OpenAir

SOAP API Reference Guide

October 7, 2023

ORACLE NETSUITE

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=trsif 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

Introduction to OpenAir Web Services

OpenAir provides OpenAir Web Services as a layer for the exchange of OpenAir data between the main site and peripheral programs. These programs include partnered Web sites, OpenAir in-house applications that don't need direct database access, and third party applications indirectly supported through OpenAir. Before you begin using this service, we recommend that you review Best Practices.

The application programming interface (API) is data-centric, but it is not a direct line into the OpenAir database. While it provides access to much of the information on OpenAir, it is a layer of indirection from the actual database structure. OpenAir's database structure may change, but applications that use the API will not need to change. OpenAir Web Services exist in addition to the OpenAir XML API and serve as a wrapper around the XML API, providing the same or very similar functionality. See OpenAir XML API Reference Guide for detailed documentation.

Technology

OpenAir Web Services are based on Simple Object Access Protocol (SOAP), an XML-based convention. The following standards are observed.

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

Since OpenAir Web Services are database-driven, it is important that the information you collect from your users is compatible with the fields in the database. We provide you with the list of commands and data types that are meaningful to an OpenAir database. Through the use of commands and data types provided by the API, and by limiting the values that can be stored in each data type, your data will always be consistent with, and able to be stored within, an OpenAir database.

Target Audience

This document is intended for developers of applications that will connect to the OpenAir Web site.

Overview

The following provides a brief description of what's included in OpenAir SOAP Web Services.

- **Getting Started** — includes steps for what you should do to begin using OpenAir Web Services. It includes general procedures for connecting to the OpenAir API. It also includes specific instructions for integrations based on Apache Axis libraries or .NET Framework with Microsoft Visual Studio as the IDE.
- **Methods** — lists the supported calls in the API and the syntax, use, arguments, and response of each call as well as sample C# code and Java code.
- **Web Services Method Complex Types** - lists OpenAir-specific datatypes that are required for executing requests and reading responses with the SOAP API. Provides datatype properties and methods in which the specific type is used.

- **Custom Fields** — introduces custom fields and provides information for reading custom field values, requesting custom fields for an object, and modifying records for custom field values.
- **OpenAir Complex Types** — lists OpenAir-specific datatypes defining the business object model exposed via the SOAP API. Provides properties and supported methods for each datatype.
- **Setting Application Switches Via the API** — describes company and user-level switches you can set using the API.
- **Code Examples** — provides code examples for specific functions such as creating multiple users.
- **Error Code Listing** — identifies common errors and associated codes.
- **OpenAir Data Dictionary** — provides a reference to the OpenAir Data Dictionary.

> **Note:** To view the OpenAir Data Dictionary, use the following URL: `https://<account-domain>/database/single_user.html`.
>
> - `<account-domain>` is the account specific domain for your account.
> - To view the details of a specific table, append a hash symbol # followed by the table name to the end of the data dictionary URL. For example, use `https://<account-domain>/database/single_user.html#project` to view the details of the Project table.
> - You can access the data dictionary from the OpenAir Help Center using the link in the navigation bar if you have the View Help Center role permission.

- **Best Practices —** provides a guide for preparing for and using the API.

Definitions

The following are definitions used in OpenAir SOAP Web Services.

- **XML** — eXtensible Markup Language
- **API** — Application Program Interface
- **Server** — The OpenAir site that understands the API
- **Client** — Application that talks to Server using the API
- **XML structure** — An XML element that contains other XML elements
- **OA** — Abbreviation for OpenAir
- **SOAP** — Simple Object Access Protocol
- **XSD** — XML Schemas specification
- **WSDL** — Web Service Description Language

Error Handling

There are two types of errors returned by the API. They are SOAP Fault Errors and API Method Level Errors. Each is described as follows.

- **SOAP Fault Errors** — are returned in the following circumstances: incorrect usage, badly formatted SOAP messages, and failed authentication. They result in SOAP Fault messages with a specific error code and often a string description. When a string description is missing, the read method can be used to retrieve the specific description for an error that occurred. (For more information, refer to the read method.)

- **API Method Level Errors** —Errors specific to a query or action are returned as a collection of oaError objects. See Error Code Listing.

> **Note:** String errors are not guaranteed to be returned. If oaError object's string is empty, look up the error code description by issuing a read method for the “Error” object.

Date Format

Most date fields use the following date format: YYYY-MM-DD HH:MM:SS

Limits

Currently there are four types of usage limits that are enforced in OpenAir Web Services.

- There is a limit of 1000 records that can be requested at one time. Use the Attributes attribute on the read method ReadRequest argument to limit the amount of records being returned and request records in batches. If read is used without the “limit” attribute, an error will be returned.
- There is a limit of 1000 objects any method can accept, so if you need to use add, upsert, modify, createUser, or submit methods, make sure to load the records in batches. The server will return an error if more objects are specified.
- There is a frequency limit of daily transactions allowed for each account.
- There is a frequency limit of transactions allowed for each 60-second interval for each account.

OpenAir will send a warning email when you are approaching your API limits.

Managing Your Account Frequency Limits

There are several ways you can track your usage limits in OpenAir:

- Use the **Remaining Limit** XML command. This command gives you the status of your 24–hour limit. Send this command at various times during your integrations to see where you are sending the highest volume of requests. See the **Remaining Limit** section of the **Other Features** chapter in the OpenAir XML API Reference Guide. Please note that SOAP does not have an equivalent Remaining Limit command.
- Contact OpenAir Customer Support and request the **Enable web services log report feature**. This feature creates a report called “Web services — Web services logs” which you can set up to show every API request made after the feature was enabled. You can find this report by navigating to Reports > Detail > Web services > Web services logs or by searching for “Web services logs” in the Report Management interface. Reviewing this report can help identify areas of potential usage limit overages.
  - Each row in the Web services log represents **one** request and response pair.
  - Records in the Web services log are only available for seven days after they are created.

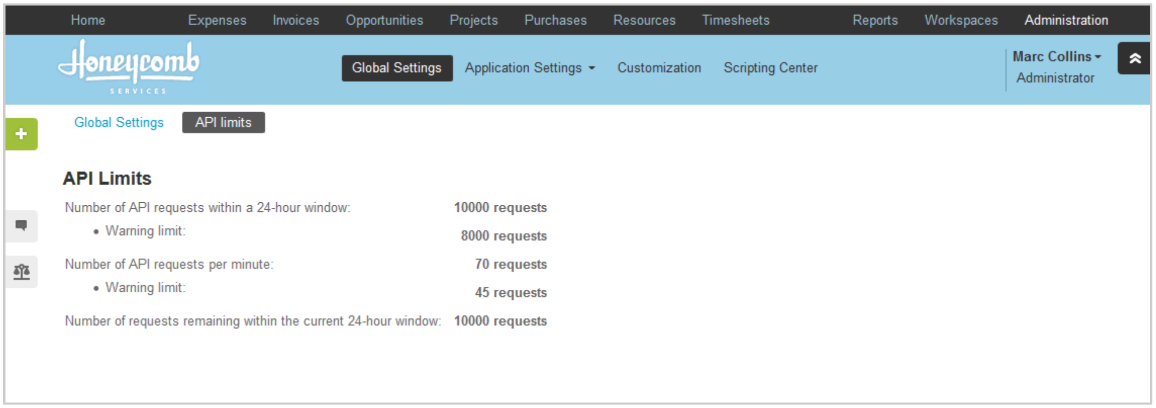
Note: This feature includes an optional component, which may be enabled to help troubleshoot any issues with the add-on services provided by OpenAir.

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.

- Go to Administration > Global Settings > Account > API Limits to view the API request limits that are currently set for your account and the number of requests remaining within the current 24–hour window.



API Limits

Number of API requests within a 24-hour window:	10000 requests
• Warning limit:	8000 requests
Number of API requests per minute:	70 requests
• Warning limit:	45 requests
Number of requests remaining within the current 24-hour window:	10000 requests

After a frequency limit is reached for either daily transactions or a 60-second interval, our web servers will respond with 403 “access denied” error until the end of the period. The best way to avoid breaching these limits is to make sure all records and fields are requested by batching many commands into one request call. Also, avoid making any requests within a loop. See Best Practices

Contact OpenAir Customer Support (see Creating a Support Case) with any further questions about frequency limits. Please note that as each customer’s integration designs and needs vary, OpenAir cannot make specific recommends to reduce your API requests. Please work with your company’s integration team, follow the best practices in this guide and use the tools described here to see where you can make improvements.

If you require further assistance, you can contact your OpenAir account manager and ask about a Professional Services engagement to help you reduce your API requests.

Getting Started

The following are instructions for what you should do to begin using OpenAir Web Services.

How to Begin

Step 1: Obtain OpenAir API Access

Contact OpenAir Customer Support or your OpenAir account manager to request API access. See Creating a Support Case for instructions. When access is granted, you will receive an API namespace and an API key. These are the two pieces of information required for API access in addition to your regular OpenAir login credentials.

The namespace and key attributes are used to verify that the request is coming from a valid partner that has permission to use our API. You will not be able to access an account with just the namespace and the key. You will also need to know the Company ID, User ID, and Password of the account.

Step 2: Generate the OpenAir Web Service WSDL

Point your browser or development environment to the following URL and the WSDL file will be automatically generated: https://www.openair.com/wsdl.pl

Replace the server name with name of the server you're using for testing or production if it is different than our production server. **We strongly recommend that you use a test account on one of our testing servers before running in production!**

You can also append Style=Document to view the Document style with literal use version of the OpenAir WSDL. The SOAP body can be validated against XML schema documents.

To access the Document/Literal Binding version of the OpenAir WSDL:

1. Go to www.openair.com/wsdl.pl?style=document

You can also use the Style=Document setting when exporting an account-specific WSDL.

To access the Document/Literal Binding version of your account-specific WSDL:

1. Go to Administration > Global Settings > Account > Integration: Import/Export and click **Account specific WSDL**. Your account-specific WSDL will open in your browser.
2. In your browser’s URL field, type **;style=document** (including the semi-colon) to the end of the URL and press **Enter**. The Document Binding version of your account-specific WSDL will open in your browser.

You can use the Document/Literal Binding version of your WSDL with custom integrations which support or require the Document Binding format, including Oracle Integration Cloud (OIC) or Oracle Integration Cloud Service (ICS).

Important: The document/literal binding version of the OpenAir WSDL or your account-specific WSDL supports only the following methods: read, add, modify, delete, and createUser.

Step 3: Import the WSDL File into Your Development Platform

After you have generated the WSDL file, you import it into your development platform. Refer to Instructions for Apache Axis Libraries and Instructions for Microsoft Visual Studio.

Step 4: Set up an OpenAir Service URL

Configure your integration to connect to an OpenAir SOAP API service at the following URL:

```
https://<account-domain>/soap
```

Note: The URL for OpenAir services includes the domain for your OpenAir account `<account-domain>`. This account-specific domain is the first part of the URL visible in the address bar of your browser **after** you log into the OpenAir web application. The URL may also include the specific path for the OpenAir service you are accessing `<service-path>`.

```
https://<account-domain>/<service-path>
```

- The URL must start with `https://` — a secure communication protocol is required.
- The account-specific domain contains a unique account identifier `<company-id>`. The account identifier is typically based on your OpenAir Company ID.
- The account-specific domain name depends on the account type:
  - Production account-specific domain: `<company-id>.app.openair.com`
  - Sandbox account-specific domain: `<company-id>.app.sandbox.openair.com`
  - Demo account-specific domain: `<company-id>.app.demo.openair.com`

Instructions for Apache Axis Libraries

Note: OpenAir Web Services has been tested and is verified to be compatible with versions 1.3 and 1.4 of the Apache Axis SOAP libraries. It is **NOT** compatible with the Axis2 libraries.

Java client-binding objects can be used to access the API. They function as proxies for server-side equivalents. You generate these objects from your WSDL file before you can begin using the API. Ensure the following:

- You have generated the Java objects from the OpenAir WSDL file.
- For Apache Axis, you are using the WSDL2Java utility. Refer to: http://ws.apache.org/axis/java/user-guide.html#WSDL2JavaBuildingStubsSkeletonsAndDataTypesFromWSDL.

- You have previously installed Axis on your system. The JAR files need to be referenced in your classpath. For more information and to obtain a copy of the Axis libraries and tools, refer to http://ws.apache.org/axis.

To proceed, refer to the following information:

- Syntax for WSDL2Java:

  java —classpath path to JAR filename `org.apache.axis.wsdl.WSDL2Java -a wsdlURL`
- -a switch generates code for elements whether they are referenced or not. For more information, refer to http://axis.apache.org/axis/java/reference.html
- JAR files in more than one location should have a semicolon separating them.

  For example, if the Axis 1.4 JAR files are installed in `C:\axis-1_4`, and the WSDL is located at `https://www.openair.com/wsdl.pl`, the following command generates all needed stub objects and proxies to access OpenAir Web Services:

```
java -cp c:\axis-1_4\lib\axis.jar;
c:\axis-1_4\lib\axis-ant.jar;
c:\axis-1_4\lib\commons-logging-1.0.4.jar;
c:\axis-1_4\lib\commons-discovery-0.2.jar;
c:\axis-1_4\lib\jaxrpc.jar;
c:\axis-1_4\lib\log4j-1.2.8.jar;
c:\axis-1_4\lib\wsdl4j-1.5.1.jar;
org.apache.axis.wsdl.WSDL2Java -a https://www.openair.com/
wsdl.pl
```

  A set of folders and Java source code files are generated and placed in the same directory from which the command is run. Once compiled, you can include them when you are creating your client application.

  Also note that adding the –p switch will allow you to specify your own package namespace instead of the default namespace of com.soaplite.namespaces.perl.

Wizard-based tools can be used instead of the command line for this process in most Java development environments.

Periodically, additional fields are added to the OpenAir WSDL document and associated SOAP objects. These changes may not be automatically reflected in stubs generated using the Axis WSDL2Java tool and may cause an exception in client code. To handle this situation, the following workaround can be used:

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

```
public class MyDeserializer extends BeanDeserializer {
     public MyDeserializer(java.lang.Class javaType,
      javax.xml.namespace.QNamexmlType,org.apache.axis.description.TypeDesc
      typeDesc)
     {
      super(javaType, xmlType, typeDesc);
     }

     public SOAPHandler onStartChild(String arg0, String arg1, String
      arg2, Attributes arg3, DeserializationContext arg4) throws
      SAXException
      {
          // TODO Auto-generated method stub
          try{
               __return super.onStartChild(arg0, arg1, arg2,arg3, arg4);
          }catch (SAXException e){
              __return null;
          }
      }
}
```

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

```
/**
 * Get Custom Deserializer
 **/
public static org.apache.axis.encoding.Deserializer getDeserializer(
       java.lang.String mechType,
       java.lang.Class _javaType,
       javax.xml.namespace.QName _xmlType) {
    return
      new MyDeserializer(
        _javaType, _xmlTpe, typeDesc);
}
```

Instructions for Microsoft Visual Studio

Visual Studio languages use classes to access the API. Objects generated from these class definitions function as proxies for their server-side equivalents. You must generate these classes from the OpenAir WSDL file before you can begin using the API. Proxy classes can be created using Visual Studio IDE or a command-line utility. Proxy classes can be added to a new or an existing project opened in Visual Studio.

An XML client is an application that talks to the OpenAir server using the OpenAir SOAP Web service. That client may be a Web or Windows Forms application, or even another Web service. When you access OpenAir Web Services using SOAP, infrastructure code is managed by proxy classes and the .NET Framework.

To access OpenAir Web Services, begin by adding a Web reference, identifying the namespace, creating an instance of the proxy class, and accessing methods of the class. For more information on supported calls in the API, refer to Methods.

To Add a Web Reference:

1. On the Project menu, select Add Web Reference.
2. In the Add Web Reference dialog box, type the URL:

   https://www.openair.com/wsdl.pl
3. Click Go.

   Information about the OpenAir Web Services displays.
4. Enter “OA” in Web Reference name edit box.
5. Click Add Reference.

   A Web reference is added and a proxy classes are generated that interfaces between your application and OpenAir Web Services.

   Note: For more information about adding a Web reference, refer to “Adding and Removing Web References” in the Microsoft Visual Studio documentation.

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.

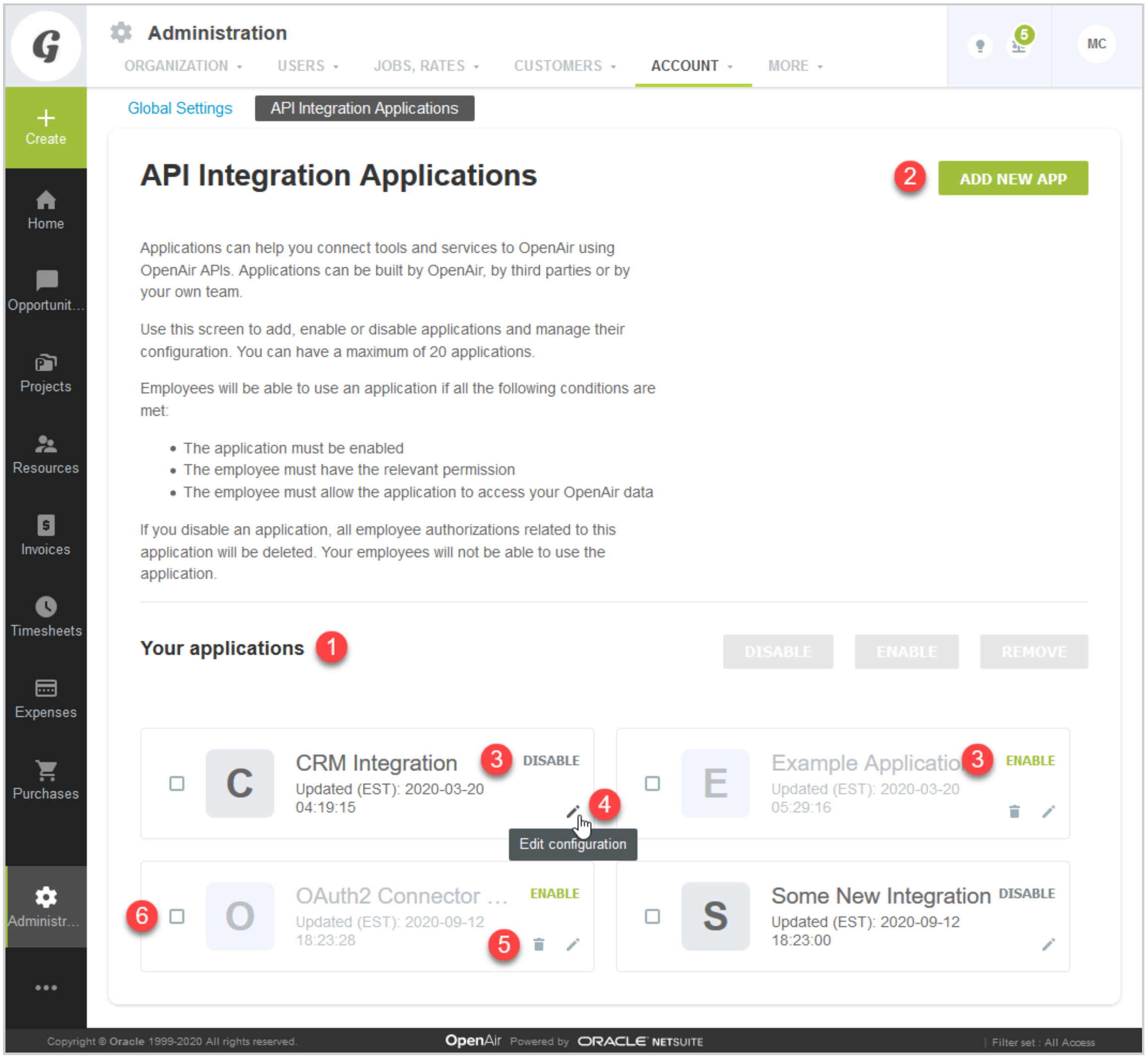
Managing API Integration Applications in OpenAir

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).



API Integration Applications

Applications can help you connect tools and services to OpenAir using OpenAir APIs. Applications can be built by OpenAir, by third parties or by your own team.

Use this screen to add, enable or disable applications and manage their configuration. You can have a maximum of 20 applications.

Employees will be able to use an application if all the following conditions are met:

- The application must be enabled
- The employee must have the relevant permission
- The employee must allow the application to access your OpenAir data

If you disable an application, all employee authorizations related to this application will be deleted. Your employees will not be able to use the application.

Your applications

CRM Integration — DISABLE — Updated (EST): 2020-03-20 04:19:15

Example Applicatio — ENABLE — Updated (EST): 2020-03-20 05:29:16

OAuth2 Connector ... — ENABLE — Updated (EST): 2020-09-12 18:23:28

Some New Integration — DISABLE — Updated (EST): 2020-09-12 18:23:00

Copyright © Oracle 1999-2020 All rights reserved. OpenAir Powered by ORACLE NETSUITE | 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.

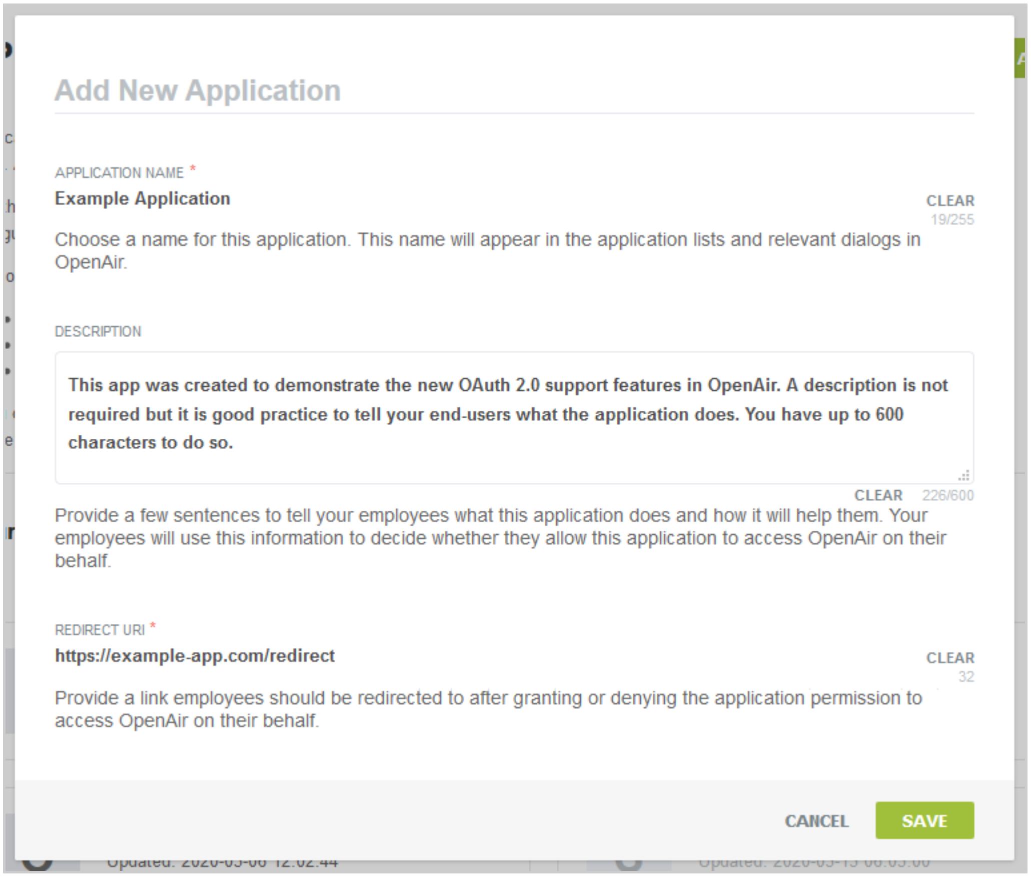
If you misplace the Client Secret, you can edit the application configuration and generate a new Client Secret for the application.

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.

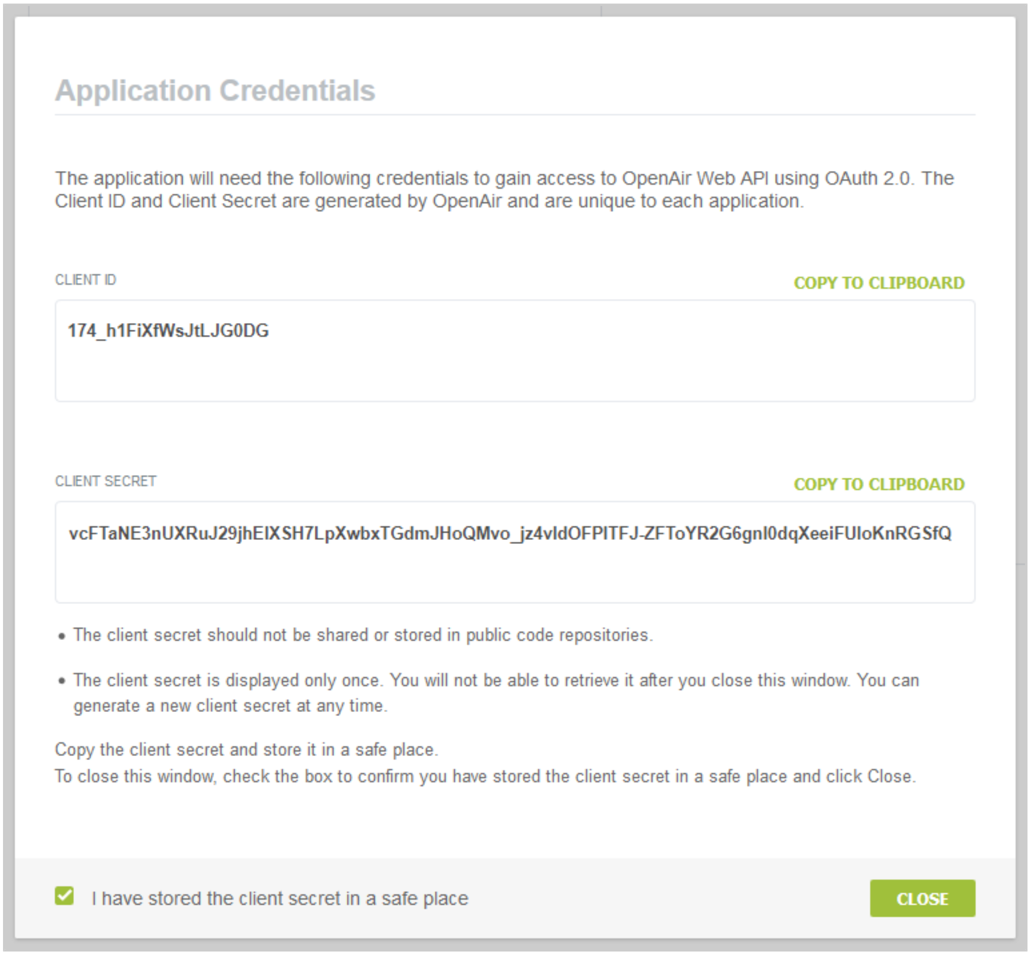

Add New Application

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
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.
CLEAR 226/600
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.

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

vcFTaNE3nUXRuJ29jhElXSH7LpXwbxTGdmJHoQMvo_jz4vldOFPITFJ-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

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:

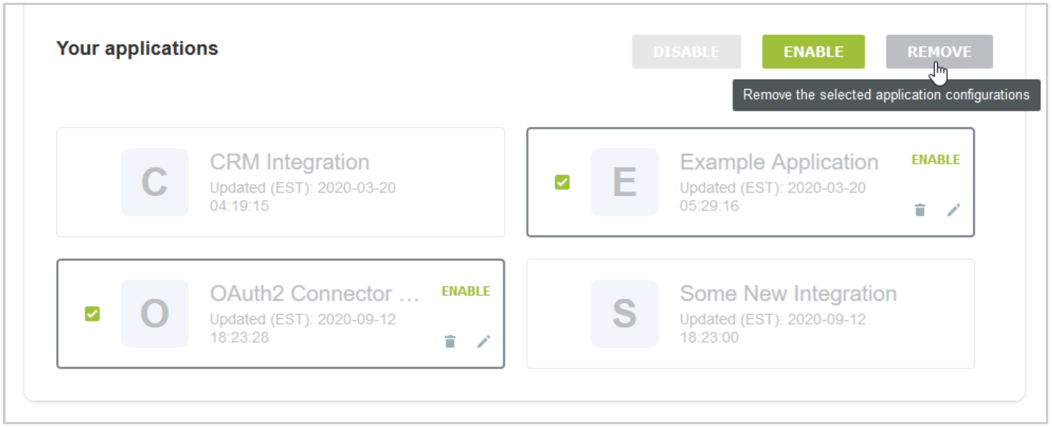
1. Go to Administration > Global settings > Account > API Integration Applications.
2. Click **ENABLE** or **DISABLE** in the top right corner of the corresponding box. A confirmation dialog box appears.
3. Click **ENABLE** or **DISABLE** to enable or disable the application. Click **Cancel** to cancel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


Your applications
DISABLE
ENABLE
REMOVE
Remove the selected application configurations
C
CRM Integration
Updated (EST): 2020-03-20 04:19:15
E
Example Application
ENABLE
Updated (EST): 2020-03-20 05:29:16
O
OAuth2 Connector ...
ENABLE
Updated (EST): 2020-09-12 18:23:28
S
Some New Integration
Updated (EST): 2020-09-12 18:23:00

3. Click **ENABLE**, **DISABLE**, or **REMOVE** to perforrm 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.

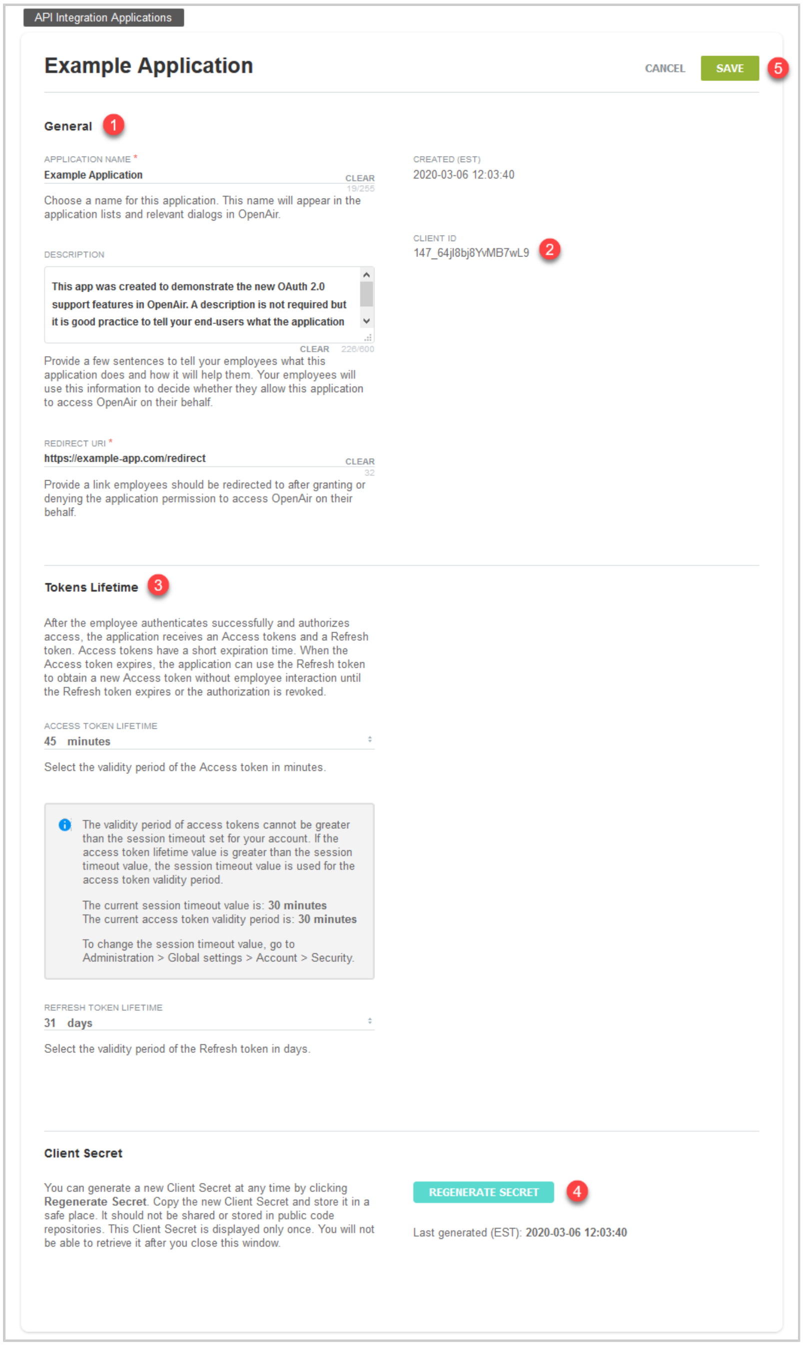
     > **Note:** Before the October 2021 OpenAir release, you could set the refresh token lifetime to values from 1 to 24 hours in one–hour increments. Values for the refresh token lifetime set before the October 2021 OpenAir release show in days (decimal values) instead of hours

   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.

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.



Example Application

General

APPLICATION NAME *
Example Application
19/255
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

CLIENT ID
147_64jl8bj8YvMB7wL9

DESCRIPTION
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
226/600
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
32
Provide a link employees should be redirected to after granting or denying the application permission to access OpenAir on their behalf.

Tokens Lifetime

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.

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.

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

Auditing and Managing OAuth 2.0 Authorizations

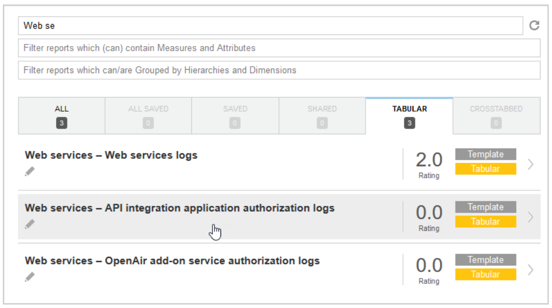
Account administrators can use web services reports to audit and revoke authorizations granted by OpenAir users to integration applications utilizing OAuth 2.0 to connect to OpenAir data.

- **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.



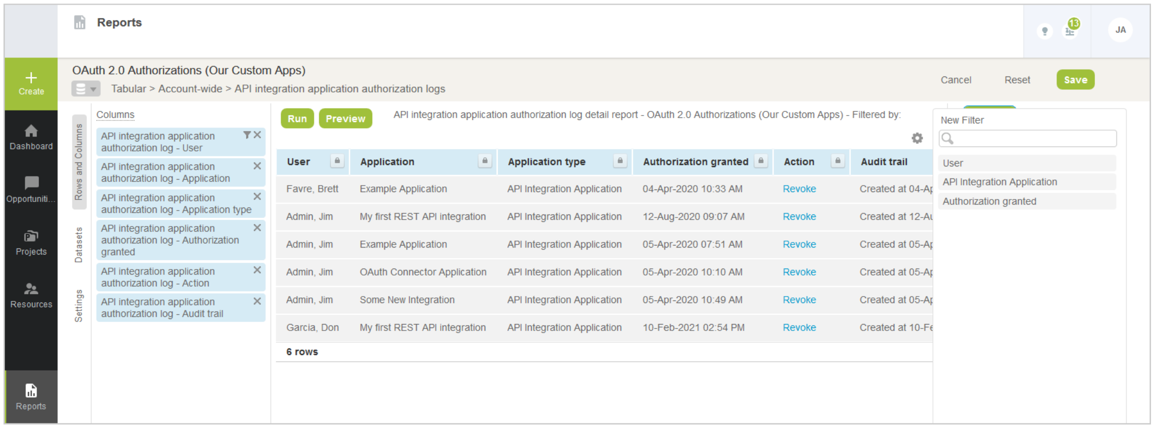
Web se

Filter reports which (can) contain Measures and Attributes

Filter reports which can/are Grouped by Hierarchies and Dimensions

ALL	ALL SAVED	SAVED	SHARED	TABULAR	CROSSTABBED
3	0	0	0	3	0

Web services – Web services logs	2.0 Rating	Template Tabular
Web services – API integration application authorization logs	0.0 Rating	Template Tabular
Web services – OpenAir add-on service authorization logs	0.0 Rating	Template Tabular



OAuth 2.0 Authorizations (Our Custom Apps)

Tabular > Account-wide > API integration application authorization logs

Columns
- API integration application authorization log - User
- API integration application authorization log - Application
- API integration application authorization log - Application type
- API integration application authorization log - Authorization granted
- API integration application authorization log - Action
- API integration application authorization log - Audit trail

API integration application authorization log detail report - OAuth 2.0 Authorizations (Our Custom Apps) - Filtered by:

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-Ap
Admin, Jim	My first REST API integration	API Integration Application	12-Aug-2020 09:07 AM	Revoke	Created at 12-Au
Admin, Jim	Example Application	API Integration Application	05-Apr-2020 07:51 AM	Revoke	Created at 05-Ap
Admin, Jim	OAuth Connector Application	API Integration Application	05-Apr-2020 10:10 AM	Revoke	Created at 05-Ap
Admin, Jim	Some New Integration	API Integration Application	05-Apr-2020 10:49 AM	Revoke	Created at 05-Ap
Garcia, Don	My first REST API integration	API Integration Application	10-Feb-2021 02:54 PM	Revoke	Created at 10-Fe

6 rows

New Filter
- User
- API Integration Application
- Authorization granted

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.

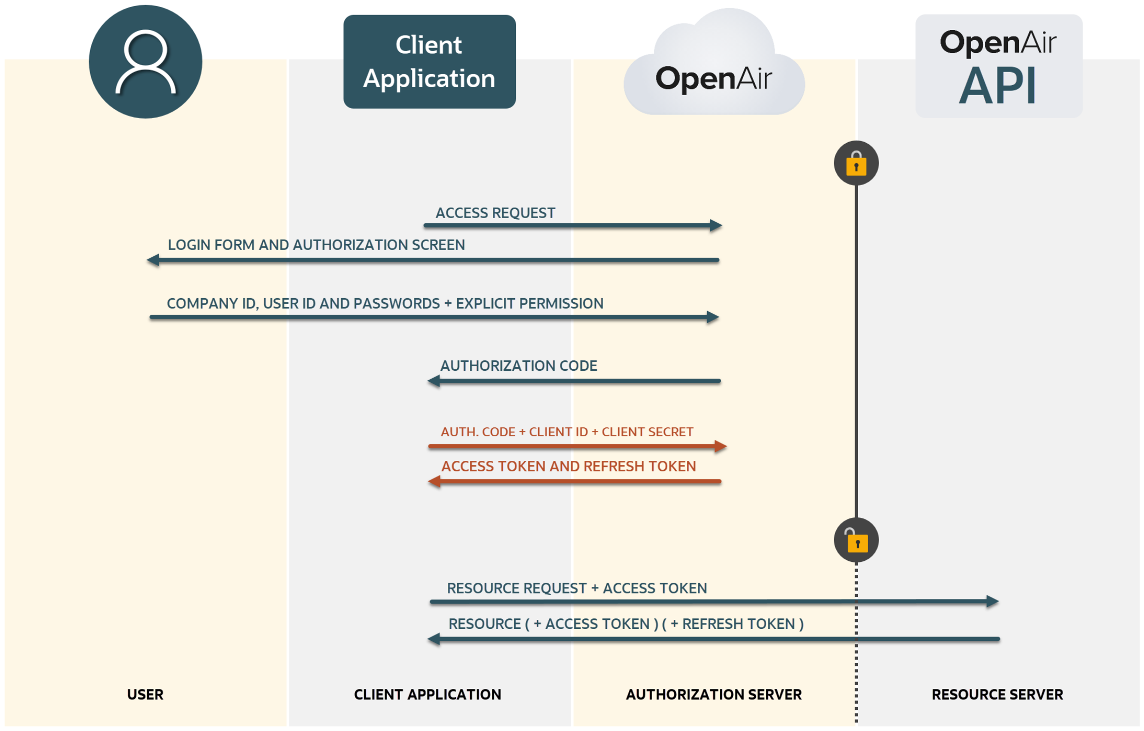
The authorization code flow includes the following steps:

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.


Client Application
OpenAir
OpenAir API
ACCESS REQUEST
LOGIN FORM AND AUTHORIZATION SCREEN
COMPANY ID, USER ID AND PASSWORDS + EXPLICIT PERMISSION
AUTHORIZATION CODE
AUTH. CODE + CLIENT ID + CLIENT SECRET
ACCESS TOKEN AND REFRESH TOKEN
RESOURCE REQUEST + ACCESS TOKEN
RESOURCE (+ ACCESS TOKEN) (+ REFRESH TOKEN)
USER
CLIENT APPLICATION
AUTHORIZATION SERVER
RESOURCE SERVER

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.

Note: The URL for OpenAir services includes the domain for your OpenAir account `<account-domain>`. This account-specific domain is the first part of the URL visible in the address bar of your browser **after** you log into the OpenAir web application. The URL may also include the specific path for the OpenAir service you are accessing `<service-path>`.

```
https://<account-domain>/<service-path>
```

- The URL must start with `https://` — a secure communication protocol is required.
- The account-specific domain contains a unique account identifier `<company-id>`. The account identifier is typically based on your OpenAir Company ID.
- The account-specific domain name depends on the account type:
  - Production account-specific domain: `<company-id>.app.openair.com`
  - Sandbox account-specific domain: `<company-id>.app.sandbox.openair.com`
  - Demo account-specific domain: `<company-id>.app.demo.openair.com`

- 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. `https://<account-domain>/login/oauth2/v1/authorize?<query-string>` 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. `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.

Request parameter	Description
`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. ■ 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`	The authorization code issued by OpenAir. ■ 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=JTlQ43UvYDKbhI_SpEWsIE_bTpbou2-
  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 `grant_type` parameter is `authorization_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.
`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:

```
POST /login/oauth2/v1/token HTTP/1.1
Host: company-id.app.openair.com
Authorization: Basic MTc0X2gxRmlYZldzSnRMSkcwREc6dmNGVGFORTNuVVhSdUoyOWpoRWxYU0g3THBYd2J4VEdkbUpIb1FNdm9fano0dmxkT0ZQSVRGSi1aRl
RvWVIyRzZnbmwwZHFYZWVpRlVJb0tuUkdTZlEK
Content-Type: application/x-www-form-urlencoded
code=JTlQ43UvYDKbhI_SpEWsIE_bTpbou2-kYeeLtKiMuR1iqZ3W3roqM4rmRC8fFCOJtBI6a85AnJPefx2szW9g4jCY&redirect_uri=https%3A%2F%2Fexam
ple-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.

JSON object properties	Description
`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):

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache
{
      "refresh_token":"WGxpeGNVTm1mNGhaS2E1djFQQ2twV1pKcWpOU0pXUkhZUU5oMTR1MFU5OUtLY3N1NkZKOG9SMHp4UnNuMjYyRTJGcm9NVUo5OWxEND
FzcW5WSDFsUEhoSF8xNzQ",
      "expires_in":900,
      "access_token":"eNNJ1GXD25-6IUylF6RZT33HqhoqSAAK53F0kxT62fBoKreDoc8Y_-Gnk2lUIqNbhwguHnxDtxUsJMY6NrDoiBnd",
      "token_type":"bearer"
}
```

If the request fails, the token endpoint returns a JSON object with the `error` and `error_descrtiption` 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 token if an API request returns an authentication failed error.

