaddr_country	—	Contact's country.
—	contact_addr_email or contactaddr_email	—	Contact's email address.
—	contact_addr_fax or contactaddr_fax	—	Contact's fax number.
—	contact_addr_first or contactaddr_first	—	Contact's first name.
—	contact_addr_id or contactaddr_id	—	The ID of the associated contact address.
—	contact_addr_last or contactaddr_last	—	Contact's last name.
—	contact_addr_middle or contactaddr_middle	—	Contact's middle name.
—	contact_addr_mobile or contactaddr_mobile	—	Contact's mobile phone number.
—	contact_addr_phone or contactaddr_phone	—	Contact's phone number.
—	contact_addr_salutation or contactaddr_salutation	—	Contact's salutation.
—	contact_addr_state or contactaddr_state	—	Contact's state.
—	contact_addr_zip or contactaddr_zip	—	Contact's zip code.
cost_centerid	cost_centerid	cost_center_id	The ID of the associated cost center.
created	created	created	[Read-only] Time the record was created. See Date Fields .
createtime	createtime	—	[Read-only] Same as the created field (for legacy systems).
credit_invoice_layout_id	credit_invoice_layout_id	credit_invoice_layout_id	The internal ID of the credit memo (negative invoice) layout associated with the project.
currency	currency	currency	Currency for the money fields in the record. Also the default currency when an invoice is created.
customer_locationid	customer_locationid	customer_location_id	The internal ID of the customer location associated with the customer.
externalid	externalid	external_id	If the record was imported from an external system you store the unique external record ID here. Uniqueness may be enforced depending on your company's OpenAir account configuration.

XML	SOAP	Database	Description
filterset_ids	filterset_ids	—	Comma delimited list - filter sets this object belongs to.
hear_aboutid	hear_aboutid	hear_about_id	How did they hear about us.
hierarchy_node_ids	hierarchy_node_ids	—	Comma delimited list - hierarchy nodes this object belongs to.
id	id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
invoice_layoutid	invoice_layoutid	invoice_layout_id	The ID of the associated invoice layout.
invoice_prefix	invoice_prefix	invoice_prefix	Text to start every invoice number with.
invoice_text	invoice_text	invoice_text	Text to display on every invoice.
name	name	name	[Required] The nickname used for display in popup windows and lists.
notes	notes	notes	Notes about the customer.
picklist_label	picklist_label	—	Label as shown on form picklist.
primary_contactid	primary_contactid	primary_contact_id	The billing contact ID.
rate	rate	rate	Hourly billing rate for this customer.
shipping_contactid	shipping_contactid	shipping_contact_id	The shipping contact ID.
sold_to_contactid	sold_to_contactid	sold_to_contact_id	The sold to contact ID.
statements	statements	statements	A 1/0 field indicating if this customer can view statements.
ta_include	ta_include	—	A 1/0 field indicating whether a Timesheet filterset is applied.
tb_approvalprocess	tb_approvalprocess	tb_approvalprocess	The approvalprocess_id of the invoice approval process. This field is mutually exclusive with tb_approver."
tb_approver	tb_approver	tb_approver	The user_id of the invoice approver if this is a single approver process. This field is mutually exclusive with tb_approvalprocess. <ul style="list-style-type: none"> ■ If -1 then the approver is the owners manager. ■ If -2 then the approver is the owners manager's manager.
te_include	te_include	—	A 1/0 field indicating whether an Expense Report filterset is applied.
terms	terms	terms	Standard payment terms for the customer. Textual description like Net 30. Defaults to the default payment terms if not set when adding a new object.
territoryid	territoryid	territory_id	The territory for this customer.

XML	SOAP	Database	Description
type	type	type	A C/P field indicating whether this is Customer or a Prospect. Defaults to C if not set when adding a customer.
updated	updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
updatetime	updatetime	—	[Read-only] Same as the updated field (for legacy systems). See Date Fields .
userid	userid	user_id	The user ID of the customer or owner.
web	web	web	Customer's Web address.

Usage Guidelines

Review the following notes and guidelines:

- To work with information about both customers and prospects using the OpenAir XML API, use the [CustomerProspect](#) object.
- When adding a Customer object, the primary filter set for the authenticated user is updated automatically to allow access to the newly added customer.
- You cannot delete a Customer object if this object is referenced by an object of any of the following type. Delete any dependent objects first before you delete a Customer object.
 - [bulk_payment](#)
 - [Invoice](#)
 - [Payment](#)
 - [Project](#)
 - [Proposal](#)
 - [Proposalblock](#)
 - [Slip](#)
 - [Task](#)
 - [Ticket](#)

CustomerLocation

A customer location [CustomerLocation] is a geographical classification information for customers.

Review the [Usage Guidelines](#) for the CustomerLocation object

—	XML	SOAP	REST	Database table
Object	CustomerLocation	oaCustomerLocation	—	customer_location
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The CustomerLocation object has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this is an active customer location.
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the customer location.
notes	notes	Notes.
updated	updated	[Read-only] Time the record was last modified. See Date Fields .

Usage Guidelines

You cannot delete a CustomerLocation object if this object is referenced by a [Customer](#) or [CustomerProspect](#) object. Delete any dependent objects first before you delete a CustomerLocation object.

Customerpo

A customer PO [Customerpo] is a funding document for projects or purchases by a customer. Customer POs may be associated with a project then assigned to the project billing rules for the tracking of balances. One or more customer POs can be associated with a project. A customer PO may encompass several project.

Review the [Usage Guidelines](#) for the Customerpo object

—	XML	SOAP	REST	Database table
Object	Customerpo	oaCustomerpo	—	customerpo
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The Customerpo object has the following properties:

XML / SOAP	Database	Description
acct_date	acct_date	The accounting period date of the customerpo. See Date Fields .
active	active	A 1/0 field indicating whether this is an active customerpo. Defaults to 1 if not set when adding a customer PO.
code	acct_code	Optional accounting system code for integration with external accounting systems.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
customerid	customer_id	[Required] The ID of the associated customer.
date	date	The date of the customerpo. See Date Fields .
externalid	external_id	If the record was imported from an external system you store the unique Customerpo, external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.

XML / SOAP	Database	Description
name	name	[Required] The name of the customerpo.
notes	notes	Notes.
number	number	The customerpo number.
picklist_label	—	Label as shown on form picklist.
total	total	The customerpo total. Dated by the date field.
updated	updated	[Read-only] Time the record was last modified. See Date Fields .

Usage Guidelines

You cannot delete a Customerpo object if this object is referenced by an object of any of the following type. Delete any dependent objects first before you delete a Customerpo object.

- [Customerpo_to_project](#)
- [Projectbillingrule](#)
- [Projectbillingtransaction](#)
- [Revenue_recognition_rule](#)
- [Revenue_recognition_rule_amount](#)
- [Revenue_recognition_transaction](#)
- [Slip](#)
- [SlipProjection](#)

Customerpo_to_project

Use the customer PO-project link [Customerpo_to_project] object to create many-to-many links between projects and agreements.

—	XML	SOAP	REST	Database table
Object	Customerpo_to_project	oaCustomerpo_to_project	—	customerpo_to_project
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Customerpo_to_project object has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this is an active customerpo.
created	created	[Read-only] Time the record was created. See Date Fields .
customerid	customer_id	The ID of the associated customer. Set automatically when adding or modifying an object if projectid is set.
customerpo_id	customerpo_id	The ID of the associated customerpo.
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.

XML / SOAP	Database	Description
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
projectid	project_id	The ID of the associated project.
updated	updated	[Read-only] Time the record was last modified. See Date Fields .

CustomerProspect

A prospect [CustomerProspect] is a prospective customer.

The OpenAir XML API uses the CustomerProspect object type to work with information about both customers and prospects. When reading prospect objects, both customers and prospects are returned.

For object properties and usage guidelines, see [Customer](#).

—	XML	SOAP	REST	Database table
Object	CustomerProspect	oaCustomer	—	customer
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

Deal

A deal [Deal] is a potential sale to a prospect or customer.

—	XML	SOAP	REST	Database table
Object	Deal	oaDeal	—	deal
Supported Commands	Read	read()	—	—

The Deal object has the following properties:

XML / SOAP	Database	Description
active	active	[Read-only] A 1/0 field indicating whether this is designated as an active deal.
closed	closed	[Read-only] When this deal was closed.
created	created	[Read-only] Time the record was created. See Date Fields .
customerid	customer_id	[Read-only] The ID of the associated customer.
exported	exported	[Read-only] Date and time the record was marked as exported. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name/description of the deal.
notes	notes	[Read-only] Notes for this deal.
opened	opened	[Read-only] When this deal was first opened.
rating	rating	[Read-only] The rating for this deal.

XML / SOAP	Database	Description
stage	deal_stage.percentage	[Read-only] The % of the work complete for this deal.
status	status	[Read-only] The status for this deal: O - Open, C - Closed, L - Lost
territoryid	territory_id	[Read-only] The territory for this deal.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	userid	[Read-only] The ID of the associated user.

Dealcontact

A deal contact [Dealcontact] links a contact with a deal.

—	XML	SOAP	REST	Database table
Object	Dealcontact	oaDealcontact	—	deal_contact
Supported Commands	Read	read()	—	—

The Dealcontact object has the following properties:

XML / SOAP	Database	Description
contactid	contact_id	[Read-only] The related contact.
created	created	[Read-only] Time the record was created. See Date Fields .
dealid	deal_id	[Read-only] The deal ID.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Dealschedule

A deal schedule [Dealschedule] is a milestone in a deal when a portion of the deal should be closed and a portion of the deal value should be secured. The total value of a deal can be broken down into smaller portions, and each portion has a potential closing date and a portion of the full deal value.

—	XML	SOAP	REST	Database table
Object	Dealschedule	oaDealschedule	—	deal_schedule
Supported Commands	Read	read()	—	—

The Dealschedule object has the following properties:

XML / SOAP	Database	Description
amount	amount	[Read-only] The amount this portion of the deal is worth (in the currency of the deal). Dated by the date field.
created	created	[Read-only] Time the record was created. See Date Fields .

XML / SOAP	Database	Description
date	date	[Read-only] The potential closing date for a deal portion. See Date Fields .
dealid	deal_id	[Read-only] ID of the deal associated with this deal portion.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Department

A department [Department] is a division or a group of employees in your company.

—	XML	SOAP	REST	Database table
Object	Department	oaDepartment	—	department
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Department object has the following standard properties:

 **Note:** Department object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The name used for display in lists.
notes	notes	Notes about the department.
picklist_label	—	Label as shown on form picklist.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	The user ID of the head of the department.

Entitytag

An entity tag [Entitytag] is a custom classification tool that allows to organize entities into groups for reporting purposes.

Review the [Usage Guidelines](#) for the Entitytag object.

—	XML	SOAP	REST	Database table
Object	Entitytag	oaEntitytag	—	entity_tag
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Entitytag object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
customerid	customer_id	The ID of the associated customer.
default_for_entity	default_for_entity	A 1/0 field indicating whether this is the default row for this entity.
end_date	end_date	End date for this entity_tag. See Date Fields .
externalid	external_id	If the record was imported from an external system you store the unique external record id here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
projectid	project_id	The ID of the associated project.
start_date	start_date	Start date for this entity_tag. See Date Fields .
tag_group_attribute_name	tag_group_attribute_name	The name of the associated tag group attribute.
tag_group_attributeid	tag_group_attribute_id	[Required] The ID of the associated tag_group_attribute.
tag_group_id	tag_group_id	The ID of the associated tag group attribute.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Required] The ID of the associated user.

Usage Guidelines

Review the following guidelines:

- You can use the following read attributes when reading Entitytag objects: `between_date` and `tag_with_name`. See [Read Attributes](#).
- If `default_for_entity` is set to 1, you cannot set a `start_date` or `end_date`.
- When you add or modify a [User](#), you can modify the Entitytag associated with that [User](#). See [Updating User Entity Tags](#).

Envelope

An expense report [Envelope] is a collection of expense items (receipts) that employees can use in OpenAir to claim reimbursement.

Review the [Usage Guidelines](#) for the Envelope object.

—	XML	SOAP	REST	Database table
Object	Envelope	oaEnvelope	ExpenseReport	envelope
Supported Commands	Add, Read, Modify, Delete, Submit, Approve, Reject, Unapprove, Report	add(), createUser(), read(), modify(), upsert(), delete(), submit(), approve(), reject(), unapprove()	See the help topic Available methods	—

The Envelope object has the following standard properties:

 **Note:** Envelope object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	REST	Database	Description
acct_date	accountingDate	acct_date	The accounting period date of the expense report.
advance	advance	advance	The amount of any cash advance on the expense report.
approved	approveDate	date_approved	[Read-only] The date the expense report was approved. See Date Fields
approver	—	approver	The internal ID of the employee who approved the expense report.
attachmentid	attachments	attachment_id	[Read-only] The attachments associated with this expense report. Comma-separated list (SOAP, XML) or array (REST) of internal IDs for attachment objects.
balance	balance	balance	The outstanding balance on the expense report.
created	created	created	[Read-only] The date the expense report was created. See Date Fields
currency	currency	currency	The currency for monetary values in the expense report. Three-letter currency code.
currency_exchange_intolerance	isForeignCurrencyExchange Intolerance	currency_exchange_intolerance	A 1/0 field indicating if the record is within the specified foreign currency tolerance as defined in database data definitions.
customerid	customerId	customer_id	The internal ID of the associated customer.
date	date	date	[Required] The date of the expense report. See Date Fields
date_end	endDate	date_end	The ending date of the expense report (only used with auto-naming). See Date Fields

XML / SOAP	REST	Database	Description
date_start	startDate	date_start	The starting date of the expense report (only used with auto-naming). See Date Fields
—	description	description	The description of the expense report. This can hold the reason for the trip, for example.
—	exported	exported	The date and time the record was marked as "exported". See Date Fields
externalid	externalId	external_id	The unique external ID of the expense report, if the record was imported from an external system.
id	id	id	[Read-only] The unique internal identifier of the expense report. Automatically assigned by OpenAir.
—	isAccounting	accounting	[Read-only] A 1/0 field indicating if an envelope was submitted to an accounting partner.
—	isAdjusting	is_adjusting	[Read-only] A 1/0 field indicating if the expense report is an adjusting expense report.
is_overlapping	—	is_overlapping	[Read-only] A 1/0 field indicating if the expense report overlaps with another expense report.
name	name	name	[Required] The name of the expense report. Required unless your company's account is configured to set the name automatically
notes	notes	notes	Notes about the expense report.
number	trackingNumber	number	The expense report tracking number. A value is automatically assigned if not set when adding an object.
projectid	projectId	projectid	The internal ID of the associated project.
status	status	status	The status of the expense report. Defaults to 0 if not set when adding an object. Possible values: <ul style="list-style-type: none"> ■ 0 — open ■ S — submitted ■ A — approved ■ R — rejected
submitted	submitDate	date_submitted	[Read-only] The date the expense report was submitted. See Date Fields

XML / SOAP	REST	Database	Description
tax_locationid	taxLocationId	tax_location_id	Default tax location for this expense report.
thin_client_id	—	thin_client_id	Used by thin clients to reconcile imported records.
total	total	total	The total value of all the receipts in the expense report.
totreimburse	totalReimburse	total_to_reimburse	[Read-only] The total amount of reimbursable expenses in the expense report.
tottickets	totalReceipts	total_tickets	[Read-only] total number of receipts in the expense report.
trip_reason	—	trip_reason	The reason for the trip.
updated	updated	updated	[Read-only] The date the expense report was last updated or modified. See Date Fields
userid	userId	user_id	The internal ID of the associated employee. Defaults to the authenticated user if not set when adding an object.

Usage Guidelines

Review the following guidelines:

- Your company's OpenAir account can be configured to allow using the OpenAir API to modify approved expense reports. To enable the feature, contact OpenAir Customer Support.
- The tracking number [number] must be unique for each object. When adding or modifying an object, if you set number make sure the value is unique.
- If your company's OpenAir account is configured to name expense reports automatically, and no valid date_start value is set, date_start and date_end are set automatically.
- Depending on your account configuration:
 - Modifying or deleting expense reports associated to a user other than the authenticated user unless the authenticated user is an account administrator.
 - Account administrators may not delete open and submitted expense reports associated to another user.
- If your company's account configuration supports the unapprove operation, expense reports cannot be unapproved if there are any reimbursements associated with the expense report, or, depending on your company's account configuration if the expense report was exported.
- You cannot delete an Envelope object if this object is referenced by an object of any of the following type. Delete any dependent objects first before you delete an Envelope object.
 - [Reimbursement](#)
 - [Ticket](#)

Estimate

An estimate [Estimate] is an approximate calculation of resource costs, fixed costs, and discounts, that is used to create profit margin estimates for pipeline deals.

—	XML	SOAP	REST	Database table
Object	Estimate	oaEstimate	—	estimate
Supported Commands	Read	read()	—	—

The Estimate object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
customerid	customer_id	[Read-only] The ID of the associated customer.
dealid	deal_id	[Read-only] The ID of the associated deal.
hide_expense	hide_expense	[Read-only] A 1/0 field indicating if expenses should be hidden in analysis report.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The short description for the estimate.
notes	notes	[Read-only] Notes about the estimate.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Estimateadjustment

An estimate adjustment [Estimateadjustment] are changes made to an estimate.

—	XML	SOAP	REST	Database table
Object	Estimateadjustment	oaEstimateadjustment	—	estimate_adjustment
Supported Commands	Read	read()	—	—

The Estimateadjustment object has the following properties:

XML / SOAP	Database	Description
adjustment_type	adjustment_type	[Read-only] A 1/0 field indicating the adjustment is for labor or expenses. If 1 - then adjustment is for labor. If 0 - then adjustment is for expenses.
amount	amount	[Read-only] The amount of adjustment in money (in the currency of the estimate) or percentage of total expense or labor. The actual type is identified by amount_type field.

XML / SOAP	Database	Description
amount_type	amount_type	[Read-only] A 1/0 field indicating the type of the amount field. If 1 - then amount field represents percentage of time. If 0 - then amount field represents number of hours.
created	created	[Read-only] Time the record was created. See Date Fields .
estimateid	estimate_id	[Read-only] The ID of the associated estimate.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name for the estimate adjustment.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Estimateexpense

An estimate expense [Estimateexpense] is an anticipated expense associated with a pipeline opportunity and includes information about any expense-related markup.

—	XML	SOAP	REST	Database table
Object	Estimateexpense	oaEstimateexpense	—	estimate_expense
Supported Commands	Read	read()	—	—

The Estimateexpense object has the following properties.

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
date	date	[Read-only] Date for the expense. See Date Fields .
description	description	[Read-only] The short description for the estimate.
estimateid	estimate_id	[Read-only] The ID of the associated estimate.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
itemid	item_id	[Read-only] The ID of the associated expense item.
markup	markup	[Read-only] The amount of markup in percent or money as designated by markup_type field. Dated by the date field.
markup_type	markup_type	[Read-only] A 1/0 field indicating the type of expense markup. <ul style="list-style-type: none"> ■ If 1 - then use percentage of the cost. ■ If 0 - then use the specific amount.
phaseid	phase_id	[Read-only] The ID of the associated estimate phase.
price	price	[Read-only] The cost of the expense. Dated by the date field.
quantity	quantity	[Read-only] The quantity for the expense.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Estimatelabor

An estimate labor [Estimatelabor] is an anticipated resource (staffing) costs associated with a pipeline opportunity and includes both loaded cost and billing rate information.

—	XML	SOAP	REST	Database table
Object	Estimatelabor	oaEstimatelabor	—	estimate_labor
Supported Commands	Read	read()	—	—

An Estimatelabor object has the following properties:

XML / SOAP	Database	Description
amount	amount	[Read-only] The number of hours or percentage of time associated with a given resource for a specific phase of an estimate. The actual type is identified by as_percentage field.
amount_type	amount_type	[Read-only] A 1/0 field indicating the type of the amount field. <ul style="list-style-type: none"> ■ If 1 - then amount field represents percentage of time. ■ If 0 - then amount field represents number of hours.
billing_rate	billing_rate	[Read-only] The billing rate for the associated resource. Dated by the start_date field.
created	created	[Read-only] Time the record was created. See Date Fields .
description	description	[Read-only] The short description for the estimate.
end_date	end_date	[Read-only] End date for resource assignment. See Date Fields .
estimateid	estimate_id	[Read-only] The ID of the associated estimate.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
loaded_cost	loaded_cost	[Read-only] The loaded cost for the associated resource. Dated by the start_date field.
phaseid	phase_id	[Read-only] The ID of the associated estimate phase.
start_date	start_date	[Read-only] Start date for resource assignment. See Date Fields .
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Read-only] The ID of the associated resource.

Estimatemarkup

An estimate markup [Estimatemarkup] is an anticipated markup on expenses associated with a pipeline opportunity.

—	XML	SOAP	REST	Database table
Object	Estimatemarkup	oaEstimatemarkup	—	estimate_markup
Supported Commands	Read	read()	—	—

The Estimatemarkup object has the following properties.

XML / SOAP	Database	Description
as_percentage	as_percentage	[Read-only] A 1/0 field indicating which expense markup to use: <ul style="list-style-type: none"> ■ If 1 - then use percentage of the total, compute total markup. ■ If 0 - then use the specific amount, compute percent markup.
created	created	[Read-only] Time the record was created. See Date Fields .
estimateid	estimate_id	[Read-only] The ID of the associated estimate.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
percent	percent	[Read-only] The percentage markup to add to the total expense amount.
phaseid	phase_id	[Read-only] The ID of the associated estimate phase.
total	total	[Read-only] The amount of expense (in the currency of the estimate) to use for this estimate in calculations.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Estimatephase

An estimate phase [Estimatephase] identifies a package a work to be delivered as part of the estimated project opportunity.

—	XML	SOAP	REST	Database table
Object	Estimatephase	oaEstimatephase	—	estimate_phase
Supported Commands	Read	read()	—	—

The Estimatephase object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
estimateid	estimate_id	[Read-only] The ID of the associated estimate.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name for the estimate phase.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Event

An event [Event] is a historical record of an activity performed on behalf of a customer or prospect. It could record the completion of a todo, the closing of a deal, or document a phone call or email message sent to a customer.

—	XML	SOAP	REST	Database table
Object	Event	oaEvent	—	event
Supported Commands	Add, Read, Modify	add(), read(), modify(), upsert()	—	—

The Event object has the following properties:

XML / SOAP	Database	Description
contactid	contact_id	The ID of the associated contact.
created	created	[Read-only] Time the record was created. See Date Fields
customerid	customer_id	The ID of the associated customer.
dealid	deal_id	The ID of the associated deal.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The name or description of the event.
notes	notes	Notes related to the event.
occurred	occurred	The date of the event. See Date Fields
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields
userid	user_id	The ID of the user who created the event.

ExpensePolicy

An expense policy [ExpensePolicy] is a set of rules to enforce any restrictions and limits on the expenses employees can claim. There can be a default expense policy for the company and expense policies associated to a specific project.

Review the [Usage Guidelines](#) for the ExpensePolicy object

—	XML	SOAP	REST	Database table
Object	ExpensePolicy	oaExpensePolicy	—	expense_policy
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The ExpensePolicy object has the following properties:

XML / SOAP	Database	Description
all_items_allowed	all_items_allowed	A 1/0 field indicating that all expense items are allowed by this expense policy.

XML / SOAP	Database	Description
created	created	[Read-only] The time the record was created. See Date Fields
customerid	customer_id	[Required] The ID of the associated customer.
deleted	deleted	A 1/0 field indicating if the record was deleted.
description	description	Optional information about expense policy.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
projectid	project_id	[Required] The ID of the project which expense policy is associated to. Any project can only have one expense policy.
updated	updated	[Read-only] The time the record was last modified. See Date Fields

Usage Guidelines

You cannot delete an ExpensePolicy object if this object is referenced by an [ExpensePolicyItem](#) object. Delete any dependent objects first before you delete an ExpensePolicy object.

ExpensePolicyItem

An expense policy item [ExpensePolicyItem] is a rule to enforce restrictions and limits applicable to a specific expense item as part of an expense policy.

Review the [Usage Guidelines](#) for the ExpensePolicyItem object.

—	XML	SOAP	REST	Database table
Object	ExpensePolicyItem	oaExpensePolicyItem	—	expense_policy_item
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The ExpensePolicyItem object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] The time the record was created. See Date Fields
currency	currency	Currency of fixed/max price.
deleted	deleted	A 1/0 field indicating if the record was deleted.
expense_policyid	expense_policy_id	[Required] The ID of the expense policy which this item belongs to.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
itemid	item_id	[Required] The ID of the item which this record belongs to.

XML / SOAP	Database	Description
price_fixed	price_fixed	If set this item has defined fixed price which cannot be overridden in the ticket form.
price_max	price_max	If set this item has a defined maximum price.
updated	updated	[Read-only] The time the record was last modified. See Date Fields

Usage Guidelines

There cannot be two expense policy item [ExpensePolicyItem] objects with the same expense_policyid and itemid combination. An error is returned if the operation would result in a duplicate expense_policyid and itemid combination.

Filter

Filter [Filter] objects limit the authenticated user to a subset of business objects of a certain type. The Filter objects designate the business objects of a particular type that the authenticated user can or cannot view. Filter objects are grouped into filter sets. See [Filterset](#).

Review the [Usage Guidelines](#) for the Filter object.

—	XML	SOAP	REST	Database table
Object	Filter	—	—	filter
Supported Commands	Read (all)	—	—	—

 **Note:** The Filter object supports the all read method only.

The Filter object has the following properties:

XML	Database	Description
id	filter_id	[Read-only] Unique ID. Automatically assigned by OpenAir.

Usage Guidelines

The Filter supports only the [Read](#) with the all method.

The Filter object element must include the attribute type. The type attribute value indicates the business object type. The only supported value is customer.

You can use OpenAir XML API to read the internal IDs of Customer objects that the authenticated user can or cannot view. For example, you can use the following request:

```
1 <Read type="Filter" method="all" limit="1000">
2 </Read>
```

In the sample response, positive internal ID [id] values reference specific business objects, and negative values reference metavalues relative to the user. The internal IDs 0 and 1 should be ignored.

```

1 <Read status = "0">
2   <Filter type="customer">
3     <id>1</id>
4     <id>0</id>
5     <id>21</id>
6     <id>24</id>
7     <id>39</id>
8     <id>-5</id>
9     <id>-17</id>
10    <id>-7</id>
11  </Filter>
12 </Read >

```



Important: When defining filter sets and filters in the OpenAir UI, account administrators can choose whether to include the selected values (that is, select the business objects that user with the filter set **can** view) or to exclude the selected values (that is, select the business objects that user with the filter set **cannot** view). This is represented by the column **exclude** in the **filter** database table. However, the **Filter** object in the XML API does not include this object property.

Filterset

A filter set [Filterset] defines what data a user has permission to view or update.

—	XML	SOAP	REST	Database table
Object	Filterset	oaFilterset	—	filter_set
Supported Commands	Read	read()	—	—

The Filterset object has the following properties:

XML / SOAP	Database	Description
active	active	[Read-only] A 1/0 field indicating whether this is designated as an active filter set.
all_access	all_access	[Read-only] A 1/0 field indicating this filterset does not filter anything and cannot be deleted.
created	created	[Read-only] Time the record was created. See Date Fields .
default_filter_set	default_filter_set	[Read-only] A 1/0 field indicating whether this is the default new-user filterset.
externalid	external_id	[Read-only] If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name of the filterset.
notes	notes	[Read-only] Notes related to the filterset.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

ForexInput

A forex input [ForexInput] is a company override value for historical (dated, past or future) foreign currency exchange rates between currencies enabled as a business transaction currency or user-defined reporting currency when the Multicurrency feature is enabled for the OpenAir account.

For more information about the Multicurrency feature, see the help topic [Multicurrency](#).

Review the [Usage Guidelines](#) for the ForexInput object.

—	XML	SOAP	REST	Database table
Object	ForexInput	oaForexInput	—	forex
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The ForexInput object has the following properties:

XML / SOAP	Database	Description
base	symbol	The symbol for the base currency used for the foreign exchange rate. Must be one of the user-defined reporting currencies defined for the OpenAir account. Defaults to the default currency for the company's OpenAir account if not set when adding or modifying an object.
created	created	[Read-only] Date the record was created. See Date Fields .
enddate	<i>Last in a range of forex_date values</i>	The last in the date range when the foreign exchange rate applies. See Date Fields . Must be empty if past or future is set.
future	<i>Determines the value for jkey: FUTURE<symbol> where <symbol> is the uppercase symbol for the base currency.</i>	Set to 1 if the conversion rate is for dates after the last date in the exchange cross rate table. If used, startdate and enddate must be empty.
past	<i>Determines the value for jkey: PAST<symbol> where <symbol> is the uppercase symbol for the base currency.</i>	Set to 1 if the conversion rate is for dates prior to the first date in the exchange cross rate table. If used, startdate and enddate must be empty.
rate	custom_<currency_symbol> where <currency_symbol> is the lowercase symbol for the counter currency.	The foreign exchange rate indicating how much of the counter currency (or quote currency) is needed to purchase one unit of the base currency.
startdate	<i>First in a range of forex_date values</i>	The first in the date range when the foreign exchange rate applies. See Date Fields . Must be empty if past or future is set.
symbol	<i>Determines the name of the custom field custom_<currency_symbol> where <currency_symbol> is the lowercase symbol for the counter currency.</i>	The symbol for the counter currency (or quote currency). Must be for one of the currencies enabled as a business transaction

XML / SOAP	Database	Description
		currency or one of the user-defined reporting currencies for the OpenAir account.
updated	updated	[Read-only] Date the record was last modified. See Date Fields .

Usage Guidelines

Use the [ForexInput](#) object, to add or modify entries in the exchange cross rate table when the Multicurrency feature is enabled for your company's account. This is the equivalent of editing exchange cross rates using the [Edit Exchange Cross Rates](#) in the OpenAir UI.

- The base currency must be one of the user-defined reporting currencies, if set. It is not possible to set historical foreign currency exchange rates with a base currency is set to any other currency – OpenAir API returns error code 837 with the following error message: "Invalid base currency specified. It must be one of user-defined currencies.". For more information about user-defined reporting currencies, see the help topic [User-Defined Reporting Currencies](#).

You can set historical foreign exchange rates against the default currency for your company's OpenAir account as base currency. To do so, leave the base property value empty.

- symbol must be the symbol for one of the currencies your company uses for business transactions as defined in Administration > Global Settings > Organization > Currencies > Multi-currency, or one of the user-defined reporting currencies.
- startdate and enddate, if set, must be within 10 years from the current date at the time of the operation. The date range defined by startdate and enddate must not span longer than 10 years.

Note: There are three object types you can use in the OpenAir XML API and SOAP API to interact with foreign currency exchange information:

- Use the [Currency](#) object to set or read custom exchange rates. This is the equivalent of reading or setting custom exchange rates in Administration > Global Settings > Organization > Currencies > Set Exchange Rate when using the OpenAir UI. The exchange rates in the Currency object and in the [currency](#) table are quoted against the default currency for the account. The default currency for the company's OpenAir account is defined by the base_currency property for the Company object.
- Use the [Currencyrate](#) to read current or historical exchange rates used for the account.
- Use the [ForexInput](#) object to add or modify entries in the exchange cross rate table when the Multicurrency feature is enabled for your company's account. Make sure you review the [Usage Guidelines](#) for the ForexInput object. This is the equivalent of editing exchange cross rates using the [Edit Exchange Cross Rates](#) in the OpenAir UI.

Fulfillment

A fulfillment [Fulfillment] is an acknowledgment that the object of a purchase order (goods or services) has been received.

—	XML	SOAP	REST	Database table
Object	Fulfillment	oaFulfillment	—	fulfillment
Supported Commands	Read, Modify	read(), modify()	—	—

The Fulfillment object has the following standard properties:

 **Note:** Fulfillment object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
acct_date	acct_date	The accounting period date of the fulfillment. See Date Fields .
carrier_id	carrier_id	Associated carrier ID.
created	created	[Read-only] Time the record was created. See Date Fields .
date	date	Date of the fulfillment. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
notes	notes	Fulfillment description notes.
purchase_item_id	purchase_item_id	Associated purchase item ID.
purchaseorder_id	purchaseorder_id	[Required] Associated purchase order ID.
purchaserequest_id	purchaserequest_id	Associated purchase request ID.
quantity	quantity	The quantity received.
request_item_id	request_item_id	Associated request item ID.
slip_id	slip_id	The ID of the associated slip if this expense was billed to a time bill.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
waybill_number	waybill_number	The waybill number.

Hierarchy

A hierarchy [Hierarchy] is a multilevel classification for users, customers or projects.

—	XML	SOAP	REST	Database table
Object	Hierarchy	oaHierarchy	—	hierarchy
Supported Commands	Add, Read, Modify	add(), read(), modify(), upsert()	—	—

The Hierarchy object has the following standard properties:

Note: Hierarchy object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this is designated as an active hierarchy. Defaults to 1 if not set when adding a hierarchy.
available_as_column	available_as_column	A 1/0 field indicating whether this hierarchy is available as a (customer, project or user) list column.
created	created	[Read-only] Time the record was created. See Date Fields .
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The hierarchy name.
notes	notes	Notes related to the hierarchy.
primary_dropdown_filter	primary_dropdown_filter	A 1/0 field indicating if this hierarchy is used as a drop-down filter. Must be 0 if type is project. Only one hierarchy for the same type can be used as a drop-down filter.
primary_user_filterset	primary_user_filterset	A 1/0 field indicating if this hierarchy determines filter set access for projects. Must be 0 if type is project. Only one hierarchy for the same type can be used to determine filter set access for projects.
required	required	A 1/0 field indicating whether this hierarchy should be a required element on the object type form.
requireonform	requireonform	A 1/0 field indicating whether this hierarchy should be added to the object type form.
type	type	The type (table name) of the hierarchy. Possible values: customer, project, or user
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

HierarchyNode

A hierarchy node [HierarchyNode] is a node in a classification tree.

Review the [Usage Guidelines](#) for the HierarchyNode object.

—	XML	SOAP	REST	Database table
Object	HierarchyNode	oaHierarchyNode	—	hierarchy_node

—	XML	SOAP	REST	Database table
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The HierarchyNode object has the following standard properties:

Note: HierarchyNode object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
hierarchyid	hierarchy_id	[Required] The ID of the associated hierarchy.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
isalevel	is_a_level	A 1/0 field indicating if this node is a level. Must be 0 if is_a_node is set to 1.
isanode	is_a_node	A 1/0 field indicating if this node is a level. Must be 0 if is_a_level is set to 1.
levelid	level_id	The ID of the associated hierarchy level. Must be 0 if is_a_level is set to 1. The level with internal ID <code>levelid</code> must also be associated with the hierarchy with internal ID <code>hierarchyid</code> .
name	name	The name of the hierarchy level or node. Required if either <code>is_a_node</code> or <code>is_a_level</code> is set to 1.
notes	notes	Notes related to the hierarchy node.
parentid	parent_id	The <code>hierarchy_node</code> ID of our immediate ancestor. If 0 or null, the object is a top-level node. The parent node with internal ID <code>parentid</code> must also be associated with the hierarchy with internal ID <code>hierarchyid</code> . If <code>is_a_node</code> is set to 1, there must be a level under the parent's level. Required if <code>record_id</code> is set and parent must be a node.
recordid	record_id	The record ID if not a node. Must be 0 if <code>is_a_level</code> is set to 1. Required if both <code>is_a_node</code> and <code>is_a_level</code> are set to 0.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Usage Guidelines

There cannot be two HierarchyNode objects with the same `hierarchyid`, `parentid` and `recordid` combination.

It is not possible to modify a record – hierarchy node association.

History

An approval history event [History] is an event in the chain of approval for an expense report or timesheet.

—	XML	SOAP	REST	Database table
Object	History	oaHistory	—	approval
Supported Commands	Read (equal to)	read() (equal to)	—	—

Note: The History object supports the `equal` to read method only.

The History has the following properties:

XML / SOAP	Database	Description
action	action	[Read-only] The approval action: S - Submittal, P - Pending, A - Acceptance, R - Rejection, U - unapproval.
created	created	[Read-only] Time the record was created. See Date Fields .
date	date	[Read-only] The date associated with this approval history event. See Date Fields .
envelopeid	envelope_id	[Read-only] The ID of the associated envelope.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
notes	notes	[Read-only] Notes associated with the approval history event.
projectid	project_id	[Read-only] The ID of the associated project.
timesheetid	timesheet_id	[Read-only] The ID of the associated timesheet.
userid	user_id	[Read-only] The ID of the user associated with this history event.

