



# OpenAir

## XML API Reference Guide

Copyright © 2013, 2018, Oracle and/or its affiliates. All rights reserved.

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

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

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

U.S. GOVERNMENT END USERS: Oracle programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, delivered to U.S. Government end users are "commercial computer software" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the programs, including any operating system, integrated software, any programs installed on the hardware, and/or documentation, shall be subject to license terms and license restrictions applicable to the programs. No other rights are granted to the U.S. Government.

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

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

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

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

# Table of Contents

Introduction to OpenAir XML API .....	1
Technology .....	1
Target Audience .....	1
Overview .....	1
Definitions .....	2
Presentation of XML .....	2
Authorization and Command Overview .....	3
Naming Conventions for Objects and Commands .....	3
Error Handling .....	4
Connecting to the API .....	6
Namespaces .....	6
Connecting to the API .....	6
Limits .....	7
Internationalization and Character Sets .....	8
XML Commands .....	9
Time .....	9
Read .....	9
Read, all .....	12
Read, equal to .....	13
Read, not equal to .....	16
Read, custom equal to .....	17
Read, user .....	17
Read, project .....	18
Read, not exported .....	18
Report .....	19
Add .....	19
Delete (id) .....	21
Modify (id) .....	21
Modify (Logo) .....	23
Modify, custom equal to .....	24
Submit .....	24
CreateAccount .....	25
CreateUser .....	25
Auth .....	26
RemoteAuth .....	26
MakeURL .....	27
Whoami .....	29
Version .....	30
Approve .....	30
Reject .....	30
Unapprove .....	31
ModifyOnCondition .....	31
Custom Fields .....	33
Requesting Custom Fields for a Datatype .....	33
Reading Custom Field Values Inline with Native Fields .....	33
Reading Custom Field Values in a Separate Request .....	33
Adding/Modifying Records with Inline Custom Field Values .....	34
Modifying Records to Set Custom Field Values .....	34
XML Datatypes .....	35
Actualcost .....	36
AccountingPeriod .....	37
Address .....	37
Agreement .....	38

Agreement_to_project .....	39
Approval .....	39
ApprovalLine .....	39
Approvalprocess .....	40
Attachment .....	41
Attribute .....	41
AttributeDescription .....	42
Attributeset .....	42
BillingSplit .....	42
Booking .....	43
BookingByDay .....	44
BookingType .....	44
Booking_request .....	44
Budget .....	45
BudgetAllocation .....	46
Category .....	46
Category_1 .....	47
Category_2 .....	47
Category_3 .....	48
Category_4 .....	48
Category_5 .....	49
Ccrate .....	49
Company .....	50
Contact .....	50
Costcategory .....	51
Costcenter .....	51
Costtype .....	51
Currency .....	52
Currencyrate .....	52
CustField .....	52
Customer .....	54
Customerpo .....	56
Customerpo_to_project .....	56
CustomerProspect .....	57
Date .....	58
Deal .....	58
Dealcontact .....	59
Dealschedule .....	59
Department .....	59
Entitytag .....	60
Envelope .....	60
Error .....	61
Estimate .....	62
Estimateadjustment .....	62
Estimateexpense .....	62
Estimatelabor .....	63
Estimatemarkup .....	63
Estimatephase .....	64
Event .....	64
ExpensePolicy .....	65
ExpensePolicyItem .....	65
Filter .....	65
Filterset .....	66
Flag .....	66
ForexInput .....	66

FormPermissionField .....	67
Fulfillment .....	67
Hierarchy .....	68
HierarchyNode .....	68
History .....	69
ImportExport .....	69
Invoice .....	70
InvoiceLayout .....	71
Issue .....	71
IssueCategory .....	72
IssueSeverity .....	72
IssueSource .....	73
IssueStage .....	73
IssueStatus .....	73
Item .....	74
ItemToUserLocation .....	74
Jobcode .....	75
Leave_accrual_rule .....	75
Leave_accrual_rule_to_user .....	76
Leave_accrual_transaction .....	76
LoadedCost .....	77
Login .....	77
Module .....	78
Notes .....	78
Payment .....	78
Paymentterms .....	79
Paymenttype .....	79
Payrolltype .....	79
PendingBooking .....	80
Preference .....	81
Product .....	81
Project .....	82
Projectassign .....	86
ProjectAssignmentProfile .....	86
Projectbillingrule .....	87
Projectbillingtransaction .....	89
ProjectBudgetGroup .....	90
ProjectBudgetRule .....	91
ProjectBudgetTransaction .....	92
Projectgroup .....	93
Projectlocation .....	94
Projectstage .....	94
Projecttask .....	95
ProjecttaskEstimate .....	97
Projecttask_type .....	98
Projecttaskassign .....	98
Proposal .....	99
Proposalblock .....	100
Proxy .....	101
Purchase_item .....	101
Purchaseorder .....	103
Purchaser .....	104
Purchaserequest .....	104
Ratecard .....	105
RateCardItem .....	105

Reimbursement .....	106
Repeat .....	106
Report .....	107
Request_item .....	107
ResourceAttachment .....	108
Resourceprofile .....	109
Resourceprofile_type .....	109
ResourceRequest .....	109
ResourceRequestQueue .....	110
Resourcearch .....	111
RevenueContainer .....	112
RevenueProjection .....	113
Revenue_recognition_rule .....	115
Revenue_recognition_rule_amount .....	117
Revenue_recognition_transaction .....	117
RevenueStage .....	119
Role .....	119
Schedulebyday .....	119
Scheduleexception .....	120
Schedulerequest .....	120
Schedulerequest_item .....	121
Slip .....	122
SlipProjection .....	123
Slipstage .....	125
TagGroup .....	125
TagGroupAttribute .....	125
TargetUtilization .....	126
Task .....	126
TaskTimecard .....	127
TaxLocation .....	128
TaxRate .....	128
Term .....	129
Ticket .....	129
Timecard .....	130
Timesheet .....	131
Timetype .....	132
Todo .....	133
Uprate .....	133
User .....	134
UserLocation .....	140
UserWorkschedule .....	140
Vendor .....	141
Viewfilter .....	142
Viewfilterrule .....	142
Workspace .....	143
Workspacelink .....	143
Workspaceuser .....	143
Setting Application Switches Via the API .....	145
Customizing the Application .....	146
Other Features .....	150
Filters .....	150
Read, user Command .....	150
Read, project Command .....	150
Hints .....	151
IDs .....	151

Remaining Limit .....	152
Code Examples .....	153
Basic Example .....	153
Intermediate Example .....	154
Advanced Example .....	155
Appendix A Error Code Listing .....	157
Error Responses .....	157
Error Codes .....	158
Appendix B Simple client (in perl) .....	170
Appendix C OpenAir Data Dictionary .....	172
Customer Table .....	172
Appendix D Best Practices .....	178
Build the API Integration .....	178
Optimize the API Integration .....	179
Maintain the API Integration .....	180
Troubleshooting .....	182
New Features .....	183
Features for April 14, 2018 .....	183
Features for October 14, 2017 .....	183
Features for April 15, 2017 .....	183
Features for October 15, 2016 .....	184
Features for April 16, 2016 .....	185
Features for October 17, 2015 .....	185
Features for April 18, 2015 .....	185
Features for October 18, 2014 .....	186
Features for May 17, 2014 .....	186
Features for February 15, 2014 .....	186
Features for November 16, 2013 .....	187
Features for August 17, 2013 .....	187
Features for May 18, 2013 .....	188
Features for March 16, 2013 .....	188
Features for January 19, 2013 .....	188
Features for November 17, 2012 .....	188
Features for July 14, 2012 .....	189
Features for May 12, 2012 .....	189
Features for March 17, 2012 .....	189
Features for January 21, 2012 .....	190
Features for November 19, 2011 .....	190
Features for September 17, 2011 .....	191
Features for May 14, 2011 .....	192
Features for March 19, 2011 .....	192
Features for January 22, 2011 .....	193
Features for November 20, 2010 .....	193
Features for September 18, 2010 .....	194
Features for July 17, 2010 .....	194
Features for May 15, 2010 .....	195
Features for March 20, 2010 .....	195
Features for January 23, 2010 .....	196
Features for November 21, 2009 .....	197

# Introduction to OpenAir XML API

OpenAir provides OpenAir XML API 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 do not need direct database access, and third party applications indirectly supported through OpenAir. Before you begin using this service, we recommend that you review [Appendix D 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.

## Technology

OpenAir XML API is based on industry standard components: HTTPS (Secure Hypertext Transfer Protocol) and XML (Extensible Markup Language).

Standard Name	Web Site Reference
RFC2660 The Secure HyperText Transfer Protocol	<a href="http://rfc.net/rfc2660.html">http://rfc.net/rfc2660.html</a>
Extensible Markup Language (XML) 1.0 (Fourth Edition)	<a href="http://www.w3.org/TR/xml/">http://www.w3.org/TR/xml/</a>

Much of the work in implementing an API-aware client can be done using off-the-shelf parts. HTTPS is used for the transport layer, providing an easy avenue to encrypt transactions. XML is used both for the command syntax (asking for information) and for the actual data content (packaging the requested information). The HTTPS request is presented in a PUT or POST request, and the response is the resulting document.

XML is essentially a subset of SGML (Standard Generalized Markup Language) and, unlike HTML, has the advantage of being able to handle user-defined tags. XML has a context-dependent, nested structure. XML tags provide a context for the data contained within each of them. This allows you to send meaningful information easily and quickly over the World Wide Web. XML is particularly suited to the transfer of data to and from databases. For the information to be useful, it must be properly identified.

Since the OpenAir 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 XML commands and datatypes that are meaningful to an OpenAir database. Through the use of commands and datatypes provided by the API and by limiting the values that can be stored in each datatype, 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

- **Namespaces and Connecting to the API** — addresses how to access namespaces and shows you how to connect to the OpenAir API.



- **XML Commands** — lists XML commands for each possible method. Provides associated code, results returned, and additional information and examples.
- **Custom Fields** — introduces custom fields and provides information for requesting custom fields for a datatype, reading custom field values, and modifying records to set custom field values.
- **XML Datatypes** — provides the OpenAir XML Datatypes with a description of the structure and sub-structure.
- **Setting Application Switches Via the API** — describes Company and User switches you can set using the API.
- **Customizing the Application** — describes the options available for customizing OpenAir.
- **Other Features** — lists ways of limiting records returned through the use of filters and IDs. Also describes how to add hints to the application.
- **Code Examples** — provides code examples for connecting to the server, receiving information, and creating a user account.
- **Appendix A - Error Code Listing** — identifies common errors and associated codes.
- **Appendix B - A Simple Client** — provides a simple client example to demonstrate exchanges to and from the API server.
- **Appendix C - OpenAir Data Dictionary** — explains database fields and how they relate to datatypes using the Customer table as an example. For more data dictionary information, refer to [OpenAir Data Dictionary](#)
- **Appendix D - Best Practices** — provides a guide for preparing for and using the API.

## Definitions

- **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

## Presentation of XML

Although the XML actually used in the application does not contain any new lines or formatting, it is presented in an easier-to-read indented style in this document. Refer to the following example.

```
<First_level>
  <Second_level>
    <Third_level/>
    <Third_level>X</Third_level>
  </Second_level>
</First_level>
```

Throughout this document, elements are referred to as first-level, second-level, third-level, etc. This corresponds to how deeply the elements are nested in the XML as shown in the previous example.

## Authorization and Command Overview

All requests to the API take this general form:

- Authorization (login/pass)
- Ask (for data)
- Answer (with data)

Each of these requests consists of at least one command, and usually some data. The 'auth' and 'ask' portions are in an HTTP PUT or POST request to the server, the 'answer' is the resulting document returned from the request. Since we are using HTTP, each connection is isolated, and must go through authorization each time. This authorization consists of sending the server an XML data structure consisting of company name, user name, and user password. This is the same data users must enter to use the OA site proper.

The Ask portion of the interaction can contain zero-to-many commands (although zero isn't very useful). These commands are used to get updated information, to update information on the server, or even just to ask for the time. The basic commands are Read, Modify, Add, and Delete. Each of these can be applied to any of the XML structure types. Refer to [XML Datatypes](#) for more information. Most commands only serve one function, but Read has several methods including 'newer than' and 'equal to'. Refer to [XML Commands](#) for more information.

The Answer portion of the exchange contains success/failure status for all commands in the Ask portion, and data for any of the commands that had success. The data is returned in the same order it was asked for, separated out by the Ask command that generated the data.

## Naming Conventions for Objects and Commands

XML is used for both the command syntax and the data exchanged. The basic layout for a request is as follows:

```
<request>
  <Command1>
    <Data>
  </Command1>

  <Command2>
    <Data>
  </Command2>

  <Command3>
    <Data>
  </Command3>
  ...
  <CommandN>
    <Data>
  </CommandN>
</request>
```

The server response looks similar:

```
<response>
```

```

<Command1 status="0">
  <Data>
</Command1>

<Command2 status="0">
  <Data>
</Command2>

<Command3 status="1"> </Command3>
...
<CommandN status="0">
  <Data>
</CommandN>
</response>

```

The request element is a first level element, the second level elements are commands, and the third level elements are data.

To make it more readable when there isn't any pretty indentation, the naming structure of the XML commands and data is:

1. request, the basic wrapper for all transactions, never capitalized.
2. Command, the second level element, always capitalized.
3. Data, XML structures are always capitalized. They represent a package or group of data (e.g., address or person). The elements they contained in that structure are all lower case.

In the following example, "Neighbor" contains the elements "name" and "billing\_address", which are both lower case. "name" contains a simple string, "billing\_address" contains an "Address" XML structure.

```

<Neighbor>
  <name>
    Bob Roberts
  </name>
  <billing_address>
    <Address>
      <city>Boston</city>
      <state>MA</state>
      <zip>02111</zip>
    </Address>
  </billing_address>
</Neighbor>

```

## Error Handling

Errors are returned via the "status" attribute of command responses. All errors are numbered. For an error code listing, refer to [Appendix A Error Code Listing](#). If the status is success or "0", then for most operations the server will return data. The exception is delete operations. In these cases the server just responds with success or failure.

Errors are also a valid datatype in the XML data set. Instead of looking up an error in the Appendix, you could use the API to query the text translation. Of course, this only works if the problem isn't that requests from the client were badly formed.

If the server encounters badly-formed XML or an incomplete request, it responds with something generic such as:

```
<response status="1">Badly formed XML, parsing aborted</response>
```

# Connecting to the API

This chapter addresses namespaces and shows you how to connect to the API.

## Namespaces

Contact the OpenAir Support Department or your account representative to request API access. See [Troubleshooting](#) 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.

Namespaces, which are used to group accounts on the OpenAir system, can be used for multiple accounts. A Company ID, however, is unique within a namespace. There will never be two identical Company IDs within the same namespace.

## Connecting to the API

The request/response of the API is done through an HTTP(S) PUT or POST request to the API server. There are many libraries that support one or both of these types. (Internally they are almost identical). The URL that you send the POST/PUT to is: <https://www.openair.com/api.pl>

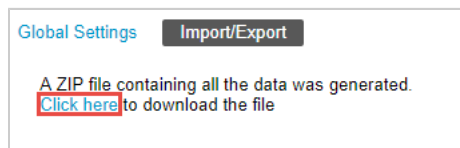
The content that you POST/PUT to the server is your formatted request. The resulting document is the API server's response.

We highly recommend that you use secure communications for all integrations by connecting over SSL (https). Client applications typically do not need to provide a SSL certificate with each HTTPS request. However, you can download OpenAir's public certificate, which is issued by VeriSign, by connecting to <https://www.openair.com> using any browser.

XSD schema files can be downloaded from the Administration page within OpenAir.

### To export the XSD schema files:

1. Go to Administration Global Settings > Account and click **Integration: Import/Export**.
2. In the Import/Export screen, click the **XSD schema files** link. OpenAir will create a ZIP file containing the generated data.
3. Click the **Click here** link to download the ZIP file with your data.



## Initial API Request

The initial request to the API server should include the following attributes:

- XML header: `<?xml version="1.0" encoding="UTF-8" standalone="yes"?>`
- `API_version`: (this will be "1.0")
- `client`: the type of client you are using to connect to the API
- `client_ver`: the version number of the client
- `namespace`: the namespace assigned to you by OpenAir (typically "default")
- `key`: the authentication key used with the namespace

For example, a request for the time on the server could look like this:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<request API_version="1.0" client="test app" client_ver="1.1"
namespace="default" key="0123456789">
  <Time/>
</request>
```

## Limits

Currently there are four types of usage limits that are enforced in the OpenAir XML API.

- There is a limit of 1000 records that can be requested at one time. Each request with the "Read" command must contain the "limit" attribute to limit the amount of records being returned. If Read is used without the "limit" attribute, an error will be returned. The "limit" attribute also allows you to request records in batches. See the **limit** attribute in [Attributes](#) for more information.
- There is a limit of 1000 objects any method can accept, so if you need to use add, 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](#) command. This command gives you the status of your 24-hour limit. Insert this command at various times during your integrations to see where you are sending the highest volume of requests.
- Contact OpenAir 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.

After a frequency limit is reached for either daily transactions or a 60-second interval, our web servers will respond with a 556 error code to the login operation 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 [Appendix D Best Practices](#).

Please contact OpenAir Support (see [Troubleshooting](#)) 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 Account Manager and ask about a Professional Services engagement to help you reduce your API requests.

## Internationalization and Character Sets

OpenAir uses UTF-8 encoding to store and display characters in our application. Please ensure that you specify `<?xml version="1.0" encoding="UTF-8" standalone="yes"?>` at the start of your XML API request to ensure any non-English characters are transmitted and stored properly.

# XML Commands

The following are XML commands with supported methods. Click on the command to review the associated code, results returned, and additional information and examples.

XML Commands		
Time	Report	CreateUser
Read	Add	Auth
Read, all	Delete (id)	RemoteAuth
Read, equal to	Modify (id)	MakeURL
Read, not equal to	Modify (Logo)	Whoami
Read, custom equal to	Modify, custom equal to	Version
Read, user	Submit	Approve
Read, project	CreateAccount	Reject
Unapprove	ModifyOnCondition	

In the examples provided, datatype is any XML Datatype that contains an id element. Refer to [XML Datatypes](#) for field names and definitions as well as a list of supported commands for each datatype.

## Time

The Time command returns the current time on our servers.

```
<Time/>
```

Returned: A Date record of the current server time.

## Read

Use the read command to retrieve data from OpenAir. Use a variety of methods, fields, and attributes to gather the information you need. Each is described as follows.

## Methods

Use one of the following methods.

Method	Result
all	Returns all records. Use this cautiously as too many records may be requested for the server or client to handle.
equal to	Returns records that are equal to the field value(s) passed in for the datatype specified.
not equal to	Returns records that are not equal to the field value(s) passed in.
custom equal to	Allows one to read custom field values for a particular record.



Method	Result
user	Returns records for the specified user
project	Returns records for the specified project
not exported	Returns records not-yet exported
order	<p>Use to specify a valid column to order by. The default order is ascending. For example:</p> <pre>//Get the 10 newest objects in descending order order="created,desc" limit="10"  //Get the latest 10 updated objects in ascending order: order="updated,asc" limit="10"  //You can omit the "asc" parameter, as ascending order is the default: order="updated" limit="10"</pre>


## 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'	<p>Restricts the number of records returned.</p> <p>Single number value: "1", "500", "1000" - simply restricts the number of records returned.</p> <p>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.</p> <p>To request data in consecutive batches, only the first part of the limit attribute should be incremented - "0,1000", "1000,1000", "2000,1000", etc.</p> <p>Sequence requests should be submitted until the result comes back empty or has less items than 1000.</p>
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.

Attribute Name	Value	Result
		 <b>Note:</b> 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.
enable_custom	1	Custom fields to be included inline with other native fields
filter_(*list of possible values) 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.
	* 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.
	* approved-revenue-recognition-transactions	Returns only revenue recognition transactions belonging to approved revenue_container records.

Attribute Name	Value	Result
	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 Date datatype in the objects collection. To compare date fields other than 'updated', add the following additional attribute information: name="field" value="[some date field]"

**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) respectively. You should 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 timesheets only: filter=" newer-than, older-than, approved-timesheets ".

## Read Examples

Refer to the following Read examples using different methods, fields, attributes, and filters. They include:

- [Read, all](#)
- [Read, equal to](#)
- [Read, not equal to](#)
- [Read, custom equal to](#)
- [Read, user](#)
- [Read, project](#)
- [Read, not exported](#)

## Read, all

### Example 1

The Read, all command returns all records for the datatype specified.