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
`grant_type`	The value of the `grant_type` parameter is `refresh_token`. It tells the token endpoint that the client application is requesting to refresh an access token.
`refresh_token`	A valid refresh_token. Refresh tokens are valid for the period configured in OpenAir for the application. See Application Configuration.
`scope`	(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.
`redirect_uri`	The valid redirect URI.
`client_id`	The public identifier for the client application.
`client_secret`	The client secret for the application.

Example POST request:

```
POST /login/oauth2/v1/token HTTP/1.1
Host: company-id.app.openair.com
Authorization: Basic MTc0X2gxRmlYZldzSnRMSkcwREc6dmNGVGFORTNuVVhSdUoyOWpoRWxYU0g3THBYd2J4VEdkbUpIb1FNdm9fano0dmxkT0ZQSVRGSi1aRl
RvWVIyRzZnbmwwZHFYZWVpRlVJb0tuUkdTZlEK
Content-Type: application/x-www-form-urlencoded
refresh_token=WGxpeGNVTm1mNGhaS2E1djFQQ2twV1pKcWpOU0pXUkhZUU5oMTR1MFU5OUtLY3N1NkZKOG9SMHp4UnNuMjYyRTJGcm9NVUo5OWxENDFzcW5WSDFsUE
hoSF8xNzQ&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
`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.

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):

```
HTTP/1.1 200 OK
Content-Type: application/json
Cache-Control: no-store
Pragma: no-cache
{
      "refresh_token":"N25RdE82N0FIMnBDRTQ1d3hITHN6dXRwQ3dNdzVHWHMydWd3WWFqUXMwMWgrcTB3UHBFQ3VLaUZQTlRuR0J3U09LaWcvWWJ2UHpPS2hWUUtx
QlJCMzBHVF8xNzQ",
      "expires_in":900,
      "access_token":"pWprqUHkgaOWai7fSjaNaN5ywpDr7a7W6hLiq-b1vBBAT48zxAPTyy-wpTNxyfwJieQzZ5E1HEOsG1hNtwJpILR5",
      "token_type":"bearer"
}
```

If the request fails, the token endpoint returns a JSON object with the `error` and `error_descrtiption` 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`	The authorization code must be valid. Possible reasons include: - **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 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`	The refresh token sent in the request is not valid. Possible reasons include: - **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`	Tokens are issued for a specific `scope`. The `scope` of the token requested must be within the `scope` of the token sent in the

error	error_description	grant_type	Reason
			request. For example, if the scope of the original token includes both `xml` and `rest`, the scope of the access token requested can be `rest` only.
access_denied	Authorization failed		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.
access_denied	API access via OAuth2 is disabled		OpenAir API access is not enabled for the OpenAir account.

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 the help topic 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**.

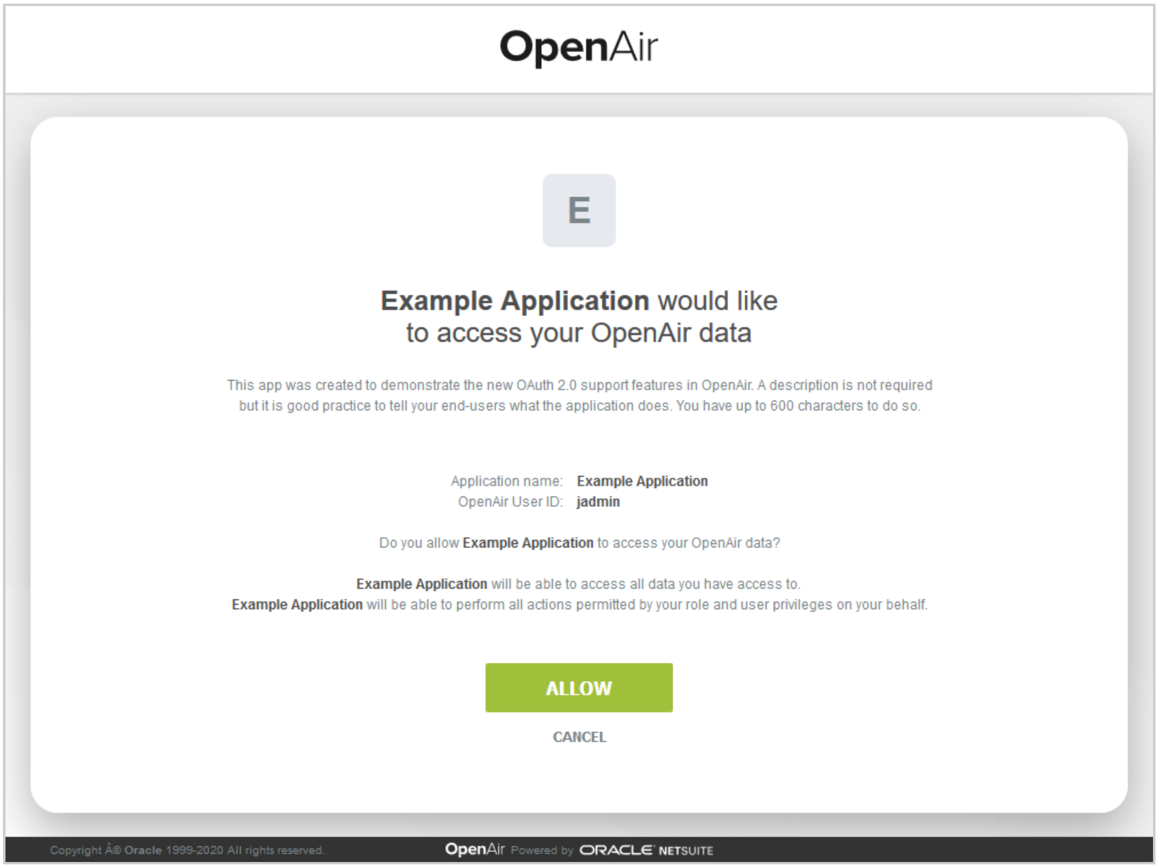
   An authorization screen will appear indicating that the application `<application name>` would like to access your OpenAir data.
3. Read the content of the authorization screen attentively. It should describe what the application does and how it will help you. It should also say what the application 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.



Example Application would like
to access your OpenAir data

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.

Application name: **Example Application**
OpenAir User ID: **jadmin**

Do you allow **Example Application** to access your OpenAir data?

Example Application will be able to access all data you have access to.
Example Application will be able to perform all actions permitted by your role and user privileges on your behalf.

ALLOW
CANCEL
Copyright Â© Oracle 1999-2020 All rights reserved.
OpenAir Powered by ORACLE NETSUITE

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.

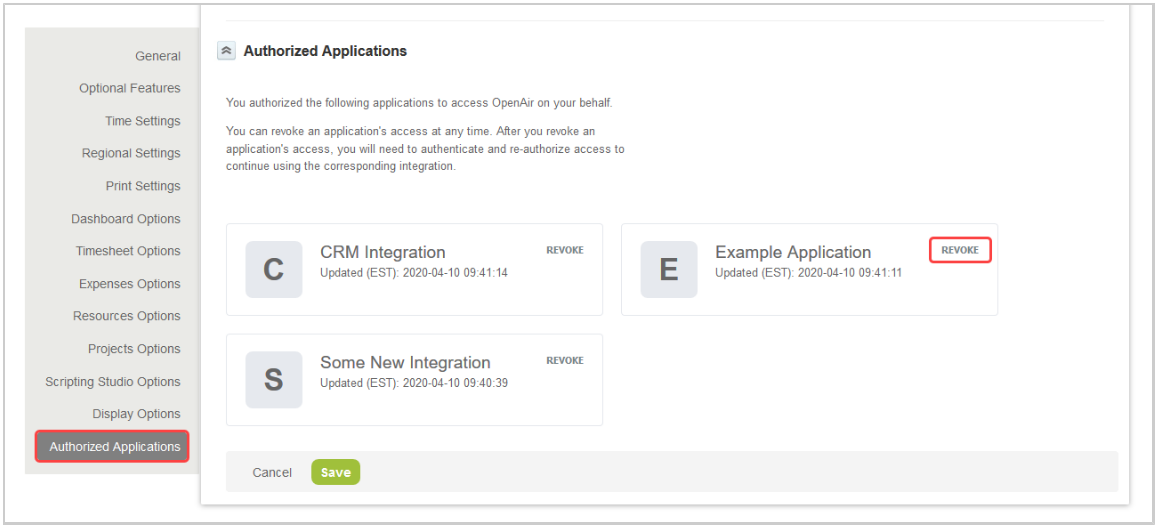
- If you have authorized an application and this application is disabled by an administrator, the application will no longer be able to interact with OpenAir.
- If an administrator enables this application again, you will need to give this application your explicit permission again before you can continue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



Authorized Applications

You authorized the following applications to access OpenAir on your behalf.

You can revoke an application's access at any time. After you revoke an application's access, you will need to authenticate and re-authorize access to continue using the corresponding integration.

C CRM Integration REVOKE
Updated (EST): 2020-04-10 09:41:14

E Example Application REVOKE
Updated (EST): 2020-04-10 09:41:11

S Some New Integration REVOKE
Updated (EST): 2020-04-10 09:40:39

Methods

The following are supported calls in the API. Click on a call name to see the syntax, use, and code samples for that call.

Operation	Description
login	Use this function to authenticate. It returns a valid sessionId which can be used for successive calls.
version	Use this function to request the current version of an application.
read	Use this function to read data from OpenAir.
add	Use this function to add data to OpenAir.
upsert	Use this function to add or modify data to OpenAir based on a lookup attribute.
createAccount	Use this function to create OpenAir accounts.
createUser	Use this function to create OpenAir users.
submit	Use this function to submit OpenAir entities for approval.
makeURL	Use this function to create a valid URL to an OpenAir page specified.
modify	Use this function to modify data in OpenAir.
delete	Use this function to delete data in OpenAir.
whoami	Use this function to get the currently logged in OpenAir user.
servertime	Use this function to get the current server time oaDate object.
logout	Use this function to log out of a session.
approve	Use this function to approve an Envelope, Invoice, or Timesheet submitted for approval.
reject	Use this function to reject an Envelope, Invoice, or Timesheet submitted for approval.
unapprove	Use this function to unapprove a previously approved Envelope, Invoice, or Timesheet.

login

Syntax

```
LoginResult lr = _svc.login(loginParams);
```

Use

The login call is used to initiate a session which can be used for making successive calls to the API. sessionId which is returned as part of the LoginResult object is the unique identifier for this user's session. It can be invalidated by calling logout.

Arguments

Name	Type	Description
login	LoginParams	LoginParam object

Response

`LoginResult`

Sample Code - C#

```
// Create service stub
OAirServiceHandlerService _svc = new OAirServiceHandlerService();

// create LoginParam object
LoginParams loginParams = new LoginParams();
loginParams.api_namespace = "my namespace";
loginParams.api_key = "********";
loginParams.company = "company name";
loginParams.user = "username";
loginParams.password = "password";
loginParams.client = "my client name";
loginParams.version = "1.0";
LoginResult loginResult = _svc.login(loginParams);

// Create a new session header object
// Add the session ID returned from the login
_svc.SessionHeaderValue = new SessionHeader();
_svc.SessionHeaderValue.sessionId = loginResult.sessionId;
```

Sample Code - Java

```
// create our login parameters
LoginParams lp = new LoginParams();
lp.setUser("username");
lp.setPassword("password");
lp.setCompany("company name");
lp.setApi_namespace("my namespace");
lp.setApi_key("********");
lp.setClient("my client name");
lp.setVersion("1.0");

// set the service URL from our arguments
OAirServiceHandlerServiceLocator locator = new
OAirServiceHandlerServiceLocator();
locator.setOAirServiceAddress("https://my-account-domain.app.openair.com/soap");

// now login
OAirServiceSoapBindingStub binding =
(OAirServiceSoapBindingStub)locator.getOAirService();
LoginResult loginResult = binding.login(lp);

// Create a new session header object
// Add the session ID returned from the login
SOAPHeaderElement header = new SOAPHeaderElement("https://
my-account-domain.app.openair.com/OAirService", "SessionHeader");
SOAPElement node = header.addChildElement("sessionId");
node.addTextNode(loginResult.getSessionId());
binding.setHeader(header);
```

version

Syntax

```
VersionResult version = stub.version("My app", "1.1");
```

Use

The version call is used to request the current version of a thin client application supported by Openair.

Arguments

Name	Type	Description
name	string	Name of the application.
number	string	Version number.

Response

VersionResult

Sample Code - C#

```
VersionResult version = stub.version("My app", "1.1");
```

Sample Code - Java

```
VersionResult version = binding.version("My app", "1.1");
```

read

Syntax

```
ReadResult[] results = stub.read(new ReadRequest[2] {read1, read2});
```

Use

Use read command to retrieve data from OpenAir. Read command accepts an array of ReadRequest objects as a parameter and returns an array of corresponding ReadResult objects. Parameter specification is discussed in detail in the ReadRequest section.

Arguments

Name	Type	Description
read	[] ReadRequest	Array of ReadRequest objects

Response

Array of ReadResult objects

C# Read Code Examples

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.

Example I. read equal to C#

Read the fields ID, nickname, updated for users with nickname ‘jsmith’.

```
ReadResult[] results;
ReadRequest rr = new ReadRequest();
rr.type = "User";
rr.method = "equal to"; //return only records that match search
criteria
rr.fields = "id, nickname, updated"; //specify fields to be returned.

//Specify search criteria
oaUser user = new oaUser();
user.nickname = "jsmith";
rr.objects = new oaBase[1] { user }; //pass in one object with search
criteria


int index = 0; //Starting index
const int LIMIT = 1000; //Return maximum of 1000 records per request
do
{
     // Limit attribute is required.
     OA.Attribute attr = new OA.Attribute();
     attr.name = "limit";
     attr.value = String.Format("{0}, {1}", index, LIMIT);


     rr.attributes = new OA.Attribute[] { attr };
     results = _svc.read(new ReadRequest[] { rr });

     if (results != null && results.Length > 0 && results[0].errors !=
null)
     {
      foreach (oaError err in results[0].errors)
      {
        Debug.WriteLine(string.Format("Error {0} - {1}", err.code,
err.text));
     }
    }

    // get next 1000 records
    index += LIMIT;

} while (results[0].objects != null && results[0].objects.Length >
0);
```

Example II. read not equal to C#

Read ID, nickname, and updated fields for users that do not match certain search criteria. For more information, see oaFieldAttribute.

```
ReadRequest rr = new ReadRequest();
rr.type = "User";
rr.method = "not equal to"; //return only records that do not match
search criteria
rr.fields = "id, nickname, updated"; //specify fields to be returned

oaUser user = new oaUser();
user.nickname = "jsmith";
rr.objects = new oaBase[1] { user }; //pass in one object with search
criteria

// Limit attribute is required.
OA.Attribute attr = new OA.Attribute();
attr.name = "limit";
attr.value = "500";
rr.attributes = new OA.Attribute[] { attr };

ReadResult[] results = _svc.read(new ReadRequest[1] { rr });
```

Example III. read custom field definitions C#

Read all custom fields associated with project record type.

```
ReadRequest rr = new ReadRequest();
rr.method = "all";
rr.type = "CustField";

oaCustField cf = new oaCustField();
cf.association = "project"; // custom field association

// Limit attribute is required.
OA.Attribute attr = new OA.Attribute();
attr.name = "limit";
attr.value = "500";
rr.attributes = new OA.Attribute[] { attr };

rr.objects = new oaBase[] { cf };
ReadResult[] results = _svc.read(new ReadRequest[] { rr });
```

Example IV. read custom field values C#

Read custom field values for a given project.

```
ReadRequest rr = new ReadRequest();
rr.method = "custom equal to"; //all custom fields associated with the
project are returned
rr.type = "Project"; //See table of objects that have custom field
support

oaCustomField cf = new oaCustomField();
cf.id = "238"; // ID of the project
rr.objects = new oaBase[] { cf };

OA.Attribute attr = new OA.Attribute();
attr.name = "limit";
attr.value = "500";
rr.attributes = new OA.Attribute[] { attr };

ReadResult[] results = _svc.read(new ReadRequest[] { rr }); //
returns name/value pairs.
```

Note: Review the chapter that addresses Custom Fields and refer to the following code: Example IV. read custom field values C#

Example V. read not exported C#

Read all slips and add a filter to retrieve not yet reported records only.

```
ReadRequest rr = new ReadRequest();
rr.method = "all";
rr.type = "Slip";

OA.Attribute attrLimit = new OA.Attribute();
attrLimit.name = "limit";
attrLimit.value = "500";

OA.Attribute attrFilter = new OA.Attribute();
attrFilter.name = "filter";
attrFilter.value = "not-exported";

rr.attributes = new OA.Attribute[] { attrLimit, attrFilter };

// Tell the server we are filtering on import_export records created
by MY_APP
oaImportExport importExport = new oaImportExport();
importExport.application = "MY_APP";
rr.objects = new oaBase[] { importExport };

ReadResult[] results = _svc.read(new ReadRequest[] { rr });

///Mark the slip with ID = 4 as exported by the application MY_APP on
4/1/2011.
///After doing this, the next read call with not-exported filter
attribute will not export this record.
oaImportExport exportRecord = new oaImportExport();
exportRecord.application = "MY_APP";
exportRecord.type = "Slip";
exportRecord.id = "4";
exportRecord.exported = "2011-04-01 00:00:00";

UpdateResult[] ur = _svc.upsert(new OA.Attribute[] {}, new oaBase[] {
exportRecord });
```

Example VI. read not-exported Envelopes with 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”.

```
ReadRequest rr = new ReadRequest();
rr.method = "all";
rr.type = "Envelope";

//Filter by date range and by the special not-exported flag
OA.Attribute attrFilter = new OA.Attribute();
```

```
attrFilter.name = "filter";
attrFilter.value = "newer-than,older-than,not-exported";

//Name of the field to apply date filter to
OA.Attribute attrField = new OA.Attribute();
attrField.name = "field";
attrField.value = "date_approved,date_approved";

OA.Attribute attrLimit = new OA.Attribute();
attrLimit.name = "limit";
attrLimit.value = "500";

rr.attributes = new OA.Attribute[] { attrFilter, attrField,
attrLimit};

// set newer-than filter date
oaDate dateNewer = new oaDate();
dateNewer.year = "2008";
dateNewer.month = "10";
dateNewer.day = "17";

// set older-than filter date
oaDate dateOlder = new oaDate();
dateOlder.year = "2008";
dateOlder.month = "10";
dateOlder.day = "17";

rr.objects = new oaBase[] { dateNewer, dateOlder };
ReadResult[] results = _svc.read(new ReadRequest[] { rr });
```

Example VII. read with lookup by custom field value C#

```
ReadRequest rr = new ReadRequest();
rr.method = "equal to";
rr.type = "Project";

//Get the first 100 records only
OA.Attribute attrLimit = new OA.Attribute();
attrLimit.name = "limit";
attrLimit.value = "100";
rr.attributes = new OA.Attribute[] { attrLimit };

//Get only records with custom field ProjectStatus set to "Yellow"
//Specify additional values in comma delimited list, e.g.
"Yellow,Green,Red"
//Provide empty string to get records that don't have any value set,
//e.g. filter.ProjectStatus__c = "";
//If the custom field type is Date provide default date to get records
that
//have no value, e.g. filter.ProjectStatus__c = "0000-00-00";
oaProject filter = new oaProject();
filter.ProjectStatus__c = "Yellow";
rr.objects = new oaBase[] { filter };

ReadResult[] results = _svc.read(new ReadRequest[] { rr });
```

Example VIII. read equal to with explicit OR condition C#

Search for all Bookings for users with ID 5 or 6.

```
ReadRequest rr = new ReadRequest();
rr.method = "equal to, or equal to";
rr.type = "Booking";

OA.Attribute attrLimit = new OA.Attribute();
attrLimit.name = "limit";
attrLimit.value = "100";
```

```
rr.attributes = new OA.Attribute[] { attrLimit };

oaBooking book1 = new oaBooking();
book1.userid = "3";

oaBooking book2 = new oaBooking();
book2.userid = "10";

rr.objects = new oaBase[] { book1, book2 };
ReadResult[] results = _svc.read(new ReadRequest[] { rr });
```

Example IX. batch multiple read equal to requests C#

In many cases, the desired records cannot be retrieved in a single read request.

In such situations we recommend batching multiple read requests in a single read call when integration logic allows it.

The code below results in a single trip to server, and therefore is clocked as one single transaction against your daily account limit.

```
int[]idsList = new int[] { 1, 22, 1633, 32, 9, 28, 39 };
ReadRequest[] readRequestsList = new ReadRequest[ IDs.Length];

for (int i = 0; i < 1000 && i <idsList.Length; i++)
{
    ReadRequest rr = new ReadRequest();
         rr.type = "Slip";
      rr.method = "equal to";

      OA.Attribute attrLimit = new OA.Attribute();
      attrLimit.name = "limit";
      attrLimit.value = "1";
      rr.attributes = new OA.Attribute[] { attrLimit };

    oaSlip slipToRead = new oaSlip();
      slipToRead.id =idsList[i].ToString();
      rr.objects = new oaBase[] { slipToRead };

    readRequestsList[i] = rr;
}

ReadResult[] results = _svc.read(readRequestsList);
```

Java Read Code Examples

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.

Example I. read equal to Java

```
// Create a read request for an envelope with internal ID 211
ReadRequest[] reads = new ReadRequest[1];
reads[0] = new ReadRequest();
reads[0].setType("Envelope"); // we are requesting Envelope type
reads[0].setMethod("equal to"); // method to return a specific
envelope
oaEnvelope env = new oaEnvelope();
env.setId("211");
reads[0].setObjects(new oaBase[]{env});
int limit = 1000; // only read 1000 records at a time
int index = 0;// record index to start the read from
```

```java
// add an attribute to our read, specifying the base record # (index)
// and the max # of records to be returned (limit)
Attribute attr = new Attribute();
attr.setName("limit");
attr.setValue(String.format("%1$d", limit));
reads[0].setAttributes(new Attribute[]{attr});

// perform the read
System.out.print("Fetching envelopes...");
ReadResult[] results = binding.read(reads);

// output the results
while(true)
{
     int numRead = 0;

     for (int i = 0; i < results.length; ++i)
     {
          ReadResult r = results[i];
          if (r.getObjects() != null)
          {
               System.out.println("Read " + r.getObjects().length
               + " envelopes\n");
               oaBase[] objs = r.getObjects();
               for (numRead = 0; numRead < objs.length; ++numRead)
               {
                    oaEnvelope envelope = (oaEnvelope)objs[numRead];
                    System.out.println("Envelope name: " +
                    envelope.getName());
                    System.out.println("Envelope number: " +
                    envelope.getNumber());
                    System.out.println("Envelope total: " +
                    envelope.getTotal());
                    System.out.println();
                    // ..etc.
                    index++;
               }
               System.out.println("Read "+numRead+" envelopes");
          }
          else
          {
               System.out.println("Read 0 envelopes\n");
          }
     }
     // if we've read up to the limit, do another read using index as
     our base
     if( numRead == limit )
     {
          System.out.println("Fetching "+ limit +" more envelopes");
          attr.setValue(String.format("%1$d, %2$d", index, limit));
          results = binding.read(reads);
     }
     else
     {
          // no more to read
          break;
     }
}
```

Example II. read not equal to Java

Note that the “limit” attribute is required as illustrated in Example I. read equal to Java.

```java
// Create a read request for envelopes with a non-zero total.
ReadRequest[] reads = new ReadRequest[1];
reads[0] = new ReadRequest();
reads[0].setType("Envelope"); // we are requesting Envelope type
reads[0].setMethod("not equal to"); // method to return a subset of
data based on search criteria.

```

```
// We are searching for an envelope with total not equal to 0.
oaEnvelope obj = new oaEnvelope();
obj.setTotal("0.00");
reads[0].setObjects(new oaEnvelope[] { obj });

// perform the read
System.out.print("Fetching envelopes...");
ReadResult[] results = binding.read(reads);

// output the results
for (int i = 0; i < results.length; ++i)
{
     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("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");
     }
}
```

Example III. read not exported Java

Note that the “limit” attribute is required as illustrated in Example I. read equal to Java.

```
// Read all slips, but add a filter so we only retrieve not-yet-exported
records.
// Below is an example on how to mark items as being exported.
Attribute attr = new Attribute();
attr.setName( "filter" );
attr.setValue( "not-exported" );
ReadRequest read = new ReadRequest();
read.setMethod( "all" );
read.setType( "Slip" );
read.setAttributes( new Attribute[]{attr} );

// Tell the server we're filtering on import_export records created by
MY_APP
oaImportExport importExport = new oaImportExport();
importExport.setApplication( "MY_APP" );
read.setObjects( new oaBase[] { importExport } );

ReadResult[] results = binding.read(new ReadRequest[] { read });

// To modify and read not-yet exported slips
// Mark the slip with ID = 4 as exported by the application
// MY_APP on 4/1/2011. By doing this, we can add a filter attribute to
// our read call that will filter out records exported by MY_APP.
oaImportExport exportRecord = new oaImportExport();
exportRecord.setApplication( "MY_APP" );
exportRecord.setType( "Slip" );
exportRecord.setId( "4" );
exportRecord.setExported( "2011-04-01 00:00:00" );
UpdateResult[] ur = binding.upsert( new Attribute[]{}, new oaBase[] {

exportRecord });
```

Example IV. read date filter Java

Note that the “limit” attribute is required as illustrated in Example I. read equal to Java.

```
// Request envelope records updated based on a certain date.
ReadRequest read = new ReadRequest();
Attribute attr = new Attribute();
attr.setName( "filter" );
attr.setValue( "newer-than,older-than" ); // filter for records in
date range separated by a comma
Attribute field = new Attribute();
field.setName( "field" );
field.setValue( = "updated,updated" ); //one for each date object
separated by a comma
read.setMethod( "all" );
read.setType( "Envelope" );
read.setAttributes( new Attribute[]{attr});

oaDate dateNewer = new oaDate();
oaDate dateOlder = new oaDate();

// set newer than date.
dateNewer.setYear( "2008" );
dateNewer.setMonth( "10" );
dateNewer.setDay( "16" );

// set older than date.
dateOlder.setYear( "2008" );
dateOlder.setMonth( "10" );
dateOlder.setDay( "17" );

read.setObjects(new oaBase[] { dateNewer,dateOlder });
ReadResult[] results = stub.read(new ReadRequest[] { read });
```

add

Syntax

```
UpdateResult[] addResults = stub.add(objects);
```

Use

Use this call to add data to OpenAir. The method returns an error if more than 1000 objects are passed in.

You can use an externalid field as a foreign key and add a record without querying first for an internal ID. The following examples for the modify command provide samples of the syntax: Example II. modify using external_id as foreign key lookup field C# and Example II. externalid as foreign key lookup field Java.

Note: Use createUser method to add OpenAir users (oaUser object).

Arguments

Name	Type	Description
Objects	[] oaBase	Array of oaBase objects

Note: Refer to Adding Records with Inline Custom Field Values for procedures on adding records with inline custom field values.

Response

`Array of UpdateResult objects`

Sample Code — C#

```
//Define a category object to create in OpenAir
oaCategory category = new oaCategory();
category.name = "New Category";
category.cost_centerid = "123";
category.currency = "USD";

//Invoke the add call
UpdateResult[] results = _svc.add(new oaBase[] { category });

//Get the new ID
string newID = results[0].id;
```

Sample Code — Java

```
// Create a category object to send to the service
oaCategory category = new oaCategory();

// Set several properties
category.setName("my new category");

// Add the category to an array of oaBase objects
oaBase[] records = new oaBase[] { category };

// Invoke the add call
UpdateResult[] results = stub.add(records);

// Get the new ID
String newID = results[0].getId();
```

upsert

Syntax

```
UpdateResult[] upsertResults = stub.upsert(attributes, objects);
```

Use

Use this call to add or modify data to OpenAir based on lookup attributes. The method returns an error if more than 1000 objects are passed in.

You can use an externalid field as a foreign key and add a record without querying first for an internal ID. See Example II. modify using external_id as foreign key lookup field C# and Example II. externalid as foreign key lookup field Java.

Arguments

Name	Type	Description
attributes	[] LoginParams	Array of Attribute objects
objects	[] oaBase	Array of oaBase objects

Note: Refer to Adding Records with Inline Custom Field Values for procedures on adding records with inline custom field values.

Response

`Array of UpdateResult` objects

Types of Attributes Used

Name: `lookup`

Value: `name of the field which should be used for lookup`

- If you pass external_id, upsert will attempt to find another record with external_id as specified in the object being upserted.
- If it finds another record, it will do a modify, otherwise the record will be added.

Sample Code — C#

```
//Define a category object to create/update in OpenAir
oaCategory category = new oaCategory();
category.name = "Updated Category";
category.externalid = "555";

// Specify that the lookup is done by external_id and not by (default)
internal ID
OA.Attribute attrLookup = new OA.Attribute();
attrLookup.name = "lookup";
attrLookup.value = "externalid";

// Invoke the upsert call, passing and saving the results in a
UpdateResult object
UpdateResult[] results = _svc.upsert(new OA.Attribute[] { attrLookup
}, new oaBase[] { category });
```

Sample Code — Java

```
// upsert call
// Create a category object to send to the service
oaCategory category = new oaCategory();

// Set several properties
category.setName("SOAP created category 12/4");
category.setExternalid("555");

// Specify lookup attribute
Attribute lookup = new Attribute();
lookup.setName("lookup");
lookup.setValue("externalid");

// Add the account to an array of oaBase objects
oaBase[] records = new oaBase[] { category };

// Invoke the upsert call, passing.
// and saving the results in a UpdateResult object
UpdateResult[] results = stub.upsert(new Attribute[] { lookup },
records);
```

createAccount

Syntax

```
UpdateResult result = stub.createAccount(user, company);
```

Use

Use this call to create OpenAir accounts. This method also creates the first administrative user. The method returns an error if more than 1000 objects are passed in.

Arguments

Name	Type	Description
user	oaUser	oaUser object
company	oaCompany	oaCompany object

Response

`UpdateResult` `object.`

Sample Code - C#

```
// Create a new OpenAir account
oaCompany comp = new oaCompany();
comp.nickname = "New Account";

oaUser user = new oaUser();
user.nickname = "Admin";
user.role_id = "1";
user.password = "%^&*&^";
user.addr_email = "sss@sss.com";
user.account_workscheduleid = "1";

UpdateResult result = _svc.createAccount(user, comp);
```

Sample Code - Java

```
oaCompany comp = new oaCompany();
```

```
oaUser cuser = new oaUser();
comp.setNickname("New Account");

cuser.setNickname("Admin");
cuser.setRole_id("1");
cuser.setPassword("%^&*&^");
cuser.setAddr_email("sss@sss.com");
cuser.setAccount_workscheduleid("1");
UpdateResult result = stub.createAccount(cuser, comp);
```

createUser

Syntax

```
UpdateResult result = stub.createUser(user, company);
```

Use

Use this call to create OpenAir users. The method returns an error if more than 1000 objects are passed in. For procedures in setting a User workschedule, refer to oaUser.

Important: Review the following guidelines:

- 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).
- 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 OpenAir Administrator Guide.

Arguments

Name	Type	Description
user	oaUser	oaUser object
company	oaCompany	oaCompany object

Note: Refer to Adding Records with Inline Custom Field Values for procedures on adding records with inline custom field values.

Response

UpdateResult object.

Sample Code - C#

```
oaCompany comp = new oaCompany();
comp.nickname = "New Account";// specify nickname of the account the
user is being added to.

oaUser newUser = new oaUser();
newUser.nickname = "userA";
newUser.role_id = "1"; // role of administrator.
newUser.password = "*****";
newUser.addr_email = "sss@sss.com";
newUser.account_workscheduleid = "1"; // Associate a valid
workschedule for the user.

UpdateResult result = _svc.createUser(newUser, comp);
```

Sample Code - Java

```
oaCompany comp = new oaCompany();
oaUser cuser = new oaUser();
comp.setNickname("openair"); // specify nickname of the account
the user is being added to.
cuser.setNickname("userA");
cuser.setRole_id("1"); // role of administrator.
cuser.setPassword("******");
cuser.setAddr_email("sss@sss.com");
cuser.setAccount_workscheduleid("1"); // Associate a valid
workschedule for the user.

UpdateResult result = stub.createUser(cuser, comp);
```

submit

Syntax

```
SubmitResult[] results = stub.submit(requests);
```

Use

Use this call to submit OpenAir entities such as bookings, envelopes, invoices, or timesheets for approval. The method returns an error if more than 1000 objects are passed in.

Arguments

Name	Type	Description
requests	[] SubmitRequest	Array of SubmitRequest objects

Response

Array of SubmitResult

Sample Code — C#

```
// submit an envelope for approval
oaEnvelope env = new oaEnvelope();
env.id = "122";

oaApproval appr = new oaApproval();
appr.cc = "help@ddd.com"; // cc approval email to additional contacts
appr.notes = "Approval notes";

SubmitRequest sr = new SubmitRequest();
sr.submit = env;
sr.approval = appr;

SubmitResult[] results = _svc.submit(new SubmitRequest[] { sr });
```

Sample Code — Java

```
// submit an envelope for approval
oaEnvelope env = new oaEnvelope();
env.setId("122");
oaApproval appr = new oaApproval();
appr.setCc("help@ddd.com"); // cc approval email to additional
contacts
appr.setNotes("approval notes");
SubmitRequest sub = new SubmitRequest();
sub.setApproval(appr);
sub.setSubmit( env );

SubmitResult[] results = stub.submit(new SubmitRequest[] { sub });
```

makeURL

Syntax

```
MakeURLResult[] mkresults = stub.makeURL(request);
```

Use

Use this call to create a valid URL to a specified OpenAir page. It requires a valid user login to succeed.

Arguments

Name	Type	Description
request	[] MakeURLRequest	Array of MakeURLRequest object

Currently, the list of valid page strings, with associated applications and arguments, includes:

- default-url

  app= km, ma, pb, rm, pm, ta, te, or tb (Points to the starting page in any one of the applications, which is the page you see when you click on the application link.)

  For example: If you are the administrator and are using pm as the app attribute, the first page is the Projects list in the Projects application. For non-administrative users, it would be the list of tasks to which the user is assigned.)
- **company-settings**

  app= ma (points to Administration > Global Settings)
- **currency-rates**

  app= ma (points to Administration > Global Settings > Organization > Currencies)
- **import-export**

  app= ma (points to Administration > Global Settings > Account > Integration: Import/Export)
- **custom-fields**

  app= ma (points to Administration > Global Settings > Custom Fields)
- **list-reports**

  app= ma (points to Reports > last page accessed)
- **list-customers**

  app= ma (points to Administration > Global Settings > Customers > Customers)
- **list-projects**

  app= pm (points to Projects > Projects)
- **list-prospects**

  app= om (points to Opportunities > Prospects)
- **list-resources**

  app= rm (points to Resources > Resources)
- **list-timesheets**

  app= ta (points to Timesheets > Timesheets > Open)
- **create-timesheet**

  app= ta (points to Timesheets > Create Timesheet)
- **list-timebills**

  app= tb (points to Invoices > Charges)
- **list-invoices**

  app= tb (points to Invoices > Invoices)
- **create-invoice**

  app= tb (points to Invoices > Invoices > Create Invoice)
- **list-envelope-receipts**

app= te (points to Expenses > Envelopes > Receipts)

`arg = oaEnvelope envelope = new oaEnvelope(); envelope.id = <envelope internal id>`

- **list-envelopes**

  app= te (points to Expenses > Expense Reports > Open)
- **create-envelope**

  app= te (points to Expenses > Expense Reports > Create Envelope)
- **create-envelope-receipt**

  app= te (points to Expenses > Expense Reports > Create Receipt
- **dashboard**

  app= ma (points to Dashboard)
- **list-purchase-requests**

  app= po (points to Purchases> Purchase Requests)
- **quick-search-resources**

  app= rm (points to Resources > Quick Search)
- **custom-search-resources**

  app= rm (points to Resources > Custom Search)
- **view-invoice**

  app= tb (displays the invoice with specified internal id)

  `arg= oaInvoice invoice = new oaInvoice(); invoice.id = <invoice internal id>`
- **dashboard-project**

  app= pm (displays the dashboard view of the project with specified internal id)

  `arg= oaProject project = new oaProject(); project.id = <project internal id>`
- **grid-timesheet**

  app= ta (displays the grid of the timesheet with specified internal id)

  `arg= oaTimesheet timesheet = new oaTimesheet(); timesheet.id = <timesheet internal id>`
- **report-timesheet**

  app= ta (displays the timesheet report of specified internal id)

  `arg= oaTimesheet timesheet = new oaTimesheet(); timesheet.id = <timesheet internal id>`
- **calendar-user**

  app= ma (displays the user calendar for the specified criteria)
  - period_view - valid values: daily, weekly, or monthly

    Note: If missing, the monthly view will be used.
  - user_view — ID of the user

    Note: If missing, the ID of the current user will be used.
  - department_view - ID of the department
  - start_date - date in the format YYYY-MM-DD e.g. 2013-01-01

    Note: If missing, the current date will be used.
  - transactions array (the types of calendar item)

- transaction - valid values: booking, schedule_request, assignment, and workschedule

Note: If missing, all transaction types will be included.

Response

Array of MakeURLResult

Sample Code — C#

```
oaEnvelope envelope = new oaEnvelope();
envelope.id = "1";

MakeURLRequest mur = new MakeURLRequest();
mur.uid = _svc.SessionHeaderValue.sessionId;
mur.app = "te";
mur.page = "list-envelope-receipts";
mur.arg = envelope;

MakeURLResult[] results = _svc.makeURL(new MakeURLRequest[] { mur });
```

Sample Code — Java

```
String sessionId = loginResult.getSessionId();
MakeURLRequest make = new MakeURLRequest();

make.setUid(sessionId);
make.setApp("te");
make.setPage( "list-envelope-receipts" );
oaEnvelope envelope = new oaEnvelope();
envelope.setId("1");
make.setArg( envelope );

// make url
MakeURLResult[] mkresults = stub.makeURL(new MakeURLRequest[] { make });
```

modify

Syntax

```
UpdateResult[] result = stub.modify(attributes, objects);
```

Use

Use this call to modify data in OpenAir. The method returns an error if more than 1000 objects are passed in. You can use an externalid field as a foreign key and modify a record without querying first for an internal ID, however, if the record doesn't exist, the API will return an error message. Refer to instructions for Using an externalid field as a foreign key. See the following code examples: Example II. modify using external_id as foreign key lookup field C# and Example II. externalid as foreign key lookup field Java. You can also modify data in OpenAir based on an internal ID.

Important: Review the following guidelines:

- If you are using SAML for authentication in to your OpenAir account:
  - You cannot set a `password` if SAML authentication is enabled for the user (`saml_auth__c` set to `true`) when using the `modify` command **to update an existing user**. 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 `modify` command 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.

Arguments

Name	Type	Description
attributes	[] Attribute	Array of Attribute objects. See note below.
objects	[] oaBase	Array of oaBase objects

Note: Refer to Modifying Records to Set Custom Field Values for information on modifying the wsdl file to include custom fields and using the lookup_custom attribute. Specific code examples follow.

Response

`Array of UpdateResult objects.`

C# Modify Code Examples

Example I. modify C#

Modify a customer’s email address.

```
oaCustomer customer = new oaCustomer();
customer.id = "37";
customer.addr_email = "newest@example.com";

UpdateResult[] res =_svc.modify(new OA.Attribute[]{},new
oaBase[]{customer});
```

Example II. modify using external_id as foreign key lookup field 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”.

```
oaFieldAttribute lookupAttr = new oaFieldAttribute();
lookupAttr.name = "external";
lookupAttr.value = "filterset_ids:Filterset:1";

oaUser user = new oaUser();
user.id = "12";
user.filterset_ids = "extrn1,extrn2,extrn3";
user.attributes = new oaBase[] { lookupAttr };

UpdateResult[] results = _svc.modify(new OA.Attribute[] {}, new
oaBase[] { user });
```

Example III. modify using custom field as lookup field 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”.

```
oaCustomer customer = new oaCustomer();
customer.name = "John Carr";
customer.CustField12__c = "somevalue";

//this attribute specifies which custom field should be used for
lookup
OA.Attribute lookupAttr = new OA.Attribute();
lookupAttr.name = "lookup_custom";
lookupAttr.value = "CustField12__c";

UpdateResult[] results = _svc.modify(new OA.Attribute[] { lookupAttr
}, new oaBase[] { customer });
```

Example IV. update custom field value using an inline (__c) property C#

```
oaProject project = new oaProject();
project.id = "123"; //id of record to modify
project.ProjectStatus__c = "Yellow"; //new custom field value

//Define attribute that directs API to update a custom field.
OA.Attribute updateCustom = new OA.Attribute();
updateCustom.name = "update_custom";
updateCustom.value = "1";

UpdateResult[] res = _svc.modify(new OA.Attribute[] { updateCustom },
new oaBase[] { project });
```

Example V. update custom field value using custom equal to method C#

```
OA.Attribute lookupAttr = new OA.Attribute();
lookupAttr.name = "method";
lookupAttr.value = "custom equal to";

oaCustomField customField = new oaCustomField();
customField.type = "User"; //name of the object the field is
associated with
customField.id = "12"; //internal ID of the user record.
customField.name = "cust_field_name"; // internal name of the custom
field.
customField.value = "My new value"; // new value

UpdateResult[] updateResults = _svc.modify(new OA.Attribute[] {
lookupAttr }, new oaBase[] { customField });
```

Example VI. modify import_export 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.