HistoryNotes

An approval history event [HistoryNotes] is an event in the chain of approval for an expense report or timesheet.

Review the [Usage Guidelines](#) for the HistoryNotes object.

—	XML	SOAP	REST	Database table
Object	HistoryNotes	oaHistoryNotes	—	approval
Supported Commands	Read (equal to)	—	—	—

Note: The HistoryNotes object supports the `equal` to read method only.

The HistoryNotes has the following properties:

XML	Database	Description
action	action	[Read-only] The approval action: S - Submittal, P - Pending, A - Acceptance, R - Rejection, U - unapproval.

XML	Database	Description
date	date	[Read-only] The date associated with this history event. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
notes	notes	[Read-only] Notes associated with the history event.
parentid	—	[Read-only] The internal ID of the object.
type	—	[Read-only] The object type – Supports "Envelope" and "TimeSheet" (with an upper case S) only.

Usage Guidelines

Review the following guidelines:

- The HistoryNotes object type is listed in the OpenAir WSDL but OpenAir SOAP API does not support this object type. Use the [History](#) object type instead with OpenAir SOAP API.
- When reading HistoryNotes object with the `equal` to method, the argument object must be an [Envelope](#) or a TimeSheet object with a valid `id` property.



Important: When reading the approval history of a timesheet, the argument object must be a TimeSheet (with an uppercase S) and not a Timesheet (with an lowercase s) like you would use when working with timesheets.

The following example shows the syntax used to read the approval history of an expense report with the OpenAir XML API:

```

1 <Read type="HistoryNotes" method="equal to" limit="100">
2   <Envelope>
3     <id>471</id>
4   </Envelope>
5 </Read>

```

The following example shows the syntax used to read the approval history of a timesheet with the OpenAir XML API:

```

1 <Read type="HistoryNotes" method="equal to" limit="100">
2   <TimeSheet>
3     <id>471</id>
4   </TimeSheet>
5 </Read>

```

ImportExport

An import / export entry [ImportExport] is a trace of when an OpenAir record was last imported or exported to an external application.

Review the [Usage Guidelines](#) for the ImportExport object.

—	XML	SOAP	REST	Database table
Object	ImportExport	oaImportExport	—	import_export

—	XML	SOAP	REST	Database table
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The ImportExport object has the following properties:

XML / SOAP	Database	Description
application	application	String describing the application making the association.
exported	last_export	Time of the last export from OpenAir. Required on import. See Date Fields .
externalid	external_id	Record identifier in the external application.
id	record_id	[Required] Internal ID of the imported or exported record (slip, task, etc.) in its native table.
imported	last_import	Time of the last import to OpenAir. Required on import. See Date Fields .
type	table_name	[Required] Exported object type. This is the case sensitive object type as used by the OpenAir API, such as Slip, Task, Projectassign, for example. For a list of supported object types, see List of Supported Business Object Types . For SOAP APO objects, omit the oa prefix.

Usage Guidelines

Review the following guidelines:

- When reading objects of other types, you can use the not-exported filter to return only objects that have not been marked as exported. Include an ImportExport argument object to specify the application that the read objects should be exported to. See [Read Not Exported Charges with all Method — C#](#) and [Read Not Exported Charges with all Method — Java](#).
- At least one of imported and exported is required.

Invoice

An invoice [Invoice] is a collection of charges [[Slip](#)] for goods and services rendered that you issue to request an amount payable from a customer.

Review the [Usage Guidelines](#) for the Invoice object.

—	XML	SOAP	REST	Database table
Object	Invoice	oaInvoice	—	invoice
Supported Commands	Add, Read, Modify, ModifyOnCondition, Delete, Submit, Approve, Reject, Unapprove	add(), read(), modify(), upsert(), delete(), submit(), approve(), reject(), unapprove()	—	—

The Invoice object has the following standard properties:

Note: Invoice object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom_read` attribute.

XML / SOAP	Database	Description
access_log	access_log	The mailing and access history of the invoice, such as when the customer accessed it.
accounting	accounting	A 1/0 field indicating if an invoice has been sent to an accounting partner.
acct_date	acct_date	The accounting period date of the invoice. See Date Fields .
approval_status	approval_status	A one-character string indicating the approval status of the invoice. Only used if invoice approvals are used. Possible values: <ul style="list-style-type: none"> ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected
approved	date_approved	Date the invoice was approved. See Date Fields .
attachmentid	attachment_id	The ID of the associated attachment.
balance	balance	[Read-only] The outstanding balance on the invoice. Dated by the date field.
contactid	contact_id	The contact ID for this invoice.
created	created	[Read-only] Time the record was created. See Date Fields .
credit	credit	The amount of any credit against the invoice. Dated by the date field.
credit_reason	credit_reason	The reason for the credit.
credit_rebill_status	credit_rebill_status	Credit/Rebill status for the original invoice: C = Credit Initiated and R = Re-Bill.
currency	currency	The currency this invoice is in.
customerid	customer_id	[Required] The ID of the associated customer.
date	date	[Required] The date of the invoice. See Date Fields .
draw	draw	The amount of any draw against retainer for this invoice. Dated by the draw_date field.
draw_date	draw_date	The date of the draw. See Date Fields .
emailed	date_emailed	Date the user emailed the invoice. For invoices created before this field existed, this field is set to 1970-01-01 to flag it as an unknown value. See Date Fields .
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
invoice_layoutid	invoice_layoutid	The ID of the associated invoice layout.

XML / SOAP	Database	Description
notes	notes	Notes associated with the invoice.
number	number	The invoice number. Must be unique
original_invoiceid	original_invoiceid	The original invoice ID for credit invoices.
paperrequest	paper_requested	Date the user requested that a paper invoice be mailed. See Date Fields .
papersend	paper_sent	Date the paper invoice was actually mailed. See Date Fields .
payment_termsid	payment_terms_id	The ID of the associated payment terms. Requires the Save Payment Terms Internal ID on Invoice Records optional feature to be enabled for the OpenAir account. The field is empty otherwise. See the help topic Save Payment Terms Internal ID on Invoice Records . Either terms or payment_termsid is required. If both are passed they must match the same payment terms.
shipping_contactid	shipping_contact_id	The shipping contact ID for this invoice.
status	status	The status of the invoice (EZ Invoice, emailed Invoice) 0 => Unknown, 1 => Not Sent, 2 => Viewed, 3 => EZ Requested, 4=> Rejected,5 => Sent, 6 => EZ Sent, 7 => Retracted
submitted	date_submitted	[Read-only] Date the invoice was submitted. See Date Fields .
tax	tax	[Read-only] The tax total for the invoice. Dated by the date field.
tax_federal	tax_federal	[Read-only] The federal tax total for the invoice. Dated by the date field.
tax_gst	tax_gst	[Read-only] The GST tax for the invoice. Dated by the date field.
tax_hst	tax_hst	[Read-only] The HST tax for the invoice. Dated by the date field.
tax_pst	tax_pst	[Read-only] The PST tax for the invoice. Dated by the date field.
tax_state	tax_state	[Read-only] The state tax total for the invoice. Dated by the date field.
terms	terms	Payment terms for this invoice. Required unless payment_termsid is passed and the Save Payment Terms Internal ID on Invoice Records optional feature to be enabled for the OpenAir account.
total	total	[Read-only] The invoice total. Dated by the date field.
updated	updated	[Read-only] Time the record was updated. See Date Fields .

Usage Guidelines

Your company's account can be configured to allow using OpenAir API to modify approved invoices. To enable the feature, contact OpenAir Customer Support.

InvoiceLayout

An invoice layout [InvoiceLayout] is a definition of how the information is presented on an invoice.

—	XML	SOAP	REST	Database table
Object	InvoiceLayout	oaInvoiceLayout	—	invoice_layout
Supported Commands	Read	read()	—	—

The InvoiceLayout object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name used for display in popups and lists.
updated	updated	[Read-only] Time the record was last modified. See Date Fields .

Issue

An issue [Issue] is a problem that has been encountered in executing project activities, or a documented event, task, activity, or action that needs to take place.

—	XML	SOAP	REST	Database table
Object	Issue	oaIssue	—	issue
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Issue object has the following standard properties:

i Note: Issue object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
attachment_id	attachment_id	If non-zero, the attachment record associated with this issue.
created	created	[Read-only] Time the record was created. See Date Fields .
customer_id	customer_id	The ID of the associated customer.
date	date	The date of the issue. See Date Fields .
date_resolution_expected	date_resolution_expected	The date the issue is expected to be resolved. See Date Fields .
date_resolution_required	date_resolution_required	The date the issue is required to be resolved. See Date Fields .

XML / SOAP	Database	Description
date_resolved	date_resolved	The date the issue was resolved. See Date Fields .
description	description	[Required] A short description of the issue, a synopsis.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
issue_category_id	issue_category_id	The ID of the associated issue category.
issue_notes	issue_notes	The description of the issue.
issue_severity_id	issue_severity_id	The ID of the associated issue severity.
issue_source_id	issue_source_id	The ID of the associated issue source.
issue_stage_id	issue_stage_id	The ID of the associated issue stage.
issue_status_id	issue_status_id	The ID of the associated issue status.
name	name	The name of the issue. Assigned automatically with the value prefix + number if not set when adding an object.
number	number	The issue number. Assigned automatically with an increment by 1 if not set when adding an object. Must be unique.
owner_id	owner_id	[Required] The ID of the associated user creating the issue.
prefix	prefix	A static alphanumeric issue number prefix.
priority	priority	The priority of the task (1 - 100).
project_id	project_id	[Required] The ID of the associated project.
project_task_id	project_task_id	The ID of the task within the associated project.
resolution_notes	resolution_notes	The description of the resolution.
updated	updated	[Read-only] Time the record was updated. See Date Fields .
user_id	user_id	The ID of the user assigned to the issue.

IssueCategory

An issue category [IssueCategory] is a classification grouping for issues.

—	XML	SOAP	REST	Database table
Object	IssueCategory	oaIssueCategory	—	issue_category
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The IssueCategory object has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this issue category is active.
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the issue category.
notes	notes	Notes associated with the issue category.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

IssueSeverity

An issue severity [IssueSeverity] indicates the degree of seriousness or priority of the problem or action item. typically the severity is a measure of the impact, the issue has on a deliverable, on the project plan, or on another key element of the project.

—	XML	SOAP	REST	Database table
Object	IssueSeverity	oaIssueSeverity	—	issue_severity
Supported Commands	Add, Read, Modify	add() , read() , modify() , upsert()	—	—

The IssueSeverity object has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this issue severity is active.
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the issue severity.
notes	notes	Notes associated with the issue severity.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

IssueSource

An issue source [IssueSource] is the person, team, or entity who reported the issue.

—	XML	SOAP	REST	Database table
Object	IssueSource	oaIssueSource	—	issue_source
Supported Commands	Add, Read, Modify	add() , read() , modify() , upsert()	—	—

The IssueSource object has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this issue source is active.
created	created	Time the record was created. See Date Fields .
id	id	Unique ID. Automatically assigned by OpenAir.
name	name	The name of the issue source.
notes	notes	Notes associated with the issue source.
updated	updated	Time the record was last updated or modified. See Date Fields .

IssueStage

An issue stage [IssueStage] is a progression step in the issue life cycle.

—	XML	SOAP	REST	Database table
Object	IssueStage	oaIssueStage	—	issue_stage
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The IssueStage object has the following properties:

XML / SOAP	Database	Description
considered_closed	considered_closed	A 1/0 field indicating whether issues in this stage are considered closed.
created	created	[Read-only] Time the record was created. See Date Fields .
default_for_new	default_for_new	A 1/0 field indicating whether this is the default stage for new issues.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the issue stage.
notes	notes	Notes associated with the issue stage.
position	position	The position of the stage.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

IssueStatus

An issue status [IssueStatus] is a measure of the health of the issue resolution.

—	XML	SOAP	REST	Database table
Object	IssueStatus	oaIssueStatus	—	issue_status
Supported Commands	Read	read()	—	—

The IssueStatus object has the following properties:

XML / SOAP	Database	Description
active	active	[Read-only] A 1/0 field indicating whether this issue status is active.
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name of the issue status.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Item

An item [Item] is a category of merchandise or service purchased, utilized or supplied in the delivery of your business activity. Items may be inventory items or expense items.

Review the [Usage Guidelines](#) for the Item object.

—	XML	SOAP	REST	Database table
Object	Item	oaItem	—	item
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

An Item object has the following properties:

 **Note:** Item object properties may also include custom fields. The object type supports the custom equal to read method and the enable_custom read attribute.

XML / SOAP	XML / SOAP	Description
active	active	A 1/0 field indicating whether this is designated as an active item. Defaults to 1 if not set when adding an item.
code	acct_code	Optional accounting system code for integration with external accounting systems.
cost	cost	The default cost per unit of measure for the item. 3 decimal places to handle items like mileage at 32.5 cents.
cost_centerid	cost_center_id	The ID of the associated cost center.
cost_is_fixed	cost_is_fixed	A 1/0 field indicating whether the user is allowed to change the cost on a receipt.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
externalid	external_id	If the record was imported from an external system you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The item name.

XML / SOAP	XML / SOAP	Description
picklist_label	—	Label as shown on form picklist.
tax_location_id	tax_location_id	The ID of the associated tax location.
taxable	taxable	A 1/0 field indicating whether this item is taxable, VAT-able, etc.
tp_comp	tp_comp	Ticket policy comparison: <ul style="list-style-type: none"> ge - greater than or equal to gt - greater than
tp_cost	tp_cost	The policy threshold amount.
tp_notes_required	tp_notes_required	Notes are required if the ticket triggers the policy.
tp_unit_or_total	tp_unit_or_total	The ticket policy is applied against: <ul style="list-style-type: none"> U - Unit price T - Total
type	type	The type of item. Add new types when type-specific information can be captured for the slip or ticket templated from this item: <ul style="list-style-type: none"> R - for regular item. M - for mileage item.
unitm	um	The unit of measure for the item, i.e., EA.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Usage Guidelines

When adding a `Item` object, the primary filter set for the authenticated user is updated automatically to allow access to the newly added customer.

You cannot delete an `Item` object if this object is referenced by an object of any of the following type. Delete any dependent objects first before you delete an `Item` object.

- Proposalblock
- Slip
- Ticket

ItemToUserLocation

Use the item-user location link [`ItemToUserLocation`] object to create many-to-many links between items, user locations and tax locations.

—	XML	SOAP	REST	Database table
Object	ItemToUserLocation	oaItemToUserLocation	—	item_to_user_location
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The `ItemToUserLocation` object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
itemid	item_id	The ID of the associated item.
tax_locationid	tax_location_id	The ID of the associated tax location.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
user_locationid	user_location_id	The location ID for this user.

Jobcode

A job code [Jobcode] is a general job category with a generic cost that can be used for estimation and billing.

—	XML	SOAP	REST	Database table
Object	Jobcode	oaJobcode	JobCode	job_code
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	See the help topic Job Codes	—

The Jobcode object has the following standard properties:

Note: Jobcode object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom` read attribute.

XML / SOAP	REST	Database	Description
active	isActive	active	A 1/0 field indicating if the job code is active.
code	accountingCode	acct_code	Optional accounting code that can be used for integration with external accounting systems.
created	created	created	[Read-only] The date the job code was created. See Date Fields
currency	currency	currency	The currency for monetary values in the job code record. Three-letter currency code.
externalid	externalId	external_id	The unique external record ID, if the record was imported from an external system.
id	id	id	[Read-only] The unique internal identifier of the job code. Assigned by OpenAir.
loaded_cost	loadedCost	loaded_cost	The loaded cost per hour for this job code.
name	name	name	[Required] The display name of the job code.
notes	notes	notes	Notes about the job code.
updated	updated	updated	[Read-only] The date the job code was last updated or modified.

XML / SOAP	REST	Database	Description
			See Date Fields
userid_fte	genericResourceId	user_id_fte	The internal ID of the generic resource used in full-time equivalent (FTE) forecasting.

JobCodeUsed

Use the job code used [JobCodeUsed] object to read information about which object types use job codes.

—	XML	SOAP	REST	Database table
Object	JobCodeUsed	oaJobCodeUsed	—	job_code_used
Supported Commands	Read	read()	—	—

A JobCodeUsed object has the following properties

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
position	position	Position in the lookup rule.
table_name	table_name	The name of the table that uses a job code.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
used_by	used_by	What it is used by. Two possible values: <ul style="list-style-type: none"> ■ 'a' for tasks ■ 's' for slips.

Leave_accrual_rule

A leave accrual rule [Leave_accrual_rule] is a rule governing the periodic increase of employees' leave entitlement and leave balance adjustments when employees take time off.

—	XML	SOAP	REST	Database table
Object	Leave_accrual_rule	oaLeave_accrual_rule	—	leave_accrual_rule
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Leave_accrual_rule object has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this is an active leave accrual rule. Defaults to 1 if not set when adding a leave accrual rule.
amount	amount	The number of hours per period.

XML / SOAP	Database	Description
cap	cap	Number of hours to cap the accrual at.
category_filter	category_filter	CSV list of categories that will trigger a draw down.
created	created	[Read-only] Time the record was created. See Date Fields .
draw_down_when	draw_down_when	Generate the draw down when: <ul style="list-style-type: none"> ■ R - When leave accrual is run. ■ A - When a timesheet is approved.
grace_days	grace_days	How many days is the grace period before accrued time is lost.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
lose_how	lose_how	How is accrued time lost: <ul style="list-style-type: none"> ■ N - Never ■ A - The users anniversary date ■ Y - End of year
name	name	[Required] The name for the leave accrual rule.
notes	notes	Notes associated with the leave accrual rule.
period	period	The period for the cap.
project_filter	project_filter	CSV list of projects that will trigger a draw down.
project_task_filter	project_task_filter	CSV list of project_tasks that will trigger a draw down.
timetype_filter	timetype_filter	CSV list of timetypes that will trigger a draw down.
timing	timing	When the accrual is applied: <ul style="list-style-type: none"> ■ S - start of the period. ■ E - end of the period.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Leave_accrual_rule_to_user

Use the leave accrual rule – user link [Leave_accrual_rule_to_user] object to associate leave accrual rules with users. A user can have multiple entries for the same rule but the date ranges must not overlap.

Review the [Usage Guidelines](#) for the Leave_accrual_rule_to_user object.

—	XML	SOAP	REST	Database table
Object	Leave_accrual_rule_to_user	oaLeave_accrual_rule_to_user	—	leave_accrual_rule_to_user
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Leave_accrual_rule_to_user object has the following properties:

XML / SOAP	dATABASE	Description
created	created	[Read-only] Time the record was created. See Date Fields .
end_date	end_date	The date the accrual rule stops applying to the user. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
leave_accrual_ruleid	leave_accrual_rule_id	[Required] The ID of the associated accrual rule.
start_date	start_date	[Required] The date the accrual rule starts applying to the user. This is required. See Date Fields .
transfer_balance_to	transfer_balance_to	ID of leave_accrual_rule_to_user record where balance should be transferred to.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Required] The ID of the associated user.

Usage Guidelines

The date range defined by start_date and end_date must not overlap with that of another Leave_accrual_rule_to_user object for the same user [userid][].

Leave_accrual_transaction

A leave accrual transaction [Leave_accrual_transaction] is either one of the periodic increases of an employee's leave entitlement or a leave balance adjustment when the employee takes time off.

—	XML	SOAP	REST	Database table
Object	Leave_accrual_transaction	oaLeave_accrual_transaction	—	leave_accrual_transaction
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Leave_accrual_transaction object has the following properties:

XML / SOAP	XML / SOAP	Description
amount	amount	The number of hours. A draw down must be a negative number. An accrual is typically a positive number but can be a negative number.
created	created	[Read-only] Time the record was created. See Date Fields .
date	date	[Required] The date of the transaction.
from_run	from_run	Indicates if this was generated from a run the leave accrual rules. See Date Fields .

XML / SOAP	XML / SOAP	Description
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
leave_accrual_ruleid	leave_accrual_rule_id	[Required] The ID of the associated accrual rule. This is a required field.
notes	notes	Notes associated with the leave accrual transaction.
taskid	task_id	The ID of the associated task if this is a draw down against a timesheet entry.
type	type	Indicates type of draw down: the type of the amount field. <ul style="list-style-type: none"> ■ D - draw down. ■ A - Accrual.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Required] The ID of the associated user.

LoadedCost

A loaded cost [LoadedCost] is the full hourly cost incurred by the company when an employee spends time on a project. Depending on your account configuration, up to three loaded cost levels can be set for each employees. Employees' loaded costs rate overrides can be set for each projects and project tasks.

Review the [Usage Guidelines](#) for the LoadedCost object.

—	XML	SOAP	REST	Database table
Object	LoadedCost	oaLoadedCost	—	loaded_cost
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The LoadedCost object has the following properties:

XML / SOAP	XML / SOAP	Description
cost	cost	The fully loaded hourly cost of the user.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	The currency of the cost field.
current	current	A 1/0 field indicating if this is the current loaded cost record.
customerid	customer_id	The ID of the associated customer.
end	end	End date for the loaded cost for historical records. See Date Fields .
externalid	external_id	If the record was imported from an external system you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.

XML / SOAP	XML / SOAP	Description
lc_level	lc_level	If multiple loaded costs are used, this holds the level of loaded cost <ul style="list-style-type: none"> 0 - primary loaded cost 1 - secondary loaded cost 2 - tertiary loaded cost
project_taskid	project_task_id	The ID if this loaded cost is associated with a specific project task. If this field is used, the project_id and customer_id must be empty.
projectid	project_id	The ID if this loaded cost is associated with a specific project.
start	start	Start date for the loaded cost for historical records. See Date Fields .
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	ID of the user.

Usage Guidelines

Review the following guidelines:

- There can be only one current cost value for a loaded cost level and a specific user. current cannot be set to 1 for more than one lc_level and userid combination.
- If current is set to 1, start and end must be empty. Otherwise, start and end are required.
- When you add or modify a [User](#), you can modify the LoadedCost associated with that [User](#). See [Updating User Loaded Costs](#).

Module

Use the module [Module] object to read whether the authenticated user can access each OpenAir application (module).

—	XML	SOAP	REST	Database table
Object	Module	oaModule	—	—
Supported Commands	Read (all)	read() (all)	—	—

The Module object has the following properties:

XML / SOAP	Database	Description
abbr	—	[Read-only] Two-letter code representing the OpenAir application (module).
enabled	—	[Read-only] A 1/0 field indicating whether the authenticated user access the OpenAir application (module).

Newsfeed

A news feed [Newsfeed] is a collection of messages with project status information.

—	XML	SOAP	REST	Database table
Object	Newsfeed	oaNewsfeed	—	newsfeed
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Newsfeed object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

NewsfeedMessage

A news feed message [NewsfeedMessage] is a message concerning a project and including project status information.

—	XML	SOAP	REST	Database table
Object	NewsfeedMessage	oaNewsfeedMessage	—	newsfeed_message
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The NewsfeedMessage object has the following properties:

XML / SOAP	Database	Description
authorid	author_id	The ID of the user who created the record. At least one of authorid and editorid is required.
content	content	The text or HTML content of the newsfeed entry. Limited to 3,000 characters. The following HTML tags are allowed (HTML Allowlist): strong, em, u, br, h3, p, ol, ul, li, a, img, span.
created	created	[Read-only] Time the record was created. See Date Fields .
editorid	editor_id	The ID of the user who last updated or modified the record. At least one of authorid and editorid is required.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
newsfeedid	newsfeed_id	[Required] The ID of the associated project status newsfeed.
tagid	tag_id	The ID of the project status tag. The tagid values correspond to predefined project status tags as follows: 0 = Empty (no status); 1 = On Track 2 = Needs Attention; 3 = Off Track; 4 = Proposed; 5 = Not Started; 6 = On Hold; 7 = Completed; 8 = Cancelled.
title	title	The title of the newsfeed entry. Limited to 125 characters. Optional.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Payment

A payment [Payment] is an amount of money received from a customer against an invoice or as an advance payment (retainer).

—	XML	SOAP	REST	Database table
Object	Payment	oaPayment	—	payment
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Payment object has the following standard properties:

 **Note:** Payment object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
bulk_paymentid	bulk_payment_id	The ID of the bulk_payment transaction if this payment is part of a bulk_payment.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
customerid	customer_id	The ID of the associated customer if this is a retainer payment.
date	date	[Required] The date of the payment. See Date Fields .
externalid	external_id	If the record was imported from an external system you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
invoice_number	invoice_number	The associated invoice number if a payment against a specific invoice.
invoiceid	invoice_id	[Required] The associated invoice ID if a payment against a specific invoice.
notes	notes	Notes associated with the payment.
total	total	The payment total. Dated by the date field.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Paymentterms

A payment terms [Paymentterms] object outlines the conditions surrounding invoice payment.

—	XML	SOAP	REST	Database table
Object	Paymentterms	oaPaymentterms	—	payment_terms
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Paymentterms object has the following standard properties:

Note: Paymentterms object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
active	active	A 1/0 field indicating where this is designated as an active shipping terms 1/0.
created	created	[Read-only] Time the record was created. See Date Fields .
default_terms	default_terms	A 1/0 field indicating whether this is the default payment terms (used for Customers, Vendors, Invoices and POs).
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The payment terms name.
notes	notes	Notes associated with the payment terms.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Paymenttype

A payment type [Paymenttype] is a method of payment that may be used for employee expenses.

Review the [Usage Guidelines](#) for the Paymenttype object.

—	XML	SOAP	REST	Database table
Object	Paymenttype	oaPaymenttype	—	payment_type
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Paymenttype object has the following standard properties:

Note: Paymenttype object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	XML / SOAP	Description
active	active	A 1/0 field specifying if the type is active.
created	created	[Read-only] Time the record was created. See Date Fields .
default_payment_type	default_payment_type	A 1/0 field indicating whether this is the default payment_type for receipts.
default_status	default_status	Default receipt status, e.g. R => Reimbursable, N => Non-reimbursable.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.

XML / SOAP	XML / SOAP	Description
name	name	[Required] The name of the payment type.
notes	notes	Notes associated with the payment type.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Usage Guidelines

You cannot delete a Paymenttype object if this object is referenced by an object of any of the following type. Delete any dependent objects first before you delete a Paymenttype object.

- [Slip](#)
- [Ticket](#)

Payrolltype

A payroll type [Payrolltype] is an attribute that can be used to categorize time entries. For example, payroll types can be used to differentiate regular and overtime work when time types are used for a different purpose.

Review the [Usage Guidelines](#) for the Payrolltype object.

—	XML	SOAP	REST	Database table
Object	Payrolltype	oaPayrolltype	—	payroll_type
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Payrolltype object has the following standard properties:

Note: Payrolltype object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
active	active	A 1/0 field specifying whether this is an active payrolltype.
created	created	[Read-only] Time the record was created. See Date Fields .
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the payroll type.
notes	notes	Notes associated with the payroll type.
picklist_label	—	Label as shown on form picklist.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Usage Guidelines

You cannot delete a Payrolltype object if this object is referenced by a [Task](#) object. Delete any dependent objects first before you delete a Payrolltype object.

PendingBooking

A pending booking [PendingBooking] is a draft booking created when searching for resources as part of the following features:

- [Automated Search Engine for Booking Creation](#) – Draft resource bookings matching the generic resource and assignment profile selected in the project task assignment.
- [Resource Demand Request](#) – Resources listed in Resources > Resource Requests > [*Select a resource request*] > Resource Request Queue Select Actioned Resources.

—	XML	SOAP	REST	Database table
Object	PendingBooking	oaPendingBooking	—	pending_booking
Supported Commands	Read	read()	—	—

The PendingBooking object has the following properties:

XML / SOAP	Database	Description
approval_status	approval_status	[Read-only] A one-character string indicating the approval status of the booking request. Possible values: <ul style="list-style-type: none"> ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected
as_percentage	as_percentage	[Read-only] A 1/0 field indicating which of the fields (hours or percentage) are , and which is derived. <ul style="list-style-type: none"> ■ 1 = percentage is and hours is derived. ■ 0 = hours in and percentage is derived.
booking_typeid	booking_type_id	[Read-only] The ID of the associated booking_type.
created	created	[Read-only] Time the record was created. See Date Fields .
customerid	customer_id	[Read-only] The ID of the associated customer.
date_approved	date_approved	[Read-only] The date the booking request was approved. See Date Fields .
date_submitted	date_submitted	[Read-only] The date the booking_request was submitted. See Date Fields .

XML / SOAP	Database	Description
dirty	dirty	[Read-only] A "2/1/0" field: 0=clean; 1=dirty; 2=in progress
enddate	enddate	[Read-only] The end date of the booking. See Date Fields .
endtime	endtime	[Read-only] End time. See Date Fields .
externalid	external_id	[Read-only] If the record was imported from an external system you store the unique external record ID here.
hours	hours	[Read-only] The number of hours booked to this project during this date range. This is either the booked hours or derived from the percentage.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
job_codeid	job_code_id	[Read-only] The ID of the associated job code.
locationid	location_id	[Read-only] The location ID for this booking.
notes	notes	[Read-only] Booking notes.
notify_owner	notify_owner	[Read-only] A 1/0 field indicating whether to send email to the requestor when the booking is modified.
ownerid	ownerid	[Read-only] The ID of the associated user creating the booking.
percentage	percentage	[Read-only] The percentage of time booked to this project during this date range. This is either the booked percentage or derived from the hours.
project_assignment_profileid	project_assignment_profile_id	[Read-only] The ID of the associated project assignment profile.
project_taskid	project_task_id	[Read-only] The ID of the task within the associated project.
projectid	project_id	[Read-only] The ID of the associated project.
repeatid	repeat_id	[Read-only] The ID of the associated repeating event.
resource_request_queue_id	resource_request_queue_id	[Read-only] The ID of the associated resource request queue.
startdate	startdate	[Read-only] The start date of the booking. See Date Fields .
starttime	starttime	[Read-only] Start time. See Date Fields .

XML / SOAP	Database	Description
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Read-only] The ID of the associated user.

Preference

A preference [Preference] is a user personal setting or company setting controlling a wide range of functionality.

—	XML	SOAP	REST	Database table
Object	Preference	oaPreference	—	preference
Supported Commands	Add, Read, Modify	add() , read() , modify() , upsert()	—	—

The preference [Preference] object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
group_name	group_name	Optional group name for the preference.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The name for the preference.
setting	setting	The preference data is stored in this field.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	If the preference is specific to a user, this will be the user ID.

Product

A product [Product] is a type of goods or services that can be purchased from external sources. Products are used to create request items, which in turn appear as line items on purchase orders.

—	XML	SOAP	REST	Database table
Object	Product	oaProduct	—	product
Supported Commands	Add, Read, Modify	add() , read() , modify() , upsert()	—	—

The Product object has the following standard properties:

Note: Product object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
active	active	A 1/0 field indicating that this is active. Defaults to 1 if not set when adding a product.
code	acct_code	Optional accounting system code for integration with external accounting systems.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	The currency this cost is quoted in.
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
manufacturer_part	manufacturer_part	The manufacturer's part number, SKU or other unique identification for this product.
manufacturerid	manufacturer_id	The manufacturer of this product.
name	name	The name for the product. This shows up on all the product pop-up windows in the application.
notes	notes	Notes associated with the product.
standard_cost	standard_cost	The current standard cost per unit of measure for the product. 3 decimal places to handle amounts like mileage at 32.5 cents.
taxable	taxable	A 1/0 field indicating whether this item is taxable.
um	um	The unit of measure for the product, that is, EA.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
vendorid	vendor_id	The preferred vendor from whom to purchase this product.
vendor_sku	vendor_sku	The preferred vendor's sku for this product.

Project

A project [Project] is a unique sequence of tasks that must be completed to reach a certain outcome to be delivered typically to a customer.

Review the [Usage Guidelines](#) for the Project object.

—	XML	SOAP	REST	Database table
Object	Project	oaProject	Project	project
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	See the help topic Projects	—

The Project object has the following standard properties:

Note: Project object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom` read attribute.

XML / SOAP	REST	Database	Description
active	isActive	active	A 1/0 field indicating if the project is active. Defaults to 1 if not set when adding a project.
attachmentid	attachments	attachment_id	The attachments associated with this project. Array (REST) of internal IDs for attachment objects.
auto_bill	isAutoBill	auto_bill	A 1/0 field indicating if the project can be billed automatically. Available only if project billing rules are not used.
auto_bill_cap	isAutoBillCap	auto_bill_cap	A 1/0 field indicating if there is a cap on the amount that can be billed automatically for the project. Available only if project billing rules are not used.
auto_bill_cap_value	autoBillCapValue	auto_bill_cap_value	The autobilling cap amount, in the currency used for the project.
auto_bill_override	isAutoBillOverriden	auto_bill_override	A 1/0 field indicating if the project-specific autobilling settings should be used instead of account-wide autobilling settings. Project-specific autobilling settings are held in the <code>auto_bill</code> table. Available only if project billing rules are not used.
az_approvalprocess	expenseAuthorizationApprovalProcess	az_approvalprocess	The internal ID of the expense authorization <i>approval process</i> for the project. Mutually exclusive with <code>expenseAuthorizationApprover</code> .
az_approver	expenseAuthorizationApprover	az_approver	The internal ID of the <i>employee</i> who approves expense authorizations for the project, if a single approver process is used. Mutually exclusive with <code>expenseAuthorizationApprovalProcess</code> . Other possible values: <ul style="list-style-type: none"> ■ -1 — the expense authorization owner's manager. ■ -2 — the manager of the expense authorization owner's manager. ■ -3 — the project owner. ■ -4 — self.
billing_code	billingCode	billing_code	The project billing code. Used in bulk invoicing.
billing_contactid	billingContactId	billing_contact_id	The internal ID of the billing <i>contact</i> , if different from the designated billing contact for the customer.
br_approvalprocess	bookingRequestApprovalProcess	br_approvalprocess	The internal ID of the booking request <i>approval process</i> for the project. Mutually exclusive with <code>bookingRequestApprover</code> .
br_approver	bookingRequestApprover	br_approver	The internal ID of the <i>employee</i> who approves booking requests for the project, if a single approver process is used. Mutually exclusive with <code>bookingApprovalProcess</code> . Other possible values:

XML / SOAP	REST	Database	Description
			<ul style="list-style-type: none"> ■ -1 — the booking request owner's manager. ■ -2 — the manager of the booking request owner's manager. ■ -3 — the project owner. ■ -4 — self.
budget	budget	budget	The budgeted revenue for the project.
—	budgetApprovalProcess	bg_approvalprocess	The internal ID of the budget <i>approval process</i> for the project. Mutually exclusive with budgetApprover.
—	budgetApprover	bg_approver	<p>The internal ID of the <i>employee</i> who approves the budget for the project, if a single approver process is used. Mutually exclusive with budgetApprovalProcess.</p> <p>Other possible values:</p> <ul style="list-style-type: none"> ■ -1 — the budget owner's manager. ■ -2 — the manager of the budget owner's manager. ■ -3 — the project owner. ■ -4 — self.
—	budgetCost	budget_cost	The budgeted cost for the project.
budget_time	budgetTime	budget_time	The budgeted amount of time for the project in hours.
category_filter	categoryFilter	category_filter	A comma-delimited list of internal IDs of categories (services) against which time can be booked for the project.
code	accountingCode	acct_code	Optional accounting code that can be used for integration with external accounting systems.
copy_approvers	copyApprovers	—	Duplicates project approvers. A 1/0 field, 1 if the project approvers should be copied.
—	copyBookings	—	A 1/0 field indicating whether to copy the bookings associated with the project or project template specified in <code>templateProjectId</code> . Used only with POST / <code>projects/from-template/</code> .
—	copyCpExcludeFilters	—	A 1/0 field indicating whether to copy the customers and projects to exclude from denominator when calculating user allocation % from the project or project template specified in <code>templateProjectId</code> . Used only with POST / <code>projects/from-template/</code> .
copy_custom_fields	copyCustomFields	—	A 1/0 field indicating whether to copy the custom field values from the project or project template specified in <code>templateProjectId</code> . Used only with POST / <code>projects/from-template/</code> .
copy_dashboard_settings	copyDashboardSettings	—	A 1/0 field indicating whether to copy the dashboard settings from the project or project template specified in <code>templateProjectId</code> . Used only with POST / <code>projects/from-template/</code> .
—	copyExpensePolicy	—	A 1/0 field indicating whether to copy the expense policy associated with the project or project template specified in <code>templateProjectId</code> . Used only with POST / <code>projects/from-template/</code> .