```
<Read type="datatype" method="all" limit="1000"/>
```

Returned: A list containing all XML objects of the datatype you specified.

Use "Read, all" cautiously as too many records may be requested for the server or client to handle. It is better to define types, methods, fields, and attributes to limit the results that are returned. Limit attribute is required for all read requests, where maximum allowed limit is 1000. Refer to the following for more information.

## Example 2

To return all tickets that are in open envelopes:

```
<Read type="Ticket" method="all" filter="open-envelopes" limit="1000"/>
```

## Example 3

To request timesheet entries whose date field is within a specific one month range, filter by newer-than and older-than. Note that "limit" attribute is required as illustrated in [Example 6](#).

```
<Read type="Task" filter="newer-than,older-than"
field="date,date" method="all" limit="1000">
  <Date>
    <year>2011</year>
    <month>07</month>
    <day>01</day>
  </Date>
  <Date>
    <year>2011</year>
    <month>08</month>
    <day>01</day>
  </Date>
</Read>
```

**Note:** The only <Read/> method that is supported by the Filter datatype is "Read, all".

## Read, equal to

### Example 1

The Read, equal to command returns records equal to the value passed in for the datatype specified.

```
<Read type="datatype" method="equal to" limit="500">
  (One object of type 'datatype' goes here.)
</Read>
```

Returned: An object of the datatype with the field specified equal to the value passed in, or failed status if that record does not exist.

**Note:** As with the "Read, all" command:

- Using the CustomerProspect datatype returns both customer and prospect records, whereas if you use the Customer datatype, it only returns customer records.

- Inserting `<_Return><field> </_Return>` syntax before `</Read>` restricts the number of fields within records returned.
- Return only deleted fields by specifying that the attribute = "1"

## Example 2

To get all records with certain fields having a specific value, add the datatype to the request with the desired values specified on the properties. For example, return all Project records whose owner is user with internal ID 123 and that have tax location ID = 5.

```
<Read type="project" method="equal to" limit="1">
  <Project>
    <userid>123</userid>
    <tax_locationid>5</tax_locationid>
  </Project>
</Read>
```

## Example 3

To get all records with certain fields set to null, add the datatype to the read request with the desired property with no value provided. For example, return all Task records that have slip ID set to NULL and customer internal ID 5.

```
<Read type="Task" method="equal to" limit="1000">
  <Task>
    <slipid/>
    <customerid>5</customerid>
  </Task>
</Read>
```

## Example 4

To return a list of all approved envelopes pending reimbursement:

```
<Read type="Envelope" method="equal to"
filter="nonreimbursed-envelopes" limit="1000">
  <Envelope>
    <status>A</status>
  </Envelope>
</Read>
```

## Example 5

To return a list of time entries that have date value newer than the specified date and are logged against project with id 13, use the following syntax:

```
<Read type="Task" method="equal to" filter="newer-than"
field="date" limit="1000">
```

```

    <Date>
      <year>2012</year>
      <month>03</month>
      <day>25</day>
      <hour>11</hour>
      <minute>01</minute>
      <second>36</second>
    </Date>
    <Task>
      <projectid>13</projectid>
    </Task>
  </Read>

```

## Example 6

To restrict the number of fields within records returned, filter by field. Insert `<_Return><field></_Return>` syntax before `</Read>`. You can return only deleted fields by specifying that the attribute = "1".

```

<Read type="Slip" limit="0,1000" method="equal to">
  <Slip>
    <customerid>1</customerid>
    <projectid>1</projectid>
  </Slip>
  <_Return>
    <customerid/>
    <id/>
    <projectid/>
  </_Return>
</Read>

```

Returned: slips in customerid, id, and projectid fields. Return 1000 records starting with index 0.

## Example 7

To get all records with custom fields having a specific value, add the datatype to the request with the desired values specified on the properties. For example, return all Project records where myCustomField is equal to "756" and that have tax location ID = 5.

```

<Read type="Project" enable_custom="1" method="equal to"
limit="1">
  <Project>
    <myCustomField__c>756</myCustomField__c>
    <tax_locationid>5</tax_locationid>
  </Project>
</Read>

```

## Example 8

To request a currency rate for a base currency to a second currency, for a specified day, with a specified precision , use the Currencyrate complex type with the Read, equal to command. This example uses

U.S. dollars as the base currency, Euros as the second currency, September 1, 2016 as the specified day, and 10 as the specified precision.

```
<Read type="Currencyrate" method="equal to" limit="1000" filter="date-equal-to" field="date"
base_currency="USD" precision="10">
  <Currencyrate>
    <csymbol>EUR</csymbol>
    <type/>
  </Currencyrate>
  <Date>
    <year>2016</year>
    <month>09</month>
    <day>01</day>
  </Date>
</Read>
```

## Read, not equal to

### Example 1

The Read, not equal to command returns records not equal to the value passed in for the datatype specified.

```
<Read type="datatype" method="not equal to" limit="500">
  (One object of type 'datatype' goes here.)
</Read>
```

**Note:** As with the "Read, all" command:

- Using the CustomerProspect datatype returns both customer and prospect records, whereas if you use the Customer datatype, it only returns customer records.
- Inserting `<_Return><field> </_Return>` syntax before `</Read>` restricts the number of fields within records returned.
- Return only deleted fields by specifying that the attribute = "1"
- Note that "limit" attribute is required as illustrated in [Example 6](#).

### Example 2

To get all records with custom fields that do not have a specific value, add the datatype to the request with the desired values specified on the properties. For example, return all Project records where myCustomField is not equal to "756" and that have tax location ID that is not equal to 5.

```
<Read type="Project" enable_custom="1" method="not equal to"
limit="500">
  <Project>
    <myCustomField__c>756</myCustomField__c>
    <tax_locationid>5</tax_locationid>
  </Project>
```

```
</Read>
```

## Read, custom equal to

Use the Read, custom equal to command to return the custom field values for the record specified in the <Read/> request. For a list of associated datatypes that can be used, refer to the [CustField datatype Association table](#). Refer to the example that follows.

```
<Read type="datatype" method="custom equal to"
  field_names="custom_field_name1,custom_field_name2" limit="500">
  <datatype>
    <id>X</id>
  </datatype>
</Read>
```

**Note:** "custom\_field\_name1" is an optional attribute which allows you to specify custom field names, such as those that appear in your OpenAir account, to be returned. If omitted, all custom field values for the record are returned.

Returned: The custom field records of an object of datatype with id equal to the id passed in, or failed status if that id doesn't exist. Note that "limit" attribute is required as illustrated in [Example 6](#).

## Read, user

Use the Read, user command to restrict the records returned in a <Read/> request.

```
<Read obtype="ObjectName" method="user" limit="500">
  <User>
    <id>X</id>
  </User>
</Read>
```

Returned: A list of "ObjectName" XML records that have a <userid> field equal to X (see above). Returns a failure message if "ObjectName" is a type that doesn't have a <userid> field.

**Note:** The following is subject to change without notice:

```
<Read type="Uprate" method="user" limit="500">
  <User>
    <id>X</id>
  </User>
</Read>
```

Returned:

```
<response>
  <Auth status="0">
  <Read status="0">
    <Uprate>
      <projectid>1</projectid>
```



```

        <userid>X</userid>
        <rate>1000000.99</rate>
    </Uprate>
</Read>
</response>

```

The only three fields are projectid, userid, and rate. Uprate objects cannot be added, deleted, or modified.

**Note:** As with the "Read, all" command, you can restrict the number of fields within records returned by inserting `<_Return><field> </_Return>` syntax before `</Read>`. You can return only deleted fields by specifying that the attribute = "1".

## Read, project

Use the Read, project command to restrict the records returned in a `<Read/>` request.

```

<Read type="datatype" method="project" limit="500">
  <Project>
    <id>X</id>
  </Project>
</Read>

```

Returned: A list of records that have a `<projectid>` field equal to X (see above). Make sure that the datatype used is a type that has a `<projectid>` field. Note that "limit" attribute is required as illustrated in [Example 6](#)

**Note:** As with the "Read, all" command, you can restrict the number of fields within records returned by inserting `<_Return><field> </_Return>` syntax before `</Read>`. You can return only deleted fields by specifying that the attribute = "1".

## Read, not exported

To request not-yet exported records, filter by not-exported. Note that "limit" attribute is required as illustrated in [Example 6](#).

```

<Read type="Slip" filter="not-exported" method="all"
limit="1000">
  <ImportExport>
    <application>MyApp</application>
  </ImportExport>
</Read>

```

Returned: slips that have not been marked as exported. Excludes exported records.

To mark returned Slip records as being exported, issue a modify command for each record returned and successfully exported in OpenAir system:

```

<Modify type="ImportExport">
  <ImportExport>

```

```

<application>MyApp</application>
<type>Slip</type>
<id>1</id>
<exported>
  <Date>
    <year>2011</year>
    <month>7</month>
    <day>15</day>
  </Date>
</exported>
</ImportExport>
</Modify>

```

## Report

Use the Report command to run a report and email a PDF copy of a Timesheet, Envelope, or Saved report.

```

<Report type="datatype"> (datatype can be "Timesheet", "Envelope", or "Report")
  <Report>
    <relatedid>X</relatedid> (Timesheet, Envelope, or Saved report ID)
    <email_report>1</email_report>
  </Report>
</Report>

```

Returned: Success if report exists. The report runs and an email with a PDF attachment gets emailed to the user requesting the report.

## Add

### Attribute Table

Use the following attribute.

Attribute Name	Value	Result
enable_custom	1	Custom fields to be included inline with other native fields

## Example 1

Use the Add command to add records.

```

<Add type="datatype">
  (One valid object matching datatype goes here.)
</Add>

```

Returned: An XML structure of type 'datatype' with all fields set to exactly how they appear in the OpenAir system. Included are a valid ID, an updated 'updated' timestamp, a correct 'created'. There is a limit of 1000 Add commands stacked in one request.

**Note:** User and Company records are added using the commands [CreateAccount](#) and [CreateUser](#).

## Example 2

```
<Add type="datatype" enable_custom="1">
  (One valid object matching datatype goes here.)
</Add>
```

**Note:** For more information, see [Adding/Modifying Records with Inline Custom Field Values](#).

## Example 3

In this example, the name of an existing category with externalid=111-222 is updated, or, if the category doesn't exist, it is created.

```
<Add type="Category" lookup="externalid">
  <Category>
    <name>XML created category - updated</name>
    <externalid>111-222</externalid>
  </Category>
</Add>
```

## Example 4

In this example, the externalid of an existing category (with XML-created name "category 1") is updated, or, if the category, doesn't exist, a new category is added.

```
<Add type="Category" lookup="name">
  <Category>
    <name>XML created category 1</name>
    <externalid>111-2222</externalid>
  </Category>
</Add>
```

## Example 5

In this example, the Add command is used to create an employee's CV in their resource profile and add it as an attachment. It is a two step process:

```
//Step 1

<Add type="ResourceAttachment">
  <ResourceAttachment>
    <type>CV</type>
    <userid>123</userid>
  </ResourceAttachment>
</Add>
```

```
//This returns an ID to use in the ownerid below, in Step 2:

//Step2

//The following step loads the CV. The CV file must be base64 encoded.
//The ownerid must be the same as the ResourceAttachment ID
//generated in Step 1.

<Add type="Attachment">
  <Attachment>
    <base64_data>U25lemth</base64_data>
    <file_name>Collins_Marc_CV.txt</file_name>
    <owner_type>ResourceAttachment</owner_type>
    <ownerid>98765</ownerid>
  </Attachment>
</Add>
```

## Delete (id)

Use the Delete (id) command to delete records. The <id> of the object to be deleted MUST be sent in order for this command to be successful. There is a limit of 1000 Delete commands stacked in one request.

```
<Delete type="datatype">
  (One object of type 'datatype', only id need be passed in.)
</Delete>
```

Returned:

- Success if a datatype with ID equal to <id> existed and was deleted.
- Failure if the ID doesn't exist.
- A list of brief records (the only data field will be ID) if the record exists, but couldn't be deleted because a list of other records depends on it and must be deleted first.

## Modify (id)

### Attribute Table

Use the following attribute.

Attribute Name	Value	Result
enable_custom	1	Custom fields to be included inline with other native fields

## Example 1

Use the Modify (id) command to change records. The <id> of the object to be modified MUST be sent in order for this command to be successful. There is a limit of 1000 Modify commands stacked in one

request. You can use an external id field as a foreign key and modify a record without querying first for an internal id (see [Example 5.](#))

```
<Modify type="datatype">
  (One object of type 'datatype', all fields included.)
</Modify>
```

Returned: An XML structure of type 'datatype' with all fields set to exactly how they appear in the OpenAir system, including an updated 'updated' timestamp.

## Example 2

To mark records as exported:

```
<Modify type="ImportExport">
  <ImportExport>
    <application>MyAppName</application>
    <type>Slip</type>
    <id>10158</id>
    <exported>
      <Date>
        <year>2007</year>
        <month>3</month>
        <day>14</day>
        <hour>11</hour>
        <minute>25</minute>
        <second>15</second>
      </Date>
    </exported>
  </ImportExport>
</Modify>
```

## Example 3

```
<Modify type="datatype" enable_custom="1">
  (One object of type 'datatype', all fields included.)
</Modify>
```



**Note:** For more information, see [Adding/Modifying Records with Inline Custom Field Values.](#)

## Example 4

To modify a customer record:

```
<Modify type="Customer" enable_custom="1">
  <Customer>
    <id>15</id>
    <name>New name</name>
    <myCustomField__c>10158</myCustomField__c>
```

```
</Customer>
</Modify>
```

**Note:** For more information, see [Adding/Modifying Records with Inline Custom Field Values](#).

## Example 5

To modify a project record using an externalid lookup (for external Customer 805-25664):

```
<Modify type="Project">
  <Project>
    <id>200</id>
    <customerid external="Customer">805-25664</customerid>
  </Project>
</Modify>
```

## Example 6

In this example, modify updates the cost\_centerid property of the category with id=3. The API looks up the internal id of the Costcenter with externalid 6655 and assigns the cost\_centerid property of the target category with a corresponding internal id.

```
<Modify type="Category">
  <Category>
    <id>3</id>
    <cost_centerid external="Costcenter">6655</cost_centerid>
  </Category>
</Modify>
```

## Example 7

In this example, modify updates the cost\_centerid property of the category with id=3. The API looks up the internal id of a Costcenter with the name "Maintenance" and assigns the cost\_centerid property of the target category with a corresponding internal id.

```
<Modify type="Category">
  <Category>
    <id>3</id>
    <cost_centerid name="Costcenter">Maintenance</cost_centerid>
  </Category>
</Modify>
```

## Modify (Logo)

**Note:** The following is subject to change without notice.

```
<Modify type="Logo">
```

```

<Logo>
  <name>html_logo</name>
  <type></type>
  <filename>companylogo.jpg
  </filename>
  <binary>FILE CONTENTS</binary>
</Logo>
</Modify>

```

where:

- `<name>` is either "html\_logo" or "pdf\_logo"
- `<type>` is calculated from the contents of the `</binary>` field (see below).
- `<filename>` is used for display purposes.
- `<binary>` is the actual content of the Logo file and should be sent in a Base64 encoded string, as XML does not support real binary fields.

Returned: An XML structure of type 'Logo' with all fields set to their actual value in the OpenAir system.

## Modify, custom equal to

**Note:** Modify, custom equal to is deprecated, but still supported. We recommend that you use inline custom fields by using the `enable_custom` attribute and specify custom fields with `"__c"` postfix. (Remember, there are two underscores before the c.) See [Adding/Modifying Records with Inline Custom Field Values](#).

Use the Modify, custom equal to command to set a custom field of an existing record. There is a limit of 1000 Modify commands stacked in one request.

```

<Modify type="datatype" method="custom equal to">
  <datatype>
    <id>X</id>
    <custom_field>custom_field_name</custom_field>
    <value>X</value>
  </datatype>
</Modify>

```

**Note:** `custom_field_name` is the field name as it appears in CustField properties, not the name of the database column that was created for that field or the Display Name of the field.

## Submit

Use the Submit command to submit an Envelope, Invoice, or Timesheet for approval. There is a limit of 1000 Submit commands stacked in one request.

```

<Submit type="Timesheet">
  <Timesheet>
    <id>35</id>

```

```

    </Timesheet>
    <Approval>
      <cc>name@company.com</cc>
      <notes>submission notes</notes>
    </Approval>
  </Submit>

```

Returned: A success or fail status is returned.

## CreateAccount

Use the CreateAccount command to create a new OpenAir account. When a new account is created, the first user (the account administrator) is also created.

The fields listed below are the minimum required fields. Any field in User and Company may be supplied at account creation.

```

<CreateAccount>
  <Company>
    <nickname/>
  </Company>
  <User>
    <nickname/>
    <password/>
    <addr>
      <Address>
        <email/>
      </Address>
    </addr>
  </User>
</CreateAccount>

```

Returned: Two XML structures — one of type 'Company' and the other of type 'User'. Both have fields set exactly as they appear in the OpenAir system. The type 'User' is always set to 'Administrator' for the first user created.

## CreateUser

Use the CreateUser command to create a new user within an existing OpenAir account. There is a limit of 1000 CreateUser commands stacked in one request.

The fields listed below are the minimum required fields. Any valid field in User may be populated at User creation time except custom fields. To set a user workschedule, refer to [User](#).

The CreateUser command will fail unless preceded by a valid Auth command for an Administrator user of this company. For more information, refer to [Auth](#).

```

<CreateUser>
  <Company>
    <nickname/>
  </Company>

```



```

<User>
  <nickname/>
  <password/>
  <addr>
    <Address>
      <email/>
    </Address>
  </addr>
</User>
</CreateUser>

```

Returned: An XML structure of type 'User' that has all fields set to exactly as they appear in the OA system. The default type 'User' is 'User' for all CreateUser requests.

## Auth

The Auth command authenticates a user into the specified account. It is the equivalent of entering company, user, and password information into the Login form on the product. It does not maintain any state after the request is finished. Many commands such as Read, Modify, and Delete require a valid Auth to succeed.

```

<Auth>
  <Login>
    <company/>
    <user/>
    <password/>
  </Login>
</Auth>

```

Returned: A success or fail status is returned. On success, subsequent commands such as Read, Modify, etc., will be able to access the data associated with the Login user.

## RemoteAuth

Use the RemoteAuth command to log in to an individual account in OpenAir. It returns a URL to the newly created session.

The difference between Auth and RemoteAuth is that RemoteAuth actually creates a valid user session, while Auth simply authenticates for the life of that individual request.

RemoteAuth is used by partners to perform Single Sign-on, where end users never have to log in to their OpenAir account. The RemoteAuth command takes care of this internally. RemoteAuth should not be used as a substitute for Auth.

```


<RemoteAuth>
  <Login>
    <company/>
    <user/>
    <password/>
  </Login>
</RemoteAuth>

```

Returned: A success or fail status is returned. On success, a URL returns that will place the user into the OpenAir system.

## MakeURL

Use the MakeURL command to obtain a URL for a specific application and screen. For instance, MakeURL can return a URL to display the Company Settings screen. It requires a valid user login to succeed. The list of valid views is listed below.

 **Note:** The following is subject to change without notice.

```
<MakeURL>
  <uid>1234</uid>
  <page>page name</page>
  <app>app abbreviation</app>
  <arg>
    <Envelope>
      <id>3</id>
    </Envelope>
  </arg>
</MakeURL>
```

where:

- <uid> is the user id of a valid logged-in user.
- <page> is a string from the valid list of pages (see below).
- <app> is 'km', 'ma', 'pb', 'rm', 'pm', 'ta', 'te', or 'tb' (See below).
- <arg> is an optional argument, should the page require it.

The following lists valid page strings with associated applications and arguments:

- **default-url**  
app= km, ma, pb, rm, pm, ta, te, or tb (points to the starting page in any one of the applications -the page you would see when you click on the application link.)  
**For example:** If you were using pm as the <app> attribute, the first page would be the Projects list in the Projects module for users with administrative privileges. 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> Currencies)
- **import-export**  
app= ma (points to Administration > Global Settings > 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)
- **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 > Expense Reports > Receipts)  
arg= <arg> <Envelope> <id>X</id> </Envelope> </arg>
- **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= <arg> <Invoice> <id>X</id> </Invoice> > </arg>
- **dashboard-project**  
app= pm (displays the dashboard view of the project with specified internal id)

```
arg= <arg> < Project > <id>X</id> </Project > </arg>
```

- **grid-timesheet**

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

```
arg= <arg> < Timesheet > <id>X</id> </Timesheet > </arg>
```

- **report-timesheet**

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

```
arg= <arg> < Timesheet > <id>X</id> </Timesheet > </arg>
```

Returned: A URL points to the desired page.

## Whoami

### Example 1

The Whoami command returns information about the currently authenticated user. It is the equivalent of using the Read command for User:

```
<Read type="User">
  <User>
    <id> X</id>
  </User>
</Read>
```

Whoami is as follows:

```
<Whoami>
  <User>1234</uid>
    <id>X</id>
  </User>
</Whoami>
```


Returned: The User XML record of the current authorized user. If a valid authorization did not occur, an error status is returned.

### Example 2

An Auth request with the Whoami command:

```
<Auth>
  <Login>
    <user>a</user>
    <company>b</company>
    <password>c</password>
  </Login>
</Auth>
<Whoami>
</Whoami>
```

## Version

 **Note:** The following Version command is for OpenAir Internal use only.

Use the Version command to look for client version information in the client's download/Versions file to see if a newer version of the client is available for download.