```
oaImportExport exportRecord = new oaImportExport();
exportRecord.application = "MY_APP";
exportRecord.type = "Envelope";
exportRecord.id = "4";
exportRecord.exported = "2008-04-01 00:00:00";
UpdateResult[] ur = _svc.upsert(new OA.Attribute[] {}, new oaBase[] {
exportRecord });

//Define read parameters
ReadRequest rr = new ReadRequest();
rr.method = "all";
rr.type = "Envelope";

//Export only records that have not yet been exported by MY_APP
OA.Attribute notExportedAttr = new OA.Attribute();
notExportedAttr.name = "filter";
notExportedAttr.value = "not-exported";
rr.attributes = new OA.Attribute[] { notExportedAttr };

//Direct the API to filter out records exported by MY_APP
oaImportExport importExport = new oaImportExport();
importExport.application = "MY_APP";
rr.objects = new oaBase[] { importExport };

ReadResult[] results = _svc.read(new ReadRequest[] { rr });
```

Java Modify Code Examples

Example I. modify Java

```
// Modify customer's email address
oaCustomer customer1 = new oaCustomer();
customer1.setAddr_email("new@example.com");
customer1.setId("66");

// Attribute not used in this case but needs to be passed in.
Attribute dummy = new Attribute();
UpdateResult[] results = binding.modify(new Attribute[] { dummy },
new oaBase[] { customer1 });
```

Example II. externalid as foreign key lookup field Java

```
// For each xxxid field that needs to be looked up, set the ID field
```

```
(filtersetids) to the externalid field value.
// 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 cases just a single ID values is used.
oaFieldAttribute attr = new oaFieldAttribute();
att.setName("external"); // type of lookup (external will lookup the
field using external_id field)
attr.setValue("filtersetids:Filterset:1"); // colon separated values:
field name (as it exists in the object, matching record type to
process lookup for, 1 is needed to process comma separated values
(it's not required for regular fields)

// Set the user field (filtersetids in this case, to the value of the
externalid (instead of the internalid used normally)
oaUser user = new oaUser();
user.setId("10119");
user.setFiltersetids("external1,external2,external3,external4");

// notice that the field used (filtersetids) matches the first part of
the attributes value.
// Set the attributes collection for user object. Set the attribute as
part of collection.
user.setAttributes(new oaBase[] {attr});

// process modify
UpdateResult[] results = service.modify(new Attribute[] { null},
new oaBase[] {user});
```

Example III. custom equal to Java

```
// Attributes can be used to specify non-default method to update
custom fields for a given object:
// create the attribute to specify method.
Attribute custom = new Attribute();
custom.setName("method");
custom.setValue("custom equal to");

// create custom field object
oaCustomField customf = new oaCustomField();
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.

customf.setName("userc"); // internal name of the custom field.
customf.setId("1"); // internal ID of the user record
customf.setValue("My new value"); // custom value

UpdateResult[] updateResults = stub.modify(new Attribute[] { custom }, new
oaBase[] { customf });
```

Example IV. not exported Java

```
// To modify and read not-yet exported envelopes
// mark the envelope with ID = 4 as exported by the application
// MY_APP on 4/1/2008. By doing this, we can add a filter attribute to
// our read call that will filter out records exported by MY_APP.
oaImportExport exportRecord = new oaImportExport();
exportRecord.setApplication( "MY_APP" );
exportRecord.setType( "Envelope" );
exportRecord.setId( "4" );
exportRecord.setExported( "2008-04-01 00:00:00" );
UpdateResult[] ur = binding.upsert( new Attribute[]{}, new oaBase[] {

exportRecord });

// Now do a read of all envelopes, but add a filter
// so we only retrieve not-yet-exported records
```

```
Attribute attr = new Attribute();
attr.setName( "filter" );
attr.setValue( "not-exported" );

ReadRequest read = new ReadRequest();
read.setMethod( "all" );
read.setType( "Envelope" );
read.setAttributes( new Attribute[]{attr} );

// tell the server we're filtering on import_export records created by
MY_APP
oaImportExport importExport = new oaImportExport();
importExport.setApplication( "MY_APP" );
read.setObjects( new oaBase[] { importExport } );

ReadResult[] results = binding.read(new ReadRequest[] { read });
```

delete

Syntax

```
UpdateResult[] deleteResults = stub.delete(object);
```

Use

Use this call to delete data in OpenAir based on an internal ID. The method returns an error if more than 1000 objects are passed in.

Arguments

Name	Type	Description
objects	[] oaBase	Array of oaBase objects

Response

`Array of UpdateResult objects.`

Sample Code - C#

```
// delete customer with internal ID 66.
oaCustomer customer = new oaCustomer();
customer.id = "66";

UpdateResult[] deleteResults = stub.delete(new oaBase[1] { customer });
```

Sample Code - Java

```
// delete customer with internal ID 66.
```

```
oaCustomer customer = new oaCustomer();
customer.setId("66");

UpdateResult[] deleteResults = stub.delete(new oaBase[] { customer });
```

whoami

Syntax

```
oaUser user = stub.whoami();
```

Use

Use this call to get the currently logged in OpenAir user.

Arguments

There are no arguments.

Response

`oaUser`

Sample Code - C#

```
oaUser user = stub.whoami();
```

Sample Code - Java

```
oaUser user = stub.whoami();
```

servertime

Syntax

```
oaDate dateNow = stub.servertime();
```

Use

Use this call to get the current server time oaDate object.

Arguments

There are no arguments.

Response

`oaDate`

Sample Code - C#

```
oaDate dateNow = stub.servertime();
```

Sample Code - Java

```
oaDate dateNow = stub.servertime();
```

logout

Syntax

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

Use

Use this call to log out of a session.

Arguments

There are no arguments.

Response

You are logged out.

Sample Code - C#

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

Sample Code - Java

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

approve

Syntax

```
ApproveResult[] results = stub.approve(requests);
```

Use

Use this call to approve OpenAir entities such as bookings, envelopes, invoices, or timesheets which were submitted for approval. The method returns an error if more than 1000 objects are passed in.

Arguments

Name	Type	Description
requests	ApproveRequest	Array of ApproveRequest objects

Response

Array of ApproveResult

Sample Code - C#

```
// approve an envelope
oaEnvelope env = new oaEnvelope();
env.id = "122";

oaApproval appr = new oaApproval();
appr.cc = "help@ddd.com"; // cc approval email to additional contacts
appr.notes = "Approval notes";

ApproveRequest ar = new ApproveRequest();
ar.approve = env;
ar.approval = appr;

ApproveResult[] results = _svc.approve(new ApproveRequest[] { ar });
```

Sample Code - Java

```
// approve an envelope which was submitted for approval
oaEnvelope env = new oaEnvelope();
env.setId("122");
```

```
oaApproval appr = new oaApproval();
appr.setCc("help@ddd.com"); // cc approval email to additional
contacts
appr.setNotes("approval notes");
ApproveRequest ar = new ApproveRequest();
ar.setApproval(appr);
ar.setApprove( env );

ApproveResult[] results = stub.approve(new ApproveRequest[] { ar });
```

reject

Syntax

```
RejectResult[] results = stub.reject(requests);
```

Use

Use this call to reject OpenAir entities such as bookings, envelopes, invoices, or timesheets which were submitted for approval. The method returns an error if more than 1000 objects are passed in.

Arguments

Name	Type	Description
requests	RejectRequest	Array of RejectRequest objects

Response

Array of RejectResult

Sample Code - C#

```
// reject an envelope which was submitted for approval
oaEnvelope env = new oaEnvelope();
env.id = "122";

oaApproval appr = new oaApproval();
appr.cc = "help@ddd.com"; // cc approval email to additional contacts
appr.notes = "Approval notes";

RejectRequest rr = new RejectRequest();
rr.reject = env;
rr.approval = appr;

RejectResult[] results = _svc.reject(new RejectRequest[] { rr });
```

Sample Code - Java

```
// reject an envelope which was submitted for approval
```

```
oaEnvelope env = new oaEnvelope();
env.setId("122");
oaApproval appr = new oaApproval();
appr.setCc("help@ddd.com"); // cc approval email to additional
contacts
appr.setNotes("approval notes");
RejectRequest rr = new RejectRequest();
rr.setApproval(appr);
rr.setReject( env );

RejectResult[] results = stub.reject(new RejectRequest[] { rr });
```

unapprove

Syntax

```
UnapproveResult[] results = stub.unapprove(requests);
```

Use

Use this call to unapprove OpenAir entities such as bookings, envelopes, invoices, or timesheets which have been approved. Entities which have already been approved and billed, or approved and archived cannot be unapproved. The method returns an error if more than 1000 objects are passed in.

Arguments

Name	Type	Description
requests	UnapproveRequest	Array of UnapproveRequest objects

Response

Array of UnapproveRequest

Sample Code - C#

```
// unapprove an envelope which has already been approved
oaEnvelope env = new oaEnvelope();
env.id = "122";

oaApproval appr = new oaApproval();
appr.cc = "help@ddd.com"; // cc approval email to additional contacts
appr.notes = "Approval notes";

UnapproveRequest ur = new UnapproveRequest();
ur.unapprove = env;
ur.approval = appr;

UnapproveResult[] results = _svc.unapprove(new UnapproveRequest[] { ur });
```

Sample Code - Java

```java
// Unapprove an envelope which has already been approved
oaEnvelope env = new oaEnvelope();
env.setId("122");
oaApproval appr = new oaApproval();
appr.setCc("help@ddd.com"); // cc approval email to additional
contacts
appr.setNotes("approval notes");
UnapproveRequest ur = new UnapproveRequest();
ur.setApproval(appr);
ur.setUnapprove( env );

UnapproveResult[] results = stub.unapprove(new UnapproveRequest[] { ur });
```

Web Services Method Complex Types

The following are Web Services parameters and return values. Each is presented with a statement about its use and a listing of children. Links to calls where these types are used are also provided.

Attribute

Use this complex type to specify attribute for various calls. Attribute has the following children.

Field Name	Description
name	Name of the method.
value	Value of an attribute.

For more information on its use, refer to the following methods: upsert, submit, modify, approve, reject, unapprove, and makeURL.

LoginParams

Use login parameters to authenticate users into OpenAir. LoginParams has the following children.

Field Name	Description
api_namespace	Namespace name.
api_key	Specify an API key assigned to you.
company	Specify companyID.
user	Specify User ID.
password	Specify password.
client	Specify the client name.
version	Specify the version of the client.

For more information on its use, refer to the login method.

LoginResult

This complex type holds the results of login call. LoginResult has the following children.

Field Name	Description
sessionId	ID associated with this session.

Field Name	Description
URL	URL of the logged in session for this user.

For more information on its use, refer to the login method.

MakeURLRequest

Use this complex type to specify parameters for makeURL call. MakeURLRequest has the following children.

Field Name	Description
uid	Valid sessonID.
page	Page abbreviation.
app	Application abbreviation.
arg	Any argument.

For more information on its use, refer to the makeURL method.

MakeURLResult

This complex type holds the results of makeURL call. MakeURLResult has the following children.

Field Name	Description
url	Valid authenticated URL.
errors	A collection of oaError objects.
status	-l in case of any errors.

For more information on its use, refer to the makeURL method.

ReadRequest

Use this complex type to specify parameters for read method. ReadRequest has the following children.

Field Name	Description
method	Method name.
fields	Comma separated list of fields to return.
attributes	A collection of Attribute objects.

Field Name	Description
type	Types of the record.
objects	Any oaBase objects as arguments.

For more information on its use, refer to the read method

Using ReadRequest

ReadRequest can be used with a number of oaComplex Types as well as with different methods. You can also specify fields and attributes. All of these help you specify what records you want or allow you to limit the ReadResults that are returned. Refer to the following sections for specific information.

Types

The following types can be specified. To review the field names and definitions of each complex type, refer to OpenAir Complex Types. A list of supported methods is provided for each complex type.

Actualcost	Address	Agreement_to_project
Agreement	Approval	Attachment
AttributeDescription		
Attributeset	Attribute	BookingType
Booking	BudgetAllocation	Budget
Category_1	Category_2	Category_3
Category_4	Category_5	Category
Ccrate	Company	Contact
Costcategory	Costcenter	Costtype
Currencyrate	Currency	CustField
Customerpo_to_project	Customerpo	Customer
CustomField	Dealcontact	Dealschedule
Deal	Department	Entitytag
Envelope	Error	Estimateadjustment
Estimateexpense	Estimatelabor	Estimatemarkup
Estimatephase	Estimate	Event
ExpensePolicy	ExpensePolicyItem	
Filterset	ForexInput	FormPermissionField
Fulfillment	HierarchyNode	Hierarchy
History	ImportExport	Invoice

IssueCategory	IssueSeverity	IssueSource
IssueStage	IssueStatus	Issue
Item	Jobcode	Leave_accrual_rule_to_user
Leave_accrual_rule	Leave_accrual_transaction	LoadedCost
Paymentterms	Paymenttype	Payment
Payrolltype	PendingBooking	Preference
Product	Projectassign	ProjectAssignmentProfile
Projectbillingrule	Projectbillingtransaction	ProjectBudgetGroup
ProjectBudgetRule	ProjectBudgetTransaction	
Projectgroup	Projectlocation	Projectstage
Projecttaskassign	Projecttask_type	Projecttask
Project	Proposalblock	Proposal
Purchase_item	Purchaseorder	Purchaserequest
Purchaser	RateCardItem	Ratecard
Reimbursement	Repeat	Report
Request_item	Resourceprofile_type	Resourceprofile
RevenueContainer	Revenue_recognition_rule_amount	Revenue_recognition_rule
Revenue_recognition_transaction	RevenueStage	Schedulebyday
Scheduleexception	Schedulerequest_item	Schedulerequest
SlipProjection	Slipstage	Slip
TagGroupAttribute	TagGroup	TargetUtilization
TaskTimecard	Task	TaxLocation
TaxRate	Ticket	Timecard
Timesheet	Timetype	Todo
Uprate	UserWorkschedule	User
Vendor	Viewfilterrule	Viewfilter
Workspacelink	Workspaceuser	

Methods

Use one of the following methods. Each is explained as follows:

all

- Returns all available records.
- Use this cautiously as too many records may be requested for the server or client to handle.

equal to

- Returns records that have fields that are equal to the field value(s) passed in.
- Calculated fields are not supported. You cannot limit the response to records with a calculated field equal to or not equal to a specific value using this method.
- Use ReadRequest objects to specify object and field values to include in this search.
- Example I. read equal to C# and Example I. read equal to Java.
- Multiple equal to and not equal to method arguments can be mixed in a single ReadRequest, which allows creating queries with AND/OR filtering logic. See Example VIII. read equal to with explicit OR condition C#. To use this feature:
  - Modify the method parameter to include multiple “equal to” and “not equal to” methods as desired, separated by commas. For example: “equal to, not equal to”.
  - Next, supply an equal number of argument objects to filter on.
  - You may precede each method by an “and” or an “or” operator. For example: “equal to, or equal to”. If no operator is supplied, a logical AND relationship is assumed.

> **Note:** This feature supports only one level of logical operators. It does not support nesting of operators like "equal to and (equal to or equal to)". In addition, if one of the filtering objects has multiple properties specified, the top level logical operator will be applied to all of them.

not equal to

- Returns records that have fields that are not equal to the field value(s) passed in.
- Calculated fields are not supported. You cannot limit the response to records with a calculated field equal to or not equal to a specific value using this method.
- Use ReadRequest objects to specify object and field values to include in this search.
- Example II. read not equal to C# and Example II. read not equal to Java.

custom equal to

- Allows you to display custom field values for a particular record.
- read.objects collection must contain a record with the internal ID set to specify the ID of the record the custom fields should be returned for.
- oaCustomField objects are returned as part of the ReadResult.objects.collection.
- Example III. read custom field definitions C#.
- **custom equal to objects** - For a list of associated objects, see the oaCustField Association Table or Type of Object.

> **Note:** Custom fields can also be read inline with the parent object. See Modifying Records to Set Custom Field Values.

Fields

Use a comma separated list of fields to limit the amount of data returned.

Attributes

Use one of the following attributes.

Attribute Name	Value	Result
limit	‘1000' or ‘0, 1000'	Restricts the number of records returned. Single number value: "1", "500", "1000" - simply restricts the number of records returned. Double number value: "0, 1000" - the first integer specifies the offset of the first record to return and the second integer limits the number of records to return. To request data in consecutive batches, only the first part of the limit attribute should be incremented - "0,1000", "1000,1000", "2000,1000", etc. Sequence requests should be submitted until the result comes back empty or has less items than 1000.
deleted	1	Returns deleted records. It can be used together with newer-than filter.
include_flags	1	Returns account or user switches, by default those are not populated.
include_nondeleted	1	Returns all records, deleted and nondeleted. **Note:** This attribute only works in conjunction with the "deleted" attribute.
with_project_only	1	Used only with type: Customer. Will only return customers which have associated project records.
base_currency	3	Letter currency code. Works with type: Currencyrate. Converts values on the fly to currency specified.
generic	1	Returns generic resources (users) only, where by default, the API returns regular users only.
calculate_hours	‘0’ or ‘1’	Used only with type: oaTimesheet. Must be set to ‘1’ to return the Minimum number of hours required on the timesheet and the Maximum number of hours allowed on the timesheet as determined by Administration > Application Settings > Timesheets > Timesheet rules. See type: oaTimesheet.
filter See Notes below.	* open-envelopes	Returns only records associated with an open envelope.
	* approved-envelopes	Returns only records associated with an approved envelope.
	* rejected-envelopes	Returns only records associated with a rejected envelope.
	* submitted-envelopes	Returns only records associated with a submitted envelope.
	* nonreimbursed-envelopes	Returns envelopes that have a non-zero balance attribute.

Attribute Name	Value	Result
	* reimbursable-envelope	Returns only records associated with a reimbursable envelope.
	* open-slips	Returns only records associated with an open slip.
	* approved-slips	Returns only records associated with an approved slip.
	* open-timesheets	Returns only records associated with an open timesheet.
	* approved-timesheets	Returns only records associated with an approved timesheet.
	* rejected-timesheets	Returns only records associated with a rejected timesheet.
	* submitted-timesheets	Returns only records associated with a submitted timesheet.
	* not-exported	Returns only records that have not been marked as exported. See Example V. read not exported C# and Example III. read not exported Java.
	* approved-revenue-recognition-transactions	Returns only revenue recognition transactions belonging to approved revenue_container records.
	date filters: * newer-than * older-than * date—equal-to * date-not-equal-to	Returns only records that have a value in the 'updated' field that is newer-than, older-than, date-equal-to, or date-not-equal-to the date specified in the oaDate object in the objects collection. To compare date fields other than 'updated', add the following additional attribute information: name="field" value="[some date field]" See Example VI. read not-exported Envelopes with date filter C# and Example IV. read date filter Java.

Note: The following notes apply to using one or more filters:

- A record is associated with an open envelope, open slip, or open timesheet if it has an ID field that points to either an envelope (envelope_id), slip (slip_id), or timesheet (timesheet_id). Do not use the filter attribute with data types that do not have associated IDs. (i.e., Do not use type Project and the filter open-envelopes.)
- Multiple date filters can be used. They should be separated by a comma: newer-than, older-than, date-equal-to, date-not-equal-to. The argument objects should be included in the same order as the filters those arguments apply to as part of the objects collection.
- Multiple filters can be used. They should be CSV concatenated in one single filter attribute. For example, you can use the following attributes to retrieve all timesheet entries in a certain date range for approved timehseets only: filter="newer-than, older-than, approved-timesheets".

ReadResult

This complex type returns any results of the read method. ReadResult has the following children.

Field Name	Description
objects	A collection of oaBase objects.
errors	A collection of oaError objects.

For more information on its use, refer to the read method.

SessionHeader

Use this complex type to hold session information. `SessionHeader` has the following children.

Field Name	Description
`sessionId`	Valid sessionID of the logged in user session. Used with session based password authentication.
`accessToken`	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 Code Examples.
- 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 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:

XML Example

```
<SessionHeader xsi:type="perl:SessionHeader" mlns:perl="http://namespaces.soaplite.com/perl">
      <accessToken xsi:type="xsd:string">eNNJ1GXD25-6IUylF6RZT33HqhoqSAAK53F0kxT62fBoKreDoc8Y_-Gnk2lUIqNbhwguHnxDtxUsJMY6NrDoiBnd</accessToken>
</SessionHeader>
```

C# Example

```
// Create service stub
OAirServiceHandlerService _svc = new OAirServiceHandlerService();

// POST Request to get access_token

// Create a new session header object and add the access token
_svc.SessionHeaderValue = new SessionHeader();
_svc.SessionHeaderValue.accessToken = response.access_token;
```

Java Example

```java
// Set the service URL
OAirServiceHandlerServiceLocator locator = new OAirServiceHandlerServiceLocator();
locator.setOAirServiceAddress("https://company-id.app.openair.com/soap");

// POST Request to get access_token

// Create a new session header object and add the access token
SOAPHeaderElement header = new SOAPHeaderElement("https://company-id.app.openair.com/OAirService", "SessionHeader");
SOAPElement node = header.addChildElement("accessToken");
node.addTextNode(access_token);
binding.setHeader(header);
```

For more information about OAuth 2.0, see OAuth 2.0 for Integration Applications Developers.

SubmitRequest

Use this complex type to specify parameters for the submit method. SubmitRequest has the following children.

Field Name	Description
attributes	A collection of Attribute objects
submit	oaEnvelope, oaInvoice, or oaTimesheet object.
approval	oaApproval object.

For more information on its use, refer to the submit method.

SubmitResult

This complex type returns any results of the submit method. SubmitResult has the following children.

Field Name	Description
id	Internal ID of the object submitted.
approval_warnings	String representing any warnings.
approval_errors	String representing any errors.
log	String representing the log of actions.
errors	A collection of oaError objects.
status	-1 in case of errors.

For more information on its use, refer to the submit method.

ApproveRequest

Use this complex type to specify parameters for the approve method. ApproveRequest has the following children.

Field Name	Description
attributes	A collection of Attribute objects
approve	oaEnvelope, oaInvoice, or oaTimesheet object.
approval	oaApproval object.

For more information on its use, refer to the approve method.

ApproveResult

This complex type returns any results of the approve method. ApproveResult has the following children.

Field Name	Description
id	Internal ID of the object submitted.
approval_warnings	String representing any warnings.
approval_errors	String representing any errors.
log	String representing the log of actions.
errors	A collection of oaError objects.
status	-1 in case of errors.

For more information on its use, refer to the approve method.

RejectRequest

Use this complex type to specify parameters for the reject method. RejectRequest has the following children.

Field Name	Description
attributes	A collection of Attribute objects
reject	oaEnvelope, oaInvoice, or oaTimesheet object.
approval	oaApproval object.

For more information on its use, refer to the reject method.

RejectResult

This complex type returns any results of the reject method. RejectResult has the following children.

Field Name	Description
id	Internal ID of the object submitted.

Field Name	Description
approval_warnings	String representing any warnings.
approval_errors	String representing any errors.
log	String representing the log of actions.
errors	A collection of oaError objects.
status	-1 in case of errors.

For more information on its use, refer to the reject method.

UnapproveRequest

Use this complex type to specify parameters for the unapprove method. UnapproveRequest has the following children.

Field Name	Description
attributes	A collection of Attribute objects
unapprove	oaEnvelope, oaInvoice, or oaTimesheet object.
approval	oaApproval object.

For more information on its use, refer to the unapprove method.

UnapproveResult

This complex type returns any results of the unapprove method. UnapproveResult has the following children.

Field Name	Description
id	Internal ID of the object submitted.
approval_warnings	String representing any warnings.
approval_errors	String representing any errors.
log	String representing the log of actions.
errors	A collection of oaError objects.
status	-1 in case of errors.

For more information on its use, refer to the unapprove method.

UpdateResult

This complex type holds the results of a given method call. UpdateResult has the following children.

Field Name	Description
id	Internal ID of the record created or updated.
errors	A collection of oaError objects.
status	U - record was updated. A - record was added. D - record was deleted. -1 - one or more errors occurred.

For more information on its use, refer to the following methods: add, modify, upsert, createUser, createAccount, and delete.

VersionResult

This complex type holds the results of the version method call. VersionResult has the following children.

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

For more information on its use, refer to the version method.

Custom Fields

Introduction to Custom fields

Custom Fields are helpful additions to your OpenAir account. You can use the oaCustomField complex type to modify or read custom field values associated with your records. You can use oaCustField to get a list of custom field definitions and their metadata in an OpenAir account such as name, association, the type of custom field (use with the read method). You can also use oaCustField to set valuelist on custom field definitions of drop-down and multi-select types (use with modify method).

Note: It is not possible to rename, change, or delete a custom field which is being used by an active script. This prevents unintended script problems.

You may also request all available custom field types that exist for a given object or you may read custom field values for a specific record. You can modify records to set custom field values and add records with inline custom field values.

Refer to the following sections for more information about working with custom fields. Links to code examples are provided for read and modify methods. Navigation for finding custom fields and XML tag limitations follow.

Finding Custom Fields

To find custom fields in the OpenAir Web application, go to Administration > Global Settings > Custom Fields.

XML Tag Limitations

General XML tag limitations apply to the names of custom fields when the wsdl is modified. They include the following: names cannot start with a number or with the letters “xml” in any form such as XML or Xml. For example, 1MyCustomField or xmlMyCustomField would not work with SOAP.

Requesting Custom Fields for an Object

You can request all custom fields that exist for a given object. Use the read method and specify the CustField type and **filter** for a particular association. Refer to the oaCustField complex type Association Table or Type of Object for a list of possible associations.

Reading Custom Field Values

You can read custom field values for a given record using several options.

1. Use the custom equal to method described in Using ReadRequest to only request custom field values for a particular record. You need to know the internal ID of the record in question. See Example IV. read custom field values C#.

2. Alternatively, modify the wsdl file to include custom fields for the object you're requesting. Use the field name and add "__c" to the end of the name. (Note that there are two underscores before the c.) See XML Tag Limitations.

Modifying Records to Set Custom Field Values

You can modify records to set custom field values. Refer to the following examples.

1. Use the custom equal to attribute of the modify call to set custom fields. See Example V. update custom field value using custom equal to method C# and Example III. custom equal to Java.
2. Use a custom field in place of an internal ID to lookup and modify a record. Modify the wsdl file to include custom fields for the object you want to look up. Use the field name, add "__c" to the end of its name, and use the lookup_custom attribute. (Note that there are two underscores before the c.) Refer to Example III. modify using custom field as lookup field C#. See XML Tag Limitations.

   Note: This method only works with modify and is not supported by the upsert method.

3. Modify custom fields inline in a single modify request with other fields. To utilize this feature, use the "update_custom" attribute in your modify request and set it to 1. Then, as in the previous step, add any needed custom fields to your wsdl file and perform a normal modify request.

Adding Records with Inline Custom Field Values

You can add records using the add or createUser with inline custom field values.

1. Modify the wsdl file to include custom fields for the object you're working with. Use the field name and add "__c" to the end of the name. (Note that there are two underscores before the c.)
2. Specify custom fields inline in an object to be used in the array passed into the add method. See add and createUser method. See XML Tag Limitations.

Setting Allocation Grid Custom Field Values

Values for allocation grid custom fields are set as a string with each value-number pairs separated by a new line character.

When setting values for allocation grid custom fields, use the format illustrated in the following example:

```
myProject.my_allocation_grid__c = "Marc Collins,0\nBill Carter,0\nMarie Porter,50\nEd Ellis,50";
```

OpenAir Complex Types

The OpenAir SOAP API contains two complex types:

- OpenAir Complex Types - that describe business objects contained in the WSDL.
- Web Services Method Complex Types - that hold parameters and return values for Web Services calls.

This chapter provides the parent and children relationships of each OpenAir Complex Type. It includes a statement of its use and a listing of children. Links to supported calls are also provided.

See Web Services Method Complex Types to review Web services calls.

Important: The following fields are read-only and cannot be modified:

- The `updated` and `created` fields are maintained automatically by OpenAir.
- Calculated fields are calculated automatically by OpenAir. Note also that the ReadRequest equal to and not equal to methods do not support calculated fields. You cannot limit the response to records with a calculated field equal to or not equal to a specific value using either of these methods in ReadRequest.

oaActualcost

Use this complex type to add or update actual cost information. oaActualcost has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
cost	The cost.
cost_typeid	The ID of the cost_type.
created	Time the record was created.
currency	Currency of the cost field.
date	Date for the actual cost.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
is_accrual	A 1/0 field indicating whether this actual cost is an accrual.
name	The name of the actual cost. This field is never populated. It is used only to satisfy subtotalling by actual cost.
notes	Notes.
period	The time period of the actual cost: Daily, Weekly, Monthly, Quarterly, Annually.
updated	Time the record was last modified.
userid	The ID of the user.

This complex type supports the following methods: read, add, modify, and upsert.

oaAccountingPeriod

The oaAccountingPeriod complex type holds a date range defining an accounting period.

Field Name	Description
active	A "1/0" field indicating whether this period is open or closed.
created	Time the record was created.
current_period	A "1/0" field indicating whether this is the current period.
default_period	A "1/0" field indicating whether this is the default period.
end_date	The ending date of the period.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the accounting period.
notes	Notes field.
period_date	The custom date to use for this period.
period_date_how	What date should be used when marking transactions to this period: ■ 'S'tart date ■ 'E'nd date ■ 'P'eriod date
start_date	The starting date of the period.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and delete.

oaAddress

Use this complex type to add or update address information. oaAddress has the following children.

Field Name	Description
addr1	Address line 1
addr2	Address line 2
addr3	Address line 3
addr4	Address line 4
attributes	A collection of additional attributes for this complex type.
city	City
country	Country
email	Email address
fax	Fax number

Field Name	Description
first	First name
id	Unique ID. Automatically assigned by OpenAir.
last	Last name
middle	Middle name
mobile	Mobile phone number
phone	Phone number
salutation	Contact's salutation
state	State
zip	Zip code

This complex type supports the following methods: add, createAccount, createUser, and modify

oaAgreement

Use this complex type to track money through projects and billings. oaAgreement has the following children.

Field Name	Description
acct_date	The accounting period date of the agreement.
active	A 1/0 field indicating whether this is an active agreement.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
created	Time the record was created.
currency	Currency for the money fields in the record.
customerid	Customer ID.
date	The date of the agreement.
externalid	External ID.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the agreement.
notes	Notes.
number	The agreement number.
picklist_label	Label as shown on form picklist.
total	The agreement total. Dated by the date field.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaAgreement_to_project

Use this complex type to create a many-to-many link between projects and agreements. oaAgreement_to_project has the following children.

Field Name	Description
id	Unique ID. Automatically assigned by OpenAir.
attributes	A collection of additional attributes for this complex type.
agreementid	The ID of the associated agreement.
customerid	The ID of the associated customer. Does not need to be input as it can be derived inline from project_id.
projectid	The ID of the associated project.
active	A 1/0 field indicating whether this is an active agreement.
created	Time the record was created.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaApproval

Use this complex type to store approval information for invoices, timesheets, expense reports, and proposals. oaApproval has the following children:

Field Name	Description
attributes	A collection of additional attributes for this complex type.
cc	Email cc field
notes	Notes

This complex type supports the following methods: submit, approve, reject, and unapprove.

oaApprovalLine

Use the oaApprovalLine datatype to read the approval table. This datatype is read-only.

Field Name	Description
action	The approval action. ▪ S - Initial submittal for approval ▪ P - Pending approval request ▪ A - Acceptance of approval request ▪ R - Rejection of approval request ▪ U - Unapproval action
approvalid	ID of the associated approval. Represents a meta-approval, or an 'approval confirmation'.

Field Name	Description
approvalprocess_ruleid	ID of the approval process rule if this is associated with an approval process.
approvalprocessid	ID of the approval process if this is associated with an approval process.
audit	Audit trail of changes
authorizationid	ID of the associated authorization
booking_requestid	ID of the associated booking request
bookingid	ID of the associated booking
created	Time the record was created
customerid	ID of the associated customer
date	Date and time of the action
deal_booking_requestid	ID of the associated deal booking request
delay_action	Delayed action
delay_to	Delay action until this time
envelopeid	ID of the associated envelope (expense report)
id	Unique ID. Automatically assigned by OpenAir.
invoiceid	ID of the associated invoice
notes	Notes, reasons, etc.
pending_done	If the action is 'P'ending, this flag is set to 1 once an 'A' or 'R' action record is created.
project_budget_groupid	ID of the associated project budget group
project_total	If this is a project-based approval this holds the total amount (money or hours) that was approved.
projectid	ID of the associated project if this is a project approval
proposalid	ID of the associated proposal
purchaseorderid	ID of the associated purchase order
purchaserequestid	ID of the associated purchase request
revenue_containerid	ID of the associated revenue container
schedule_requestid	ID of the associated schedule request
seq_number	If this is associated with an approval process, this is the sequence number associated with it.
status	The status of the child meta-approval. Only assigned a value if the record has a meta-approval. ▪ S - Submitted ▪ A - Approved ▪ R - Rejected
submitter	ID of the user submitting the approval. Only valid for a submittal record (action = 'S').

Field Name	Description
timesheetid	ID of the associated timesheet.
updated	Time the record was last updated or modified
userid	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.

This complex type supports the read method.

oaApprovalProcess

Use this complex type to read approval process information. oaApproval has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
externalid	The unique external record ID if the record was imported from an external system
id	Unique ID. Automatically assigned by OpenAir.
name	The name used for display in popups and lists.
updated	Time the record was last modified.

This complex type supports the read method.

oaAttachment

Use this complex type to specify information about task and proposal attachments and documents or folders. oaAttachment has the following children.

Field Name	Description
attachmentid	If non-zero, the attachment record associated with this attachment.
attributes	A collection of additional attributes for this complex type.
base64_data	Base 64 encoded binary data of the actual attachment file.
created	Time the record was created.
file_name	The true attachment name, as provided by the user on upload.
hash_name	The name of the file as stored on disk in our system. This is the relative path to the file from the document root directory.
id	Unique ID. Automatically assigned by OpenAir.
is_a_folder	A “1/0” field indicating if nay other attachments have us as a parent.

Field Name	Description
locked_by	The ID of the user who uploaded the file, 0 if unlocked.
name	The display name of the attachment.
notes	Notes associated with the attachment.
owner_type	The owner of this attachment, e.g. 'user', 'envelope', 'ticket', 'timesheet', 'agreement', or 'customerpo'.
ownerid	The ID of the record linking to this attachment.
parentid	The attachment ID of our immediate ancestor. If zero/null, this is a top-level document/folder.
size	The size in bytes of the associate file. This attribute is read-only.
updated	Time the record was last modified.
workspaceid	The ID of the associated workspace.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: 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` must be included in the request and 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.

oaAttribute

Use this complex type as a table that describes an attribute. oaAttribute has the following children.

Field Name	Description
attribute_setid	ID of the associated attribute set.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	Name of the attribute.
notes	Notes.
updated	Time the record was last modified.

This complex type supports the read method.

oaAttributeDescription

Use this complex type for descriptions of attributes in resource profiles, for example, detailed descriptions of what characteristics define various language levels (beginner, intermediate, advanced) or technical competencies. oaAttributeDescription has the following children:

Field Name	Description
attributeid	ID of the attribute.
created	The time the record was created.
deleted	A "1/0" field indicating if the record was deleted.
description	Information about the attribute in context of specific resourceprofile_type.
id	Unique ID. Automatically assigned by OpenAir.
resourceprofile_typeid	ID of the resourceprofile_type.
updated	The time the record was last modified.

This complex type supports the read, add, modify, and delete methods.

oaAttributeset

Use this complex type to describe an attribute set. oaAttributeset has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	Name of the attribute.
notes	Attributeset notes
updated	Time the record was updated.

This complex type supports the read method.

oaBase

oaBase is a base object for most OpenAir Complex Types. Collections are passed as oaBase objects and most OpenAir Complex Types are derived from oaBase.

The following complex types use oaBase:

oaAddress	oaEstimatephase	oaPurchaser
oaAgreement	oaEvent	oaPurchaserequest
oaApproval	oaHierarchy	oaRatecard
oaBooking	oaHistory	oaResourceprofile
oaBookingType	oaImportExport	oaResourceprofile_type
oaBudget	oaInvoice	oaResourceRequest
oaBudgetAllocation	oaItem	oaResourceRequestQueue

oaCategory	oaJobcode	oaResourcesearch
oaCcrate	oaLeave__rule	oaRevenue_recognition_rule
oaCompany	oaLeave_accrual_rule_to_user	oaRevenue_recognition_rule_amount
oaContact	oaLeave_accrual_transaction	oaRevenue_recognition_transaction
oaCostcenter	oaLoadedCost	oaSchedulerequest
oaCurrency	oaModule	oaSchedulerequest_item
oaCurrencyrate	oaPayment	oaSlip
oaCustField	oaPaymentterms	oaSlipstage
oaCustomer	oaPaymenttype	oaSwitch
oaCustomerpo	oaPayrolltype	oaTask
oaCustomerpo_to_project	oaPreference	oaTaskTimecard
oaCustomField	oaProduct	oaTaxLocation
oaDate	oaProject	oaTaxRate
oaDeal	oaProjectbillingrule	oaTerm
oaDealcontact	oaProjectbillingtransaction	oaTicket
oaDealschedule	oaProjectlocation	oaTimecard
oaDepartment	oaProjectstage	oaTimesheet
oaEntitytag	oaProjecttask	oaTimetype
oaEnvelope	oaProjecttask_type	oaTodo
oaError	oaProjecttaskassign	oaUprate
oaEstimate	oaProposal	oaUser
oaEstimateadjustment	oaProposalblock	oaUserWorkschedule
oaEstimateexpense	oaPurchase_item	oaVendor
oaEstimatelabor	oaPurchaseorder	oaWorkspace
oaEstimatemarkup	oaReimbursement	oaWorkspacelink
oaHierarchyNode	oaRequest_item	oaWorkspaceuser

oaBillingSplit

Use this complex type for the many to one or one to many lookup for entities that do not have a one to one relationship with the billed slip. The one to one relationship is still modeled with the embedded slip_id field.

Field Name	Description
attributes	A collection of additional attributes for this complex type.

Field Name	Description
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
project_billing_transaction	The ID of the associated project billing transaction.
slipid	The ID of the slip that was created.
taskid	The ID of the associated task.
updated	Time the record was updated.

This complex type supports the read method.

oaBooking

Use this complex type to book a user to a project. oaBooking has the following children.

Field Name	Description
approval_status	A one-character string indicating the approval status of the booking request. Possible values: ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected
as_percentage	A 1/0 field indicating which of the fields (hours or percentage) are actual, and which is derived. ■ 1 = percentage is actual and hours is derived. ■ 0 = hours in actual and percentage is derived.
attributes	A collection of additional attributes for this complex type.
booking_typeid	The ID of the associated booking_type.
created	Time the record was created.
customerid	The ID of the associated customer.
date_approved	The date the booking request was approved.
date_submitted	The date the booking_request was submitted.
enddate	The end date of the booking.
endtime	End time.
externalid	If the record was imported from an external system you store the unique external record ID here.
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	Unique ID. Automatically assigned by OpenAir.

Field Name	Description
job_codeid	The ID of the associated job code.
locationid	The location ID for this booking.
notes	Booking notes.
notify_owner	A 1/0 field indicating whether to send email to the requestor when the booking is modified.
ownerid	The ID of the associated user creating the booking.
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_profile_id	The ID of the associated project assignment profile.
project_task_id	The ID of the task within the associated project.
projectid	The ID of the associated project.
repeatid	The ID of the associated repeating event.
resource_request_queue_id	The ID of the associated resource request queue.
source_booking_id	ID of the booking used to create this booking.
startdate	The start date of the booking.
starttime	Start time.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaBooking_request

Use this complex type to access a day by day representation of the booking table.

Field Name	Description
approval_status	A one-character string indicating the approval status of the booking request. Possible values: ■ O - Open ■ P - Pending approval ■ A - Approved ■ R - Rejected
as_percentage	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	If non-zero, the attachment record associated with this booking_request.
attributes	A collection of additional attributes for this complex type.
booking_type_id	The ID of the associated booking_type.

Field Name	Description
created	Time the record was created.
customer_id	The ID of the associated customer.
date_approved	The date the booking_request was approved.
date_submitted	The date the booking_request was submitted.
description	The description or purpose for the booking_request.
enddate	The end date of the booking_request.
external_id	If the record was imported from an external system you store the unique external record ID here.
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	Unique ID. Automatically assigned by OpenAir.
job_code_id	The ID of the associated job code.
name	The name of the booking_request (Prefix + number).
notes	Booking notes
notify_owner	A 1/0 field indicating whether to send email to the booking request owner changes occur to the resulting bookings.
number	The booking_request number that increments by 1.
owner_id	The ID of the associated user creating the booking request.
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.
prefix	A static alphanumeric booking_request number prefix.
project_id	The ID of the associated project.
project_task_id	The ID of the task within the associated project.
repeat_id	The ID of the associated repeating event
startdate	The start date of the booking_request.
updated	Time the record was last updated or modified.
user_id	The ID of the associated user.

This complex type supports the read method.

oaBookingByDay

Use this complex type to access a day by day representation of the booking table.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
booking_id	The ID of the associated booking.

Field Name	Description
booking_type_id	The ID of the associated booking_type.
created	Time the record was created
customer_id	The ID of the associated customer.
date	The date of the booking.
hours	The number of booked hours on this date for this customer/project/user/booking_type. High precision to reduce effect of rounding.
id	Unique ID. Automatically assigned by OpenAir.
job_code_id	The ID of the associated job code.
project_id	The ID of the associated project.
project_task_id	The ID of the task within the associated project.
updated	Time the record was last modified.
userid	The ID of the associated user.

This complex type supports the read method.

oaBookingType

Use this complex type to describe a booking type such as billable, non-billable, or business development used in Resources module bookings. oaBookingType has the following children.

Field Name	Description
active	A 1/0 field specifying if the type is active.
approval_status	A one-character string indicating the approval status of the booking request. Possible values: ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
default_for_approval_status	A “1/0” field indicating whether this is the default booking type used when the approval status changes.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the booking type.
notes	Booking notes.
picklist_label	Label as shown on form picklist.
priority	The priority of the booking type (1 - 9).

Field Name	Description
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaBudget

Use the oaBudget complex type to create a budget entry. oaBudget has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
budgetcategory_id	The ID of the budget category.
categoryid	The ID of the associated category.
created	Time the record was created.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer.
date	The date of the budget entry.
id	Unique ID. Automatically assigned by OpenAir.
name	The name.
notes	Budget notes.
projectid	The ID of the associated project.
total	The total value of budget entry. Dated by the date field.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaBudgetAllocation

Use this complex type to allocate users and activity to a budget. oaBudgetAllocation has the following children.

Field Name	Description
allocation	The percentage of the budget entry that this user was allocated to.
attributes	A collection of additional attributes for this complex type.
budgetactivity_id	The ID of the budget activity.
budgetcategory_id	The ID of the budget category.
budgetid	The ID of the associated budget.

Field Name	Description
created	Time the record was created.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer.
date	The date of the budget entry.
id	Unique ID. Automatically assigned by OpenAir.
projectid	The ID of the associated project.
total	The total value of budget entry. Dated by the date field.
updated	Time the record was last modified.
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, and upsert.

oaCategory

Use this complex type for a service, category, activity or time type in the Proposals, Timesheets, and Invoices modules. oaCategory has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is designated as an active customer.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
cost_centerid	The ID of the associated cost center.
created	Time the record was created.
currency	Currency for the money fields in the record.
externalid	If the record was imported from an external system you store the unique external record ID here.
fixed_fee	The fixed fee value of this service.
id	Unique ID. Automatically assigned by OpenAir.
name	The category name.
notes	Category notes.
other_rate	The rate for another time billing metric.
other_rate_type	The time the other_rate field applies to. Valid entries are Day, Week, Month, Quarter, Year and Session.
picklist_label	Label as shown on form picklist.
rate	The hourly billing rate.
taxable	A 1/0 field indicating whether this item is taxable, vat-taxable, etc.

Field Name	Description
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaCategory_1

Use this complex type for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as oaCategory_2, oaCategory_3, oaCategory_4, and oaCategory_5. oaCategory_1 has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is designated as an active customer.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
created	Time the record was created.
externalid	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The category name.
notes	Category notes_2
picklist_label	Label as shown on form picklist.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete. Use custom equal to when requesting custom fields.

oaCategory_2

Use this complex type for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as oaCategory_1, oaCategory_3, oaCategory_4, and oaCategory_5. oaCategory_2 has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is designated as an active customer.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
created	Time the record was created.
externalid	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.

Field Name	Description
name	The category name.
notes	Category notes_2
picklist_label	Label as shown on form picklist.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete. Use custom equal to when requesting custom fields.

oaCategory_3

Use this complex type for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as oaCategory_1, oaCategory_2, oaCategory_4, and oaCategory_5. oaCategory_3 has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is designated as an active customer.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
created	Time the record was created.
externalid	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The category name.
notes	Category notes_2
picklist_label	Label as shown on form picklist.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete. Use custom equal to when requesting custom fields.

oaCategory_4

Use this complex type for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as oaCategory_1, oaCategory_2, oaCategory_3, and oaCategory_5. oaCategory_4 has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is designated as an active customer.
attributes	A collection of additional attributes for this complex type.

Field Name	Description
code	Optional accounting system code for integration with external accounting systems.
created	Time the record was created.
externalid	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The category name.
notes	Category notes_2
picklist_label	Label as shown on form picklist.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete. Use custom equal to when requesting custom fields.

oaCategory_5

Use this complex type for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as oaCategory_1, oaCategory_2, oaCategory_3, and oaCategory_4. oaCategory_5 has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is designated as an active customer.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
created	Time the record was created.
externalid	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The category name.
notes	Category notes_2
picklist_label	Label as shown on form picklist.
updated	Time the record was last updated or modified.
id	Unique ID. Automatically assigned by OpenAir.

This complex type supports the following methods: read, add, modify, upsert, and delete. Use custom equal to when requesting custom fields.

oaCcrate

Use this complex type to document the category customer rate table. oaCcrate has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
categoryid	The ID of the category this rate is associated with.
created	Time the record was created.
currency	The currency these rates are quoted in.
customerid	The ID of the customer this rate is associated with.
id	Unique ID. Automatically assigned by OpenAir.
notes	Notes about the table.
rate	The hourly billing rate.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaCompany

Use this complex type to specify basic company information and the company switches. oaCompany has the following children.

Field Name	Description
addr_addr1	First line of the address. *See notes below this table*
addr_addr2	Second line of the address. *See notes below this table*
addr_addr3	Third line of the address. *See notes below this table*
addr_addr4	Fourth line of the address. *See notes below this table*
addr_city	City name. *See notes below this table*
addr_country	Country name. *See notes below this table*
addr_email	Email address. *See notes below this table*
addr_fax	Fax number. *See notes below this table*
addr_first	First name.

Field Name	Description
	See notes below this table
addr_id	The ID of the associated address. *See notes below this table*
addr_last	Last name. *See notes below this table*
addr_middle	Middle name. *See notes below this table*
addr_mobile	Mobile phone number. *See notes below this table*
addr_phone	Phone number. *See notes below this table*
addr_salutation	Salutation. *See notes below this table*
addr_state	State name. *See notes below this table*
addr_zip	Zip code of the address. *See notes below this table*
attributes	A collection of additional attributes for this complex type.
base_currency	Base currency.
businesstype	General business category.
company	The company name, as it should be printed on invoices, etc.
created	The time the record was created.
currencies	The currencies for the money fields in the record.
flags	Company-specific flags.
hide_rate	Hide hourly rate from normal user types in the company.
id	Unique ID. Automatically assigned by OpenAir.
is_multicurrency	Multiple currencies.
nickname	The company nickname.
rate_from	Billing rate is pulled from: category, user, customer/project, or user/project.
updated	Time the record was last updated or modified.
VAT_registration_number	VAT registration number.
workscheduleid	The ID of the associated primary account workschedule. (read-only field)

This complex type supports the following methods: read, add, modify, and upsert. Also refer to oaSwitch for information on company-specific flags.

> **Note:** Prior to the October 2022 OpenAir release, you could not specify the address fields to be returned when reading company records. 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_`).
>
> You can now list the specific address information in the `fields` of the `ReadRequest` complex type to return only the address information required.

oaContact

Use this complex type to specify contact information. A contact is associated with a customer or prospect company. oaContact has the following children.

Field Name	Description
active	A 1/0 field indicating an active contact.
addr_addr1	First line of the address. *See notes below this table*
addr_addr2	The second line of the address. *See notes below this table*
addr_addr3	Third line of the address. *See notes below this table*
addr_addr4	Fourth line of the address. *See notes below this table*
addr_city	City. *See notes below this table*
addr_country	Country. *See notes below this table*
addr_email	Email address. *See notes below this table*
addr_fax	Fax number. *See notes below this table*
addr_first	First line of the address. *See notes below this table*
addr_id	The ID of the associated address. *See notes below this table*
addr_last	The contact's last name. *See notes below this table*
addr_middle	The contact's middle name.

Field Name	Description
	See notes below this table
addr_mobile	Mobile phone number. *See notes below this table*
addr_phone	Phone number. *See notes below this table*
addr_salutation	The contact's salutation. *See notes below this table*
addr_state	State. *See notes below this table*
addr_zip	Zip code. *See notes below this table*
attributes	A collection of additional attributes for this complex type.
can_bill_to	A 1/0 field indicating if the contact can be a billing contact.
can_ship_to	A 1/0 field indicating if the contact can be a shipping contact.
can_sold_to	A 1/0 field indicating if the contact can be a sold to contact.
code	Optional accounting system code for integration with external accounting systems.
created	Time the record was created.
customer_company	Import-only field to specify customer by company name.
customerid	The ID of the associated customer.
exported	Date and time the record was marked as exported"."
externalid	If record is imported from external system, store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
job_title	The contact's job title.
name	The name of the contact. This will be automatically generated if not supplied.
notes	Notes field.
picklist_label	Label as shown on form picklist.
updated	Time the record was updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: Prior to the October 2022 OpenAir release, you could not specify the address fields to be returned when reading contact records. 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_`).

You can now list the specific address information in the `fields` of the `ReadRequest` complex type to return only the address information required.

oaCostcategory

Use this complex type to add or update cost category information. oaCostcategory has the following children.

Field Name	Description
active	A 1/0 field indicating if this cost category is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the actual cost. This field is never populated. It is used only to satisfy subtotalling by actual cost.
notes	Notes.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaCostcenter

Use this complex type to specify cost center information. oaCostcenter has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is active.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
created	Time the record was created.
externalid	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the cost center.
notes	Cost center notes.

Field Name	Description
picklist_label	Label as shown on form picklist.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, delete, and upsert.

oaCosttype

Use this complex type to add or update cost type information. oaCosttype has the following children.

Field Name	Description
active	A 1/0 field indicating if this cost category is active.
attributes	A collection of additional attributes for this complex type.
cost_categoryid	The ID of the associated cost category.
created	Time the record was created.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the actual cost. This field is never populated. It is used only to satisfy subtotalling by actual cost.
notes	Notes.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaCurrency

Use the currency complex type to specify exchange rates that override market rates. oaCurrency has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
rate	The account's custom conversion rate.
symbol	The currency symbol.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaCurrencyrate

Use the currency rate complex type to read currency rates. oaCurrencyrate has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
cname	The name of the currency rate.
crate	The account's currency conversion rate.
csymbol	The currency symbol.
date	The date of the rate.
type	Blank for rates with date filled in, otherwise: ▪ PAST - conversion rates for dates prior to the first date in the table FUTURE - conversion rate for dates in the future

This complex type supports the read method.

oaCustField

Use this complex type to retrieve metadata about custom fields such as name, association, and picker type. oaCustField has the following children.

Field Name	Description
active	A 1/0 field indicating if this alert is active.
association	The association table or type of object this field is associated with. See Association Table or Type of Object for a list of associations.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
decpos	The decimal size of the field.
defnow	A 1/0 field indicating if date fields default to today.
description	The description of the custom field.
divider	A 1/0 field indicating whether to paint a divider.
divider_text	Optional divider text.
force_unique	A 1/0 field indicating if this field is unique.
hidden_data_entry	A 1/0 field indicating whether the custom field should be hidden on the data entry UI.
hint	The hint used on forms.
id	Unique ID. Automatically assigned by OpenAir.
maxlength	The maximum length of data in the field.
mover	A 1/0 field indicating if the selector should have mover controls.
name	The name of the custom field.
never_copy	A 1/0 field indicating if the field can be cloned.
next_seq	Next sequence number to use.

Field Name	Description
picker	The type of field for on screen representation: numeric, currency, date, text, textarea, check, radio, drop down, drop text, selector, or alloc_gr.
required	A 1/0 field indicating if this field is required.
rows	The number of display rows for text area fields
seq	The sequence number of the field.
size	The display size of the field on forms.
title	The title used on forms with this custom field.
updated	Time the record was last updated or modified.
userid	The ID of the user who created or owns this custom field.
valuelist	A list of values for radio groups and popup menu fields in csv format.

This complex type supports the following methods: read, add, modify, and upsert.

Association Table or Type of Object

The oaCustField complex type uses the following associations. The association is the name of the table that the custom field is related to. For more information, see the `association` field under the `cust_field` table in the OpenAir data dictionary using the following URL: `https://<account-domain>/database/single_user.html#cust_field`.

accounts_payable	event	receiving
agreement	fulfillment	request_item
attachment	invoice	revenuerecognitionrule
authorization	item	revenue_container
authorization_item	issue	revenue_stage
booking	manufacturer	revenue_recognition_transaction
booking_request	payment_type	schedule_by_day
carrier	payroll_type	schedule_request
category	phase	schedule_request_item
contact	product	slip
cost_center	project	ticket
customer	projectbillingrule	timesheet
customerpo	project_task	timetype
deal	proposal	todo
deal_booking_request	purchase_item	user
department	purchaseorder	vendor

discussion	purchaser	workspace
envelope	purchaserequest	

oaCustomField

Use this complex type to specify custom fields. oaCustomField has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the custom field.
type	Association of the custom field.
value	Value of the field for a specific record.

This complex type supports the following methods: modify (with custom equal to attribute) and read (with custom equal to attribute).

oaCustomer

Use this complex type for customer, client or patient information. The customer is the individual or company that is billed or expensed. oaCustomer has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is designated as an active customer.
addr_addr1	First line of the customer's address. *See notes below this table*
addr_addr2	Second line of the customer's address. *See notes below this table*
addr_addr3	Third line of the customer's address. *See notes below this table*
addr_addr4	Fourth line of the customer's address. *See notes below this table*
addr_city	Customer's city. *See notes below this table*
addr_country	Customer's country. *See notes below this table*
addr_email	Customer's email address. *See notes below this table*

Field Name	Description
addr_fax	Customer's fax number. *See notes below this table*
addr_first	Customer's first name. *See notes below this table*
addr_id	The ID of the associated address. *See notes below this table*
addr_last	Customer's last name. *See notes below this table*
addr_middle	Customer's middle name. *See notes below this table*
addr_mobile	Customer's mobile phone number. *See notes below this table*
addr_phone	Customer's phone number. *See notes below this table*
addr_salutation	Customer's salutation. *See notes below this table*
addr_state	Customer's state. *See notes below this table*
addr_zip	Customer's zip code. *See notes below this table*
attributes	A collection of additional attributes for this complex type.
`billing_addr_addr1` or `billingaddr_addr1`	First line on the billing address. *See notes below this table*
`billing_addr_addr2` or `billingaddr_addr2`	Second line on the billing address. *See notes below this table*
`billing_addr_addr3` or `billingaddr_addr3`	Third line on the billing address. *See notes below this table*
`billing_addr_addr4` or `billingaddr_addr4`	Fourth line on the billing address. *See notes below this table*
`billing_addr_city` or `billingaddr_city`	City on the billing address. *See notes below this table*
`billing_addr_country` or `billingaddr_country`	Country on the billing address. *See notes below this table*
`billing_addr_email` or `billingaddr_email`	Email address on the billing address.

Field Name	Description
	See notes below this table
`billing_addr_fax` or `billingaddr_fax`	Fax number on the billing address. *See notes below this table*
`billing_addr_first` or `billingaddr_first`	First name on the billing address. *See notes below this table*
`billing_addr_id` or `billingaddr_id`	The ID of the associated billing address. *See notes below this table*
`billing_addr_last` or `billingaddr_last`	Last name on the billing address. *See notes below this table*
`billing_addr_middle` or `billingaddr_middle`	Middle name on the billing address. *See notes below this table*
`billing_addr_` or `billingaddr_`	Mobile phone number on the billing address. *See notes below this table*
`billing_addr_mobile` or `billingaddr_mobile`	Phone number on the billing address. *See notes below this table*
`billing_addr_salutation` or `billingaddr_salutation`	Salutation on the billing address. *See notes below this table*
`billing_addr_state` or `billingaddr_state`	State on the billing address. *See notes below this table*
`billing_addr_zip` or `billingaddr_zip`	Zip code on the billing address. *See notes below this table*
billing_code	The customer billing code. Used in bulk invoicing.
billing_contact_id	The billing contact ID.
bus_typeid	Type of business this customer is in.
code	Optional user-defined code.
company	The company name.
company_sizeid	This customer's company size.
`contact_addr_addr1` or `contactaddr_addr1`	First line of the contact's address. *See notes below this table*
`contact_addr_addr2` or `contactaddr_addr2`	Second line of the contact's address. *See notes below this table*
`contact_addr_addr3` or `contactaddr_addr3`	Third line of the contact's address. *See notes below this table*
`contact_addr_addr4` or `contactaddr_addr4`	Fourth line of the contact's address.

OpenAir

Field Name	Description
	See notes below this table
`contact_addr_` or `contactaddr_`	Contact's city. *See notes below this table*
`contact_addr_city` or `contactaddr_city`	Contact's country. *See notes below this table*
`contact_addr_email` or `contactaddr_email`	Contact's email address. *See notes below this table*
`contact_addr_fax` or `contactaddr_fax`	Contact's fax number. *See notes below this table*
`contact_addr_first` or `contactaddr_first`	Contact's first name. *See notes below this table*
`contact_addr_id` or `contactaddr_id`	The ID of the associated contact address. *See notes below this table*
`contact_addr_last` or `contactaddr_last`	Contact's last name. *See notes below this table*
`contact_addr_middle` or `contactaddr_middle`	Contact's middle name. *See notes below this table*
`contact_addr_mobile` or `contactaddr_mobile`	Contact's mobile phone number. *See notes below this table*
`contact_addr_phone` or `contactaddr_phone`	Contact's phone number. *See notes below this table*
`contact_addr_salutation` or `contactaddr_salutation`	Contact's salutation. *See notes below this table*
`contact_addr_state` or `contactaddr_state`	Contact's state. *See notes below this table*
`contact_addr_zip` or `contactaddr_zip`	Contact's zip code. *See notes below this table*
cost_centerid	The ID of the associated cost center.
created	Time the record was created.
createtime	Same as the created field (for legacy systems).
currency	Currency for the money fields in the record. Also the default currency when an invoice is created.
customer_location_id	The internal ID of the customer location associated with the customer.

Field Name	Description
externalid	If the record was imported from an external system you store the unique external record ID here.
filterset_ids	Comma delimited list - filter sets this object belongs to.
hear_aboutid	How did they hear about us.
hierarchy_node_ids	Comma delimited list - hierarchy nodes this object belongs to.
id	Unique ID. Automatically assigned by OpenAir.
invoice_layoutid	The ID of the associated invoice layout.
invoice_prefix	Text to start every invoice number with.
invoice_text	Text to display on every invoice.
name	The nickname used for display in popup windows and lists.
notes	Notes about the customer.
picklist_label	Label as shown on form picklist.
primary_contactid	The billing contact ID.
rate	Hourly billing rate for this customer.
shipping_contactid	The shipping contact ID.
sold_to_contact_id	The sold to contact ID.
statements	A 1/0 field indicating if this customer can view statements.
ta_include	A 1/0 field indicating whether a Timesheet filterset is applied.
tb_approvalprocess	The approvalprocess_id of the invoice approval process. This field is mutually exclusive with 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. ■ If -1 then the approver is the owners manager. If -2 then the approver is the owners manager's manager.
te_include	A 1/0 field indicating whether an Expense Report filterset is applied.
terms	Standard payment terms for the customer. Textual description like Net 30.
territoryid	The territory for this customer.
type	A C/P field indicating whether this is Customer or a Prospect.
updated	Time the record was last updated or modified.
updatetime	Same as the updated field (for legacy systems).
userid	The user ID of the customer or owner.
web	Customer's Web address.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: Prior to the October 2022 OpenAir release, you could not specify the address fields to be returned when reading customer records. 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_`).

You can now list the specific address information in the `fields` of the `ReadRequest` complex type to return only the address information required.

oaCustomerLocation

Use this complex type for customer location information. oaCustomerLocation has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is an active customer location.
created	Time the record was created.
deleted	A 1/0 field indicating whether this customer location record was deleted
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the customer location.
notes	Notes.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaCustomerpo

Use this complex type to track money through projects and billings. oaCustomerpo has the following children.

Field Name	Description
acct_date	The accounting period date of the customerpo.
active	A 1/0 field indicating whether this is an active customerpo.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
created	Time the record was created.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer.

Field Name	Description
date	The date of the customerpo.
externalid	If the record was imported from an external system you store the unique oaCustomerpo,external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the customerpo.
notes	Notes.
number	The customerpo number.
picklist_label	Label as shown on form picklist.
total	The customerpo total. Dated by the date field.
updated	Time the record was last modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaCustomerpo_to_project

Use this complex type to create a many-to-many link between projects and customers. oaCustomerpo_to_project has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is an active customerpo.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
customerid	The ID of the associated customer.
customerpoid	The ID of the associated customerpo.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
projectid	The ID of the associated project.
updated	Time the record was last modified.
id	Unique ID. Automatically assigned by OpenAir.
attributes	A collection of additional attributes for this complex type.
customerpoid	The ID of the associated customerpo.
created	Time the record was created.
customerid	The ID of the associated customer.
active	A 1/0 field indicating whether this is an active customerpo.
updated	Time the record was last modified.

Field Name	Description
externalid	If the record was imported from an external system, you store the unique external record ID here.
projectid	The ID of the associated project.

This complex type supports the following methods: read, add, modify, and upsert.

oaDate

Use this complex type to specify date information. The correct value format is a four-digit number for year and a two-digit number for all other fields. oaDate has the following children.

Field Name	Description
year	Year (YYYY).
month	Month (MM).
day	Day (DD).
hour	Hour (HH).
minute	Minute (MM).
second	Second (SS).

oaDeal

Use this complex type to specify a potential sale to a prospect or customer. A deal can also be associated with a contact, an estimate, a todo, or an event. oaDeal has the following children.

Field Name	Description
active	Is this record active?
attributes	A collection of additional attributes for this complex type.
closed	When this deal was closed.
created	Time the record was created.
customerid	The ID of the associated customer.
exported	Date and time the record was marked as exported.
id	Unique ID. Automatically assigned by OpenAir.
name	The name/description of the deal.
notes	Notes for this deal.
opened	When this deal was first opened.
rating	The rating for this deal.

Field Name	Description
stage	The % of the work complete for this deal.
status	The status for this deal: O - Open, C - Closed, L - Lost
territoryid	The territory for this deal.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the read method.

oaDealcontact

Use this complex type to specify contact information for a deal. oaDealcontact has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
contactid	The related contact.
created	Time the record was created.
dealid	The deal ID.
id	Unique ID. Automatically assigned by OpenAir.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaDealschedule

Use this complex type to specify schedule information for a deal. A deal, among other things, consists of a total deal amount and a potential closing date. However, this total amount can be broken down into smaller portions, each with its own potential closing date. A dealschedule is one of these smaller amount portions, and is associated with a particular deal. oaDealschedule has the following children.

Field Name	Description
amount	The amount this portion of the deal is worth (in the currency of the deal). Dated by the date field."
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
date	The potential closing date for a deal portion.
dealid	ID of the deal associated with this deal portion.
id	Unique ID. Automatically assigned by OpenAir.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaDepartment

Use this complex type to specify department information and associate a user with a department. oaDepartment has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The name used for display in lists.
notes	Notes about the department.
picklist_label	Label as shown on form picklist.
updated	Time the record was last updated or modified.
userid	The user ID of the head of the department.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaEntitytag

Use this complex type to specify entity tag information. oaEntitytag has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
customerid	The ID of the associated customer.
default_for_entity	A 1/0 field indicating whether this is the default row for this entity.
end_date	End date for this entity_tag.
id	Unique ID. Automatically assigned by OpenAir.
projectid	The ID of the associated project.
start_date	Start date for this entity_tag.
tag_group_attribute_name	The name of the associated tag group attribute.
tag_group_attributeid	The ID of the associated tag_group_attribute.
tag_group_id	The ID of the associated tag group attribute.
updated	Time the record was last updated or modified.

Field Name	Description
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaEnvelope

Use this complex type to specify information about tickets in an envelope. Envelopes are used to group individual receipts into an expense report. oaEnvelope has the following children.

Field Name	Description
acct_date	The accounting period date of the envelope.
advance	The amount of any cash advance on the envelope.
approved	The date the envelope was approved.
approver	The userid of the envelope approver.
attachmentid	If non-zero, the attachment record associated with this envelope.
attributes	A collection of additional attributes for this complex type.
balance	The outstanding balance on the envelope.
created	Time the record was created.
currency	The currency this envelope is in.
currency_exchange_intolerance	A 1/0 field indicating if the record is within the specified foreign currency tolerance as defined in database data definitions.
date	The date of the envelope.
date_end	The ending date of the envelope (only used with auto-naming).
date_start	Starting date of the envelope (only used with auto-naming).
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
is_overlapping	Read only flag returns is an envelope overlaps with another envelope.
name	The name of the envelope.
notes	Notes about the envelope.
number	The envelope tracking number.
status	The status of the envelope: ▪ O - open ▪ S - submitted ▪ A - approved ▪ R - rejected

Field Name	Description
submitted	The date the envelope was submitted.
tax_locationid	Default tax location for this envelope.
thin_client_id	Used by thin clients to reconcile imported records.
total	The total value of all the tickets in the envelope.
totreimburse	The total amount of reimbursable expenses in the envelope.
tottickets	The total number of tickets in the envelope.
trip_reason	The reason for the trip.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: There is an OpenAir internal switch that can be enabled to allow API editing of approved expense reports. To use this feature, open a support ticket and request that the following switch be enabled: API will allow editing of approved Expense reports. See Creating a Support Case for instructions on how to create a support ticket.

oaError

Use this complex type to specify information about an error. oaError has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
code	Error code returned by the API.
comment	Additional comments.
text	Text of the error.

This complex type supports the read method.

oaEstimate

Use this complex type to specify estimate records for staffing, fixed costs, and discounts. It is used to create profit margin estimates for deals that are in the pipeline. oaEstimate has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
customerid	The ID of the associated customer.

Field Name	Description
dealid	The ID of the associated deal.
hide_expense	A 1/0 field indicating if expenses should be hidden in analysis report.
id	Unique ID. Automatically assigned by OpenAir.
name	The short description for the estimate.
notes	Notes about the estimate.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaEstimateadjustment

Use this complex type to specify estimate adjustment records. Estimate adjustments are the adjustment records associated with particular estimates. oaEstimateadjustment has the following children.

Field Name	Description
adjustment_type	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	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.
amount_type	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.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
estimateid	The ID of the associated estimate.
id	Unique ID. Automatically assigned by OpenAir.
name	The name for the estimate adjustment.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaEstimateexpense

Use this complex type to specify estimate expense records. oaEstimateexpense has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
date	Date for the expense.

Field Name	Description
description	The short description for the estimate.
estimateid	The ID of the associated estimate.
id	Unique ID. Automatically assigned by OpenAir.
itemid	The ID of the associated expense item.
markup	The amount of markup in percent or money as designated by markup_type field. Dated by the date field.
markup_type	A 1/0 field indicating the type of expense markup. ■ If 1 - then use percentage of the cost. ■ If 0 - then use the specific amount.
phaseid	The ID of the associated estimate phase.
price	The cost of the expense. Dated by the date field.
quantity	The quantity for the expense.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaEstimatelabor

Use this complex type to specify estimate staffing records. oaEstimatelabor has the following children.

Field Name	Description
amount	The number of hours or percentage of time associated with a given resource for a specific phase of an estimate. The actual typeis identified by as_percentage field.
amount_type	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.
attributes	A collection of additional attributes for this complex type.
billing_rate	The billing rate for the associated resource. Dated by the start_date field.
created	Time the record was created.
description	The short description for the estimate.
end_date	End date for resource assignment.
estimateid	The ID of the associated estimate.
id	Unique ID. Automatically assigned by OpenAir.
loaded_cost	The loaded cost for the associated resource. Dated by the start_date field.
phaseid	The ID of the associated estimate phase.
start_date	Start date for resource assignment.

Field Name	Description
updated	Time the record was last updated or modified.
userid	The ID of the associated resource.

This complex type supports the read method.

oaEstimatemarkup

Use this complex type to specify information about phases for the estimate. oaEstimatemarkup has the following children.

Field Name	Description
as_percentage	A 1/0 field indicating which expense markup to use: ▪ If 1 - then use percentage of the total, compute total markup. If 0 - then use the specific amount, compute percent markup.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
estimateid	The ID of the associated estimate.
id	Unique ID. Automatically assigned by OpenAir.
percent	The percentage markup to add to the total expense amount.
phaseid	The ID of the associated estimate phase.
total	The amount of expense (in the currency of the estimate) to use for this estimate in calculations.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaEstimatephase

Use this complex type to specify information about phases for the estimate. oaEstimatephase has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
estimateid	The ID of the associated estimate.
id	Unique ID. Automatically assigned by OpenAir.
name	The name for the estimate adjustment.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaEvent

Use this complex type to specify information about events. An 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. oaEvent has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
contact_id	The ID of the associated contact.
created	Time the record was created.
customerid	The ID of the associated customer.
dealid	The ID of the associated deal.
id	Unique ID. Automatically assigned by OpenAir.
name	The name or description of the event.
notes	Notes related to the event.
occurred	The date of the event.
updated	Time the record was last updated or modified.
userid	The ID of the user who created the event.

This complex type supports the following methods: read, add, modify, and upsert.

oaExpensePolicy

Use this complex type to specify information about expense policies. oaExpensePolicy has the following children:

Field Name	Description
all_items_allowed	A 1/0 field indicating that all expense items are allowed by this expense policy.
created	The time the record was created.
customerid	The ID of the associated customer .
deleted	A 1/0 field indicating if the record was deleted.
description	Optional information about expense policy.
id	Unique ID. Automatically assigned by OpenAir.
projectid	The ID of the project which expense policy is associated to. If zero/null then this is company default expense policy.
updated	The time the record was last modified.

This complex type supports the following methods: read, add, modify, and delete.

oaExpensePolicyItem

Use this complex type to specify information about items allowed for an expense policy. oaExpensePolicyItem has the following children:

Field Name	Description
created	The time the record was created.
currency	Currency of fixed/max price.
deleted	A 1/0 field indicating if the record was deleted.
expense_policyid	The ID of the expense policy which this item belongs to.
id	Unique ID. Automatically assigned by OpenAir.
itemid	The ID of the item which this record belongs to.
price_fixed	If set this item has defined fixed price which cannot be overridden in the ticket form.
price_max	If set this item has a defined maximum price.
updated	The time the record was last modified.

This complex type supports the following methods: read, add, modify, and delete.

oaFieldAttribute

Use this complex type to supply additional attributes to all other complex types, with the exception of oaAddress, oaFieldAttribute, oaDate, and oaModule. You can use externalid and name fields as a foreign key, allowing you to read, add, createUser, modify, and upsert records in a single step instead of querying a record to first obtain the internal ID.

Using an externalid field as a foreign key

1. Set the ID field to the externalid field value instead of internal ID field value.
2. Create a collection of oaFieldAttribute objects, one for each field that needs to be overridden.
3. Assign the collection to ‘attributes' member of the target record. For code examples, refer to the following: Example II. modify using external_id as foreign key lookup field C# and Example II. externalid as foreign key lookup field Java.

oaFieldAttribute has the following children.

Field Name	Description
name	Specifies the type of lookup: external - lookup the field using external_id field and name- lookup by the name field value.
value	A colon separated list consisting of two or three values. The value can either be external or name. Each is defined as follows: - External - matching record type to process lookup for - Name - field name as it exists in the object In addition, there can be an Optional flag 1 - instructs API to process comma separated list of values. This flag is not required for regular fields.

This complex type supports the following methods: read, add, createUser, modify, and upsert.

oaFilterset

Use this complex type to list names and IDs that define the table/id pairs to be filtered for each filterset. oaFilterset has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is designated as an active filter set.
all_access	A 1/0 field indicating this filterset does not filter anything and cannot be deleted.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
default_filter_set	A 1/0 field indicating whether this is the default new-user filterset.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the filterset.
notes	Notes related to the filterset.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaForexInput

Use this complex type to allow multi-currency accounts to override historical and future currency conversion rates. oaForexInput has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
base	The currency symbol used as a base currency for the currency conversion table.
created	Date the record was created.
enddate	Optional end date for currency being set.
future	1 - if this is for future overrides. If used, start and end dates must be blank.
past	1 - if this is for past overrides. If used, start and end dates must be blank.
rate	Rate against the base currency for the account.
startdate	Optional start date for currency being set.
symbol	Currency symbol. Must be for one of the multiple currencies enabled in the account.
updated	Date the record was last modified.

This complex type supports the following methods: read, add, modify, and upsert.

Note: There is an OpenAir internal switch that allows you to specify the rate against a user-defined currency. To use this feature, open a support ticket and request that the following switch be enabled: Enable user defined reporting currencies. See Creating a Support Case for instructions on how to create a support ticket.

oaFulfillment

Use this complex type to specify information about the receipt of goods and services ordered by a purchase order. oaFulfillment has the following children.

Field Name	Description
acct_date	The accounting period date of the fulfillment.
attributes	A collection of additional attributes for this complex type.
carrier_id	Associated carrier ID.
created	Time the record was created.
date	Date of the fulfillment.
id	Unique ID. Automatically assigned by OpenAir.
notes	Fulfillment description notes.
purchase_item_id	Associated purchase item ID.
purchaseorder_id	Associated purchase order ID.
purchaserequest_id	Associated purchase request ID.
quantity	The quantity received.
request_item_id	Associated request item ID.
slip_id	The ID of the associated slip if this expense was billed to a time bill.
updated	Time the record was last updated or modified.
waybill_number	The waybill number.

This complex type supports the read method.

oaHierarchy

Use this complex type to specify hierarchy information. oaHierarchy has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is designated as an active hierarchy.
attributes	A collection of additional attributes for this complex type.

Field Name	Description
available_as_column	A 1/0 field indicating whether this hierarchy is available as a (customer, project or user) list column. Only one hierarchy per
created	Time the record was created.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The hierarchy name.
notes	Notes related to the hierarchy.
primary_dropdown_filter	A 1/0 field indicating if this hierarchy is used as a drop-down filter.
primary_user_filterset	A 1/0 field indicating if this hierarchy determines filter set access for projects.
required	A 1/0 field indicating whether this hierarchy should be a required element on the object type form.
requireonform	A 1/0 field indicating whether this hierarchy should be added to the object type form.
type	The type (table name) of the hierarchy: customer, project, or user.
type can be displayed as a column in a list."	
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaHierarchyNode

Use this complex type to specify information about a hierarchy node. oaHierarchyNode has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
externalid	If the record was imported from an external system, you store
hierarchyid	The ID of the associated hierarchy.
id	Unique ID. Automatically assigned by OpenAir.
isalevel	A 1/0 field indicating if this node is a level.
isanode	The name of the hierarchy node.
levelid	The ID of the associated hierarchy level.
name	The hierarchy name.
notes	Notes related to the hierarchy node.

Field Name	Description
parentid	The hierarchy_node ID of our immediate ancestor. If zero/null, is a top-level node.
recordid	The record ID if not a node.
the unique external record ID here."	
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaHistory

Use this complex type to specify history events. oaHistory has the following children.

Field Name	Description
action	The approval action: S - Submittal, P - Pending, A - Acceptance, R - Rejection, U - unapproval.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
date	The date associated with this history event.
envelopeid	The ID of the associated envelope.
id	Unique ID. Automatically assigned by OpenAir.
notes	Notes associated with the history event.
userid	The ID of the user associated with this history event.

For more information on its use, refer to the Example I. read equal to C# and Example I. read equal to Java. Note that the read method for oaHistory only works with the “equal to” retrieval method.

oaImportExport

Use this complex type to specify table and ID pairs corresponding to an external application. It can be used in conjunction with read and the not-exported filter to request records that have not been exported. oaImportExport has the following children.

Field Name	Description
application	String describing the application making the association.
attributes	A collection of additional attributes for this complex type.
exported	Time of the last export from OpenAir. Required on import.
externalid	External identifier for the application.
id	Internal ID of the actual record (slip, task, etc.) in its native table.

Field Name	Description
imported	Time of the last import to OpenAir. Required on import.
type	Complex type name of the exported record: Slip, Task, Projectassign, etc. Refer to the Types table. Please note that these names are case sensitive."

This complex type supports the following methods: read, add, modify, and upsert.

oaInvoice

Use this complex type to specify invoice information for the header. oaInvoice has the following children.

Field Name	Description
access_log	The mailing and access history of the invoice, such as when the customer accessed it.
accounting	A 1/0 field indicating if an invoice has been sent to an accounting partner.
acct_date	The accounting period date of the invoice.
approval_status	A one-character string indicating the approval status of the invoice. Only used if invoice approvals are used. Possible values: ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected
approved	Date the invoice was approved.
attachmentid	The ID of the associated attachment.
attributes	A collection of additional attributes for this complex type.
balance	The outstanding balance on the invoice. Dated by the date field.
contactid	The contact ID for this invoice.
created	Time the record was created.
credit	The amount of any credit against the invoice. Dated by the date field.
credit_reason	The reason for the credit.
credit_rebill_status	Credit/Rebill status for the original invoice: C = Credit Initiated and R = Re-Bill.
currency	The currency this invoice is in.
customerid	The ID of the associated customer.
date	The date of the invoice.
draw	The amount of any draw against retainer for this invoice. Dated by the draw_date field.
draw_date	The date of the draw.
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.

Field Name	Description
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
invoice_layoutid	The ID of the associated invoice layout.
notes	Notes associated with the invoice.
number	The invoice number.
original_invoiceid	The original invoice ID for credit invoices.
paperrequest	Date the user requested that a paper invoice be mailed.
papersend	Date the paper invoice was actually mailed.
payment_termsid	The ID of the associated payment terms.
shipping_contactid	The shipping contact ID for this invoice.
submitted	Date the invoice was submitted.
tax	The tax total for the invoice. Dated by the date field.
tax_federal	The federal tax total for the invoice. Dated by the date field.
tax_gst	The GST tax for the invoice. Dated by the date field.
tax_hst	The HST tax for the invoice. Dated by the date field.
tax_pst	The PST tax for the invoice. Dated by the date field.
tax_state	The state tax total for the invoice. Dated by the date field.
terms	Payment terms for this invoice.
total	The invoice total. Dated by the date field.
updated	Time the record was updated.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: Review the following guidelines:

- If not all invoices are returned from an API request, determine whether the following OpenAir internal switch is enabled: API will allow editing of approved Invoices. Speak with OpenAir Professional Services or create a support ticket. See the help topic Troubleshooting for instructions on creating a support ticket.
- To set the `payment_termsid` field, the **Save Payment Terms Internal ID on Invoice Records** optional feature must be enabled for your account. The field is empty otherwise.

oaInvoiceLayout

Use this complex type to read invoice layout information. oaInvoiceLayout has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	The name used for display in popups and lists.
updated	Time the record was last modified.

This complex type supports the read method.

oaIssue

Use this complex type to specify issue information. oaIssue has the following children.

Field Name	Description
attachment_id	If non-zero, the attachment record associated with this issue.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
customer_id	The ID of the associated customer.
date	The date of the issue.
date_resolution_expected	The date the issue is expected to be resolved.
date_resolution_required	The date the issue is required to be resolved.
date_resolved	The date the issue was resolved.
description	A short description of the issue, a synopsis.
id	Unique ID. Automatically assigned by OpenAir.
issue_category_id	The ID of the associated issue category.
issue_notes	The description of the issue.
issue_severity_id	The ID of the associated issue severity.
issue_source_id	The ID of the associated issue source.
issue_stage_id	The ID of the associated issue stage.
issue_status_id	The ID of the associated issue status.
name	The name of the issue (Prefix + number).
number	The issue number that increments by 1.
owner_id	The ID of the associated user creating the issue.
prefix	A static alphanumeric issue number prefix.
priority	The priority of the task (1 - 100).

Field Name	Description
project_id	The ID of the associated project.
project_task_id	The ID of the task within the associated project.
resolution_notes	The description of the resolution.
updated	Time the record was updated.
user_id	The ID of the user assigned to the issue.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaIssueCategory

Use this complex type to specify information about the issue category. oaIssueCategory has the following children.

Field Name	Description
active	A 1/0 field indicating whether this issue category is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the issue category.
notes	Notes associated with the issue category.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaIssueSeverity

Use this complex type to specify information about the issue severity. oaIssueSeverity has the following children.

Field Name	Description
active	A 1/0 field indicating whether this issue severity is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the issue severity.
notes	Notes associated with the issue severity.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaIssueSource

Use this complex type to specify information about the issue source. oaIssueSource has the following children.

Field Name	Description
active	A 1/0 field indicating whether this issue source is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the issue source.
notes	Notes associated with the issue source.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaIssueStage

Use this complex type to specify information about the issue stage. oaIssueStage has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
considered_closed	A 1/0 field indicating whether issues in this stage are considered closed.
created	Time the record was created.
default_for_new	A 1/0 field indicating whether this is the default stage for new issues.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the issue stage.
notes	Notes associated with the issue stage.
position	The position of the stage.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaIssueStatus

Use this complex type to specify information about the issue status. oaIssueStatus has the following children.

Field Name	Description
active	A 1/0 field indicating whether this issue status is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the issue status.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaItem

Use this complex type to specify item information such as expense item, expense type, expense, inventory item. oaItem has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is designated as an activecustomer.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
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	The ID of the associated cost center.
cost_is_fixed	A 1/0 field indicating whether the user is allowed to change the cost on a receipt.
created	Time the record was created.
currency	Currency for the money fields in the record.
externalid	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The item name.
tax_location_id	The ID of the associated tax location.
taxable	A 1/0 field indicating whether this item is taxable, VAT-able, etc.
tp_comp	Ticket policy comparison: ▪ ge - greater than or equal to ▪ gt - greater than
tp_cost	The policy threshold amount.
tp_notes_required	Notes are required if the ticket triggers the policy.
tp_unit_or_total	The ticket policy is applied against:

Field Name	Description
	■ U - Unit price ■ T - Total
type	The type of item. Add new types when type-specific information can be captured for the slip or ticket templated from this item: ■ R - for regular item. ■ M - for mileage item.
unitm	The unit of measure for the item, i.e., EA.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaItemToUserLocation

Use this complex type to specify associations between oaItem, oaUserLocation, and oaTaxLocation. oaItemToUserLocation has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
itemid	The ID of the associated item.
tax_locationid	The ID of the associated tax location.
updated	Time the record was last updated or modified.
user_locationid	The location ID for this user.

This complex type supports the following methods: read, add, modify, and upsert.

oaJobcode

Use this complex type to specify job code information. oaJobcode has the following children.

Field Name	Description
active	A 1/0 field indicating if this is an active job code.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
created	Time the record was created.
currency	Currency for the money fields in the record.
externalid	If the record was imported from an external system you storet the unique external record ID here.

Field Name	Description
id	Unique ID. Automatically assigned by OpenAir.
loaded_cost	Loaded cost for this job code.
name	The name for the job code.
notes	Notes associated with the job code.
updated	Time the record was last updated or modified.
userid_fte	The user ID of the FTE (Full Time Equivalent) generic resource.

This complex type supports the following methods: read, add, modify, and upsert.

oaJobCodeUsed

Use this complex type to read information about which tables use job codes. oaJobCodeUsed has the following children.

Field Name	Description
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
position	Position in the lookup rule.
table_name	The name of the table that uses a job code.
updated	Time the record was last updated or modified.
used_by	What it is used by. Two possible values: ■ 'a' for tasks ■ 's' for slips.

This complex type supports the following methods: read.

oaLeave_accrual_rule

Use this complex type to specify leave accrual rule information. oaLeave_accrual_rule has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is an active billing rule.
amount	The number of hours per period.
attributes	A collection of additional attributes for this complex type.
cap	Number of hours to cap the accrual at.
category_filter	CSV list of categories that will trigger a draw down.

Field Name	Description
created	Time the record was created.
draw_down_when	Generate the draw down when: ■ R - When leave accrual is run. ■ A - When a timesheet is approved.
grace_days	How many days is the grace period before accrued time is lost.
id	Unique ID. Automatically assigned by OpenAir.
lose_how	How is accrued time lost: ■ N - Never ■ A - The users anniversary date ■ Y - End of year
name	The name for the leave accrual rule.
notes	Notes associated with the leave accrual rule.
period	The period for the cap.
project_filter	CSV list of projects that will trigger a draw down.
project_task_filter	CSV list of project_tasks that will trigger a draw down.
timetype_filter	CSV list of timetypes that will trigger a draw down.
timing	When the accrual is applied: ■ S - start of the period. ■ E - end of the period.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaLeave_accrual_rule_to_user

Use this complex type to map leave accrual rules to users. oaLeave_accrual_rule_to_user has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
end_date	The date the accrual rule stops applying to the user.
id	Unique ID. Automatically assigned by OpenAir.
leave_accrual_ruleid	The ID of the associated accrual rule.
start_date	The date the accrual rule starts applying to the user. This is required.
transfer_balance_to	ID of leave_accrual_rule_to_user record where balance should be transferred to.

Field Name	Description
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, and upsert.

oaLeave_accrual_transaction

Use this complex type to specify leave accrual transaction information. oaLeave_accrual_transaction has the following children.

Field Name	Description
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.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
date	The date of the transaction.
from_run	Indicates if this was generated from a run the leave accrual rules.
id	Unique ID. Automatically assigned by OpenAir.
leave_accrual_ruleid	The ID of the associated accrual rule. This is a required field.
notes	Notes associated with the leave accrual transaction.
taskid	The ID of the associated task if this is a draw down against a timesheet entry.
type	Indicates type of draw down: the type of the amount field. ▪ D - draw down. ▪ A - Accrual.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, and upsert.

oaLoadedCost

Use this complex type to specify loaded cost values for a user. oaLoaded Cost has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
cost	The fully loaded hourly cost of the user.
created	Time the record was created.

Field Name	Description
currency	The currency of the cost field.
current	A 1/0 field indicating if this is the current loaded cost record.
customerid	The ID of the associated customer.
end	End date for the loaded cost for historical records.
Ic_level	■ If multiple loaded costs are used, this holds the level of loaded cost ■ 0 - primary loaded cost ■ 1 - secondary loaded cost ■ 2 - tertiary loaded cost
id	Unique ID. Automatically assigned by OpenAir.
project_taskid	The ID if this loaded cost is associated with a specific projec task. If this field is used, the project_id and customer_id must be empty.
projectid	The ID if this loaded cost is associated with a specific project.
start	Start date for the loaded cost for historical records.
updated	Time the record was last updated or modified.
userid	ID of the user.

This complex type supports the following methods: read, add, modify, and upsert.

oaModule

Use this complex type to specify Module availability. oaModule has the following children.

Field Name	Description
abbr	Abbreviation within OpenAir.
enabled	A 1/0 field indicating whether the module is enabled.

This complex type supports the read method.

oaNewsfeed

Use this complex type to specify project status newsfeed data. oaNewsfeed has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaNewsfeedMessage

Use this complex type to specify project status newsfeed message data. oaNewsfeedMessage has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
authorid	The ID of the user who created the record.
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	Time the record was created.
editorid	The ID of the user who last updated or modified the record.
id	Unique ID. Automatically assigned by OpenAir.
newsfeedid	The ID of the associated project status newsfeed.
tagid	The ID of the project status tag. The tagid values correspond to pre-defined 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	The title of the newsfeed entry. Limited to 125 characters. Optional.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaPayment

Use this complex type to specify payment information. oaPayment has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
bulk_paymentid	The ID of the bulk_payment transaction if this payment is part of a bulk_payment.
created	Time the record was created.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer if this is a retainer payment.
date	The date of the payment.
externalid	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
invoice_number	The associated invoice number if a payment against a specific invoice.
invoiceid	The associated invoice ID if a payment against a specific invoice.
notes	Notes associated with the payment.

Field Name	Description
total	The payment total. Dated by the date field.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaPaymentterms

Use this complex type to specify payment terms information. oaPaymentterms has the following children.

Field Name	Description
active	A 1/0 field indicating where this is designated as an active shipping terms 1/0.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
default_terms	A 1/0 field indicating whether this is the default payment terms (used for Customers, Vendors, Invoices and POs).
id	Unique ID. Automatically assigned by OpenAir.
name	The payment terms name.
notes	Notes associated with the payment terms.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaPaymenttype

Use this complex type to specify payment type information. Payment types are used to specify the payment methods for individual receipts. oaPaymenttype has the following children.

Field Name	Description
active	A 1/0 field specifying if the type is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
default_payment_type	A 1/0 field indicating whether this is the default payment_type for receipts.
default_status	Default receipt status, e.g. R => Reimbursable, N => Non-reimbursable.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the payment type.
notes	Notes associated with the payment type.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaPayrolltype

Use this complex type to specify payroll type information. oaPayrolltype has the following children.

Field Name	Description
active	A 1/0 field specifying whether this is an active payrolltype.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the payroll type.
notes	Notes associated with the payroll type.
picklist_label	Label as shown on form picklist.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaPendingBooking

Use this complex type to read pending booking information. oaPendingBooking has the following children.

Field Name	Description
approval_status	A one-character string indicating the approval status of the booking request. Possible values: ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected
as_percentage	A 1/0 field indicating which of the fields (hours or percentage) are actual, and which is derived. ■ 1 = percentage is actual and hours is derived. ■ 0 = hours in actual and percentage is derived.
attributes	A collection of additional attributes for this complex type.
booking_typeid	The ID of the associated booking_type.
created	Time the record was created.
customerid	The ID of the associated customer.
date_approved	The date the booking request was approved.

Field Name	Description
date_submitted	The date the booking_request was submitted.
enddate	The end date of the booking.
endtime	End time.
externalid	If the record was imported from an external system you store the unique external record ID here.
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	Unique ID. Automatically assigned by OpenAir.
job_codeid	The ID of the associated job code.
locationid	The location ID for this booking.
notes	Booking notes.
notify_owner	A 1/0 field indicating whether to send email to the requestor when the booking is modified.
ownerid	The ID of the associated user creating the booking.
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_profile_id	The ID of the associated project assignment profile.
project_taskid	The ID of the task within the associated project.
projectid	The ID of the associated project.
repeatid	The ID of the associated repeating event.
resource_request_queue_id	The ID of the associated resource request queue.
startdate	The start date of the booking.
starttime	Start time.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the read method.

oaPreference

Use this complex type to specify preference or setting information. oaPreference has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.

Field Name	Description
group_name	Optional group name for the preference.
id	Unique ID. Automatically assigned by OpenAir.
name	The name for the preference.
setting	The preference data is stored in this field.
updated	Time the record was last updated or modified.
userid	If the preference is specific to a user, this will be the user ID.

This complex type supports the following methods: read, add, modify, and upsert.

oaProduct

Use this complex type to specify product information. Products are used to create request items, which ultimately appear as line items on purchase orders. oaProduct has the following children.

Field Name	Description
active	A 1/0 field indicating that this is active.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
created	Time the record was created.
currency	The currency this cost is quoted in.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
manufacturer_part	The manufacturer's part number, SKU or other unique identification for this product.
manufacturerid	The manufacturer of this product.
name	The name for the product. This shows up on all the product pop-up windows in the application.
notes	Notes associated with the product.
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	A 1/0 field indicating whether this item is taxable.
um	The unit of measure for the product, i.e., EA.
updated	Time the record was last updated or modified.
vendor_id	The preferred vendor from whom to purchase this product.
vendor_sku	The preferred vendor's sku for this product.

This complex type supports the following methods: read, add, modify, and upsert.

oaProject

Use this complex type to specify project information and to create one project from another project. Indicate the rules and settings to copy. oaProject has the following children.

Field Name	Description
active	A 1/0 field indicating an active project.
attachmentid	If non-zero, the attachment record associated with this project.
attributes	A collection of additional attributes for this complex type.
auto_bill	A 1/0 field, 1 if the project can be auto-billed.
auto_bill_cap	A 1/0 field, 1 if the project should have a cap on auto-billings.
auto_bill_cap_value	The auto-billings cap amount (in the currency of the project).
auto_bill_override	A 1/0 field, 1 if the project overrides the global auto_billing settings. The auto_bill table will hold the settings for the project.
az_approvalprocess	The approvalprocess_id of the project expense authorization approval process. This field is mutually exclusive with az_approver.
az_approver	The user_id of the project expense authorization approver if this is a single approver process. This field is mutually exclusive with az_approvalprocess. ▪ If -1 then the approver is the owners manager ▪ If -2 then the approver is the owners manager's manager ▪ If -3 then the approver is the project owner ▪ If -4 then the approver is self
billing_code	The project billing code. Used in bulk invoicing.
billing_contactid	The billing contact ID if different than the customer designated billing contact.
br_approvalprocess	The approvalprocess_id of the project booking request approval process. This field is mutually exclusive with br_approver.
br_approver	The user_id of the project booking request approver if this is a single approver process. This field is mutually exclusive with br_approvalprocess. ▪ If -1 then the approver is the owners manager ▪ If -2 then the approver is the owners manager's manager ▪ If -3 then the approver is the project owner ▪ If -4 then the approver is self
budget	The budgeted revenue for the project.
budget_time	The budgeted amount of time for the project, in hours.
category_filter	A category (service) filter. This will hold a list of the categories that are allowed to book time to this project.
code	Optional system code for integration with external accounting systems.

Field Name	Description
copy_approvers	Duplicates project approvers. A 1/0 field, 1 if the project approvers should be copied.
copy_custom_fields	Duplicates custom fields. A 1/0 field, 1 if the custom fields should be copied.
copy_dashboard_settings	Duplicates dashboard settings. A 1/0 field, 1 if the dashboard settings should be copied.
copy_invoice_layout_settings	Duplicates invoice layout settings. A 1/0 field, 1 if the invoice layout settings should be copied.
copy_issues	Duplicates issues. A 1/0 field, 1 if the issues should be copied.
copy_loaded_cost	Duplicates project pricing information. A 1/0 field, 1 if the loaded costs should be copied.
copy_notification_settings	Duplicates notification settings. A 1/0 field, 1 if the notification settings should be copied.
copy_project_billing_auto_settings	Duplicates project auto-bill settings. A 1/0 field, 1 if the auto-bill settings should be copied.
copy_project_billing_rules	Duplicates project billing rules. A 1/0 field, 1 if the billing rules should be copied.
copy_project_pricing	Duplicates project pricing information. A 1/0 field, 1 if the project pricing information should be copied.
copy_revenue_recognition_auto_settings	Duplicates revenue recognition rules auto-run settings. A 1/0 field, 1 if the auto-run settings should be copied.
copy_revenue_recognition_rules	Duplicates revenue recognition rules. A 1/0 field, 1 if the recognition rules should be copied.
copy_revenuerecognition_auto_settings	Duplicate of copy_revenue_recognition_auto_settings.
cost_centerid	The ID of the associated cost center.
create_workspace	A 1/0 field, 1 if an associated workspace is automatically created by the API. The project owner becomes the workspace owner.
created	Time the record was created.
currency	Currency for the money fields in the record.
customer_name	The customer's name.
customerid	The ID of the associated customer.
externalid	If the record was imported from an external system, you store the unique external record ID here.
filterset_ids	A comma separated list of filter set IDs this record should be part of.
filtersetids	A comma separated list of filter set IDs this record should be part of.
finish_date	The calculated finish date of the project.
hierarchy_node_ids	The ID of the hierarchy node for this project.
id	Unique ID. Automatically assigned by OpenAir.

Field Name	Description
invoice_layoutid	The ID of the associated invoice layout.
invoice_text	Text to display on every invoice.
is_portfolio_project	A 1/0 field - 1 if the project is a portfolio project.
locationid	The location ID for this project (DEPRECATED).
main_contactid	The ID of the main project contact.
message	Dashboard message.
msp_link_type	If imported from Microsoft project, this field describes the state: ▪ "" not imported from MSP ▪ 'I' imported and locked for edit ▪ 'U' imported but unlocked for edit
name	The project name. This shows upon all the project pop-up windows in the application.
newsfeedid	The ID of the associated project status newsfeed.
no_dirty	A 1/0 field, 1 if we want this project to be marked dirty when it has finished the current recalc.
notes	Notes associated with this project.
notify_assignees	A 1/0 field indicating whether to send email to assigned users whenever a task in this project is added, modified, or deleted.
notify_issue_assigned_to	A 1/0 field indicating whether to send email to a user whenever assigned to an issue.
notify_issue_closed_assigned_to	A 1/0 field indicating whether to send email to the assigned user whenever an issue is moved to a considered closed issue stage.
notify_issue_closed_customer_owner	A 1/0 field indicating whether to send email to the customer owner whenever an issue is moved to a considered closed issue stage.
notify_issue_closed_project_owner	A 1/0 field indicating whether to send email to the project owner whenever an issue is moved to a considered closed issue stage.
notify_issue_created_customer_owner	A 1/0 field indicating whether to send email to the customer owner whenever an issue is created.
notify_issue_created_project_owner	A 1/0 field indicating whether to send email to the project owner whenever an issue is created.
notify_owner	A 1/0 field indicating whether to send email to the project owner when an ownership change is made.
notify_sr_submitted_project_owner	A 1/0 field indicating whether to send email to the project owner when a schedule request is submitted for a user booked or assigned to the project.
only_owner_can_edit	A 1/0 field indicating whether only the project owner can edit this project.
payroll_type_filter	A payroll type filter. This holds a list of the payroll types that are allowed to book time to this project.

Field Name	Description
picklist_label	Label as shown on form picklist.
pm_approver_1	The user_id of the project approver 1 that is substituted into the approval processes. If -6 then the approver is the 1st additional project approver.
pm_approver_2	The user_id of the project approver 2 that is substituted into the approval processes. If -7 then the approver is the 2nd additional project approver.
pm_approver_3	The user_id of the project approver 3 that is substituted into the approval processes. If -8 then the approver is the 3rd additional project approver.
po_approvalprocess	The approvalprocess_id of the project purchase order approval process. This field is mutually exclusive with po_approver.
po_approver	The user_id of the project purchase order approver if this is a single approver process. This field is mutually exclusive with po_approvalprocess. ▪ If -1 then the approver is the owners manager ▪ If -2 then the approver is the owners manager's manager ▪ If -3 then the approver is the project owner ▪ If -4 then the approver is self
portfolio_projectid	The ID of the associated portfolio project.
pr_approvalprocess	The approvalprocess_id of the project purchase request approval process. This field is mutually exclusive with pr_approver.
pr_approver	The user_id of the project purchase request approver if this is a single approver process. This field is mutually exclusive with pr_approvalprocess. ▪ If -1 then the approver is the owners manager ▪ If -2 then the approver is the owners manager's manager ▪ If -3 then the approver is the project owner ▪ If -4 then the approver is self
project_locationid	The location ID for this project.
project_stageid	The ID of the project stage.
rate	The hourly billing rate.
rate_cardid	The ID of the associated rate card if using rate cards.
rv_approvalprocess	The approvalprocess_id of the project revenue_container approval process. This field is mutually exclusive with rv_approver.
rv_approver	The user_id of the project revenue_container approver if this is a single approver process. This field is mutually exclusive with rv_approvalprocess. ▪ If -1 then the approver is the owners manager ▪ If -2 then the approver is the owners manager's manager ▪ If -3 then the approver is the project owner

Field Name	Description
	▪ If -4 then the approver is self
sga_labor	The allocated cost (SG and A) overhead percentage to apply to labor for profitability analysis.
shipping_contact_id	The shipping contact ID if different than the customer designated shipping contact.
sold_to_contact_id	The sold to contact ID if different than the customer designated sold to contact
start_date	The scheduled starting date of the project.
sync_workpace	A 1/0 field indicating whether to keep project resources in sync with linked workspace members.
ta_approvalprocess	The approvalprocess_id of the project timesheet approval process. This field is mutually exclusive with ta_approver.
ta_approver	The user_id of the project timesheet approver if this is a single approver process. This field is mutually exclusive with ta_approvalprocess. ▪ If -1 then the approver is the owners manager ▪ If -2 then the approver is the owners manager's manager ▪ If -3 then the approver is the project owner ▪ If -4 then the approver is self
ta_include	A 1/0 field indicating whether a Timesheet filterset is applied.
tax_location_name	Name of the tax location.
tax_locationid	The ID of the associated tax location.
tb_approvalprocess	The approvalprocess_id of the project invoice approval process. This field is mutually exclusive with tb_approver.
tb_approver	The user_id of the project invoice approver if this is a single approver process. This field is mutually exclusive with tb_approvalprocess. ▪ If -1 then the approver is the owners manager ▪ If -2 then the approver is the owners manager's manager ▪ If -3 then the approver is the project owner ▪ If -4 then the approver is self
te_approvalprocess	The approvalprocess_id of the project expense report approval process. This field is mutually exclusive with te_approver.
te_approver	The user_id of the project expense report approver if this is a single approver process. This field is mutually exclusive with te_approvalprocess. ▪ If -1 then the approver is the owners manager ▪ If -2 then the approver is the owners manager's manager ▪ If -3 then the approver is the project owner ▪ If -4 then the approver is self
te_include	A 1/0 field indicating whether an Expense Report filterset is applied.

Field Name	Description
template_project_id	ID of the project from which tasks and phases, billing rules, revenue recognition rules and other items will be copied.
timetype_filter	A timetype filter. This will hold a list of the timetypes that are allowed to book time to this project.
updated	Time the record was last updated or modified.
user_filter	Also allow these users to edit the project if the only_owner_can_edit switch is on.
userid	The user ID of the project owner.
te_approver	The user_id of the project expense report approver if this is a single approver process. This field is mutually exclusive with te_approvalprocess. ▪ If -1 then the approver is the owners manager ▪ If -2 then the approver is the owners manager's manager ▪ If -3 then the approver is the project owner ▪ If -4 then the approver is self
te_include	A 1/0 field indicating whether an Expense Report filterset is applied.
template_project_id	ID of the project from which tasks and phases, billing rules, revenue recognition rules and other items will be copied.
timetype_filter	A timetype filter. This will hold a list of the timetypes that are allowed to book time to this project.
updated	Time the record was last updated or modified.
user_filter	Also allow these users to edit the project if the only_owner_can_edit switch is on.
userid	The user ID of the project owner.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: 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. Use a “1” to indicate the settings and rules you want copied. Available fields begin with copy_.

oaProjectAssignmentProfile

Use this complex type to assign profiles to projects. oaProjectAssignmentProfile has the following children.

Field Name	Description
allocation	The percentage of time the associated user is allocated to this task.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
customer_id	The ID of the associated customer.

Field Name	Description
id	Unique ID. Automatically assigned by OpenAir.
job_codeid	The ID of the associated job code.
project_groupid	The ID of the project group if the user was assigned as part of a project group.
project_id	The ID of the project to which this user is assigned.
updated	Time the record was last updated or modified.
user_id	The ID of the user assigned to this task.

This complex type supports the following methods: read, add, modify, and upsert.

oaProjectassign

Use this complex type for the assignment by project feature to track users assigned to a project. oaProjectassign has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
customerid	The ID of the associated customer.
id	Unique ID. Automatically assigned by OpenAir.
name	The project_assignment_profile name.
projectid	Id of the project to which this project_assignment_profile is associated.
updated	Time the record was last updated or modified OpenAir.
user_filter	A user filter list. The project_assignment_profile only be applied to the users in this list.

This complex type supports the following methods: read, add, modify, and upsert.

oaProjectbillingrule

Use this complex type to specify project billing rules. oaProjectbillingrule has the following children.

Field Name	Description
accounting_period_id	The ID of the associated accounting period.
acct_date	The accounting period date to assign to the transaction.
acct_date_how	The accounting period date of the transaction is determined by: ■ 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

Field Name	Description
	▪ M - set by the specified accounting date ▪ P - set by the specified accounting period
active	A 1/0 field indicating whether this is an active billing rule.
adjust_if_capped	If a transaction will exceed the cap, should it be adjusted to fit under the cap.
agreementid	The ID of the associated agreement.
amount	The amount for a fixed fee rule.
assigned_user	The user to assign to fixed fee billings.
attributes	A collection of additional attributes for this complex type.
backout_gst	If they are using GST/HST/PST taxes, back out the GST/HST taxes from re-billed expenses.
cap	The amount to cap total billing for this rule at (in the currency of the project).
cap_by_customerpo	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`. **Note:** You can only read or modify the `cap_by_customerpo` field if all the following conditions are met: ▪ 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: ▫ Portfolio Projects and Subordinate Projects. ▫ Single Billing Cap across Multiple Subprojects Within a Portfolio Project. The `cap_by_customerpo` attribute is listed in the Scripting Center SOAP Explorer if these features are enabled for your account.
cap_hours	The number of hours to cap the billing at for a time billing rule.
category_1id	The ID of the associated category_1. Mutually exclusive with project_task_id.
category_2id	The ID of the associated category_2. Mutually exclusive with project_task_id.
category_3id	The ID of the associated category_3. Mutually exclusive with project_task_id.
category_4id	The ID of the associated category_4. Mutually exclusive with project_task_id.
category_5id	The ID of the associated category_5. Mutually exclusive with project_task_id.

Field Name	Description
category_filter	CSV list of categories to limit the rule to.
category_when	When the category be applied: ■ N - Use the selected category if the time entry does not have a category. ■ A - Always use the selected category.
categoryid	The ID of the category to assign to the transaction if it doesn't have a category.
cost_center_id	The ID of the associated cost center.
created	Time the record was created.
currency	Currency for the money fields in the record.
customer_id	The ID of the associated customer.
customerpoid	The ID of the associated customerpo.
daily_cap_hours	The number of hours to cap the period billing per user at.
daily_cap_is_per_user	Is the daily cap on a per user basis.
daily_cap_period	Period for the cap: ■ D - day ■ W - week ■ M - month ■ Q - quarter ■ Y - year
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	If the period cap is hit move the remainder to the next rule.
description	The rule description.
end_date	End date of the rule.
end_milestone	The ID of the ending milestone (project_task).
exclude_archived_ts	Exclude time from archive timesheets in time billing rules.
exclude_non_billable	Exclude non-billable expenses.
exclude_non_billable_task	Exclude non-billable tasks.
exclude_non_reimbursable	Exclude non-reimbursable expenses.
extra_data	Holds extra data fields associated with the rule (see Extra data)
id	Unique ID. Automatically assigned by OpenAir.
item_filter	CSV list of items to limit the rule to.
job_code_filter	CSV list of filters to limit the rule to.
markup	The amount of markup in percent or monetary amount as designated by markup_type field.

Field Name	Description
markup_category	The ID of the category a markup on expense receipts should be assigned to.
markup_type	A field indicating the type of expense markup: ■ P - percentage of the cost. ■ S - specific amount.
name	Name of this project billing rule.
notes	Notes associated with this project billing rule.
percent	The percentage value for a fixed fee percent trigger.
percent_how	If the fixed fee is triggered by a percent complete, this holds how it is triggered: ■ 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	The position of the rule (0,1,2 etc.). Rules are evaluated in order and evaluation stops once a rule is satisfied.
product_filter	CSV list of products to limit the rule to.
project_task_filter	CSV list of tasks to limit the rule to.
project_task_id	The ID of the associated task. ■ `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	The ID of the associated project.
rate_cardid	The ID of the associated rate card if using rate cards.
rate_from	Where we get the rate from: ■ U - Users ■ R - Rate cards ■ C - Category
rate_multiplier	Optional multiplier to adjust the time billing rate by.
repeatid	The ID of the associated repeating event.
round_rules	Rules for rounding time.
slip_stageid	The ID of the slip stage to assign to the transaction.
start_date	Start date of the rule.
start_milestone	The ID of the starting milestone (project_task).
stop_if_capped	If a transaction is not billed because it exceeds the cap, should the billing stop for this transaction.
ticket_maximums	Holds data on ticket maximums per expense type.

Field Name	Description
timetype_filter	CSV list of timetypes to limit the rule to.
type	The type of the billing rule: ■ T - time billing rule. ■ E - expense billing rule. ■ F - fixed fee billing rule. ■ P - purchase billing rule.
updated	Time the record was last updated or modified.
user_filter	CSV list of users to limit the rule to.

This complex type supports the following methods: read, add, modify, and upsert.

Extra data

The `extra_data` field holds additional data associated with the project billing rule.

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:

- 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:

```
Projectbillingrule.extra_data = "$h->{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\'}}";
```

Note: Review the following guidelines:

- 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 `'%%OA_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.

oaProjectbillingtransaction

Use this complex type to specify project billing transactions. oaProjectbillingtransaction has the following children.

Field Name	Description
agreementid	ID of the associated agreement.
attributes	A collection of additional attributes for this complex type.
categoryid	The ID of the associated category.
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	The ID of the associated cost center.
created	Time the record was created.
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	The ID of the associated customer.
customerpoid	ID of the associated customerpo.
date	The date of the transaction.
description	Description associated with billing rule transaction.
fulfillmentid	The ID of the associated fulfillment record.
hour	The number of hours for a T type.

Field Name	Description
id	Unique ID. Automatically assigned by OpenAir.
itemid	The ID of the associated item.
job_codeid	The ID of the associated job code.
minute	The number of minutes for a T type.
notes	Notes associated with this project billing rule transaction.
payroll_typeid	The ID of the associated payroll type.
project_billing_ruleid	The ID of the associated project billing rule.
project_taskid	The ID of the associated project task.
projectid	The ID of the associated project.
quantity	The quantity for an E or P type.
rate	The hourly rate for a T type. Dated by the date field.
slip_stage_id	The ID of the slip stage.
slipid	The ID of slip that was created.
taskid	The ID of the associated task.
ticketid	The ID of the associated ticket.
timetypeid	The ID of the associated time type.
total	The total currency value. Dated by the date field.
type	The type of the transaction. Matches the type field in project_billing_rule.
um	The unit of measure for an E or P type.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the read method.

oaProjectBudgetGroup

The ProjectBudgetGroup datatype represents the complete project budget, accessible in the web application by going to Projects > Financials > Project Budget > Properties. 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. When deleting a budget, the same rules which the web application uses apply through the API. To delete a budget, you must have edit access, and approved or archived budgets cannot be deleted.

Note: Approvals are not supported for project budgets through the API.

Field Name	Description
approval_status	A one-character string indicating the approval status of the project budget group. Possible values:

Field Name	Description
	■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected ■ X - Archived
approved	The date the project budget group was approved
archived	The date the project budget group was archived
budget_by	Sets the "Budget by" option: ■ 1 — budget by project ■ 2 — budget by phase ■ 3 — budget by task
calculated_total	The total calculated from project budget transactions. Date set by the date" field (obsolete attribute)."
cf_opt	Optimistic contingency factor for this project budget.
cf_pes	Pesimistic contingency factor for this project budget.
created	Time the record was created.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer.
date	The date of the budget entry.
eac	Read only calculated field.
eac_expense	Read only calculated field.
eac_labout	Read only calculated field.
eac_purchase	Read only calculated field.
etc	Read only calculated field.
etc_expense	Read only calculated field.
etc_labor	Read only calculated field.
etc_purchase	Read only calculated field.
externalid	If the record was imported from an external system you store the unique external record ID here.
funding_total	Total calculated from project funding documents. Date set by the date" field."
id	Unique ID. Automatically assigned by OpenAir.
internal_total	Manually entered total. Date set by the date" field."
itd	Read only calculated field.
itd_expense	Read only calculated field.
itd_labor	Read only calculated field.

Field Name	Description
itd_purchase	Read only calculated field.
labor_subcategory	Labor subcategory: ▪ 0 - category ▪ 1 - job code
name	The name of the project budget group.
notes	Notes associated with this project budget.
parentid	The project budget group ID of this budget's immediate ancestor. If zero or null, this is a top-level project budget group.
profitability	The profitability of this project budget group.
projectid	The ID of the associated project.
setting	Miscellaneous settings are stored in this field as a serialized hash
submitted	The date the project budget group was submitted
total	The total value of the budget entry. Date set by the date" field."
total_calculated_billing	Billing total calculated from project budget transactions. Date set by the date" field."
total_calculated_cost	Cost total calculated from project budget transactions. Date set by the date" field."
total_expected_billing	Billing budget expected total. Date set by the date" field."
total_expected_cost	Cost total calculated from project budget transactions. Date set by the date" field."
total_from_funding	A 1/0 field indicating whether to use the total from project funding documents.
unassigned_task	A 1/0 field indicating whether it is possible to create budget entries not connected to any particular task.
updated	Time the record was last modified.
userid	The user ID of the budget owner.
version	Version of the project budget group.

This complex type supports the add, read, modify, and delete methods.

oaProjectBudgetRule

The ProjectBudgetRule datatype defines a line in the project budget grid. When adding a budget, OpenAir checks the validity of the project budget group ID, and returns error 945 if invalid (see Error Code Listing). The project and customer fields are populated 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:

- project_taskid
- category
- categoryid
- job_codeid

- itemid
- productid

Note: Approvals are not supported for project budgets through the API.

You cannot delete a rule if you do not have rights to delete the project budget.

Field Name	Description
category	The main category for this budget line: ■ 1 — labor ■ 2 — expense item ■ 3 — purchase order
categoryid	The ID of the associated category.
created	Time the record was created.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer.
date	The date of the budget rule.
end_date	End date of the period.
id	Unique ID. Automatically assigned by OpenAir.
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	The ID of the associated item.
job_codeid	The ID of the associated job code.
notes	Notes associated with this project budget line.
period	The period of the total: ■ 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	The ID of the associated product.
profitability	The profitability of this project budget rule.
project_budget_groupid	The ID of the associated project_budget_group.
project_taskid	The ID of the associated project task.
projectid	The ID of the associated project.
quantity	The quantity for this project budget rule.
quantity_best	The best-case quantity estimate for this project budget rule.
quantity_most_likely	The most likely quantity estimate for this project budget rule.

Field Name	Description
quantity_worst	The worst-case quantity estimate for this project budget rule.
rate	The rate of this project budget rule.
start_date	Start date of the period.
total	The total for this project budget rule. Date set by the date" attribute."
total_best	The best-case estimate for this project budget rule. Date set by the date" attribute."
total_most_likely	The most likely estimate for this project budget rule. Date set by the date" attribute."
total_worst	The worst-case estimate for this project budget rule. Date set by the date" attribute."
updated	Time the record was last modified.

This complex type supports the add, read, modify, and delete methods.

oaProjectBudgetTransaction

The ProjectBudgetTransaction datatype defines a transaction in one project budget grid line. When adding a new project budget transaction, OpenAir checks the validity of the project budget rule, and returns error 946 if invalid (see Error Code Listing). The following fields are populated from the project budget rule:

- project_budget_groupid
- customerid
- projectid
- project_taskid
- categoryid
- job_codeid
- itemid
- productid

Note: Approvals are not supported for project budgets through the API.

When modifying an existing transaction, you cannot change the project_budget_ruleid or any of the fields populated by the add method. You cannot delete a transaction if you don't have rights to delete the project budget group.

Field Name	Description
category	The main category for this budget transaction: - 1 — labor - 2 — expense item - 3 — purchase order
categoryid	The ID of the associated category.
created	Time the record was created.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer.

Field Name	Description
date	The date of the project budget transaction.
id	Unique ID. Automatically assigned by OpenAir.
itemid	The ID of the associated item.
job_codeid	The ID of the associated job code.
productid	The ID of the associated product.
project_budget_groupid	The ID of the associated project budget group.
project_budget_ruleid	The ID of the associated project budget rule.
project_taskid	The ID of the associated project task.
projectid	The ID of the associated project.
quantity	Quantity for this project budget transaction.
quantity_best	The best-case quantity estimate for this project budget transaction.
quantity_most_likely	The most likely quantity estimate for this project budget transaction.
quantity_worst	The worst-case quantity estimate for this project budget transaction.
total	The total for this budget transaction. Date set by the date" attribute."
total_best	The best-case estimate for this project budget transaction. Date set by the date" attribute."
total_most_likely	The most likely estimate for this project budget transaction. Date set by the date" attribute."
total_worst	The worst-case estimate for this project budget transaction. Dated by the date field.
updated	Time the record was last updated or modified.

This complex type supports the add, read, modify, and delete methods.

oaProjectgroup

Use this complex type to document users who are assigned to a project task as a group. oaProjectgroup has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is active.
assigned_users	The users assigned to this project group. Can be a comma delimited list.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	The name for the project group.
notes	Notes associated with the project group.

Field Name	Description
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaProjectlocation

Use this complex type to specify project location information. oaProjectlocation has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	The name for the project location.
notes	Notes associated with the project location.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaProjectstage

Use this complex type to specify project stage information. oaProjectstage has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
enable_analysis	Is financial analysis enabled at this stage.
enable_billing	Is the project billing tab enabled at this stage.
enable_phase_and_task	Are phases and tasks enabled at this stage.
enable_pricing	Is project pricing enabled at this stage. Off by default.
enable_project_assignments	Are project level assignments enabled at this stage.
enable_recognition	Is the recognition tab enabled at this stage.
enable_team	A 1/0 field indicating if this should be the default stage for new projects.
enable_utilization	Is the utilization enabled at this stage.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the project stage.
notes	Notes associated with the project stage.

Field Name	Description
picklist_label	Label as shown on form picklist.
position	The position of the stage.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaProjecttask

Use this complex type to specify information about the individual tasks or work packages that comprise a project. oaProjecttask has the following children.

Field Name	Description
all_can_assign	Is everyone able to assign time/expenses to this task.
assign_user_names	A comma separated list of user nicknames to assign this task to. (project_task_assign object types can also be used.)
attributes	A collection of additional attributes for this complex type.
calculated_finishes	Calculated finish date.
calculated_starts	Calculated start date of the project task.
classification	Calculated, read-only classification of the project task with the following values: ■ 'P' for Phase ■ 'T' for Task ■ 'M' for Milestone
closed	A 1/0 field indicating if this is closed task. Additional time can not be booked against closed task.
cost_centerid	The ID of the associated cost center
created	Time the record was created.
currency	Currency for the money fields in the record. This should be the same as the project currency.
customer_name	The name of the associated customer.
customerid	The ID of the associated customer.
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_1	A feature, if enabled, would assign this default_category_1 to the category_1 for many transactions that have a category_1_id and project_task_id by searching the project_task and phase work breakdown structure for the first default_category_1 defined.
default_category_2	A feature, if enabled, would assign this default_category_2 to the category_2 for many transactions that have a category_2_id and project_task_id by searching the project_task and phase work breakdown structure for the first default_category_2 defined

Field Name	Description
default_category_3	A feature, if enabled, would assign this default_category_3 to the category_3 for many transactions that have a category_3_id and project_task_id by searching the project_task and phase work breakdown structure for the first default_category_3 defined.
default_category_4	A feature, if enabled, would assign this default_category_4 to the category_4 for many transactions that have a category_4_id and project_task_id by searching the project_task and phase work breakdown structure for the first default_category_4 defined.
default_category_5	A feature, if enabled, would assign this default_category_5to the category_5 for many transactions that have a category_5_idand project_task_id by searching the project_task and phase work breakdownstructure for the first default_category_5 defined.
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 happened so the estimate is the same as the plan.
externalid	If the record was imported from an external system you store the unique external record ID here.
fnlt_date	The finish no later than date of the task. The task must be finished by this date.
id	Unique ID. Automatically assigned by OpenAir.
id_number	User-defined task ID.
is_a_phase	A 1/0 field indicating if any other project_tasks have us as a parent.
manual_task_budget	If set to 1 then the task budget is manually entered rather than calculated by OpenAir.
name	Short description of this task.
non_billable	If set to 1, this is not billable. This is only applicable for project billing rules.
notes	Notes associated with the project task.
parentid	The task ID of our immediate ancestor. If zero or null, this is a project-level (top-level) task or phase.
percent_complete	This field is an estimate of the percentage of planned time which has been completed. It has no relation to the actual time spent on a task. (A 5-hour task could consume 50 hours of work but still be only 25% complete.)
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	Comma delimited list of task IDs which must complete before this task can start.
predecessors_lag	Comma delimited list for task ID:days of lag time for predecessors. Only populated if there is a lag time.
predecessors_type	Comma delimited list of task ID:relationship type for predecessors. Only populated if the relationship type is not finish-to-start.
priority	The priority of the task (1 - 9).
project_name	The name of the associated project.
projectid	The ID of the associated project.
projecttask_typeid	The ID of the associated project task type. Not for phases.

Field Name	Description
seq	The sequence number of this task.
starts	Optional scheduled starting date of this task. Overrides computed date Start_date.
task_budget_cost	If task budgeting is enabled, this is the total cost of the task.
task_budget_revenue	If task budgeting is enabled this is the total projected billing for the task.
timetype_filter	A timetype filter. This will hold a list of the timetypes that are allowed to book time to this task.
updated	Time the record was last updated or modified.
use_project_assignment	Flag set to 1 if they are using the project level user assignment.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Using the limit attribute with oaProjecttask

When reading project tasks, the limit attribute is applied to **projects** and not project tasks if the following four conditions are met:

- method=”all”
- no filters are used in the request
- deleted records are not requested
- the client_type is “RW Project”

For example, if you set the limit attribute to “0,1000”, read will find **all** tasks for the first 1,000 projects, and may return more than 1,000 tasks. In other words, the limit attribute limits the number of projects returned, and does not limit the number of project tasks returned.

In all other cases, the limit attribute is applied to **project tasks**. For example, if the method is set to “equal to” or “not equal to”, or if a filter is used in the request, the limit attribute applies to project tasks.

oaProjecttask_type

Use this complex type to specify information about a project task type. oaProjecttask_type has the following children.

Field Name	Description
active	A 1/0 field specifying if the type is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the projecttask_type.
notes	Notes associated with the project task type.
picklist_label	Label as shown on form picklist.
suppress_notification	Suppress task notifications for this project task type.

Field Name	Description
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaProjecttaskassign

Use this complex type to specify the list of users assigned to each task. oaProjecttaskassign has the following children.

Field Name	Description
allocation	The percentage of time the associated user is allocated to this task.
attributes	A collection of additional attributes for this complex type.
booking_id	The ID of the associated booking.
created	Time the record was created.
externalid	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
job_codeid	The ID of the associated job code.
pending_booking_id	The ID of the associated pending booking.
planned_hours	The hours for this user if the planned hours at the user level feature is enabled.
project_assignment_profile_id	The ID of the associated project assignment profile.
project_groupid	The ID of the project group if the user was assigned as part of a project group.
projecttaskid	ID of the project task to which this user is assigned
rule_rate_override	Hourly billing rate for the user assigned to the task. This is effective only if the internal switch “Enable billing rule rate override on task assignments” is enabled on your account. Negative values are not allowed.
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 multi-currency is enabled on your account).
updated	Time the record was last updated or modified.
userid	The ID of the user assigned to this task.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaProjecttaskEstimate

This complex type holds the user estimates for time remaining against project tasks. This is used to drive percent complete calculations against the project. Required fields for this object are hours, user_id, and project_task_id. When adding or modifying an oaProjecttaskEstimate complex type:

- A user must be assigned to the task
- That user's user_id must exist in OpenAir
- The timesheet_id must exist in OpenAir
- There must be a time entry for the project_task_id if sent with timesheet_id
- The same estimate must not already exist

Note: You cannot modify an approved or archived timesheet's ProjecttaskEstimate unless you have the Allow Editing of Approved and Archived Timesheets through API feature enabled. In addition, the project task recalculation for hours remaining depends on the "Disable job recalc triggering from API" setting.

oaProjecttaskEstimate has the following children:

Field Name	Description
changed_by	ID of the user who changed the estimate. If this does not have an ID, then the estimate was automatically generated by OpenAir.
created	Time the record was created
date_changed	The date and time the estimate was last changed
hours	The number of hours estimated to be remaining
id	Unique ID. Automatically assigned by OpenAir.
project_taskid	The ID of the associated project_task
timesheet_id	The ID of the associated timesheet if this was updated from the timesheet
updated	Time the record was last updated or modified
user_id	The ID of the user who is assigned to the task

This complex type supports the following methods: read, add, modify, and delete.

oaProposal

Use this complex type to specify proposal information. oaProposal has the following children.

Field Name	Description
access_log	The mailing and access history of the proposal.
approved	The date and time the proposal was approved.
approved_by	The ID of the user who approved this proposal.
attachments	If non-zero, the attachment record associated with this proposal. attachment_id
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
created_by	The ID of the user who created this proposal.
customerid	The ID of the associated customer.

Field Name	Description
description	The description of this proposal.
expires	The date the proposal is valid until.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of this proposal.
notes	Notes associated with this proposal.
number	The proposal number.
projectid	The ID of the associated project.
responded	The date and time the customer accepted or refused.
response	Customer response notes.
sent	The date and time the proposal was delivered to the customer.
status	The status of the proposal: ■ D - Draft ■ M - Submitted ■ P - Approved ■ Q - Rejected ■ S - Sent ■ V - Viewed ■ A - Accepted ■ R - Refused
submitted	The date and time the proposal was submitted for approval.
total	The total amount. Dated by the currency_date field.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.
viewed	The date and time the customer first viewed the proposal.

This complex type supports the read method.

oaProposalblock

Use this complex type to specify proposal blocks, the blocks of text that a proposal is composed of. A block can be free form text or it can be associated with a template. If associated with a template, it is updated when the template is updated. oaProposalblock has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
categoryid	The ID of the associated category.
content	The content of the template.

Field Name	Description
cost	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	Time the record was created.
description	The description of this proposal.
hour	The number of hours for a T block.
id	Unique ID. Automatically assigned by OpenAir.
itemid	The ID of the associated item.
minute	The number of minutes for a T block.
name	The name of the this proposal block.
proposalid	The ID of the associated proposal.
quantity	The quantity for an E block or an O block.
rate	The hourly rate for a T block. Dated by the currency_date field.
seq	The sequence number of the block.
slipid	The ID of the associated slip if this block was billed to TB.
templateid	The ID of the associated template.
total	The total value of the block. Dated by the currency_date field.
type	The type of the slip: ■ 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	The unit of measure for an E block or the rate description for an O block.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaProxy

The Proxy datatype holds data about user proxies.

Field Name	Description
audit	Audit trail of changes.
created	Time the record was created
deleted	A 1/0 field indicating if the record was deleted

Field Name	Description
expiration	The date the proxy expires
id	Unique ID. Automatically assigned by OpenAir.
own	A 1/0 field indicating if the proxy was created by proxy_id using 'create own proxy' feature.
proxy_id	The user ID for whom you are proxying
role_id	Role to use while proxying for this user
updated	Time the record was last updated or modified
user_id	ID of the user who is doing the proxying

This complex type supports the following methods: read, add, modify, and delete.

oaPurchase_item

Use this complex type to specify purchase item information, a single entry in a purchase order. oaPurchase_item has the following children.

Field Name	Description
acct_date	The accounting period date of the purchase item.
allow_vendor_substitution	A 1/0 field indicating whether the vendor may be substituted.
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	If non-zero, the attachment record associated with this purchaseitem.
attributes	A collection of additional attributes for this complex type.
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	Time the record was created.
currency	Currency for the money fields in the record.
cusomerid	The ID of the associated customer.
date	The date of the purchase item. The same as the purchase orderdate.
date_fulfilled	The date on which all of the quantity was fulfilled.
id	Unique ID. Automatically assigned by OpenAir.
manufacturer_part	The manufacturer's part number, SKU or other unique identification for this product.
manufacturerid	The ID of the associated manufacturer.
name	The purchase name.
non_po	A 1/0 field indicating that this purchase item was created without a purchase order.

Field Name	Description
notes	Notes associated with this purchase order block.
order_reference_number	Unique reference number within purchase order.
productid	The ID of the associated product.
project_taskid	The ID of the associated project task.
projectid	The ID of the associated project.
purchaseorderid	The ID of the associated purchase order.
purchaserequestid	The ID of the associated purchase request.
purchaserid	The ID of the purchaser or purchasing agent. This is always the same as the purchase order creator (purchaser_id).
quantity	The quantity of product_id for this purchase.
quantity_fulfilled	The quantity that has been fulfilled.
quantity_payable	The quantity that is payable.
request_itemid	The ID of the associated request item.
tax_location_name	The name of the tax location.
total	The total value of the purchase (in the currency of the purchase order). Dated by the date field.
total_with_tax	The total value of the purchase (in the currency of the purchase order)including tax. Dated by the date field.
um	The unit of measure for the product, i.e., EA.
updated	Time the record was last updated or modified.
userid	The ID of the requester.
vendor_quote_number	The vendor's quote number.
vendor_sku	The vendor's sku for this product.
vendorid	The ID of the associated vendor

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: There are several limitations regarding purchase items: Only short-order purchase items can be added to OpenAir and only project/customer combinations can be updated on a non short-order purchase item (switch enabled). See details as follows:

- For the capability to add short-order purchase items in OpenAir, go to Administration > Application Settings > Purchases > Other Settings. Scroll down and check the **Enable the ability to create non-PO purchase items** box. Non-PO purchase items are purchase items for purchases made without an OpenAir PO.
- For the capability to modify non short-order purchase items, submit a support ticket to enable the following setting: **API can modify purchase items project association even when associated with a PO**. Associated request items will also be updated. See Creating a Support Case for instructions on how to create a support ticket.

oaPurchaseorder

Use this complex type to specify information about a purchase order. oaPurchaseorder has the following children.

Field Name	Description
accounts_payableid	The accounts payable location for this purchase order.
approval_status	A one-character string indicating the approval status of the purchase request. Possible values: ■ O - Open ■ P- Pending approval ■ A - Approved ■ R - Rejected
attachmentid	If non-zero, the attachment record associated with this purchase order.
attributes	A collection of additional attributes for this complex type.
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	The carrier to be used for shipping. Ship Via.
created	Time the record was created.
currency	The currency this purchase order is in.
date	The date of the purchase order.
date_approved	The date the purchase order was approved.
date_expected	The date the materials are expected if known.
date_fulfilled	The date on which all of the total quantity was fulfilled.
date_order_placed	The date the purchase order was placed with the vendor.
date_required	The date the purchase items on this purchase order are required.
date_shipped	The date the materials were shipped if known.
date_submitted	The date the purchase order was submitted.
description	The description or purpose for the purchase order.
id	Unique ID. Automatically assigned by OpenAir.
locationid	The F.O.B. location_id (DEPRECATED).
name	The name of the purchase order (Prefix + number).
notes	Notes to print on the purchase order.
number	The purchase order number that increments by 1.
prefix	A static alphanumeric purchase order number prefix.
purchase_items_fulfilled	The total number of fulfilled purchase items in the purchase order.

Field Name	Description
quantity_fulfilled	The quantity fulfilled on all the purchase items in this purchase order.
receivingid	The receiving location for this purchase order.
ship_complete_only	A 1/0 field indicating that full order must ship together.
shipping_cost	The cost of shipping, if known. Dated by the date field.
shipping_termsid	The ID of the associated shipping payment terms, indicating how the shipping costs will be charged.
terms	Payment terms for this purchase order.
total	The purchase order total cost. Dated by the date field.
total_purchase_items	The total number of purchase items in the purchase order.
total_quantity	The total quantity of all the purchase items in this purchase order.
updated	Time the record was last updated or modified.
userid	The ID of the user creating the purchase order. The purchasing agent.
vendorid	The ID of the associated vendor that the purchase order is for.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaPurchaser

Use this complex type to specify information about a user who creates purchase orders. oaPurchaser has the following children.

Field Name	Description
accounts_payableid	The default accounts payable location for this purchaser.
active	A 1/0 field indicating where this is designated as an active receiving location.
attributes	A collection of additional attributes for this complex type.
carrierid	The default carrier to be used for shipping. Ship Via.
created	Time the record was created.
exported	Date and time the record was marked as exported.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the purchaser.
notes	Notes associated with the purchaser.
receivingid	The default receiving location for this purchaser.
ship_complete_only	The default for the 1/0 field indicating that full order must ship together."
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the read method.

oaPurchaserequest

Use this complex type to specify purchase request information. oaPurchaserequest has the following children.

Field Name	Description
approval_status	A one-character string indicating the approval status of the purchase request. Possible values: ■ O - Open ■ P - Pending approval ■ A - Approved ■ R - Rejected
attachmentid	If non-zero, the attachment record associated with this purchase request.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
currency	The currency of the total field.
customerid	The ID of the associated customer that the material on this purchase request is for.
date	The date of the purchase request.
date_approved	The date the purchaserequest was approved.
date_fulfilled	The date on which all of the total quantity was fulfilled.
date_required	The date the material on this purchase request is required.
date_submitted	The date the purchaserequest was submitted.
description	The description or purpose for the purchaserequest.
exported	Date and time the record was marked as exported.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the purchaserequest (Prefix + number).
notes	Notes associated with the purchase request.
number	The purchase request number that increments by 1.
ordered_request_items	The total number of request items on the purchase request which are part of a purchase order.
prefix	A static alphanumeric purchase request number prefix.
projectid	The ID of the associated project that the material on this purchase request is for.
quantity_fulfilled	The quantity fulfilled on all the request items in this purchase request.
request_items_fulfilled	The total number of fulfilled request items in the purchase request.
total	The purchase request total cost. Dated by the date field.

Field Name	Description
total_quantity	The total quantity of all the request items in this purchase request
total_request_items	The total number of request items in the purchase request.
updated	Time the record was last updated or modified.
userid	The ID of the associated user creating the purchase request. The requester.

This complex type supports the read method.

oaRatecard

Use this complex type to map job codes to hourly rates. oaRatecard has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is an active rate card.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the rate card.
notes	Notes associated with the rate card.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaRateCardItem

Use this complex type to specify rate card item information. oaRateCardItem has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
currency	Currency for the money fields in the record.
current	A 1/0 field indicating if the record is the current rate.
end	End date of the rate for historical records.
id	Unique ID. Automatically assigned by OpenAir.
job_code_id	The ID of the associated job code.
rate	The hourly billing rate.
rate_card_id	The ID of the rate card that it is associated with.

Field Name	Description
start	Start date of the rate for historical records.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaReimbursement

Use this complex type to specify reimbursement information. oaReimbursement has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
audit	Audit trail changes.
created	Time the record was created.
currency	Currency for the money fields in the record.
date	The date of the reimbursement.
envelope_number	The number of the associated envelope the reimbursement is applied to.
envelopeid	The associated envelope the reimbursement is applied to.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
notes	Notes associated with the reimbursement.
total	The reimbursement total. Dated by the date field.
updated	Time the record was last updated or modified.
userid	The user associated with the envelope the reimbursement is applied to.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaRepeat

Use this complex type to specify repeating event information. oaRepeat has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
end	End date of the event.
every	The spacing between each repeating event.

Field Name	Description
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	The repeating interval of the event: D – daily, W – weekly, M – monthly, Y – yearly.
how_end	How does this event end: D – date or O - occurrence
id	Unique ID. Automatically assigned by OpenAir.
occur_number	Number of occurrences.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaReport

Use this complex type to hold saved report definitions and settings. oaReport has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
date_created	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.
email_report	A 1/0 field. 1 = report executes and sends an email with a pdf attachment to the session user.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the report.
relatedid	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	A 1/0 field indicating that this report can be requested via thin clients.
type	The type of report: S = saved reports and T = sTandard reports.
updated	Time the record was last updated or modified.
userid	The ID of the user who created the report. This is 0 for standard reports.

This complex type supports the read method.

oaRequest_item

Use this complex type to specify a request item, a single entry in a purchase request. oaRequest_item has the following children.

Field Name	Description
allow_vendor_substitution	A 1/0 field indicating whether the vendor may be substituted.

Field Name	Description
attachmentid	If non-zero, the attachment record associated with this request_item.
attributes	A collection of additional attributes for this complex type.
cost	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	Time the record was created.
currency	The currency used for this request item.
customerid	The ID of the associated customer. This is always the same as the purchase request customer_id.
date	The date of the request_item. The same as the purchaserequest date.
date_fulfilled	The date on which all of the quantity was fulfilled.
exported	Date and time the record was marked as exported.
id	Unique ID. Automatically assigned by OpenAir.
manufacturer_part	The manufacturer's part number, SKU or other unique ID for this product.
manufacturerid	The ID of the associated manufacturer.
name	The request item name.
notes	Notes associated with this request item.
productid	The ID of the associated product.
projectid	The ID of the associated project. This is always the same as the purchase request project_id.
purchase_itemid	The ID of the associated purchase_item.
purchaseorderid	The ID of the associated purchase order.
purchaserequestid	The ID of the associated purchaserequest.
quantity	The quantity of product_id for this request item.
quantity_fulfilled	The quantity that has been fulfilled.
request_reference_number	Unique reference number within purchase request.
total	The total value of the request item. Dated by the date field.
um	The unit of measure for the product, i.e., EA.
updated	Time the record was last updated or modified.
userid	The ID of the requester. This is always the same as the purchaserequest requester (user_id).
vendor_quote_number	The vendor's quote number.
vendor_sku	The vendor's sku for this product.
vendorid	The ID of the associated vendor.

This complex type supports the read and delete methods.

oaResourceAttachment

Use the ResourceAttachment datatype to specify a user's CV attachment in their Consolidated Resource Profile. oaResourceAttachment has the following children.

Field Name	Description
attachment_id	The attachment record associated with this document.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
latest_attachment_id	ID of the latest attachment from the attachment table.
type	The document type. Only "CV" is supported.
updated	Time the record was last updated or modified.
userid	The ID of the user to whom this attachment belongs.

This complex type supports the add, read, modify, and delete methods.

oaResourceprofile

Use this complex type to specify items the make up a resource profile. oaResourceprofile has the following children.

Field Name	Description
attributeid	The ID of the optional resourceprofile attribute.
attributes	A collection of additional attributes for this complex type.
comment	Additional comment describing this resourceprofile.
created	Time the record was created.
externalid	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The resourceprofile name. Stub.
resourceprofile_typeid	The ID of the resourceprofile_type.
type	The resourceprofile type. The entity on which this resourceprofile is based. ■ Skill ■ Education ■ Location ■ Jobrole ■ Industry ■ Customprofile_1..20
updated	Time the record was last updated or modified.
userid	The ID of the user for which this resourceprofile describes.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaResourceprofile_type

Use this complex type to specify information about a resource profile type. oaResourceprofile_type has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is active.
attribute_set_id	The ID of the associated attribute set.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
externalid	If record was imported from an external system, store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The resourceprofile_type name. This shows up on all the resourceprofile_type popup windows in the application.
related_table	The name of the table related with this table.
relatedid	The ID of the related item in the related table.
type	The resourceprofile type. The entity on which this resourceprofile is based. ■ Skill ■ Education ■ Location ■ Jobrole ■ Industry ■ Customprofile_1..20
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaResourceRequest

Use this complex type to specify resource request information. oaResourceRequest has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
booking_type_id	The booking type of bookings created for this resource request.
created	Time the record was created.
customerid	The ID of the associated customer
date_end	The ending date of the resource request.

Field Name	Description
date_finalized	The date the resource request was finialzed and marked ready for booking.
date_start	The starting date of the resource request.
date_start_expected	The expected starting date of the resource request.
external_id	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the resource request.
notes	Notes field
number	The resource request tracking number.
ownerid	The ID of the associated user creating the resource request.
percent_fulfilled	Percent fulfilled for the resource request.
projectid	The ID of the associated project.
status	The status of the resource request: ■ 'O' - Open ■ 'P' - Partial ■ 'S' - Complete ■ 'C' - Canceled
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaResourceRequestQueue

Use this complex type to specify request queue information. oaResourceRequestQueue has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
booking_type_id	The booking type of bookings created for this resource request queue.
created	Time the record was created.
customerid	The ID of the associated customer.
date_end	The ending date of the resource request queue.
date_start	The starting date of the resource request queue.
external_id	If the record was imported from an external system you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the resource request queue.

Field Name	Description
notes	Notes field.
percent_fulfilled	Percent fulfilled for the resource request queue.
projectid	The ID of the associated project.
resource_request_id	The ID of the associated resource request.
resourcesearch_id	The ID of the associated base resource search. **Note:** If you don't specify a resourcesearch_id the API will automatically create an empty oaResourcesearch and populate the ID here.
slots	The number of slots available in this queue.
status	The status of the resource request queue: ■ 'O' - Open ■ 'P' - Partial ■ 'S' - Complete ■ 'C' - Canceled
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaResourcesearch

Use this complex type to specify resource search information. oaResourcesearch has the following children. See also Resource Search Virtual Fields.

Field Name	Description
as_percentage	A "1/0" field indicating which of the fields... hours or percentage is to be used in the search ■ If 1 then use percentage. ■ If 0 then use hours.
attributes	A collection of additional attributes for this complex type.
availability_search	A 1/0 field indicating whether to search by availability.
consecutive_availability	A 1/0 field indicating no intervening bookings.
created	Time the record was created.
customprofile_1 ... customprofile_35	Comma delimited list of customprofile_nn that make up this search Each element is the ID followed by an optional attribute ID, separated by a colon (:).
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	The end date for availability.
excluding	See Resource Search Virtual Fields.

Field Name	Description
external_id	If the record was imported from an external system you store the unique external record ID here.
hours	The number of hours of availability required over this range.
id	Unique ID. Automatically assigned by OpenAir.
include_generic_resources	A 1/0 field. Include generic resources in search?
include_inactive_resources	A 1/0 field. Include inactive resources in search?
include_regular_resources	A 1/0 field. Include regular resources in search?
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	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	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	The resourcesearch name.
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	The ID of the associated resource request queue.
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	The start date for availability.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Resource Search Virtual Fields

oaResourcesearch uses three virtual fields: required, excluding, and preferred. 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 resources.

For example, if you had the following data setup:

- resourceprofile_type.id = 10 for “Linux skill” and 12 for “Master’s degree”.
- attribute.id = 1 for “Beginner”, 2 for “Intermediate”, and 3 for “Expert”.

> **Note:** attribute.id = 0 means “Any”
>
> If Attribute set is not defined for appropriate resource_profile then set attribute.id = 0.

With the data described above set, the following XML:

```
<preferred>10:1,10:2,12:0</preferred>
```

would search for resources with beginner Linux skill, intermediate Linux skill, and a Master’s degree.

oaRevenue_recognition_rule

Use this complex type to specify revenue recognition rules. oaRevenue_recognition_rule has the following children.

Field Name	Description
accounting_period_id	The ID of the associated accounting period.
acct_code	Optional accounting system code for integration with external accounting systems.
acct_date	The accounting period date to assign to the transaction.
acct_date_how	The accounting period date of the transaction is determined by: ■ N - none, clear the value ■ E - the entity (no change) ■ C - container of the entity if available (i.e., timesheet, envelope) ■ M - set by the specified accounting date ■ P - set by the specified accounting period
active	A 1/0 field indicating whether this is an active rule.
agreementid	ID of the associated agreement.
amount	The amount. If we have multiple amounts, the values are held in the revenue_recognition_rule_amount table.
asb_exclude_slip_type	CSV list of the slip types to exclude from the as billed rule.
asb_which_slips	Which slips should be considered for the as billed rule: ■ A - all slips ■ I - slips on invoices ■ P - slips on approved invoices
assigned_user	The user to assign to fixed fee recognition.
attributes	A collection of additional attributes for this complex type.
break_by_user	Break out the transactions by user. Currently only implemented for the incurred rules.
category_1id	The ID of the associated category_1. Mutually exclusive with project_task_id.
category_2id	The ID of the associated category_2. Mutually exclusive with project_task_id.
category_3id	The ID of the associated category_3. Mutually exclusive with project_task_id.
category_4id	The ID of the associated category_4. Mutually exclusive with project_task_id
category_5id	The ID of the associated category_5. Mutually exclusive with project_task_id.
categoryid	The ID of the associated category.
cost_center_id	The ID of the associated cost center.
created	Time the record was created.

Field Name	Description
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer.
customerpo_id	ID of the associated customerpo.
end_date	End date of the rule.
end_milestone	ID of the ending milestone (project_task).
expense_how	How expenses should be recognized: ■ M - mark up/down on billed expenses. ■ B - billed expenses. ■ I - incurred expenses.
extra_data	Holds extra data fields associated with the rule.
id	Unique ID. Automatically assigned by OpenAir.
item_filter	CSV list of items to limit the rule to.
marked_as_ready	Trigger recognition when a task (ID in phase) is marked as ready to recognize.
name	Name of the rule.
notes	Notes associated with this revenue recognition rule.
percent	The percentage value for a fixed fee percent trigger.
percent_complete	The calculated percent complete value if a type P transaction.
percent_how	How percent complete should be calculated: ■ 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	If the fixed fee is triggered by a percent complete, this holds how it is triggered: ■ 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	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	CSV list of products to limit the rule to.
project_billing_rule_filter	CSV list of project billing rule id's to limit a type T rule to.
project_billing_ruleid	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	CSV list of tasks to limit the rule to.
projectid	The ID of the associated project.

Field Name	Description
purchase_how	How purchases should be recognized: ▪ M - mark up/down on billed purchases. ▪ B - billed purchases.
recognition_type	What we are recognizing: ▪ R - revenue ▪ C - cost ▪ O - other
repeatid	The ID of the associated repeating event.
slip_stage_filter	CSV list of slip_stage ID to limit a type A rule to.
start_date	Start date of the rule.
start_milestone	ID of the starting milestone (project_task).
timetype_filter	CSV list of timetypes to limit the rule to.
type	The type of the rule: ▪ 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	Time the record was last updated or modified.
user_filter	CSV list of users to limit the rule to.

This complex type supports the following methods: read, add, modify, and upsert.

oaRevenue_recognition_rule_amount

Use this complex type to specify multiple amounts for a recognition rule. oaRevenue_recognition_rule_amount has the following children.

Field Name	Description
acct_code	Optional accounting system code for integration with external accounting systems.
agreement_id	The ID of the associated agreement.
amount	The amount.
attributes	A collection of additional attributes for this complex type.
category_1id	The ID of the associated category_1. Mutually exclusive with project_task_id.
category_2id	The ID of the associated category_2. Mutually exclusive with project_task_id.

Field Name	Description
category_3id	The ID of the associated category_3. Mutually exclusive with project_task_id.
category_4id	The ID of the associated category_4. Mutually exclusive with project_task_id.
category_5id	The ID of the associated category_5. Mutually exclusive with project_task_id.
category_id	The ID of the associated category.
cost_center_id	The ID of the associated category.
created	Time the record was created.
currency	Currency for the money fields in the record.
customerpo_id	The ID of the associated customerpo.
id	Unique ID. Automatically assigned by OpenAir.
recognition_type	Recognition type: ■ R - revenue ■ C - cost ■ O - other
revenue_recognition_rule_id	The ID of the associated rule.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaRevenue_recognition_transaction

Use this complex type to specify revenue recognition transactions. This is a record of a single transaction created when revenue recognition was run for a particular project. oaRevenue_recognition_transaction has the following children.

Field Name	Description
acct_code	Optional accounting system code for integration with external accounting systems.
acct_date	The accounting period date of the transaction.
agreementid	The ID of the associated agreement.
attributes	A collection of additional attributes for this complex type.
category_1id	The ID of the associated category_1.
category_2id	The ID of the associated category_2.
category_3id	The ID of the associated category_3.
category_4id	The ID of the associated category_4.
category_5id	The ID of the associated category_5.
categoryid	The ID of the associated category.
cost_center_id	The ID of the associated cost center.

Field Name	Description
created	Time the record was created.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer.
customerpo_id	The ID of the associated customerpo.
date	The date of the transaction.
decimal_hours	The number of decimal hours.
hour	The number of hours for a T type.
id	Unique ID. Automatically assigned by OpenAir.
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	The ID of the associated job code.
minute	The number of minutes for a T type.
notes	Notes associated with this revenue recognition transaction.
offsetsid	The ID of the revenue_recognition_transaction which this revenue_recognition_transaction offsets.
originatingid	The ID of the originating revenue_recognition_transaction forthis revenue_recognition_transaction.
percent_complete	The calculated percent complete value if it is a type P transaction.
portfolio_projectid	The ID of the associated portfolio project
project_billing_ruleid	The ID of the associated project billing rule.
project_taskid	The ID of the associated project task.
projectid	The ID of the associated project.
rate	The hourly rate for a T type. Dated by the date field.
recognition_type	Recognition type: ▪ R - revenue ▪ C - cost ▪ O - other
revenue_containerid	The ID of the associated revenue_container once posted.
revenue_recognition_ruleid	The ID of the associated rule.
revenue_stageid	The ID of the associated revenue stage.
slipid	The ID of the associated slip.
taskid	The ID of the associated task.

Field Name	Description
ticketid	The ID of the associated ticket.
total	The amount of the transaction. Dated by the date field.
type	The type of the transaction. Matches the type field in revenue_recognition_rule.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, and upsert.

oaRevenueContainer

Use this complex type to specify information about the revenue_container header table. oaRevenueContainer has the following children.

Field Name	Description
acct_date	The accounting period date of the revenue_container.
approval_status	A one-character string indicating the approval status of the invoice. Only used if invoice approvals are used. Possible values: ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected
attributes	A collection of additional attributes for this complex type.
balancing_type	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: ■ A - Agreement ■ C - CustomerPO ■ P - Project ■ X - Agreement and CustomerPO
created	Time the record was created.
currency	The currency of this revenue_container.
customerid	The ID of the associated customer.
date	The date of the revenue_container.
date_approved	The date the invoice was approved.
date_submitted	The date the invoice was submitted.
exported	Date and time the record was marked as exported.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.

Field Name	Description
name	The name of the revenue_container (Prefix + number).
notes	Notes to print on the revenue_container.
number	The revenue_container number that increments by 1.
prefix	A static alphanumeric revenue_container number prefix.
projectid	The ID of the associated project.
total_accrued	The revenue_container accrued total. Dated by the date field.
total_deferred	The revenue_container deferred total. Dated by the date field.
total_invoiced	The revenue_container invoice total. Dated by the date field.
total_posted	The revenue_container posted total. Dated by the date field.
total_recognized	The revenue_container recognized total. Dated by the date field.
updated	Time the record was last updated or modified.

This complex type supports the read and modify methods.

oaRevenueProjection

Use this complex type to access the slips created from a projected revenue run.

Field Name	Description
acct_date	The accounting period date of the slip.
agreement_id	The ID of the associated agreement.
attributes	A collection of additional attributes for this complex type.
booking_type_id	ID of the booking type if this was generated from bookings.
category_1_id	The ID of the associated category_1.
category_2_id	The ID of the associated category_2.
category_3_id	The ID of the associated category_3.
category_4_id	The ID of the associated category_4.
category_5_id	The ID of the associated category_5.
category_id	The ID of the associated category. If this is set, the slip is based on this category.
city	The slip city or location.
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_center_id	The ID of the associated cost center.
cost_includes_tax	A 1/0 field indicating whether the cost includes the tax.
created	Time the record was created.

Field Name	Description
currency	Currency for the money fields in the record.
customer_id	The ID of the associated customer.
customerpo_id	The ID of the associated customerpo.
date	The date of the billing slip.
description	The description of the billing slip.
exported	Date and time the record was marked as exported.
hour	The number of hours for a T slip.
id	Unique ID. Automatically assigned by OpenAir.
incomplete	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	The ID of the associated invoice once billed.
item_id	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	The ID of the associated job code.
minute	The number of minutes for a T slip.
name	The name of the slip. This field is never populated.
notes	Notes associated with the slip.
originating_id	For use with split slips feature. If set, the slip.id of the originating slip for this split portion.
payment_type_id	The ID of the associated payment type.
payroll_type_id	The ID of the associated payroll type.
portfolio_project_id	The ID of the associated portfolio project.
product_id	The ID of the associated product.
project_billing_rule_id	The ID of the associated project billing rule.
project_id	The ID of the associated project.
project_task_id	The ID of the task within the associated project.
projecttask_type_id	The ID of the projecttask_type of the associated projecttask.
quantity	The quantity for an E, O, or P slip.
rate	The hourly rate for a T slip. Dated by the date field.
ref_slip_id	For credit/rebill, ID of the original slip ID.
repeat_id	The ID of the associated repeating event.
revenue_projection_type	The type of the projection: ▪ R - Revenue from an "As billed" recognition rule ▪ F - Revenue from an "Fixed fee" recognition rule

Field Name	Description
	■ 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	Id of the revenue recognition rule that created this projection.
revenue_stage_id	Id of the revenue_stage. This will always be 'no revenue stage' 0 for revenue projections.
slip_projection_id	Id of the slip_projection that was used for an as billed rule
slip_projection_type	The type of the slip_projection: ■ 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 ■ 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	The ID of the associated slip stage.
slip_type_id	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	The starting time of the timer.
timetype_id	The ID of the associated time type.
total	The total value of the slip. Dated by the date field.
total_hp	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 can be a small number Dated by the date field."
total_tax_paid	The total tax paid. Dated by the date field.
transaction_id	For internal user only. It is used only to satisfy subtotalling by slip in summary reports.
type	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	The unit of measure for an E or P slip or the rate description for an O slip.
updated	Time the record was last updated or modified.
user_id	The ID of the associated user.
vehicle_id	The ID of the associated vehicle.

This complex type supports the read method.

Note: This complex type cannot be read while projections are running. This is because the table data may be incomplete until the job completes. No results will be returned if a read is attempted while projects are running.

oaRevenueStage

Use this complex type to specify revenue recognition transaction stage information. Index the attributes and use them to filter revenue recognition transactions. oaRevenueStage has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the revenue stage.
revenue_stage_type	A one-character key indicating the type of revenue for this revenue_stage: ■ D - Deferral ■ A - Accrual ■ F - Final
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaRole

Use this complex datatype to specify role information.

Field Name	Description
admin_role	A 1/0 field indicating whether this is the chief administrator role, with full rights.
created	Time the record was created.
default_role	A 1/0 field indicating whether this is the default new user role.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of this role.
notes	Notes associated with this role.
permissions	A set of role permissions
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaSchedulebyday

Use this complex type to retrieve the day-by-day representation of users' work schedules. oaSchedulebyday has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
base_hours	The number of base hours on this date for this user.
created	Time the record was created.
date	The date.
hours	The number of schedule hours on this date for this user, including exceptions.
id	Unique ID. Automatically assigned by OpenAir.
target_base_hours	The number of target base hours for this user on this date Target_utilization.percentage * base_hours.
target_hours	The number of target hours for this user on this date. Target_utilization.percentage * hours.
updated	Time the record was last updated or modified.
user_id	The ID of the associated user.

This complex type supports the read method.

oaScheduleexception

Use this complex type to describe changes to the default work schedule for a company or user. oaScheduleexception has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
enddate	The end date for the exception.
exception_type	The type of exception. R - Date range of the exception.
id	Unique ID. Automatically assigned by OpenAir.
name	The exception name and/or description, e.g. New Years Day.
schedule_request_itemid	The ID of the schedule change item from a schedule request.
startdate	The start date for the exception.
timetypeid	The ID of the associated time type.
updated	Time the record was last updated or modified.
userid	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.

Field Name	Description
workhours	The number of hours per day during this dater ange. This overrides any work hours on each day of either the account schedule or the account/user schedule.
workscheduleid	The ID of the corresponding work schedule.

This complex type supports the read, add, modify, and delete methods.

oaSchedulerequest

Use this complex type to specify schedule request details. oaSchedulerequest has the following children.

Field Name	Description
approval_status	A one-character string indicating the approval status of the schedule request. Possible values: ■ O - Open ■ P - Pending approval ■ A - Approved ■ R - Rejected
attachmentid	If non-zero, the attachment record associated with this schedule request.
attributes	A collection of additional attributes for this complex type.
categoryid	The ID of the associated category.
created	Time the record was created.
customerid	The ID of the associated customer.
date	The date of the schedule request creation.
date_approved	The date the schedule request was approved.
date_submitted	The date the schedule request was submitted.
description	The description or purpose for the schedule request.
enddate	The end date of the schedule request.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the schedule request (Prefix + number).
notes	Notes to print on the schedule request.
number	The schedule request number that increments by 1.
prefix	A static alphanumeric schedule request number prefix.
project_taskid	The ID of the associated project task.
projectid	The ID of the associated project.
startdate	The start date of the schedule request.

Field Name	Description
timetype	The time type of this schedule request: R - regular time or P - personal time.
timetypeid	The ID of the associated time type.
total	The amount of the transaction. Dated by the date field.
updated	Time the record was last updated or modified.
userid	The ID of the user creating the schedule request.

This complex type supports the following methods: read, add, modify, unapprove, upsert, and delete.

oaSchedulerequest_item

Use this complex type to specify information for multiple schedule request items. oaSchedulerequest_item has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
categoryid	The ID of the associated category.
created	Time the record was created.
customerid	The ID of the associated customer.
date	The date of the schedule request item.
externalid	If the record was imported from an external system, you store the unique external record ID here.
hours	The number of hours represented by this schedule request item.
id	Unique ID. Automatically assigned by OpenAir.
name	The schedule request item name. It is the same as the schedule request description.
project_taskid	The ID of the associated project task.
projectid	The ID of the associated project.
request_reference_number	Unique reference number within schedule request.
schedule_requestid	The ID of the associated schedule request.
timetypeid	The ID of the associated time type.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the following methods: read, modify, and delete.

oaSlip

Use this complex type to specify slip information. A slip is an individual timebill or an individual charge to a customer. Multiple slips are aggregated into an invoice. oaSlip has the following children.

Field Name	Description
acct_date	The accounting period date of the slip.
agreementid	The ID of the associated agreement.
attributes	A collection of additional attributes for this complex type.
billing_contactid	The ID of the associated billing contact.
category_1id	The ID of the associated category_1.
category_2id	The ID of the associated category_2.
category_3id	The ID of the associated category_3.
category_4id	The ID of the associated category_4.
category_5id	The ID of the associated category_5.
categoryid	The ID of the associated category. When set, the slip is based on this category.
city	The slip city or location.
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	The ID of the associated cost center.
created	Time the record was created.
currency	Currency for the money fields in the record
customerid	The ID of the associated customer.
customerpoid	The ID of the associated customerpo.
date	The date of the billing slip.
decimal_hours	The number of decimal hours for a T slip.
description	The description of the billing slip.
gl_code	The fixed code 1234455454.
hour	The number of hours for a T slip.
id	Unique ID. Automatically assigned by OpenAir.
invoioceid	The ID of the associated invoice once billed.
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	The ID of the associated job code.
minute	The number of minutes for a T slip.
notes	Notes associated with the slip.
originating_id	For use with split slips feature. If set, the slip.id of the originating slip for this split portion.
payment_typeid	The ID of the associated payment type.
payroll_type_id	The ID of the associated payroll type.

Field Name	Description
portfolio_projectid	The ID of the associated portfolio project.
productid	The ID of the associated product.
project_billing_ruleid	The ID of the associated project billing rule.
projectid	The ID of the associated project.
projecttask_type_id	The ID of the projecttask_type of the associated projecttask.
projecttaskid	The ID of the task within the associated project.
quantity	The quantity for an E, O, or P slip.
rate	The hourly rate for a T slip. Dated by the date field.
ref_slipid	For credit/rebill, ID of the original slip ID.
shipping_contactid	The ID of the associated shipping contact.
skip_recognition	A 1/0 field indicating if this record should be recognized. Used for split charges which were already recognized.
slip_stageid	The ID of the associated slip stage.
sold_to_contactid	The ID of the contact sold to.
tax_location_name	The name of the tax location
timer_start	The starting time of the timer.
timetypeid	The ID of the associated time type.
total	The total value of the slip. Dated by the date field.
total_tax	The total tax paid. Dated by the date field.
total_with_tax	A 1/0 field indicating whether the cost includes the tax.
type	The type of the slip: ▪ T - hourly rate slip ▪ E - expense slip ▪ F - flat price slip ▪ O - other time slip ▪ M - incomplete slip ▪ P - product slip
unitm	The unit of measure for an E or P slip or the rate description for an O slip.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: If not all slips are returned from an API request, determine whether one or both of the following OpenAir internal switches are enabled:

- API should convert charges money fields to invoice currency. Only invoiced slips are returned.

- API should filter out charges associated with charge stages marked to be excluded from invoicing.

To enable or disable them, speak with OpenAir Professional Services or create a support ticket. See Creating a Support Case for instructions on creating a support ticket.

oaSlipProjection

Use the oaSlipProjection complex type to hold slips created from a projected billing run. This complex type includes many of the oaSlip fields with additional fields. The oaSlipProjection has the following children.

Field Name	Description
acct_date	The associated accounting period date.
agreementid	The ID of the associated agreement.
attributes	A collection of additional attributes for this complex type.
billing_contactid	The ID of the associated billing contact.
booking_typeid	ID of the booking type if this was generated from bookings.
categoryid	The ID of the associated category. When set, the slip is based on this category.
city	The slip city or location.
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	The ID of the associated cost center.
created	Time the record was created.
currency	Currency for the money fields in the record
customerid	The ID of the associated customer.
customerpoid	The ID of the associated customerpo.
date	The date of the billing slip.
decimal_hours	The number of decimal hours for a T slip.
description	The description of the billing slip.
hour	The number of hours for a T slip.
id	Unique ID. Automatically assigned by OpenAir.
invoioceid	The ID of the associated invoice once billed.
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_codeid	The ID of the associated job code.
minute	The number of minutes for a T slip.
notes	Notes associated with the slip.

Field Name	Description
payment_typeid	The ID of the associated payment type.
productid	The ID of the associated product.
project_billing_ruleid	The ID of the associated project billing rule.
projectid	The ID of the associated project.
projecttask_typeid	The ID of the associated project task type.
projecttaskid	The ID of the task within the associated project.
quantity	The quantity for an E, O, or P slip.
rate	The hourly rate for a T slip. Dated by the date field.
shipping_contactid	The ID of the associated shipping contact.
slip_projection_type	The type of the slip projection: ■ 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 ■ 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	The ID of the associated slip stage.
sold_to_contactid	The ID of the contact sold to.
timer_start	The starting time of the timer.
timetypeid	The ID of the associated time type.
total	The total value of the slip. Dated by the date field.
transactionid	For internal use only.
type	The type of the slip: ■ T - hourly rate slip ■ E - expense slip ■ F - flat price slip ■ O - other time slip ■ M - incomplete slip ■ P - product slip
unitm	The unit of measure for an E or P slip or the rate description for an O slip.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the read method.

oaSlipstage

Use this complex type to specify the various stages a slip can be in. oaSlipstage has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
enable_slip_tab	Display slips of this stage in a separate tab.
exclude_from_invoicing	Exclude slips of this stage from invoicing.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the stage.
notes	Notes associated with this slip stage.
position	The position of the stage.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaSwitch

Use this complex type to specify information about company and user switches. oaSwitch has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
name	The name of the switch setting.
setting	The contents of the switch setting. A zero length field is considered 'undef'.

Refer to oaCompany and oaUser for more information.

oaTagGroup

Use this complex type to specify user entity tags for users, customers, or projects. oaTagGroup has the following children.

Field Name	Description
active	A 1/0 field indicating whether the record is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
entity_type	The tag group type: U - user, C - customer, or P - project.
id	Unique ID. Automatically assigned by OpenAir.

Field Name	Description
name	Name of the tag group.
searchable	A 1/0 field indicating whether this tag group is searchable. Used only for tag group type = U.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaTagGroupAttribute

Use this complex type to specify attributes associated with user entity tags for users, customers, or projects. oaTagGroupAttribute has the following children.

Field Name	Description
active	A 1/0 field indicating whether the record is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
name	Name of the tag group attribute.
tag_group_id	The ID of the tag group this attribute is in.
updated	Time the record was last updated or modified.

This complex type supports the read method.

oaTargetUtilization

Use this complex type to specify target utilization values for a user. oaTargetUtilization has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
dirty	A 2/1/0 field: ■ 0 - Clean ■ 1 - Dirty ■ 2 - Inprogress
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.
id	Unique ID. Automatically assigned by OpenAir.
percentage	Target utilization for this user as a percentage. For example, 75.30.
start_date	The start date for the target utilization.

Field Name	Description
updated	Time the record was last updated or modified.
user_id	The ID of the associated user.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaTask

Use this complex type to specify task information. oaTask has the following children.

Field Name	Description
acct_date	The accounting period date of the task.
attributes	A collection of additional attributes for this complex type.
category_1id	The ID of the associated category_1.
category_2id	The ID of the associated category_2.
category_3id	The ID of the associated category_3.
category_4id	The ID of the associated category_4.
category_5id	The ID of the associated category_5.
categoryid	The ID of the associated category.
cost_centerid	The ID of the associated cost center.
created	Time the record was created.
customerid	The ID of the associated customer.
date	The date of the task.
decimal_hours	The number of decimal hours for the task.
description	Description of the task.
end_time	The end time of the task.
hours	The number of hours for the task.
id	Unique ID. Automatically assigned by OpenAir.
job_codeid	The ID of the associated job code.
loaded_cost	The loaded cost for the associated resource, using the forex future rate from the exchange rate table."
loaded_cost_2	User's second level loaded cost, using the forex future rate from the exchange rate table.
loaded_cost_3	User's third level loaded cost, using the forex future rate from the exchange rate table.
minutes	The number of minutes for the task.
notes	Notes associated with this task.
payroll_typeid	The ID of the associated payroll type.

Field Name	Description
project_loaded_cost	User's project cost override in project currency. Uses the future rate from the exchange rate table.
project_loaded_cost_2	User's project second cost in project currency. Uses the future rate from the exchange rate table.
project_loaded_cost_3	User's project third cost in project currency. Uses the future rate from the exchange rate table.
projectid	The ID of the associated project.
projecttask_typeid	The ID of the projecttask_type of the associated project_task.
projecttaskid	The ID of the task within the associated project.
slipid	The ID of the associated slip if this task was billed.
start_time	The start time of the task.
thin_client_id	Used by thin clients to reconcile imported records.
timesheetid	The ID of the associated timesheet.
timetypeid	The ID of the associated time type.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: Review the following:

- The **Require a task on time entries** application setting (Administration > Application Settings > Timesheets > Other settings) is not supported. OpenAir API lets you add or modify time entries without an associated task (`projecttaskid`) even if a task is required on time entries in the OpenAir UI.
- By default, the task's **loaded_cost** and **project_loaded_cost** use the forex future rate from the exchange rate table. To force these values to use the exchange rate for the date of the time entry and not the future exchange rate, contact OpenAir Customer Support and request the "API to Respect Time Entry Date for Currency Conversion in Loaded Costs" feature.

Using start time and end time with oaTask

You can read and modify `start_time` and `end_time` for Task.

To modify `start_time` and `end_time`:

- You need the **Enable start and end time entry on timesheets** switch to be able to change `start_time` and `end_time`. The API returns an error (error code: 1406) if you attempt to edit the start/ end times and the feature is not enabled.
- The valid time format is **hh:mm:ss**. Examples: 10:30:15, 2:30, 2:30:15, 2:3, 2:2:2 . The API returns an error (error code: 1404) if an invalid time format or value is passed.

- 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 specify the duration in the API call using `decimal_hours` and/or `hours` and `minutes`.

> **Note:** If you specify `start_time` and `end_time`, the duration is NOT calculated. 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`.
>
> The duration can be set using `decimal_hours` and/or `hours` and `minutes`.

- To clear a `start_time` or `end_time` set it to 00:00:00.
- Setting `start_time` and `end_time` to 00:00:00 will remove hours.

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”;`.

Important: `minutes` does not accept a decimal part.

Important: It is recommended not to 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.

When using the API and both features are enabled, passing `decimal_hours` and `minutes` but not `hours` in the API call will result in an error (error code 1407).

oaTaskAdjustment

Use this complex type to specify time entry adjustments. oaTaskAdjustment has the following children.

Field name	Description
created	Time the record was created.
id	Unique ID. Automatically assigned by the system.
new_taskid	The ID of the adjustment task.
new_timesheetid	The ID of the adjustment timesheet.
old_taskid	The ID of the original task.
old_timesheetid	The ID of the original timesheet.
updated	Time the record was updated.

This complex type supports the read method.

oaTaskTimecard

Use this complex type to specify tasks associated with timecards. oaTaskTimecard has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
category_1id	The ID of the associated category_1.
category_2id	The ID of the associated category_2.
category_3id	The ID of the associated category_3.
category_4id	The ID of the associated category_4.
category_5id	The ID of the associated category_5.
categoryid	The ID of the associated category.

Field Name	Description
cost_centerid	The ID of the associated cost center.
created	Time the record was created.
customerid	The ID of the associated customer.
date	The date of the task timecard.
decimal_hours	The number of decimal hours for the task timecard.
description	The description of the task timecard.
hours	The number of hours for the task timecard.
id	Unique ID. Automatically assigned by OpenAir.
minutes	The number of minutes for the task timecard.
notes	Notes associated with this task timecard.
payroll_typeid	The ID of the associated payroll type.
project_phaseid	The ID of the project phase.
projectid	The ID of the associated project.
projecttask_typeid	The ID of the project task type.
projecttaskid	The ID of the task within the associated project.
slipid	The ID of the associated slip.
time_cardid	The ID of the associated timecard.
timesheetid	The ID of the associated timesheet.
timetypeid	The ID of the associated time type.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the read method.

oaTaxLocation

Use this complex type to specify tax location information. oaTaxLocation has the following children.

Field Name	Description
acct_code_federal	GL accounting code for the federal entries.
acct_code_gst	GL accounting code for the GST entries.
acct_code_hst	GL accounting code for the HST entries.
acct_code_pst	GL accounting code for the PST entries.
acct_code_state	GL accounting code for the state entries.

Field Name	Description
active	A 1/0 field specifying if the location is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
federal_rate	The federal tax rate.
gst_rate	The GST rate.
hst_rate	The HST rate. This is used instead of GST and PST in some locations.
id	Unique ID. Automatically assigned by OpenAir.
name	The name for the estimate adjustment.
notes	Notes associated with this tax location.
pst_rate	The PST rate.
state_rate	The state tax rate.
tax_method	The tax method: ■ G - GST and PST ■ H - HST ■ F - Federal and State
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaTaxRate

Use this complex type to specify tax rate information. oaTaxRate has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
currency	Currency for the money fields in the record.
date	The date (used for currency conversions).
federal	The federal tax. Dated by the date field.
gst	The GST tax. Dated by the date field.
hst	The HST tax. Dated by the date field.
id	Unique ID. Automatically assigned by OpenAir.
notes	Notes associated with this tax rate.
pst	The PST tax. Dated by the date field.
purchase_itemid	The ID of the associated purchase order item.

Field Name	Description
slipid	The ID of the associated slip.
state	The state tax. Dated by the date field.
tax_locationid	The ID of the associated tax location.
ticketid	The ID of the associated ticket.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaTerm

Use this complex type to specify term information. oaTerm has the following children.

Field Name	Description
display	Display the term as.
attributes	A collection of additional attributes for this complex type.
name	The name for the term.

This complex type supports the read method.

oaTicket

Use this complex type to specify ticket information. oaTicket has the following children.

Field Name	Description
acct_date	The accounting period date of the ticket.
attachmentid	If non-zero, the attachment record associated with this ticket.
attributes	A collection of additional attributes for this complex type.
categoryid	The ID of the associated category.
city	The ticket city or location.
cost	The cost per unit of measure. Dated by the date field.
cost_centerid	The ID of the associated cost center.
created	Time the record was created.
currency	Currency for the money fields in the record.
currency_cost	The cost per unit of measure in the foreign currency if this is a foreign currency receipt.
currency_exchange_intolerance	A 1/0 field indicating if the record is within the specified foreign currency tolerance as defined in database data definitions.
currency_rate	The conversion rate if this is a foreign currency receipt.

Field Name	Description
currency_symbol	The currency for foreign currency receipts.
currency_total_tax_paid	The tax paid in the foreign currency if this is a foreign currency receipt.
customerid	The ID of the associated customer.
date	The date of the ticket.
description	The description of the ticket.
envelopeid	The ID of the associated envelope.
externalid	If the record was imported from an external system, store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
itemid	The ID of the associated item.
missing_receipt	If set to 1, the paper receipt is missing for this ticket.
non_billable	If set to 1, this is not billable.
notes	Notes associated with the ticket.
payment_typeid	The ID into the payment type field for the payment method.
paymethod	Payment method now comes from payment_type table. Keep for backwards compatibility.
project_taskid	The ID of the associated project task.
projectid	The ID of the associated project.
projecttask_typeid	The ID of the associated projecttask_type. Only if project_task_id switch is on.
quantity	The quantity.
reference_number	Unique reference number within envelope. Used to cross-reference digital receipts with paper receipts."
slipid	The ID of the associated slip.
status	The status of the ticket: ■ R - reimbursable ■ N - non-reimbursable
tax_location_id	The ID of the associated tax location.
tax_location_name	The name of the tax location.
tax_rateid	The ID of the associated tax rate.
thin_client_id	Used by thin clients to reconcile imported records.
total	The total value of the ticket. Dated by the date field.
total_no_tax	The total paid before tax added. Dated by the date field.
total_tax_paid	The tax paid. Dated by the date field.

Field Name	Description
unitm	The unit of measure.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.
vendorid	The ID of the associated vendor.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: Review the following:

- 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.
- The following switches may impact the ability to add or modify `ticket` records using the API. To enable or disable any of the following switches contact OpenAir Professional Services or create a support ticket. See the help topic Troubleshooting for instructions on creating a support ticket.
  - **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).

oaTimecard

Use this complex type to specify timecard information. oaTimecard has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
break_end	Time they ended the break.
break_start	Time they started the break.
created	Time the record was created.
date	The date of the timecard.
hours	Hours worked.
id	Unique ID. Automatically assigned by OpenAir.
notes	Notes associated with the timecard.
time_end	Time they stopped working.

Field Name	Description
time_start	Time they started working.
timesheetid	The ID of the associated timesheet.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the read method.

oaTimesheet

Use this complex type to specify timesheet information. oaTimesheet has the following children.

Field Name	Description
acct_date	The accounting period date of the task.
approved	The date the timesheet was approved.
approved_by	Empty value kept for backwards compatibility.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
default_categoryid	The default category ID this timesheet is associated with. All new task entries get this default value.
default_customerid	The default customer ID this timesheet is associated with. All new task entries get this default value.
default_payrolltypeid	The default payroll type ID this timesheet is associated with.
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.
default_projectid	The default project ID this timesheet is associated with.
default_projecttaskid	The default task ID this timesheet is associated with. All new task entries get this default value.
default_timetypeid	The default time type ID this timesheet is associated with.
duration	The duration of the timesheet: ■ W - Weekly ■ D - Daily ■ M - Monthly ■ B - Bi-weekly ■ S - Semi-monthly
ends	The ending date of the timesheet.
history	History of events that occurred to the TimeSheet.
id	Unique ID. Automatically assigned by OpenAir.

Field Name	Description
max_hour	Calculated maximum number of hours allowed 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.
min_hour	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	The name of the timesheet.
notes	Notes related to this timesheet.
starts	The starting date of the timesheet.
status	The status of the timesheet: ■ O - Open ■ S - Submitted ■ A - Approved ■ R - Rejected ■ X - Archived
submitted	The date the timesheet was submitted.
thin_client_id	Used by thin clients to reconcile imported records.
total	The total number of hours in the timesheet.
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, upsert, submit, approve, reject, unapprove, and delete.

Note: Refer to the following notes regarding the Timesheet datatype:

- To be able to edit an approved or archived timesheet, the following internal switch must be enabled: API will allow editing of approved and archived Timesheets.
- If the following switches are enabled, timesheets cannot be edited:
  - Do not allow the owner to edit a submitted timesheet
  - Disable editing of exported timesheets
- The minimum number of hours required on timesheet (`min_hours`) and/or maximum number of hours allowed on timesheet (`max_hours`) may be set as fixed hours or as a percentage of the work schedule in Administration > Application Settings > Timesheets > Timesheet rules. These calculated fields support the Read method only. Values are returned only if the corresponding rule is active and the attribute `calculate_hours` is set to '1' in the Read request. Using the attribute `calculate_hours` may slow the response time significantly, particularly if the Timesheet rules are active and set to 'Percent of work schedule'

A user who attempts to modify another user's timesheet must have a full Account Administrator role. Refer to Add/Modify Errors for more information on error code 821 relating to Timesheets.

If you would like to determine whether any of these internal switches are enabled, speak with OpenAir Professional Services or create a support ticket. See Creating a Support Case for instructions on how to create a support ticket.

oaTimetype

Use this complex type to specify information for time types such as regular time, overtime, sick time. oaTimetype has the following children.

Field Name	Description
active	A 1/0 field indicating whether this is time type is active.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems.
cost_centerid	The ID of the associated cost center.
created	Time the record was created.
externalid	If the record was imported from an external system, you store the unique external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the time type.
notes	Notes associated with this time type.
payroll_code	The payroll code for this time type.
picklist_label	Label as shown on form picklist.
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaTodo

Use this complex type to specify information about something that needs to be done. oaTodo has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
contactid	The ID of the associated contact.
created	Time the record was created.
createdbyid	The ID of the user who created the todo item.
customerid	The ID of the associated customer.
dealid	The ID of the associated deal.
due	Date and time the task is due.
finished	Date and time the task was finished.

Field Name	Description
id	Unique ID. Automatically assigned by OpenAir.
name	The name or description of the todo item.
notes	Notes associated with the todo item.
priority	Todo priority (1 - 9).
start	Date and time the task is to be started.
status	Todo status: ▪ A - Active ▪ C - Completed ▪ D - Deferred ▪ N - Not Started ▪ W - Waiting
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the read method.

oaUprate

Use this complex type to specify information about user and project rate combinations. oaUprate has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
categoryid	The ID of the associated category.
created	Time the record was created.
currency	Currency for the money fields in the record.
customerid	The ID of the associated customer.
duration	Billing rate: ▪ H - hourly ▪ D - Daily
id	Unique ID. Automatically assigned by OpenAir.
job_codeid	The ID of the job code this rate is associated with. This is only used in the context of project billing rules.
notes	Notes associated with the user project rate (uprate).
project_billing_ruleid	If project billing rules are used, this is the ID of the associated project billing rule.
projectid	The ID of the associated project.
rate	The billing rate.

Field Name	Description
updated	Time the record was last updated or modified.
userid	The ID of the associated user.

This complex type supports the following methods: read, add, modify, upsert and delete.

oaUser

Use this complex type to specify user information. oaUser has the following children.

Field Name	Description
account_workscheduleid	The ID of the associated user account work schedule.
acct_code	Optional accounting system code for integration with external accounting systems.
active	A 1/0 field indicating where this is designated as an active user.
addr_addr1	Address line one. *See notes below this table*
addr_addr2	Address line two. *See notes below this table*
addr_addr3	Address line three. *See notes below this table*
addr_addr4	Address line four. *See notes below this table*
addr_city	The city. *See notes below this table*
addr_country	The country. *See notes below this table*
addr_email	The user's email address. *See notes below this table*
addr_fax	The user's fax number. *See notes below this table*
addr_first	The user's first name. *See notes below this table*
addr_id	The ID of the associated address. *See notes below this table*
addr_last	The user's last name. *See notes below this table*
addr_middle	The user's middle name.

Field Name	Description
	See notes below this table
addr_mobile	Mobile number. *See notes below this table*
addr_phone	The user's phone number. *See notes below this table*
addr_salutation	The user's salutation. *See notes below this table*
addr_state	The state. *See notes below this table*
addr_zip	The zip code. *See notes below this table*
attributes	A collection of additional attributes for this complex type.
az_approvalprocess	The approvalprocess_id of the expense authorization approval process. This field is mutually exclusive with 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. ▪ 1 - approver is the manager. ▪ 2 - approver is the manager's manager.
book_assign_stamp	Internal hash key.
br_approvalprocess	The approvalprocess_id of the deal_booking_request approval process. This field is mutually exclusive with 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. ▪ 1 - approver is the manager. ▪ 2 - approver is the manager's manager.
br_approver	The user ID of the booking request approver if this is a single user approver process. This field is mutually exclusive with br_approvalprocess. ▪ 1 - approver is the manager. ▪ 2 - approver is the manager's manager.
code	The acct_code.
cost	New cost value.
cost_centerid	The ID of the associated cost center.
cost_currency	Currency of the cost.
cost_end_date	End date for the new loaded cost. If left blank, the new cost will have no end date.
cost_lc_level	If multiple loaded cost levels are enabled, use this field to hold the level of the loaded cost.

Field Name	Description
cost_start_date	Start date for the new loaded cost. If left blank, the new cost will assume the current date as it's start date.
created	Time the record was created.
currency	The currency for money fields.
cv_attachment_id	The ID of the user’s latest CV.
departmentid	The ID of the associated department.
dr_approvalprocess	The approvalprocess_id of the deal_booking_request approval process.This field is mutually exclusive with br_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. ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
external_id	The unique external record ID if the record was imported from an external system .
externalid	If the record was imported from an external system, you store the unique external record ID here.
filterset_ids	A comma separated list of filter set IDs this record should be part of.
filterset_stamp	A unique string which changes when the primary filter set changes for the user.
flags	A collection of oaSwitch values.
generic	A 1/0 field indicating whether this is a generic resource.
hierarchy_node_ids	The IDs of the associated hierarchy nodes.
hint	Password hint.
id	Unique ID. Automatically assigned by OpenAir.
is_user_schedule	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.
job_codeid	The ID of the current job code this user belongs to.
km_filter_set	The ID of the optional filter set for the Workspaces module.
line_managerid	The ID of this user's line manager (will actually be another user_id).
locked	A 1/0 field indicating if this user is locked.
logintime	The date and time of the user's last login.
ma_filter_set	The ID of the optional filter set for the My Account module.
name	The name used for display in lists. This is programmatically generated if not entered."
nickname	The users nickname. This must be unique.
om_filter_set	The ID of the optional filter set for the Opportunities module.

Field Name	Description
password	The user's password.
password_forced_change	A 1/0 field indicating whether the password must change at next login.
payroll_code	The payroll code for this user.
pb_approvalprocess	The approvalprocess_id of the proposals approval process. This field is mutually exclusive with 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. ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
picklist_label	Label as shown on form picklist.
pm_filter_set	The ID of the optional filter set for the Projects module.
po_approvalprocess	The approvalprocess_id of the purchase order approval process. This field is mutually exclusive with 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. ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
po_filter_set	The ID of the optional filter set for the Purchases module.
pr_approvalprocess	The approvalprocess_id of the purchase request approval process. This field is mutually exclusive with 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. ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
primary_filter_set	The ID of the primary filter set for this user.
project_access_nodes	Comma delimited list of hierarchy node IDs for project level access control.
rate	The hourly billing rate.
report_filter_set	The ID of the optional filter set for Reporting.
rm_filter_set	The ID of the optional filter set for the Resources module.
role_id	The ID of the associated role.
sr_approvalprocess	The approvalprocess_id of the schedule_request approval process. This field is mutually exclusive with 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. ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
ssn	The users's social security number.

Field Name	Description
ta_approvalprocess	The approvalprocess_id of the timesheet approval process. This field is mutually exclusive with 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. ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
ta_filter_set	The ID of the optional filter set for the Timesheets module.
tag_end_date	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.
tag_group_attribute_id	The ID of the tag group attribute that is being assigned to the new tag.
tag_group_id	The ID of the tag group for the new tag.
tag_start_date	Start date for the new tag. If left blank, the start date for the new tag will be set to the current date.
tb_filter_set	The ID of the optional filter set for the Invoices module.
te_approvalprocess	The approvalprocess_id of the expense report approval process. This field is mutually exclusive with te_approver.
te_approver	The user_ID of the expense report approver if this is a single approver process. This field is mutually exclusive with te_approvalprocess. ■ 1 - approver is the manager. ■ 2 - approver is the manager's manager.
te_filter_set	The ID of the optional filter set for the Expenses module.
timezone	The user's timezone.
type	Legacy field.
update_cost	Set this field to 1 to enable automatic updating of user loaded cost.
update_tag	Set this field to 1 to enable automatic updating of user entity tags.
update_workschedule	A 1/0 field indicating an update to the user's workschedule.
updated	Time the record was last updated or modified.
user_locationid	The location ID for this user.
week_starts	The day the week starts for this user: ■ 0 - Monday ■ 6 - Sunday
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.
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

Field Name	Description
	sets the user's default hours to 8 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.
workscheduleid	The ID of the associated user workschedule.

This complex type supports the following methods: read, createUser, modify, and delete. Also refer to oaSwitch.

Notes and 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.
  - 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 return generic users in a ReadRequest, add a “generic” attribute to the request. See **generic** in the table above.
- For generic users, you can use both add and upsert: generic flag=1.
- To create OpenAir users, use the createUser method instead of the add method.
- 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), and the `modify` command 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.
- Prior to the October 2022 OpenAir release, you could not specify the address fields to be returned when reading user records. 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_`).

  You can now list the specific address information in the `fields` of the `ReadRequest` complex type to return only the address information required.

Set User Workschedule

Refer to oaUserWorkschedule to read user workschedule information.

To set the user workschedule while updating or creating users:

1. Set the update_workschedule field of the oaUser object to 1.
2. To set up a user-specified work schedule, set the is_user_schedule flag to 1.
   - Populate the workschedule_workdays field with a CSV list of user workdays. The values in the list should be numbers ranging from 0 (Monday) to 6 (Sunday). For example, 0,1,4 would mean that the user works Monday, Tuesday, Friday, while populating the field with a value of just 0 would mean the user only works on Monday.
   - Populate the workschedule_workhours field with a CSV list of hours to be worked each day. The first value corresponds to the Default value, while subsequent values correspond to the days specified in workschedule_workdays. Using the above example, 8,1,2,3 would set the default workhours value to 8, Monday to 1, Tuesday to 2 and Friday to 4.

   Note: If the internal switch "Enable distinct work hours per day on workschedule" is not set, the workschedule_workhours field should only contain one value, the default.
3. Set the is_user schedule flag to 0 to use the company work schedule specified in the account_workscheduleid field.

Update User Entity Tags Automatically

To automatically update a user's entity tags, set the following fields in the oaUser object:

1. update_tag: Set this field to 1 to enable automatic updating of user entity tags.
2. tag_start_date: Start date for the new tag. If left blank, the start date for the new tag will be set to the current date.
3. tag_end_date: 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.
4. tag_group_id: ID of the tag group for the new tag.
5. tag_group_attribute_id: ID of the tag group attribute that is being assigned to the new tag.

Refer to the following example for initial imports:

1. Set update_tag=1. (This enables automatic updating of user entity tag.)
2. Set tag_start_date=blank. (This indicates that the start date should be the current date.)
3. Set tag_end_date=blank.
4. Set tag_group_id=ID of a valid tag group.
5. Set tag_group_attribute_id=ID of a valid tag group attribute.

Note: If the user has no tags currently set and a modify is performed, the user will receive a new default tag with a start date of the current date and the supplied tag_group_id and tag_group_attribute_id.

Refer to the following example for subsequent imports:

1. Set update_tag=1.
2. Set tag_start_date=blank.

3. Set tag_end_date=blank.

Note: On subsequent imports of user tag information, the existing tags are automatically adjusted to accommodate the new tag. The previously imported tag's end date is set to the day before the start date of the new tag, i.e., yesterday, and the tag loses its default status. The new tag assumes default status and has a start date of the current date, i.e., today. Using the above example, assume the following fields were set during a modify on a user object.

After this update, the previously imported tag will have its end date set to the day before the start date of the new tag (yesterday) and will also lose its default status. The new tag will assume default status and will have a start date of today.

Update User Loaded Costs Automatically

The way you automatically update user loaded costs is similar to updating user entity tags, although there are a few key differences.

- First, default costs cannot be set using this method. All costs loaded using this method are treated as historical cost records.
- Second, only costs with the same cost_lc_level are compared when determining which historical records should be altered. If no cost_lc_level is specified, an lc_level of 0 is assumed.

To automatically update user loaded costs, the following fields should be set:

1. update_cost: Set this field to 1 to enable automatic updating of user loaded cost.
2. cost_start_date: Start date for the new loaded cost. If left blank, the new cost will assume the current date as it's start date.
3. cost_end_date: End date for the new loaded cost. If left blank, the new cost will have no end date.
4. cost: New cost value.
5. cost_currency: Currency of the cost.
6. cost_lc_level: If multiple loaded costs are enabled, use this field to hold the level of the loaded cost.

oaUserLocation

Use this complex type to specify user location information. oaUserLocation has the following children.

Field Name	Description
acct_code	Optional accounting system code for integration with external accounting systems.
active	A 1/0 field indicating whether the record is active.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
external_id	The unique external record ID if the record was imported from an external system.
id	Unique ID. Automatically assigned by OpenAir.
name	The name of the user location.
notes	Notes associated with this user location.

Field Name	Description
updated	Time the record was last updated or modified.

This complex type supports the following methods: read, add, modify, and upsert.

oaUserWorkschedule

Use this complex type to retrieve information about user-specific and company-wide work schedules. oaUserWorkschedule has the following children.

Field Name	Description
account_workscheduleid	The ID of the company workschedule to use when userid in not 0.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created
id	Unique ID. Automatically assigned by OpenAir.
name	The company-wide schedule name for company schedules or user's first and last name for user schedules."
updated	Time the record was last updated or modified.
use_this_schedule	Can be 0 or 1. ▪ 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	ID of the user if this is a users work schedule. 0 - if there is a company work schedule.
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”).
workhours	The number of hours worked per day.

This complex type supports the read method.

oaVendor

Use this complex type to specify vendor information. oaVendor has the following children.

Field Name	Description
active	A 1/0 field indicating where this is designated as an active vendor 1/0.
addr_addr1	Address line one. *See notes below this table*
addr_addr2	Address line two. *See notes below this table*

Field Name	Description
addr_addr3	Address line three. *See notes below this table*
addr_addr4	Address line four. *See notes below this table*
addr_city	The city. *See notes below this table*
addr_country	The country. *See notes below this table*
addr_email	The vendor's email address. *See notes below this table*
addr_fax	Fax number. *See notes below this table*
addr_first	The vendor's first name. *See notes below this table*
addr_id	The ID of the associated address. *See notes below this table*
addr_last	The vendor's last name. *See notes below this table*
addr_middle	The vendor's middle name. *See notes below this table*
addr_mobile	The mobile phone number. *See notes below this table*
addr_phone	The phone number. *See notes below this table*
addr_salutation	The vendor's salutation. *See notes below this table*
addr_state	The state. *See notes below this table*
addr_zip	The zip code. *See notes below this table*
attention	To whom purchase orders should be sent.
attributes	A collection of additional attributes for this complex type.
code	Optional accounting system code for integration with external accounting systems."
created	Time the record was created.

Field Name	Description
currency	Currency for the money fields in the record. Also the default currency when a purchase order is created.
externalid	If record is imported from an external system, store external record ID here.
id	Unique ID. Automatically assigned by OpenAir.
name	Vendor name. Displays on all the vendor pop-up windows in the application.
notes	Notes associated with this vendor.
picklist_label	Label as shown on form picklist.
purchaseorder_text	Text to display on every purchase order.
purchaseprder_email_text	Extra text to include in emails announcing purchase orders.
tax_locationid	The ID of the associated tax location.
terms	Standard payment terms for the vendor.
updated	Time the record was last updated or modified.
web	Vendor's Web address.

This complex type supports the following methods: read, add, modify, upsert, and delete.

Note: Prior to the October 2022 OpenAir release, you could not specify the address fields to be returned when reading vendor records. 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_`).

You can now list the specific address information in the `fields` of the `ReadRequest` complex type to return only the address information required.

oaViewfilter

Use this complex type to filter lists or calendars. oaViewfilter has the following children.

Field Name	Description
action	The filter action.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
fields	Comma delimited list of fields.
id	Unique ID. Automatically assigned by OpenAir.
label	The name given to this filter. It appears in the Filter: drop-down list.
limit_values	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:

Field Name	Description

	`_project_stage_id:&quot;3&quot;,&quot;4&quot;`
	`currency:&quot;USD&quot;,&quot;GBP&quot;`
	`_custom_RF_cf_Project_dropdown:&quot;dropdown_value2&quot;,&quot;dropdown_value3&quot;`
	`_custom_RF_cf_Project_pick_list:&quot;2&quot;`

match_all	A 1/0 field. 1 = if all rules met. 0 = if rules must be met.
name	The internal name of the list or calendar this filter is applied to.
updated	Time the record was last updated or modified.
userid	The user who created this filter.

This complex type supports the read method.

oaViewfilterrule

Use this complex type to specify the individual rules for a particular viewfilter. oaViewfilter has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
condition	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	Time the record was created.
field	The field or column to be compared.
id	Unique ID. Automatically assigned by OpenAir.
required	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	The underlying type of the field or column to be compared: C = character string, N = number, D = date, B = Yes/No, P1 = picker_button, P2 = pop-up menu.
updated	Time the record was last updated or modified.
value	The value the field is compared to.
viewfilterid	The viewfilter to which this rule belongs.

This complex type supports the read method.

oaWorkscheduleWorkhour

Use this complex type to read the number of hours worked per day. oaWorkscheduleWorkhour has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.

Field Name	Description
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
updated	Time the record was last updated or modified.
workday	A one-letter string indicating which day of the week. Monday is '0', Tuesday is '1', ..., Sunday is '6'
workhours	The number of hours worked for this day.
workscheduleid	The ID of the associated primary account workschedule.

This complex type supports the read method.

oaWorkspace

Use this complex type to specify workspace associations with other records. oaWorkspacelink has the following children.

Field Name	Description
allow_guests	A 1/0 field indicating whether guests can be subscribed to this.
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
date	The date of the workspace.
description	The description of the workspace.
global	A 1/0 field indiciating if this is a global workspace (available to all users)
global_access	The access permissions for all users ■ 'R' - Read-only ■ 'W' - Read/write ■ 'A' - Administrator
id	Unique ID. Automatically assigned by OpenAir.
name	The workspace name.
notes	Notes.
open	A “1/0” field indicating whether this workspace is open.
updated	Time the record was last updated or modified.
userid	The user ID of the workspace owner.
id	Unique ID. Automatically assigned by OpenAir.
attributes	A collection of additional attributes for this complex type.
name	The workspace name.
description	The description of the workspace.

Field Name	Description
date	The date of the workspace.
userid	The user ID of the workspace owner.

This complex type supports the following methods: read, add, modify, upsert, and delete.

oaWorkspacelink

Use this complex type to specify workspace associations with other records. oaWorkspacelink has the following children.

Field Name	Description
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
external	A 1/0 field indicating if the record is an external link.
id	Unique ID. Automatically assigned by OpenAir.
recordid	The table ID the workspace is associated with.
updated	Time the record was last updated or modified.
url	The URL of external link.
workspaceid	The ID of the associated workspace.

This complex type supports the following methods: read, add, modify, and upsert.

oaWorkspaceuser

Use this complex type to specify workspace user permission information. oaWorkspaceuser has the following children.

Field Name	Description
access	The access permissions for the user: ▪ R - Read-only ▪ W - Read/write ▪ A - Administrator
attributes	A collection of additional attributes for this complex type.
created	Time the record was created.
id	Unique ID. Automatically assigned by OpenAir.
id	Unique ID. Automatically assigned by OpenAir.
userid	The ID of the associated user.

Field Name	Description
workspaceid	The ID of the associated workspace.

This complex type supports the following methods: read, add, modify, and upsert.

Setting Application Switches Via the API

Certain company and user-level switches can be set via the API. Switches are settings that customize the application. They are not settings that affect actual record data.

Switches are set using the oaSwitch object. Currently, only the Company and User objects have a flags collection that holds individual oaSwitch objects. Switches set at the company level will affect an entire account; switches set at the user level will affect only a particular user in an account.

To obtain a list of switches supported by the system, contact OpenAir Customer Support and open a support ticket. See Creating a Support Case.

Code Examples

Code examples are provided for login functions and read functions.

Login Functions

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. The example demonstrates the following functions:

1. Login to the OpenAir Web Services using user-supplied credentials.
2. Create multiple new users in the logged-in user's account with user-supplied information.
3. Logout of the OpenAir web service.

Login Code Example