XML / SOAP	REST	Database	Description
copy_invoice_layout_settings	copyInvoiceLayoutSettings	—	A 1/0 field indicating whether to copy the invoice layout settings associated with the project or project template specified in templateProjectId. Used only with POST / projects/from-template/.
copy_issues	copyIssues	—	A 1/0 field indicating whether to copy the issues associated with the project or project template specified in templateProjectId. Used only with POST / projects/from-template/.
copy_loaded_cost	copyLoadedCost	—	A 1/0 field indicating whether to copy the loaded costs from the project or project template specified in templateProjectId. Used only with POST / projects/from-template/.
copy_notification_settings	copyNotificationSettings	—	A 1/0 field indicating whether to copy the notification settings from the project or project template specified in templateProjectId. Used only with POST / projects/from-template/.
—	copyProjectAssignmentProfiles	—	A 1/0 field indicating whether to copy the project assignment profiles associated with the project or project template specified in templateProjectId. Used only with POST / projects/from-template/.
copy_project_billing_auto_settings	copyProjectBillingAutoSetting	—	A 1/0 field indicating whether to copy the automated project autobilling settings from the project or project template specified in templateProjectId. Used only with POST / projects/from-template/.
copy_project_billing_rules	copyProjectBillingRules	—	A 1/0 field indicating whether to copy the project billing rules associated with the project or project template specified in templateProjectId. Used only with POST / projects/from-template/.
—	copyProjectBudgetGroups	—	A 1/0 field indicating whether to copy the project budget groups associated with the project or project template specified in templateProjectId. Used only with POST / projects/from-template/.
copy_project_pricing	copyProjectPricing	—	A 1/0 field indicating whether to copy the project pricing associated with the project or project template specified in templateProjectId. Used only with POST / projects/from-template/.
copy_revenue_recognition_auto_settings	copyRevenueRecognitionAutoSettings	—	A 1/0 field indicating whether to copy the project revenue recognition autorun settings associated with the project or project template specified in templateProjectId. Used only with POST / projects/from-template/.
copy_revenue_recognition_rules	copyRevenueRecognitionRules	—	A 1/0 field indicating whether to copy the project revenue recognition rules associated with the project or project template specified in templateProjectId. Used only with POST / projects/from-template/.
copy_revenuerecognition_auto_settings	—	—	See copy_revenue_recognition_auto_settings.
cost_centerid	costCenterId	cost_center_id	The internal ID of the cost center associated with the project.

XML / SOAP	REST	Database	Description
create_workspace	—	—	A 1/0 field indicating whether to create a workspace associated with the project automatically when creating a project. The project owner becomes the workspace owner.
created	created	created	[Read-only] The date the project was created. See Date Fields
credit_invoice_layout_id	creditInvoiceLayoutId	credit_invoice_layout_id	The internal ID of the credit memo (negative invoice) layout associated with the project.
currency	currency	currency	The currency for monetary values in the project record. Three-letter currency code.
current_dr	—	—	[Read-only] Calculated deferred revenue total.
current_wip	—	—	[Read-only] Calculated work-in-progress revenue total.
customer_name	—	customer.name	The internal ID of the customer associated with the project .
customerid	customerId	customer_id	[Required] The internal ID of the customer associated with the project.
—	dashboardFrom	dashboard_from	The item that determines if the dashboard is enabled. Possible values: <ul style="list-style-type: none"> ■ S — Project stage ■ P — Project
deleted	—	deleted	A 1/0 field indicating whether the record was deleted
exported_dr	—	—	[Read-only] Calculated deferred revenue total using exported transactions only. <div style="border: 1px solid #ccc; background-color: #fff9c4; padding: 5px; margin-top: 10px;">  Important: Only available for a specific use case. </div>
exported_wip	—	—	[Read-only] Calculated work-in-progress revenue total using exported transactions only. <div style="border: 1px solid #ccc; background-color: #fff9c4; padding: 5px; margin-top: 10px;">  Important: Only available for a specific use case. </div>
externalid	externalId	external_id	The unique external ID of the project, if the record was imported from an external system.
filtersetids	filterSets	filterset_ids	The filter sets associated with this project. Array of internal IDs for filter sets. Users who have access to any of the filter sets associated with the project have access to the project. Required if the Require a filter set selection when adding or editing project box is checked on the filter set settings form in OpenAir (Administration > Global Settings > Account > Filter Set Settings).

XML / SOAP	REST	Database	Description
finish_date	finishDate	finish_date	The finish date of the project. Defaults to the current date if not set when adding a project. See Date Fields
hierarchy_node_ids	hierarchyNodes	—	The hierarchy nodes associated with this project. Array of internal IDs for hierarchy nodes. Only hierarchy nodes associated with a hierarchy that is shown on the project property form (Show this hierarchy when editing projects box checked on the hierarchy entity form in OpenAir) are included.
id	id	id	[Read-only] The unique internal identifier of the project. Assigned by OpenAir.
invoice_layoutid	invoiceLayoutId	invoice_layout_id	The internal ID of the invoice layout associated with the project.
invoice_text	invoiceText	invoice_text	Text to display on every invoice.
—	isDashboardEnabled	enable_dashboard	A 1/0 field indicating if the project dashboard is enabled, when dashboardFrom is set to P.
is_portfolio_project	isPortfolioProject	is_portfolio_project	A 1/0 field indicating if the project is a portfolio project.
locationid	—	—	The internal ID for the location of this project (DEPRECATED).
main_contactid	contactId	—	The internal ID of the main contact associated with the project.
message	dashboardMessage	message	The dashboard message.
msp_link_type	m脾LinkType	m脾_link_type	The Microsoft Project import status. Possible values: <ul style="list-style-type: none"> ■ <i>Empty value</i> — Not imported ■ I — Imported and locked for edit ■ U — Imported and open for edit
name	name	name	[Required] The name of the project task. Depending on your company's account configuration, each project for a specific customer must have a unique name.
newsfeedid	newsFeedId	newsfeed_id	The internal ID of the news feed associated with the project.
no_dirty	—	—	A 1/0 field indicating if the project should be marked dirty when the current recalculation finishes.
notes	notes	notes	Notes about the project.
notify_assignees	notifyAssignees	notify_assignees	A 1/0 field indicating whether to send notification email to assigned employees when a task associated with the project is added, modified, or deleted.
—	notifyAssignmentCc	assignment_cc	A comma-delimited list of internal IDs of employees to be copied (Cc) into assignment notification email. Other possible listed values: <ul style="list-style-type: none"> ■ -1 — the assignee's manager.

XML / SOAP	REST	Database	Description
			<ul style="list-style-type: none"> ■ -2 — the manager of the assignee's manager. ■ -3 — the project owner. ■ -5 — the customer owner.
notify_issue_assigned_to	notifyIssueAssignedTo	notify_issue_assigned_to	A 1/0 field indicating whether to send notification email to the assigned employee when an issue is assigned to an employee.
notify_issue_closed_assigned_to	notifyIssueClosedAssignedTo	notify_issue_closed_assigned_to	A 1/0 field indicating whether to send notification email to the assigned employee when an issue is progressed to a closed issue stage.
notify_issue_closed_customer_owner	notifyIssueClosedCustomerOwner	notify_issue_closed_customer_owner	A 1/0 field indicating whether to send notification email to the customer owner when an issue is progressed to a closed issue stage.
notify_issue_closed_project_owner	notifyIssueClosedProjectOwner	notify_issue_closed_project_owner	A 1/0 field indicating whether to send notification email to the project owner when an issue is progressed to a closed issue stage.
notify_issue_created_customer_owner	notifyIssueCreatedCustomerOwner	notify_issue_created_customer_owner	A 1/0 field indicating whether to send notification email to the customer owner when an issue is created.
notify_issue_created_project_owner	notifyIssueCreatedProjectOwner	notify_issue_created_project_owner	A 1/0 field indicating whether to send notification email to the project owner when an issue is created.
notify_owner	notifyOwner	notify_owner	A 1/0 field indicating whether to send notification email to the project owner when ownership is changed.
notify_sr_submitted_project_owner	notifySrSubmittedProjectOwner	notify_sr_submitted_project_owner	A 1/0 field indicating whether to send notification email to the project owner when a schedule request is submitted for a user booked or assigned to the project.
only_owner_can_edit	onlyOwnerCanEdit	only_owner_can_edit	A 1/0 field indicating whether only the project owner can edit this project.
payroll_type_filter	payrollTypeFilter	payroll_type_filter	A comma-delimited list of internal IDs of payroll types against which time can be booked for the project.
picklist_label	—	—	Label as shown on form picklist.
pm_approver_1	projectApprover1	pm_approver_1	<p>The internal ID of the first project approver (employee) that is substituted into the approval processes.</p> <p>Other possible value:</p> <ul style="list-style-type: none"> ■ -6 — the first additional project approver.
pm_approver_2	projectApprover2	pm_approver_2	<p>The internal ID of the second project approver (employee) that is substituted into the approval processes.</p> <p>Other possible value:</p> <ul style="list-style-type: none"> ■ -7 — the second additional project approver.
pm_approver_3	projectApprover3	pm_approver_3	<p>The internal ID of the third project approver (employee) that is substituted into the approval processes.</p> <p>Other possible value:</p>

XML / SOAP	REST	Database	Description
			<ul style="list-style-type: none"> -8 — the third additional project approver.
po_approvalprocess	purchaseOrderApprovalProcess	po_approvalprocess	<p>The internal ID of the purchase order <i>approval process</i> for the project. Mutually exclusive with purchaseOrderApprover.</p> <p>The approval internal process ID of the project purchase order approval process. This field is mutually exclusive with purchaseOrderApprover</p>
po_approver	purchaseOrderApprover	po_approver	<p>The internal ID of the <i>employee</i> who approves purchase orders for the project, if a single approver process is used. Mutually exclusive with purchaseOrderApprovalProcess.</p> <p>Other possible values:</p> <ul style="list-style-type: none"> -1 — the purchase order owner's manager. -2 — the manager of the purchase order owner's manager. -3 — the project owner. -4 — self.
portfolio_projectid	portfolioProjectId	portfolio_project_id	<p>The internal ID of the portfolio <i>project</i> associated with the project, if the project is a subordinate project. The project with internal ID portfolio_projectid must be designated as a portfolio project [is_portfolio_project].</p>
pr_approvalprocess	purchaseRequestApprovalProcess	pr_approvalprocess	<p>The internal ID of the purchase request <i>approval process</i> for the project. Mutually exclusive with purchaseRequestApprover.</p>
pr_approver	purchaseRequestApprover	pr_approver	<p>The internal ID of the <i>employee</i> who approves purchase requests for the project, if a single approver process is used. Mutually exclusive with purchaseRequestApprovalProcess.</p> <p>Other possible values:</p> <ul style="list-style-type: none"> -1 — the purchase request owner's manager. -2 — the manager of the purchase request owner's manager. -3 — the project owner. -4 — self.
project_locationid	projectLocationId	project_location_id	<p>The internal ID of the project location associated with the project.</p>
project_stageid	projectStageId	project_stage_id	<p>The internal ID of the project stage associated with the project. Defaults to 1 if not set when adding a project.</p>
rate	rate	rate	<p>The hourly billing rate for the project.</p>
rate_cardid	rateCardId	rate_card_id	<p>The rate card to be used for projected billing, if not zero.</p>
rm_approvalprocess	bookingApprovalProcess	rm_approvalprocess	<p>The internal ID of the booking <i>approval process</i> for the project. Mutually exclusive with bookingApprover.</p>
rm_approver	bookingApprover	rm_approver	<p>The internal ID of the <i>employee</i> who approves bookings for the project, if a single approver process is used. Mutually exclusive with bookingApprovalProcess.</p>

XML / SOAP	REST	Database	Description
			<p>Other possible values:</p> <ul style="list-style-type: none"> ■ -1 — the booking owner's manager. ■ -2 — the manager of the booking owner's manager. ■ -3 — the project owner. ■ -4 — self.
rv_approvalprocess	revenueApprovalProcess	rv_approvalprocess	The internal ID of the revenue container <i>approval process</i> for the project. Mutually exclusive with revenueApprover.
rv_approver	revenueApprover	rv_approver	<p>The internal ID of the <i>employee</i> who approves revenue containers for the project, if a single approver process is used. Mutually exclusive with revenueApprovalProcess.</p> <p>Other possible values:</p> <ul style="list-style-type: none"> ■ -1 — the revenue container owner's manager. ■ -2 — the manager of the container owner's manager. ■ -3 — the project owner. ■ -4 — self.
sga_labor	sgaLabor	sga_labor	The allocated cost (SG and A) overhead percentage applied to labor for profitability analysis.
shipping_contact_id	shippingContactId	shipping_contact_id	The internal ID of the shipping <i>contact</i> associated with the project, if different from the designated shipping contact for the customer.
sold_to_contact_id	soldToContactId	sold_to_contact_id	The internal ID of the sold to <i>contact</i> associated with the project, if different from the designated sold to contact for the customer.
start_date	startDate	start_date	<p>The scheduled start date of the project. Defaults to the current date if not set when adding a project.</p> <p>See Date Fields</p>
sync_workspace	syncWorkspace	sync_workspace	A 1/0 field indicating whether to synchronize the project resources with membership of the workspace associated with the project.
ta_approvalprocess	timesheetApprovalProcess	ta_approvalprocess	<p>The internal ID of the timesheet <i>approval process</i> for the project. Mutually exclusive with timesheetApprover.</p> <p>The approval internal process ID of the project timesheet approval process. This field is mutually exclusive with timesheetApprover.</p>
ta_approver	timesheetApprover	ta_approver	<p>The internal ID of the <i>employee</i> who approves timesheets for the project, if a single approver process is used. Mutually exclusive with timesheetApprovalProcess.</p> <p>Other possible values:</p> <ul style="list-style-type: none"> ■ -1 — the timesheet owner's manager. ■ -2 — the manager of the timesheet owner's manager. ■ -3 — the project owner. ■ -4 — self.

XML / SOAP	REST	Database	Description
ta_include	—	—	A 1/0 field indicating whether a Timesheet filterset is applied.
—	taskBudgetCost	task_budget_cost	[Read-only] The total projected cost across all tasks in the project, if task budgeting is enabled.
—	taskBudgetRevenue	task_budget_revenue	[Read-only] The total projected billing across all tasks in the project, if task budgeting is enabled.
tax_location_name	—	tax_location.name	The name of the of the tax location associated with the project.
tax_locationid	taxLocationId	tax_location_id	The internal ID of the of the tax location associated with the project.
—	taxRateFederal	tax_rate_federal	The federal tax rate for the project.
—	taxRateState	tax_rate_state	The state/HST tax rate for the project.
tb_approvalprocess	invoiceApprovalProcess	tb_approvalprocess	The internal ID of the invoice <i>approval process</i> for the project. Mutually exclusive with <i>invoiceApprover</i> .
tb_approver	invoiceApprover	tb_approver	The internal ID of the <i>employee</i> who approves invoices for the project, if a single approver process is used. Mutually exclusive with <i>invoiceApprovalProcess</i> . Other possible values: <ul style="list-style-type: none"> ■ -1 — the invoice owner's manager. ■ -2 — the manager of the invoice owner's manager. ■ -3 — the project owner. ■ -4 — self.
te_allowance_approvalprocess	expenseallowanceApprovalProcess	te_allowance_approvalprocess	The internal ID of the allowance report <i>approval process</i> for the project. Mutually exclusive with <i>expenseallowanceApprover</i> .
te_allowance_approver	expenseallowanceApprover	te_allowance_approver	The internal ID of the <i>employee</i> who approves allowance reports for the project, if a single approver process is used. Mutually exclusive with <i>expenseallowanceApprovalProcess</i> . Other possible values: <ul style="list-style-type: none"> ■ -1 — the allowance report owner's manager. ■ -2 — the manager of the allowance report owner's manager. ■ -3 — the project owner. ■ -4 — self.
te_approvalprocess	expenseApprovalProcess	te_approvalprocess	The internal ID of the project <i>approval process</i> for the project. Mutually exclusive with <i>expenseApprover</i> .
te_approver	expenseApprover	te_approver	The internal ID of the <i>employee</i> who approves projects for the project, if a single approver process is used. Mutually exclusive with <i>expenseApprovalProcess</i> . Other possible values: <ul style="list-style-type: none"> ■ -1 — the project owner's manager. ■ -2 — the manager of the project owner's manager. ■ -3 — the project owner.

XML / SOAP	REST	Database	Description
			<ul style="list-style-type: none"> -4 — self.
te_include	—	—	A 1/0 field indicating whether an Expense Report filterset is applied.
template_project_id	templateProjectId	—	The internal ID of the project or project template to be copied. Used only with POST /projects/from-template/.
timetype_filter	timetypeFilter	timetype_filter	A comma-delimited list of internal IDs of time types against which time can be booked for the project.
updated	updated	updated	<p>[Read-only] The date the project was last updated or modified.</p> <p>See Date Fields</p>
user_filter	userFilter	user_filter	A comma-delimited list of internal IDs of employees who can edit the project if only the project owner can edit the project (only_owner_can_edit or onlyOwnerCanEdit is not 0).
userid	userId	user_id	The internal ID of the employee associated with the project — the project owner.

Usage Guidelines

Review the following guidelines:

- To create one project from another project, use the add method. Use the template_project_id field to designate the ID of the project from which project data will be copied. Indicate the settings and dependent objects you want copied by setting the property copy_<setting> to 1, where <setting> is the programmatic name for the setting or the dependent object type to be copied.
- Adding or modifying a project triggers the project recalculation process in OpenAir unless you use the no_recalc attribute (see [Add, Update and Upsert Attributes](#)), or unless your company's account is configured never to trigger the project recalculation process in OpenAir when a change is made using OpenAir API.
- You cannot delete a Project object if this object is referenced by an object of any of the following type. Delete any dependent objects first before you delete a Project object.
 - [Projecttask](#)
 - [Proposal](#)
 - [Proposalblock](#)
 - [signoff](#)
 - [Slip](#)
 - [Task](#)
 - [Ticket](#)

Projectassign

A project assignment [Projectassign] is an allocation of project work to a user at the project level.

Review the [Usage Guidelines](#) for the Projectassign object.

—	XML	SOAP	REST	Database table
Object	Projectassign	oaProjectassign	—	project_assign
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Projectassign object has the following properties:

XML / SOAP	XML / SOAP	Description
allocation	allocation	The percentage of time the associated user is allocated to this task.
created	created	[Read-only] Time the record was created. See Date Fields .
customer_id	customer_id	The ID of the associated customer.
deleted	deleted	A "1/0" field indicating if the record was deleted
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
job_codeid	job_code_id	The ID of the associated job code.
project_groupid	project_group_id	The ID of the project group if the user was assigned as part of a project group.
project_id	project_id	[Required] The ID of the project to which this user is assigned.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
user_id	user_id	[Required] The ID of the user assigned to this task.

Usage Guidelines

Review the following guidelines:

- The `project_id` and `user_id` combination must be unique. A named resource can only be assigned one time to a specific project.
- Adding or modifying a project assignment triggers the project recalculation process in OpenAir unless you use the `no_recalc` attribute (see [Add, Update and Upsert Attributes](#)), or unless your company's account is configured never to trigger the project recalculation process in OpenAir when a change is made using OpenAir API.
- Adding or modifying a project assignment automatically creates new `Projecttaskassign` objects for project tasks with property `use_project_assignment` set to 1. Modifying a project assignment automatically deletes previous `Projecttaskassign` objects for these project tasks.

ProjectAssignmentProfile

A project assignment profile [`ProjectAssignmentProfile`] is a set of skills and competencies associated with the project. Projects and project tasks can be assigned to an assignment profile instead of a named or generic resource.

—	XML	SOAP	REST	Database table
Object	ProjectAssignmentProfile	oaProjectAssignmentProfile	—	project_assignment_profile

—	XML	SOAP	REST	Database table
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The ProjectAssignmentProfile object has the following properties:

XML / SOAP	XML / SOAP	Description
created	created	[Read-only] Time the record was created. See Date Fields .
customerid	customer_id	[Read-only] The ID of the associated customer. Set automatically to the internal ID of the customer associated with the project [project_id]
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The project_assignment_profile name. Each project assignment profile for the same project must have a unique name.
projectid	project_id	[Required] Id of the project to which this project_assignment_profile is associated.
updated	updated	[Read-only] Time the record was last updated or modified in OpenAir. See Date Fields .
user_filter	user_filter	A user filter list. The project_assignment_profile only be applied to the users in this list.

Projectbillingrule

A project billing rule [Projectbillingrule] is a rule governing how and when charges are generated automatically.

—	XML	SOAP	REST	Database table
Object	Projectbillingrule	oaProjectbillingrule	—	project_billing_rule
Supported Commands	Add, Read, Modify	add(), read(), modify(), upsert()	—	—

The Projectbillingrule object has the following standard properties:

Note: Projectbillingrule object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
accounting_period_id	accounting_period_id	The ID of the associated accounting period.
acct_date	acct_date	The accounting period date to assign to the transaction. See Date Fields .
acct_date_how	acct_date_how	The accounting period date of the transaction is determined by: <ul style="list-style-type: none"> ■ N - none, clear the value ■ E - the entity (no change) ■ C - container of the entity if available (i.e, timesheet, envelope) ■ S - submitted date of the container ■ A - approved date of the container

XML / SOAP	Database	Description
		<ul style="list-style-type: none"> ■ M - set by the specified accounting date ■ P - set by the specified accounting period
active	active	A 1/0 field indicating whether this is an active billing rule.
adjust_if_capped	adjust_if_capped	If a transaction will exceed the cap, should it be adjusted to fit under the cap.
agreementid	agreementid	The ID of the associated agreement.
amount	amount	The amount for a fixed fee rule.
assigned_user	assigned_user	The user to assign to fixed fee billings.
backout_gst	backout_gst	If they are using GST/HST/PST taxes, back out the GST/HST taxes from re-billed expenses.
cap	cap	The amount to cap total billing for this rule at (in the currency of the project).
cap_by_customerpo	cap_by_customerpo	<p>A "1/0" field. If set to "1" customerpo.total or customerpo.hours is used instead of the project_billing_rule.cap or project_billing_rule.cap_hours.</p> <div style="border: 1px solid #00a0e3; padding: 10px; background-color: #e6f2ff;"> <p>Note: You can only read or modify the cap_by_customerpo field if all the following conditions are met:</p> <ul style="list-style-type: none"> ■ The project associated to the project billing rule is a portfolio project. ■ The project associated to the project billing rule is associated with at least one customer PO. ■ The billing rule is associated with a customer PO. ■ The billing rule type is one of the following: Expense item, Purchase item, Time. ■ The following features are enabled for your account: <ul style="list-style-type: none"> □ Portfolio Projects and Subordinate Projects. □ Single Billing Cap across Multiple Subprojects Within a Portfolio Project. <p>The cap_by_customerpo attribute is listed in the Scripting Center SOAP Explorer if these features are enabled for your account.</p> </div>
cap_hours	cap_hours	The number of hours to cap the billing at for a time billing rule.
category_1id	category_1_id	The ID of the associated category_1. Mutually exclusive with project_task_id.
category_2id	category_2_id	The ID of the associated category_2. Mutually exclusive with project_task_id.
category_3id	category_3_id	The ID of the associated category_3. Mutually exclusive with project_task_id.

XML / SOAP	Database	Description
category_4id	category_4_id	The ID of the associated category_4. Mutually exclusive with project_task_id.
category_5id	category_5_id	The ID of the associated category_5. Mutually exclusive with project_task_id.
category_filter	category_filter	CSV list of categories to limit the rule to.
category_when	category_when	When the category be applied: <ul style="list-style-type: none"> ■ N - Use the selected category if the time entry does not have a category. ■ A - Always use the selected category.
categoryid	category_id	The ID of the category to assign to the transaction if it doesn't have a category.
cost_center_id	cost_center_id	The ID of the associated cost center.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
customerid	customer_id	The ID of the associated customer.
customerpo	customerpo_id	The ID of the associated customerpo.
daily_cap_hours	daily_cap_hours	The number of hours to cap the period billing per user at.
daily_cap_is_per_user	daily_cap_is_per_user	Is the daily cap on a per user basis.
daily_cap_period	daily_cap_period	Period for the cap: <ul style="list-style-type: none"> ■ D - day ■ W - week ■ M - month ■ Q - quarter ■ Y - year
daily_rate_multiplier	daily_rate_multiplier	Optional daily multiplier to adjust the time billing rate by. This is a comma delimited list of the multipliers for the days of the week starting with Monday and ending with Sunday.
daily_roll_to_next	daily_roll_to_next	If the period cap is hit move the remainder to the next rule.
description	description	The rule description.
end_date	end_date	End date of the rule. See Date Fields .
end_milestone	end_milestone	The ID of the ending milestone (project_task).
exclude_archived_ts	exclude_archived_ts	Exclude time from archive timesheets in time billing rules.
exclude_non_billable	exclude_non_billable	Exclude non-billable expenses.
exclude_non_billable_task	exclude_non_billable_task	Exclude non-billable tasks.
exclude_non_reimbursable	exclude_non_reimbursable	Exclude non-reimbursable expenses.
extra_data	extra_data	The extra_data field holds additional data associated with the project billing rule.

XML / SOAP	Database	Description
		<p>Information about Billing rule filters set to use custom fields to limit the billing rule to specific employees will be stored in the extra_data field. Review the following notes before setting Employee custom fields as billing rule filters:</p> <ul style="list-style-type: none"> Time, Expense and Purchase billing rules support the use of Employee custom fields as billing rule filters. The following custom field types are supported: Checkbox, Dropdown, Dropdown and text, Pick list, and Radio buttons. The extra_data field stores custom field filters as a hash. The hash always needs to be formatted as follows: <p>C# example (SOAP API):</p> <pre>1 Projectbillingrule.extra_data = "\$->{extra_data} = {'user' => { 'custom_field_id_1' => 'custom_field_id_1_value_1,custom_field_id_1_val ue_2','custom_field_id_2' => 'custom_field_id_2_value_1,cus tom_field_id_2_value_2'}}";</pre> <p>XML example (XML API):</p> <pre>1 <extra_data>\$->{extra_data} = {'user' => { 'custom_field_id_1' => 'custom_field_id_1_value_1,custom_field_id_1_value_2','custom_field_id_2' => 'custom_field_id_2_value_1,custom_field_id_2_value_2'}};</extra_data></pre> <div style="border: 1px solid #00a0e3; padding: 10px; margin-top: 10px;"> <p> Note: Review the following guidelines:</p> <ul style="list-style-type: none"> The syntax for a field-value pair can be 'custom_field_id'=>'value' or 'custom_field_id', 'value' For Checkbox Custom fields, use the value '%0A_EMPTYSTRING%' for unchecked / False Use a comma separated list of values if a custom field can have multiple values 'value_1,value_2,value_3' Make sure all values are correct. Any invalid value set for the custom field billing rule filter will have an adverse impact on billing transaction creation. Always the entire hash for custom field billing rule filters even if you are only changing one value. If a field-value pair is omitted from the hash, the corresponding filter will be removed. </div>
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
item_filter	item_filter	CSV list of items to limit the rule to.
job_code_filter	job_code_filter	CSV list of filters to limit the rule to.
markup	markup	The amount of markup in percent or monetary amount as designated by markup_type field.

XML / SOAP	Database	Description
markup_category	markup_category	The ID of the category a markup on expense receipts should be assigned to.
markup_type	markup_type	A field indicating the type of expense markup: <ul style="list-style-type: none"> ■ P - percentage of the cost. ■ S - specific amount.
name	name	Name of this project billing rule.
notes	notes	Notes associated with this project billing rule.
percent	percent	The percentage value for a fixed fee percent trigger.
percent_how	percent_how	If the fixed fee is triggered by a percent complete, this holds how it is triggered: <ul style="list-style-type: none"> ■ A - % complete of planned hours for the project ■ B - % complete of planned hours for a phase or task (the task ID is held in the start_milestone field)
position	position	The position of the rule (0,1,2 etc.). Rules are evaluated in order and evaluation stops once a rule is satisfied.
product_filter	product_filter	CSV list of products to limit the rule to.
project_task_filter	project_task_filter	CSV list of tasks to limit the rule to.
project_task_id	project_task_id	The ID of the associated task. <ul style="list-style-type: none"> ■ project_task_id must be for a top level phase in the project schedule. ■ The account must be configured to require either a Service or Service 1-5 line on top level phases. ■ The billing rule type must be 'F' (fixed fee billing rule).
projectid	project_id	The ID of the associated project.
rate_cardid	rate_card_id	The ID of the associated rate card if using rate cards.
rate_from	rate_from	Where we get the rate from: <ul style="list-style-type: none"> ■ U - Users ■ R - Rate cards ■ C - Category
rate_multiplier	rate_multiplier	Optional multiplier to adjust the time billing rate by.
repeatid	repeatid	The ID of the associated repeating event.
round_rules	round_rules	Rules for rounding time.
slip_stageid	slip_stage_id	The ID of the slip stage to assign to the transaction.
start_date	start_date	Start date of the rule. See Date Fields .
start_milestone	start_milestone	The ID of the starting milestone (project_task).
stop_if_capped	stop_if_capped	If a transaction is not billed because it exceeds the cap, should the billing stop for this transaction.
ticket_maximums	ticket_maximums	Holds data on ticket maximums per expense type.
timetype_filter	timetype_filter	CSV list of timetypes to limit the rule to.

XML / SOAP	Database	Description
type	type	[Required] The type of the billing rule: <ul style="list-style-type: none"> ■ T - time billing rule. ■ E - expense billing rule. ■ F - fixed fee billing rule. ■ P - purchase billing rule.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
user_filter	user_filter	CSV list of users to limit the rule to.

Projectbillingtransaction

A project billing transaction [Projectbillingtransaction] is a transaction created automatically based on a project billing rule.

Review the [Usage Guidelines](#) for the Projectbillingtransaction object.

—	XML	SOAP	REST	Database table
Object	Projectbillingtransaction	oaProjectbillingtransaction	—	project_billing_transaction
Supported Commands	Add, Read, Modify, Delete See Usage Guidelines	add() , read() , modify() , upsert() , delete() See Usage Guidelines	—	—

The Projectbillingtransaction object has the following properties:

XML / SOAP	Database	Description
agreementid	agreement_id	ID of the associated agreement.
categoryid	category_id	The ID of the associated category.
cost	cost	The cost per unit of measure for an E type. The fixed price for an F type. Dated by the date field.
cost_centerid	cost_center_id	The ID of the associated cost center.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	The 3-letter currency code for the project billing transaction currency. Defaults to the currency field in the associated project billing rule. Can be set to any currency specified in Administration > Global Settings > Organization > Currencies > Multi-currency (if multi-currency is enabled on your account).
customerid	customer_id	The ID of the associated customer.
customerpo	customerpo_id	ID of the associated customerpo.
date	date	The date of the transaction. See Date Fields .

XML / SOAP	Database	Description
description	description	Description associated with billing rule transaction.
fulfillmentid	fulfillment_id	The ID of the associated fulfillment record.
hour	hour	The number of hours for a T type.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
itemid	item_id	The ID of the associated item.
job_codeid	job_code_id	The ID of the associated job code.
minute	minute	The number of minutes for a T type.
notes	notes	Notes associated with this project billing rule transaction.
payroll_typeid	payroll_type_id	The ID of the associated payroll type.
project_billing_ruleid	project_billing_rule_id	The ID of the associated project billing rule.
project_taskid	project_task_id	The ID of the associated project task.
projectid	project_id	The ID of the associated project.
quantity	quantity	The quantity for an E or P type.
rate	rate	The hourly rate for a T type. Dated by the date field.
slip_stage_id	slip_stage_id	The ID of the slip stage.
slipid	slip_id	The ID of slip that was created.
taskid	task_id	The ID of the associated task.
ticketid	ticket_id	The ID of the associated ticket.
timetypeid	timetype_id	The ID of the associated time type.
total	total	[Read-only] The total monetary value. Dated by the date field.
type	type	The type of the transaction. Matches the type field in project_billing_rule.
um	um	The unit of measure for an E or P type.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	The ID of the associated user.

Usage Guidelines

The Projectbillingtransaction object supports only the [Read](#) (XML API) and [read\(\)](#) (SOAP API) commands by default. Your company's OpenAir account can be configured to allow add, modify, upsert and delete operations for project billing transactions using OpenAir API.

ProjectBudgetGroup

A project budget group [ProjectBudgetGroup] is a complete project budget as defined in Projects > [Select a project] > Financials > Project Budget > Properties using the OpenAir UI.

Review the [Usage Guidelines](#) for the ProjectBudgetGroup object.

—	XML	SOAP	REST	Database table
Object	ProjectBudgetGroup	oaProjectBudgetGroup	—	project_budget_group
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The ProjectBudgetGroup object has the following properties:

XML / SOAP	Database	Description
approval_status	approval_status	A one-character string indicating the approval status of the project budget group. Possible values: <ul style="list-style-type: none"> ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected ■ X - Archived
approved	date_approved	The date the project budget group was approved. See Date Fields .
archived	date_archived	The date the project budget group was archived. See Date Fields .
budget_by	budget_by	Sets the "Budget by" option: <ul style="list-style-type: none"> ■ 1 — budget by project ■ 2 — budget by phase ■ 3 — budget by task
calculated_total	calculated_total	[Read-only] The total calculated from project budget transactions. Date set by the date" field (obsolete attribute)."
cf_opt	cf_opt	Optimistic contingency factor for this project budget. Used if your account is configured to allow for a simplified contingency factor on project budgets. Defaults to 80% if not set when adding a ProjectBudgetGroup object.
cf_pes	cf_pes	Pessimistic contingency factor for this project budget. Used if your account is configured to allow for a simplified contingency factor on project budgets. Defaults to 120% if not set when adding a ProjectBudgetGroup object.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.

XML / SOAP	Database	Description
customerid	customerid	[Required] The ID of the associated customer.
date	date	The date of the budget entry. See Date Fields .
eac	—	[Read-only] Calculated estimate at completion (EAC) total.
eac_expense	—	[Read-only] Calculated estimate at completion (EAC) – expense subtotal.
eac_labor	—	[Read-only] Calculated estimate at completion (EAC) – labor subtotal.
eac_purchase	—	[Read-only] Calculated estimate at completion (EAC) – purchase subtotal.
etc	—	[Read-only] Calculated estimate to complete (ETC) total.
etc_expense	—	[Read-only] Calculated estimate to complete (ETC) – expense subtotal.
etc_labor	—	[Read-only] Calculated estimate to complete (ETC) – labor subtotal.
etc_purchase	—	[Read-only] Calculated estimate to complete (ETC) – purchase subtotal.
externalid	externalid	If the record was imported from an external system you store the unique external record ID here.
funding_total	funding_total	[Read-only] Total calculated from project funding documents. Date set by the date" field."
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
internal_total	internal_total	Manually entered total. Date set by the date" field."
itd	—	[Read-only] Calculated inception to date (ITD) total.
itd_expense	—	[Read-only] Calculated inception to date (ITD) – expense subtotal.
itd_labor	—	[Read-only] Calculated inception to date (ITD) – labor subtotal.
itd_purchase	—	[Read-only] Calculated inception to date (ITD) – purchase subtotal.
labor_subcategory	labor_subcategory	Labor subcategory: <ul style="list-style-type: none"> ■ 0 - category ■ 1 - job code
name	name	The name of the project budget group.
notes	notes	Notes associated with this project budget.
parentid	parentid	The project budget group ID of this budget's immediate ancestor. If zero or null, this is a top-level project budget group.