```
<Version status="0">
  <number>version number</number>
  <url>url for download</url>
  <size>12345</size>
</Version>
```

where:

- <number> is the version number of the client.
- <url> is the URL for the download of the latest version of the client.
- <size> is the size of the downloadable file.

## Approve

Use the Approve command to approve an Envelope, Invoice, or Timesheet submitted for approval. There is a limit of 1000 Approve commands stacked in one request.

```
<Approve type="Timesheet">
  <Timesheet>
    <id>308</id>
  </Timesheet>
  <Approval>
    <cc>name@company.com</cc>
    <notes>Approved</notes>
  </Approval>
</Approve>
```

Returned: A success or fail status is returned.

## Reject

Use the Reject command to reject an Envelope, Invoice, or Timesheet submitted for approval. There is a limit of 1000 Reject commands stacked in one request.

```
<Reject type="Timesheet">
  <Timesheet>
    <id>341</id>
  </Timesheet>
  <Approval>
    <cc>name@company.com</cc>
    <notes>Rejected</notes>
  </Approval>
</Reject>
```

```
</Approval>
</Reject>
```

Returned: A success or fail status is returned.

## Unapprove

Use the Unapprove command to unapprove a previously approved Envelope, Invoice, or Timesheet. There is a limit of 1000 Unapprove commands stacked in one request.

```
<Unapprove type="Timesheet">
  <Timesheet>
    <id>231</id>
  </Timesheet>
  <Approval>
    <cc>name@company.com</cc>
    <notes>Approved</notes>
  </Approval>
</Unapprove>
```

Returned: A success or fail status is returned.

## ModifyOnCondition

Use the ModifyOnCondition command to perform actions such as updating the external\_id of a record type only if the update time on the OpenAir server is older.

```
//ModifyOnCondition command supporting the "If-not-updated" condition
<ModifyOnCondition condition="if-not-updated" type="Booking">

//Followed by the object to update, for example, "Booking"
<Booking>
  <id>2</id>
  <external_id>123456789</external_id>
  <notes>New notes</notes>
  <user_id>152</user_id>
</Booking>

//Next, the date object to compare against
<Date>
  <day>1</day>
  <month>1</month>
  <year>2017</year>
  <hour>20</hour>
  <minute>5</minute>
  <second>09</second>
</Date>

</ModifyOnCondition>
```

Returned:

If <Date> is older than "modified" in the database, the command will return the full record from the database and the status "1200" (Command wasn't executed because condition wasn't met. Returning the record from DB.).

If <Date> is newer than or equal to "modified" in the database, the command will modify the record and return only the saved information (as with a standard Modify command). The Status will be "0" (Success).

# Custom Fields

Custom fields are helpful additions to your OpenAir account. Use the [CustField](#) datatype to get a list of custom field metadata related to the custom fields in your OpenAir account such as name, association, and picker type (use with [Read, custom equal to](#) method).

Several options exist for working with custom fields. You may request all available custom fields that exist for a given datatype or you may read custom field values for a specific record. You can also modify records to set custom field values or add/modify records with inline custom field values.

Refer to the following sections for more information on working with custom fields. Links to commands and code examples are provided.

**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.

## Requesting Custom Fields for a Datatype

You can request all custom fields that exist for a given datatype. Use the [Read, equal to](#) command and specify the [CustField](#) type and filter for a particular association. Refer to the [CustField](#) datatype [Association](#) table for a list of possible associations.

## Reading Custom Field Values Inline with Native Fields

You can read custom records with inline custom field values.

1. Custom fields can optionally be returned using custom field names (as defined in your OpenAir account) with “\_\_c” added to the end of the name. (Note that there are two underscores before the c.) The `enable_custom = “1”` attribute is required to include custom fields inline. See the [enable\\_custom](#) attribute for [Read](#).
2. Use custom fields as lookup values using “equal to” or “not equal to” methods. Use the “\_\_c” naming syntax as specified above and include the custom fields in the argument to the `read` method. The `enable_custom = “1”` attribute is required to include custom fields inline. See [XML Commands](#) for `read` and the following examples:
  - [Example 7](#) for `Read “equal to”`
  - [Example 2](#) for `Read “not equal to”`

**Note:** Remember, custom field names cannot start with a number or with the letters “xml” in any form such as XML or Xml. For example, 1MyCustomField or xmlMyCustomField will not work.

## Reading Custom Field Values in a Separate Request

You can read custom field values for a given record. Use the [Read, custom equal to](#) command to request custom field values for a particular record. You need to know the internal ID of the record in question.



## Adding/Modifying Records with Inline Custom Field Values

You can add or modify records with inline custom field values.

1. 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 as part of an argument object to an add or modify request. Set the “enable\_custom” attribute equal to 1. See the [XML Commands](#) for add and modify and the following examples:
  - [Example 2](#) for Add custom field
  - [Example 3](#) and [Example 4](#) for Modify custom fields

**Note:** Remember, custom field names cannot start with a number or with the letters “xml” in any form such as XML or Xml. For example, 1MyCustomField or xmlMyCustomField will not work.

**Note:** If a custom field update fails due to: 1) optional uniqueness of the field, or 2) when the value of the custom field does not match an acceptable value, you may receive a warning. The Add or Modify request status is set to the following:

1106 and detailed <error> message would be returned with the description of specific custom field errors.

It is important to note that the parent record would be saved successfully, but the custom fields will fail to be updated. Refer to the following example:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<response>
  <Modify status = "1106">
    <Customer>
      <cust_cust__c/><cust_date__c>2012-03-29</cust_date__c><cust_cust1__c/>
    </Customer>
    <errors>Custom field cust_cust__c failed to save with status code: 1104</errors>
  </Modify >
</response>
```

See Appendix A Error Code Listing for [Custom Field Errors](#), specifically error code 1106.

## Modifying Records to Set Custom Field Values

You can modify records to set custom field values. Use the [Modify, custom equal to](#) command to set custom fields.

# XML Datatypes

OpenAir contains the following XML datatypes. Click on the datatype to see the list of field names and associated descriptions as well as links to supported commands.



**Important:** The updated and created fields are maintained automatically by the system. You can read these values, but they cannot be modified.

XML Datatypes		
Actualcost	AccountingPeriod	
Address	Agreement	
Agreement_to_project	Approval	ApprovalLine
Approvalprocess	Attachment	Attribute
AttributeDescription		
Attributeset	BillingSplit	Booking
Booking_request	BookingByDay	BookingType
Budget	BudgetAllocation	Category
Category_1	Category_2	Category_3
Category_4	Category_5	Ccrate
Company	Contact	Costcategory
Costcenter	Costtype	Currency
Currencyrate	CustField	Customer
Customerpo	Customerpo_to_project	CustomerProspect
Date	Deal	Dealcontact
Dealschedule	Department	Entitytag
Envelope	Error	Estimate
Estimateadjustment	Estimateexpense	Estimatelabor
Estimatemarkup	Estimatephase	Event
ExpensePolicy	ExpensePolicyItem	
Filter	Filterset	Flag
ForexInput	FormPermissionField	Fulfillment
Hierarchy	HierarchyNode	History
ImportExport	Invoice	InvoiceLayout
Issue	IssueCategory	IssueSeverity
IssueSource	IssueStage	IssueStatus
Item	ItemToUserLocation	
Jobcode	Leave_accrual_rule	
Leave_accrual_rule_to_user	Leave_accrual_transaction	LoadedCost

XML Datatypes		
Login	Module	Notes
Payment	Paymentterms	Paymenttype
Payrolltype	PendingBooking	Preference
Product	Project	Projectassign
ProjectAssignmentProfile	Projectbillingrule	
Projectbillingtransaction	ProjectBudgetGroup	ProjectBudgetRule
ProjectBudgetTransaction	Projectgroup	Projectlocation
Projectstage	Projecttask	ProjecttaskEstimate
Projecttask_type		
Projecttaskassign	Proposal	Proposalblock
Proxy		
Purchase_item	Purchaseorder	Purchaser
Purchaserequest	Ratecard	RateCardItem
Reimbursement	Repeat	Report
Request_item	ResourceAttachment	Resourceprofile
Resourceprofile_type		
ResourceRequest	ResourceRequestQueue	Resourcsearch
RevenueContainer	RevenueProjection	Revenue_recognition_rule
Revenue_recognition_rule_amount	Revenue_recognition_transaction	RevenueStage
Role	Schedulebyday	Scheduleexception
Schedulerequest	Schedulerequest_item	Slip
SlipProjection	Slipstage	TagGroup
TagGroupAttribute	TargetUtilization	Task
TaskTimecard	TaxLocation	TaxRate
Term	Ticket	Timecard
Timesheet	Timetype	Todo
Uprate	User	
UserLocation	UserWorkschedule	Vendor
Viewfilter	Viewfilterrule	Workspace
Workspacelink	Workspaceuser	

## Actualcost

Use the Actualcost datatype to add or update actual cost information.

```
<Actualcost>
```

```

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

```

This datatype supports the read, add, and modify [XML Commands](#).

## AccountingPeriod

The AccountingPeriod datatype holds a date range defining an accounting period.

```

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

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Address

Use the Address datatype for XML sub-structures of address-related information.

```

<Address>
  <id/> Unique ID. Automatically assigned by the system.

```

```

<salutation/> Contact's salutation
<mobile/> Mobile phone number
<state/> State
<email/> Email address
<addr2/> Address line 2
<city/> City
<fax/> Fax number
<contact_id/> The ID of the associated contact.
<addr1/> Address line 1
<middle/> Middle name
<country/> Country
<first/> First name
<last/> Last name
<phone/> Phone number
<addr4/> Address line 4
<zip/> Zip code
<addr3/> Address line 3
</Address>

```

This datatype supports the add, CreateAccount, CreateUser, and modify [XML Commands](#). The following is an example of how a contact's city would be represented.

```

<Contact>
  <addr>
    <Address>
      <city>X</city>
    </Address>
  </addr>
</Contact>

```

**Note:** The <Address/> datatype now has a "customer\_only" attribute, which is used for backwards compatibility support of <billingaddr/> for <Customer/>. If you set this attribute to "yes", only the billing address of the customer, and not its associated contact, will be updated when modifying the customer record. See [Customer](#) for more information.

## Agreement

Use the Agreement datatype to track money through projects and billings.

```

<Agreement>
  <id/> Unique ID. Automatically assigned by the system.
  <number/> The agreement number.
  <date/> The date of the agreement.
  <name/> The name of the agreement.
  <active/> A 1/0 field indicating whether this is an active
  agreement.
  <externalid/> External ID.
  <total/> The agreement total. Dated by the date field.
  <created/> Time the record was created.
  <currency/> Currency for the money fields in the record.
  <notes/> Notes.
  <customerid/> Customer ID.
  <updated/> Time the record was last modified.

```

```

<code/> Optional accounting system code for integration with
external accounting systems.
<acct_date/> The accounting period date of the agreement.
<picklist_label/> Label as shown on form picklist.
</Agreement>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Agreement\_to\_project

Use the Agreement\_to\_project datatype to create a many-to-many link between projects and agreements.

```


<Agreement_to_project>
  <id/> Unique ID. Automatically assigned by the system.
  <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.
</Agreement_to_project>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Approval

Use the Approval datatype to store approval information for timesheets, expense reports, and proposals.

 **Note:** This datatype is not used to read the “approval” table. See [ApprovalLine](#).

```

<Approval>
  <cc/> Email cc field.
  <notes/> Notes.
</Approval>

```

This datatype supports the [Submit](#) command.

## ApprovalLine

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

```

<ApprovalLine>
  <id/> Unique ID. Automatically assigned by the system.
  <approvalid/> ID of the associated approval. Represents a meta-approval, or an "approval
confirmation".
  <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
    <timesheetid/> ID of the associated timesheet.
    <envelopeid/> ID of the associated envelope (expense report)
    <proposalid/> ID of the associated proposal
    <purchaserequestid/> ID of the associated purchaserequest
    <purchaseorderid/> ID of the associated purchaseorder
    <authorizationid/> ID of the associated authorization
    <schedule_requestid/> ID of the associated schedule request
    <booking_requestid/> ID of the associated booking request
    <deal_booking_requestid/> ID of the associated deal booking request
    <invoiceid/> ID of the associated invoice
    <revenue_containerid/> ID of the associated revenue_container
    <bookingid/> ID of the associated booking
    <customerid/> ID of the associated customer
    <project_budget_groupid/> ID of the associated project budget group
    <projectid/> ID of the associated project if this is a project approval
    <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.
    <submitter/> ID of the user submitting the approval. Only valid for a submittal record
        (action = 'S').
    <approvalprocessid/> ID of the approval process if this is associated with an approval
        process.
    <approvalprocess_ruleid/> ID of the approval process rule if this is associated with an
        approval process.
    <seq_number/> If this is associated with an approval process, this is the sequence
        number associated with it.
    <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
    <date/> Date and time of the action
    <pending_done/> If the action is 'P'ending, this flag is set to 1 once an 'A' or 'R'
        action record is created.
    <project_total/> If this is a project-based approval this holds the total amount
        (money or hours) that was approved.
    <notes/> Notes, reasons, etc.
    <created/> Time the record was created
    <updated/> Time the record was last updated or modified
    <audit/> Audit trail of changes
    <delay_to/> Delay action until this time
    <delay_action/> Delayed action
</ApprovalLine>

```

This datatype supports the [Read](#) command.

## Approvalprocess

Use the Approval Process datatype to read approval process information.

```

<Approvalprocess>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The name used for display in popups and lists.
  <externalid/> If the record was imported from an external system you store the unique external record ID here.
  <updated/> Time the record was last modified.
  <created/> Time the record was created.
</Approvalprocess>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Attachment

Use the Attachment datatype to specify information about task and proposal attachments and documents or folders.

```

<Attachment>
  <id/> Unique ID. Automatically assigned by the system.
  <file_name/> The true attachment name, as provided by the user on upload.
  <locked_by/> The ID of the user who uploaded the file, 0 if unlocked.
  <notes/> Notes associated with the attachment.
  <created/> Time the record was created.
  <workspaceid/> The ID of the associated workspace.
  <base64_data/> Base 64 encoded binary data of the actual attachment file.
  <updated/> Time the record was last modified.
  <attachmentid/> If non-zero, the attachment record associated with this attachment.
  <parentid/> The attachment ID of our immediate ancestor. If zero/null, this is a top-level document/folder.
  <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.
  <size/> The size, in bytes of the associated file. This attribute is read-only.
  <ownerid/> The id of the record linking to this attachment.
  <is_a_folder/> A "1/0" field indicating if any other attachments have us as a parent.
  <owner_type/> The owner of this attachment, e.g. 'User', 'Envelope', 'Ticket', 'Timesheet', 'Agreement', or 'Customerpo'.
  <name/> The display name of the attachment.
</Attachment>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Attribute

Use the Attribute datatype to read attribute information.

```

<Attribute>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The name of the attribute.
  <attribute_setid/> The name of the attribute.
  <updated/> Time the record was last modified.
  <created/> Time the record was created.

```



```
<notes/> Attribute notes.
</Attribute>
```

This datatype supports the read [XML Commands](#).

## AttributeDescription

Use this datatype for descriptions of attributes in resource profiles, for example, detailed descriptions of what characteristics define various language levels (beginner, intermediate, advanced) or technical competencies.

```
<AttributeDescription>
  <id/> Unique ID. Automatically assigned by the system.
  <resourceprofile_typeid/> ID of the resourceprofile_type.
  <attributeid/> ID of the attribute.
  <description/> Information about the attribute in context of specific
  resourceprofile_type.
  <deleted/> A "1/0" field indicating if the record was deleted.
  <created/> Time the record was created.
  <updated/> Time the record was last modified.
</AttributeDescription>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Attributeset

Use the Attributeset datatype to read attributeset information.

```
<Attributeset>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The name of the attributeset.
  <updated/> Time the record was last modified.
  <created/> Time the record was created.
  <notes/> Attributeset notes.
</Attributeset>
```

This datatype supports the read [XML Commands](#).

## BillingSplit

Use the BillingSplit datatype to read attributeset information.

```
<BillingSplit>
  <id/> Unique ID. Automatically assigned by the system.
  <slipid/> The id of the slip that was created.
  <project_billing_transactionid/> The ID of the associated
  project billing transaction.
  <taskid/> The ID of the associated task.
```

```

    <updated/> Time the record was last modified.
    <created/> Time the record was created.
  </BillingSplit>

```

This datatype supports the read [XML Commands](#).

## Booking

Use the Booking datatype to book a user to a project.

```

<Booking>
  <id/> Unique ID. Automatically assigned by the system.
  <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.
  <ownerid/> The ID of the associated user creating the booking.
  <userid/> The ID of the associated user.
  <startdate/> The start date of 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.
  <projectid/> The ID of the associated project.
  <externalid/> If the record was imported from an external system
  you store the unique external record ID here.
  <booking_typeid/> The ID of the associated booking_type.
  <project_taskid/> The ID of the task within the assoc. project.
  <created/> Time the record was created.
  <repeatid/> The ID of the associated repeating event.
  <enddate/> The end date of the booking.
  <notes/> Booking notes.
  <customerid/> The ID of the associated customer.
  <updated/> Time the record was last updated or modified.
  <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.
  <starttime/> Start time.
  <endtime/> End time.
  <job_code_id/> The ID of the associated job code.
  <locationid/> The location ID for this booking.
  <notify_owner/> A 1/0 field indicating whether to send email to
  the requestor when the booking is modified.
  <date_approved/> The date the booking request was approved.
  <date_submitted/> The date the booking_request was submitted.
  <approval_status/> The approval status of the booking request (0
  - Open, S - Submitted, A - Approved, R - Rejected).
  <project_assignment_profile_id/>The id of the associated project
  assignment profile.
  <source_booking_id/> Id of the booking used to create this
  booking.
</Booking>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## BookingByDay

Use the BookingByDay datatype to access a day by day representation of the booking table.

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

This datatype supports the read [XML Commands](#).

## BookingType

Use the BookingType datatype to describe a booking type such as billable, non-billable, or business development used in Resources module bookings.

```
<BookingType>
  <id/> Unique ID. Automatically assigned by the system.
  <priority/> The priority of the booking type (1 - 9).
  <created/> Time the record was created.
  <notes/> Booking notes.
  <name/> The name of the booking type.
  <active/> A 1/0 field specifying if the type is active.
  <updated/> Time the record was last modified.
  <picklist_label/> Label as shown on form picklist.
</BookingType>
```

This datatype supports the read, add, and modify [XML Commands](#).

## Booking\_request

Use the Booking\_request datatype to read booking requests.

```
<Booking_request>
  <id/> Unique ID. Automatically assigned by the system.
  <number/> The booking_request number that increments by 1.
  <project_task_id/> The id of the task within the associated
  project.
```

```

<startdate/> The start date of the booking_request.
<job_code_id/> The id of the associated job code.
<notify_owner/> A "1/0" field indicating whether to send email
to the booking request owner changes occur to the resulting
bookings.
<customer_id/> The id of the associated customer.
<date_approved/> The date the booking_request was approved.
<enddate/> The end date of the booking_request.
<updated/> Time the record was last updated or modified.
<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.
<project_id/> The id of the associated project.
<date_submitted/> The date the booking_request was submitted.
<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.
<attachment_id/> If non-zero, the attachment record associated
with this booking_request.
<approval_status/> The approval status of the booking request
'O'pen
'P'ending approval
'A'pproved
'R'ejected
<booking_type_id/> The id of the associated booking_type.
<name/> The name of the booking_request (Prefix + number).
<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.
<description/> The description or purpose for the
booking_request.
<repeat_id/> The id of the associated repeating event.
<created/> Time the record was created.
<external_id/> If the record was imported from an external
system you store the unique external record id here.
<notes/> Booking notes
<user_id/> The id of the associated user.
<owner_id/> The id of the associated user creating the booking
request.
<prefix/> A static alphanumeric booking_request number prefix.
</Booking_request>

```

This datatype supports the read [XML Commands](#).

## Budget

Use the Budget datatype to create a budget entry.