```java
import java.rmi.RemoteException;
import javax.xml.soap.SOAPElement;
import org.apache.axis.message.SOAPHeaderElement;
import java.io.*;

class Program
{
     // Instance of our OpenAir web service proxy object
     private static OAirServiceSoapBindingStub m_svc;

     // Name of the company any new users will be added to
     private static String m_strCompany;

     // Prompts the user for input from the console
     private static String GetUserInput(String prompt)
     {
          {
               BufferedReader reader = new BufferedReader(new InputStreamReader(System.in));
               return reader.readLine();
          }
          catch (java.io.IOException e)
          {
               return null;
          }
     }

     // Logs into the OpenAir web service using the supplied login credentials.
     // Returns true if successful, false if not
     private static boolean Login() throws javax.xml.soap.SOAPException, javax.xml.rpc.ServiceException
     {
          // setup our login information
          LoginParams lp = new LoginParams();
          lp.setApi_key("************");
          lp.setApi_namespace("company_namespace");
          lp.setUser( GetUserInput("Enter username: ") );
          lp.setPassword( GetUserInput("Enter password: ") );
          lp.setCompany( GetUserInput("Enter company: ") );
          m_strCompany = lp.getCompany();

          try
          {
               // get an instance of our service and login
               OAirServiceHandlerServiceLocator locator = new OAirServiceHandlerServiceLocator();
               m_svc = (OAirServiceSoapBindingStub)locator.getOAirService();

               LoginResult loginResult = m_svc.login(lp);
               System.out.println("Logged in, session ID = " + loginResult.getSessionId()+"\n");

```

```
                // set our session header to contain the session ID so we can perform further operations
                SOAPHeaderElement header = new SOAPHeaderElement("https://my-account-domain.app.openair.com/OAirService",
                "SessionHeader");
                SOAPElement node = header.addChildElement("sessionId");
                node.addTextNode(loginResult.getSessionId());
                m_svc.setHeader(header);
           }
           catch (java.rmi.RemoteException e)
           {
                // catch any login problems and return
                System.out.println(e.toString());
                return false;
           }
           return true;
      }
      // Create a new user using user-supplied information
      private static void CreateUser()
      {
           System.out.println("----------------------------");
           System.out.println("Enter new user information\n");

           // create the company object that the new user will be associated with
           oaCompany company = new oaCompany();
           company.setNickname(m_strCompany);

           // get the new user information
           oaUser user = new oaUser();
           user.setNickname( GetUserInput("Enter username: ") );
           user.setRole_id( GetUserInput("Enter role ID: ") );
           user.setAddr_first( GetUserInput("Enter first name: ") );
           user.setAddr_last( GetUserInput("Enter last name: ") );
           user.setAddr_email( GetUserInput("Enter email: ") );
           user.setPassword( GetUserInput("Enter password: ") );
           try
           {
                // add the user and output any errors encountered
                UpdateResult result = m_svc.createUser(user, company);
                if (result.getErrors() != null)
                {
                     for (oaBase base : result.getErrors())
                     {
                          oaError err = (oaError)base;
                          System.out.println("Error: " + err.getCode() + "\t" +
                          err.getComment() + "\t" + err.getText());
                     }
                }
                if (result.getStatus() == "A")
                     System.out.println("User successfully added");
           }
           catch (Exception e)
           {
                System.out.println("Error while adding user:\n"+e.toString());
           }
      }
      // Application entry point
      public static void main(String[] args)
      {
           try
           {
                // Login to the OpenAir web service and add users
                if (Login())
                {
                {
                     do
                     {
                          CreateUser();
                     } while (GetUserInput("\nAdd another (y/n)?
                     ").toLowerCase().startsWith("y"));

                     m_svc.logout();
                }
           }
           catch (Exception e)
```

```
          {
               System.out.println(e.toString());
          }
          System.out.println("\nDone");
     }
}
```

Read Functions

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. This example demonstrates the following functions:

1. Login to the OpenAir Web Services using user-supplied credentials.
2. Perform a read of all Envelopes in the user's account using the "all" method of the Read function.
3. Perform a read of a single envelope with a user-supplied ID. This portion uses the “equal to” method of the Read function.
4. Log out of the OpenAir WebService.

Read Code Example

```
using System;
using SoapApplication.OpenAir;

namespace SoapApplication
{
     class Program
     {
          /// <summary>
          /// Instance of our OpenAir web service proxy object
          /// </summary>
          private static OAirServiceHandlerService m_svc;

          /// <summary>
          /// Prompts the user for input from the console
          /// </summary>
          private static string GetUserInput(string prompt)
          {
               Console.Write(prompt);
               try
               {
               return Console.ReadLine();
               }
               catch (System.IO.IOException e)
               {
               return null;
               }
          }

          /// <summary>
          /// Logs into the OpenAir web service using the supplied login credentials.
          /// </summary>
          /// <returns>True if successful, false if not</returns>
          private static bool Login()
          {
               // setup our login information
               OpenAir.LoginParams lp = new OpenAir.LoginParams();
               lp.api_key = "********************";
               lp.api_namespace = "company_namespace";
               lp.user = GetUserInput("Enter username: ");
               lp.password = GetUserInput("Enter password: ");
               lp.company = GetUserInput("Enter company: ");

               // get an instance of our service and login
               m_svc = new OAirServiceHandlerService();
```

```
                LoginResult loginResult;
                try
                {
                     loginResult = m_svc.login(lp);
                     Console.WriteLine("Logged in, session ID = " + loginResult.sessionId);

                     // set our session header to contain the session ID so we can perform further
operations
                     SessionHeader header = new SessionHeader();
                     header.sessionId = loginResult.sessionId;
                     m_svc.SessionHeaderValue = header;
                }
                catch (System.Web.Services.Protocols.SoapException e)
                {
                     // catch any login problems and return
                     Console.WriteLine(e.Message);
                     return false;
                }
                return true;
          }

          /// <summary>
          /// Outputs the supplied envelope to the console.
          /// </summary>
          private static void PrintEnvelope(oaEnvelope env)
          {
                Console.WriteLine("-----------------------------");
                Console.WriteLine("ID:\t" + env.id);
                Console.WriteLine("Name:\t" + env.total);
                Console.WriteLine("Date:\t" + env.date);
                Console.WriteLine("Total:\t" + env.total);
                Console.WriteLine("# receipts:\t" + env.tottickets);
                // output any other needed envelope information here
          }

          /// <summary>
          /// Reads all envelops from the OpenAir web service
          /// </summary>
          static void ReadAllEnvelopes()
          {
                // perform the read
                Console.WriteLine("\nPerforming read of ALL envelopes\n");
                ReadRequest req = new ReadRequest();
                req.type = "Envelope";
                req.method = "all";

                try
                {
                     ReadResult[] results = m_svc.read(new ReadRequest[] { req });
                     // iterate through our results and output them to console
                     foreach (ReadResult result in results)
                     {
                          // output any errors
                          if (result.errors != null)
                          {
                               foreach (oaError err in result.errors)
                               {
                                    Console.WriteLine("Error "+err.code + ": " + err.comment + "\t" +
err.text);
                               }
                          }

                          // output the envelope read results
                          if (result.objects != null)
                          {
                               Console.WriteLine("Received "+result.objects.Length+" envelope(s) from
OpenAir");
                               foreach (oaEnvelope env in result.objects)
                               {
                                    PrintEnvelope(env);
                               }
                          }
                     }
```

```
            }
            catch (Exception e)
            {
                Console.WriteLine("Error while reading envelopes:\n" + e);
            }
        }

        /// <summary>
        /// Read a single envelope from OpenAir
        /// </summary>
        private static void ReadSingleEnvelope()
        {
            Console.WriteLine("\n\nPerforming read using \"equal to\" method");
            oaEnvelope envelope = new oaEnvelope();
            envelope.id = GetUserInput("Enter an envelope id: ");
            ReadRequest req = new ReadRequest();
            req.objects = new oaBase[] { envelope };
            req.type = "Envelope";
            req.method = "equal to";
            try
            {
                ReadResult[] results = m_svc.read(new ReadRequest[] { req });

                // iterate through our results and output them to console
                foreach (ReadResult result in results)
                {
                    // output any errors
                    if (result.errors != null)
                    {
                        foreach (oaError err in result.errors)
                        {
                            Console.WriteLine("Error " + err.code + ": " + err.comment + "\t" +
err.text);
                        }
                    }

                    // output the envelope read results
                    if (result.objects != null)
                    {
                        Console.WriteLine("Received " + result.objects.Length + " envelope(s) from
OpenAir");
                        foreach (oaEnvelope env in result.objects)
                        {
                            PrintEnvelope(env);
                        }
                    }
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("Error while reading envelopes:\n" + e);
            }
        }

        /// <summary>
        /// Application entry point
        /// </summary>
        static void Main(string[] args)
        {
            if (Login())
            {
                ReadAllEnvelopes();
                ReadSingleEnvelope();