XML / SOAP	Database	Description
profitability	profitability	The profitability of this project budget group.
projectid	projectid	[Read-only] The ID of the associated project.
setting	setting	Miscellaneous settings are stored in this field as a serialized hash.
submitted	date_submitted	The date the project budget group was submitted. See Date Fields .
total	total	The total value of the budget entry. Date set by the date" field."
total_calculated_billing	total_calculated_billing	[Read-only] Billing total calculated from project budget transactions. Date set by the date" field."
total_calculated_cost	total_calculated_cost	[Read-only] Cost total calculated from project budget transactions. Date set by the date" field."
total_expected_billing	total_expected_billing	[Read-only] Billing budget expected total. Date set by the date" field."
total_expected_cost	total_expected_cost	[Read-only] Cost total calculated from project budget transactions. Date set by the date" field."
total_from_funding	total_from_funding	A 1/0 field indicating whether to use the total from project funding documents.
unassigned_task	unassigned_task	A 1/0 field indicating whether it is possible to create budget entries not connected to any particular task.
updated	updated	[Read-only] Time the record was last modified. See Date Fields .
userid	userid	The user ID of the budget owner.
version	version	Version of the project budget group.

Usage Guidelines

Review the following guidelines:

- When adding a budget, OpenAir checks the validity of the customer and the project.
- When modifying an existing budget, you cannot change the project or the customer for the budget.
- To delete a budget, you must have edit access. Approved or archived budgets cannot be deleted.
- The OpenAir API does not support project budget approvals.

ProjectBudgetRule

A project budget rule [ProjectBudgetRule] defines a line in the project budget grid.

Review the [Usage Guidelines](#) for the ProjectBudgetRule object.

—	XML	SOAP	REST	Database table
Object	ProjectBudgetRule	oaProjectBudgetRule	—	project_budget_rule

—	XML	SOAP	REST	Database table
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The ProjectBudgetRule object has the following properties:

XML / SOAP	Database	Description
category	category	The main category for this budget line: <ul style="list-style-type: none"> 1 — labor 2 — expense item 3 — purchase order When setting category to 1, the property labor_subcategory for the associated project budget group project_budget_groupid must be set.
categoryid	category_id	The ID of the associated category.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
customerid	customer_id	[Read-only] Copied from project budget group. The ID of the associated customer.
date	date	The date of the budget rule. See Date Fields .
end_date	end_date	End date of the period. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
imported	imported	A 0/1 field which indicates whether the rule was imported from project task assignments or bookings (related only to the labor category).
itemid	item_id	The ID of the associated item.
job_codeid	job_code_id	The ID of the associated job code.
notes	notes	Notes associated with this project budget line.
period	period	The period of the total: <ul style="list-style-type: none"> D — daily W — weekly M — monthly T — total (for example, the sum entered in the web application is only for one date, and not periodically recurring)
productid	product_id	The ID of the associated product.
profitability	profitability	The profitability of this project budget rule.
project_budget_groupid	project_budget_group_id	[Required] The ID of the associated project_budget_group. Cannot be modified.

XML / SOAP	Database	Description
project_taskid	project_task_id	The ID of the associated project task.
projectid	project_id	[Read-only] Copied from project budget group. The ID of the associated project.
quantity	quantity	The quantity for this project budget rule.
quantity_best	quantity_best	The best-case quantity estimate for this project budget rule.
quantity_most_likely	quantity_most_likely	The most likely quantity estimate for this project budget rule.
quantity_worst	quantity_worst	The worst-case quantity estimate for this project budget rule.
rate	rate	The rate of this project budget rule.
start_date	start_date	Start date of the period. See Date Fields .
total	total	The total for this project budget rule. Date set by the date" attribute."
total_best	total_best	The best-case estimate for this project budget rule. Date set by the date" attribute."
total_most_likely	total_most_likely	The most likely estimate for this project budget rule. Date set by the date" attribute."
total_worst	total_worst	The worst-case estimate for this project budget rule. Date set by the date" attribute."
updated	updated	[Read-only] Time the record was last modified. See Date Fields .

Usage Guidelines

Review the following guidelines:

- When adding a new project budget rule, OpenAir checks the validity of the project budget group ID, and returns error 945 if invalid (see [Error Codes](#)). The project and customer fields are sourced from the project budget group.
- When modifying an existing project budget rule, you cannot change the project, customer, or budget group ID fields. If you change any of the following fields in a project budget rule, these fields are copied to all related project budget transactions: category, categoryid, itemid, job_codeid, productid, project_taskid.
- To delete a project budget rule, you must have edit access to the project budget group, and the project budget must not be approved or archived.
- The OpenAir API does not support project budget approvals.

ProjectBudgetTransaction

A project budget transaction [ProjectBudgetTransaction] defines a transaction in one project budget grid line.

Review the [Usage Guidelines](#) for the ProjectBudgetTransaction object.

—	XML	SOAP	REST	Database table
Object	ProjectBudgetTransaction	oaProjectBudgetTransaction	—	project_budget_transaction
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The ProjectBudgetTransaction object has the following properties:

XML / SOAP	Database	Description
category	category	[Read-only] Copied from project budget rule. The main category for this budget transaction: <ul style="list-style-type: none"> ■ 1 — labor ■ 2 — expense item ■ 3 — purchase order When setting category to 1, the property labor_subcategory for the associated project budget group project_budget_groupid must be set.
categoryid	category_id	[Read-only] Copied from project budget rule. The ID of the associated category.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
customerid	customer_id	[Read-only] Copied from project budget group. The ID of the associated customer.
date	date	The date of the project budget transaction. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
itemid	item_id	[Read-only] Copied from project budget rule. The ID of the associated item.
job_codeid	job_code_id	[Read-only] Copied from project budget rule. The ID of the associated job code.
productid	product_id	[Read-only] Copied from project budget rule. The ID of the associated product.
project_budget_groupid	project_budget_group_id	The ID of the associated project budget group.
project_budget_ruleid	project_budget_rule_id	The ID of the associated project budget rule.
project_taskid	project_task_id	[Read-only] Copied from project budget rule. The ID of the associated project task.
projectid	project_id	[Read-only] Copied from project budget group. The ID of the associated project.
quantity	quantity	Quantity for this project budget transaction.
quantity_best	quantity_best	The best-case quantity estimate for this project budget transaction.

XML / SOAP	Database	Description
quantity_most_likely	quantity_most_likely	The most likely quantity estimate for this project budget transaction.
quantity_worst	quantity_worst	The worst-case quantity estimate for this project budget transaction.
total	total	The total for this budget transaction. Date set by the date" attribute."
total_best	total_best	The best-case estimate for this project budget transaction. Date set by the date" attribute."
total_most_likely	total_most_likely	The most likely estimate for this project budget transaction. Date set by the date" attribute."
total_worst	total_worst	The worst-case estimate for this project budget transaction. Dated by the date field.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Usage Guidelines

Review the following guidelines:

- When adding a new project budget rule, OpenAir checks the validity of the the project budget rule, and returns error 946 if invalid (see [Error Codes](#)). The following fields are sourced from the project budget rule: category, categoryid, customerid, itemid, job_codeid, productid, projectid, project_budget_groupid, project_taskid
- When modifying an existing project budget rule, you cannot change the project_budget_ruleid or any of the fields sourced from the project budget rule.
- To delete a project budget transaction, you must have edit access to the project budget group, and the project budget must not be approved or archived.
- The OpenAir API does not support project budget approvals.

Projectgroup

A project assignment group [Projectgroup] is a collection of users who can be assigned to a project task as a group, instead of individually.

—	XML	SOAP	REST	Database table
Object	Projectgroup	oaProjectgroup	—	project_group
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Projectgroup object has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this is active.
assigned_users	—	A comma-separated list of internal ID for the users included in the group

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name for the project group. Must be unique.
notes	notes	Notes associated with the project group.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Projectlocation

A project location [Projectlocation] is a geographical classification information for projects.

—	XML	SOAP	REST	Database table
Object	Projectlocation	oaProjectlocation	—	project_location
Supported Commands	Read	read()	—	—

The Projectlocation object has the following properties:

XML / SOAP	Database	Description
active	active	[Read-only] A 1/0 field indicating whether this is active.
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name for the project location.
notes	notes	[Read-only] Notes associated with the project location.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

ProjectPricing

A project pricing [ProjectPricing] object holds information to generate estimates for specific projects based on different staffing, rate, and time required scenarios, and to view the key financial analysis for those scenarios.

—	XML	SOAP	REST	Database table
Object	ProjectPricing	oaProjectPricing	—	project_pricing
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The ProjectPricing object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .

XML / SOAP	Database	Description
customerid	customer_id	[Required] The id of the associated customer. Must be the customer associated with the project. Cannot be modified.
discount_rate_cardid	discount_rate_card_id	Internal ID of the discount rate card to use.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
projectid	project_id	[Required] The id of the associated project. There can only be one project pricing for a specific project. Cannot be modified.
standard_rate_cardid	standard_rate_card_id	Internal ID of the standard rate card to use. Cannot be modified.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

ProjectStage

An project stage [Projectstage] is a progression step in the project life cycle.

—	XML	SOAP	REST	Database table
Object	Projectstage	oaProjectstage	—	project_stage
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Projectstage object has the following standard properties:

 **Note:** Projectstage object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
enable_analysis	enable_analysis	Is financial analysis enabled at this stage.
enable_billing	enable_billing	Is the project billing tab enabled at this stage.
enable_phase_and_task	enable_phase_and_task	Are phases and tasks enabled at this stage.
enable_pricing	enable_pricing	Is project pricing enabled at this stage. Off by default.
enable_project_assignments	enable_project_assignments	Are project level assignments enabled at this stage.
enable_recognition	enable_recognition	Is the recognition tab enabled at this stage.

XML / SOAP	Database	Description
enable_team	enable_team	A 1/0 field indicating if this should be the default stage for new projects.
enable_utilization	enable_utilization	Is the utilization enabled at this stage.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the project stage.
notes	notes	Notes associated with the project stage.
picklist_label	—	Label as shown on form picklist.
position	position	The position of the stage.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Projecttask

A Projecttask object can be:

- A project task or a project phase is a work package to be completed as part of a project. It includes information about the work to be done, when, by whom, and at what cost.
- A project milestone is a reference point in your project life cycle that marks a significant event or goal to be completed as part of a project.

Review the [Usage Guidelines](#) for the Projecttask object.

—	XML	SOAP	REST	Database table
Object	Projecttask	oaProjecttask	ProjectMilestone ProjectPhase ProjectTask	project_task
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	See the help topics Project Milestones , Project Phases , and Project Tasks	—

The Projecttask object has the following standard properties:

Note: Projecttask object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML/SOAP	REST	Database	Description
all_can_assign	useAllCanAssign (ProjectMilestone and ProjectTask)	all_can_assign	Is everyone able to assign time/ expenses to this task.
assign_user_names	userAssignments (ProjectTask only)	—	A comma separated list of employee nicknames assigned to the task. Must be empty if <code>is_a_phase</code> is set to 1.

XML/SOAP	REST	Database	Description
			You can also use project task assignment objects with XML API and SOAP API. See Projecttaskassign .
—	assignmentCc (ProjectTask only)	—	A comma-delimited list of internal IDs of employees to be copied (Cc) into assignment notification email. Other possible listed values: <ul style="list-style-type: none"> ■ -1 — the assignee's manager. ■ -2 — the manager of the assignee's manager. ■ -3 — the project owner. ■ -5 — the customer owner.
—	assignmentCcEmails (ProjectTask only)	—	A comma-delimited lists of additional recipient email addresses to be copied (Cc) into assignment notification email. Used for recipients who are not OpenAir users.
—	attachments	—	The attachments associated with this task. Array of internal IDs for attachment objects. Array of {"id": <integerValue>} objects.
calculated_finishes	calculatedFinishes (ProjectPhase and ProjectTask only)	calculated_finishes	[Read-only] Calculated finish date. See Date Fields
calculated_starts	calculatedStarts	calculated_starts	[Read-only] Calculated start date of the milestone, phase or task. See Date Fields
classification	—	classification	Calculated, read-only classification of the project task with the following values: <ul style="list-style-type: none"> ■ 'P' for Phase ■ 'T' for Task ■ 'M' for Milestone
closed	isClosed	closed	A 1/0 field indicating if this is closed task. Additional time can not be booked against closed task.
cost_centerid	costCenterId (ProjectTask only)	cost_center_id	The internal ID of the associated cost center
created	created	created	[Read-only] The date the milestone, phase or task was created. See Date Fields
currency	currency	currency	The currency for monetary values in the milestone, phase or task record. Three-letter currency code. Should be the same as the project currency.
customer_name	—	customer .name	[Read-only] The name of the associated customer.

XML/SOAP	REST	Database	Description
customerid	customerId	customer_id	The internal ID of the associated customer.
default_category	defaultCategory (ProjectMilestone and ProjectTask) defaultCategoryOnPhase (ProjectPhase)	default_category	The category to assign to a timesheet entry assigned to this task. The feature has to be enabled for this assignment to work.
default_category_<N> where <N> is an integer from 1 to 5	defaultCategory<N> (ProjectMilestone and ProjectTask) defaultCategoryOnPhase<N> (ProjectPhase) where <N> is an integer from 1 to 5	default_category_<N> where <N> is an integer from 1 to 5	A feature, if enabled, would assign this default_category_<N> to the category_<N> for many transactions that have a category_<N>_id and project_task_id by searching the project_task and phase work breakdown structure for the first default_category_<N> defined
deleted	—	deleted	A "1/0" field indicating if the record was deleted
early_finish	—	early_finish	The earliest this task can finish, measured in days from project start. This field is recalculated whenever any project dates, user assignments, or user schedules are changed. See Date Fields .
early_start	—	early_start	The earliest this task can start, measured in days from project start. This field is recalculated whenever any project dates, user assignments, or user schedules are changed. See Date Fields .
estimated_hours	—	estimated_hours	If the use task estimating feature is turned on, this field will have the estimated total time the task will take to complete. If zero, no estimating has occurred so the estimate is the same as the plan.
externalid	externalId	externalid	The unique external ID of the milestone, phase or task, if the record was imported from an external system.
fnlt_date	fnltDate (ProjectTask only)	fnlt_date	The finish no later than date of the task. The task must be finished by this date. See Date Fields .
—	hoursRemainingEstimates (ProjectTask only)	—	—
id	id	id	[Read-only] The unique internal identifier of the milestone, phase or task. Assigned by OpenAir.
id_number	idNumber	id_number	User-defined task ID. Value assigned automatically if not set when adding a project task. Must be unique for each Projecttask object associated with the same project.
is_a_phase	—	is_a_phase	A 1/0 field indicating if any other project_tasks have us as a parent.

XML/SOAP	REST	Database	Description
—	isReadyForRecognition (ProjectPhase and ProjectTask only)	—	Flag 1/0 indicating if phase is ready for recognition
—	isSignOffRequired (ProjectTask only)	—	—
manual_task_budget	useManualTaskBudget (ProjectTask only)	manual_task_budget	If set to 1 then the task budget is manually entered rather than calculated by OpenAir.
name	name	name	[Required] Short description of this task.
non_billable	isNonBillable (ProjectMilestone and ProjectTask only)	non_billable	If set to 1, this is not billable. This is only applicable for project billing rules.
notes	notes	notes	Notes about the project task.
parentid	parentId	parentid	The internal ID of the immediate ancestor. If zero or null, this is a project-level (top-level) task or phase.
percent_complete	percentComplete (ProjectPhase and ProjectTask only)	percent_complete	This field is an estimate of the percentage of planned time which has been completed. It has no relation to the real time spent on a task. (A 5-hour task could consume 50 hours of work but still be only 25% complete.)
planned_hours	plannedHours (ProjectPhase and ProjectTask only)	planned_hours	Total number of hours the task is estimated to require. This is the total amount of time the task should take if worked on continuously by one person with no interruptions. A task with zero planned hours is also known as a milestone.
predecessors	predecessors	predecessors	Comma delimited list of task IDs which must complete before this task can start.
predecessors_lag	predecessorsLag	predecessors_lag	Comma delimited list for task ID:days of lag time for predecessors. Only populated if there is a lag time.
predecessors_type	predecessorsType	predecessors_type	Comma delimited list of task ID:relationship type for predecessors. Only populated if the relationship type is not finish-to-start.
priority	priority	priority	The priority of the task (1 - 9).
project_name	—	project.name	[Read-only] The name of the associated project.
projectid	projectId	projectid	[Required] The ID of the associated project.
projecttask_typeid	projectTaskTypeId (ProjectMilestone and ProjectTask only)	projecttask_typeid	The ID of the associated project task type. Not for phases.

XML/SOAP	REST	Database	Description
seq	seq	seq	The sequence number of this task.
starts	startDate (ProjectMilestone and ProjectTask only)	start_date	Optional scheduled starting date of this task. Overrides computed date Start_date. See Date Fields .
task_budget_cost	taskBudgetCost (ProjectTask only)	task_budget_cost	If task budgeting is enabled, this is the total cost of the task.
task_budget_revenue	taskBudgetRevenue (ProjectTask only)	task_budget_revenue	If task budgeting is enabled this is the total projected billing for the task.
—	taskSplits (ProjectTask only)	—	—
timetype_filter	timetypeFilter (ProjectTask only)	timetype_filter	A timetype filter. This will hold a list of the timetypes that are allowed to book time to this task.
updated	updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
use_project_assignment	isUsingProjectAssignment (ProjectTask only)	use_project_assignment	Flag set to 1 if they are using the project level user assignment. Must not be set to 1 if is_a_phase is set to 1.

Usage Guidelines

Review the following guidelines:

- When reading Projecttask objects with the all method, the limit attribute applies to referenced Project objects instead of Projecttask objects unless the filter or deleted attributes are used. It limits the number of referenced Project objects and not the number of Projecttask objects returned.

The following code sample reads all Projecttask objects associated with the first thousand Project objects.

```
1 | <Read type="Projecttask" method="all" limit="0,1000"/>
```

To limit the number of Projecttask objects, you can use a different read method or set the filter attribute to all.

- Adding or modifying a project task triggers the project recalculation process in OpenAir unless you use the no_recalc attribute (see [Add, Update and Upsert Attributes](#)), or unless your company's account is configured never to trigger the project recalculation process in OpenAir when a change is made using OpenAir API.
- When using OpenAir XML API, the Projecttask object element accepts the attribute index. The index attribute value can be any one of the column names in the [project_task](#) table. This lets you reference Projecttask objects by the designated index property instead of referencing Projecttask objects by internal ID. If you set the index attribute to id_number, for example, you can reference predecessor tasks [predecessors] by these user-defined task numbers instead of internal IDs. OpenAir API looks up and store the internal IDs to the database.

The following code sample looks up the Projecttask objects associated with the same project with id_number set to 123 and abc and stores their internal IDs in the predecessors property for the Projecttask with internal ID 427.

```
1 | <Modify type="Projecttask">
2 |   <Projecttask index="id_number">
```

```

3     <id>427</id>
4     <predecessors>123,abc</predecessors>
5   </Projecttask>
6 </Modify>

```

- You cannot delete a Projecttask object if this object is referenced by an object of any of the following type. Delete any dependent objects first before you delete a Projecttask object.
 - ProjecttaskEstimate
 - Schedulerequest
 - Schedulerequest_item
 - Slip
 - Task
 - Ticket

Note: You can delete a Projecttask object that is referenced by other Projecttask objects as predecessors.

Projecttaskassign

A project task assignment [Projecttaskassign] is an allocation of work to a user as part of a project task.

Review the [Usage Guidelines](#) for the Projecttaskassign object.

—	XML	SOAP	REST	Database table
Object	Projecttaskassign	oaProjecttaskassign	—	project_task_assign
Supported Commands	Add, Read, Modify, ModifyOnCondition, Delete	add(), read(), modify(), upsert(), delete()	—	—

The Projecttaskassign object has the following standard properties:

Note: Projecttaskassign object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
allocation	allocation	The percentage of time the associated user is allocated to this task.
booking_id	booking_id	The ID of the associated booking.
created	created	[Read-only] Time the record was created. See Date Fields .
externalid	external_id	If the record was imported from an external system you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
job_codeid	job_code_id	The ID of the associated job code.
pending_booking_id	pending_booking_id	The ID of the associated pending booking.

XML / SOAP	Database	Description
planned_hours	planned_hours	The hours for this user if the planned hours at the user level feature is enabled.
project_assignment_profile_id	project_assignment_profile_id	The ID of the associated project assignment profile.
project_groupid	project_group_id	The ID of the project group if the user was assigned as part of a project group.
projecttaskid	projecttask_id	[Required] ID of the project task to which this user is assigned. The project task must be a task and not a phase.
rule_rate_override	rule_rate_override	Hourly billing rate for the user assigned to the task. Effective only if your company's account configuration allows billing rule rate override on task assignments. Negative values are not allowed.
rule_rate_override_currency	rule_rate_override_currency	The 3-letter currency code for the billing rate currency. Defaults to the company's base currency if not set. Can be set to any currency specified in Administration > Global Settings > Organization > Currencies > Multi-currency if the Multicurrency feature is enabled for your company's OpenAir account.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Required] The ID of the user assigned to this task.

Usage Guidelines

Adding or modifying a Projecttaskassign assign triggers the project recalculation process in OpenAir unless you use the `no_recalc` attribute (see [Add, Update and Upsert Attributes](#)), or unless your company's account is configured never to trigger the project recalculation process in OpenAir when a change is made using OpenAir API.

ProjecttaskEstimate

A project task estimate [ProjecttaskEstimate] is a user estimation of the time remaining against a project task. The time remaining estimate is entered on timesheets and drives percent complete calculations against the project.

Review the [Usage Guidelines](#) for the ProjecttaskEstimate object.

—	XML	SOAP	REST	Database table
Object	ProjecttaskEstimate	oaProjecttaskEstimate	—	project_task_estimate
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The ProjecttaskEstimate object has the following properties:

XML / SOAP	Database	Description
changed_by	changed_by	[Read-only] ID of the user who changed the estimate. If this does not have an ID, then the estimate was automatically generated by OpenAir.
created	created	[Read-only] Time the record was created. See Date Fields .
date_changed	date_changed	[Read-only] The date and time the estimate was last changed. See Date Fields .
deleted	deleted	[Read-only] A "1/0" field indicating if the record was deleted
hours	hours	[Required] The number of hours estimated to be remaining
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
project_taskid	project_task_id	[Required] The ID of the associated project_task
timesheetid	timesheet_id	The ID of the associated timesheet if this was updated from the timesheet
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Required] The ID of the user who is assigned to the task

Usage Guidelines

Review the following guidelines:

- To add or modify a project task estimate:
 - The **Enable the "Hours remaining" on tasks estimating feature** box must be checked in the Timesheets application settings in OpenAir (Administration > Application Settings > Timesheets > other Settings).
 - The user with internal ID userid must be assigned to the task.
 - There must be a time entry against project_task_id in the timesheet with internal ID timesheet_id.
 - The same project task estimate must not already exist. The combination project_taskid, userid, and timesheetid (if set) must be unique.
- By default, you cannot modify a project task estimate associated with an approved or archived timesheet. Your account can be configured to allow using the OpenAir API to modify approved and archived timesheets. To enable the feature, contact OpenAir Customer Support.
- By default, updating a project task estimate triggers a recalculation for the project. Your account can be configured to disable recalculation when using the OpenAir API to modify projects and project tasks. To enable the feature, contact OpenAir Customer Support.

Projecttask_type

A project task type [Projecttask_type] is a classification group for project tasks used for time tracking and reporting purposes.

—	XML	SOAP	REST	Database table
Object	Projecttask_type	oaProjecttask_type	—	project_task_type

—	XML	SOAP	REST	Database table
Supported Commands	Add, Read, Modify	add(), read(), modify(), upsert()	—	—

The Projecttask_type object has the following standard properties:

Note: Projecttask_type object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
active	active	A 1/0 field specifying if the type is active.
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the projecttask_type.
notes	notes	Notes associated with the project task type.
picklist_label	—	Label as shown on form picklist.
suppress_notification	suppress_notification	Suppress task notifications for this project task type.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Proposal

A project proposal [Proposal] is a document outlining the cost of a project as part of a pipeline deal to seek an agreement from the customer.

—	XML	SOAP	REST	Database table
Object	Proposal	oaProposal	—	proposal
Supported Commands	Read	read()	—	—

The Proposal object has the following properties:

XML / SOAP	Database	Description
access_log	access_log	[Read-only] The mailing and access history of the proposal.
approved	approved	[Read-only] The date and time the proposal was approved. See Date Fields .
approved_by	—	[Read-only] The ID of the user who approved this proposal.
attachmentid	attachment_id	[Read-only] If non-zero, the attachment record associated with this proposal. attachment_id
created	created	[Read-only] Time the record was created. See Date Fields .
created_by	created_by	[Read-only] The ID of the user who created this proposal.

XML / SOAP	Database	Description
customerid	customer_id	[Read-only] The ID of the associated customer.
description	description	[Read-only] The description of this proposal.
expires	expires	[Read-only] The date the proposal is valid until. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name of this proposal.
notes	notes	[Read-only] Notes associated with this proposal.
number	number	[Read-only] The proposal number.
projectid	—	[Read-only] The ID of the associated project.
responded	responded	[Read-only] The date and time the customer accepted or refused. See Date Fields .
response	response	[Read-only] Customer response notes.
sent	sent	[Read-only] The date and time the proposal was delivered to the customer. See Date Fields .
status	status	[Read-only] The status of the proposal: <ul style="list-style-type: none"> ■ D - Draft ■ M - Submitted ■ P - Approved ■ Q - Rejected ■ S - Sent ■ V - Viewed ■ A - Accepted ■ R - Refused
submitted	submitted	[Read-only] The date and time the proposal was submitted for approval. See Date Fields .
total	total	[Read-only] The total amount. Dated by the currency_date field.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Read-only] The ID of the associated user.
viewed	viewed	[Read-only] The date and time the customer first viewed the proposal. See Date Fields .

Proposalblock

A proposal block [Proposalblock] is a component of a proposal including information as text or a project cost (labor time, fixed fee, expense, or purchase item).

—	XML	SOAP	REST	Database table
Object	Proposalblock	oaProposalblock	—	proposal_block
Supported Commands	Read	read()	—	—

The Proposalblock object has the following properties:

XML / SOAP	Database	Description
categoryid	category_id	[Read-only] The ID of the associated category.
content	content	[Read-only] The content of the template.
cost	cost	[Read-only] The cost per unit of measure (in the currency of the proposal) for an E block, the billing rate for an O block, or the fixed price for a F block. Dated by the currency_date field.
created	created	[Read-only] Time the record was created. See Date Fields .
description	description	[Read-only] The description of this proposal.
hour	hour	[Read-only] The number of hours for a T block.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
itemid	item_id	[Read-only] The ID of the associated item.
minute	minute	[Read-only] The number of minutes for a T block.
name	name	[Read-only] The name of the this proposal block.
proposalid	proposal_id	[Read-only] The ID of the associated proposal.
quantity	quantity	[Read-only] The quantity for an E block or an O block.
rate	rate	[Read-only] The hourly rate for a T block. Dated by the currency_date field.
seq	seq	[Read-only] The sequence number of the block.
slipid	slip_id	[Read-only] The ID of the associated slip if this block was billed to TB.
templateid	template_id	[Read-only] The ID of the associated template.
total	total	[Read-only] The total value of the block. Dated by the currency_date field.
type	type	[Read-only] The type of the slip: <ul style="list-style-type: none"> ■ X - is a text only block ■ T - is an hourly rate block ■ E - is an expense block ■ F - is a flat price block ■ O - is an other type block ■ P - is a product block
um	um	[Read-only] The unit of measure for an E block or the rate description for an O block.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Proxy

A proxy [Proxy] is a user who can act as another user in OpenAir.

Review the [Usage Guidelines](#) for the Proxy object.

—	XML	SOAP	REST	Database table
Object	Proxy	oaProxy	—	proxy
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The Proxy object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
expiration	expiration	The date the proxy expires. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
own	own	A 1/0 field indicating if the proxy was created by proxy_id using 'create own proxy' feature.
proxy_id	proxy_id	[Required] Internal ID of the user on behalf of whom the user with internal ID user_id is accessing OpenAir.
role_id	role_id	[Required] Internal ID of the role to use when the user with internal ID user_id is accessing OpenAir as and on behalf of the user with internal ID proxy_id.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
user_id	user_id	[Required] Internal ID of the user who is accessing OpenAir as and on behalf of the user with internal ID proxy_id.

Usage Guidelines

Your account can be configured to allow using the OpenAir API to add, modify or delete Proxy objects. To enable the feature, contact OpenAir Customer Support.

When enabled, the authenticated user must be an account administrator to add, modify or delete Proxy objects.

The combination of proxy_id and user_id must be unique.

Purchase_item

A purchase item [Purchase_item] is an entry in a purchase order.

Review the [Usage Guidelines](#) for the Purchase_item object.

—	XML	SOAP	REST	Database table
Object	Purchase_item	oaPurchase_item	—	purchase_item
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The Purchase_item object has the following standard properties:

Note: Purchase_item object properties may also include custom fields. The object type supports the custom equal to read method and the enable_custom read attribute.

XML / SOAP	Database	Description
acct_date	acct_date	The accounting period date of the purchase item. See Date Fields .
allow_vendor_substitution	allow_vendor_substitution	A 1/0 field indicating whether the vendor may be substituted.
approved_cost	approved_cost	A snap-shot of the approved cost from the request item (in the currency of the purchase order). 3 decimal places for handling amounts like mileage at 32.5 cents. Dated by the date field.
attachmentid	attachment_id	If non-zero, the attachment record associated with this purchaseitem.
cost	cost	The cost per unit of measure at which the product is ordered (in the currency of the purchase order). 3 decimal places for handling amounts like mileage at 32.5 cents. Dated by the date field.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
customerid	customer_id	The ID of the associated customer.
date	date	[Required] The date of the purchase item. The same as the purchase order date. See Date Fields .
date_fulfilled	date_fulfilled	The date on which all of the quantity was fulfilled. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
manufacturer_part	manufacturer_part	The manufacturer's part number, SKU or other unique identification for this product.
manufacturerid	manufacturer_id	The ID of the associated manufacturer.
name	name	The purchase name.
non_po	non_po	A 1/0 field indicating that this purchase item was created without a purchase order.
notes	notes	Notes associated with this purchase order block.
order_reference_number	order_reference_number	Unique reference number within purchase order.
productid	product_id	The ID of the associated product.
project_taskid	project_task_id	The ID of the associated project task.

XML / SOAP	Database	Description
projectid	project_id	The ID of the associated project.
purchaseorderid	purchaseorder_id	[Read-only] The ID of the associated purchase order.
purchaserequestid	purchaserequest_id	The ID of the associated purchase request.
purchaserid	purchaser_id	The ID of the purchaser or purchasing agent. This is always the same as the purchase order creator (purchaser_id).
quantity	quantity	[Read-only] The quantity of product_id for this purchase.
quantity_fulfilled	quantity_fulfilled	The quantity that has been fulfilled.
quantity_payable	quantity_payable	The quantity that is payable.
request_itemid	request_item_id	The ID of the associated request item.
tax_location_name	—	The name of the tax location.
total	total	The total value of the purchase (in the currency of the purchase order). Dated by the date field. Overridden by calculated value when adding or modifying an object.
total_with_tax	—	The total value of the purchase (in the currency of the purchase order) including tax. Dated by the date field.
um	um	The unit of measure for the product, e.g. EA.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	userid	The ID of the requester.
vendor_quote_number	vendor_quote_number	The vendor's quote number.
vendor_sku	vendor_sku	The vendor's sku for this product.
vendorid	vendor_id	The ID of the associated vendor

Usage Guidelines

There are several limitations impacting the addition or modification of purchase item information into OpenAir:

- You can add or modify a purchase item record only if it is not associated with a PO record. These purchase items are also referred to as "Quick POs" or "non-po purchase items".
 - The Quick PO functionality must be enabled for your OpenAir account. Otherwise, the OpenAir Integration Manager log shows "Error code 846: Cannot create non-po purchase items".

To enable the Quick PO functionality, go to Administration > Application Settings > Purchases > Other Settings. Scroll down and check the **Enable the ability to create quick POs. These are purchase items for purchases made without an OpenAir PO** box.

- non_po must be set to 1. Otherwise, OpenAir API returns the following error: "Error code 848: Only non_po purchase items can be added/modified".
- purchaseorderid must be empty. Otherwise, OpenAir API returns the following error: "Error code 847: purchaseorderid must be blank".
- An optional feature lets you update the project association (Customer: Project) for existing purchase item records associated with a PO using the OpenAir API. This is the only information you can modify using the OpenAir API. To enable project association update for purchase items associated with a PO, contact OpenAir Customer Support.

Purchaseorder

A purchase order (PO) [Purchaseorder] is a document used to purchase goods or services from an external source. It is a collection of purchase items.

Review the [Usage Guidelines](#) for the Purchaseorder object.

—	XML	SOAP	REST	Database table
Object	Purchaseorder	oaPurchaseorder	—	purchaseorder
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Purchaseorder object has the following standard properties:

 **Note:** Purchaseorder object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
accounts_payableid	accounts_payable_id	The accounts payable location for this purchase order.
approval_status	approval_status	A one-character string indicating the approval status of the purchase request. Possible values: <ul style="list-style-type: none"> ■ O - Open ■ P- Pending approval ■ A - Approved ■ R - Rejected
attachmentid	attachment_id	If non-zero, the attachment record associated with this purchase order.
auto_track_payable_with_fulfilled	auto_track_payable_with_fulfilled	A 1/0 field indicating that payability of quantities of items on this purchase order track automatically and directly with the fulfillment of those items.
carrierid	carrier_id	The carrier to be used for shipping. Ship Via.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	The currency this purchase order is in.

XML / SOAP	Database	Description
date	date	The date of the purchase order. See Date Fields .
date_approved	date_approved	The date the purchase order was approved. See Date Fields .
date_expected	date_expected	The date the materials are expected if known. See Date Fields .
date_fulfilled	date_fulfilled	The date on which all of the total quantity was fulfilled. See Date Fields .
date_order_placed	date_order_placed	The date the purchase order was placed with the vendor. See Date Fields .
date_required	date_required	The date the purchase items on this purchase order are required. See Date Fields .
date_shipped	date_shipped	The date the materials were shipped if known. See Date Fields .
date_submitted	date_submitted	The date the purchase order was submitted. See Date Fields .
description	description	The description or purpose for the purchase order.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
locationid	fob_location_id	The F.O.B. location_id (DEPRECATED).
name	name	The name of the purchase order (Prefix + number).
notes	notes	Notes to print on the purchase order.
number	number	The purchase order number. Automatically assigned with an increment of 1, if not set when adding a purchase order. Must be unique.
prefix	prefix	A static alphanumeric purchase order number prefix.
purchase_items_fulfilled	purchase_items_fulfilled	The total number of fulfilled purchase items in the purchase order.
quantity_fulfilled	quantity_fulfilled	The quantity fulfilled on all the purchase items in this purchase order.
receivingid	receiving_id	The receiving location for this purchase order.
ship_complete_only	ship_complete_only	A 1/0 field indicating that full order must ship together.
shipping_cost	shipping_cost	The cost of shipping, if known. Dated by the date field.

XML / SOAP	Database	Description
shipping_termsid	shipping_terms_id	The ID of the associated shipping payment terms, indicating how the shipping costs will be charged.
terms	terms	Payment terms for this purchase order.
total	total	The purchase order total cost. Dated by the date field.
total_purchase_items	total_purchase_items	The total number of purchase items in the purchase order.
total_quantity	total_quantity	The total quantity of all the purchase items in this purchase order.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Read-only] The ID of the user creating the purchase order. The purchasing agent.
vendorid	vendor_id	The ID of the associated vendor that the purchase order is for.

Usage Guidelines

You cannot delete a Purchaseorder object if this object is referenced by a [Purchase_item](#) object. Delete any dependent objects first before you delete a Purchaseorder object.

Purchaser

A purchaser [Purchaser] is a user who creates purchase orders.

—	XML	SOAP	REST	Database table
Object	Purchaser	oaPurchaser	—	purchaser
Supported Commands	Read	read()	—	—

The Purchaser object has the following standard properties:

Note: Purchaser object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
accounts_payableid	accounts_payable_id	[Read-only] The default accounts payable location for this purchaser.
active	active	[Read-only] A 1/0 field indicating where this is designated as an active receiving location.
attributes	attributes	[Read-only] A collection of additional attributes for this complex type.

XML / SOAP	Database	Description
carrierid	carrier_id	[Read-only] The default carrier to be used for shipping. Ship Via.
created	created	[Read-only] Time the record was created. See Date Fields .
exported	exported	[Read-only] Date and time the record was marked as exported. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name of the purchaser.
notes	notes	[Read-only] Notes associated with the purchaser.
receivingid	receiving_id	[Read-only] The default receiving location for this purchaser.
ship_complete_only	ship_complete_only	[Read-only] The default for the 1/0 field indicating that full order must ship together."
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Read-only] The ID of the associated user.