```

<Budget>
  <id/> Unique ID. Automatically assigned by the system.
  <date/> The date of the budget entry.

```

```

<name/> The name.
<projectid/> The ID of the associated project.
<total/> The total value of budget entry. Dated by the date
field.
<budgetcategory_id/> The ID of the budget category.
<created/> Time the record was created.
<currency/> Currency for the money fields in the record.
<notes/> Budget notes.
<customerid/> The ID of the associated customer.
<updated/> Time the record was last modified.
<categoryid/> The ID of the associated category.
</Budget>

```

This datatype supports the read, add, and modify [XML Commands](#).

## BudgetAllocation

Use the BudgetAllocation datatype to allocate users and activity to a budget.

```

<BudgetAllocation>
  <id/> Unique ID. Automatically assigned by the system.
  <budgetid/> The ID of the associated budget.
  <userid/> The ID of the associated user.
  <date/> The date of the budget entry.
  <projectid/> The ID of the associated project.
  <budgetactivity_id/> The ID of the budget activity.
  <total/> The total value of budget entry. Dated by the date
field.
  <budgetcategory_id/> The ID of the budget category.
  <created/> Time the record was created.
  <currency/> Currency for the money fields in the record.
  <customerid/> The ID of the associated customer.
  <updated/> Time the record was last modified.
  <allocation/> The percentage of the budget entry that this user
was allocated to.
</BudgetAllocation>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Category

Use the Category datatype for a service, category, activity or time type in the Proposals, Timesheets, and Invoices modules. Typically, only one of the rate mechanisms will be set.

```

<Category>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The category name.
  <active/> A 1/0 field indicating whether this is designated as an
active customer.
  <taxable/> A 1/0 field indicating whether this item is taxable,
vat-taxable, and so on.
  <externalid/> If the record was imported from an external system

```

```

you store the unique external record id here.
<other_rate_type/> The time the other_rate field applies to.
Valid entries are Day, Week, Month, Quarter, Year and Session.
<other_rate/> The rate for another time billing metric.
<currency/> Currency for the money fields in the record.
<created/> Time the record was created.
<rate/> The hourly billing rate.
<cost_centerid/> The ID of the associated cost center.
<fixed_fee/> The fixed fee value of this service.
<updated/> Time the record was last updated or modified.
<code/> Optional accounting system code for integration with
external accounting systems.
<notes/> Category notes.
<picklist_label/> Label as shown on form picklist.
</Category>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Category\_1

Use the Category\_1 datatype for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as Category\_2, Category\_3, Category\_4, and Category\_5.

```

<Category_1>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The category name.
  <code/> Optional accounting system code for integration with
external accounting systems.
  <externalid/> If the record was imported from an external system
you store the unique external record ID here.
  <active/> A 1/0 field indicating whether this is designated as an
active customer.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <notes/> Category notes_1.
  <picklist_label/> Label as shown on form picklist.
</Category_1>

```

This datatype supports the read, add, modify, and delete [XML Commands](#). Use custom equal to when requesting custom fields.

## Category\_2

Use the Category\_2 datatype for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as Category\_1, Category\_3, Category\_4, and Category\_5.

```

<Category_2>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The category name.
  <code/> Optional accounting system code for integration with

```

```

external accounting systems.
<externalid/> If the record was imported from an external system
you store the unique external record ID here.
<active/> A 1/0 field indicating whether this is designated as an
active customer.
<created/> Time the record was created.
<updated/> Time the record was last updated or modified.
<notes/> Category notes_2.
<picklist_label/> Label as shown on form picklist.
</Category_2>

```

This datatype supports the read, add, modify, and delete [XML Commands](#). Use custom equal to when requesting custom fields.

## Category\_3

Use the Category\_3 datatype for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as Category\_1, Category\_2, Category\_4, and Category\_5.

```

<Category_3>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The category name.
  <code/> Optional accounting system code for integration with
external accounting systems.
  <externalid/> If the record was imported from an external system
you store the unique external record ID here.
  <active/> A 1/0 field indicating whether this is designated as an
active customer.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <notes/> Category notes_3.
  <picklist_label/> Label as shown on form picklist.
</Category_3>

```

This datatype supports the read, add, modify, and delete [XML Commands](#). Use custom equal to when requesting custom fields.

## Category\_4

Use the Category\_4 datatype for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as Category\_1, Category\_2, Category\_3, and Category\_5.

```

<Category_4>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The category name.
  <code/> Optional accounting system code for integration with
external accounting systems.
  <externalid/> If the record was imported from an external system
you store the unique external record ID here.
  <active/> A 1/0 field indicating whether this is designated as an

```

```

    active customer.
    <created/> Time the record was created.
    <updated/> Time the record was last updated or modified.
    <notes/> Category notes_4.
    <picklist_label/> Label as shown on form picklist.
  </Category_4>

```

This datatype supports the read, add, modify, and delete [XML Commands](#). Use custom equal to when requesting custom fields.

## Category\_5

Use the Category\_5 datatype for extended category capability for transactions. It provides the ability to relate transactions to one or more built-in categories such as Category\_1, Category\_2, Category\_3, and Category\_4.

```

<Category_5>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The category name.
  <code/> Optional accounting system code for integration with
  external accounting systems.
  <externalid/> If the record was imported from an external system
  you store the unique external record ID here.
  <active/> A 1/0 field indicating whether this is designated as an
  active customer.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <notes/> Category notes_5.
  <picklist_label/> Label as shown on form picklist.
</Category_5>

```

This datatype supports the read, add, modify, and delete [XML Commands](#). Use custom equal to when requesting custom fields.

## Ccrate

Use the Ccrate datatype to document the category customer rate table.

```

<Ccrate>
  <id/> Unique ID. Automatically assigned by the system.
  <categoryid/> The ID of the category this rate is associated
  with.
  <currency/> The currency these rates are quoted in.
  <rate/> The hourly billing rate.
  <created/> Time the record was created.
  <notes/> Notes about the table.
  <customerid/> The ID of the customer this rate is associated
  with.
  <updated/> Time the record was last updated or modified.
</Ccrate>

```

This datatype supports the read [XML Commands](#).



## Company

Use the Company datatype to specify basic company information and the company switches.

```
<Company>
  <id/> Unique ID. Automatically assigned by the system.
  <addr/> The company's address.
  <VAT_registration_number/> VAT registration number.
  <hide_rate/> Hide hourly rate from normal user types in the
  company.
  <updated/> Time the record was last updated or modified.
  <company/> The company name, as it should be printed on invoices.
  <nickname/> The company nickname.
  <is_multicurrency/> Multiple currencies.
  <currencies/> The currencies for the money fields in the record.
  <businesstype/> General business category.
  <created/> The time the record was created.
  <base_currency/> Base currency.
  <rate_from/> Billing rate is pulled from: category, user,
  customer/project, or user/project.
  <workscheduleid/> The ID of the associated primary account
  workschedule. (read-only field)
  <flags/> Company-specific flags.
</Company>
```

This datatype supports the read, add, and modify [XML Commands](#). Also refer to the [Flag](#) datatype.

## Contact

Use the Contact datatype to specify contact information. A contact is associated with a customer, particular client, or prospect company.

```
<Contact>
  <id/> Unique ID. Automatically assigned by the system.
  <addr/> The contact's address.
  <customer_company/> Import-only field to specify customer by
  company name. Can be used in place of </customerid>.
  <job_title/> The contact's job title.
  <updated/> Time the record was updated or modified.
  <can_bill_to/> A 1/0 field indicating if the contact can be a
  billing contact.
  <code/> Optional accounting system code for integration with
  external accounting systems.
  <name/> The name of the contact. This will be automatically
  generated if not supplied.
  <active/> A 1/0 field indicating an active contact.
  <externalid/> If the record was imported from an external system
  you store the unique external record ID here.
  <can_sold_to/> A 1/0 field indicating if the contact can be a
  sold to contact.
  <created/> Time the record was created.
  <notes/> Notes field.
  <customerid/> The ID of the associated customer.
```

```

    <customer_externalid/> The external ID for the associated
    customer.
    <can_ship_to/> A 1/0 field indicating if the contact can be a
    shipping contact.
    <exported/> Date and time the record was marked as "exported".
    <picklist_label/> Label as shown on form picklist.
  </Contact>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Costcategory

Use the Costcategory datatype to add or update cost category information.

```

<Costcategory>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The name of the cost category.
  <active/> A 1/0 field indicating if this cost category is active.
  <notes/> Notes.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <externalid/> If the record was imported from an external
    system, you store the unique external record ID here.
</Costcategory>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Costcenter

Use the Costcenter datatype to specify cost center information.

```

<Costcenter>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <notes/> Cost center notes.
  <name/> The name of the cost center.
  <active/> A 1/0 field indicating whether this is active.
  <updated/> Time the record was last updated or modified.
  <externalid/> If the record was imported from an external system
    you store the unique external record ID here.
  <code/> Optional accounting system code for integration with
    external accounting systems.
  <picklist_label/> Label as shown on form picklist.
</Costcenter>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Costtype

Use the Costtype datatype to add or update cost category information.

```

<Costtype>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The name of the cost category.
  <active/> A 1/0 field indicating if this cost category is active.
  <externalid/> If the record was imported from an external
  system, you store the unique external record ID here.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <notes/> Notes.
  <cost_categoryid/> The ID of the associated cost category.
</Costtype>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Currency

Use the Currency datatype to specify exchange rates that override market rates.

```

<Currency>
  <rate/> The account's custom conversion rate.
  <created/> Time the record was created.
  <symbol/> The currency symbol.
  <updated/> Time the record was last updated or modified.
</Currency>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Currencyrate

Use the Currencyrate datatype to read currency rates.

```

<Currencyrate>
  <crate/> The account's currency conversion rate.
  <csymbol/> The currency symbol.
  <cname/> The name of the currency rate.
  <date/> The date of the rate.
  <type/> Blank for rates with date filled in, otherwise: PAST for
  conversion rates for dates prior to the first date in the table
  and FUTURE for conversion rate for dates in the future.
</Currencyrate>

```

This datatype supports the read [XML Commands](#).

## CustField

Use the CustField datatype to retrieve metadata about custom fields such as name, association, and picker type.

```

<CustField>

```

```

<id/> Unique ID. Automatically assigned by the system.
<userid/> The ID of user who created or owns this custom field.
<rows/> The number of display rows for text area fields
<size/> The display size of the field on forms.
<valuelist/> A list of values for radio groups and popup menu
fields in csv format.
<required/> A 1/0 field indicating if this field is required.
<decpos/> The decimal size of the field.
<picker/> The type of field for on screen representation:
numeric, currency, date, text, textarea, check, radio, drop
down, drop text, selector, or alloc_gr.
<association/> The table or datatype this field is associated
with. See the Association table for possible associations.
<updated/> Time the record was last updated or modified.
<seq/> The sequence number of the field.
<divider_text/> Optional divider text.
<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.
<active/> A 1/0 field indicating if this alert is active.
<next_seq/> Next sequence number to use.
<description/> The description of the custom field.
<force_unique/> A 1/0 field indicating if this field is unique.
<defnow/> A 1/0 field indicating if date fields default to today.
<created/> Time the record was created.
<hint/> The hint used on forms.
<title/> The title used on forms with this custom field.
<divider/> A 1/0 field indicating whether to paint a divider.
<never_copy/> A 1/0 field indicating if the field can be cloned.
<hidden_data_entry/> A 1/0 field indicating whether the custom
field should be hidden on the data entry UI.
</CustField>

```

This datatype supports the read and modify [XML Commands](#).

## Association table

The CustField datatype uses the following associations. The association is the name of the table that the custom field is related to. For more information, see association at the following URL: [http://www.openair.com/database/single\\_user.html#cust\\_field](http://www.openair.com/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	

## Customer

Use the Customer datatype for customer, client, or patient information. The customer is the individual or company that is billed or expensed.

```

<Customer>
  <id/> Unique ID. Automatically assigned by the system.
  <addr/> The customer's address.
  <invoice_layoutid/> The ID of the associated invoice layout.
  <rate/> Hourly billing rate for this customer.
  <bus_typeid/> Type of business this customer is in.
  <code/> Optional user-defined code.
  <name/> The nickname used for display in popups and lists.
  <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.
  <territoryid/> The territory for this customer.
  <hierarchy_node_ids/> Comma delimited list - hierarchy nodes
  this object belongs to.
  <hear_aboutid/> How did they hear about us.
  <statements/> A 1/0 field indicating if this customer can view
  statements.
  <company_sizeid/> This customer's company size.
  <web/> Customer's Web address.
  <currency/> Currency for the money fields in the record. Also the
  default currency when an invoice is created.
  <cost_centerid/> The ID of the associated cost center.
  <contactaddr/> The contact address for the customer.
  <billingaddr/> The billing address for the customer.
  <billing_contact_id/> The billing contact ID.
  <notes/> Notes about the customer.
  <tb_approvalprocess/> The approvalprocess_id of the invoice
  approval process. This field is mutually exclusive with

```

```

    tb_approver.
    <primary_contactid/> The billing contact ID.
    <filterset_ids/> Comma delimited list - filter sets this object
    belongs to.
    <active/> A 1/0 field indicating whether this is designated as an
    active customer.
    <externalid/> If the record was imported from an external system
    you store the unique external record ID here.
    <invoice_prefix/> Text to start every invoice number with.
    <type/> A C/P field indicating whether this is Customer or a
    Prospect.
    <userid/> The user ID of the customer or owner.
    <terms/> Standard payment terms for the customer. Textual
    description like Net 30.
    <created/> Time the record was created.
    <invoice_text/> Text to display on every invoice.
    <company/> The company name.
    <updated/> Time the record was last updated or modified.
    <shipping_contactid/> The shipping contact ID.
    <sold_to_contact_id/> The sold to contact ID.
    <billing_code/> The customer billing code. Used in bulk
    invoicing.
    <createtime/> Same as the created field (for legacy systems).
    <ta_include/> A 1/0 field indicating whether a Timesheet filter
    set is applied.
    <te_include/> A 1/0 field indicating whether an Expense Report
    filter set is applied.
    <updatetime/> Same as the updated field (for legacy systems).
    <picklist_label/> Label as shown on form picklist.
  </Customer>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

**Note:** Refer to the following notes regarding the Customer datatype and fields:

- With the introduction of the <Contact/> datatype, the <billingaddr/> field for Customer is somewhat obsolete since each customer now has a primary billing contact that is designated in the contact table. The new <billing\_contact\_id/> field for Customer is used to associate a billing contact from the contact table with a customer.
- The <billingaddr/> field will continue to be supported for backward compatibility. If you do use <billingaddr/> when adding a new customer, not only will it add the customer record, but it will also add a contact record with the same billing information, so you create both a customer and a contact. If you later modify the <billingaddr/> of your customer, the contact record will also be modified.
- If you are still using <billingaddr/> and want to modify the customer's billing address but not the contact's address, you can do that by setting the "customer\_only" attribute in the <Address/> to "yes". See the following example:

```

<Modify type="customer">
  <Customer>
    <billingaddr>
      <Address customer_only="yes">
        <addr1>1234 Main St</addr1>
      </Address>
    </billingaddr>
  </Customer>
</Modify>

```

```

    </billingaddr>
  </Customer>
</Modify>

```

**Note:** If you would like to work with information about both customers and prospects, use the [CustomerProspect](#) datatype. You can also use the [CustomerProspect](#) datatype.

## Customerpo

Use the Customerpo datatype to track money through projects and billings.

```

<Customerpo>
  <id/> Unique ID. Automatically assigned by the system.
  <number/> The customerpo number.
  <date/> The date of the customerpo.
  <name/> The name of the customerpo.
  <active/> A 1/0 field indicating whether this is an active
    customerpo.
  <externalid/> If the record was imported from an external system
    you store the unique external record id here.
  <total/> The customerpo total. Dated by the date field.
  <created/> Time the record was created.
  <currency/> Currency for the money fields in the record.
  <notes/> Notes.
  <customerid/> The ID of the associated customer.
  <customer_externalid/> The external ID for the associated
    customer.
  <updated/> Time the record was last modified.
  <code/> Optional accounting system code for integration with
    external accounting systems.
  <acct_date/> The accounting period date of the customerpo.
  <picklist_label/> Label as shown on form picklist.
</Customerpo>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Customerpo\_to\_project

Use the Customerpo\_to\_project datatype to create a many-to-many link between projects and customers.

```

<Customerpo_to_project>
  <id/> Unique ID. Automatically assigned by the system.
  <customerpo_id/> 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.
  <projectid/> The ID of the associated project.
  <externalid/> If the record was imported from an external

```

```

    system, you store the unique external record ID here.
  </Customerpo_to_project>

```

This datatype supports the read, add, and modify [XML Commands](#).

## CustomerProspect

Use the CustomerProspect datatype to specify information about prospective customers or clients. The field names and definitions are similar to those associated with [Customer](#) datatype.

```

<CustomerProspect>
  <id/> Unique ID. Automatically assigned by the system.
  <addr/> The prospective customer's address.
  <invoice_layoutid/> The ID of the associated invoice layout.
  <rate/> Hourly billing rate for this prospective customer.
  <bus_typeid/> Type of business this prospective customer is in.
  <code/> Optional user-defined code.
  <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 and if -2 then the approver is the owners manager's
    manager.
  <territoryid/> The territory for this prospective customer.
  <name/> The nickname used for display in popups and lists.
  <hierarchy_node_ids/> Comma delimited list - hierarchy nodes
    this object belongs to.
  <hear_aboutid/> How did they hear about us.
  <statements/> A 1/0 field indicating if this prospective
    customer can view statements.
  <company_sizeid/> This prospective customer's company size.
  <web/> Prospective customer's Web address.
  <currency/> Currency for the money fields in the record. Also the
    default currency when an invoice is created.
  <cost_centerid/> The ID of the associated cost center.
  <contactaddr/> The contact address for the prospective customer.
  <billingaddr/> The billing address for the prospective customer.
  <billing_contact_id/> The billing contact ID.
  <notes/> Notes about the prospective customer.
  <tb_approvalprocess/> The approvalprocess_id of the invoice
    approval process. This field is mutually exclusive with
    tb_approver.
  <primary_contactid/> The billing contact ID.
  <filterset_ids/> Comma delimited list - filter sets this object
    belongs to.
  <active/> A 1/0 field indicating whether this is designated as an
    active customer.
  <externalid/> If the record was imported from an external system
    you store the unique external record ID here.
  <invoice_prefix/> Text to start every invoice number with.
  <type/> A C/P field indicating whether this is Customer or a
    Prospect.
  <userid/> The user ID of the prospective customer or owner.
  <terms/> Standard payment terms for the prospective customer.

```



```

    Textual description like Net 30.
    <createtime/> The date the record was created.
    <invoice_text/> Text to display on every invoice.
    <company/> The company name.
    <updatetime/> The last date the record was changed.
    <shipping_contactid/> The shipping contact ID.
    <billing_code/> The customer billing code. Used in bulk
    invoicing.
    <created/> Time the record was created.
    <sold_to_contactid/> The sold to contact ID.
    <ta_include/> A 1/0 field indicating whether a Timesheet filter
    set is applied.
    <te_include/> A 1/0 field indicating whether an Expense Report
    filter set is applied.
    <updated/> Time the record was last updated or modified.
  </CustomerProspect>

```

This datatype supports the read, add, modify, and delete [XML Commands](#). When doing a <Read/> with CustomerProspect as the datatype, both customer and prospect records are returned.

## Date

Use the Date datatype to specify date information.

```

<Date>
  <year/> Year.
  <month/> Month.
  <day/> Day.
  <hour/> Hour.
  <minute/> Minute.
  <second/> Second.
</Date>

```

## Deal

Use the Deal datatype 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.

```

<Deal>
  <id/> Unique ID. Automatically assigned by the system.
  <closed/> When this deal was closed.
  <stage/> The % of the work complete for this deal.
  <userid/> The ID of the associated user.
  <status/> The status for this deal: O - Open, C - Closed, or
  L - Lost.
  <name/> The name/description of the deal.
  <territoryid/> The territory for this deal.
  <active/> Is this record active?
  <rating/> The rating for this deal.
  <created/> Time the record was created.
  <opened/> When this deal was first opened.

```

```

    <notes/> Notes for this deal.
    <customerid/> The ID of the associated customer.
    <exported/> Date and time the record was marked as exported.
    <updated/> Time the record was last updated or modified.
  </Deal>

```

This datatype supports the read [XML Commands](#).