                // end our session
                m_svc.logout();
            }
            Console.WriteLine("\nPress enter to exit");
            Console.ReadLine();
        }
    }
}
```

Error Code Listing

The API returns error codes that you can use to help you identify problems specific to a query or action on a particular object. The errors are broken out by their type and you can search for one using the error code number.

Server Errors

Error Code	Short Message	More Information
0	Success	The operation was successful
1	Unknown Error	
2	not logged in	Command required a valid Auth, but Auth failed, or was left out of the request
3	too many arguments	More arguments (XML records) were passed to a command than the command accepts
4	too few arguments	Fewer arguments were passed to a command than were expected
5	Unknown Command	There is no command by that name, request failed
6	Access from an invalid URL	Please use the URL you were provided with to access the API
7	Invalid OffLine version	Please upgrade your version of OpenAir OffLine
8	Failure + Dynamic Message	The operation has failed, Please consult the Error record that was passed, this code is reserved for dynamically generated error codes
9	Logged out	Your session is no longer valid, please issue a login command
10	Invalid parameters	Invalid parameters were used, please consult documentation

CreateUser Errors

Error Code	Short Message	More Information
201	invalid company	Create the company first, then create users
202	duplicate user nick	A user with this nickname already exists, try another one
203	too few arguments	You need to specify both a Company object and a User object
204	Namespace error	Users must be created in the same namespace as the company
205	Workschedule error	Invalid account workschedule specified
206	Role error	Invalid role specified

CreateAccount Errors

Error Code	Short Message	More Information
301	duplicate company nick	This company nick is already in use, try another one
302	too few arguments	You need to specify both a Company and User object
303	please pick a different password	The password entered was not hard enough to guess, please pick another to continue
304	Not enabled	CreateAccount operation is not permitted
305	Not enabled to edit password	Editing of passwords is not allowed

Auth Errors

Error Code	Short Message	More Information
401	Auth failed	Generic error message used for all authentication issues other than those specified in this table
402	Old TB login	Internal TB error
409	Account Canceled	This account has been canceled
411	Account conflict, contact customer service	There is a problem with the account. Contacting customer service will allow you to use the account again
413	Account not privileged to access API	This user is not allowed to access the API functionality
414	Temporarily unavailable	The service is temporarily unavailable, please try back in a few minutes
415	Account archived	This account is archived
417	Restricted IP address	Access is not allowed from the client IP address
418	Invalid uid session	The uid passed in is not valid, please login
419	Authentication failed, please retry	If used, a new session ID may be required
420	Authentication failed	If the problem keeps recurring, contact your identify vendor.
421	Account misconfiguration or invalid assertion	Verify account configuration or check identity vendor if issue persists.
422	LDAP server unavailable	Unable to connect LDAP server
423	No permissions to read ServerStatus data	No permissions to read ServerStatus data.
424	No permissions to modify date	User is not allowed to modify another user's data
425	Functionality not available	The functionality is not available for your company
426	You must use account-specific domain	See Getting Started.

API Login Errors

Error Code	Short Message	More Information
501	API authentication required	API access must first be authenticated
502	API authentication failed	The request element must contain key & name attributes
503	Invalid or missing key attribute	N/A
504	Invalid or missing namespace attribute	N/A
505	The namespace and key do not match	N/A
506	Authentication key disabled	This key has been disabled. Contact support for more information
555	You have exceeded the limit set for the account for input objects	Please make sure to observe the limit for input data set for your account
556	XML API rate limit exceeded	The limit of requests allowed for your company has been reached

Read Errors

Error Code	Short Message	More Information
601	Invalid id/code	There isn't a record matching the ID or code you asked for
602	Invalid field	N/A
603	Invalid type or method	N/A
604	Attachment size exceeds space available	Contact your OpenAir administrator to request more space
605	Limit clause must be specified and be less than the account limit for output data	N/A
606	Projections are running, please try again in a few minutes	N/A

Delete Errors

Error Code	Short Message	More Information
701	Cannot delete, failed dependency check	You must first delete all the records that have an index pointing to this record
702	Invalid note	The note could not be deleted

Add/Modify Errors

Add/Modify Errors

Error Code	Short Message	More Information
800	Synchronization failed	Failed to propagate the changes to identity sub-system. Please try again later.
801	Not a valid Customer ID	The Customer ID you tried to associate with this Project does not exist, or is deleted
802	This Envelope number is already taken	Please select a different Envelope number, or specify none for auto-numbering
803	This user does not have permission to modify the record	The non-administrative user is trying to modify an administrator only record
804	Not a valid Item type	The only valid types are R and M
805	Reference number in use	The reference number is already in use, please select a different one
806	Already accepted by signer	You cannot modify tasks or tickets that have already been accepted by a signer
807	Invalid payment type	The payment type passed is not valid (possibly inactive, or deleted)
808	Invalid note	The note you are trying to modify is not valid
809	Invalid Timesheet	The timesheet you specified for this task does not exist, or has been deleted
810	Invalid index	The index you specified doesn't exist in that table
811	Invalid predecessor	One or more IDs in the predecessor list could not be found
812	Invalid parentid	The parentid field has an ID that is not valid
813	Invalid projectid	The projectid specified doesn't exist, or was deleted
814	duplicate id_number	This id_number is already in use for this project
815	Projecttask does not exist	The Projecttask you specified does not exist
816	User role/type does not exist	The role_id or type you specified is invalid
817	Invalid envelope	The envelope ID specified does not exist
818	duplicate user nick	A user with this nickname already exists, try another one
819	Slip cannot be deleted	This slip is part of an Invoice, and cannot be deleted
820	Envelope not open	The envelope cannot be modified because it is no longer open
821	Timesheet not open	This error is returned under the following conditions: - The timesheet cannot be modified, it has been submitted for approval.

Error Code	Short Message	More Information
		- The timesheet is has status (A/X - Approved/Archived) and internal switch allowing editing of approved timesheets is not enabled. - Timesheet is not in Open Period and user's role doesn't allow editing of timesheets outside of Open Periods. - Timesheet has status S (Submitted) and is modified by the submitter, while internal switch doesn't allow editing of submitted timesheets by owner. - Timesheet has been exported and internal switch disallowing modification of exported timesheets is turned on. - When a user who does not own a full Account Administrator role attempts to modify timesheet of another user via API.
822	Slip cannot be modified	This slip cannot be edited
823	Slip, bad invoice id	The slip is already in an invoice, and cannot be moved to another invoice
824	Must specify name or company	The Customer/Prospect must have a valid name or company
825	Invalid invoice	The invoice ID specified does not exist
826	Date is required	N/A
827	Reimbursements can only be applied after the envelope is approved	N/A
828	This Invoice number is already taken	Please select a different Invoice number, or specify none for auto-numbering
829	Not a valid user	The user you specified is invalid
830	Not a valid booking type	The booking type you specified is invalid
831	No startdate or enddate specified	You must specify startdate and enddate
832	Illegal date range	Startdate must be before enddate
833	Percentage not specified	Percentage must be specified
834	Hours not specified	Hours must be specified
835	Only owner can edit this project	N/A
836	Not allowed to add entity	You must have permission to add entity
837	Not a valid account currency	You can only specify a currency currently enabled for the account
838	Not allowed to have more than one current costs per user	You can only have one cost current record per user
839	base64_data must be set to add an attachment	N/A
840	Not a valid primary filter set	The primary filter set you specified is invalid

Error Code	Short Message	More Information
841	Invalid email	Email is a required field
842	Invalid period	Period is a required field
843	Invalid timing	Timing is a required field
844	Invalid leave accrual rule	leave_accrual_ruleid is a required field
845	Invalid task	Task is a required field
846	Cannot create non-po purchase items	Your account or role is not configured for non-po purchase items
847	Purhaseorderid must be blank	Non-po purchase items should not be associated with a PO
848	Only non_po purchase items can be added/ modified	Non-po must be set to 1
849	Another record with the same date range already exists	Overlapping records are not allowed
850	Another record already exists as a default for this user and group	Only one default record can be added for the user and group
851	Not a valid tag_group_attribute	The tag group attribute you specified is invalid
852	Duplicate external_id	Another record with the same external_id is already present
853	Invalid Loaded Cost parameters	When current is set to '1', start and end dates must not be filled and vise versa
854	Too many records requested	Please modify your filter parameters to limit the data returned. If using Integration Manager, contact OpenAir Customer Support.
855	Number of commands passed in is greater than the account limit for the API	Please separate your commands into separate requests
856	Date overlaps with existing record	The start or end dates you specified overlap with those of an existing record
857	Date range exceeded maximum	The date range specified exceeded maximum allowed
858	ForexInput error	Please note the update error
859	Invalid customer id	The customer ID specified doesn't exist or was deleted
860	default_for_entity and start and end dates are mutually exclusive	Cannot set default_for_entity and start and end dates for the same record
861	Invalid customer id	The customer ID specified does not match the parent invoice customer id
862	Invalid project id	The project ID specified is not associated with the parent invoice customer
863	Only one project per invoice	The invoice specified is already associated with a different project
864	Error while saving user workschedule	There was an error saving the user workschedule

Error Code	Short Message	More Information
865	Invalid workdays	Workdays must be a CSV list containing digits between 0 (Monday) and 6(Sunday)
866	Invalid workdays or workshours	Workday and workhour values are required when setting a user workschedule
867	Distinct workhours not enabled	Only one workhour value (the default) can be specified when updating a user workschedule
868	Invalid type specified	Type must be filled and one of (project, user, customer)
869	Invalid value for primary_user_filter	primary_user_filterset can only be specified for one hierarchy of project type
870	Invalid value for primary_dropdown_filter	primary_dropdown_filter can only be specified for one hierarchy of project type
871	Invalid number of read arguments supplied	The number of argument objects must equal the number of filter clauses
872	Invalid cost type	There is no cost type with specified id
873	Invalid period	Period must be specified
874	Schedule request error	Please note the update error
875	Repeat error	Please note the update error
876	Attachment too small	Attachment size is too small
877	Invalid project group	project_group_id specified does not exist
878	Purchaseorder not open	The purchase order cannot be modified because it is no longer open
879	Invalid purchase order	The purchaseorder_id specified does not exist
880	Invalid purchase item	Non-PO purchase items must have a positive quality
881	Invalid attachment	Specified parent ID does not exist or parent is in a different workspace
882	Invalid reference slip ID	Specified reference slip doesn't exist or was deleted
883	Invalid portfolio project ID	Specified portfolio project ID is invalid, doesn't exist or doesn't match customer
884	Invalid portfolio link	Portfolio project cannot be subordinated to another portfolio project
885	Invalid purchase item	Mandatory date is missing in purchase item
886	Project task type mismatch	Project task type invalid, or project task not defined
887	Wrong project assignment profile name	This project assignment profile ID is already taken for the project
888	Timesheet task invalid date	The task date is not within the required project task assignment date range
889	Ticket cannot be modified	This ticket cannot be edited

Error Code	Short Message	More Information
890	User cannot be modified	This user cannot be edited
891	Invalid user	The user ID specified does not exist
892	Invalid envelope	The envelope ID specified does not exist
893	Invalid receipt	The receipt ID specified does not exist
894	Invalid timesheet	The timesheet ID specified does not exist
895	Invalid customerpo	The customerpo ID specified does not exist
896	Agreement cannot be modified	This agreement cannot be edited
897	Customerpo cannot be modified	This customerpo cannot be edited
898	Invalid workspace	The workspace ID specified does not exist
899	Invalid expense policy	The expense_policy ID specified does not exist
900	Invalid item	The item ID specified does not exist
947	Project already has an expense policy	A project can have only one expense policy associated
948	Duplicate itemid for expense policy	A unique expense_policy_id and item_id pair must be specified
949	Invalid Resourceprofile_type ID specified	An existing Resourceprofile_type must be specified
950	Invalid Attribute ID specified	An existing Attribute must be specified
951	Duplicate Attribute for Resourceprofile_type	A unique attribute_id and resourceprofile_type_id pair must be specified
952	Duplicate entry for user	The entry must be unique for given user
953	Missing labor subcategory	A labor subcategory must be set for the project budget group. See the help topic ProjectBudgetGroup.
954	Invalid Project billing rule ID specified	An existing Project billing rule must be specified
1400	Missing start_end_month_ts flag	Please specify a valid start_end_month_ts flag for the Timesheet.
1401	Invalid associated_tmid	Specified associated_tmid is invalid, please consult documentation.
1402	Non-overlapping timesheet	You cannot specify associated_tmid nor start_end_month_ts flag for non-overlapping timesheets.
1403	Cannot modify timesheet with associated_tmid	You cannot modify specific field of associated timesheets, please consult documentation.
1404	Invalid time	Time must be a valid value.
1405	Illegal time range	Start time must be before end time.
1406	No permission to edit time data	Account does not have allowed feature to edit start/end time data.
1407	Invalid hours	The hours do not match the start and end time.

Error Code	Short Message	More Information
1408	Invalid newsfeed	The newsfeed with specified ID does not exist
1409	Both author or editor not set	The newsfeed message requires author or editor to be set
1410	Deactivate ns integration user	The user is set as an integration netsuite user, therefore it is not possible to deactivate the user.
1411	Change admin role of ns integration user	The user is set as an integration netsuite user, therefore it is not possible to change the user's administrator role.
1412	Invalid quantity	Quantity must be non-zero number
1413	Invalid payment terms	Payment terms ID must correspond with used terms
1414	Invalid approval status	The approval status value must be valid. The approval status value is a one-character string and must be one of the possible values for that field.
1415	Phase cannot be assigned	Phase cannot have project task assignments.
1416	Invalid cap by customer PO	The cap by customer PO is not valid. See oaProjectbillingrule.
1417	Invalid project task id	The project task ID is not valid. ■ `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).
1418	Invalid preference settings format	The preference settings format is not valid.
1419	No full user licenses available	User cannot be granted access to modules other than Account, Expenses and Timesheets.
1420	No T&E or full user licenses available	Cannot add T&E user or mark T&E user as active.
1421	No guest or full user licenses available	Cannot add guest user or mark guest user as active.

MakeURL Errors

Error Code	Short Message	More Information
901	The combination of uid, app, arg, and page is not valid	The values passed don't combine to represent a valid page, check the values and try again
902	A valid record could not be created from the arg passed	Check to make sure the required fields are being passed in the arg record
903	The user does not have access to that page	That combination of app, arg, and page is not valid for this user
904	This Purchaseorder number is already taken	Please select a different Purchaseorder number, or specify none for auto-numbering

Error Code	Short Message	More Information
905	Invalid purchaseorder	The purchaseorder ID specified does not exist
906	Invalid Cost Center	The cost_centerid specified does not exist or is inactive
907	Invalid Contact	First name, Last name and email are required fields
908	Invalid Name	Please specify a valid name for the record
909	Invalid Contact	The contact must exist, and belong to the same Customer
910	Lookup record not located	One or more lookup fields specified for the record do not exist
911	No Timesheet specified	Timesheet ID must be specified to edit a task
912	Invalid type Specified	Type must be set
913	Invalid project task specified for a project	Project task must belong to a project specified
914	Invalid resourceprofile_type_id specified	An existing resourceprofile_type ID must be specified
915	Invalid type specified	The type and resourceprofile_type_id must be provided and must match the type-id pair in an existing record in the resourceprofile_type table
916	Table specified does not have external_id field	Make sure you selected correct association
917	This Issue number is already taken	Please select a different Issue number, or specify none for auto-numbering
918	No description specified	Issue description must be set
919	Only one default issue stage is permitted	Only one issue stage may be marked as default_for_new
920	No rate card ID specified	Rate card ID must be specified
921	Job code in use for rate card	The supplied job code is already in use for the associated rate card
922	Invalid job code specified	An existing job code must be specified
923	Invalid rate card specified	An existing rate card must be specified
924	No job code ID specified	Job code ID must be specified
925	Invalid template project ID specified	A valid project ID must be supplied for the template project ID
926	Invalid value for user cost	User cost must contain a valid value
927	Invalid user cost start date	User cost start date must not be before any previous cost start date
928	Invalid project group ID for workspace user	Project group ID must contain a valid value
929	Workspace user cannot contain both project group ID and user ID	Only project group ID or user ID can be set
930	Generic flag cannot be modified	Cannot change generic resources into users and vice versa

Error Code	Short Message	More Information
931	Duplicate project assignment	A user can only be assigned to a project o
932	Only admin users may update proxies	Only users in the administrator role may update proxy information
933	Not a valid proxy user	The proxy user ID you specified is invalid
934	Error while creating project from template	There was an error while creating a project from a template project
935	Invalid user tag start date	User tag start date must not be before any previous tag start date
936	Error while creating project group assignments	There was an error while creating project group assignments
937	Invalid agreement ID specified	An existing agreement must be specified
938	Duplicate agreement_to_project	A unique project_id and agreement_id pair must be specified
939	View is not allowed for this user	Please check that the user logged in has the required role
940	Dashboard view is not allowed for this project	Please check that the project is configured for dashboard view
941	Invalid timezone specified for user	Timezone string must contain a +/- sign, four digit offset, and optionally a single letter, e.g.,: -0500,+0330, +1300a
942	Loaded costs not allowed for generic resources	Loaded costs are not allowed for generic resources
943	Project names must be unique by customer	Project names must be unique by customer
944	Invalid date	Date must be a valid value

Project Budget Errors

Error Code	Short Message	More Information
945	Invalid Project budget group ID specified	An existing Project budget group must be specified
946	Invalid project budget rule ID specified	An existing Project budget rule must be specified

Resource Attachment Errors

Error Code	Short Message	More Information
960	Invalid Resource attachment type	Allowed types: CV
961	Duplicate entry for user	Each user can have only one resource attachment of given type
962	ResourceAttachment cannot by modified	This ResourceAttachment cannot be edited

Error Code	Short Message	More Information
963	Invalid attachment id	This attachment ID does not exist or it does not have association/ record for given user.
964	Invalid ResourceAttachment id	This ResourceAttachment ID does not exist
965	File could not be saved	The attachment record was created but marked as deleted. Try adding the attachment again and if the error persists, contact OpenAir Customer Support.

Approve/Submit Errors

Error Code	Short Message	More Information
1001	Invalid state	Record could not be submitted, because it is currently not Open or Rejected
1002	Submit/Approve error	There are errors associated with this request
1003	Submit/Approve warning	There are warnings associated with this request

Hierarchy Errors

Error Code	Short Message	More Information
1050	Invalid hierarchy node specified	Please specify a valid hierarchy node
1051	You cannot assign multiple nodes within one hierarchy	Please specify a different hierarchy node

Custom Field Errors

Error Code	Short Message	More Information
1100	Invalid value specified for a checkbox custom field	Please specify either empty string or 1
1101	Value specified is not on the list of values for this custom field	Please specify one of the valid values for this custom field
1102	Custom field could not be saved	Please check specific error descriptions
1103	Modification of the field specified is not supported	Only valuelist field can be modified at this time
1104	This custom field value is not unique	You must enter a unique value
1105	Value specified is not on the list of values in the source pick list defined for this custom field	Please specify one of the valid values from the source pick list for this custom field
1106	One or more inline custom fields failed to be updated	Please review specific errors returned

ModifyOnCondition status/error

Error Code	Short Message	More Information
1200	Condition not met	Command wasn't executed because condition wasn't met. Returning the record from DB

Filterset Errors

Error Code	Short Message	More Information
1300	Invalid filter set specified	Please specify a valid filter set

Module Access Errors

Error Code	Short Message	More Information
1500	Access to the Expenses module denied.	Contact your administrator.

Note: User application access rights for the Expenses module determines if the authenticated user can make supported calls on the `oaEnvelope` and `oaTicket` complex types. Application access rights for other modules have no effect on what the authenticated user can access using OpenAir SOAP API.

XML Errors

Error Code	Short Message	More Information
2001	Invalid argument passed	Please make sure to pass valid arguments
2002	Invalid format passed	Please make sure to pass valid format

OpenAir Data Dictionary

Note: To view the OpenAir Data Dictionary, use the following URL: `https://<account-domain>/database/single_user.html`.

- `<account-domain>` is the account specific domain for your account.
- To view the details of a specific table, append a hash symbol # followed by the table name to the end of the data dictionary URL. For example, use `https://<account-domain>/database/single_user.html#project` to view the details of the Project table.
- You can access the data dictionary from the OpenAir Help Center using the link in the navigation bar if you have the View Help Center role permission.

Best Practices

Before you begin using OpenAir SOAP API functionality, ensure your OpenAir account is fully configured and in production. As you know, OpenAir provides a number of ways you can customize your company's account to meet unique business requirements. While this flexibility allows you to maximize its effectiveness for your organization, it is helpful to establish the system before you try to access the tables and data fields within it.

Note: We highly recommend that you work with your OpenAir Professional Services (PS) consultant to design the API integration. 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 SOAP 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 in your sandbox account, which provides you with a safe environment. Each step is explained in more detail in the following.

Step 1: Plan What You Want To Do

Think about what you are trying to achieve in your OpenAir implementation and how the SOAP 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 PS 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 begin the actual development.

Step 3: Develop Your Integration

Read this document in its entirety, talk with your PS consultant, and learn about the OpenAir data model and how it is used. Links to key information are provided in Introduction to OpenAir Web Services as well as in Getting Started.

- Develop the API integration with the help of your PS 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 corrupt 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 PS 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, we recommend that records be grouped 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. We recommend running batch operations during off-peak hours to minimize impact on integration performance.

Make Fewer Calls

Reducing the number of calls you make to the API improves the performance of your integration. Because API calls require a call/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. When working with custom fields in OpenAir SOAP Web Services, we recommend including the custom fields in the wsdl file and reading and updating custom fields as part of the native object update, as opposed to using the legacy “custom equal to” methods. See Modifying Records to Set Custom Field Values. Running batch operations during off-peak hours also minimizes impact on performance.

While the API technically allows concurrent connections, running the API from multiple clients simultaneously is NOT recommended. This 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: Please refer to the Limits section in Introduction to OpenAir Web Services for more information regarding possible throttling controls. Batching multiple API operations into one request and caching data locally are the best methods to avoid our servers ever triggering throttling controls.

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. For example, you could retrieve many charges with the same project_id or many timesheet entries with the same user_id.

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 then hold those project records locally, either in memory or via persistent storage to use with the next batch. When you use a persistent cache, the integration could make sure it's up to date via use of our “newer-than” filters. We recommend getting list data, caching it, and then keeping it in sync by requesting records that have changed since the previous update. OpenAir Web Services also allows you to request deleted data 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 very small. For example, even a very large OpenAir account may have only 3 or 4 timetypes and every time entry record will then have one of those 3 or 4 values. Once the timetype records have been retrieved, they can be held locally for an indefinite period as timetype 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 external_id values (as saved in OpenAir) in place of internal OpenAir IDs when related data is being referenced on a OpenAir record during an modify/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 external_id was not found in OpenAir system. See Example II. modify using external_id as foreign key lookup field C# and Example II. externalid as foreign key lookup field 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, we recommend that you keep a local cache of records. See Cache Locally. Issuing a read method call that requests records that have been added/modified and/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 Example VI. read not-exported Envelopes with date filter C# and Example IV. read date filter 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, we recommend exporting approved entries and then marking 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 and their Slips. Since slip records are the list/child element of an Invoice, you can mark each individual Slip record as being exported. Subsequent integration runs issue a read request and the "not-exported" attribute/filter only returns qualifying transactions, i.e., 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 Example V. read not exported C# and Example III. read not exported 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 PS 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. We recommend that you create a communication log that stores a minimum of the last seven days transactions. See Creating a Support Case for information on getting help from OpenAir Customer Support.

Upgrade With Caution

Once your API integration is tested and you move it into production, you need to determine a process for upgrading or making changes to the OpenAir system. We recommend that you do not make changes to the OpenAir production system before testing them in your sandbox account against the API integration. 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 in the production system.

Creating a Support Case

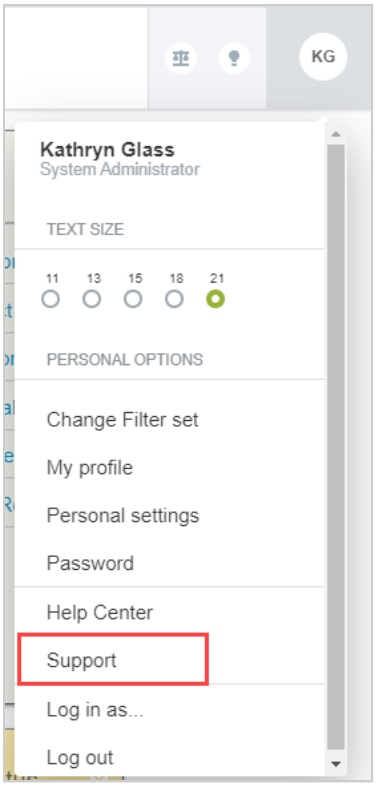
If you are experiencing difficulties with OpenAir or would like to enable an optional feature, go to SuiteAnswers through your OpenAir account and create a support case.

Our support staff and engineers will work with you to find a solution to your problem.

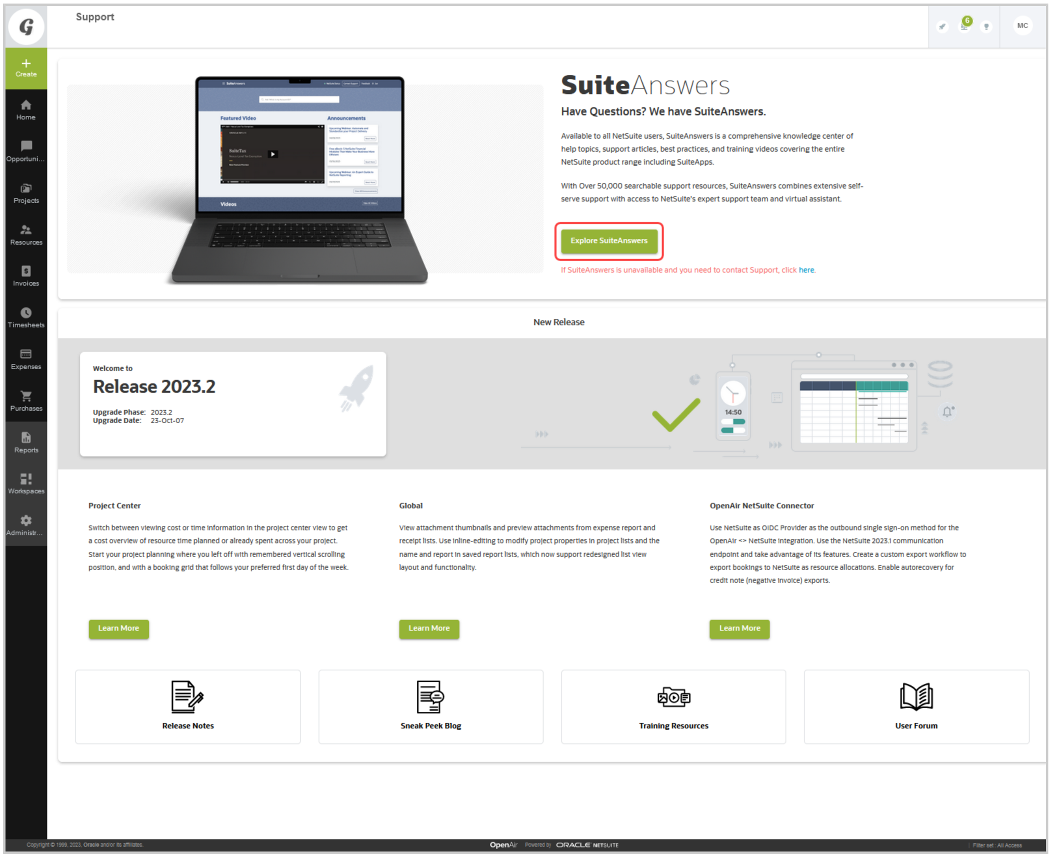
Important: As a part of the support case creation process you will be presented with existing answers that may solve your problem. Take a moment to view the available answers before proceeding to create a support case.

To create a support case:

1. Log in to your OpenAir account and select **Support** from the User Center menu.



2. Click **Go to SuiteAnswers**.



SuiteAnswers

Have Questions? We have SuiteAnswers.

Available to all NetSuite users, SuiteAnswers is a comprehensive knowledge center of help topics, support articles, best practices, and training videos covering the entire NetSuite product range including SuiteApps.

With Over 50,000 searchable support resources, SuiteAnswers combines extensive self-serve support with access to NetSuite's expert support team and virtual assistant.

If SuiteAnswers is unavailable and you need to contact Support, click here.

Welcome to
Release 2023.2
Upgrade Phase: 2023.2
Upgrade Date: 23-Oct-07

Project Center

Switch between viewing cost or time information in the project center view to get a cost overview of resource time planned or already spent across your project. Start your project planning where you left off with remembered vertical scrolling position, and with a booking grid that follows your preferred first day of the week.

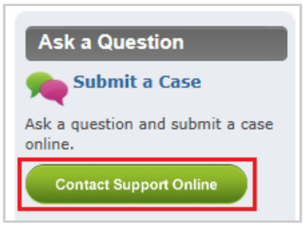
Global

View attachment thumbnails and preview attachments from expense report and receipt lists. Use inline-editing to modify project properties in project lists and the name and report in saved report lists, which now support redesigned list view layout and functionality.

OpenAir NetSuite Connector

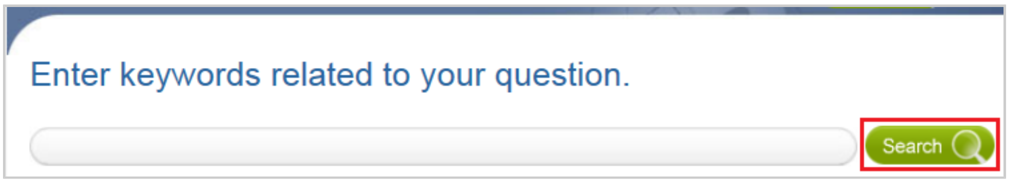
Use NetSuite as OIDC Provider as the outbound single sign-on method for the OpenAir <> NetSuite integration. Use the NetSuite 2023.1 communication endpoint and take advantage of its features. Create a custom export workflow to export bookings to NetSuite as resource allocations. Enable autorecovery for credit note (negative invoice) exports.

3. On the OpenAir SuiteAnswers website, click **Contact Support Online**.



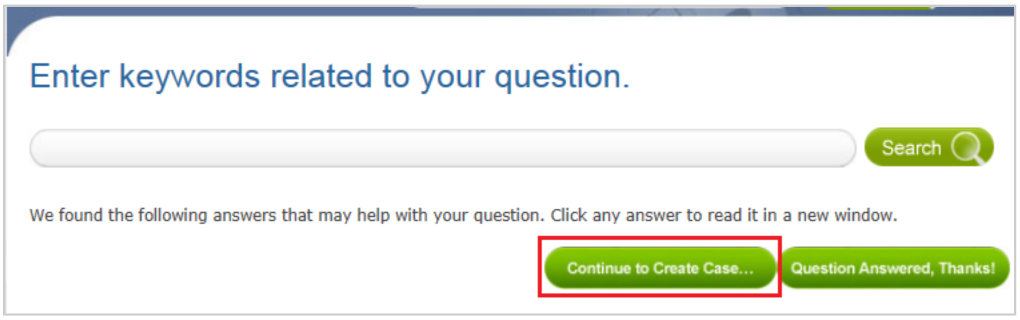
Ask a question and submit a case online.

4. Enter keywords corresponding to the question or problem you want to resolve and click **Search**.



Note: If you do not have a question but need a feature enabled, for example, click **Search**.

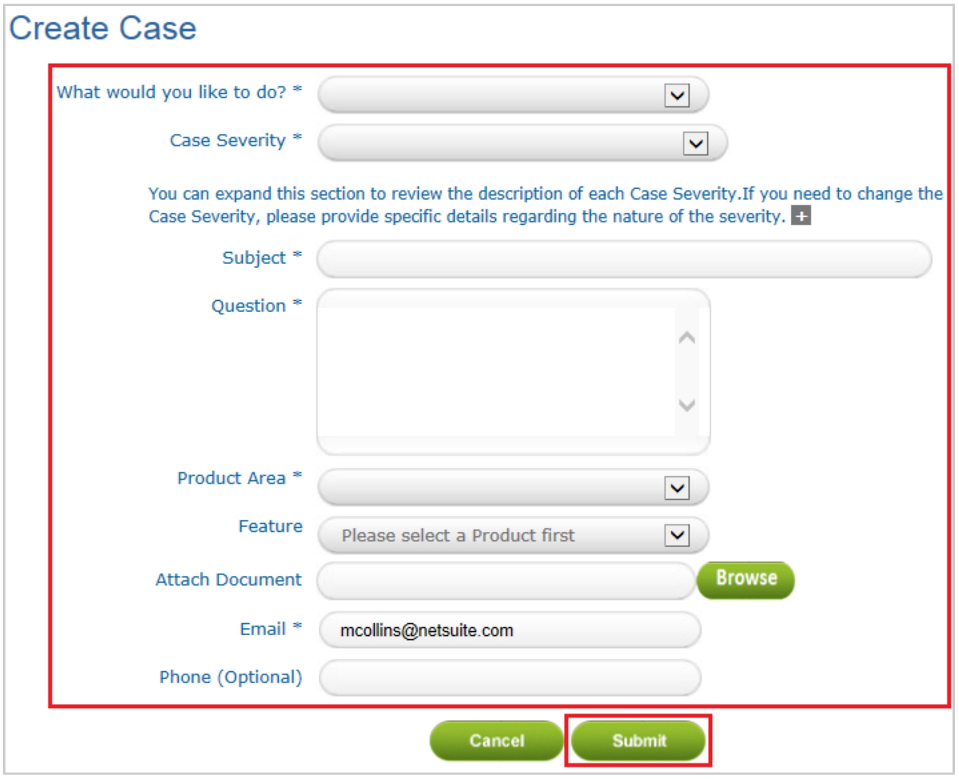
5. Oftentimes, the answer to your question will be displayed. If you still want to create a support case, click **Continue to Create Case**.



Enter keywords related to your question.

We found the following answers that may help with your question. Click any answer to read it in a new window.

6. Fill out the **Create Case** form and then click **Submit**. You will receive an email confirmation with your support case reference (OpenAir Customer Care #).



Create Case

What would you like to do? *	
Case Severity *	

You can expand this section to review the description of each Case Severity.If you need to change the Case Severity, please provide specific details regarding the nature of the severity.

Subject *	
Question *	
Product Area *	
Feature	Please select a Product first
Attach Document	
Email *	mcollins@netsuite.com
Phone (Optional)	

Note: An asterisk * displays after required fields.

New Features

The following summarizes the additions, updates, and enhancements to the SOAP API categorized under the release date.

Interim

- Added error code 965 — File could not be saved. See Resource Attachment Errors

Features for April 15, 2023

- Added the following error codes:
  - 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 Add/Modify Errors. See also complex type oaUser, and methods createUser and modify.
- Added error code 426 — You must use an account-specific domain. See Auth Errors.
- Added support for the unapprove method to the oaSchedulerequest complex type.
- Added support for the Attachment Thumbnail feature. See oaAttachment.

Features for October 8, 2022

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaCustomer	customer_location_id
oaCustomerLocation	active, created, deleted, ID, name, notes, updated

- Solved a previous limitation that prevented specifying the address information to be returned. Impacted complex types and fields:

Complex Type	Fields Exposed
oaCompany	Property names beginning with `addr_`
oaContact	Property names beginning with `addr_`
oaCustomer	Property names beginning with `addr_`, `billing_addr_`, or `contact_addr_`
oaUser	Property names beginning with `addr_`
oaVendor	Property names beginning with `addr_`

Features for April 9, 2022

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaProjectbillingtransaction	fulfillmentid
oaViewfilter	limit_values

- Added Error code 206 — Role error. See CreateUser Errors.

Features for October 9, 2021

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaProjectbillingrule	cap_by_customerpo, project_task_id

- Added Error code 1416 — Invalid cap by customer PO. See Add/Modify Errors.
- Added Error code 1417 — Invalid project task ID. See Add/Modify Errors.
- 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.

Features for April 10, 2021

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaResourcesearch	location, skill, industry, jobrole, education, customprofile_1 — customprofile_35

- Added Auditing and Managing OAuth 2.0 Authorizations under OAuth 2.0 Authorization — Account administrators can use web services reports to audit and revoke authorizations granted by OpenAir users to integration applications.
- Added Error code 1414 — Invalid approval status. See Add/Modify Errors.
- Added Error code 1415 — Phase cannot be assigned. See Add/Modify Errors.
- Added Error code 1500 — Access to the Expenses module denied. See Module Access Errors.

Features for October 10, 2020

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaInvoice	payment_termsid

- Added an option to disallow adding or modifying `oaTicket` complex type with the field `quantity` set to zero and corresponding error code (1412 — Invalid quantity). See oaTicket and Add/Modify Errors.

Features for April 18, 2020

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaJobCodeUsed	id, table_name, used_by, position, created, updated
oaResourceRequestQueue	booking_type_id

- Added support for OAuth 2.0 token based authentication. See OAuth 2.0 Authorization and the SessionHeader web services method complex type.

Features for October 12, 2019

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaProjectbillingrule	extra_data
oaProjecttaskassign	rule_rate_override, rule_rate_override_currency
oaTimesheet	min_hours, max_hours
oaWorkscheduleWorkhour	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 on the timesheet in the oaTimesheet complex type as determined by Timesheet rules. This includes:
  - Added calculated Fields `min_hours` and `max_hours` to oaTimesheet complex type.
  - Added Attribute `calculate_hours` for the ReadRequest.
- Added support for the delete method to the oaUprate complex type.

Features for April 13, 2019

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaNewsfeed	id, created, updated, attributes
oaNewsfeedMessage	id, newsfeedid, title, content, tagid, created, authorid, updated, editorid, attributes
oaProject	newsfeedid
oaProjectbillingtransaction	currency

- Added Administration > Global Settings > Account > API Limits screen. See Managing Your Account Frequency Limits.

Features for October 13, 2018

Bookings are now supported for the submit, reject, approve, and unapprove Methods.

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaTask	start_time, end_time
oaProjectBudgetGroup	etc, etc_labor, etc_expense, etc_purchase, eac, eac_labor, eac_expense, eac_purchase, itd, itd_labor, itd_expense, itd_purchase

- Error codes and related information was added for error codes 1404, 1405, 1406 and 1407.

Features for April 14, 2018

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaProjecttaskEstimate	id, project_task_id, user_id, timesheet_id, hours, date_changed, changed_by, created, updated

- Added the ability to access a Document style with literal use version of your OpenAir WSDL. See Step 2: Generate the OpenAir Web Service WSDL.

Features for October 14, 2017

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaProxy	id, user_id, proxy_id, own, role_id, expiration, deleted, created, updated, audit
oaResourceprofile_type	id, name, description, type, related_table, related_id, active, external_id, deleted, created, updated, audit
oaUser	cv_attachment_id
oaResourceAttachment	id, userid, attachment_id, type, latest_attachment_id, created, updated
oaAccountingPeriod	id, name, start_date, end_date, period_date_how, period_date, current_period, default_period, notes, active, created, updated
oaRevenue_recognition_rule	project_billing_ruleid

- Error codes and related information was added for error codes 960, 961, 962, 963, and 964.

Features for April 15, 2017

- Enabled delete support for oaCategory_1. (Interim change)

- Enabled delete support for oaCategory_2. (Interim change)
- Enabled delete support for oaCategory_3. (Interim change)
- Enabled delete support for oaCategory_4. (Interim change)
- Enabled delete support for oaCategory_5. (Interim change)
- Enabled delete support for oaCostcenter. (Interim change)
- Enabled delete support for oaRequest_item. (Interim change)

Expose Objects and Fields

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaPurchase_item	project_taskid
oaProject	main_contactid
oaExpensePolicy	id, customerid, projectid, description, deleted, created, updated, audit, all_items_allowed
oaExpensePolicyItem	id, expense_policyid, itemid, price_max, price_fixed, currency, deleted, created, updated, audit
oaAttributeDescription	id, resourceprofile_typeid, attributeid, description, deleted, created, updated, audit
oaAttachment	size

- Error codes and related information was added for Add/Modify error codes 899, 900, 947, 948, 949, 950, and 951.
- Added note to clarify that **limit** attribute limits projects rather than project tasks when using read method with oaProjecttask Complex Type. See oaProjecttask.
- Added note to clarify oaTask **loaded_cost** and **project_loaded_cost** default functionality, and corrected their descriptions.
- Removed “oaFormPermissionField”.

Features for October 15, 2016

- The following methods were added: approve, reject, unapprove
- The following complex types were added: ApproveRequest, ApproveResult, RejectRequest, RejectResult, UnapproveRequest, UnapproveResult

Expose Objects and Fields

The following complex types and fields were exposed:

Complex Type	Fields Exposed
oaProjectBudgetGroup	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

Complex Type	Fields Exposed
oaProjectBudgetRule	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
oaProjectBudgetTransaction	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
oaApprovalLine	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
oaProjecttask	classification

Features for April 16, 2016

The following complex type was added: oaRole

Features for October 17, 2015

- Error code and related information was added for MakeURL error code 943. Project names must be unique by customer.
- Added language clarifying that submit is for envelopes, invoices, and timesheets.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaPaymenttype	default_status, default_payment_type
oaSlip	skip_recognition
oaTaskAdjustment	created, id, new_taskid, new_timesheetid, old_taskid, old_timesheetid, updated

Features for April 18, 2015

- The following complex type was added: oaBooking_request

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaProject	rate_cardid
oaAgreement, oaBookingType, oaCategory, oaCategory_1, oaCategory_2, oaCategory_3, oaCategory_4, oaCategory_5, oaContact, oaCostcenter, oaCustomer, oaCustomerpo, oaDepartment, oaItem, oaPayrolltype, oaProject, oaProjectstage, oaProjecttask_type, oaTimetype, oaUser, oaVendor	picklist_label

Features for October 18, 2014

- The following complex types were added: oaItemToUserLocation and oaUserLocation

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaBooking	source_booking_id
oaProjectbillingrule	assigned_user
oaTicket	user_locationid

Features for May 17, 2014

- The following complex types were added: oaResourceRequest, oaResourceRequestQueue, oaResourcesearch, and oaWorkspace.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaAttachment	ownerid, is_a_folder, owner_type, name

Features for February 15, 2014

The following fields were exposed:

Object	Fields Exposed
oaAddress	contact_id

Features for November 16, 2013

- The following complex type was added: oaBillingSplit.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaApprovalProcess	externalid
oaUser	addr_id
oaCustomer	billing_addr_id, contact_addr_id, addr_id
oaCompany	addr_id
oaLoadedCost	externalid
oaBookingByDay	userid
oaProjectbillingtransaction	customerpoid, cost_centerid, timetypeid, customerid, agreementid, payroll_typeid
oaSlipProjection	projecttask_typeid, cost_centerid, acct_date, job_codeid
oaAddress	id
oaInvoice	submitted, approved
oaContact	exported, addr_id
oaReimbursement	userid, audit
oaVendor	addr_id

Features for August 17, 2013

- Added restriction on reading oaRevenueProjection. This complex type cannot be read while projections are running.
- Added oaPendingBooking and oaProjectAssignmentProfile.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaProjecttaskassign	project_assignment_profile_id, pending_booking_id, booking_id
oaBooking	project_assignment_profile_id

Features for May 18, 2013

- The following datatypes were added: oaBookingByDay and oaRevenueProjection.
- New **calendar—user** argument for the makeURL method.

Expose Datatypes and Fields

The following fields were exposed.

Object	Fields Exposed
oaProjectbillingtransaction	slip_stage_id
oaSlip	originating_id

Features for March 16, 2013

Expose Datatypes and Fields

The following fields were exposed.

Object	Fields Exposed
oaBooking	date_approved, date_submitted, approval_status

Features for January 19, 2013

Custom fields associated with oaBudget may be requested using the read method.

Features for November 17, 2012

- Provide support for "Require use of expense type price on receipts" option for Android devices. See **cost_is_fixed** in oaItem.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaItem	cost_is_fixed

Features for July 14, 2012

- Allow the setting of the "Notify requester when booking is modified" field on the booking form through SOAP API. See **notify_owner** in oaBooking.
- Added error code and related information to Add/Modify error code **885**. Force error on bad date in Purchase item import. Mandatory date is missing in purchase item.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaBooking	notify_owner
oaProjectbillingrule	exclude_non_billable_task
oaRevenue_recognition_transaction	portfolio_projectid
oaSlip	portfolio_projectid

Features for May 12, 2012

- Expanded the definition of the **limit** attribute on the read method ReadRequest argument.
- Added a reference for an internal switch to oaForexInput. You can have an internal switch enabled in your account for user defined reporting currencies.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaProject	portfolio_projectid, is_portfolio_project

Features for March 17, 2012

- Added a “generic” attribute for read commands. By default, the API returns regular users. When you add the generic attribute, the read request returns generic users.
- Added references for internal switches that affect the behavior of the API for the following complex types: oaEnvelope, oaInvoice, oaPurchase_item, oaSlip, oaTicket, and oaTimesheet.
- Added error code and related information for MakeURL error code 941. Reject oaUser add/modify requests that contain invalid time zone identifiers.
- Added clarifying information to Add/Modify error code 821. While it is returned when a timesheet cannot be modified because it was already submitted, it is also returned when other conditions exist. See oaTimesheet.
- Added error code and related information for Custom Field error code 1106. One or more inline custom fields failed to be updated

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaCustomer	created, updated, billing_code

Features for January 21, 2012

- The following complex type was added: oaSchedulebyday. Custom fields associated with oaSchedulebyday may be requested using the read method.
- Custom fields associated with oaPurchaseorder may now also be requested using the read method.
- Custom fields associated with oaRequest_item may now also be requested using the read method.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaSchedulebyday	id, date, user_id, hours, base_hours, target_hours, target_base_hours, created, updated

Features for November 19, 2011

The following complex type was added: oaRevenueStage

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaBooking	locationid
oaRevenueStage	id, name, revenue_stage_type, created, updated
oaRevenue_recognition_transaction	is_from_open_stage

Features for September 17, 2011

- The following complex type was added: oaUserWorkschedule.
- SOAP API handles all existing task rounding rules.
- Error code and related information was added for Add/Modify error code 882.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaInvoice	credit_rebill_status, original_invoiceid
oaProject	rv_approver, rv_approvalprocess
oaProjectbillingrule	category_1id, category_2id, category_3id, category_4id, category_5id

Object	Fields Exposed
oaRevenueContainer	project_billing_rule_filter, category_1id, category_2id, category_3id, category_4id, category_5id
oaRevenue_recognition_rule_amount	category_1id, category_2id, category_3id, category_4id, category_5id
oaSlip	ref_slipid
oaTaskTimecard	category_1id, category_2id, category_3id, category_4id, category_5id
oaUserWorkschedule	id, name, userid, use_this_schedule, account_workscheduleid, workdays, workhours, created, updated

Features for July 16, 2011

- The following arguments /options were added to the makeURL method: view-invoice, dashboard-project, grid-timesheet, report-timesheet.
- Custom fields associated with oaPayment may now be requested using the read method.
- Custom fields associated with oaUser may now be returned for regular as well as generic users.

Features for May 14, 2011

- The following complex type was added: oaRevenueContainer. Enabled support for read including oaCustField and update of externalid only.
- API will not allow negative quantity on non-PO purchase items.
- Fixed an issue where email field was reset on oaContact update when value was not provided.
- Added parentid field to oaAttachment.
- Error codes and related information were added for Add/Modify error codes 880 and 881.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaAttachment	parentid
oaProjecttask	default_category_1, default_category_2, default_category_3, default_category_4, default_category_5
oaRevenueContainer	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

Features for March 19, 2011

- oaTargetUtilization records can now be added for inactive users.

- When a new customer is created and payment terms are not explicitly specified, default payment terms are used.
- Job_codeid can have a value of 0 in modify operations on oaProjectassign and oaProjecttaskassign.
- Short PO Purchase_item import sets the date on fulfillments to date_fulfilled (if present) or to today's date.
- Error codes and related information were added for API Login error code 555 and MakeURL error codes 914 - 915

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaCustField	never_copy
oaTask	category_1id, category_2id, category_3id, category_4id, category_5id

Features for January 22, 2011

- Enabled custom equal to support for oaPaymenttype.
- Enabled modify and delete support for oaAttachment.
- Enabled delete support for oaBooking.
- Error codes and related information were added for Add/Modify error codes 878 - 879 and Custom Field error code 1105.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaRevenue_recognition_rule_amount	cost_center_id
oaTask	acct_date
oaTicket	externalid
oaTimesheet	acct_date

Features for November 20, 2010

- The following complex type was added: oaProjectgroup.
- The following complex types were changed: oaAttribute and oaFieldAttribute.
  - oaAttribute, a table that describes an attribute, was modified to include the following children: ID, name, attribute_setid, updated, created, and notes.
  - oaFieldAttribute was added and has two children: name and value. It is used to lookup foreign keys. See Example II. modify using external_id as foreign key lookup field C# and Example II. externalid as foreign key lookup field Java.

- Enabled add, modify, and delete support for oaAgreement_to_project.
- Enabled delete support for oaEntitytag.
- Error codes and related information were added for Add/Modify error codes 876 - 877, Make URL error codes 936 - 938, and Custom Field error code 1104.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaAttribute	id, name, attribute_setid, updated, created, and notes.
oaFieldAttribute	name and value
oaProjectassign	job_codeid
oaProjectbillingrule	daily_rate_multiplier and job_code_filter
oaProjectbillingtransaction	job_codeid
oaProjectgroup	id, attributes, assigned_users, created, updated, name, notes, and active
oaProjecttaskassign	job_codeid
oaRevenueContainer	asb_which_slips
oaUprate	job_codeid

Features for September 18, 2010

- The following complex types were added: oaAgreement_to_project, oaAttributeset, and oaIssueStatus
- Enabled read support for oaAttribute.
- The following **filter** was added: approved-revenue-recognition-transactions.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaAgreement_to_project	agreementid, attribute, customerid, projectid, active, created, and updated
oaAttributeset	id, name, attribute, notes, created, and updated
oaBooking	job_code_id
oaCustomer	sold_to_contact_id
oaIssueStatus	id, name, attribute, active, created, and updated
oaRevenue_recognition_transaction	project_billing_rule_id, job_code_id, rate, decimal_hours, hour, minute, revenue_containerid, revenue_stageid, originatingid, and offsetsid
oaSlip	projecttask_type_id, job_code_id, and payroll_type_id

Features for July 17, 2010

- The following complex types were added: oaCategory_1, oaCategory_2, oaCategory_3, oaCategory_4, and oaCategory_5.
- Enabled custom equal to support for oaRevenue_recognition_transaction.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaBooking	starttime and endtime
oaCategory_1	id, name, code, externalid, active, created, updated, and notes
oaCategory_2	id, name, code, externalid, active, created, updated, and notes
oaCategory_3	id, name, code, externalid, active, created, updated, and notes
oaCategory_4	id, name, code, externalid, active, created, updated, and notes
oaCategory_5	id, name, code, externalid, active, created, updated, and notes
oaRevenue_recognition_transaction	category_1id, category_2id, category_3id, category_4id, and category_5id

Features for May 15, 2010

Error codes and related information were added for Add/Modify error codes 871 - 875.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaAgreement	acct_date
oaCustomerpo	acct_date
oaProject	attachmentid

Features for March 20, 2010

- Lookup and Modify Custom Fields without using “custom equal to” Method
  - Added the ability to lookup and modify custom fields without having to use the “custom equal to” method. This is now the default behavior when reading SOAP objects.
  - Custom fields may now also be modified inline in a single modify request with other fields. To enable this behavior, supply the "update_custom" attribute in your modify request and set it to 1. (This attribute is not necessary when performing add requests.)
- Use Mulitple Argument Objects in a Single ReadRequest

- Added the ability to mix “equal to” and “not equal to” argument objects in a single ReadRequest.
- Multiple argument objects can be supplied in one ReadRequest to create an AND/OR filtering logic. To use this feature:
  - Modify the method parameter to include multiple “equal to” and “not equal to” methods as desired, separated by commas. For example: “equal to, not equal to”.
  - Next, supply an equal number of argument objects to filter on.
  - Additionally, you may precede each method by an “and” or an “or” operator. For example: “equal to, or equal to”. If no operator is supplied, a logical AND relationship is assumed.

    Note: This feature supports only one level of logical operators does not support nesting of \ operators (i.e., equal to and (equal to or equal to).

- Lookup Foreign Keys using oaFieldAttribute - oaFieldAttribute, a new complex type, was created and it has two children: name and value. It gives you the option of adding attributes to all other complex types except: oaAddress, oaFieldAttribute, oaDate, and oaModule. You can use externalid fields as a foreign key, allowing you to add, create User, modify and upsert records in a single step instead of querying a record to first obtain the internal ID. To use an externalid field as a foreign key, set the ID field to the externalid field value instead of internal ID field value. Then, create an oaFieldAttribute object for each field that needs to be overridden and pass those as a collection of oaFieldAttribute objects. See code examples: Example II. modify using external_id as foreign key lookup field C# and Example II. externalid as foreign key lookup field Java.

  Warning: The wsdl file definition for oaAttribute is changed to oaFieldAttribute. If you were using oaAttribute to lookup internal ID values from externalid values on related records, the old binaries referencing the old name will still work. However, if your code is generated using the newer wsdl file, you need to change the objects to oaFieldAttribute to continue using this feature.

- Enable Mobile Services - The internal switch to Enable mobile services is now required for all of the following add-on services: OffLine, iPhone, Blackberry, Pocket PC, and Palm.
- Programming Fixes, Checks, and Validations
  - Enabled “custom equal to” method for Fulfillment object.
  - Established imported and exported as required fields on import for oaImportExport.
  - Allow more than one node per hierarchy on oaUser project_access_node.
  - Updating oaProjectBillingRule does not require that cost_centerid to be populated.
  - Added Add and Modify methods to oaSchedulerequest.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaActualcost	id, name, userid, date, period, currency, cost, cost_typeid, is_accrual, externalid, notes, created, updated
oaAttachment	attachmentid
oaCostcategory	id, name, active, notes, created, updated, externalid
oaCosttype	id, name, active, notes, created, updated, externalid

Object	Fields Exposed
oaEnvelope	currency_exchange_intolerance
oaFieldAttribute	name, value
oaRepeat	id, frequency, every, end, occur_number, how_end, exclude_dow, created, updated
oaRevenueContainer	cost_centerid
oaTicket	attachmentid, currency_exchange_intolerance

Note: The attributes field is added in many complex types. Excluded are oaAddress, oaFieldAttribute, oaDate, and oaModule. Refer to OpenAir Complex Types for field names and descriptions.

Features for January 23, 2010

- Enabled read support for oaReport.
- Enabled modify and add support for oaHierarchy.
- Enabled modify, add, and delete support for oaHierarchyNode.
- Fixed issue where user workschedule was not being set when user is created through SOAP.
- Fixed the logic that requests deleted records, applying the not-exported filter against a deleted import_export record.
- Changed the way we handle invalid utf8 characters: strip them out completely instead of converting them to decimal numbers. Removed more utf8 encoding errors in the server log for accounts not configured for utf8.
- Adjusted createUser logic to more closely follow UI logic when setting user.name field.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaBooking	owner_id
oaCompany	workscheduleid (read-only field)
oaHierarchy	externalid
oaHierarchyNode	available_as_column, externalid, primary_dropdown_filter, primary_user_filterset
oaReport	id, userid, name, type, thin_client_context, date_created, email_report, relatedid, created, updated

Features for November 21, 2009

- Set User Workschedule - Added the ability to set the user workschedule via the User API object or during user account creation. See oaUser.

- Check for valid project and customer on slip add.
- Allow use of default filtering mechanism when reading project_task.
- Made sure duplicate import_export records are never created, specific to upsert or add calls.
- Modified createUser routine to return error codes.
- Modified user tag update feature to ensure tags receive a valid start_date.
- Allow 0 offset in limit clause.
- Allow modification of an entity tag for inactive users.

Expose Objects and Fields

The following fields were exposed.

Object	Fields Exposed
oaEnvelope	attachmentid
oaProject	pm_approver_1, pm_approver_2, pm_approver_3, payroll_type_filter
oaResourceprofile	externalid
oaResourceRequest	externalid
oaUser	update_workschedule, is_user_schedule, workschedule_workdays, workschedule_workhours