Purchaserequest

A purchase request [Purchaserequest] is a used to request the purchase of goods or services from external sources.

—	XML	SOAP	REST	Database table
Object	Purchaserequest	oaPurchaserequest	—	purchase_request
Supported Commands	Read	read()	—	—

The Purchaserequest object has the following standard properties:

Note: Purchaserequest object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
approval_status	approval_status	[Read-only] A one-character string indicating the approval status of the purchase request. Possible values: <ul style="list-style-type: none"> ■ O - Open ■ P - Pending approval ■ A - Approved ■ R - Rejected
attachmentid	attachment_id	[Read-only] If non-zero, the attachment record associated with this purchase request.

XML / SOAP	Database	Description
attributes	attributes	[Read-only] A collection of additional attributes for this complex type.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	[Read-only] The currency of the total field.
customerid	customer_id	[Read-only] The ID of the associated customer that the material on this purchase request is for.
date	date	[Read-only] The date of the purchase request. See Date Fields .
date_approved	date_approved	[Read-only] The date the purchaserequest was approved. See Date Fields .
date_fulfilled	date_fulfilled	[Read-only] The date on which all of the total quantity was fulfilled. See Date Fields .
date_required	date_required	[Read-only] The date the material on this purchase request is required. See Date Fields .
date_submitted	date_submitted	[Read-only] The date the purchaserequest was submitted. See Date Fields .
description	description	[Read-only] The description or purpose for the purchaserequest.
exported	exported	[Read-only] Date and time the record was marked as exported. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name of the purchaserequest (Prefix + number).
notes	notes	[Read-only] Notes associated with the purchase request.
number	number	[Read-only] The purchase request number that increments by 1.
ordered_request_items	ordered_request_items	[Read-only] The total number of request items on the purchase request which are part of a purchase order.
prefix	prefix	[Read-only] A static alphanumeric purchase request number prefix.
projectid	project_id	[Read-only] The ID of the associated project that the material on this purchase request is for.
quantity_fulfilled	quantity_fulfilled	[Read-only] The quantity fulfilled on all the request items in this purchase request.
request_items_fulfilled	request_items_fulfilled	[Read-only] The total number of fulfilled request items in the purchase request.
total	total	[Read-only] The purchase request total cost. Dated by the date field.

XML / SOAP	Database	Description
total_quantity	total_quantity	[Read-only] The total quantity of all the request items in this purchase request
total_request_items	total_request_items	[Read-only] The total number of request items in the purchase request.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Read-only] The ID of the associated user creating the purchase request. The requester.

Ratecard

A rate card [Ratecard] is a collection of hourly billing rates associated with job codes.

—	XML	SOAP	REST	Database table
Object	Ratecard	oaRatecard	—	rate_card
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Ratecard object has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this is an active rate card.
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The name of the rate card.
notes	notes	Notes associated with the rate card.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

RateCardItem

A rate card item [RateCardItem] is an hourly billing rate associated with a job code.

Review the [Usage Guidelines](#) for the RateCardItem object.

—	XML	SOAP	REST	Database table
Object	RateCardItem	oaRateCardItem	—	rate_card_item
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The RateCardItem object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
current	current	A 1/0 field indicating if the record is the current rate.
end	end	End date of the rate for historical records. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
job_code_id	job_code_id	[Required] The ID of the associated job code.
rate	rate	The hourly billing rate.
rate_card_id	rate_card_id	[Required] The ID of the rate card that it is associated with.
start	start	Start date of the rate for historical records. See Date Fields .
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Usage Guidelines

The combination of `job_code_id` and `rate_card_id` must be unique.

Reimbursement

A reimbursement [Reimbursement] is a sum of money paid to an employee against expenses claimed in an expense report.

—	XML	SOAP	REST	Database table
Object	Reimbursement	oaReimbursement	—	reimbursement
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Reimbursement object has the following standard properties:

Note: Reimbursement object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
audit	audit	[Read-only] Audit trail changes.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
date	date	[Required] The date of the reimbursement. See Date Fields .

XML / SOAP	Database	Description
envelope_number	—	The number of the associated envelope the reimbursement is applied to.
envelopeid	envelope_id	[Required] The associated envelope the reimbursement is applied to. The associated envelope must be approved.
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
notes	notes	Notes associated with the reimbursement.
total	total	The reimbursement total. Dated by the date field.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Read-only] The user associated with the envelope the reimbursement is applied to. Set automatically when adding objects.

Repeat

A recurrence [Repeat] specifies the frequency, number of occurrences or end date, and other information about a recurring event.

—	XML	SOAP	REST	Database table
Object	Repeat	oaRepeat	—	repeat
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Repeat object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
end	end	End date of the event. See Date Fields .
every	every	The spacing between each repeating event.
exclude_dow	exclude_dow	When frequency is in days, which days of the week (e.g. Mon, Tue, etc.) to exclude. This is a comma delimited list with 0 being Mon.
frequency	frequency	The repeating interval of the event: D – daily, W – weekly, M – monthly, Y – yearly.
how_end	how_end	How does this event end: D – date or O – occurrence
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
occur_number	occur_number	Number of occurrences.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Report

A report [Report] is a process of compiling and presenting OpenAir information for analysis and auditing purposes. The report object holds configuration information for a saved report.

—	XML	SOAP	REST	Database table
Object	Report	oaReport	—	report
Supported Commands	Read, Report	read()	—	—

The Report object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
date_created	date_created	[Read-only] The date and time the report was created. This may or may not be the same as the created column. For example, reports created before 2010-01-11 and reports copied from other accounts will have different values. See Date Fields .
email_report	—	[Read-only] A 1/0 field. 1 = report executes and sends an email with a pdf attachment to the session user.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name of the report.
relatedid	—	[Read-only] Related ID for attributes type. Report = ID of saved report, Timesheet = ID of timesheet, and Envelope = ID of related expense for expense report.
thin_client_context	thin_client_context	[Read-only] A 1/0 field indicating that this report can be requested via thin clients.
type	type	[Read-only] The type of report: S = saved reports and T = sTandard reports.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	userid	[Read-only] The ID of the user who created the report. This is 0 for standard reports.

Request_item

A request item [Request_item] is an entry in a purchase request.

—	XML	SOAP	REST	Database table
Object	Request_item	oaRequest_item	—	request_item
Supported Commands	ReadDelete	read()delete()	—	—

The Request_item object has the following standard properties:

Note: Request_item object properties may also include custom fields. The object type supports the custom equal to read method and the enable_custom read attribute.

XML / SOAP	Database	Description
allow_vendor_substitution	allow_vendor_substitution	[Read-only] A 1/0 field indicating whether the vendor may be substituted.
attachmentid	attachment_id	[Read-only] If non-zero, the attachment record associated with this request_item.
attributes	attributes	[Read-only] A collection of additional attributes for this complex type.
cost	cost	[Read-only] The cost per unit of measure at which the product is being requested. 3 decimal places for handling amounts like mileage at 32.5 cents. Dated by the date field.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	[Read-only] The currency used for this request item.
customerid	customer_id	[Read-only] The ID of the associated customer. This is always the same as the purchase request customer_id.
date	date	[Read-only] The date of the request_item. The same as the purchaserequest date. See Date Fields .
date_fulfilled	date_fulfilled	[Read-only] The date on which all of the quantity was fulfilled. See Date Fields .
exported	exported	[Read-only] Date and time the record was marked as exported. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
manufacturer_part	manufacturer_part	[Read-only] The manufacturer's part number, SKU or other unique ID for this product.
manufacturerid	manufacturer_id	[Read-only] The ID of the associated manufacturer.
name	name	[Read-only] The request item name.
notes	notes	[Read-only] Notes associated with this request item.
productid	product_id	[Read-only] The ID of the associated product.
projectid	project_id	[Read-only] The ID of the associated project. This is always the same as the purchase request project_id.

XML / SOAP	Database	Description
purchase_itemid	purchase_item_id	[Read-only] The ID of the associated purchase_item.
purchaseorderid	purchaseorder_id	[Read-only] The ID of the associated purchase order.
purchaserequestid	purchaserequest_id	[Read-only] The ID of the associated purchaserequest.
quantity	quantity	[Read-only] The quantity of product_id for this request item.
quantity_fulfilled	quantity_fulfilled	[Read-only] The quantity that has been fulfilled.
request_reference_number	request_reference_number	[Read-only] Unique reference number within purchase request.
total	total	[Read-only] The total value of the request item. Dated by the date field.
um	um	[Read-only] The unit of measure for the product, e.g. EA.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Read-only] The ID of the requester. This is always the same as the purchaserequest requester (user_id).
vendor_quote_number	vendor_quote_number	[Read-only] The vendor's quote number.
vendor_sku	vendor_sku	[Read-only] The vendor's sku for this product.
vendorid	vendor_id	[Read-only] The ID of the associated vendor.

ResourceAttachment

A resource attachment [ResourceAttachment] is an attachment associated to an employee's consolidated resource profile.

—	XML	SOAP	REST	Database table
Object	ResourceAttachment	oaResourceAttachment	—	resource_attachment
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The ResourceAttachment object has the following properties:

XML / SOAP	Database	Description
attachment_id	attachment_id	The attachment record associated with this document.

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
latest_attachment_id	latest_attachment_id	ID of the latest attachment from the attachment table.
type	type	[Read-only] The document type. Must be either "CV" or "AVATAR".
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Required] The ID of the user to whom this attachment belongs.

Usage Guidelines

Review the following guidelines:

- A resource can only have one attachment of each supported type. The combination of type and userid must be unique.
- Adding an attachment file as a resource attachment is a two-step process:
 1. Add a resource attachment object.
 2. An an attachment and set the owner_type value to ResourceAttachment and the owner_id to the internal ID of the resource attachment added in the first step. See [Attachment](#).

For an XML API sample code, see [Sample Code — Adding CV as Attachment to a Resource Profile](#).

Resourceprofile

A resource profile item [Resourceprofile] is a skill or competency that is part of an employee's resource profile.

—	XML	SOAP	REST	Database table
Object	Resourceprofile	oaResourceprofile	—	resourceprofile
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Resourceprofile object has the following properties:

XML / SOAP	Database	Description
attributeid	attribute_id	The ID of the optional resourceprofile attribute.
comment	comment	Additional comment describing this resourceprofile.
created	created	[Read-only] Time the record was created. See Date Fields .

XML / SOAP	Database	Description
externalid	external_id	If the record was imported from an external system you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The resourceprofile name.
resourceprofile_typeid	resourceprofile_type_id	[Required] The ID of the resourceprofile_type.
type	type	[Required] The resourceprofile type. The entity on which this resourceprofile is based. <ul style="list-style-type: none"> ■ Skill ■ Education ■ Location ■ Jobrole ■ Industry ■ Customprofile_1..20
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Required] The ID of the user for which this resourceprofile describes.

Resourceprofile_type

A resource profile item type [Resourceprofile_type] is a classification grouping for resource profile items.

—	XML	SOAP	REST	Database table
Object	Resourceprofile_type	oaResourceprofile_type	—	resourceprofile_type
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

A Resourceprofile_type has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether this is active.
attribute_set_id	attribute_set_id	The ID of the associated attribute set.
created	created	[Read-only] Time the record was created. See Date Fields .
externalid	external_id	If record was imported from an external system, store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The resourceprofile_type name. This shows up on all the resourceprofile_type popup windows in the application.

XML / SOAP	Database	Description
related_table	related_table	The name of the table related with this table.
relatedid	related_id	The ID of the related item in the related table.
type	type	The resourceprofile type. The entity on which this resourceprofile is based. <ul style="list-style-type: none"> ■ Skill ■ Education ■ Location ■ Jobrole ■ Industry ■ Customprofile_1..20
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

ResourceRequest

A resource request [ResourceRequest] is a collection of resource request queues.

—	XML	SOAP	REST	Database table
Object	ResourceRequest	oaResourceRequest	—	resource_request
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The ResourceRequest object has the following standard properties:

 **Note:** ResourceRequest object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	XML / SOAP	Description
booking_type_id	booking_type_id	The booking type of bookings created for this resource request. Required if the Require booking type when booking resources box is checked in the Resources application settings in OpenAir (Administration > Application Settings > Resources > Other Settings).
created	created	[Read-only] Time the record was created. See Date Fields .
customerid	customer_id	[Required] The ID of the associated customer
date_end	date_end	[Required] The ending date of the resource request. See Date Fields . <code>date_start</code> and <code>date_end</code> must define a valid date range.
date_finalized	date_finalized	The date the resource request was finalized and marked ready for booking. See Date Fields .
date_start	date_start	[Required] The starting date of the resource request. See Date Fields . <code>date_start</code> and <code>date_end</code> must define a valid date range.

XML / SOAP	XML / SOAP	Description
date_start_expected	date_start_expected	The expected starting date of the resource request. See Date Fields .
external_id	external_id	If the record was imported from an external system you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The name of the resource request.
notes	notes	Notes field
number	number	The resource request tracking number. Assigned automatically if not set when adding a resource request. Must be unique.
ownerid	owner_id	The ID of the associated user creating the resource request. Defaults to the internal ID of the authenticated if not set when adding a resource request.
percent_fulfilled	percent_fulfilled	Percent fulfilled for the resource request.
projectid	project_id	[Required] The ID of the associated project.
status	status	The status of the resource request: <ul style="list-style-type: none"> ■ 'O' - Open ■ 'P' - Partial ■ 'S' - Complete ■ 'C' - Canceled
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

ResourceRequestQueue

A resource request queue [ResourceRequestQueue] is a container of search criteria, pending bookings for resources matching the search criteria, and bookings created from pending bookings.

—	XML	SOAP	REST	Database table
Object	ResourceRequestQueue	oaResourceRequestQueue	—	resource_request_queue
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The ResourceRequestQueue object has the following standard properties:

Note: ResourceRequestQueue object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom_read` attribute.

XML / SOAP	Database	Description
booking_type_id	booking_type_id	The booking type of bookings created for this resource request queue.
created	created	[Read-only] Time the record was created. See Date Fields .

XML / SOAP	Database	Description
customerid	customer_id	[Required] The ID of the associated customer.
date_end	date_end	[Required] The ending date of the resource request queue. See Date Fields . date_start and date_end must define a valid date range.
date_start	date_start	[Required] The starting date of the resource request queue. See Date Fields . date_start and date_end must define a valid date range.
external_id	external_id	If the record was imported from an external system you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The name of the resource request queue.
notes	notes	Notes field.
number	number	The resource_request_queue tracking number. Assigned automatically if not set when adding a resource request. Must be unique.
percent_fulfilled	percent_fulfilled	Percent fulfilled for the resource request queue.
projectid	projectid	[Required] The ID of the associated project.
resource_request_id	resource_request_id	The ID of the associated resource request.
resourcesearch_id	resourcesearch_id	The ID of the associated base resource search. If resourcesearch_id is empty when adding a resource research queue, OpenAir API creates an empty resource search and sets resourcesearch_id to the internal ID of this empty resource search.
slots	slots	The number of slots available in this queue.
status	status	The status of the resource request queue: <ul style="list-style-type: none"> ■ 'O' - Open ■ 'P' - Partial ■ 'S' - Complete ■ 'C' - Canceled
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Resourcesearch

A resource search [Resourcesearch] is a definition of resource search criteria for a resource request queue.

—	XML	SOAP	REST	Database table
Object	Resourcesearch	oaResourcesearch	—	resourcesearch

—	XML	SOAP	REST	Database table
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The ResourceSearch object has the following properties:

XML / SOAP	Database	Description
advanced_options	advanced_options	JSON object detailing advanced search options (i.e. Required, Preferred, Excluding). Used for post-processing search results for ranking purposes.
as_percentage	as_percentage	A "1/0" field indicating which of the fields... hours or percentage is to be used in the search <ul style="list-style-type: none"> ■ If 1 then use percentage. ■ If 0 then use hours.
availability_search	availability_search	A 1/0 field indicating whether to search by availability.
consecutive_availability	consecutive_availability	A 1/0 field indicating no intervening bookings.
created	created	[Read-only] Time the record was created. See Date Fields .
customprofile_<N> Where <N> is an integer from 1 to 35	customprofile_<N> Where <N> is an integer from 1 to 35	Comma delimited list of customprofile_<N> that make up this search Each element is the ID followed by an optional attribute ID, separated by a colon (:).
education	education	Comma delimited list of educations that make up this search. Each element is the ID followed by an optional attribute ID, separated by a colon (:).
enddate	enddate	The end date for availability. See Date Fields .
essential	—	See Resource Search Virtual Fields .
excluding	—	See Resource Search Virtual Fields .
external_id	external_id	If the record was imported from an external system you store the unique external record ID here.
hours	hours	The number of hours of availability required over this range.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
include_generic_resources	include_generic_resources	A 1/0 field. Include generic resources in search?
include_inactive_resources	include_inactive_resources	A 1/0 field. Include inactive resources in search?

XML / SOAP	Database	Description
include_regular_resources	include_regular_resources	A 1/0 field. Include regular resources in search?
industry	industry	Comma delimited list of industries that make up this search. Each element is the ID followed by an optional attribute ID, separated by a colon (:).
jobrole	jobrole	Comma delimited list of job roles that make up this search. Each element is the ID followed by an optional attribute ID, separated by a colon (:).
location	location	Comma delimited list of locations that make up this search. Each element is the ID followed by an optional attribute ID, separated by a colon (:).
name	name	The resourcesearch name.
percentage	percentage	The percentage of time booked to this project during this daterange. This is either the actual booked percentage or derived from the hours.
preferred	—	See Resource Search Virtual Fields .
required	—	See Resource Search Virtual Fields .
resource_request_queue_id	resource_request_queue_id	The ID of the associated resource request queue.
skill	skill	Comma delimited list of skills that make up this search. Each element is the ID followed by an optional attribute ID, separated by a colon (:).
startdate	startdate	The start date for availability. See Date Fields .
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Resource Search Virtual Fields

Resource searches use four virtual fields: essential, excluding, preferred and required. These fields are processed during read and write operations for Resource Demand Request (RDR) searches.

The fields use comma separated resourceprofile_type.id : attribute.id pairs to specify criteria.



Note: Set attribute.id to 0 to include any level of the competency defined for the resource profile type or if there are no attribute sets defined for the resource profile.

The following example searches for resources who preferably have beginner CRM competencies, intermediate Linux competencies and a Master's degree:

- preferred set to the value 11:1,10:2,12:0.
- resourceprofile_type internal ID and name (id # name) pairs:

- 10 # "Linux"
- 11 # "CRM"
- 12 # "Master's degree"
- attribute internal ID and name (id # name) pairs:
 - 1 # "Beginner"
 - 2 # "Intermediate"
 - 3 # "Expert"
- attribute.id = 1 for "Beginner", 2 for "Intermediate", and 3 for "Expert".

RevenueContainer

A revenue container [RevenueContainer] is a collection of recognition transactions in different revenue stages that can be used for approval.

—	XML	SOAP	REST	Database table
Object	RevenueContainer	oaRevenueContainer	—	revenue_container
Supported Commands	Read, Modify	read(), modify(), upsert()	—	—

The RevenueContainer object has the following properties:

XML / SOAP	Database	Description
acct_date	acct_date	[Read-only] The accounting period date of the revenue_container. See Date Fields .
approval_status	approval_status	[Read-only] A one-character string indicating the approval status of the invoice. Only used if invoice approvals are used. Possible values: <ul style="list-style-type: none"> ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected
balancing_type	balancing_type	[Read-only] A one-character key indicating the type of balancing for this revenue_container. Note that All revenue_containers for a project have the same balancing_type: <ul style="list-style-type: none"> ■ A - Agreement ■ C - CustomerPO ■ P - Project ■ X - Agreement and CustomerPO
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	[Read-only] The currency of this revenue_container.
customerid	customer_id	[Read-only] The ID of the associated customer.
date	date	[Read-only] The date of the revenue_container. See Date Fields .
date_approved	date_approved	[Read-only] The date the invoice was approved. See Date Fields .

XML / SOAP	Database	Description
date_submitted	date_submitted	[Read-only] The date the invoice was submitted. See Date Fields .
exported	exported	[Read-only] Date and time the record was marked as exported. See Date Fields .
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name of the revenue_container (Prefix + number).
notes	notes	[Read-only] Notes to print on the revenue_container.
number	number	[Read-only] The revenue_container number that increments by 1.
prefix	prefix	[Read-only] A static alphanumeric revenue_container number prefix.
projectid	project_id	[Read-only] The ID of the associated project.
total_accrued	total_accrued	[Read-only] The revenue_container accrued total. Dated by the date field.
total_deferred	total_deferred	[Read-only] The revenue_container deferred total. Dated by the date field.
total_invoiced	total_invoiced	[Read-only] The revenue_container invoice total. Dated by the date field.
total_posted	total_posted	[Read-only] The revenue_container posted total. Dated by the date field.
total_recognized	total_recognized	[Read-only] The revenue_container recognized total. Dated by the date field.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

RevenueProjection

A revenue projection [RevenueProjection] is a charge (slip) created from a charge projection run.

—	XML	SOAP	REST	Database table
Object	RevenueProjection	oaRevenueProjection	—	revenue_projection , slip
Supported Commands	Read	read()	—	—

The RevenueProjection object has the following standard properties:

Note: RevenueProjection object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	XML / SOAP	Description
acct_date	slip.acct_date	[Read-only] The accounting period date of the slip. See Date Fields .

XML / SOAP	XML / SOAP	Description
agreement_id	slip.agreement_id	[Read-only] The ID of the associated agreement.
booking_type_id	revenue_projection.booking_type_id	[Read-only] ID of the booking type if this was generated from bookings.
category_<N>_id where <N> is an integer from 1 to 5	slip.category_<N>_id where <N> is an integer from 1 to 5	[Read-only] The ID of the associated category_<N>.
category_id	slip.category_id	[Read-only] The ID of the associated category. If this is set, the slip is based on this category.
city	slip.city	[Read-only] The slip city or location.
cost	slip.cost	[Read-only] The cost per unit of measure for an E or P slip, the billing rate for an O slip, or the fixed price for a F slip. Dated by the date field.
cost_center_id	slip.cost_center_id	[Read-only] The ID of the associated cost center.
cost_includes_tax	slip.cost_includes_tax	[Read-only] A 1/0 field indicating whether the cost includes the tax.
created	slip.created	[Read-only] Time the record was created. See Date Fields .
currency	slip.currency	[Read-only] Currency for the money fields in the record.
customer_id	slip.customer_id	[Read-only] The ID of the associated customer.
customerpo_id	slip.customerpo_id	[Read-only] The ID of the associated customerpo.
date	slip.date	[Read-only] The date of the billing slip. See Date Fields .
description	slip.description	[Read-only] The description of the billing slip.
exported	slip.exported	[Read-only] Date and time the record was marked as exported. See Date Fields .
hour	slip.hour	[Read-only] The number of hours for a T slip.
id	slip.id	[Read-only] Unique ID. Automatically assigned by OpenAir.
incomplete	slip.incomplete	[Read-only] Is the slip complete, e.g. can it be included in an invoice. If 1 it must be edited before it can be added to an invoice.
invoice_id	slip.invoice_id	[Read-only] The ID of the associated invoice once billed.
item_id	slip.item_id	[Read-only] The ID of the associated item. If this is set, the slip is based on this item. Use the associated item type to determine the subtype of this slip.
job_code_id	slip.job_code_id	[Read-only] The ID of the associated job code.
minute	slip.minute	[Read-only] The number of minutes for a T slip.
name	slip.name	[Read-only] The name of the slip. This field is never populated.
notes	slip.notes	[Read-only] Notes associated with the slip.

XML / SOAP	XML / SOAP	Description
originating_id	slip.originating_id	[Read-only] For use with split slips feature. If set, the slip.id of the originating slip for this split portion.
payment_type_id	slip.payment_type_id	[Read-only] The ID of the associated payment type.
payroll_type_id	slip.payroll_type_id	[Read-only] The ID of the associated payroll type.
portfolio_project_id	slip.portfolio_project_id	[Read-only] The ID of the associated portfolio project.
product_id	slip.product_id	[Read-only] The ID of the associated product.
project_billing_rule_id	revenue_projection.project_billing_rule_id	[Read-only] The ID of the associated project billing rule.
project_id	slip.project_id	[Read-only] The ID of the associated project.
project_task_id	slip.project_task_id	[Read-only] The ID of the task within the associated project.
projecttask_type_id	slip.projecttask_type_id	[Read-only] The ID of the projecttask_type of the associated projecttask.
quantity	slip.quantity	[Read-only] The quantity for an E, O, or P slip.
rate	slip.rate	[Read-only] The hourly rate for a T slip. Dated by the date field.
ref_slip_id	slip.ref_slip_id	[Read-only] For credit/rebill, ID of the original slip ID.
repeat_id	slip.repeat_id	[Read-only] The ID of the associated repeating event.
revenue_projection_type	revenue_projection.revenue_projection_type	[Read-only] The type of the projection: <ul style="list-style-type: none"> ■ R - Revenue from an "As billed" recognition rule ■ F - Revenue from an "Fixed fee" recognition rule ■ G - Revenue from an "Percent complete" recognition rule ■ H - Revenue from an "Incurred vs. forecast" recognition rule ■ J - Revenue from a "Time project billing rule" rule ■ U - Time billed but not recognized ■ T - Time not billed
revenue_recognition_rule_id	revenue_projection.revenue_recognition_rule_id	[Read-only] Id of the revenue recognition rule that created this projection.
revenue_stage_id	revenue_projection.revenue_stage_id	[Read-only] Id of the revenue_stage. This will always be 'no revenue stage' 0 for revenue projections.
slip_projection_id	revenue_projection.slip_projection_id	[Read-only] Id of the slip_projection that was used for an as billed rule
slip_projection_type	revenue_projection.slip_projection_type	[Read-only] The type of the slip_projection: <ul style="list-style-type: none"> ■ X - slip projection generated from billing rule

XML / SOAP	XML / SOAP	Description
		<ul style="list-style-type: none"> ■ B - Time from potentially billable transaction which did not match any billing rule ■ N - Time from transaction with non-billable project-task ■ P - Time from transaction matching a billing rule, but is Partially over cap ■ S - Time from transaction matching a billing rule, but is completely over cap and rule indicates to Stop if capped ■ C - Time from transaction matching a billing rule, but is completely over cap and no more rules match
slip_stage_id	slip.slip_stage_id	[Read-only] The ID of the associated slip stage.
slip_type_id	slip.slip_type_id	[Read-only] This field is redundant with the type field. It provides a linkage to the slip type table allowing the slip_type table to be used in the reporting mechanism.
timer_start	slip.timer_start	[Read-only] The starting time of the timer. See Date Fields .
timetype_id	slip.timetype_id	[Read-only] The ID of the associated time type.
total	slip.total	[Read-only] The total value of the slip. Dated by the date field.
total_hp	revenue_projection.total_hp	[Read-only] A high precision version of the total field. This is used for G" type transactions as the percent complete is calculated on a daily basis can be a small number Dated by the date field."
total_tax_paid	slip.total_tax_paid	[Read-only] The total tax paid. Dated by the date field.
transaction_id	revenue_projection.transaction_id	[Read-only] For internal user only. It is used only to satisfy subtotalling by slip in summary reports.
type	slip.type	[Read-only] The type of the slip: T - hourly rate slip, E - expense slip, F - flat price slip, O - other time slip, M - incomplete slip, or P - product slip.
um	slip.um	[Read-only] The unit of measure for an E or P slip or the rate description for an O slip.
updated	slip.updated	[Read-only] Time the record was last updated or modified. See Date Fields .
user_id	slip.user_id	[Read-only] The ID of the associated user.
vehicle_id	slip.vehicle_id	[Read-only] The ID of the associated vehicle.

Usage Guidelines

You can only read revenue projection objects using the OpenAir API if there's no charge projection run in progress. Otherwise, no results are returned. This is because the data may be incomplete until the charge projection job completes.

Revenue_recognition_rule

A revenue recognition rule [Revenue_recognition_rule] is a rule governing how and when revenue is recognized.

—	XML	SOAP	REST	Database table
Object	Revenue_recognition_rule	oaRevenue_recognition_rule	—	revenue_recognition_rule
Supported Commands	Add, Read, Modify	add(), read(), modify(), upsert()	—	—

The Revenue_recognition_rule object has the following standard properties:

Note: Revenue_recognition_rule object properties may also include custom fields. The object type supports the custom equal to read method and the enable_custom read attribute.

XML / SOAP	Database	Description
accounting_period_id	accounting_period_id	The ID of the associated accounting period.
acct_code	acct_code	Optional accounting system code for integration with external accounting systems.
acct_date	acct_date	The accounting period date to assign to the transaction. See Date Fields .
acct_date_how	acct_date_how	The accounting period date of the transaction is determined by: <ul style="list-style-type: none"> ■ N - none, clear the value ■ E - the entity (no change) ■ C - container of the entity if available (e.g. timesheet, envelope) ■ M - set by the specified accounting date ■ P - set by the specified accounting period
active	active	A 1/0 field indicating whether this is an active rule.
agreementid	agreement_id	ID of the associated agreement.
amount	amount	The amount. If we have multiple amounts, the values are held in the revenue_recognition_rule_amount table.
asb_exclude_slip_type	asb_exclude_slip_type	CSV list of the slip types to exclude from the as billed rule.
asb_which_slips	asb_which_slips	Which slips should be considered for the as billed rule: <ul style="list-style-type: none"> ■ A - all slips ■ I - slips on invoices

XML / SOAP	Database	Description
		<ul style="list-style-type: none"> ■ P - slips on approved invoices
assigned_user	assigned_user	The user to assign to fixed fee recognition.
break_by_user	break_by_user	Break out the transactions by user. Currently only implemented for the incurred rules.
category_<N>id where <N> is an integer from 1 to 5	category_<N>_id where <N> is an integer from 1 to 5	The ID of the associated category_<N>. Mutually exclusive with project_task_id.
categoryid	category_id	The ID of the associated category.
cost_center_id	cost_center_id	The ID of the associated cost center.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
customerid	customer_id	The ID of the associated customer.
customerpo_id	customerpo_id	ID of the associated customerpo.
end_date	end_date	End date of the rule. See Date Fields .
end_milestone	end_milestone	ID of the ending milestone (project_task).
expense_how	expense_how	How expenses should be recognized: <ul style="list-style-type: none"> ■ M - mark up/down on billed expenses. ■ B - billed expenses. ■ I - incurred expenses.
extra_data	extra_data	Holds extra data fields associated with the rule.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
item_filter	item_filter	CSV list of items to limit the rule to.
marked_as_ready	marked_as_ready	Trigger recognition when a task (ID in phase) is marked as ready to recognize.
name	name	Name of the rule.
notes	notes	Notes associated with this revenue recognition rule.
percent	percent	The percentage value for a fixed fee percent trigger.
percent_complete	—	[Read-only] The calculated percent complete value if a type P transaction.
percent_how	percent_how	How percent complete should be calculated:

XML / SOAP	Database	Description
		<ul style="list-style-type: none"> ■ A - % complete of planned hours for the project. ■ B - % complete of planned hours for a phase. ■ C - Approved hours versus planned hours for the project. ■ D - Approved hours versus planned hours for a phase. ■ E - Approved hours versus budget hours for the project.
percent_trigger	percent_trigger	<p>If the fixed fee is triggered by a percent complete, this holds how it is triggered:</p> <ul style="list-style-type: none"> ■ A - % complete of planned hours for the project. ■ B - % complete of planned hours for a phase or task (the task ID is held in the phase field).
phase	phase	ID of the phase if percent_how is B or D. ID of the phase/task if this is a marked_as_ready or percent_trigger rule.
product_filter	product_filter	CSV list of products to limit the rule to.
project_billing_rule_filter	project_billing_rule_filter	CSV list of project billing rule id's to limit a type T rule to.
project_billing_ruleid	project_billing_rule_id	The ID of the associated project billing rule. Only available if the optional feature "Show billing rules on revenue recognition forms" is enabled."
project_task_filter	project_task_filter	CSV list of tasks to limit the rule to.
projectid	projectid	The ID of the associated project.
purchase_how	purchase_how	<p>How purchases should be recognized:</p> <ul style="list-style-type: none"> ■ M - mark up/down on billed purchases. ■ B - billed purchases.
recognition_type	recognition_type	<p>What we are recognizing:</p> <ul style="list-style-type: none"> ■ R - revenue ■ C - cost ■ O - other
repeatid	repeat_id	The ID of the associated repeating event.
slip_stage_filter	slip_stage_filter	CSV list of slip_stage ID to limit a type A rule to.
start_date	start_date	Start date of the rule. See Date Fields .
start_milestone	start_milestone	ID of the starting milestone (project_task).

XML / SOAP	Database	Description
timetype_filter	timetype_filter	CSV list of timetypes to limit the rule to.
type	type	The type of the rule: <ul style="list-style-type: none"> ■ A - as billed rule ■ P - percent of time complete rule ■ E - expense incurred rule ■ F - fixed amount rule ■ U - purchase item rule ■ I - incurred versus forecast rule ■ T - generated from a time project billing rule
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
user_filter	user_filter	CSV list of users to limit the rule to.

Revenue_recognition_rule_amount

A revenue recognition rule amount [Revenue_recognition_rule_amount] is one of multiple amounts associated with a revenue recognition rule. Used if a revenue recognition rule can have multiple amounts.

—	XML	SOAP	REST	Database table
Object	Revenue_recognition_rule_amount	oaRevenue_recognition_rule_amount	—	revenue_recognition_rule_amount
Supported Commands	Add, Read, Modify	add(), read(), modify(), upsert()	—	—

The Revenue_recognition_rule_amount has the following properties:

XML / SOAP	Database	Description
acct_code	acct_code	Optional accounting system code for integration with external accounting systems.
agreement_id	agreement_id	The ID of the associated agreement.
amount	amount	The amount.
category_<N>id where <N> is an integer from 1 to 5	category_<N>_id where <N> is an integer from 1 to 5	The ID of the associated category_<N>. Mutually exclusive with project_task_id.
category_id	category_id	The ID of the associated category.
cost_center_id	cost_center_id	The ID of the associated category.
created	created	[Read-only] Time the record was created. See Date Fields .

XML / SOAP	Database	Description
currency	currency	Currency for the money fields in the record.
customerpo_id	customerpo_id	The ID of the associated customerpo.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
recognition_type	recognition_type	Recognition type: <ul style="list-style-type: none"> ■ R - revenue ■ C - cost ■ O - other
revenue_recognition_rule_id	revenue_recognition_rule_id	The ID of the associated rule.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Revenue_recognition_transaction

A revenue recognition transaction [Revenue_recognition_transaction] is a transaction created based on a revenue recognition rule.

—	XML	SOAP	REST	Database table
Object	Revenue_recognition_transaction	oaRevenue_recognition_transaction	—	revenue_recognition_transaction
Supported Commands	Add, Read, Modify	add(), read(), modify(), upsert()	—	—

The Revenue_recognition_transaction object has the following standard properties:

Note: Revenue_recognition_transaction object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
acct_code	acct_code	Optional accounting system code for integration with external accounting systems.
acct_date	acct_date	The accounting period date of the transaction. See Date Fields .
agreementid	agreement_id	The ID of the associated agreement.
categoryid	category_id	The ID of the associated category.
category_<N>id where <N> is an integer from 1 to 5	category_<N>_id where <N> is an integer from 1 to 5	The ID of the associated category_<N>. Mutually exclusive with project_task_id.
cost_center_id	cost_center_id	The ID of the associated cost center.