## Dealcontact

Use the Dealcontact datatype to specify contact information for a deal.

```

<Dealcontact>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <contactid/> The related contact.
  <dealid/> The deal ID.
  <updated/> Time the record was last updated or modified.
</Dealcontact>

```

This datatype supports the read [XML Commands](#).

## Dealschedule

Use the Dealschedule datatype 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.

```

<Dealschedule>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <amount/> The amount this portion of the deal is worth (in the
    currency of the deal). Dated by the date/> field.
  <dealid/> ID of the deal associated with this deal portion.
  <date/> The potential closing date for a deal portion.
  <updated/> Time the record was last updated or modified.
</Dealschedule>

```

This datatype supports the read [XML Commands](#).

## Department

Use the Department datatype to specify department information and associate a user with a department.

```

<Department>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <userid/> The user ID of the head of the department.

```

```
<notes/> Notes about the department.  
<name/> The name used for display in lists.  
<updated/> Time the record was last updated or modified.  
<externalid/> If the record was imported from an external  
system, you store the unique external record ID here.  
<picklist_label/> Label as shown on form picklist.  
</Department>
```


This datatype supports the read, add, modify, and delete [XML Commands](#).

## Entitytag

Use the Entitytag datatype to specify entity tag information.

```
<Entitytag>  
  <id/> Unique ID. Automatically assigned by the system.  
  <default_for_entity/> A 1/0 field indicating whether this is the  
  default row for this entity.  
  <userid/> The ID of the associated user.  
  <projectid/> The ID of the associated project.  
  <created/> Time the record was created.  
  <tag_group_attributeid/> The ID of the associated  
  tag_group_attribute.  
  <end_date/> End date for this entity_tag.  
  <customerid/> The ID of the associated customer.  
  <updated/> Time the record was last updated or modified.  
  <start_date/> Start date for this entity_tag.  
  <tag_group_attribute_name/> The name of the associated tag group  
  attribute.  
  <tag_group_id/> The ID of the associated tag group attribute.  
</Entitytag>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

 **Note:** You can use Entitytags in a special way with the read command. Refer to the following examples:

tag_with_name="1"	attribute to return the name
between_date='2008-07-01'	attribute specifies the date for which to retrieve the current entity tag record
between_date='0000-00-00'	returns all entries with no start and no end dates

## Envelope

Use the Envelope datatype to specify information about tickets in an envelope. Envelopes are used to group individual receipts into an expense report.

```
<Envelope>  
  <id/> Unique ID. Automatically assigned by the system.
```

```

<totreimburse/> The total amount of reimbursable expenses in the
envelope.
<advance/> The amount of any cash advance on the envelope.
<number/> The envelope tracking number.
<date/> The date of the envelope.
<userid/> The ID of the associated user.
<status/> The status of the envelope (O - open, S - submitted, A
- approved, R - rejected).
<currency/> The currency this envelope is in.
<tottickets/> The total number of tickets in the envelope.
<trip_reason/> The reason for the trip.
<approver/> The userid of the envelope approver.
<date_start/> Starting date of the envelope (only used with
auto-naming).
<updated/> Time the record was last updated or modified.
<date_end/> The ending date of the envelope (only used with autonaming).
<name/> The name of the envelope.
<submitted/> The date the envelope was submitted.
<total/> The total value of all the tickets in the envelope.
<tax_locationid/> Default tax location for this envelope.
<created/> Time the record was created.
<approved/> The date the envelope was approved.
<balance/> The outstanding balance on the envelope.
<notes/> Notes about the envelope.
<is_overlapping/> Read only flag returns is an envelope overlaps
with another envelope.
<attachmentid/> If non-zero, the attachment record associated
with this envelope.
<externalid/> If the record was imported from an external
system, you store the unique external record ID here.
<currency_exchange_intolerance/> A 1/0 field indicating if the
record is within the specified foreign currency tolerance as
defined in database data definitions.
<thin_client_id/> Used by thin clients to reconcile imported
records.
<acct_date/> The accounting period date of the envelope.
</Envelope>

```

This datatype supports the read, add, modify, submit, and delete [XML Commands](#).

**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 [Troubleshooting](#) for instructions on how to create a support ticket.

## Error

Use the Error datatype to specify information about an error.

```

<Error>
  <comment/> Additional comments.
  <text/> Text of the error.
  <code/> Error code returned by the API.

```

```
</Error>
```

This datatype supports the read [XML Commands](#).

## Estimate

Use the Estimate datatype 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.

```
<Estimate>
  <id/> Unique ID. Automatically assigned by the system.
  <hide_expense/> A 1/0 field indicating if expenses should be
  hidden in analysis report.
  <dealid/> The ID of the associated deal.
  <name/> The short description for the estimate.
  <created/> Time the record was created.
  <notes/> Notes about the estimate.
  <customerid/> The ID of the associated customer.
  <updated/> Time the record was last updated or modified.
</Estimate>
```

This datatype supports the read [XML Commands](#).

## Estimateadjustment

Use the Estimateadjustment datatype to specify estimate adjustment records. Estimate adjustments are the adjustment records associated with particular estimates.

```
<Estimateadjustment>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <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.
  <estimateid/> The ID of the associated estimate.
  <name/> The name for the estimate adjustment.
  <updated/> Time the record was last updated or modified.
  <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_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.
</Estimateadjustment>
```

This datatype supports the read [XML Commands](#).

## Estimateexpense

Use the Estimateexpense datatype to specify estimate expense records.

```

<Estimateexpense>
  <id/> Unique ID. Automatically assigned by the system.
  <estimateid/> The ID of the associated estimate.
  <itemid/> The ID of the associated expense item.
  <date/> Date for the expense.
  <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.
  <quantity/> The quantity for the expense.
  <description/> The short description for the estimate.
  <created/> Time the record was created.
  <phaseid/> The ID of the associated estimate phase.
  <markup/> The amount of markup in percent or money as designated
  by markup_type field. Dated by the date field.
  <price/> The cost of the expense. Dated by the date field.
  <updated/> Time the record was last updated or modified.
</Estimateexpense>

```

This datatype supports the read [XML Commands](#).

## Estimatelabor

Use the Estimatelabor datatype to specify estimate staffing records.

```

<Estimatelabor>
  <id/> Unique ID. Automatically assigned by the system.
  <estimateid/> The ID of the associated estimate.
  <loaded_cost/> The loaded cost for the associated resource.
  Dated by the start_date field.
  <userid/> The ID of the associated resource.
  <description/> The short description for the estimate.
  <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.
  <created/> Time the record was created.
  <amount/> The number of hours or percentage of time associated
  with a given resource for a specific phase of an estimate. The
  actual type is identified by as_percentage field.
  <phaseid/> The ID of the associated estimate phase.
  <end_date/> End date for resource assignment.
  <billing_rate/> The billing rate for the associated resource.
  Dated by the start_date field.
  <start_date/> Start date for resource assignment.
  <updated/> Time the record was last updated or modified.
</Estimatelabor>

```

This datatype supports the read [XML Commands](#).

## Estimatemarkup

Use the Estimatemarkup datatype to specify information about phases for the estimate.

```

<Estimatemarkup>
  <id/> Unique ID. Automatically assigned by the system.
  <estimateid/> The ID of the associated estimate.
  <percent/> The percentage markup to add to the total expense
  amount.
  <total/> The amount of expense (in the currency of the estimate)
  to use for this estimate in calculations.
  <created/> Time the record was created.
  <phaseid/> The ID of the associated estimate phase.
  <updated/> Time the record was last updated or modified.
  <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.
</Estimatemarkup>

```

This datatype supports the read [XML Commands](#).

## Estimatephase

Use the Estimatephase datatype to specify information about phases for the estimate.

```

<Estimatephase>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <estimateid/> The ID of the associated estimate.
  <name/> The name for the estimate adjustment.
  <updated/> Time the record was last updated or modified.
</Estimatephase>

```

This datatype supports the read [XML Commands](#).

## Event

Use the Event datatype to specify information about events. An event is a historical record of an activity performed on behalf of a client or prospect. It could record the completion of a todo, the closing of a deal, and even a phone call or email message sent to a customer.

```

<Event>
  <id/> Unique ID. Automatically assigned by the system.
  <contact_id/> The ID of the associated contact.
  <userid/> The ID of the user who created the event.
  <dealid/> The ID of the associated deal.
  <name/> The name or description of the event.
  <occurred/> The date of the event.
  <created/> Time the record was created.
  <notes/> Notes related to the event.
  <updated/> Time the record was last updated or modified.
  <customerid/> The id of the associated customer.
</Event>

```

This datatype supports the read, add, and modify [XML Commands](#).

## ExpensePolicy

Use this datatype to specify information about expense policies.

```
<ExpensePolicy>
  <id/> Unique ID. Automatically assigned by the system.
  <customerid/> The ID of the associated customer.
  <projectid/> The ID of the project which expense policy
  is associated to. If zero/null then this is company
  default expense policy.
  <description/> Optional information about expense policy.
  <deleted/> A "1/0" field indicated if the record was deleted.
  <created/> Time the record was created.
  <updated/> Time the record was last modified.
  <all_items_allowed/> A "1/0" field indicating that all expense
  items are allowed by this expense policy.
</ExpensePolicy>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## ExpensePolicyItem

Use this datatype to specify information about items allowed for an expense policy.

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

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Filter

Use the Filter datatype to limit the user to a subset of a certain kind of account data.

For more information on it's use, refer to the [Read, all](#) command. Currently, only the customer list can be filtered, using the type [Customer](#) as an attribute of the <Filter/> datatype. Refer to the following example:



```
<Filter type="customer">
  <id/> the customer ID
</Filter>
```

## Filterset

Use the Filterset datatype to list names and IDs that define the table/id pairs to be filtered for each filterset.

```
<Filterset>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The name of the filterset.
  <notes/> Notes related to the filterset.
  <all_access/> A 1/0 field indicating this filterset does not
  filter anything and can not be deleted.
  <default_filter_set/> A 1/0 field indicating whether this is the
  default new-user filterset.
  <active/> A 1/0 field indicating whether this is designated as an
  active filter set.
  <externalid/> If the record was imported from an external system
  you store the unique external record id here.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
</Filterset>
```

This datatype supports the read [XML Commands](#).

## Flag

Use the Flag datatype to customize the appearance of the product using switches and settings.

```
<Flag>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The name of the switch.
  <setting/> The value to which the switch is set.
</Flag>
```

This datatype supports the read, add, and modify [XML Commands](#). Also refer to the [Company](#) and [User](#) datatypes.

## ForexInput

Use the ForexInput datatype to allow multi-currency accounts to override historical and future currency conversion rates.

```
<ForexInput>
  <symbol/> Currency symbol. Must be for one of the multiple
  currencies enabled in the account.
  <startdate/> Optional start date for currency being set.
```

```

<enddate/> Optional end date for currency being set.
<rate/> Rate against the base currency for the account.
<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.
<base/> The currency symbol used as a base currency for the
currency conversion table.
<created/> Date the record was created.
<updated/> Date the record was last modified.
</ForexInput>

```

This datatype supports the read, add, and modify [XML Commands](#).



**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 [Troubleshooting](#) for instructions on how to create a support ticket.

## FormPermissionField

This datatype is for internal use only.

```

<FormPermissionField>
  <form_name/> Internal GUI form name.
  <field_name/> Internal GUI field name.
  <readonly/> A 1/0 field indicating whether this is to be readonly
in the GUI.
  <required/> A 1/0 field indicating whether this is to be required
in the GUI.
  <default_value/> Value to be prefilled when a new form is
created.
  <hidden/> A 1/0 field indicating whether this is to be hidden in
the GUI.
  <save_and_create/> List of field names prefilled with previous
saved values.
</FormPermissionField>

```

This datatype supports the read [XML Commands](#).

## Fulfillment

Use the Fulfillment datatype to specify information about the receipt of goods and services ordered by a purchase order.

```

<Fulfillment>
  <id/> Unique ID. Automatically assigned by the system.
  <purchaseorder_id/> Associated purchase order ID.
  <purchaserequest_id/> Associated purchase request ID.
  <request_item_id/> Associated request item ID.
  <carrier_id/> Associated carrier ID.

```

```

<slip_id/> The ID of the associated slip if this expense was
billed to a time bill.
<purchase_item_id/> Associated purchase item ID.
<waybill_number/> The waybill number.
<date/> Date of the fulfillment.
<acct_date/> The accounting period date of the fulfillment.
<quantity/> The quantity received.
<notes/> Fulfillment description notes.
<created/> Time the record was created.
<updated/> Time the record was last updated or modified.
</Fulfillment>

```

This datatype supports the read [XML Commands](#).

## Hierarchy

Use the Hierarchy datatype to specify hierarchy information.

```

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

```

This datatype supports the read, add, and modify [XML Commands](#).

## HierarchyNode

Use the HierarchyNode datatype to specify information about a hierarchy node.

```

<HierarchyNode>
  <id/> Unique ID. Automatically assigned by the system.

```

```

<hierarchyid/> The ID of the associated hierarchy.
<levelid/> The id of the associated hierarchy level.
<isalevel/> A 1/0 field indicating if this node is a level.
<created/> Time the record was created.
<recordid/> The record ID if not a node.
<name/> The hierarchy name.
<isanode/> The name of the hierarchy node.
<updated/> Time the record was last updated or modified.
<parentid/> The hierarchy_node id of our immediate ancestor. If
zero/null, is a top-level node.
<externalid/> If the record was imported from an external
system, you store the unique external record ID here.
<notes/> Notes related to the hierarchy node.
</HierarchyNode>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## History

Use the History datatype to specify history events.

```

<History>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <userid/> The ID of the user associated with this history event.
  <notes/> Notes associated with the history event.
  <envelopeid/> The ID of the associated envelope.
  <date/> The date associated with this history event.
  <action/> The approval action: S - Submittal, P - Pending,
  A - Acceptance, R - Rejection, U - Unapproval.
</History>

```

This datatype only supports the [Read, equal to](#) command.

## ImportExport

Use the ImportExport datatype to specify table and id pairs corresponding to an external application. It can be used in conjunction with read, all and the not-exported filter to request records that have not been exported.

```

<ImportExport>
  <id/> Internal ID of the actual record (slip, task, etc.) in its
  native table.
  <application/> String describing the application making the
  association.
  <exported/> Time of the last export from OpenAir. Required on
  import.
  <type/> XML Datatype name of the exported record: Slip, Task,
  Projectassign, etc. Please note that these names are case
  sensitive.
  <externalid/> External identifier for the application.
  <imported/> Time of the last import to OpenAir. Required on

```

```
import.
</ImportExport>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Invoice

Use the Invoice datatype to specify invoice information for the header.

```
<Invoice>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <draw_date/> The date of the draw.
  <number/> The invoice number.
  <status/> The status of the invoice (EZ Invoice, emailed
  Invoice): 0 - Unknown, 1 - Not Sent, 2 - Viewed, 3 - EZ
  Requested, 4 - Rejected, 5 - Sent, 6 - EZ Sent, and 7 -
  Retracted.
  <date/> The date of the invoice.
  <terms/> Payment terms for this invoice.
  <invoice_layoutid/> The ID of the associated invoice layout.
  <credit_reason/> The reason for the credit.
  <currency/> The currency this invoice is in.
  <tax_state/> The state 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.
  <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.
  <tax_pst/> The PST tax for the invoice. Dated by the date field.
  draw The amount of any draw against retainer for this invoice.
  Dated by the draw_date field.
  <draw/> The amount of any draw against retainer for this invoice.
  Dated by the draw_date field.
  <contactid/> The contact id for this invoice.
  <shipping_contactid/> The shipping contact id for this invoice.
  <approval_status/> The approval status of the invoice, only used
  if invoice approvals are used: 0 - Open, S - Submitted, A -
  Approved, and R - Rejected.
  <access_log/> The mailing and access history of the invoice,
  such as when the customer accessed it.
  <credit/> The amount of any credit against the invoice. Dated by
  the date field.
  <tax_hst/> The HST tax for the invoice. Dated by the date field.
  <tax/> The tax total for the invoice. Dated by the date field.
  <total/> The invoice total. Dated by the date field.
  <updated/> Time the record was updated.
  <balance/> The outstanding balance on the invoice. Dated by the
  date field.
  <notes/> Notes associated with the invoice.
  <paperrequest/> Date the user requested that a paper invoice be
  mailed.
```

```

<customerid/> The id of the associated customer.
<accounting/> A 1/0 field indicating if an invoice has been sent
to an accounting partner.
<papersend/> Date the paper invoice was actually mailed.
<acct_date/> The accounting period date of the invoice.
<externalid/> If the record was imported from an external
system, you store the unique external record ID here.
<credit_rebill_status/> Credit/Rebill status for the original
invoice: C = Credit Initiated and R = Re-Bill.
<original_invoiceid/> The original invoice ID for credit
invoices.
<attachmentid/> The ID of the associated attachment.
</Invoice>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

**Note:** 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 your Professional Services Consultant or create a support ticket. See [Troubleshooting](#) for instructions on creating a support ticket.

## InvoiceLayout

Use the InvoiceLayout datatype to read invoice layout information.

```

<InvoiceLayout>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> Name The name used for display in popups and lists.
  <created/> Created Time the record was created.
  <updated/> Updated Time the record was last modified.
</InvoiceLayout>

```

This datatype supports the read [XML Commands](#).

## Issue

Use the Issue datatype to specify issue information.

```

<Issue>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <number/> The issue number that increments by 1.
  <prefix/> A static alphanumeric issue number prefix.
  <name/> The name of the issue (Prefix + number).
  <owner_id/> The ID of the associated user creating the issue.
  <description/> A short description of the issue, a synopsis.
  <customer_id/> The ID of the associated customer.
  <project_id/> The ID of the associated project.
  <project_task_id/> The ID of the task within the associated
project.
  <issue_category_id/> The ID of the associated issue category.

```

```

<issue_status_id/> The ID of the associated issue status.
<issue_stage_id/> The ID of the associated issue stage.
<issue_severity_id/> The ID of the associated issue severity.
<issue_source_id/> The ID of the associated issue source.
<issue_notes/> The description of the issue.
<resolution_notes/> The description of the resolution.
<date/> The date of the issue.
<date_resolution_required/> The date the issue must be resolved.
<date_resolution_expected/> The date the issue is expected to be
resolved.
<date_resolved/> The date the issue was resolved.
<user_id/> The ID of the user assigned to the issue.
<priority/> The priority of the task (1 - 100).
<attachment_id/> If non-zero, the attachment record associated
with this issue.
<updated/> Time the record was updated.
<submitted/> Date the invoice was submitted.
<approved/> Date the invoice was approved.
</Issue>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## IssueCategory

Use the IssueCategory datatype to specify information about the issue category.

```

<IssueCategory>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <name/> The name of the issue category.
  <active/> A 1/0 field indicating whether this issue category is
active.
  <notes/> Notes associated with the issue category.
  <updated/> Time the record was last updated or modified.
</IssueCategory>

```

This datatype supports the read, add, and modify [XML Commands](#).

## IssueSeverity

Use the IssueSeverity datatype to specify information about the issue severity.

```

<IssueSeverity>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <name/> The name of the issue severity.
  <active/> A 1/0 field indicating whether this issue severity is
active.
  <notes/> Notes associated with the issue severity.
  <updated/> Time the record was last updated or modified.
</IssueSeverity>

```

This datatype supports the read, add, and modify [XML Commands](#).

## IssueSource

Use the IssueSource datatype to specify information about the issue source.

```
<IssueSource>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <name/> The name of the issue source.
  <active/> A 1/0 field indicating whether this issue source is
  active.
  <notes/> Notes associated with the issue source.
  <updated/> Time the record was last updated or modified.
</IssueSource>
```

This datatype supports the read, add, and modify [XML Commands](#).

## IssueStage

Use the IssueStage datatype to specify information about the issue stage.

```
<IssueStage>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <name/> The name of the issue stage.
  <default_for_new/> A 1/0 field indicating whether this is the
  default stage for new issues.
  <considered_closed/> A 1/0 field indicating whether issues in
  this stage are considered closed.
  <position/> The position of the stage.
  <notes/> Notes associated with the issue stage.
  <updated/> Time the record was last updated or modified.
</IssueStage>
```

This datatype supports the read, add, and modify [XML Commands](#).

## IssueStatus

Use the IssueStatus datatype to specify information about the issue status.

```
<IssueStatus>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The name of the issue status.
  <active/> A 1/0 field indicating if this issue status is active.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
</IssueStatus>
```

This datatype supports the read, add, and modify [XML Commands](#).



## Item

Use the Item datatype to specify item information such as an expense item, expense type, expense, and inventory item.

```
<Item>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <cost/> The default cost per unit of measure for the item. 3
  decimal places to handle items like mileage at 32.5 cents.
  <active/> A 1/0 field indicating whether this is designated as an
  active customer.
  <taxable/> A 1/0 field indicating whether this item is taxable,
  VAT-able, etc.
  <externalid/> If the record was imported from an external system
  you store the unique external record ID here.
  <unitm/> The unit of measure for the item, i.e., EA.
  <currency/> Currency for the money fields in the record.
  <cost_centerid/> The id of the associated cost center.
  <name/> The item name.
  <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 and M - for mileage item.
  <code/> Optional accounting system code for integration with
  external accounting systems.
  <updated/> Time the record was last updated or modified.
  <tp_cost/> The policy threshold amount.
  <tp_comp/> Ticket policy comparison:
  ge - greater than or equal to
  gt - greater than
  <tp_notes_required/> Notes are required if the ticket triggers
  the policy.
  <tax_location_id/> The ID of the associated tax location.
  <tp_unit_or_total/> The ticket policy is applied against:
  U - Unit price
  T - Total
  <cost_is_fixed/> A 1/0 field indicating whether the user is
  allowed to change the cost on a receipt
  <picklist_label/> Label as shown on form picklist.
</Item>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## ItemToUserLocation

Use the ItemToUserLocation datatype to specify associations between Item, UserLocation, and TaxLocation.

```
<ItemToUserLocation>
  <id/> Unique ID. Automatically assigned by the system.
  <itemid/> The ID of the associated item.
  <tax_locationid/> The ID of the associated tax location.
```

```

<user_locationid/> The location ID for this user.
<created/> Time the record was created.
<updated/> Time the record was last updated or modified.
</ItemToUserLocation>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Jobcode

Use the Jobcode datatype to specify job code information.

```

<Jobcode>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <userid_fte/> The user ID of the FTE (Full Time Equivalent)
  generic resource.
  <loaded_cost/> Loaded cost for this job code.
  <name/> The name for the job code.
  <updated/> Time the record was last updated or modified.
  <notes/> Notes associated with the job code.
  <externalid/> If the record was imported from an external system
  you store the unique external record id here.
  <currency/> Currency for the money fields in the record.
  <code/> Optional accounting system code for integration with
  external accounting systems.
  <active/> A 1/0 field indicating if this is an active job code.
</Jobcode>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Leave\_accrual\_rule

Use the Leave\_accrual\_rule datatype to specify leave accrual rule information.

```

<Leave_accrual_rule>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <project_task_filter/> CSV list of project_tasks that will
  trigger a draw down.
  <category_filter/> CSV list of categories that will trigger a
  draw down.
  <lose_how/> How is accrued time lost: N - Never, A - The users
  anniversary date, and Y - End of year.
  <cap/> Number of hours to cap the accrual at.
  <timetype_filter/> CSV list of timetypes that will trigger a
  draw down.
  <project_filter/> CSV list of projects that will trigger a draw
  down.
  <period/> The period for the cap.
  <draw_down_when/> Generate the draw down when: R - When leave
  accrual is run or A - When a timesheet is approved.

```

```

<notes/> Notes associated with the leave accrual rule.
<active/> A 1/0 field indicating whether this is an active
billing rule.
<name/> The name for the leave accrual rule.
<updated/> Time the record was last updated or modified.
<grace_days/> How many days is the grace period before accrued
time is lost.
<timing/> When the accrual is applied: S - start of the period or
E - end of the period.
<amount/> The number of hours per period.
</Leave_accrual_rule/>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Leave\_accrual\_rule\_to\_user

Use the Leave\_accrual\_rule\_to\_user datatype to map leave accrual rules to users.

```

<Leave_accrual_rule_to_user>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <userid/> The id of the associated user.
  <end_date/> The date the accrual rule stops applying to the user.
  <start_date/> The date the accrual rule starts applying to the
user. This is required.
  <updated/> Time the record was last updated or modified.
  <transfer_balance_to/> ID of leave_accrual_rule_to_user record
where balance should be transferred to.
  <leave_accrual_ruleid/> The id of the associated accrual rule.
</Leave_accrual_rule_to_user>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Leave\_accrual\_transaction

Use the Leave\_accrual\_transaction datatype to specify leave accrual transaction information.

```

<Leave_accrual_transaction>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <date/> The date of the transaction.
  <taskid/> The id of the associated task if this is a draw down
against a timesheet entry.
  <notes/> Notes associated with the leave accrual transaction.
  <updated/> Time the record was last updated or modified.
  <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.
  <type/> Indicates type of draw down: the type of the amount
field. D - draw down or A - Accrual.
  <from_run/> Indicates if this was generated from a run the leave

```

```

    accrual rules.
    <userid/> The ID of the associated user.
    <leave_accrual_ruleid/> The ID of the associated accrual rule.
    This is a required field.
  </Leave_accrual_transaction>

```

This datatype supports the read, add, and modify [XML Commands](#).

## LoadedCost

Use the LoadedCost datatype to specify loaded cost values for a user.

```

<LoadedCost>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <userid/> ID of the user.
  <projectid/> The id if this loaded cost is associated with a
    specific project.
  <externalid/> If the record was imported from an external system
    you store the unique external record ID here.
  <end/> End date for the loaded cost for historical records.
  <updated/> Time the record was last updated or modified.
  <currency/> The currency of the cost field.
  <customerid/> The ID of the associated customer.
  <current/> A 1/0 field indicating if this is the current loaded
    cost record.
  <project_taskid/> The id if this loaded cost is associated with a
    specific project task. If this field is used, the project_id and
    customer_id must be empty.
  <cost/> The fully loaded hourly cost of the user.
  <start/> Start date for the loaded cost for historical records.
  <lc_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
</LoadedCost>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Login

Use the Login datatype to authenticate a user into an account. Many commands require a valid <Auth/>, and hence a <Login/> before being executed. They include the read, modify, submit, and delete commands.

```

<Login>
  <company/> The nickname of the company.
  <user/> The nickname of the user.
  <password/> The password for the user.
</Login>

```

## Module

Use the Module datatype to specify Module availability.

```
<Module>
  <abbr/> Abbreviation within OpenAir.
  <enabled/> A 1/0 field indicating whether the module is enabled.
</Module>
```

This datatype supports the read [XML Commands](#).

## Notes

Use the Notes datatype to store partner-specific information on the OpenAir system. You might store partner-specific preferences. Refer to the following example:

```
<Notes>
  <name/> Name.
  <user_id/> ID of the user.
  <setting/> Setting information.
  <created/> Time the record was created.
  <updated/> Time the record was last modified.
</Notes>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Payment

Use the Payment datatype to specify payment information.

```
<Payment>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <date/> The date of the payment.
  <invoiceid/> The associated invoice ID if a payment against a
    specific invoice.
  <notes/> Notes associated with the payment.
  <updated/> Time the record was last updated or modified.
  <total/> The payment total. Dated by the date field.
  <invoice_number/> The associated invoice number if a payment
    against a specific invoice.
  <currency/> Currency for the money fields in the record.
  <customerid/> The ID of the associated customer if this is a
    retainer payment.
  <bulk_paymentid/> The ID of the bulk_payment transaction if this
    payment is part of a bulk_payment.
  <externalid/> If the record was imported from an external system
    you store the unique external record id here.
</Payment>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Paymentterms

Use the Paymentterms datatype to specify payment terms information.

```
<Paymentterms>
  <id/> Unique ID. Automatically assigned by the system.
  <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).
  <name/> The payment terms name.
  <notes/> Notes associated with the payment terms.
  <updated/> Time the record was last updated or modified.
  <active/> A 1/0 field indicating where this is designated as an
    active shipping terms 1/0.
  <default_status/> Default receipt status, e.g. R =>
    Reimbursable, N => Non-reimbursable.
  <default_payment_type/> A "1/0" field indicating whether this is
    the default payment_type for receipts.
</Paymentterms>
```

This datatype supports the read, add, and modify [XML Commands](#).

## Paymenttype

Use the Paymenttype datatype to specify payment type information. Payment types are used to specify the payment methods for individual receipts.

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

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Payrolltype

Use the Payrolltype datatype to specify payroll type information.

```
<Payrolltype>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <active/> A 1/0 field specifying whether this is an active
    payrolltype.
```

```

<externalid/> If the record was imported from an external
system, you store the unique external record ID here.
<notes/> Notes associated with the payroll type.
<name/> The name of the payroll type.
<updated/> Time the record was last updated or modified.
<picklist_label/> Label as shown on form picklist.
</Payrolltype>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## PendingBooking

Use the PendingBooking datatype to read pending booking records.

```

<PendingBooking>
  <userid/> The ID of the associated user.
  <startdate/> The start date of the booking.
  <repeatid/> The ID of the associated repeating event.
  <booking_typeid/> The ID of the associated booking_type.
  <notify_owner/> A 1/0 field indicating whether to send email to
the requestor when the booking is modified.
  <date_approved/> The date the booking request was approved.
  <starttime/> Start time.
  <enddate/> The end date of the booking.
  <updated/> Time the record was last updated or modified.
  <endtime/> End time.
  <id/> Unique ID. Automatically assigned by the system.
  <project_taskid/> The ID of the task within the assoc. project.
  <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.
  <date_submitted/> The date the booking_request was submitted.
  <ownerid/> The ID of the associated user creating the booking.
  <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.
  <approval_status/> The approval status of the booking request (0
- Open, S - Submitted, A - Approved, R - Rejected).
  <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.
  <projectid/> The ID of the associated project.
  <job_code_id/> The ID of the associated job code.
  <externalid/> If the record was imported from an external system
you store the unique external record ID here.
  <resource_request_queue_id/> The id of the associated resource
request queue.
  <created/> Time the record was created.
  <locationid/> The location ID for this booking.
  <notes/> Booking notes.
  <customerid/> The ID of the associated customer.
  <project_assignment_profile_id/>The id of the associated project
assignment profile.

```

```
</PendingBooking>
```

This datatype supports the read [XML Commands](#).

## Preference

Use the Preference datatype to specify preference or setting information.

```
<Preference>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <name/> The name for the preference.
  <updated/> Time the record was last updated or modified.
  <group_name/> Optional group name for the preference.
  <userid/> If the preference is for a user, this is the user ID.
  <setting/> The preference data is stored in this field.
</Preference>
```

This datatype supports the read, add, and modify [XML Commands](#).

## Product

Use the Product datatype to specify product information. Products are used to create request items, which ultimately appear as line items on purchase orders.

```
<Product>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <manufacturerid/> The manufacturer of this product.
  <um/> The unit of measure for the product, i.e., EA.
  <name/> The name for the product. This shows up on all the
  product pop-up windows in the application.
  <updated/> Time the record was last updated or modified.
  <currency/> The currency this cost is quoted in.
  <code/> Optional accounting system code for integration with
  external accounting systems.
  <manufacturer_part/> The manufacturer's part number, SKU or
  other unique identification for this product.
  <active/> A 1/0 field indicating that this is active.
  <taxable/> A 1/0 field indicating whether this item is taxable.
  <externalid/> If the record was imported from an external
  system, you store the unique external record id here.
  <vendor_sku/> The preferred vendor's sku for this product.
  <vendor_id/> The preferred vendor from whom to purchase this
  product.
  <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.
</Product>
```

This datatype supports the read, add, and modify [XML Commands](#).



## Project

Use the Project datatype to specify project information and to create one project from another project. Indicate the rules and settings to copy.

```
<Project>
  <id/> Unique ID. Automatically assigned by the system.
  <user_filter/> Also allow these users to edit the project if the
    only_owner_can_edit switch is on.
  <message/> Dashboard message.
  <auto_bill/> A 1/0 field, 1 if the project can be auto-billed.
  <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 project owner's manager,
    If -2 then the approver is the project owner's manager's manager,
    If -3 then the approver is the project owner, and
    If -4 then the approver is self.
  <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 project owner's manager,
    If -2 then the approver is the project owner's manager's manager,
    If -3 then the approver is the project owner, and
    If -4 then the approver is self.
  <auto_bill_cap/> A 1/0 field, 1 if the project should have a cap
    on auto-billings.
  <invoice_layoutid/> The ID of the associated invoice layout.
  <category_filter/> A category (service) filter. This will hold a
    list of the categories that are allowed to book time to this
    project.
  <rate/> The hourly billing rate.
  <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.
  <sync_workpace/> A 1/0 field indicating whether to keep project
    resources in sync with linked workspace members.
  <notify_owner/> A 1/0 field indicating whether to send email to
    the project owner when an ownership change is made.
  <br_approvalprocess/> The approvalprocess_id of the project
    booking request approval process. This field is mutually
    exclusive with br_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 project owner's manager,
    If -2 then the approver is the project owner's manager's manager,
    If -3 then the approver is the project owner, and
    If -4 then the approver is self.
  <ta_approvalprocess/> The approvalprocess_id of the project
    timesheet approval process. This field is mutually exclusive
    with ta_approver.
  <te_approvalprocess/> The approvalprocess_id of the project
    expense report approval process. This field is mutually
    exclusive with te_approver.
```

<auto\_bill\_cap\_value/> The auto-billings cap amount (in the currency of the project).

<updated/> Time the record was last updated or modified.

<code/> Optional system code for integration with external accounting systems.

<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 project owner's manager,  
 If -2 then the approver is the project owner's manager's manager,  
 If -3 then the approver is the project owner, and  
 If -4 then the approver is self.

<timetype\_filter/> A timetype filter. This will hold a list of the timetypes that are allowed to book time to this project.

<tax\_location\_name/> Name of the tax location.

<active/> A 1/0 field indicating an active project.

<name/> The project name. This shows upon all the project pop-up windows in the application.

<hierarchy\_node\_ids/> The ID of the hierarchy node for this project.

<externalid/> If the record was imported from an external system, you store the unique external record ID here.

<po\_approvalprocess/> The approvalprocess\_id of the project purchase order approval process. This field is mutually exclusive with po\_approver.

<project\_stageid/> The ID of the project stage.

<tax\_locationid/> The ID of the associated tax location.

<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 project owner's manager,  
 If -2 then the approver is the project owner's manager's manager,  
 If -3 then the approver is the project owner, and  
 If -4 then the approver is self.

<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 project owner's manager,  
 If -2 then the approver is the project owner's manager's manager,  
 If -3 then the approver is the project owner, and  
 If -4 then the approver is self.

<locationid/> The location ID for this project (DEPRECATED).

<finish\_date/> The calculated finish date of the project.

<misp\_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.

<customerid/> The ID of the associated customer.

<customer\_name/> The customer's name.

<only\_owner\_can\_edit/> A 1/0 field indicating whether only the project owner can edit this project.

<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 project owner's manager,

If -2 then the approver is the project owner's manager's manager,  
 If -3 then the approver is the project owner, and  
 If -4 then the approver is self.

<userid/> The user ID of the project owner.

<az\_approvalprocess/> The approvalprocess\_id of the project expense authorization approval process. This field is mutually exclusive with az\_approver.

<project\_locationid/> The location ID for this project.

<budget/> The budgeted revenue for the project.

<currency/> Currency for the money fields in the record.

<cost\_centerid/> The ID of the associated cost center.

<sga\_labor/> The allocated cost (SG and A) overhead percentage to apply to labor for profitability analysis.

<invoice\_text/> Text to display on every invoice.

<budget\_time/> The budgeted amount of time for the project, in hours.

<start\_date/> The scheduled starting date of the project.

<pr\_approvalprocess/> The approvalprocess\_id of the project purchase request approval process. This field is mutually exclusive with pr\_approver.

<billing\_contactid/> The billing contact ID if different than the customer designated billing contact.

<billing\_code/> The project billing code. Used in bulk invoicing.

<created/> Time the record was created.

<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.

<create\_workspace/> A 1/0 field, 1 if an associated workspace is automatically created by the API. The project owner becomes the workspace owner.

<tb\_approvalprocess/> The approvalprocess\_id of the project invoice approval process. This field is mutually exclusive with tb\_approver.

<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.

<template\_project\_id/> ID of the project from which tasks and phases, billing rules, revenue recognition rules and other items will be copied.

<copy\_revenue\_recognition\_rules/> Duplicates revenue recognition rules.

<copy\_revenue\_recognition\_auto\_settings/> Duplicates revenue recognition rules auto-run settings.

<copy\_project\_billing\_rules/> Duplicates project billing rules.

<copy\_project\_billing\_auto\_settings/> Duplicates project autobill settings.

<copy\_project\_pricing/> Duplicates project pricing information.

<copy\_custom\_fields/> Duplicates custom fields.

<copy\_loaded\_cost/> Duplicates project pricing information.

<copy\_approvers/> Duplicates project approvers.

<copy\_issues/> Duplicates issues.

<copy\_notification\_settings/> Duplicates notification settings.

<copy\_dashboard\_settings/> Duplicates dashboard settings.

<copy\_invoice\_layout\_settings/> Duplicates invoice layout

settings.

<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.

<payroll\_type\_filter/> A payroll type filter. This holds a list of the payroll types that are allowed to book time to this project.

<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.

<filterset\_ids/> A comma separated list of filter set IDs this record should be part of.

<attachmentid/> If non-zero, the attachment record associated with this project.

<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

If -4 then the approver is self

<rv\_approvalprocess/> The approvalprocess\_id of the project revenue\_container approval process. This field is mutually exclusive with rv\_approver.

<portfolio\_projectid/> The ID of the associated portfolio project.

<is\_portfolio\_project/> A 1/0 field - 1 if the project is a portfolio project.

<copy\_revenue\_recognition\_auto\_settings/> Duplicate of copy\_revenue\_recognition\_auto\_settings.

<filtersetids/> A comma separated list of filter set IDs this record should be part of.

<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_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.
<ta_include/> A 1/0 field indicating whether a Timesheet
filterset is applied.
<te_include/> A 1/0 field indicating whether an Expense Report
filterset is applied.
<rm_approver/> The user_id of the project booking approver if
this is a single approver process.
This field is mutually exclusive with rm_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
<rm_approvalprocess/> The approvalprocess_id of the project
booking approval proces. This field is mutually exclusive with
rm_approver.
<rate_cardid/> The ID of the associated rate card if using rate
cards.
<main_contactid/> The ID of the main project contact.
<picklist_label/> Label as shown on form picklist.
</Project>

```

This datatype supports the read, add, modify, and delete [XML Commands](#). To modify a project without recalculating it, use <Project no\_recalc='1'>.

## Projectassign

Use the Projectassign datatype for the assignment by project feature to track users assigned to a project.

```

<Projectassign>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <user_id/> The ID of the user assigned to this task.
  <project_groupid/> The ID of the project group if the user was
  assigned as part of a project group.
  <updated/> Time the record was last updated or modified.
  <job_codeid/> The ID of the associated job code.
  <customer_id/> The ID of the associated customer.
  <project_id/> The ID of the project to which this user is
  assigned.
  <allocation/> The percentage of time the associated user is
  allocated to this task.
</Projectassign>

```

This datatype supports the read, add, and modify [XML Commands](#).

## ProjectAssignmentProfile

Use the ProjectAssignmentProfile datatype to assign profiles to projects.

```

<ProjectAssignmentProfile>
  <id/> Unique project_assignment_profile id. Automatically
  assigned by
  <created/> Time the record was created
  <user_filter/> A user filter list. The
  project_assignment_profile only be applied to the users in this
  list.
  <customerid/> The ID of the associated customer
  <name/> The project_assignment_profile name.
  <updated/> Time the record was last updated or modified
  the system.
  <projectid/> Id of the project to which this
  project_assignment_profile is associated.
</ProjectAssignmentProfile>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Projectbillingrule

Use the Projectbillingrule datatype to specify project billing rules.