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
customerid	customer_id	The ID of the associated customer.
customerpo_id	customerpo_id	The ID of the associated customerpo.
date	date	The date of the transaction. See Date Fields .
decimal_hours	—	The number of decimal hours.
hour	hour	The number of hours for a T type.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
is_from_open_stage	is_from_open_stage	A 1/0 field indicating that the revenue recognition transaction was added to the revenue container from the virtual open stage, otherwise the transaction was added through revenue container revenue_recognition_transaction generation logic. If revenue_container_id is zero, revenue_stage_id should be 0 and is_from_open_stage should be 0.
job_codeid	job_code_id	The ID of the associated job code.
minute	minute	The number of minutes for a T type.
notes	notes	Notes associated with this revenue recognition transaction.
offsetsid	offsets_id	The ID of the revenue_recognition_transaction which this revenue_recognition_transaction offsets.
originatingid	originating_id	The ID of the originating revenue_recognition_transaction for this revenue_recognition_transaction.
percent_complete	percent_complete	[Read-only] The calculated percent complete value if it is a type P transaction.
portfolio_projectid	portfolio_project_id	The ID of the associated portfolio project
project_billing_ruleid	project_billing_rule_id	The ID of the associated project billing rule.
project_taskid	project_task_id	The ID of the associated project task.
projectid	project_id	The ID of the associated project.
rate	rate	The hourly rate for a T type. Dated by the date field.
recognition_type	recognition_type	Recognition type: <ul style="list-style-type: none"> ■ R - revenue ■ C - cost

XML / SOAP	Database	Description
		<ul style="list-style-type: none"> ○ - other
revenue_containerid	revenue_container_id	The ID of the associated revenue_container once posted.
revenue_recognition_ruleid	revenue_recognition_rule_id	The ID of the associated rule.
revenue_stageid	revenue_stage_id	The ID of the associated revenue stage.
slipid	slip_id	The ID of the associated slip.
taskid	task_id	The ID of the associated task.
ticketid	ticket_id	The ID of the associated ticket.
total	total	The amount of the transaction. Dated by the date field.
type	type	The type of the transaction. Matches the type field in revenue_recognition_rule.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	The ID of the associated user.

RevenueStage

A revenue stage [RevenueStage] is a progression step in the revenue recognition transaction life cycle.

—	XML	SOAP	REST	Database table
Object	RevenueStage	oaRevenueStage	—	revenue_stage
Supported Commands	Read	read()	—	—

The RevenueStage object has the following standard properties:

Note: RevenueStage object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name of the revenue stage.
revenue_stage_type	revenue_stage_type	[Read-only] A one-character key indicating the type of revenue for this revenue_stage: <ul style="list-style-type: none"> ■ D - Deferral ■ A - Accrual ■ F - Final

XML / SOAP	Database	Description
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Role

A role [Role] is a function of an employee and determines what functionality the employee has access to in OpenAir.

—	XML	SOAP	REST	Database table
Object	Role	oaRole	—	role
Supported Commands	Read	read()	—	—

The Role object has the following properties:

XML / SOAP	Database	Description
admin_role	admin_role	[Read-only] A 1/0 field indicating whether this is the chief administrator role, with full rights.
created	created	[Read-only] Time the record was created. See Date Fields .
default_role	default_role	[Read-only] A 1/0 field indicating whether this is the default new user role.
deleted	deleted	[Read-only] A "1/0" field indicating if the record was deleted
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name of this role.
notes	notes	[Read-only] Notes associated with this role.
permissions	permissions	[Read-only] A set of role permissions. <ul style="list-style-type: none"> ■ A collection of Flag objects (XML API). ■ A comma-separated list of enabled role permission names (SOAP API).
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Schedulebyday

A schedule by day [Schedulebyday] is an employee's work schedule on a specific date.

Review the [Usage Guidelines](#) for the Schedulebyday object.

—	XML	SOAP	REST	Database table
Object	Schedulebyday	oaSchedulebyday	—	schedule_by_day
Supported Commands	Read	read()	—	—

The Schedulebyday object has the following properties:

XML / SOAP	Database	Description
base_hours	base_hours	[Read-only] The number of base hours on this date for this user.
created	created	[Read-only] Time the record was created. See Date Fields .
date	date	[Read-only] The date. See Date Fields .
hours	hours	[Read-only] The number of schedule hours on this date for this user, including exceptions.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
target_base_hours	target_base_hours	[Read-only] The number of target base hours for this user on this date $\text{Target_utilization.percentage} * \text{base_hours}$.
target_hours	target_hours	[Read-only] The number of target hours for this user on this date. $\text{Target_utilization.percentage} * \text{hours}$.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
user_id	user_id	[Read-only] The ID of the associated user.

Usage Guidelines

Review the following guidelines:

- You must use the following read attributes when reading Schedulebyday objects using the all read method: start_date, end_date and user_filter. See [Read Attributes](#).
- When using the **XML API**, reading Schedulebyday objects using the all read method only returns exceptions to the base schedule. That is, the response includes only Schedulebyday objects for which hours is not equal to base_hours.

For example, the following request, returns all schedule exceptions for the employees with internal ID 145 and 146, or the exceptions to the company work schedule associated with these employees, between October 1st and October 31st, 2023.

```
<Read type="Schedulebyday" method="all" start_date="2023-10-01" end_date="2023-10-31"
user_filter="145,146" limit="10"> ...</Read>
```

To return all records instead of exceptions only use the equal to read method. See [Read Methods](#).

Scheduleexception

A schedule exception [Scheduleexception] is an exception to the normal work pattern in the company's or the employee's work schedule.

Review the [Usage Guidelines](#) for the Scheduleexception object.

—	XML	SOAP	REST	Database table
Object	Scheduleexception	oaScheduleexception	—	scheduleexception
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

A Scheduleexception object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
enddate	enddate	[Required] The end date for the exception. See Date Fields . Must be after startdate.
exception_type	exception_type	[Read-only] The type of exception. The only possible value is R – Date range of the exception.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The exception name or description, for example New Years Day.
schedule_request_itemid	schedule_request_item_id	The ID of the schedule change item from a schedule request.
startdate	startdate	[Required] The start date for the exception. See Date Fields . Must be before enddate.
timetypeid	timetype_id	The ID of the associated time type. The time type must be allowed in schedules.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	The ID of the user of this is an exception to the user's work schedule. 0 if this is an exception to an account work schedule.
workhours	workhours	The number of hours per day during this date range. This overrides any work hours on each day of either the account schedule or the account/user schedule.
workscheduleid	workschedule_id	The ID of the corresponding work schedule.

Usage Guidelines

Review the following guidelines:

- One of `userid` or `workscheduleid` is required. If the schedule exception is for a user work schedule, the value of one property is derived from the other.
- The date range defined by `startdate` and `enddate` must not overlap with the date range for any other schedule exceptions associated with the same work schedule, unless all the following conditions are met:
 - The **Automatically create schedule exceptions when a time-off request is approved** box is checked in the Timesheets application settings in OpenAir (Administration > Application Settings > Timesheets > Other Settings).
 - The `schedule_request_itemid` value is not empty or 0.
 - Your account is configured to allow multiple schedule exceptions on the same day.
- Changes to schedule exceptions trigger a recalculation of the associated work schedule and automatic changes to the [booking_by_day](#) table.

Schedulerequest

A time off request [Schedulerequest] is a change to an employee work schedule requested by the employee.

Review the [Usage Guidelines](#) for the Schedulerequest object.

—	XML	SOAP	REST	Database table
Object	Schedulerequest	oaSchedulerequest	—	schedule_request
Supported Commands	Add , Read , Modify , Delete Unapprove	add() , read() , modify() , upsert() , delete() unapprove()	—	—

The Schedulerequest object has the following standard properties:

 **Note:** Schedulerequest object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
approval_status	approval_status	A one-character string indicating the approval status of the schedule request. Possible values: <ul style="list-style-type: none"> ■ O - Open ■ P - Pending approval ■ A - Approved ■ R - Rejected
attachmentid	attachment_id	If non-zero, the attachment record associated with this schedule request.
categoryid	category_id	The ID of the associated category.
created	created	[Read-only] Time the record was created. See Date Fields .
customerid	customer_id	The ID of the associated customer.
date	date	The date of the schedule request creation. See Date Fields .
date_approved	date_approved	The date the schedule request was approved. See Date Fields .
date_submitted	date_submitted	The date the schedule request was submitted. See Date Fields .
description	description	The description or purpose for the schedule request.
enddate	enddate	The end date of the schedule request. See Date Fields .
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The name of the schedule request. Defaults to a concatenation of prefix and number unless set when adding a schedule request.
notes	notes	Notes to print on the schedule request.
number	number	The schedule request number that increments by 1. Assigned automatically unless set when adding a schedule request. Must be unique.

XML / SOAP	Database	Description
prefix	prefix	A static alphanumeric schedule request number prefix. Defaults to SR- unless set when adding a schedule request.
project_taskid	project_task_id	The ID of the associated project task.
projectid	project_id	The ID of the associated project.
startdate	startdate	The start date of the schedule request. See Date Fields .
timetype	—	The time type of this schedule request: R - regular time or P - personal time.
timetypeid	timetype_id	The ID of the associated time type.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	The ID of the user creating the schedule request. Defaults to the internal ID of the authenticated user if not set when adding a schedule request. If

Usage Guidelines

To add or modify a Schedulerequest object either the schedule request must be for the authenticated user or the authenticated user must be an account administrator.

You cannot delete a Schedulerequest object if this object is referenced by a [Schedulerequest_item](#) object. Delete any dependent objects first before you delete a Schedulerequest object.

Schedulerequest_item

A time off request item [[Schedulerequest_item](#)] is an element in a time off request (schedule request) covering distinct periods.

—	XML	SOAP	REST	Database table
Object	Schedulerequest_item	oaSchedulerequest_item	—	schedule_request_item
Supported Commands	Read , Modify , ModifyOnCondition , Delete	read() , modify() , delete()	—	—

The [Schedulerequest_item](#) object has the following properties:

XML / SOAP	Database	Description
categoryid	category_id	The ID of the associated category.
created	created	[Read-only] Time the record was created. See Date Fields .
customerid	customer_id	The ID of the associated customer.
date	date	The date of the schedule request item. See Date Fields .

XML / SOAP	Database	Description
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
hours	hours	The number of hours represented by this schedule request item.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The schedule request item name. It is the same as the schedule request description.
project_taskid	project_task_id	The ID of the associated project task.
projectid	project_id	The ID of the associated project.
request_reference_number	request_reference_number	Unique reference number within schedule request.
schedule_requestid	schedule_request_id	[Required] The ID of the associated schedule request.
timetypeid	timetype_id	The ID of the associated time type.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	The ID of the associated user.

Slip

A charge [Slip] is a customer billing entry, whether it is time, a fixed fee, an expense, or a product that is being billed. Charges are grouped into invoices.

Review the [Usage Guidelines](#) for the Slip object.

—	XML	SOAP	REST	Database table
Object	Slip	oaSlip	—	slip
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Slip object has the following standard properties:

 **Note:** Slip object properties may also include custom fields. The object type supports the custom equal to read method and the enable_custom read attribute.

XML / SOAP	Database	Description
acct_date	acct_date	The accounting period date of the slip. See Date Fields
agreementid	agreement_id	The ID of the associated agreement.

XML / SOAP	Database	Description
billing_contactid	–	[Read-only] The ID of the billing contact associated with the invoice this charge is part of.
category_<N>id where <N> is an integer from 1 to 5	category_<N>_id where <N> is an integer from 1 to 5	The ID of the associated category_<N>.
categoryid	category_id	The ID of the associated category. When set, the slip is based on this category.
city	city	The slip city or location.
cost	cost	The cost per unit of measure for an E or P slip, the billing rate for an O slip, or the fixed price for a F slip. Dated by the date field.
cost_centerid	cost_center_id	The ID of the associated cost center. If not set when adding an object, it set to the internal ID of the cost center for the time entry or receipt associated with this charge.
created	created	[Read-only] Time the record was created. See Date Fields
currency	currency	Currency for the money fields in the record
customerid	customer_id	The ID of the associated customer.
customerpo	customerpo_id	The ID of the associated customerpo.
date	date	[Required] The date of the billing slip. See Date Fields
decimal_hours	–	The number of decimal hours for a T slip.
description	description	The description of the billing slip.
gl_code	–	[Read-only] The fixed code 1234455454.
hour	hour	The number of hours for a T slip.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
invoiceid	invoice_id	The ID of the associated invoice once billed.
itemid	itemid	The ID of the associated item. If this is set, the slip is based on this item. Determine the subtype using the associated item type.
job_code_id	job_code_id	The ID of the associated job code.
minute	minute	The number of minutes for a T slip.
notes	notes	Notes associated with the slip.

XML / SOAP	Database	Description
originating_id	originating_id	For use with split slips feature. If set, the slip.id of the originating slip for this split portion.
payment_typeid	payment_type_id	The ID of the associated payment type.
payroll_typeid	payroll_type_id	The ID of the associated payroll type.
portfolio_projectid	portfolio_project_id	The ID of the associated portfolio project.
productid	product_id	The ID of the associated product.
project_billing_ruleid	-	[Read-only] The ID of the associated project billing rule.
projectid	project_id	The ID of the associated project.
projecttask_typeid	projecttask_type_id	The ID of the projecttask_type of the associated projecttask.
projecttaskid	projecttask_id	The ID of the task within the associated project.
quantity	quantity	The quantity for an E, O, or P slip.
rate	rate	The hourly rate for a T slip. Dated by the date field.
ref_slipid	ref_slipid	For credit/rebill, ID of the original slip ID.
shipping_contactid	shipping_contact_id	[Read-only] The ID of the shipping contact associated with the invoice this charge is part of.
skip_recognition	skip_recognition	A 1/0 field indicating if this record should be recognized. Used for split charges which were already recognized.
slip_stageid	slip_stage_id	The ID of the associated slip stage.
sold_to_contactid	-	[Read-only] The ID of the sold-to contact associated with the invoice this charge is part of.
tax_location_name	-	[Read-only] The name of the tax location
tax_rate_adjusted	-	[Read-only] A 1/0 field indicating if the tax amount for this charge was adjusted to ensure that the sum of tax amounts for every charges in the invoice match the total tax amount for the invoice.
tax_rateid	-	[Read-only] The ID of the tax rate
timer_start	timer_start	The starting time of the timer. See Date Fields
timetypeid	timetype_id	The ID of the associated time type.
total	total	The total value of the slip. Dated by the date field.

XML / SOAP	Database	Description
total_tax	total_tax_paid	[Read-only] The total tax paid. Dated by the date field.
total_with_tax	–	[Read-only] The total value of the slip including tax.
type	type	The type of the slip: <ul style="list-style-type: none"> ■ T - hourly rate slip ■ E - expense slip ■ F - flat price slip ■ O - other time slip ■ M - incomplete slip ■ P - product slip
unitm	um	The unit of measure for an E or P slip or the rate description for an O slip.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields
userid	user_id	[Required] The ID of the associated user.

Usage Guidelines

Review the following guidelines:

- When reading Slip objects, if the response does not include all expected objects, verify your company's OpenAir account configuration. The following account configuration settings, in particular, can modify the API behavior:
 - OpenAir API can be configured to convert all monetary values on charges into the invoice currency. In this case, only invoiced charges are returned.
 - OpenAir API can be configured to filter out charges associated with charge stages excluded from invoicing. In this case, only charges in charge stages that can be invoiced are returned

To review these and other configuration settings, contact your OpenAir Professional Services representative or OpenAir Customer Support.
- When adding Slip objects associated with an invoice:
 - The customerid if specified must be the internal ID of the customer associated with that invoice.
 - The customer associated with the invoice must be the same as the customer associated with the project with internal ID projectid, if specified.
- When modifying Slip objects, invoiceid is read-only. Charges cannot be moved from one invoice to another.
- Some properties are required depending on the type:
 - rate and either hours, minutes or decimal_hours for a type T slip.
 - cost for a type E, F, O, or P slip.
 - quantity and unitm for a type E, O, or P slip.

SlipProjection

A charge projection [SlipProjection] is a charge created from a billing projection run. The charge projection object includes many of the charge object properties and additional properties.

—	XML	SOAP	REST	Database table
Object	SlipProjection	oaSlipProjection	—	slip_projection , slip
Supported Commands	Read	read()	—	—

The SlipProjection object has the following properties:

XML / SOAP	Database	Description
acct_date	slip.acct_date	[Read-only] The associated accounting period date. See Date Fields
agreementid	slip.agreement_id	[Read-only] The ID of the associated agreement.
billing_contactid	—	[Read-only] The ID of the associated billing contact.
booking_typeid	slip_projection.booking_type_id	[Read-only] ID of the booking type if this was generated from bookings.
categoryid	slip.category_id	[Read-only] The ID of the associated category. When set, the slip is based on this category.
city	slip.city	[Read-only] The slip city or location.
cost	slip.cost	[Read-only] The cost per unit of measure for an E or P slip, the billing rate for an O slip, or the fixed price for a F slip. Dated by the date field.
cost_centerid	slip.cost_center_id	[Read-only] The ID of the associated cost center.
created	slip.created	[Read-only] Time the record was created. See Date Fields
currency	slip.currency	[Read-only] Currency for the money fields in the record
customerid	slip.customer_id	[Read-only] The ID of the associated customer.
customerpo	slip.customerpo_id	[Read-only] The ID of the associated customerpo.
date	slip.date	[Read-only] The date of the billing slip. See Date Fields
decimal_hours	—	[Read-only] The number of decimal hours for a T slip.

XML / SOAP	Database	Description
description	slip.description	[Read-only] The description of the billing slip.
hour	slip.hour	[Read-only] The number of hours for a T slip.
id	slip.id	[Read-only] Unique ID. Automatically assigned by OpenAir.
invoiceid	slip.invoice_id	[Read-only] The ID of the associated invoice once billed.
itemid	slip.item_id	[Read-only] The ID of the associated item. If this is set, the slip is based on this item. Determine the subtype using the associated item type.
job_codeid	slip.job_code_id	[Read-only] The ID of the associated job code.
minute	slip.minute	[Read-only] The number of minutes for a T slip.
notes	slip.notes	[Read-only] Notes associated with the slip.
payment_typeid	slip.payment_type_id	[Read-only] The ID of the associated payment type.
productid	slip.product_id	[Read-only] The ID of the associated product.
project_billing_ruleid	slip_projection.project_billing_rule_id	[Read-only] The ID of the associated project billing rule.
projectid	slip.project_id	[Read-only] The ID of the associated project.
projecttask_typeid	slip.projecttask_type_id	[Read-only] The ID of the associated project task type.
projecttaskid	slip.projecttask_id	[Read-only] The ID of the task within the associated project.
quantity	slip.quantity	[Read-only] The quantity for an E, O, or P slip.
rate	slip.rate	[Read-only] The hourly rate for a T slip. Dated by the date field.
shipping_contactid	slip.shipping_contact_id	[Read-only] The ID of the associated shipping contact.
slip_projection_type	slip_projection.slip_projection_type	[Read-only] The type of the slip projection: <ul style="list-style-type: none"> ■ X - slip projection generated from billing rule ■ B - Time from potentially billable transaction which did not match any billing rule ■ N - Time from transaction with non-billable project-task ■ P - Time from transaction matching a billing rule but is Partially over cap

XML / SOAP	Database	Description
		<ul style="list-style-type: none"> ■ S - Time from transaction matching a billing rule but is completely over cap and rule indicates to Stop if capped ■ C - Time from transaction matching a billing rule but is completely over cap and no more rules match.
slip_stageid	slip .slip_stage_id	[Read-only] The ID of the associated slip stage.
sold_to_contactid	–	[Read-only] The ID of the contact sold to.
timer_start	slip .timer_start	[Read-only] The starting time of the timer. See Date Fields
timetypeid	slip .timetype_id	[Read-only] The ID of the associated time type.
total	slip .total	[Read-only] The total value of the slip. Dated by the date field.
transactionid	slip_projection .transaction_id	[Read-only] For internal use only.
type	slip .type	[Read-only] The type of the slip: <ul style="list-style-type: none"> ■ T - hourly rate slip ■ E - expense slip ■ F - flat price slip ■ O - other time slip ■ M - incomplete slip ■ P - product slip
unitm	slip .um	[Read-only] The unit of measure for an E or P slip or the rate description for an O slip.
updated	slip .updated	[Read-only] Time the record was last updated or modified. See Date Fields
userid	slip .user_id	[Read-only] The ID of the associated user.

Slipstage

A charge stage [Slipstage] can be used to record phase or status information for a charge. By default, charges can either be in Open or Billed stage. Additional charge stages can be created for the account. For more information, see the help topic [Charge Stages](#).

—	XML	SOAP	REST	Database table
Object	Slipstage	oaSlipstage	—	slip_stage
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Slipstage object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields
enable_slip_tab	enable_slip_tab	Display slips of this stage in a separate tab.
exclude_from_invoicing	exclude_from_invoicing	Exclude slips of this stage from invoicing.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the stage.
notes	notes	Notes associated with this slip stage.
position	position	The position of the stage.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields

SummaryView

A summary view [SummaryView] is a summary of the number of expense reports and timesheets with each approval status.

—	XML	SOAP	REST	Database table
Object	SummaryView	oaSummaryView	—	—
Supported Commands	Read (all)	read() (all)	—	—

 **Note:** The SummaryView object supports the all read method only.

The SummaryView object has the following properties:

XML / SOAP	Database	Description
en_approved	—	[Read-only] Count of approved envelopes
en_open	—	[Read-only] Count of open envelopes
en_rejected	—	[Read-only] Count of rejected envelopes
en_submitted	—	[Read-only] Count of submitted envelopes
en_waiting	—	[Read-only] Count of waiting envelopes to approve
tm_approved	—	[Read-only] Count of approved timesheets
tm_open	—	[Read-only] Count of open timesheets
tm_rejected	—	[Read-only] Count of rejected timesheets
tm_submitted	—	[Read-only] Count of submitted timesheets
tm_waiting	—	[Read-only] Count of timesheets to approve

TagGroup

A tag group [TagGroup] is a tag type. Tags are a custom classification tool that allows to organize users, customers, or projects into groups for reporting purposes.

—	XML	SOAP	REST	Database table
Object	TagGroup	oaTagGroup	—	tag_group
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The TagGroup object has the following properties:

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether the record is active.
created	created	[Read-only] Time the record was created. See Date Fields .
entity_type	entity_type	The tag group type: U - user, C - customer, or P - project.
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	Name of the tag group.
searchable	searchable	A 1/0 field indicating whether this tag group is searchable. Used only for group type = U.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

TagGroupAttribute

A tag group attribute [TagGroupAttribute] is a possible tag value that can be associated a user, customer, or project.

—	XML	SOAP	REST	Database table
Object	TagGroupAttribute	oaTagGroupAttribute	—	tag_group_attribute
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The TagGroupAttribute object has the following properties

XML / SOAP	Database	Description
active	active	A 1/0 field indicating whether the record is active.
created	created	[Read-only] Time the record was created. See Date Fields .
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.

XML / SOAP	Database	Description
name	name	Name of the tag group attribute.
tag_groupid	tag_groupid	The ID of the tag group this attribute is in.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

TargetUtilization

A target utilization [TargetUtilization] is the utilization expected from an employee for capacity planning purposes.

—	XML	SOAP	REST	Database table
Object	TargetUtilization	oaTargetUtilization	—	target_utilization
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The TargetUtilization object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
end_date	end_date	The end date for the target utilization. This field is automatically determined based on the next subsequently later start date row for the user. This field can be 0000-00-00 for one row which represents the unbounded value. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
percentage	percentage	Target utilization for this user as a percentage. For example, 75.30.
start_date	start_date	The start date for the target utilization. See Date Fields .
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
user_id	user_id	[Required] The ID of the associated user.

Task

A time entry [Task] is a time slot worked by an employee on a work package.

Review the [Usage Guidelines](#) for the Task object.

—	XML	SOAP	REST	Database table
Object	Task	oaTask	TimeEntry	task
Supported Commands	Add , Read , Modify , Delete , Reject	add() , read() , modify() , upsert() , delete() , reject()	See the help topic Time Entries	—

The Task object has the following standard properties:

Note: Task object properties may also include custom fields. The object type supports the custom equal to read method and the enable_custom read attribute.

XML / SOAP	REST	Database	Description
acct_date	accountingDate	acct_date	The accounting period date of the time entry. See Date Fields
categoryid	categoryId	category_id	The internal ID of the associated service (category).
category_<N>id where <N> is an integer from 1 to 5	category<N>Id where <N> is an integer from 1 to 5	category_<N>id where <N> is an integer from 1 to 5	The internal ID of the associated service line <N> [category_<N>].
cost_centerid	costCenterId	cost_center_id	The internal ID of the associated cost center.
created	created	created	[Read-only] The date the time entry was created. See Date Fields
customerid	customerId	customer_id	The internal ID of the associated customer. Determined automatically from projectId if not set when adding a time entry.
date	date	date	[Required] The date of the time entry. See Date Fields
decimal_hours	decimalHours	—	The number of decimal hours for the time entry. If the Use Days Instead of Hours for All Time Entries feature is not enabled for your account: <ul style="list-style-type: none"> decimal_hours accepts decimal values and the decimal part is converted to minutes. For example, decimal_hours = "5.5" is equivalent to hours = "5" and minutes = "30". The integer part of decimal_hours is ignored if a value is set for hours. The decimal part of decimal_hours is ignored if a value is set for minutes. If values are set for both decimal_hours and hours but not for minutes, the decimal parts of decimal_hours and hours are added and converted to minutes. For example, decimal_hours = "5.5" and hours = "2.1" is equivalent to hours = "2" and minutes = "36". See also Usage Guidelines .
description	description	description	The description of the time entry.
end_time	endTime	end_time	The time entry end time.
—	exported	exported	Date and time the time entry was marked as "exported".
hours	hour	hour	The number of hours for the time entry.

XML / SOAP	REST	Database	Description
			<p>If the Use Days Instead of Hours for All Time Entries feature is not enabled for your account:</p> <ul style="list-style-type: none"> hours accepts decimal values and the decimal part is converted to minutes and added to the minutes value to obtain the total number of minutes for the time entry. The total number of minutes for the time entry is $(\{hours\} \times 60 \div 100) + minutes$, where $\{hours\}$ is the decimal part of hours. For example, hours = "5.5" and minutes = "6" is equivalent to hours = "5" and minutes = "36". If values are set for both decimal_hours and hours but not for minutes, the decimal parts of decimal_hours and hours are added and converted to minutes. For example, decimal_hours = "5.5" and hours = "2.1" is equivalent to hours = "2" and minutes = "36". <p>See also Usage Guidelines.</p>
—	hoursRemaining	—	The number of hours remaining of the associated project task.
id	id	id	[Read-only] The unique internal identifier of the time entry . Assigned by OpenAir.
job_codeid	jobCodeId	job_code_id	The internal ID of the associated job code.
loaded_cost	—	loaded_cost . cost	The loaded cost for the associated resource, using the forex future rate from the exchange rate table.
loaded_cost_2	—	loaded_cost . cost	User's second level loaded cost, using the forex future rate from the exchange rate table.
loaded_cost_3	—	loaded_cost . cost	User's third level loaded cost, using the forex future rate from the exchange rate table.
minutes	minute	minute	<p>The number of minutes for the time entry.</p> <div style="border: 1px solid orange; padding: 5px; background-color: #fff9c4;">  Important: minutes does not accept values with a decimal part. </div>
notes	notes	notes	Notes about the time entry.
payroll_typeid	payrollTypeId	payroll_type_id	The internal ID of the associated payroll type.
project_loaded_cost	—	loaded_cost . cost	User's project cost override in project currency. Uses the future rate from the exchange rate table.
project_loaded_cost_2	—	loaded_cost . cost	User's project second cost in project currency. Uses the future rate from the exchange rate table.
project_loaded_cost_3	—	loaded_cost . cost	User's project third cost in project currency. Uses the future rate from the exchange rate table.
projectid	projectId	project_id	The internal ID of the associated project.

XML / SOAP	REST	Database	Description
projecttask_typeid	projectTaskId	projecttask_type_id	The internal ID of the project task type of the associated project task.
projecttaskid	projectTaskTypeId	projecttask_id	The internal ID of the task within the associated project.
—	scheduleRequestItemId	schedule_request_item_id	The internal ID of the schedule change item from a schedule request.
slipid	slipId	slipid	The internal ID of the associated slip if this task was billed.
start_time	startTime	start_time	The time entry start time.
thin_client_id	thinClientId	thin_client_id	Used by thin clients to reconcile imported records.
timesheetid	timesheetId	timesheet_id	The internal ID of the associated timesheet. Set automatically to the internal ID matching the time entry date, if one exists, or that of a new timesheet (created automatically), otherwise, when adding a time entry. Cannot be modified.
timetypeid	timeTypeId	timetype_id	The internal ID of the associated time type.
updated	updated	updated	[Read-only] The date the time entry was last updated or modified.
userid	userId	user_id	[Required] The internal ID of the associated employee.

Usage Guidelines

Review the following guidelines:

- The **Require a task on time entries** setting for the Timesheets application (Administration > Application Settings > Timesheets > Other settings) is not supported. OpenAir API lets you add or modify time entries without an associated projecttask (projecttaskid) even if a task is required on time entries in the OpenAir UI.
- If a projecttaskid is specified and time entry is restricted to the date range covered by that project task, OpenAir API returns an error (error code 888) if the time entry date is outside the project task date range.
- By default, the loaded cost [loaded_cost] and project loaded cost override [project_loaded_cost] in the time entry [Task] object use the future currency exchange rate from the foreign exchange conversion table [forex]. To force these values to use the exchange rate at the date of the time entry instead of the future exchange rate, contact OpenAir Customer Support and request the following account configuration setting to be enabled: API to Respect Time Entry Date for Currency Conversion in Loaded Costs.
- You cannot modify a time entry if the authenticated user cannot modify the associated timesheet, or if the Signers feature is enabled and the time entry has been accepted (signed off).
- You can enter decimal values for the number of hours [hours] if the Use Days Instead of Hours for All Time Entries feature is not enabled for your account. See [Decimal time entry \(hours\)](#).
- You can read and modify the start time [start_time] and end time [end_time] for time entries using the OpenAir XML API or SOAP API. To modify start_time and [end_time], start and end time entry on timesheets must be enabled for your account. See [Modifying the Time Entry Start and End Times](#).



Important: You should not use **Enable start and end time entry on timesheets** and **Use Days Instead of Hours for All Time Entries** in conjunction.

When **Enable start and end time entry on timesheets** is enabled and a user enters a start time and end time in OpenAir, the duration is calculated in hours and not converted to days.

If both features are enabled, OpenAir API returns an error (error code 1407) if you set both `decimal_hours` and `minutes` but not `hours` in the request.

Modifying the Time Entry Start and End Times

You can read and modify the start time [`start_time`] and end time [`end_time`] for time entries using the OpenAir XML API or SOAP API.

Review the following requirements to modify `start_time` and `end_time`:

- Start and end time entry on timesheets must be enabled. The **Enable start and end time entry on timesheets** box must be checked on the Administration > Application Settings > Timesheets > Other settings form. OpenAir API returns an error (error code: 1406) if you attempt to edit the start or end times and the functionality is not enabled.
- The format for `start_time` and `end_time` must be hh:mm:ss. OpenAir API returns an error (error code: 1404) if either properties are set to an invalid value or a value with the wrong format.

The following examples are valid values: 10:30:15, 2:30 (equivalent to 02:30:00), 2:30:15 (equivalent to 02:30:15), 2:3 (equivalent to 02:03:00), 2:3:4 (equivalent to 02:03:04).

- The `start_time` value must be before the `end_time` value. The API returns an error (error code: 1405) if an invalid time range is passed.
- When setting `start_time` and `end_time`, you must also set the duration using either `decimal_hours` or `hours` and `minutes`.



Note: The duration is not calculated when you set `start_time` and `end_time`. However, the duration is validated. The API returns an error (error code: 1407) if the duration does not match the period between `start_time` and `end_time`.

- To clear the `start_time` or `end_time`, set it to 00:00:00. Setting both `start_time` and `end_time` to 00:00:00 sets `decimal_hours`, `hours` and `minutes` to 0 automatically.

Decimal time entry (hours)

Decimal time entry for the number of hours is supported if the feature **Use Days Instead of Hours for All Time Entries** is disabled for your account:

- `hours` accepts decimal part and the decimal part is converted to minutes. For example, if `task.hours = "5.5"`; is equivalent to `task.hours = "5"`; `task.minutes = "30"`;
- Minutes passed as the decimal part of hours and minutes are added. For example, `task.hours = "5.5"`; `task.minutes = "6"`; is equivalent to `task.hours = "5"`; `task.minutes = "36"`;
- `decimal_hours` accepts decimal part and the decimal part is converted to minutes. For example, `task.decimal_hours = "5.5"`; is equivalent to `task.hours = "5"`; `task.minutes = "30"`;

- Minutes passed as the decimal part of decimal_hours are ignored if minutes is also passed. For example, task.decimal_hours = "5.5"; task.minutes = "6"; is equivalent to task.hours = "5"; task.minutes = "36";.
- If both decimal_hours and hours are passed, the integer part of decimal_hours is ignored and only the integer part of hours is used. However, the decimal parts of decimal_hours and hours are added. For example, task.decimal_hours = "5.5"; task.hours = "2.1"; is equivalent to task.hours = "2"; task.minutes = "36";.
- If decimal_hours, hours and minutes are passed, both the decimal and integer parts of decimal_hours are ignored. Minutes passed as the decimal part of hours and minutes are added. For example, task.decimal_hours = "5.5"; task.hours = "2.1"; task.minutes = "20"; is equivalent to task.hours = "2"; task.minutes = "26";.

TaskAdjustment

A time entry adjustment [TaskAdjustment] is a change made to a time entry using an adjustment timesheet after the original timesheet was approved.

—	XML	SOAP	REST	Database table
Object	TaskAdjustment	oaTaskAdjustment	—	task_adjustment
Supported Commands	Read	read()	—	—

The TaskAdjustment object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by the system.
new_taskid	new_task_id	[Read-only] The ID of the adjustment task.
new_timesheetid	new_timesheet_id	[Read-only] The ID of the adjustment timesheet.
old_taskid	old_task_id	[Read-only] The ID of the original task.
old_timesheetid	old_timesheet_id	[Read-only] The ID of the original timesheet.
updated	updated	[Read-only] Time the record was updated. See Date Fields .

TaskTimecard

Use the time entry – time card link [TaskTimecard] object to read time entries associated with time cards.

—	XML	SOAP	REST	Database table
Object	TaskTimecard	oaTaskTimecard	—	—
Supported Commands	Read	read()	—	—

The TaskTimecard object has the following properties:

XML / SOAP	Database	Description
category_<N>id where <N> is an integer from 1 to 5	category_<N>_id where <N> is an integer from 1 to 5	[Read-only] The ID of the associated category_<N>.
categoryid	category_id	[Read-only] The ID of the associated category.
cost_centerid	cost_center_id	[Read-only] The ID of the associated cost center.
created	created	[Read-only] Time the record was created. See Date Fields .
customerid	customer_id	[Read-only] The ID of the associated customer.
date	date	[Read-only] The date of the task timecard. See Date Fields .
decimal_hours	decimal_hours	[Read-only] The number of decimal hours for the task timecard.
description	description	[Read-only] The description of the task timecard.
hours	hours	[Read-only] The number of hours for the task timecard.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
minutes	minutes	[Read-only] The number of minutes for the task timecard.
notes	notes	[Read-only] Notes associated with this task timecard.
payroll_typeid	payroll_type_id	[Read-only] The ID of the associated payroll type.
project_phaseid	project_phase_id	[Read-only] The ID of the project phase.
projectid	project_id	[Read-only] The ID of the associated project.
projecttask_typeid	projecttask_type_id	[Read-only] The ID of the project task type.
projecttaskid	projecttask_id	[Read-only] The ID of the task within the associated project.
slipid	slipid	[Read-only] The ID of the associated slip.
time_cardid	time_cardid	[Read-only] The ID of the associated timecard.
timesheetid	timesheetid	[Read-only] The ID of the associated timesheet.
timetypeid	timetypeid	[Read-only] The ID of the associated time type.

XML / SOAP	Database	Description
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	userid	[Read-only] The ID of the associated user.

TaxLocation

A TaxLocation is a definition of tax rates applicable in a region.