```

<Projectbillingrule>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <name/> Name of this project billing rule.
  <user_filter/> CSV list of users to limit the rule to.
  <cap_hours/> The number of hours to cap the billing at for a time
  billing rule.
  <backout_gst/> If they are using GST/HST/PST taxes, back out the
  GST/HST taxes from re-billed expenses.
  <ticket_maximums/> Holds data on ticket maximums per expense
  type.
  <markup_type/> A field indicating the type of expense markup:
  P - percentage of the cost.
  S - specific amount.
  <percent/> The percentage value for a fixed fee percent trigger.
  <end_milestone/> The ID of the ending milestone (project_task).
  <daily_roll_to_next/> If the period cap is hit move the
  remainder to the next rule.
  <category_filter/> CSV list of categories to limit the rule to.
  <exclude_non_reimbursable/> Exclude non-reimbursable expenses.
  <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).
  <adjust_if_capped/> If a transaction will exceed the cap, should
  it be adjusted to fit under the cap.
  <slip_stageid/> The ID of the slip stage to assign to the
  transaction.
  <markup_category/> The ID of the category a markup on expense
  receipts should be assigned to.
  <timetype_filter/> CSV list of timetypes to limit the rule to.

```

<cap/> The amount to cap total billing for this rule at (in the currency of the project).  
 <daily\_cap\_period/> Period for the cap: D - day, W - week, M - month, Q - quarter, or Y - year.  
 <active/> A 1/0 field indicating whether this is an active billing rule.  
 <description/> The rule description.  
 <categoryid/> The ID of the category to assign to the transaction if it doesn't have a category.  
 <start\_milestone/> The ID of the starting milestone (project\_task).  
 <end\_date/> End date of the rule.  
 <rate\_from/> Where we get the rate from: U - Users, R - Rate cards, or C - Category.  
 <type/> The type of the billing rule: T - time billing rule, E - expense billing rule, F - fixed fee billing rule, or P - purchase billing rule.  
 <agreementid/> The ID of the associated agreement.  
 <customerpooid/> The ID of the associated customerpo.  
 <item\_filter/> CSV list of items to limit the rule to.  
 <position/> The position of the rule (0,1,2 etc.). Rules are evaluated in order and evaluation stops once a rule is satisfied.  
 <rate\_multiplier/> Optional multiplier to adjust the time billing rate by.  
 <project\_task\_filter/> CSV list of tasks to limit the rule to.  
 <rate\_cardid/> The ID of the associated rate card if using rate cards.  
 <product\_filter/> CSV list of products to limit the rule to.  
 <currency/> Currency for the money fields in the record.  
 <repeatid/> The ID of the associated repeating event.  
 <exclude\_archived\_ts/> Exclude time from archive timesheets in time billing rules.  
 <exclude\_non\_billable/> Exclude non-billable expenses.  
 <markup/> The amount of markup in percent or monetary amount as designated by markup\_type field.  
 <start\_date/> Start date of the rule.  
 <category\_when/> When the category should be applied:  
 N - Use the selected category if the time entry does not have a category.  
 A - Always use the selected category.  
 <projectid/> The ID of the associated project.  
 <stop\_if\_capped/> If a transaction is not billed because it exceeds the cap, should the billing stop for this transaction.  
 <amount/> The amount for a fixed fee rule.  
 <round\_rules/> Rules for rounding time.  
 <daily\_cap\_is\_per\_user/> Is the daily cap on a per user basis.  
 <acct\_date/> Accounting period date to assign to 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

```

M - set by the specified accounting date
P - set by the specified accounting period
<accounting_period_id/> The ID of the assoc. accounting period.
<daily_cap_hours/> The number of hours to cap the period billing
per user at.
<cost_center_id/> The ID of the associated cost center.
<notes/> Notes associated with this project billing rule.
<customerid/> The ID of the associated customer.
<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.
<job_code_filter/> CSV list of filters to limit the rule to.
<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.
<exclude_non_billable_task/> Exclude non-billable tasks.
<assigned_user/> The user to assign to fixed fee billings.
</Projectbillingrule>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Projectbillingtransaction

Use the Projectbillingtransaction datatype to specify project billing transactions.

```

<Projectbillingtransaction>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <notes/> Notes associated with this project billing rule
transaction.
  <hour/> The number of hours for a T type.
  <userid/> The ID of the associated user.
  <date/> The date of the transaction.
  <taskid/> The ID of the associated task
  <um/> The unit of measure for an E or P type.
  <rate/> The hourly rate for a T type. Dated by the date field.
  <slipid/> The ID of slip that was created.
  <ticketid/> The ID of the associated ticket.
  <project_taskid/> The ID of the associated project task.
  <project_billing_ruleid/> The ID of the associated project
billing rule.
  <cost/> The cost per unit of measure for an E type. The fixed
price for an F type. Dated by the date field.
  <itemid/> The ID of the associated item.
  <quantity/> The quantity for an E or P type.

```



```

<projectid/> The ID of the associated project.
<description/> Description associated with billing rule
transaction.
<total/> The total currency value. Dated by the date field.
<categoryid/> The ID of the associated category.
<minute/> The number of minutes for a T type.
<type/> The type of the transaction. Matches the type field in
project_billing_rule.
<slip_stage_id/> The ID of the slip stage.
<job_codeid/> The ID of the associated job code.
<customerpoId/> The ID of the associated customerpo.
<cost_centerid/> The ID of the associated cost center.
<timetypeid/> The ID of the associated time type.
<customerid/> The ID of the associated customer.
<agreementid/> The ID of the associated agreement.
<payroll_typeid/> The ID of the associated payroll type.
</Projectbillingtransaction>

```

This datatype supports the read [XML Commands](#).

## ProjectBudgetGroup

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, the system 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.

```

<ProjectBudgetGroup>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The name of the project budget group.
  <customerid/> The ID of the associated customer.
  <projectid/> The ID of the associated project.
  <date/> The date of the budget record.
  <currency/> Currency for the money fields in the record.
  <total/> The total value of the budget entry. Date set by the "date" attribute.
  <notes/> Notes associated with this project budget.
  <created/> Time the record was created.
  <updated/> Time the record was last modified.
  <total_expected_cost/> Cost total calculated from project budget transactions.
    Date set by the "date" attribute.
  <total_expected_billing/> Billing budget expected total. Date set by
    the "date" attribute.
  <total_calculated_cost/> Cost total calculated from project budget transactions.
    Date set by the "date" attribute.
  <total_calculated_billing/> Billing total calculated from project budget transactions.
    Date set by the "date" attribute.
  <total_from_funding/> A "1/0" field indicating whether to use the total from
    project funding documents.
  <profitability/> The profitability of this project budget group.

```

```

<funding_total/> Total calculated from project funding documents.
    Date set by the "date" attribute.
<internal_total/> Manually entered total. Date set by the "date" attribute.
<calculated_total/> The total calculated from project budget transactions.
    Date set by the "date" attribute (obsolete attribute).
<budget_by/> Sets the "Budget by" option:
    1 - budget by project
    2 - budget by phase
    3 - budget by task
<unassigned_task/> A "1/0" field indicating whether it is possible to create
    budget entries not connected to any particular task.
<userid/> The user ID of the budget owner.
<approval_status/> The approval status of the project budget group:
    0 - Open
    S - Submitted
    A - Approved
    R - Rejected
    X - Archived
<date_submitted/> The date the project budget group was submitted
<date_approved/> The date the project budget group was approved
<date_archived/> The date the project budget group was archived
<version/> Version of the project budget group.
<parentid/> The project budget group id of this budget's immediate ancestor. If zero or
    null, this is a top-level project budget group.
<labor_subcategory/> Labor subcategory:
    0 - category
    1 - job code
<setting/> Miscellaneous settings are stored in this field as a serialized hash
<cf_pes/> Pesimistic contingency factor for this project budget.
<cf_opt/> Optimistic contingency factor for this project budget.
</ProjectBudgetGroup>

```

This datatype supports the add, read, modify, and delete [XML Commands](#).

## ProjectBudgetRule

The ProjectBudgetRule datatype defines a line in the project budget grid. When adding a budget, the system checks the validity of the project budget group ID, and returns error 945 if invalid (see [Error Codes](#)). 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 don't have rights to delete the project budget.

```

<ProjectBudgetRule>
  <id/> Unique ID. Automatically assigned by the system.
  <project_budget_groupid/> The ID of the associated project_budget_group.
  <customerid/> The ID of the associated customer.
  <projectid/> The ID of the associated project.
  <project_taskid/> The ID of the associated project task.
  <category/> The main category for this budget line:
    1 - labor
    2 - expense item
    3 - purchase order
  <categoryid/> The ID of the associated category.
  <job_codeid/> The ID of the associated job code.
  <itemid/> The ID of the associated item.
  <productid/> The ID of the associated product.
  <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)
  <total_worst/> The worst-case estimate 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/> The total for this project budget rule. Date set by the "date" attribute.
  <quantity_worst/> The worst-case quantity estimate 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.
  <quantity/> The quantity for this project budget rule.
  <rate/> The rate of this project budget rule.
  <profitability/> The profitability of this project budget rule.
  <date/> The date of the budget rule.
  <start_date/> Start date of the period.
  <end_date/> End date of the period.
  <currency/> Currency for the money fields in the record.
  <notes/> Notes associated with this project budget line.
  <created/> Time the record was created.
  <updated/> Time the record was last modified.
  <imported/> A 0/1 field which indicates whether the rule was imported from project task
    assignments or bookings (related only to the labor category).
</ProjectBudgetRule>

```


This datatype supports the add, read, modify, and delete [XML Commands](#).

## ProjectBudgetTransaction

The ProjectBudgetTransaction datatype defines a transaction in one project budget grid line. When adding a new project budget transaction, the system checks the validity of the project budget rule, and returns error 946 if invalid (see [Error Codes](#)). 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.

```
<ProjectBudgetTransaction>
  <id/> Unique ID. Automatically assigned by the system.
  <project_budget_ruleid/> The ID of the associated project budget rule.
  <project_budget_groupid/> The ID of the associated project budget group.
  <customerid/> The ID of the associated customer.
  <projectid/> The ID of the associated project.
  <project_taskid/> The ID of the associated project task.
  <category/> The main category for this budget transaction:
    1 - labor
    2 - expense item
    3 - purchase order
  <categoryid/> The ID of the associated category.
  <job_codeid/> The ID of the associated job code.
  <itemid/> The ID of the associated item.
  <productid/> The ID of the associated product.
  <date/> The date of the project budget transaction.
  <currency/> Currency for the money fields in the record.
  <total_worst/> The worst-case estimate for this project budget transaction.
    Dated by the date field.
  <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/> The total for this budget transaction. Date set by the "date" attribute.
  <quantity_worst/> The worst-case quantity estimate 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/> Quantity for this project budget transaction.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
</ProjectBudgetTransaction>
```

This datatype supports the add, read, modify, and delete [XML Commands](#).

## Projectgroup

Use the Projectgroup datatype to document users who are assigned to a project task as a group.

```

<Projectgroup>
  <id/> Unique ID. Automatically assigned by the system.
  <assigned_users/> The users assigned to this project group. Can
  be a comma delimited list.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <notes/> Notes associated with the project group.
  <name/> The name for the project group.
  <active/> A 1/0 field indicating whether this is active.
</Projectgroup>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Projectlocation

Use the Projectlocation datatype to specify project location information.

```

<Projectlocation>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <notes/> Notes associated with the project location.
  <name/> The name for the project location.
  <active/> A 1/0 field indicating whether this is active.
</Projectlocation>

```

This datatype supports the read [XML Commands](#).

## Projectstage

Use the Projectstage datatype to specify project stage information.

```

<Projectstage>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <notes/> Notes associated with the project stage.
  <name/> The name of the project stage.
  <position/> The position of the 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.
  <enable_project_assignments/> Are project level assignments
  enabled at this stage.
  <enable_phase_and_task/> Are phases and tasks enabled at this
  stage.
  <enable_analysis/> Is financial analysis enabled at this stage.
  <enable_billing/> Is the project billing tab enabled at this
  stage.
  <enable_recognition/> Is the recognition tab enabled at this
  stage.
  <enable_pricing/> Is project pricing enabled at this stage. Off
  by default.

```

```
<picklist_label/> Label as shown on form picklist.
</Projectstage>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Projecttask

Use the Projecttask datatype to specify information about the individual tasks or work packages that comprise a project.

```
<Projecttask>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <notes/> Notes associated with the project task.
  <name/> Short description of this task.
  <priority/> The priority of the task (1 - 9).
  <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.)
  <task_budget_cost/> If task budgeting is enabled this is the
    total cost of the task.
  <is_a_phase/> A 1/0 field indicating if any other project_tasks
    have us as a parent.
  <seq/> The sequence number of this task.
  <timetype_filter/> A timetype filter. This will hold a list of
    the timetypes that are allowed to book time to this task.
  <non_billable/> If set to 1, this is not billable. This is only
    applicable for project billing rules.
  <externalid/> If the record was imported from an external system
    you store the unique external record ID here.
  <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.
  <all_can_assign/> Is everyone able to assign time/expenses to
    this task.
  <predecessors/> Comma delimited list of task IDs which must
    complete before this task can start.
  <customer_name/> The name of the associated customer.
  <parentid/> The task ID of our immediate ancestor. If zero or
    null, this is a project-level (top-level) task or phase.
  <projecttask_typeid/> The ID of the associated projecttask_type.
    Not for phases.
  <calculated_finishes/> Calculated finish date.
  <predecessors_lag/> Comma delimited list for task ID:days of lag
    time for predecessors. Only populated if there is a lag time.
  <currency/> Currency for the money fields in the record. This
    should be the same as the project currency.
  <cost_centerid/> The ID of the associated cost center
  <calculated_starts/> Calculated start date of the project task.
  <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.
<project_name/> The name of the associated project.
<id_number/> User-defined task ID.
<closed/> A 1/0 field indicating if this is closed task.
Additional time can not be booked against closed task.
<task_budget_revenue/> If task budgeting is enabled this is the
total projected billing for the task.
<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.
<use_project_assignment/> Flag set to 1 if they are using the
project level user assignment.
<projectid/> The ID of the associated project.
<assign_user_names/> A comma separated list of user nicknames to
assign this task to. (project_task_assign can also be used.)
<starts/> Optional scheduled starting date of this task.
Overrides computed date Start_date.
<fnlt_date/> The finish no later than date of the task. The task
must be finished by this date.
<customerid/> The ID of the associated customer.
<predecessors_type/> Comma delimited list of task
ID:relationship type for predecessors. Only populated if the
relationship type is not finish-to-start.
<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.
<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_5 to the category_5 for many transactions that
have a category_5_id and project_task_id by searching the
project_task and phase work breakdown structure for the first
default_category_5 defined.
<manual_task_budget> If set to 1 then the task budget is manually
entered rather than calculated by the system.
</Projecttask>

```

This datatype supports the read, add, modify, and delete [XML Commands](#). To modify a project without recalculating it, use <Project no\_recalc='1'>

**Note:** The <Projecttask/> datatype accepts an index attribute, which can be set to any of the fields in the project\_task table. For example, when used with id\_number, it allows the list of predecessors to be a list of these user-defined task numbers. The list is then translated to the real IDs, and the result is stored in the database. Refer to the following example:

```
<Add type="Projecttask">
  <Projecttask index="id_number">
    <predecessors>123,abc</predecessors>
  </Projecttask>
</Add>
```

The records that have their id\_number fields set to "123" and "abc" respectively are found, and their actual "id" values are stored in the database as the "predecessors" of the current record.

### Using the limit attribute with Projecttask

When reading project tasks, the limit attribute is applied to **projects** and not project tasks if all of 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" as in the example below, it 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.

```
<Read type="Projecttask" method="all" limit="0,1000"/>
```

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. The example below would apply the limit attribute to project tasks, and not projects:

```
<Read type="Projecttask" method="equal to" limit="0,1000"/>
```

## ProjecttaskEstimate

This datatype 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 a ProjecttaskEstimate object:

- A user must be assigned to the task
- That user's user\_id must exist in the system
- The timesheet\_id must exist in the system
- 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.

```
<ProjecttaskEstimate>
  <id/> Unique ID. Automatically assigned by the system.
  <project_task_id/> The ID of the associated project_task.
  <user_id/> The ID of the user who is assigned to the task.
  <timesheet_id/> The ID of the associated timesheet if this
    was updated from the timesheet.
  <hours/> The number of hours estimated to be remaining.
  <date_changed/> The date and time the estimate was last
    changed.
  <changed_by/> ID of the user who changed the estimate. If
    this does not have an ID, then the estimate was
    automatically generated by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Projecttask\_type

Use the Projecttask\_type datatype to specify information about a project task type.

```
<Projecttask_type>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <notes/> Notes associated with the project task type.
  <active/> A 1/0 field specifying if the type is active.
  <name/> The name of the project task type.
  <updated/> Time the record was last updated or modified.
  <suppress_notification/> Suppress task notifications for this
    project task type.
  <picklist_label/> Label as shown on form picklist.
</Projecttask_type>
```

This datatype supports the read, add, and modify [XML Commands](#).

## Projecttaskassign

Use the Projecttaskassign datatype to specify the list of users assigned to each task.

```
<Projecttaskassign>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <userid/> The ID of the user assigned to this task.
  <planned_hours/> The hours for this user if the planned hours at
    the user level feature is enabled.
  <updated/> Time the record was last updated or modified.
```

```

<projecttaskid/> ID of the project task to which this user is
assigned.
<externalid/> If the record was imported from an external system
you store the unique external record ID here.
<project_groupid/> The ID of the project group if the user was
assigned as part of a project group.
<allocation/> The percentage of time the associated user is
allocated to this task.
<job_codeid/> The ID of the associated job code.
<project_assignment_profile_id/> The id of the associated
project assignment profile.
<pending_booking_id/> The id of the associated pending booking.
<booking_id/> The id of the associated booking.
</Projecttaskassign>

```

This datatype supports the read, add, modify, and delete [XML Commands](#). To modify a project without recalculating it, use `<Project no_recalc='1'>`

**Note:** The `<Projecttaskassign>` datatype is used in conjunction with `<Projecttask>` to assign users to project tasks. Refer to the following example:

```

<Add>
  <Projecttaskassign>
    <userid>3</userid>
    <projecttaskid>13</projecttaskid>
    <allocation>75</allocation>
  </Projecttaskassign>
</Add>

```

**Note:** The `<Projecttaskassign/>` datatype accepts an index attribute, which works the same way as in `Projecttask`. We can use an alternate index to associate `Projecttask` by a field other than the internal id. The index attribute can be any field in `Projecttask` that is unique in combination with `project_id`. However, only the `id_number` is guaranteed to be unique, so the index attribute should not be used with other fields. Refer to the following example:

```

<Add>
  <Projecttaskassign index="id_number">
    <userid>3</userid>
    <allocation>75</allocation>
    <projectid>15</projectid>
    <projecttaskid>AQ99</projecttaskid>
  </Projecttaskassign>
</Add>

```

**Note:** Assuming that in these examples there is a `Projecttask` with an ID of 13, with an `id_number` of AQ99 and a `projectid` of 15, these two examples are identical.

You can add as many user/allocations to a `Projecttask` as you like.

## Proposal

Use the `Proposal` datatype to specify proposal information.

```

<Proposal>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <number/> The proposal number.
  <userid/> The ID of the associated user.
  <status/> The status of the proposal: D - Draft, M - Submitted, P
- Approved, Q - Rejected, S - Sent, V - Viewed, A - Accepted, or
R - Refused.
  <attachments/> If non-zero, the attachment record associated
with this proposal. attachment_id
  <responded/> The date and time the client accepted or refused.
  <sent/> The date and time the proposal was delivered to the
client.
  <access_log/> The mailing and access history of the proposal.
  <response/> Client response notes.
  <name/> The name of this proposal.
  <submitted/> The date and time the proposal was submitted for
approval.
  <approved_by/> The ID of the user who approved this proposal.
  <projectid/> The ID of the associated project.
  <description/> The description of this proposal.
  <total/> The total amount. Dated by the currency_date field.
  <approved/> The date and time the proposal was approved.
  <viewed/> The date and time the client first viewed the proposal.
  <notes/> Notes associated with this proposal.
  <customerid/> The ID of the associated customer.
  <created_by/> The ID of the user who created this proposal.
  <expires/> The date the proposal is valid until.
</Proposal>

```

This datatype supports the read [XML Commands](#).

## Proposalblock

Use the Proposalblock datatype 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.

```

<Proposalblock>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <hour/> The number of hours for a T block.
  <templateid/> The ID of the associated template.
  <content/> The content of the template.
  <um/> The unit of measure for an E block or the rate description
for an O block.
  <rate/> The hourly rate for a T block. Dated by the currency_date
field.
  <proposalid/> The ID of the associated proposal.
  <slipid/> The ID of the associated slip if this block was billed
to TB.

```

```

<seq/> The sequence number of the block.
<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.
<itemid/> The ID of the associated item.
<name/> The name of the this proposal block.
<quantity/> The quantity for an E block or an O block.
<total/> The total value of the block. Dated by the currency_date
field.
<description/> The description of this proposal.
<categoryid/> The ID of the associated category.
<minute/> The number of minutes for a T block.
<type/> The type of the block: X - text only block, T - hourly
rate block, E - expense block, F - flat price block, O - other
type block, or P - product block.
</Proposalblock>

```

This datatype supports the read [XML Commands](#).

## Proxy

The Proxy datatype holds data about user proxies.