—	XML	SOAP	REST	Database table
Object	TaxLocation	oaTaxLocation	—	tax_location
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The TaxLocation object has the following standard properties:

Note: TaxLocation object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
acct_code_federal	acct_code_federal	GL accounting code for the federal entries.
acct_code_gst	acct_code_gst	GL accounting code for the GST entries.
acct_code_hst	acct_code_hst	GL accounting code for the HST entries.
acct_code_pst	acct_code_pst	GL accounting code for the PST entries.
acct_code_state	acct_code_state	GL accounting code for the state entries.
active	active	A 1/0 field specifying if the location is active.
created	created	[Read-only] Time the record was created. See Date Fields .
federal_rate	federal_rate	The federal tax rate.
gst_rate	gst_rate	The GST rate.
hst_rate	hst_rate	The HST rate. This is used instead of GST and PST in some locations.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The name for the estimate adjustment.
notes	notes	Notes associated with this tax location.
pst_rate	pst_rate	The PST rate.
state_rate	state_rate	The state tax rate.
tax_method	tax_method	The tax method:

XML / SOAP	Database	Description
		<ul style="list-style-type: none"> ■ G - GST and PST ■ H - HST ■ F - Federal and State
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

TaxRate

A tax rate [TaxRate] is the tax rate applied to a transaction according to the tax location information at the date of the transaction.

—	XML	SOAP	REST	Database table
Object	TaxRate	oaTaxRate	—	tax_rate
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The TaxRate object has the following properties:

XML / SOAP	XML / SOAP	Description
adjusted	adjusted	A "1/0" field indicating if the tax amounts were adjusted to match the invoice taxes
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
date	date	The date (used for currency conversions). See Date Fields .
federal	federal	The federal tax. Dated by the date field.
gst	gst	The GST tax. Dated by the date field.
hst	hst	The HST tax. Dated by the date field.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
manual	manual	A "1/0" field indicating if the tax amounts were manually entered rather than calculated
notes	—	Notes associated with this tax rate.
pst	pst	The PST tax. Dated by the date field.
purchase_itemid	purchase_item_id	The ID of the associated purchase order item.
slipid	slip_id	The ID of the associated slip.
state	state	The state tax. Dated by the date field.
tax_locationid	tax_location_id	The ID of the associated tax location.
ticketid	ticket_id	The ID of the associated ticket.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Term

A term [Term] is a default or custom label for a specific concept in OpenAir.

Use the term object to read the custom terminology used in the company.

The OpenAir XML API and SOAP use the term object as a substructure to pass company or user setting information when reading the [Company](#) record.

—	XML	SOAP	REST	Database table
Object	Term	oaTerm	—	—
Supported Commands	Read (all, equal to)	read() (all, equal to)	—	—

 **Note:** The Term object supports the `all` and `equal` to read methods only.

The Term object has the following properties:

XML / SOAP	Database	Description
display	display	[Read-only] Display the term as.
name	name	[Read-only] The name for the term.

Ticket

A receipt [Ticket] is an expense items that contains information about cost incurred by an employee and collected in an expense report.

Review the [Usage Guidelines](#) for the Ticket object.

—	XML	SOAP	REST	Database table
Object	Ticket	oaTicket	Receipt	ticket
Supported Commands	Add , Read , Modify , Delete , Reject	add() , read() , modify() , upsert() , delete() , reject()	—	—

The Ticket object has the following standard properties:

 **Note:** Ticket object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML/SOAP	REST	Database	Description
acct_date	accountingDate	acct_date	The accounting period date of the receipt. See Date Fields
attachmentid	attachments	attachmentid	The attachments associated with this expense report. Array (REST) or comma-delimited list (SOAP, XML) of internal IDs for attachment objects.
categoryid	—	—	The internal ID of the associated category.
city	receiptLocation	city	The city or location where the cost was incurred.

XML/SOAP	REST	Database	Description
cost	costPerUnit	cost	The cost per unit of measure. Dated by the date field.
cost_centerid	costCenterId	cost_centerid	The internal ID of the cost center associated with the receipt.
created	created	created	[Read-only] The date the receipt was created. See Date Fields
currency	currency	currency	The currency for monetary values in the receipt record. Three-letter currency code.
currency_cost	foreignCurrencyCost	currency_cost	The cost per unit of measure in the selected foreign currency, if this is a foreign currency receipt.
currency_exchange_intolerance	isForeignCurrencyExchangeIntolerance	currency_exchange_intolerance	A 1/0 field indicating if the record is within the specified foreign currency tolerance.
currency_rate	foreignCurrencyRate	currency_rate	[Read-only] The foreign currency conversion rate, if this is a foreign currency receipt.
currency_symbol	foreignCurrencySymbol	currency_symbol	The foreign currency, if this is a foreign currency receipt. Three-letter currency code.
currency_total_tax_paid	foreignCurrencyTotalTaxPaid	currency_total_tax_paid	The tax paid in the foreign currency, if this is a foreign currency receipt.
customerid	customerId	customerid	The internal ID of the customer associated with the receipt. Required if the Require a customer selection on receipts box is checked in the Expenses application settings in OpenAir (Administration > Application Settings > Expenses > Other Settings).
date	date	date	[Required] The date of the receipt. See Date Fields
description	description	description	The description of the receipt.
envelopeid	expenseReportId	envelopeid	[Required] The internal ID of the expense report associated with the receipt.
externalid	externalId	externalid	The unique external ID of the receipt, if the record was imported from an external system.
—	federalTax	—	The total federal tax for the receipt.
—	federalTaxRate	—	The federal tax rate for the receipt.
—	gst	—	The total GST tax for the receipt.
—	gstRate	—	The GST tax rate for the receipt.
—	hst	—	The total HST tax for the receipt.
—	hstRate	—	The HST tax rate for the receipt.

XML/SOAP	REST	Database	Description
id	id	id	[Read-only] The unique internal identifier of the receipt. Assigned by OpenAir.
– See status–	isReimbursable	– See status–	A 1/0 field indicating if the receipt is reimbursable.
—	isTaxIncludedInCost	—	A 1/0 field indicating if the cost includes the tax.
itemid	itemId	itemid	The internal ID of the item associated with the receipt. The type of item can be used to determine the receipt subtype.
missing_receipt	isMissingPaperReceipt	missing_receipt	A 1/0 field indicating if the paper receipt is missing for this receipt.
non_billable	isNonBillable	non_billable	A 1/0 field indicating if the receipt is not billable.
notes	notes	notes	Notes about the receipt.
payment_typeid	paymentTypeId	payment_typeid	The internal ID of the payment type associated with the receipt. The payment type indicates how the payment was made and may determine if the receipt is reimbursable. Can be derived automatically from paymethod.
paymethod	—	payment_method	[Deprecated] The payment method. Used for backward compatibility only. Use payment_typeid instead. Can be derived automatically from payment_typeid. See Paymenttype .
project_taskid	projectTaskId	project_taskid	The internal ID of the project task associated with the receipt. Requires an option to be enabled in OpenAir (Administration > Application Settings > Expenses > Other settings).
projectid	projectId	projectid	The internal ID of the project associated with the receipt. Required if the Require a customer selection on receipts box is checked in the Expenses application settings in OpenAir (Administration > Application Settings > Expenses > Other Settings).
projecttask_typeid	—	projecttask_typeid	The international ID of project task type of the associated project task. Requires an option to be enabled in OpenAir (Administration > Application Settings > Expenses > Other settings). Automatically set to an empty value if project_taskid is set to an empty value.
—	pst	—	The total PST tax for the receipt.
—	pstRate	—	The PST tax rate for the receipt.
quantity	quantity	quantity	The quantity (number of units of measure).
reference_number	trackingNumber	reference_number	The unique reference number of the receipt within the associated expense report. This attribute is used to cross-

XML/SOAP	REST	Database	Description
			reference digital receipts with paper receipts.
slipid	slipId	slipid	[Read-only] The internal ID of the charge (slip) associated with the receipt, if the expense was billed.
—	stateTax	—	The total state tax for the receipt.
—	stateTaxRate	—	The state tax rate for the receipt.
status	<i>See isReimbursable</i>	status	The status of the ticket: <ul style="list-style-type: none"> ■ R – reimbursable ■ N – non-reimbursable Can be derived automatically from payment_typeid.
tax_location_id	taxLocationId	—	The internal ID of the tax location associated with the receipt.
tax_location_name	—	—	The name of the tax location associated with the receipt.
tax_rateid	—	tax_rateid	The internal ID of the associated tax rate.
thin_client_id	—	thin_client_id	Used by thin clients to reconcile imported records.
total	total	total	The total value of the receipt. Dated by the date field.
total_no_tax	totalNoTax	—	The total value of the receipt excluding tax. Dated by the date field.
total_tax_paid	totalTaxPaid	total_tax_paid	The total tax paid. Dated by the date field.
unitm	—	um	The unit of measure.
updated	updated	updated	[Read-only] The date the receipt was last updated or modified. See Date Fields
use_server_currency_rate	—	—	[Read-only] A 1/0 field indicating whether currency_rate should be set to the foreign currency conversion rate for the company's OpenAir account even if a currency_rate value is specified for the receipt. Always 1.
userid	userId	userid	The internal ID of the employee associated with the receipt.
user_locationid	userLocationId	user_locationid	The internal ID of the employee location associated with the receipt.
—	vehicleId	The internal ID of the vehicle associated with the receipt.	The internal ID of the vehicle associated with the receipt.
vendorid	vendorId	vendorid	The internal ID of the vendor associated with the receipt.

Usage Guidelines

Review the following guidelines:

- The **Require a task selection on receipts** application setting (Administration > Application Settings > Expenses > Other settings) is not supported. OpenAir API lets you add or modify receipts without an associated task (project_taskid) even if a task is required on receipts in the OpenAir UI.
- You cannot modify a receipt [Ticket] if the authenticated user cannot modify the associated expense report [Envelope], or if the Signers feature is enabled and the receipt [Ticket] has been accepted (signed off).
- The following account configuration settings may impact the ability to add or modify ticket records using the API. To enable or disable any of these configuration settings, contact OpenAir Customer Support. See the help topic [Creating a Support Case](#).
 - **Do not allow editing of receipts with an American Express transaction number** — When this option is enabled, you cannot modify the fields date, quantity, cost, currency, payment_typeid, or total for any tickets created using the American Express receipt import wizard. If editing is necessary, you can request for the switch to be temporarily disabled.
 - **Do not allow receipt quantity to be set to zero using API** — By default, the API allows you to add or modify a ticket and set quantity to zero. If the switch is enabled, you cannot add or modify a ticket and set quantity to zero and the API returns an error (1412 Invalid quantity: Quantity must be non-zero number).
 - **Allow the adjustment of the receipt total instead of total always being a calculated based on quantity and cost** — When enabled, you can set a total value and this total value will be saved as is. By default the total value is calculated and cannot be modified.

Timecard

A time card [Timecard] is a time entry with information about start and end time of the working day and start and end time of a break period during the working day.

—	XML	SOAP	REST	Database table
Object	Timecard	oaTimecard	—	time_card
Supported Commands	Read	read()	—	—

The Timecard object has the following properties:

XML / SOAP	Database	Description
break_end	break_end	[Read-only] Time they ended the break. See Date Fields .
break_start	break_start	[Read-only] Time they started the break. See Date Fields .
created	created	[Read-only] Time the record was created. See Date Fields .
date	date	[Read-only] The date of the timecard. See Date Fields .
hours	hours	[Read-only] Hours worked.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
notes	notes	[Read-only] Notes associated with the timecard.
time_end	time_end	[Read-only] Time they stopped working. See Date Fields .

XML / SOAP	Database	Description
time_start	time_start	[Read-only] Time they started working. See Date Fields .
timesheetid	timesheet_id	[Read-only] The ID of the associated timesheet.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Read-only] The ID of the associated user.

Timesheet

A timesheet [Timesheet] is a collection of time entries submitted by an employee to claim compensation for time worked.

Review the [Usage Guidelines](#) for the Timesheet object.

—	XML	SOAP	REST	Database table
Object	Timesheet	oaTimesheet	—	timesheet
Supported Commands	Add , Read , Modify , Delete , Submit , Approve , Reject , Unapprove , Report	add() , createUser() , read() , modify() , upsert() , delete() , submit() , approve() , reject() , unapprove()	—	—

The Timesheet object has the following standard properties:

Note: Timesheet object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
acct_date	acct_date	The accounting period date of the task. See Date Fields .
approved	date_approved	The date the timesheet was approved. See Date Fields .
approved_by	—	Empty value kept for backward compatibility.
associated_tm_id	associated_tm_id	Id of complementary timesheet in case of cross month timesheet. Used only when the value of <code>start_end_month_ts</code> is not empty. The associated split timesheet must have the same <code>userid</code> , <code>starts</code> and <code>ends</code> and the counterpart value for <code>start_end_month_ts</code> .
created	created	[Read-only] Time the record was created. See Date Fields .
default_categoryid	default_category	The default category ID this timesheet is associated with. All new task entries get this default value.
default_customerid	default_customer	The default customer ID this timesheet is associated with. All new task entries get this default value.
default_payrolltypeid	default_payrolltype	The default payroll type ID this timesheet is associated with.
default_per_row	default_per_row	Holds a data structure of per row defaults. The format is as follows: Multiple CSV rows with each row having the element name ('cp','category' etc.) as the first record and then the ID values per row.

XML / SOAP	Database	Description
default_projectid	default_project	The default project ID this timesheet is associated with.
default_projecttaskid	default_task	The default task ID this timesheet is associated with. All new task entries get this default value.
default_timetypeid	default_timetype	The default time type ID this timesheet is associated with.
duration	duration	The duration of the timesheet: <ul style="list-style-type: none"> ■ W - Weekly ■ D - Daily ■ M - Monthly ■ B - Bi-weekly ■ S - Semi-monthly
ends	date_end	[Required] The ending date of the timesheet. See Date Fields . Must be after starts.
history	history	[Read-only] History of events that occurred to the TimeSheet.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
max_hours	—	[Read-only] Calculated maximum number of hours allowed on the timesheet as determined by the corresponding timesheet rule. A value is returned only if the rule is active and the attribute calculate_hours is set to 1 in the Read request.
min_hours	—	[Read-only] Calculated minimum number of hours required on the timesheet as determined by the corresponding timesheet rule. Supports the Read method only. A value is returned only if the rule is active and the attribute calculate_hours is set to 1 in the Read request.
name	name	The name of the timesheet.
notes	notes	Notes related to this timesheet.
start_end_month_ts	start_end_month_ts	Indicator of whether the cross_month timesheet belongs to the end of the current month or beginning of the next month'E' - belongs to the end of current month, first date in timesheet'S' - belongs to the start of next month, last date in timesheet. Must be used only for split timesheets at month end.
starts	date_start	[Required] The starting date of the timesheet. See Date Fields . Must be before ends.
status	status	The status of the timesheet: <ul style="list-style-type: none"> ■ O - Open (default) ■ S - Submitted ■ A - Approved ■ R - Rejected ■ X - Archived

XML / SOAP	Database	Description
submitted	date_submitted	The date the timesheet was submitted. See Date Fields .
thin_client_id	thin_client_id	Used by thin clients to reconcile imported records.
total	total	[Read-only] The total number of hours in the timesheet.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	userid	[Required] The ID of the associated user.

Usage Guidelines

Review the following notes and guidelines:

- Your account can be configured to:
 - Allow the modification of approved and archived timesheets using the OpenAir API.
 - Disallow the modification of submitted timesheet by the timesheet owner.
 - Disallow the modification of exported timesheets.

To enable any of the above features, contact OpenAir Customer Support.
- To modify another user's timesheet, the authenticated user must be an account administrator
- Set the attribute `calculate_hours` to 1 to return the minimum number of hours required [`min_hours`] and maximum number of hours allowed [`max_hours`] on the timesheet if an active timesheet rule set these values as a fixed number of hours or as a percentage of the work schedule. For more information about timesheet rules, see the help topic [Timesheet Rules](#).



Important: Using the attribute `calculate_hours` may slow the response time significantly, particularly when there are active timesheet rules are active and the minimum and maximum number of hours are set as a percentage of the work schedule.

- You cannot delete a Timesheet object if this object is referenced by a [Task](#) object. Delete any dependent objects first before you delete a Timesheet object.

Timetype

A time type [Timetype] is a classification grouping for time entries. Examples of time types can include "regular time", "overtime", "sick time", for example.

Review the [Usage Guidelines](#) for the Timetype object.

—	XML	SOAP	REST	Database table
Object	Timetype	oaTimetype	—	time_type
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Timetype object has the following standard properties:

Note: Timetype object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom` read attribute.

XML / SOAP	XML / SOAP	Description
active	active	A 1/0 field indicating whether this is time type is active.
code	code	Optional accounting system code for integration with external accounting systems.
cost_centerid	cost_center_id	The ID of the associated cost center.
created	created	[Read-only] Time the record was created. See Date Fields .
externalid	external_id	If the record was imported from an external system, you store the unique external record ID here.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	The name of the time type.
notes	notes	Notes associated with this time type.
payroll_code	payroll_code	The payroll code for this time type.
picklist_label	—	Label as shown on form picklist.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

Usage Guidelines

You cannot delete a Timetype object if this object is referenced by a [Task](#) object. Delete any dependent objects first before you delete a Timetype object.

Todo

A to do [Todo] is an action item that needs to be done as part of work on a pipeline deal.

—	XML	SOAP	REST	Database table
Object	Todo	oaTodo	—	todo
Supported Commands	Read	read()	—	—

The Todo object has the following properties:

XML / SOAP	Database	Description
contactid	contact_id	[Read-only] The ID of the associated contact.
created	created	[Read-only] Time the record was created. See Date Fields .
createdbyid	created_by_id	[Read-only] The ID of the user who created the todo item.
customerid	customer_id	[Read-only] The ID of the associated customer.
dealid	deal_id	[Read-only] The ID of the associated deal.

XML / SOAP	Database	Description
due	due	[Read-only] Date and time the task is due. See Date Fields .
finished	finished	[Read-only] Date and time the task was finished. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Read-only] The name or description of the todo item.
notes	notes	[Read-only] Notes associated with the todo item.
priority	priority	[Read-only] Todo priority (1 - 9).
start	start	[Read-only] Date and time the task is to be started. See Date Fields .
status	status	[Read-only] Todo status: <ul style="list-style-type: none"> ■ A - Active ■ C - Completed ■ D - Deferred ■ N - Not Started ■ W - Waiting
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Read-only] The ID of the associated user.

Uprate

A user rate per project [Uprate] is the hourly or daily rate used to bill for a user's time spent on a particular project. Used when your OpenAir account is configured to get the billing rates from Employee / Project [account.rate_from = up].

—	XML	SOAP	REST	Database table
Object	Uprate	oaUprate	—	up_rate
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Uprate object has the following standard properties:

 **Note:** Uprate object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
categoryid	category_id	The ID of the associated category.
created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	Currency for the money fields in the record.
customerid	customer_id	The ID of the associated customer.
duration	duration	Billing rate:

XML / SOAP	Database	Description
		<ul style="list-style-type: none"> ■ H – hourly ■ D – Daily
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
job_codeid	job_code_id	The ID of the job code this rate is associated with. This is only used in the context of project billing rules.
notes	notes	Notes associated with the user project rate (uprate).
project_billing_ruleid	project_billing_rule_id	If project billing rules are used, this is the ID of the associated project billing rule.
projectid	project_id	The ID of the associated project.
rate	rate	The billing rate.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	The ID of the associated user.

User

A user [User] is an individual that has access to your OpenAir account. A user can be an employee (company employee or subcontractor) or a guest (customer).

Review the [Usage Guidelines](#) for the User object.

—	XML	SOAP	REST	Database table
Object	User	oaUser	—	user
Supported Commands	CreateUser , Read , Modify , Delete	createUser() , read() , modify() , upsert() , delete()	—	—

The User object has the following standard properties:

 **Note:** User object properties may also include custom fields. The object type supports the custom equal to read method and the enable_custom read attribute.

XML	SOAP	Database	Description
account_workscheduleid	account_workscheduleid	account.workschedule_id	The ID of the associated company work schedule.
acct_code	acct_code	acct_code	Optional accounting system code for integration with external accounting systems.
active	active	active	A 1/0 field indicating where this is designated as an active user.
addr	—	—	The user's address. See Address Fields . The Address object must include an email.

XML	SOAP	Database	Description
—	addr_addr1	address1	Address line one.
—	addr_addr2	address2	Address line two.
—	addr_addr3	address3	Address line three.
—	addr_addr4	address4	Address line four.
—	addr_city	city	The city.
—	addr_contact_id	—	The ID of the associated contact record
—	addr_country	country	The country.
—	addr_email	email	[Required] The user's email address.
—	addr_fax	fax	The user's fax number.
—	addr_first	first	The user's first name.
—	addr_id	—	The ID of the associated address.
—	addr_last	last	The user's last name.
—	addr_middle	middle	The user's middle name.
—	addr_mobile	mobile	Mobile number.
—	addr_phone	phone	The user's phone number.
—	addr_salutation	salutation	The user's salutation.
—	addr_state	state	The state.
—	addr_zip	zip	The zip code.
az_approvalprocess	az_approvalprocess	az_approvalprocess	The approvalprocess_id of the expense authorization approval process. This field is mutually exclusive with az_approver.
az_approver	az_approver	az_approver	The user ID of the expense authorization approver if this is a single approver process. This field is mutually exclusive with az_approvalprocess. <ul style="list-style-type: none"> ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
book_assign_stamp	book_assign_stamp	—	Internal hash key.
br_approvalprocess	br_approvalprocess	br_approvalprocess	The approvalprocess_id of the deal_booking_request approval process. This field is mutually exclusive with br_approver.
br_approver	br_approver	br_approver	The user ID of the booking request approver if this is a single approver process. This field is mutually exclusive with br_approvalprocess. <ul style="list-style-type: none"> ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.

XML	SOAP	Database	Description
code	code	acct_code	The acct_code.
cost	cost	loaded_cost.cost	[Write-only] New cost value. Used with update_cost Updating User Loaded Costs .
cost_centerid	cost_centerid	cost_center_id	The ID of the associated cost center.
cost_currency	cost_currency	loaded_cost.currency	[Write-only] Currency of the cost. Used with update_cost Updating User Loaded Costs .
cost_end_date	cost_end_date	loaded_cost.end_date	[Write-only] End date for the new loaded cost. If left blank, the new cost will have no end date. See Date Fields . Used with update_cost Updating User Loaded Costs .
cost_lc_level	cost_lc_level	loaded_cost.lc_level	[Write-only] If multiple loaded cost levels are enabled, use this field to hold the level of the loaded cost. Used with update_cost Updating User Loaded Costs .
cost_start_date	cost_start_date	loaded_cost.start_date	[Write-only] Start date for the new loaded cost. If left blank, the new cost will assume the current date as it's start date. See Date Fields . Used with update_cost Updating User Loaded Costs .
created	created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	currency	The currency for money fields.
cv_attachment_id	cv_attachment_id	resource_attachment.attachment_id	The ID of the user's latest CV.
departmentid	departmentid	department_id	The ID of the associated department.
dr_approvalprocess	dr_approvalprocess	dr_approvalprocess	The approvalprocess_id of the deal_booking_request approval process. This field is mutually exclusive with br_approver.
dr_approver	dr_approver	dr_approver	The user ID of the deal booking request approver if this is a single approver process. This field is mutually exclusive with dr_approvalprocess. <ul style="list-style-type: none"> ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
external_id	external_id	external_id	The unique external record ID if the record was imported from an external system.

XML	SOAP	Database	Description
externalid	externalid	externalid	If the record was imported from an external system, you store the unique external record ID here.
filterset_ids	filterset_ids	—	A comma separated list of filter set IDs this record should be part of.
filterset_stamp	filterset_stamp	—	A unique string which changes when the primary filter set changes for the user.
flags	flags	flags	User-specific settings. See Company and User Settings .
generic	generic	generic	A 1/0 field indicating whether this is a generic resource. Cannot be modified.
hierarchy_node_ids	hierarchy_node_ids	—	The IDs of the associated hierarchy nodes.
hint	hint	hint	Password hint.
id	id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
is_user_schedule	is_user_schedule	—	[Write-only] A 1/0 field indicating whether the user should draw their workschedule from an account_workschedule or draw from a custom workschedule. 0 sets the user workschedule to the account workschedule specified in account_workscheduleid, 1 constructs a custom workschedule from the supplied workschedule_workdays and workschedule_workhours fields. Used with update_workschedule Setting User Work Schedule .
job_codeid	job_codeid	job_code_id	The ID of the current job code this user belongs to.
km_filter_set	km_filter_set	km_filter_set	The ID of the optional filter set for the Workspaces module.
line_managerid	line_managerid	line_manager_id	The ID of this user's line manager (will actually be another user_id).
locked	locked	locked	A 1/0 field indicating if this user is locked.
logintime	logintime	user_login.logintime	The date and time of the user's last login. See Date Fields .
ma_filter_set	ma_filter_set	ma_filter_set	The ID of the optional filter set for the My Account module.
mfa_status	mfa_status	mfa_status	A 1/0 field indicating whether the user must sign in using two-factor authentication. The property is read-only if two-factor authentication is not enabled for your company's OpenAir account.

XML	SOAP	Database	Description
name	name	name	The name used for display in lists. This is programmatically generated if not entered.
nickname	nickname	nickname	[Required] The users nickname. This must be unique.
om_filter_set	om_filter_set	om_filter_set	The ID of the optional filter set for the Opportunities module.
password	password	password	[Required] The user's password. Not returned when reading objects.
password_forced_change	password_forced_change	password_forced_change	A 1/0 field indicating whether the password must change at next login.
payroll_code	payroll_code	payroll_code	The payroll code for this user.
pb_approvalprocess	pb_approvalprocess	pb_approvalprocess	The approvalprocess_id of the proposals approval process. This field is mutually exclusive with pb_approver.
pb_approver	pb_approver	pb_approver	The user ID of the booking request approver if this is a single approver process. This field is mutually exclusive with br_approvalprocess. <ul style="list-style-type: none"> ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
picklist_label	picklist_label	—	Label as shown on form picklist.
pm_filter_set	pm_filter_set	pm_filter_set	The ID of the optional filter set for the Projects module.
po_approvalprocess	po_approvalprocess	po_approvalprocess	The approvalprocess_id of the purchase order approval process. This field is mutually exclusive with po_approver.
po_approver	po_approver	po_approver	The user_id of the purchase order approver if this is a single user approver process. This field is mutually exclusive with po_approvalprocess. <ul style="list-style-type: none"> ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
po_filter_set	po_filter_set	po_filter_set	The ID of the optional filter set for the Purchases module.
pr_approvalprocess	pr_approvalprocess	pr_approvalprocess	The approvalprocess_id of the purchase request approval process. This field is mutually exclusive with pr_approver.
pr_approver	pr_approver	pr_approver	The user ID of the purchase request approver if this is a single user approver process. This field is mutually exclusive with pr_approvalprocess. <ul style="list-style-type: none"> ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.

XML	SOAP	Database	Description
primary_filter_set	primary_filter_set	primary_filter_set	The ID of the primary filter set for this user. Defaults to the default primary filter set if not set when adding a user.
project_access_nodes	project_access_nodes	project_access_nodes	Comma delimited list of hierarchy node IDs for project level access control.
rate	rate	rate	The hourly billing rate.
report_filter_set	report_filter_set	report_filter_set	The ID of the optional filter set for Reporting.
rm_filter_set	rm_filter_set	rm_filter_set	The ID of the optional filter set for the Resources module.
rm_approvalprocess	rm_approvalprocess	rm_approvalprocess	The approvalprocess_id of the booking approval process. This field is mutually exclusive with rm_approver.
rm_approver	rm_approver	rm_approver	The user_id of the booking approver if this is a single user approver process. This field is mutually exclusive with rm_approvalprocess. <ul style="list-style-type: none"> ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
role_id	role_id	role_id	The ID of the associated role. Defaults to the default role if not set when adding a user.
rpc_api_updated	rpc_api_updated	rpc_api_updated	Time of the last update of resource profile for this user via API. See Date Fields .
rpc_updated_by	rpc_updated_by	rpc_updated_by	User ID of user who updated the resource profile
rpc_user_updated	rpc_user_updated	rpc_user_updated	Time of the last update of resource profile for this user via web. See Date Fields .
sr_approvalprocess	sr_approvalprocess	sr_approvalprocess	The approvalprocess_id of the schedule_request approval process. This field is mutually exclusive with sr_approver.
sr_approver	sr_approver	sr_approver	The user ID of the schedule request approver if this is a singleapprover process. This field is mutually exclusive with sr_approvalprocess. <ul style="list-style-type: none"> ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
ssn	ssn	ssn	The user's social security number. Returned only if the authenticated user is an account administrator.
ta_approvalprocess	ta_approvalprocess	ta_approvalprocess	The approvalprocess_id of the timesheet approval process. This field is mutually exclusive with ta_approver.

XML	SOAP	Database	Description
ta_approver	ta_approver	ta_approver	The user ID of the timesheet approver if this is a single approver process. This field is mutually exclusive with ta_approvalprocess. <ul style="list-style-type: none"> 1 - approver is the manager. 2 - approver is the manager's manager.
ta_filter_set	ta_filter_set	ta_filter_set	The ID of the optional filter set for the Timesheets module.
tag_end_date	tag_end_date	entity_tag.end_date	[Write-only] End date for the new tag. If left blank, the end date for the new tag will be undefined and the new tag will assume default status for the user. See Date Fields . Used with update_tag Updating User Entity Tags .
tag_group_attribute_id	tag_group_attribute_id	entity_tag.tag_group_attribute_id	[Write-only] The ID of the tag group attribute that is being assigned to the new tag. Used with update_tag Updating User Entity Tags .
tag_group_id	tag_group_id	entity_tag.tag_group_id	[Write-only] The ID of the tag group for the new tag. Used with update_tag Updating User Entity Tags .
tag_start_date	tag_start_date	entity_tag.start_date	[Write-only] Start date for the new tag. If left blank, the start date for the new tag will be set to the current date. See Date Fields . Used with update_tag Updating User Entity Tags .
tb_filter_set	tb_filter_set	tb_filter_set	[Write-only] The ID of the optional filter set for the Invoices module.
te_allowance_approvalprocess	te_allowance_approvalprocess	te_allowance_approvalprocess	The approvalprocess_id of the allowance report approval process. This field is mutually exclusive with po_approver.
te_allowance_approver	te_allowance_approver	te_allowance_approver	The user_id of the allowance report approver if this is a single user approver process. This field is mutually exclusive with te_allowance_approvalprocess. <ul style="list-style-type: none"> 1 - approver is the manager. 2 - approver is the manager's manager.
te_approvalprocess	te_approvalprocess	te_approvalprocess	The approvalprocess_id of the expense report approval process. This field is mutually exclusive with te_allowance_approver.
te_approver	te_approver	te_approver	The user_ID of the expense report approver if this is a single approver

XML	SOAP	Database	Description
			process. This field is mutually exclusive with te_approvalprocess. <ul style="list-style-type: none"> 1 - approver is the manager. 2 - approver is the manager's manager.
te_filter_set	te_filter_set	te_filter_set	The ID of the optional filter set for the Expenses module.
timezone	timezone	timezone	The user's timezone. Defaults to the time zone for the company's account if not set when adding a user.
type	type	—	Legacy field.
update_cost	update_cost	—	[Write-only] Set to 1 to update loaded cost information automatically when adding or modifying a user. See Updating User Loaded Costs .
update_tag	update_tag	—	[Write-only] Set to 1 to update entity tag information automatically when adding or modifying a user. See Updating User Entity Tags .
update_workschedule	update_workschedule	—	[Write-only] Set to 1 to update work schedule information automatically when adding or modifying a user. See Setting User Work Schedule .
updated	updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
user_locationid	user_locationid	user_location_id	The location ID for this user.
week_starts	week_starts	week_starts	The first day of the week for this user: <ul style="list-style-type: none"> 0 - Monday 6 - Sunday Defaults to the first day of the week for the company's account if not set when adding a user.
workschedule_workdays	workschedule_workdays	—	A CSV list of workdays, with each value indicating a day in the schedule and values ranging from 0(Monday) to 6(Sunday). For example, "0,1,4" indicates that a user works on Monday, Tuesday and Friday. <p>Used with update_workschedule Setting User Work Schedule.</p>
workschedule_workhours	workschedule_workhours	—	A CSV list of values for the user's default workhours and workhours for each day. At least one value for workschedule_workhours must be submitted, but a value for each day may be submitted as well. For example, if the user's workschedule_workdays is set to "0,1,4", then submitting a value of only "8" for workschedule_workhours sets the user's default hours to 8

XML	SOAP	Database	Description
			and each workday assumes this value as well. In addition, submitting a workschedule_workdays value of "8,1,2,3" sets the user's default workhours to 8, sets Monday to 1, Tuesday to 2, and Friday to 3. Used with update_workschedule Setting User Work Schedule .
workscheduleid	workscheduleid	workschedule_id	The ID of the associated user workschedule.

Usage Guidelines

Review the following guidelines:

- Notes on relevant access privileges and role permissions:
 - To view, create or modify a user record or generic user record, the primary filter set assigned to the authenticated user must allow access to that user record or generic user record.
 - To modify a guest user record, the primary filter set assigned to the authenticated user must allow access to all users.
 - To create or modify a user record (other than the authenticated user's own record), the authenticated user must be an administrator or have the **View, modify, and create new users** or **View and modify users** role permission.
 - To modify user settings [flags], the authenticated user must be an administrator.
 - All authenticated users can modify their own user record without the **View, modify, and create new users** and **View and modify users** role permission role permissions.
 - The role_id property can be set to 1 (Administrator) only if the role authenticated user is Administrator.
 - The filterset_ids property can be changed by any authenticated user with the **Modify filter sets for existing things** role permission, even if that authenticated does not have the **View, modify, and create new users** or **View and modify users** role permission.
 - To create or modify a generic user record, the authenticated user must be an administrator or have the **View and modify generics** role permission.
- To add users, use the [CreateUser](#) (XML API) or `createUser()` (SOAP API) command instead of the `Add` or `add()` command.
- Limits are enforced to prevent you from creating or activating users if doing so would exceed the number of user licenses purchased for your account. If no user licenses of the appropriate type are available, the [CreateUser](#) (XML API) or `createUser()` (SOAP API) command creates a new user record, but sets it as inactive (clears the **Active** box on the employee record), and the [CreateUser](#)(XML API) or [Modify](#)(XML API) or `modify()` (SOAP API) commands cannot be used to activate a user record (to check the **Active** box on the employee record). For more information about OpenAir licensing and compliance, see [OpenAir Administrator Guide](#).
- You can use the generic attribute to 1 when reading User objects to return generic resources only. See [Read Attributes](#).
- When adding a user, default values are used for approval fields (<transaction>_approvalprocess and <transaction>_approver where <transaction>, typically a two-letter code, designates the transaction type) if not set. Unless default values were set when adding a user in the OpenAir UI, the head of the department associated with the user (if set), or the first account administrator (otherwise) is the default approver.

- When adding or modifying a user, you can the following information for that user:
 - Loaded cost – See [Updating User Loaded Costs](#).
 - Entity tag – See [Updating User Entity Tags](#).
 - Work schedule – See [Setting User Work Schedule](#).

Updating User Loaded Costs

You can set or update user loaded cost information when creating or updating a user object. To do so, use the following property values:

Property	Value	Notes
update_cost	1	—
cost_start_date	<ul style="list-style-type: none"> ■ Start date for the new loaded cost. See Date Fields. ■ <i>Empty value</i> to use the current date. 	—
cost_end_date	<ul style="list-style-type: none"> ■ End date for the new loaded cost. See Date Fields. ■ <i>Empty value</i> to leave the end date undefined, until the same level loaded cost is updated again. 	If an empty value is used, the cost_end_date will be set automatically to a date preceding the cost_start_date of the new default loaded cost when the same level loaded cost is updated again. It is not possible to set a default value. All loaded cost information updated using this method is historical information
cost	New loaded cost value.	—
cost_currency	The currency of the new loaded cost value.	—
cost_lc_level	The loaded cost level, if multiple cost levels are enabled for your account. Defaults to 0 (primary loaded cost) if not specified.	—

To read user entity tag information, use the [LoadedCost](#) project.

Updating User Entity Tags

You can set or update the user entity tags when creating or updating a user object. To do so, use the following property values:

Property	Value	Notes
update_tag	1	—
tag_start_date	<ul style="list-style-type: none"> ■ Start date for the new entity tag. See Date Fields. ■ <i>Empty value</i> to use the current date. 	—
tag_end_date	<ul style="list-style-type: none"> ■ End date for the new entity tag. See Date Fields. 	Use an empty value to set the entity tag as the default entity tag for the user valid from

Property	Value	Notes
	<ul style="list-style-type: none"> Empty value to leave the end date undefined, until the entity tag is updated with an empty value for tag_end_date again. 	tag_start_date. If a default entity tag was set previously, the tag_end_date for the previous default entity tag is set automatically according to a date preceding the tag_start_date of the new default entity tag.
tag_group_id	ID of the tag group for the new entity tag.	—
tag_group_attribute_id	The ID of the tag group attribute assigned to the new entity tag.	—

To read user entity tag information, use the [Entitytag](#) object.

Setting User Work Schedule

You can set or update the user work schedule when creating or updating a user object. To do so, use the following property values:

Property	Value	Notes
update_workschedule	1	—
is_user_schedule	<ul style="list-style-type: none"> 1 to define a work schedule specific to the user. 0 to use the company work schedule with ID account_workscheduleid. 	—
workschedule_workdays	Comma-separated list of numbers for usual work days. Each day of the week is represented by a number from 0 (Monday) to 6 (Sunday).	Used if is_user_schedule = 1.
workschedule_workhours	<ul style="list-style-type: none"> Comma-separated list of numbers, if your account configuration allows for different work hours per day. The first value is the default number of hours per day, subsequent values are the number of hours for each work day listed in workschedule_workdays. Default number of hours per day, otherwise. 	Used if is_user_schedule = 0. To allow for different work hours, check the Enable distinct work hours per day on work schedule box on the Optional Features administration form (Administration > Global Settings > Optional Features)
account_workscheduleid	The ID of the company work schedule.	Used if is_user_schedule = 0.

To read user work schedule information, use the [UserWorkSchedule](#) object.

UserLocation

A user location [UserLocation] is a geographical classification information for users.

—	XML	SOAP	REST	Database table
Object	UserLocation	oaUserLocation	—	user_location

—	XML	SOAP	REST	Database table
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The UserLocation object has the following properties:

XML / SOAP	Database	Description
acct_code	acct_code	Optional accounting system code for integration with external accounting systems.
active	active	A 1/0 field indicating whether the record is active.
created	created	[Read-only] Time the record was created. See Date Fields .
external_id	external_id	The unique external record ID if the record was imported from an external system.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The name of the user location.
notes	notes	Notes associated with this user location.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .

UserWorkschedule

A work schedule [UserWorkschedule] is the weekly or multiweekly pattern of work followed by the company or by an employee. It determines the employees' normal working days and working hours of users.

Review the [Usage Guidelines](#) for the UserWorkschedule object.

—	XML	SOAP	REST	Database table
Object	UserWorkschedule	oaUserWorkschedule	—	workschedule
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The UserWorkschedule object has the following properties:

XML / SOAP	Database	Description
account_workscheduleid	account_workschedule_id	The ID of the company workschedule to use when userid is not 0.
acct_code	acct_code	Optional accounting system code for integration with external accounting systems.
created	created	[Read-only] Time the record was created. See Date Fields .
externalid	external_id	If the record was imported from an external system you store the unique external record id here.

XML / SOAP	Database	Description
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
master_workscheduleid	master_workschedule_id	ID of master workschedule. This workschedule is part of sequence in a recurring multi-week schedule.
name	name	[Required] The company-wide schedule name for company schedules or user's first and last name for user schedules."
sample_date	sample_date	See Date Fields .
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
use_this_schedule	use_this_schedule	Can be 0 or 1. <ul style="list-style-type: none"> ■ If "1" and userid has a value, then this is a user schedule (with userid above) which overrides the company schedule. ■ If "1" and userid is 0, then this is a company schedule. ■ If "0" then the user (with userid above) is using the company schedule indicated by account_workscheduleid.
userid	user_id	ID of the user if this is a users work schedule. 0 - if there is a company work schedule. Required if use_this_schedule is set to 1.
week_num	week_num	Sequence/week number of recurring schedule. Master records will always be week 1. Child workschedules will be begin at week 2.
workdays	workdays	A seven-letter string indicating which days of the week are available for project work. (Monday is 0, Sunday is 6; 01234 = Mon. - Fri.; 0123456 = every day). Always begins with the letter "x" (So "Monday only" would be "x0"). Required if use_this_schedule is set to 1.
workhourid	workhour_id	ID of the workhour if this is a users work schedule.
workhours	workhours	[Required] The number of hours worked per day. Must be a numerical value between 0 (not included) and 24 (included).

Usage Guidelines

Review the following guidelines:

- For exceptions to the normal work pattern, see [Scheduleexception](#).
- When you add or modify a [User](#), you can modify the UserWorkschedule associated with that [User](#). See [Setting User Work Schedule](#).

- A user can have only one work schedule. There cannot be more than one UserWorkschedule object with the same userid.

Vendor

A vendor [Vendor] is an external source your company purchases goods or services from.

—	XML	SOAP	REST	Database table
Object	Vendor	oaVendor	—	vendor
Supported Commands	Add , Read , Modify , Delete	add() , read() , modify() , upsert() , delete()	—	—

The Vendor object has the following standard properties:

Note: Vendor object properties may also include custom fields. The object type supports the `custom_equal` to read method and the `enable_custom` read attribute.

XML	SOAP	Database	Description
active	active	active	A 1/0 field indicating where this is designated as an active vendor 1/0.
addr	—	—	The vendor's address. An Address object with the company's address details. See Address Fields .
—	addr_addr1	address1	Address line one.
—	addr_addr2	address2	Address line two.
—	addr_addr3	address3	Address line three.
—	addr_addr4	address4	Address line four.
—	addr_city	city	The city.
—	addr_contact_id	—	The ID of the associated contact record
—	addr_country	country	The country.
—	addr_email	email	The vendor's email address.
—	addr_fax	fax	The vendor's fax number.
—	addr_first	—	The vendor's first name.
—	addr_id	—	The ID of the associated address.
—	addr_last	—	The vendor's last name.
—	addr_middle	—	The vendor's middle name.
—	addr_mobile	—	Mobile number.
—	addr_phone	phone	The vendor's phone number.
—	addr_salutation	—	The vendor's salutation.

XML	SOAP	Database	Description
—	addr_state	state	The state.
—	addr_zip	zip	The zip code.
attention	attention	attention	To whom purchase orders should be sent.
code	code	acct_code	Optional accounting system code for integration with external accounting systems.
created	created	created	[Read-only] Time the record was created. See Date Fields .
currency	currency	currency	Currency for the money fields in the record. Also the default currency when a purchase order is created.
externalid	externalid	externalid	If record is imported from an external system, store external record ID here.
id	id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	name	Vendor name. Displays on all the vendor pop-up windows in the application.
notes	notes	notes	Notes associated with this vendor.
picklist_label	picklist_label	—	Label as shown on form picklist.
purchaseorder_email_text	purchaseorder_email_text	purchaseorder_email_text	Extra text to include in emails announcing purchase orders.
purchaseorder_text	purchaseorder_text	purchaseorder_text	Text to display on every purchase order.
tax_locationid	tax_locationid	tax_locationid	The ID of the associated tax location.
terms	terms	terms	Standard payment terms for the vendor.
updated	updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
web	web	web	Vendor's Web address.

Usage Guidelines

You cannot delete a Vendor object if this object is referenced by a [Ticket](#) object. Delete any dependent objects first before you delete a Vendor object.

Viewfilter

A view filter [Viewfilter] is a collection of user-defined criteria that can be used to filter list view data.

—	XML	SOAP	REST	Database table
Object	Viewfilter	oaViewfilter	—	viewfilter
Supported Commands	Read	read()	—	—

The Viewfilter object has the following properties:

XML / SOAP	Database	Description
action	—	[Read-only] The filter action.
created	created	[Read-only] Time the record was created. See Date Fields .
fields	—	[Read-only] Comma delimited list of fields.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
label	label	[Read-only] The name given to this filter. It appears in the Filter: drop-down list.
limit_values	—	[Read-only] For permission rule with action set to “limit values”, the comma-separated list of limit values for each limited field, one field per line. If limit values correspond to entity names on the UI, the internal IDs for these entities are returned. For example: <pre> 1 _project_stage_id:&quot;3&quot;;&quot;4&quot;; 2 currency:&quot;USD&quot;;&quot;GBP&quot;; 3 _custom_RF_cf_Project_dropdown:&quot;dropdown_value2&quot;;&quot;dropdown_value3&quot;; 4 _custom_RF_cf_Project_pick_list:&quot;2&quot;; </pre>
match_all	match_all	[Read-only] A 1/0 field. 1 = if all rules met. 0 = if rules must be met.
name	name	[Read-only] The internal name of the list or calendar this filter is applied to.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	userid	[Read-only] The user who created this filter.

Viewfilterrule

A view filter rule [Viewfilterrule] is one of the criteria used in a view filter.

—	XML	SOAP	REST	Database table
Object	Viewfilterrule	oaViewfilterrule	—	viewfilter_rule
Supported Commands	Read	read()	—	—

The Viewfilterrule object has the following properties:

XML / SOAP	Database	Description
condition	condition	[Read-only] One of the following conditions: ct = contains, nc = does not contain, eq = is equal to, ne = is not equal to, bw = begins with, ew = ends with, gt = is greater than, ge = is greater than or equal to, lt = is less than, le = is less than or equal to, in = in the set of.
created	created	[Read-only] Time the record was created. See Date Fields .

XML / SOAP	Database	Description
field	field	[Read-only] The field or column to be compared.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
required	required	[Read-only] A 1/0 field. 1 = if this condition must be met. 0 = if this is one of many that will satisfy this viewfilter. (If 1, this condition is ANDed with the others. If 0, this condition is ORed with the others.)
type	type	[Read-only] The underlying type of the field or column to be compared: C = character string, N = number, D = date, 'MS' - milliseconds, B = Yes/No, P1 = picker_button, P2 = pop-up menu, 'R' - reference value table.field.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
value	value	[Read-only] The value the field is compared to.
viewfilterid	viewfilter_id	[Read-only] The viewfilter to which this rule belongs.

WorkscheduleWorkhour

A work schedule work hour [WorkscheduleWorkhour] is the number of hours worked in a week day as part of a normal work pattern.

—	XML	SOAP	REST	Database table
Object	WorkscheduleWorkhour	oaWorkscheduleWorkhour	—	workschedule_workhour
Supported Commands	Read	read()	—	—

The WorkscheduleWorkhour object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
workday	workday	[Read-only] A one-letter string indicating which day of the week. Monday is '0', Tuesday is '1', ..., Sunday is '6'
workhours	workhours	[Read-only] The number of hours worked for this day.
workscheduleid	workschedule_id	[Read-only] The ID of the associated primary account workschedule.

Workspace

A workspace [Workspace] is a collection of documents, discussions and links.

—	XML	SOAP	REST	Database table
Object	Workspace	oaWorkspace	—	workspace

—	XML	SOAP	REST	Database table
Supported Commands	Add, Read, Modify, Delete	add(), read(), modify(), upsert(), delete()	—	—

The Workspace object has the following standard properties:

Note: Workspace object properties may also include custom fields. The object type supports the custom `equal` to read method and the `enable_custom` read attribute.

XML / SOAP	Database	Description
allow_guests	allow_guests	A 1/0 field indicating whether guests can be subscribed to this.
created	created	[Read-only] Time the record was created. See Date Fields .
date	date	[Required] The date of the workspace. See Date Fields .
description	description	The description of the workspace.
global	global	A 1/0 field indicating if this is a global workspace (available to all users). Must be set to 1 if <code>global_access</code> is not empty.
global_access	global_access	The access permissions for all users. Required if <code>global</code> is set to 1. <ul style="list-style-type: none"> ■ 'R' - Read-only ■ 'W' - Read/write ■ 'A' - Administrator
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
name	name	[Required] The workspace name.
notes	notes	Notes.
open	open	A "1/0" field indicating whether this workspace is open.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	[Required] The user ID of the workspace owner.

Workspacelink

A workspace link [Workspacelink] is the association of a workspace with a record of a given type.

—	XML	SOAP	REST	Database table
Object	Workspacelink	oaWorkspacelink	—	workspace_link
Supported Commands	Add, Read, Modify	add(), read(), modify(), upsert()	—	—

The Workspacelink object has the following properties:

XML / SOAP	Database	Description
created	created	[Read-only] Time the record was created. See Date Fields .

XML / SOAP	Database	Description
description	description	The description of the workspace link. For internal links, defaults to a concatenation of table_name and the name of the record with internal ID recordid separated by a colon (:).
external	external	A 1/0 field indicating if the record is an external link.
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
recordid	record_id	The table ID the workspace is associated with.
table_name	table_name	The table the workspace is associated with. Possible values: 'customer','deal','department','estimate','project','proposal','resource' (virtual),'user'
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
url	url	The URL of external link.
workspaceid	workspace_id	The ID of the associated workspace.

Workspaceuser

A workspace user [Workspaceuser] defines workspace access control.

—	XML	SOAP	REST	Database table
Object	Workspaceuser	oaWorkspaceuser	—	workspace_user
Supported Commands	Add , Read , Modify	add() , read() , modify() , upsert()	—	—

The Workspaceuser object has the following properties:

XML / SOAP	Database	Description
access	access	The access permissions for the user: <ul style="list-style-type: none"> ■ R - Read-only ■ W - Read/write ■ A - Administrator
created	created	[Read-only] Time the record was created. See Date Fields .
id	id	[Read-only] Unique ID. Automatically assigned by OpenAir.
projectgroupid	projectgroup_id	The ID of the project group if the user was assigned as part of a project group. Mutually exclusive with userid.
updated	updated	[Read-only] Time the record was last updated or modified. See Date Fields .
userid	user_id	The ID of the associated user. Mutually exclusive with projectgroupid.
workspaceid	workspace_id	The ID of the associated workspace.

Release History

The following summarizes the main changes to OpenAir XML API and SOAP API for each OpenAir release.

April 13, 2023

Extended coverage to include the following object types and properties.

Object type	Properties
Customer	credit_invoice_layout_id
Project	credit_invoice_layout_id
User	mfa_status

October 7, 2023

Added the following [Error Codes](#):

- 965 — File could not be saved.
- 1422 — Missing Address object.

April 15, 2023

- Added the following error codes [Error Codes](#):
 - 426 — You must use an account-specific domain.
 - 1418 — Invalid preference settings format.
 - 1419 — No full user licenses available.
 - 1420 — No T&E or full user licenses available
 - 1421 — No guest or full user licenses available

See also object type [User](#), and commands [CreateUser](#) and [Modify](#) (XML API) or [createUser\(\)](#) and [modify\(\)](#) (SOAP API).

- Added support for commands [Unapprove](#) (XML API) and [unapprove\(\)](#) (SOAP API) to the [Schedulerequest](#) object type.
- Added support for the Attachment Thumbnail feature. See object type [Attachment](#).

October 8, 2022

- Extended coverage to include the following object types and properties.

Object type	Properties
Customer	customer_location_id
CustomerLocation	active, created, deleted, ID, name, notes, updated

- Fixed a previous limitation that prevented specifying the address information properties to be returned when reading [Company](#), [Contact](#), [Customer](#), [CustomerProspect](#), [User](#) or [Vendor](#) records. The XML API returned values for all address fields. The SOAP API either returned values for all address fields if the XML property names were used, or did not return any address field values if the SOAP property names were used (property names beginning with `addr_`, `billing_addr_`, or `contact_addr_`).

April 9, 2022

- Extended coverage to include the following object types and properties.

Object type	Properties
Projectbillingtransaction	<code>fulfillmentid</code>
Projectbillingtransaction	<code>limit_values</code>

- Added Error code 206 — Role error. See [Error Codes](#).

October 9, 2021

- Extended coverage to include the following object types and properties.

Object type	Properties
Projectbillingrule	<code>cap_by_customerpo</code> , <code>project_task_id</code>

- Added the following error codes [Error Codes](#):
 - 1416 — Invalid cap by customer PO.
 - 1417 — Invalid project task ID.
- Changes to OAuth 2.0 Authorization:
 - OAuth 2.0 access token validity period cannot be greater than session timeout — see [Application Configuration](#).
 - OAuth 2.0 refresh token validity period can be between 1 and 31 days in one-day increments — see [Application Configuration](#).

April 10, 2021

- Extended coverage to include the following object types and properties.

Object type	Properties
Resourcesearch	<code>location</code> , <code>skill</code> , <code>industry</code> , <code>jobrole</code> , <code>education</code> , <code>customprofile_1</code> – <code>customprofile_35</code>

- Added audit and management capabilities for user authorizations. Account administrators can use web services reports to audit and revoke authorizations granted by OpenAir users to integration applications. See [Auditing and Managing OAuth 2.0 Authorizations](#) under [OAuth 2.0 Authorization](#).
- Added the following [Error Codes](#):
 - 1414 — Invalid approval status.
 - 1415 — Phase cannot be assigned.
 - 1500 — Access to the Expenses module denied.

October 10, 2020

- Extended coverage to include the following object types and properties.

Object type	Properties
Invoice	payment_termsid

- Added an option to disallow adding or modifying a [Ticket](#) object with the property quantity set to zero. Added corresponding error code (1412 — Invalid quantity). See [Error Codes](#).

April 18, 2020

- Extended coverage to include the following object types and properties.

Object type	Properties
JobCodeUsed	id, table_name, used_by, position, created, updated
ResourceRequestQueue	booking_type_id

- Added support for OAuth 2.0 token based authentication. See [OAuth 2.0 Authorization and Authentication](#).

October 12, 2019

- Extended coverage to include the following object types and properties.

Object type	Properties
Projectbillingrule	extra_data
Projecttaskassign	rule_rate_override, rule_rate_override_currency
Timesheet	min_hours, max_hours
Workscheduleworkhour	id, attributes, workscheduleid, workday, workhours, created, updated

- Added support for returning the minimum number of hours required on the timesheet and maximum number of hours allowed in a [Timesheet](#) object type as determined by Timesheet rules. This includes:
 - Added calculated fields min_hours and max_hours to [Timesheet](#) object type.
 - Added read attribute calculate_hours. See [Read Attributes](#).
- Added support for commands [Delete](#) (XML API) and [delete\(\)](#) (SOAP API) to the [Uprate](#) object type.

April 13, 2019

- Extended coverage to include the following object types and properties.

Object type	Properties
Newsfeed	id, created, updated, attributes
NewsfeedMessage	id, newsfeedid, title, content, tagid, created, authorid, updated, editorid, attributes

Object type	Properties
Project	newsfeedid
Projectbillingtransaction	currency

- Added Administration > Global Settings > Account > API Limits page in the OpenAir UI. See [API Limits](#).

October 13, 2018

- Extended coverage to include the following object types and properties.

Object type	Properties
ProjectBudgetGroup	etc, etc_labor, etc_expense, etc_purchase, eac, eac_labor, eac_expense, eac_purchase, itd, itd_labor, itd_expense, itd_purchase
Task	start_time, end_time

- Added support for approval operation [Approval-Related Operations](#) to the [Booking](#) object type.
- Added the following [Error Codes](#):
 - 1404 — Invalid time.
 - 1405 — Illegal time range.
 - 1406 — No permission to edit time data.
 - 1407 — Invalid hours.

April 14, 2018

- Extended coverage to include the following object types and properties.

Object type	Properties
ProjecttaskEstimate	id, project_task_id, user_id, timesheet_id, hours, date_changed, changed_by, created, updated

- Added the ability to generate the OpenAir WSDL with wrapped document-literal binding. See [XML Schema and WSDL Definition Documents](#).

October 14, 2017

- Extended coverage to include the following object types and properties.

Object type	Properties
AccountingPeriod	id, name, start_date, end_date, period_date_how, period_date, current_period, default_period, notes, active, created, updated
Proxy	id, user_id, proxy_id, own, role_id, expiration, deleted, created, updated, audit
ResourceAttachment	id, userid, attachment_id, type, latest_attachment_id, created, updated
Resourceprofile_type	id, name, description, type, related_table, related_id, active, external_id, deleted, created, updated, audit

Object type	Properties
Revenue_recognition_rule	project_billing_ruleid
User	cv_attachment_id

- Added XML API command [ModifyOnCondition](#).
- Added read attribute order. See [Read Attributes](#).
- Added the following [Error Codes](#): 960, 961, 962, 963, and 964.

April 15, 2017

- Extended coverage to include the following object types and properties.

Object type	Properties
Attachment	size
AttributeDescription	id, resourceprofile_typeid, attributeid, description, deleted, created, updated, audit
ExpensePolicy	id, customerid, projectid, description, deleted, created, updated, audit, all_items_allowed
ExpensePolicyItem	id, expense_policyid, itemid, price_max, price_fixed, currency, deleted, created, updated, audit
Project	main_contactid
Purchase_item	project_taskid

- Added support for commands [Delete](#) (XML API) and [delete\(\)](#) (SOAP API) to the [Category_<N>](#), [Costcenter](#) and [Request_item](#) object types.
- Added the following [Error Codes](#): 899, 900, 947, 948, 949, 950, and 951.

October 15, 2016

- Extended coverage to include the following object types and properties.

Object type	Properties
ApprovalLine	id, approvalid, status, timesheetid, envelopeid, proposalid, purchaserequestid, purchaseorderid, authorizationid, schedule_requestid, booking_requestid, deal_booking_requestid, invoiceid, revenue_containerid, bookingid, customerid, project_budget_groupid, projectid, userid, submitter, approvalprocessid, approvalprocess_ruleid, seq_number, action, date, pending_done, project_total, notes, created, updated, audit, delay_to, delay_action
ProjectBudgetGroup	approval_status, budget_by, calculated_total, cf_opt, cf_pes, created, currency, customerid, date, date_approved, date_archived, date_submitted, funding_total, ID, internal_total, labor_subcategory, name, notes, parentid, profitability, projectid, setting, total, total_calculated_billing, total_calculated_cost, total_expected_billing, total_expected_cost, total_from_funding, unassigned_task, updated, userid, version

Object type	Properties
ProjectBudgetRule	category, categoryid, created, currency, customerid, date, end_date, ID, imported, itemid, job_codeid, notes, period, productid, profitability, project_budget_groupid, project_taskid, projectid, quantity, quantity_best, quantity_most_likely, quantity_worst, rate, start_date, total, total_best, total_most_likely, total_worst, updated
ProjectBudgetTransaction	category, categoryid, created, currency, customerid, date, ID, itemid, job_codeid, productid, project_budget_groupid, project_budget_ruleid, project_taskid, projectid, quantity, quantity_best, quantity_most_likely, quantity_worst, total, total_best, total_most_likely, total_worst, updated
Projecttask	classification

- Added SOAP API commands [approve\(\)](#), [reject\(\)](#) and [unapprove\(\)](#).
- Added support for commands [Unapprove](#) (XM API) and [unapprove\(\)](#) (SOAP API) to the [Envelope](#), [Invoice](#) and [Timesheet](#) object types.
- Added the following [Error Codes](#): 945 and 946.

April 16, 2016

- Extended coverage to include the [Role](#) object type.
- Added controls to prevent renaming, modifying, or deleting a custom field if it is used by an active script.

October 17, 2015

- Extended coverage to include the following object types and properties.

Object	Properties
Paymenttype	default_status, default_payment_type
Slip	skip_recognition
TaskAdjustment	created, id, new_taskid, new_timesheetid, old_taskid, old_timesheetid, updated

- Added error code 943 — Project names must be unique by customer. See [Error Codes](#).

April 18, 2015

Extended coverage to include the following object types and properties.

Object	Properties
Booking_request	—
Project	rate_cardid
Agreement , BookingType , Category , Category_<N> , Contact , Costcenter , Customer , Customerpo , Department , Item , Payrolltype , Project , ProjectStage , Projecttask_type , Timetype , User , Vendor	picklist_label

October 18, 2014

Extended coverage to include the following object types and properties.

Object	Properties
Booking	source_booking_id
ItemToUserLocation	—
Projectbillingrule	assigned_user
Ticket	user_locationid
UserLocation	—

May 17, 2014

Extended coverage to include the following object types and properties.

Object	Properties
Attachment	ownerid, is_a_folder, owner_type, name
ResourceRequest	—
ResourceRequestQueue	—
ResourceSearch	—
Workspace	—

February 15, 2014

Extended coverage to include the following object types and properties.

Object	Properties
Address	contact_id

November 16, 2013

Extended coverage to include the following object types and properties.

Object	Properties
Address	id
ApprovalProcess	externalid
BillingSplit	—
BookingByDay	userid

Object	Properties
Company	addr_id
Contact	exported, addr_id
Customer	billing_addr_id, contact_addr_id, addr_id
Invoice	submitted, approved
LoadedCost	externalid
Projectbillingtransaction	customerpid, cost_centerid, timetypeid, customerid, agreementid, payroll_typeid
Reimbursement	userid, audit
SlipProjection	projecttask_typeid, cost_centerid, acct_date, job_codeid
User	addr_id
Vendor	addr_id

August 17, 2013

- Extended coverage to include the following object types and properties.

Object	Properties
Booking	project_assignment_profile_id
PendingBooking	—
Project	rm_approver, rm_approvalprocess
ProjectAssignmentProfile	—
Projecttaskassign	project_assignment_profile_id, pending_booking_id, booking_id
User	rm_approver, rm_approvalprocess

- Added restriction on reading [RevenueProjection](#) objects if projections are running.
- Added error code 606. See [Error Codes](#).

May 18, 2013

- Extended coverage to include the following object types and properties.

Object	Properties
BookingByDay	—
Projectbillingtransaction	slip_stage_id
RevenueProjection	—
Slip	originating_id

- Extended the page coverage for command [MakeURL](#) (XML API) or [makeURL\(\)](#) (SOAP API) with the following new page attribute value: calendar-user.

- Added support for using the XML API to read the number of requests remaining within the current 24-hour window. To do so, use the [Read XML API](#) command and the [RateLimit](#) object. See [Tracking API Usage Against Frequency Limits](#).

March 16, 2013

Extended coverage to include the following object types and properties.

Object	Properties
Booking	date_approved, date_submitted, approval_status

January 19, 2013

Added support for reading custom fields for Custom fields associated with [Budget](#) objects.

November 17, 2012

Extended coverage to include the following object types and properties.

Object	Properties
Item	cost_is_fixed

July 14, 2012

- Extended coverage to include the following object types and properties.

Object	Properties
Booking	notify_owner
Projectbillingrule	exclude_non_billable_task
Revenue_recognition_transaction	portfolio_projectid
Slip	portfolio_projectid

- Added error code 885. See [Error Codes](#).

May 12, 2012

Extended coverage to include the following object types and properties.

Object	Properties
Project	portfolio_projectid, is_portfolio_project

March 17, 2012

- Extended coverage to include the following object types and properties.

Object	Properties
Customer	created, updated, billing_code

- Added the read attribute generic. Set the generic attribute to 1 to return generic resources (users) only. By default the API returns named resources (users) only. See [Read Attributes](#).
- Added the following [Error Codes](#): 941, 1106.

January 21, 2012

- Extended coverage to include the following object types and properties.

Object	Properties
Schedulebyday	id, date, user_id, hours, base_hours, target_hours, target_base_hours, created, updated

- Added custom field support for [Purchaseorder](#), [Request_item](#), and [Schedulebyday](#) objects.

November 19, 2011

Extended coverage to include the following object types and properties.

Object	Properties
Customer	locationid
RevenueStage	id, name, revenue_stage_type, created, updated
Revenue_recognition_transaction	is_from_open_stage

September 17, 2011

- Extended coverage to include the following object types and properties.

Object	Properties
Invoice	credit_rebill_status, original_invoiceid
Project	rv_approver, rv_approvalprocess
Projectbillingrule	category_1id, category_2id, category_3id, category_4id, category_5id
RevenueContainer	project_billing_rule_filter, category_1id, category_2id, category_3id, category_4id, category_5id
Revenue_recognition_transaction	category_1id, category_2id, category_3id, category_4id, category_5id

Object	Properties
Slip	ref_slipid
TaskTimecard	category_1id, category_2id, category_3id, category_4id, category_5id
UserWorkschedule	id, name, userid, use_this_schedule, account_workscheduleid, workdays, workhours, created, updated

- Added API support for all existing time entry [[Task](#)] rounding rules.
- Added error code 882. See [Error Codes](#).

July 16, 2011

- Extended the page coverage for command [MakeURL](#) (XML API) or [makeURL\(\)](#) (SOAP API) with the following new page attribute values: view-invoice, dashboard-project, grid-timesheet, report-timesheet.
- Added custom field support for [Payment](#), [User](#).
- Extended [User](#) object type usage to support generic users.
- Added error code 556. See [Error Codes](#).

May 14, 2011

- Extended coverage to include the following object types and properties.

Object	Properties
Attachment	parentid
Projecttask	default_category_1, default_category_2, default_category_3, default_category_4, default_category_5
RevenueContainer	id, number, date, balancing_type, total_recognized, currency, date_approved, updated, date_submitted, approval_status, total_deferred, name, acct_date, total_accrued, projectid, externalid, total_posted, created, notes, total_invoiced, customerid, exported, prefix

- Added controls to prevent negative values for quantity on non-PO purchase items.
- Fixed an issue with updating [Contact](#) objects where email value was cleared if not explicitly set.
- Added the following [Error Codes](#): 880, 881.

March 19, 2011

- Extended coverage to include the following object types and properties.

Object	Properties
CustField	never_copy
Task	category_1id, category_2id, category_3id, category_4id, category_5id

- Extended [TargetUtilization](#) object type usage to support target utilization for inactive users.
- Changes to [Customer](#): terms defaults to the default payment terms if not set when adding a new object.
- Changes to [Projectassign](#) and [Projecttaskassign](#): job_codeid can be set to 0 when modifying an object.
- Changes to [Purchase_item](#): date of associated [Fulfillment](#) object is set to date_fulfilled, if specified, or to the current date otherwise.
- Added the following [Error Codes](#): 555, 914, 915

January 22, 2011

- Extended coverage to include the following object types and properties.

Object	Properties
Revenue_recognition_rule_amount	cost_center_id
Task	acct_date
Ticket	externalid
Timesheet	acct_date

- Added custom field support for [Paymentterms](#).
- Added support for commands [Modify](#) (XML API) and [modify\(\)](#) (SOAP API) to the [Attachment](#) object type.
- Added support for commands [Delete](#) (XML API) and [delete\(\)](#) (SOAP API) to the [Attachment](#) and [Booking](#) object types.
- Added the following [Error Codes](#): 878, 879, 1105.

November 20, 2010

- Extended coverage to include the following object types and properties.

Object	Properties
Attribute	id, name, attribute_setid, updated, created, notes.
Projectassign	job_codeid
Projectbillingrule	daily_rate_multiplier, job_code_filter
Projectbillingrule	job_codeid
Projectgroup	id, attributes, assigned_users, created, updated, name, notes, active
Projecttaskassign	job_codeid
RevenueContainer	asb_which_slips
Uprate	job_codeid

- Related object lookup when adding or modifying objects using the SOAP API now uses [oaFieldAttribute](#) instead of [oaAttribute](#) to specify the reference fields and the lookup properties. The [oaAttribute](#) object is now used to represent a measurement level for a skill or competency (see [Attribute](#)). See also [Related Object Lookup Using the SOAP API](#).



Important: If you update the OpenAir WSDL in your development environment, replace all references to the `oaAttribute` object with `oaFieldAttribute` in your code. Related object look ups will stop working otherwise.

- Added support for add and modify operations to the [Agreement_to_project](#) object type.
- Added support for delete operations to the [Agreement_to_project](#) and [Entitytag](#) object types.
- Added the following [Error Codes](#): 876, 877, 936, 937, 938, 1104.

September 18, 2010

- Extended coverage to include the following object types and properties.

Object	Properties
Agreement_to_project	agreementid, attribute, customerid, projectid, active, created, updated
Attributeset	id, name, attribute, notes, created, updated
Booking	job_code_id
Customer	sold_to_contact_id
IssueStatus	id, name, attribute, active, created, and updated
Revenue_recognition_transaction	project_billing_rule_id, job_code_id, rate, decimal_hours, hour, minute, revenue_containerid, revenue_stageid, originatingid, offsetsid
Slip	projecttask_type_id, job_code_id, payroll_type_id

- Added the following filter: approved-revenue-recognition-transactions. See [Filtering](#).

July 17, 2010

- Extended coverage to include the following object types and properties.

Object	Properties
Booking	starttime and endtime
Category_<N>	id, name, code, externalid, active, created, updated, and notes
Revenue_recognition_transaction	category_1id, category_2id, category_3id, category_4id, and category_5id

- Added custom field support for [Revenue_recognition_transaction](#).

May 15, 2010

- Extended coverage to include the following object types and properties.

Object	Properties
Agreement	acct_date

Object	Properties
Customerpo	acct_date
Project	attachmentid

- Added the following [Error Codes](#): 871, 872, 873, 874.

March 20, 2010

- Extended coverage to include the following object types and properties.

Object	Properties
<i>all except Address, Date, oaFieldAttribute and Module</i>	attributes
Address , Date , oaFieldAttribute and Module	id, name, userid, date, period, currency, cost, cost_typeid, is_accrual, externalid, notes, created, updated
Attachment	attachmentid
Costcategory	id, name, active, notes, created, updated, externalid
Costtype	id, name, active, notes, created, updated, externalid
Envelope	currency_exchange_intolerance
Repeat	id, frequency, every, end, occur_number, how_end, exclude_dow, created, updated
RevenueContainer	cost_centerid
Ticket	attachmentid, currency_exchange_intolerance

- Added SOAP API support for reading and setting custom field values. See [Reading or Setting Custom Field Values Inline](#). Custom fields are returned by default when reading objects using OpenAir SOAP API.
- Added SOAP API support for looking up objects matching a custom field value. See [Look Up and Update Objects Matching a Custom Field Value](#).
- Added SOAP API support for multiple argument objects and combined relational read methods using logical operators. See [Combining Relational Methods Using Logical Operators \(SOAP API\)](#).
- Added SOAP API support for looking up related object by externalid or name when adding or modifying objects. See [Related Object Lookup Using the SOAP API](#).
- Added custom field support for the [Fulfillment](#).
- Added support for add and update operations for the [Schedulerequest](#) object type. Changes to [ImportExport](#): either one of the imported and exported properties is required.
- Changes to [User](#): project_access_node allows more than one node per hierarchy.
- Changes to [Projectbillingrule](#): cost_centerid is not required when updating objects.
- When adding [Attachment](#) objects, the response includes [Attachment](#) object properties.

January 23, 2010

- Extended coverage to include the following object types and properties.

Object	Properties
Booking	owner_id
Company	workscheduleid
Hierarchy	externalid
HierarchyNode	available_as_column, externalid, primary_dropdown_filter, primary_user_filterset
Report	id, userid, name, type, thin_client_context, date_created, email_report, relatedid, created, updated

- Added XML API command [Report](#).
Added support for add and update operations to the [Hierarchy](#) and [HierarchyNode](#) object types.
Added support for the delete operation to the [HierarchyNode](#) object type.
- Fixed an issue when creating [User](#) objects where [UserWorkSchedule](#) was not set.
- Fixed an issue with reading [ImportExport](#) with deleted attribute set to 1 and filter attribute set to not-exported.
- Changed invalid UTF-8 character handling: invalid UTF-8 characters are stripped out instead of converted to decimal numbers. Removed more UTF-8 encoding errors in the server log for accounts not configured for UTF-8.
- Changes to commands [CreateUser](#) (XML API) and [createUser\(\)](#) (SOAP API) to follow the behavior of the OpenAir UI more closely when setting a name for the new [User](#).

November 21, 2009

- Extended coverage to include the following object types and properties.

Object	Properties
Envelope	attachmentid
Project	pm_approver_1, pm_approver_2, pm_approver_3, payroll_type_filter
Resourceprofile	externalid
Resourceprofile_type	externalid
User	update_workschedule, is_user_schedule, workschedule_workdays, workschedule_workhours

- Changes to [User](#):
 - Added support for adding or updating a [UserWorkSchedule](#) when adding or updating a [User](#).
 - Fixed issue with invalid start_date value for added or updated [Entitytag](#) when adding or updating a [User](#).
 - Added support for updating [Entitytag](#) for inactive [User](#).
- Changes to [Slip](#): projectid and customerid validation when adding objects
- Changes to [Projecttask](#): Ability to use default filtering mechanism when reading objects.
- Changes to [ImportExport](#): Added controls to prevent duplicate objects when adding objects.
- Changes to commands [CreateUser](#) (XML API) and [createUser\(\)](#) (SOAP API): Added support for returning error codes.
- Added support for 0 offset in limit read attribute.