```

<Proxy>
  <id/> Unique ID. Automatically assigned by the system.
  <user_id/> ID of the user who is doing the proxying
  <proxy_id/> The user ID for whom you are proxying
  <own/> A "1/0" field indicating if the proxy was created by proxy_id using 'create own
proxy' feature.
  <role_id/> Role to use while proxying for this user
  <expiration/> The date the proxy expires
  <created/> Time the record was created
  <updated/> Time the record was last updated or modified
</Proxy>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Purchase\_item

Use the Purchase\_item datatype to specify purchase item information, a single entry in a purchase order.

```

<Purchase_item>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <date/> The date of the purchase_item. The same as the
purchaseorder date.
  <um/> The unit of measure for the product, i.e., EA.
  <purchaserid/> The ID of the purchaser or purchasing agent. This
is always the same as the purchaseorder creator (purchaser_id).
  <attachmentid/> If non-zero, the attachment record associated
with this purchase_item.

```

```

<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.
<manufacturer_part/> The manufacturer's part number, SKU or
other unique identification for this product.
<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.
<tax_location_name/> The name of the tax location.
<non_po/> A 1/0 field indicating that this purchase item was
created without a purchase order.
<name/> The purchase name.
<total/> The total value of the purchase (in the currency of the
purchase order). Dated by the date field.
<quantity_payable/> The quantity that is payable.
<cusomerid/> The ID of the associated customer.
<request_itemid/> The ID of the associated request_item
<vendor_quote_number/> The vendor's quote number.
<userid/> The ID of the requester.
<purchaserequestid/> The ID of the associated purchaserequest.
<manufacturerid/> The ID of the associated manufacturer.
<currency/> Currency for the money fields in the record.
<quantity_fulfilled/> The quantity that has been fulfilled.
<date_fulfilled/> The date on which all of the quantity was
fulfilled.
<purchaseorderid/> The ID of the associated purchaseorder.
<allow_vendor_substitution/> A 1/0 field indicating whether the
vendor may be substituted.
<order_reference_number/> Unique reference number within
purchase order.
<total_with_tax/> The total value of the purchase (in the
currency of the purchase order)including tax. Dated by the date
field.
<quantity/> The quantity of product_id for this purchases.
<projectid/> The ID of the associated project.
<project_taskid/> The id of the associated project task.
<vendor_sku/> The vendor's sku for this product.
<vendorid/> The ID of the associated vendor
<notes/> Notes associated with this purchase order block.
<productid/> The ID of the associated product.
<acct_date/> The accounting period date of the purchase item.
</Purchase_item>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).



**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 to OpenAir, go to Administration > Application Settings > Purchases Settings and select Other settings. Scroll down and select the check box to Enable the ability to create non-PO purchase items. These are purchase items for purchases made without an OpenAir PO.

- For the capability to modify non short-order purchase items, submit a support ticket and request that the following switch be enabled: API can modify purchase items' project association even when associated with a PO. Associated request items will also be updated. See [Troubleshooting](#) for instructions on how to create a support ticket.

## Purchaseorder

Use the Purchaseorder datatype to specify information about a purchase order.

```
<Purchaseorder>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <receivingid/> The receiving location for this purchase order.
  <carrierid/> The carrier to be used for shipping. Ship Via.
  <date/> The date of the purchase order.
  <date_required/> The date the purchase items on this purchase
order are required.
  <attachmentid/> If non-zero, the attachment record associated
with this purchase order.
  <date_shipped/> The date the materials were shipped if known.
  <date_expected/> The date the materials are expected if known.
  <date_submitted/> The date the purchase order was submitted.
  <name/> The name of the purchase order (Prefix + number).
  <total/> The purchase order total cost. Dated by the date field.
  <description/> The description or purpose for the purchase
order.
  <locationid/> The F.O.B. location_id (DEPRECATED).
  <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.
  <accounts_payableid/> The accounts payable location for this
purchase order.
  <number/> The purchase order number that increments by 1.
  <userid/> The id of the user creating the purchase order. The
purchasing agent.
  <terms/> Payment terms for this purchase order.
  <total_purchase_items/> The total number of purchase items in
the purchase order.
  <currency/> The currency this purchase order is in.
  <quantity_fulfilled/> The quantity fulfilled on all the purchase
items in this purchase order.
  <date_approved/> The date the purchase order was approved.
  <date_order_placed/> The date the purchase order was placed with
the vendor.
  <date_fulfilled/> The date on which all of the total quantity was
fulfilled.
  <auto_track_payable_with_fulfilled/> A 1/0 field indicating that
payability of quantities of items on this purchase order track
automatically and directly with the fulfillment of those items.
  <total_quantity/> The total quantity of all the purchase items
in this purchase order.
  <purchase_items_fulfilled/> The total number of fulfilled
```

```

    purchase_items in the purchase order.
    <approval_status/> The approval status of the purchase request:
    0 - open, P - pending approval, A - approved, or R - rejected.
    <vendorid/> The id of the associated vendor that the purchase
    order is for.
    <notes/> Notes to print on the purchase order.
    <ship_complete_only/> A 1/0 field indicating that full order
    must ship together.
    <prefix/> A static alphanumeric purchase order number prefix.
  </Purchaseorder>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Purchaser

Use the <Purchaser> datatype to specify information about a user who creates purchase orders.

```

<Purchaser>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <receivingid/> The default receiving location for this
  purchaser.
  <userid/> The ID of the associated user.
  <name/> The name of the purchaser.
  <updated/> Time the record was last updated or modified.
  <carrierid/> The default carrier to be used for shipping. Ship
  Via.
  <notes/> Notes associated with the purchaser.
  <exported/> Date and time the record was marked as exported.
  <accounts_payableid/> The default accounts payable location for
  this purchaser.
  <ship_complete_only/> The default for the 1/0 field indicating
  that full order must ship together.
  <active/> A 1/0 field indicating where this is designated as an
  active receiving location 1/0.
</Purchaser>

```

This datatype supports the read [XML Commands](#).

## Purchaserequest

Use the Purchaserequest datatype to specify purchase request information.

```

<Purchaserequest>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <number/> The purchase request number that increments by 1.
  <date/> The date of the purchase request.
  <userid/> The ID of the associated user creating the purchase
  request. The requester.
  <request_items_fulfilled/> The total number of fulfilled request
  items in the purchase request.

```

```

<total_request_items/> The total number of request items in the
purchase request.
<ordered_request_items/> The total number of request items on
the purchase request which are part of a purchase order.
<date_required/> The date the material on this purchase request
is required.
<attachmentid/> If non-zero, the attachment record associated
with this purchase request.
<currency/> The currency of the total field.
<quantity_fulfilled/> The quantity fulfilled on all the request
items in this purchase request.
<date_approved/> The date the purchaserequest was approved.
<date_fulfilled/> The date on which all of the total quantity was
fulfilled.
<total_quantity/> The total quantity of all the request items in
this purchase request
<date_submitted/> The date the purchaserequest was submitted.
<approval_status/> The approval status of the purchase request:
0 - open, P - pending approval, A - approved, or R - rejected.
<name/> The name of the purchaserequest (Prefix + number).
<projectid/> The ID of the associated project that the material
on this purchase request is for.
<description/> The description or purpose for the
purchaserequest.
<total/> The purchase request total cost. Dated by date field.
<notes/> Notes associated with the purchase request.
<customerid/> The ID of the associated customer that the
material on this purchase request is for.
<exported/> Date and time the record was marked as exported.
<prefix/> A static alphanumeric purchase request number prefix.
</Purchaserequest>

```

This datatype supports the read [XML Commands](#).

## Ratecard

Use the Ratecard datatype to map job codes to hourly rates.

```

<Ratecard>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <notes/> Notes associated with the rate card.
  <active/> A 1/0 field indicating whether this is an active rate
card.
  <name/> The name of the rate card.
  <updated/> Time the record was last updated or modified.
</Ratecard>

```

This datatype supports the read, add, and modify [XML Commands](#).

## RateCardItem

Use the RateCardItem datatype to specify rate card item information.



```

<RateCardItem>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <rate_card_id/> The ID of the rate card that it is associated
  with.
  <job_code_id/> The ID of the associated job code.
  <currency/> Currency for the money fields in the record.
  <updated/> Time the record was last updated or modified.
  <rate/> The hourly billing rate.
  <start/> Start date of the rate for historical records.
  <end/> End date of the rate for historical records.
  <current/> A 1/0 field indicating if the record is the current
  rate.
</RateCardItem>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Reimbursement

Use the Reimbursement datatype to specify reimbursement information.

```

<Reimbursement>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <envelopeid/> The associated envelope the reimbursement is
  applied to.
  <date/> The date of the reimbursement.
  <total/> The reimbursement total. Dated by the date field.
  <updated/> Time the record was last updated or modified.
  <currency/> Currency for the money fields in the record.
  <envelope_number/> The number of 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.
  <notes/> Notes associated with the reimbursement.
  <userid/> The user associated with the envelope the
  reimbursement is applied to.
  <audit/> Audit trail changes.
</Reimbursement>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Repeat

Use the Repeat datatype to specify repeating event information.

```

<Repeat>
  <id/> Unique ID. Automatically assigned by the system.
  <frequency/> The repeating interval of the event: D - daily, W -
  weekly, M - monthly, Y - yearly.
  <every/> The spacing between each repeating event.
  <end/> End date of the event.
  <occur_number/> Number of occurrences.

```

```

<how_end/> How does this event end: D - date or O - occurrence
<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.
<created/> Time the record was created.
<updated/> Time the record was last updated or modified.
</Repeat>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Report

Use the Report datatype to hold saved report definitions and settings.

```

<Report>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <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.
  <name/> The name of the report.
  <type/> The type of report: S = saved reports and T = sTandard
  reports.
  <thin_client_context/> A 1/0 field indicating that this report
  can be requested via thin clients.
  <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.
  <relatedid/> Related ID for attribute type. Report = ID of saved
  report, Timesheet = ID of timesheet, and Envelope = ID of related
  expense for expense report.
</Report>

```

This datatype supports the read [XML Commands](#).

## Request\_item

Use the Request\_item datatype to specify a request item, a single entry in a purchase request.

```

<Request_item>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <vendor_quote_number/> The vendor's quote number.
  <userid/> The ID of the requester. This is always the same as the
  purchaserequest requester (user_id).
  <date/> The date of the request_item. The same as the
  purchaserequest date.
  <manufacturerid/> The ID of the associated manufacturer.
  <purchaserequestid/> The ID of the associated purchaserequest.

```

```

<um/> The unit of measure for the product, i.e., EA
<request_reference_number/> Unique reference number within
purchase request.
<attachmentid/> If non-zero, the attachment record associated
with this request_item.
<currency/> The currency used for this request item.
<quantity_fulfilled/> The quantity that has been fulfilled.
<productid/> The ID of the associated product.
<date_fulfilled/> The date on which all of the quantity was
fulfilled.
<purchase_itemid/> The ID of the associated purchase_item.
<allow_vendor_substitution/> A 1/0 field indicating whether the
vendor may be substituted.
<manufacturer_part/> The manufacturer's part number, SKU or
other unique identification for this product.
<purchaseorderid/> The ID of the associated purchase order.
<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.
<name/> The request item name.
<quantity/> The quantity of product_id for this request item.
<projectid/> The ID of the associated project. This is always the
same as the purchase request project_id.
<total/> The total value of the request item. Dated by the date
field.
<vendorid/> The ID of the associated vendor.
<vendor_sku/> The vendor's sku for this product.
<notes/> Notes associated with this request item.
<customerid/> The ID of the associated customer. This is always
the same as the purchase request customer_id.
<exported/> Date and time the record was marked as exported.
</Request_item>

```

This datatype supports the read and delete [XML Commands](#).

## ResourceAttachment

Use the ResourceAttachment datatype to specify a user's CV attachment in their Consolidated Resource Profile.

**Note:** Uploading a CV using the ResourceAttachment datatype is a two-step process. For an example of this process, please see [Example 5](#) for the [Add](#) command.

```

<ResourceAttachment>
  <id/> Unique ID. Automatically assigned by the system.
  <userid/> The ID of the user to whom this attachment belongs.
  <attachment_id/> The attachment record associated with this document.
  <type/> The document type. Only "CV" is supported.
  <latest_attachment_id/> ID of the latest attachment from the attachment table.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
</ResourceAttachment>

```

This datatype supports the add, read, modify, and delete [XML Commands](#).

## Resourceprofile

Use the Resourceprofile datatype to specify items the make up a resource profile.

```
<Resourceprofile>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <userid/> The ID of the user for which this resourceprofile
  describes.
  <resourceprofile_typeid/> The ID of the resourceprofile_type.
  <name/> The resourceprofile name. Stub.
  <updated/> Time the record was last updated or modified.
  <attributeid/> The ID of the optional resourceprofile attribute.
  <comment/> Additional comment describing this resourceprofile.
  <externalid/> If the record was imported from an external system
  you store the unique external record ID here.
  <type/> The resourceprofile type. The entity on which this
  resourceprofile is based: Skill, Education, Location, Jobrole,
  Industry, or Customprofile_1..20.
</Resourceprofile>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Resourceprofile\_type

Use the Resourceprofile\_type datatype to specify information about a resource profile type.

```
<Resourceprofile_type>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <active/> A 1/0 field indicating whether this is active.
  <related_table/> The name of the table related with this table.
  <name/> The resourceprofile_type name. This shows up on all the
  resourceprofile_type pop-up windows in the application.
  <updated/> Time the record was last updated or modified.
  <relatedid/> The ID of the related item in the related table.
  <externalid/> If the record was imported from an external system
  you store the unique external record ID here.
  <type/> The resourceprofile type. The entity on which this
  resourceprofile is based: Skill, Education, Location, Jobrole,
  Industry, or Customprofile_1..20.
  <attribute_set_id/>The ID of the associated attribute set.
</Resourceprofile_type>
```

This datatype supports the read, add, modify and delete [XML Commands](#).

## ResourceRequest

Use the ResourceRequest datatype to specify information about a resource request type.

```
<ResourceRequest>
  <id/> Unique ID. Automatically assigned by the system.
```

```

<number/> The resource request tracking number.
<status/> The status of the resource request:
'O' - Open
'P' - Partial
'S' - Complete
'C' - Canceled
<percent_fulfilled/> Percent fulfilled for the resource request.
<date_finalized/> The date the resource request was finalized
and marked ready for booking.
<date_start/> The starting date of the resource request.
<ownerid/> The id of the associated user creating the resource
request.
<date_end/> The ending date of the resource request.
<booking_type_id/> The booking type of bookings created for this
resource request.
<name/> The name of the resource request.
<projectid/> The id of the associated project.
<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.
<notes/> Notes field
<customerid/> The id of the associated customer.
</ResourceRequest>

```

This datatype supports the read, add, modify and delete [XML Commands](#).

## ResourceRequestQueue

Use the ResourceRequestQueue datatype to specify information about a resource request queue type.

```

<ResourceRequestQueue>
  <date_end/> The ending date of the resource request queue.
  <status/> The status of the resource request queue:
  'O' - Open
  'P' - Partial
  'S' - Complete
  'C' - Canceled
  <name/>The name of 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 oaResourcessearch and populate the
  id here.
  <external_id/> If the record was imported from an external
  system you store the unique external record id here.
  <date_start/> The starting date of the resource request queue.
  <percent_fulfilled/> Percent fulfilled for the resource request
  queue.
  <notes/> Notes field.
  <customerid/> The id of the associated customer.

```

```
<slots/> The number of slots available in this queue.
</ResourceRequestQueue>
```

This datatype supports the read, add, modify and delete [XML Commands](#).

## ResourceSearch

Use the ResourceSearch datatype to specify information about a resource search type. See also [Resource Search Virtual Fields](#)

```
<ResourceSearch>
  <id/> Unique ID. Automatically assigned by the system.
  <include_inactive_resources/> A "1/0" field. Include inactive
  resources in search?
  <startdate/> The start date for availability
  <include_regular_resources/> A "1/0" field. Include regular
  resources in search?
  <required/> See Resource Search Virtual Fields.
  <enddate/> The end date for availability.
  <updated/> Time the record was last updated or modified.
  <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.
  <hours/> The number of hours of availability required over this
  range.
  <excluding/> See Resource Search Virtual Fields.
  <consecutive_availability/> A "1/0" field indicating no
  intervening bookings.
  <availability_search/> A "1/0" field indicating whether to
  search by availability.
  <name/> The resourceSearch name.
  <preferred/> See Resource Search Virtual Fields.
  <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.
  <resource_request_queue_id/> The id of the associated resource
  request queue
  <include_generic_resources/> A "1/0" field. Include generic
  resources in search?
  <external_id/> If the record was imported from an external
  system you store the unique external record id here.
</ResourceSearch>
```

This datatype supports the read, add, modify and delete [XML Commands](#).

## Resource Search Virtual Fields

ResourceSearch 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.

## RevenueContainer

Use the RevenueContainer datatype to specify information about the revenue\_container header table.

```
<RevenueContainer>
  <id/> Unique ID. Automatically assigned by the system.
  <number/> The revenue_container number that increments by 1.
  <date/> The date of the revenue_container.
  <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.
  <total_recognized/> The revenue_container recognized total.
  Dated by the date field.
  <currency/> The currency of this revenue_container.
  <date_approved/> The date the invoice was approved.
  <updated/> Time the record was last updated or modified.
  <date_submitted/> The date the invoice was submitted.
  <approval_status/> The approval status of the invoice. Used only
  if invoice approvals are used: O - Open, S - Submitted,
  A - Approved, or R - Rejected
  <total_deferred/> The revenue_container deferred total. Dated by
  the date field.
  <name/> The name of the revenue_container (Prefix + number).
  <acct_date/> The accounting period date of the
  revenue_container.
  <total_accrued/> The revenue_container accrued total. Dated by
  the date field.
  <projectid/> The ID of the associated project.
  <externalid/> If the record was imported from an external
  system, you store the unique external record ID here.
  <total_posted/> The revenue_container posted total. Dated by the
  date field.
  <created/> Time the record was created.
  <notes/> Notes to print on the revenue_container.
  <total_invoiced/> The revenue_container invoice total. Dated by
  the date field.
  <customerid/> The ID of the associated customer.
  <exported/> Date and time the record was marked as exported.
  <prefix/> A static alphanumeric revenue_container number prefix.
```

```
</RevenueContainer>
```

This datatype supports the read [XML Commands](#).

## RevenueProjection

Use the RevenueProjection datatype to access the slips created from a projected revenue run.

```
<RevenueProjection>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <hour/> The number of hours for a T slip.
  <date/> The date of the billing slip.
  <um/> The unit of measure for an E or P slip or the rate
  description for an O slip.
  <rate/> The hourly rate for a T slip. Dated by the date field.
  <slip_stage_id/> The ID of the associated slip stage.
  <project_billing_rule_id/> The ID of the associated project
  billing rule.
  <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.
  <description/> The description of the billing slip.
  <total/> The total value of the slip. Dated by the date field.
  <category_id/> The ID of the associated category. If this is set,
  the slip is based on this category.
  <timer_start/> The starting time of the timer.
  <minute/> The number of minutes for a T slip.
  <customer_id/> The ID of the associated customer.
  <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.
  <agreement_id/> The ID of the associated agreement.
  <total_tax_paid/> The total tax paid. Dated by the date field.
  <customerpo_id/> The ID of the associated customerpo.
  <user_id/> The ID of the associated user.
  <invoice_id/> The ID of the associated invoice once billed.
  <currency/> Currency for the money fields in the record
  <city/> The slip city or location.
  <payment_type_id/> The ID of the associated payment type.
  <total_with_tax/> A 1/0 field indicating whether the cost
  includes the tax.
  <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.
  <timetype_id/> The ID of the associated time type.
  <quantity/> The quantity for an E, O, or P slip.
  <project_id/> The ID of the associated project.
  <project_task_id/> The ID of the task within the associated
  project.
  <product_id/> The ID of the associated product.
  <notes/> Notes associated with the slip.
  <cost_center_id/> The ID of the associated cost center.
```



<acct\_date/> The accounting period date of the slip.  
 <projecttask\_type\_id/> The ID of the projecttask\_type of the associated projecttask.  
 <job\_code\_id/> The ID of the associated job code.  
 <payroll\_type\_id/> The ID of the associated payroll type.  
 <ref\_slip\_id/> For credit/rebill, ID of the original slip id.  
 <portfolio\_project\_id/> The ID of the associated portfolio project.  
 <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.  
 <revenue\_recognition\_rule\_id/>Id of the revenue recognition rule that created this projection.  
 <revenue\_projection\_type/>The type of the projection:  
 R - Revenue from an "As billed" recognition rule  
 F - Revenue from an "Fixed fee" recognition rule  
 G - Revenue from an "Percent complete" recognition rule  
 H - Revenue from an "Incurred vs. forecast" recognition rule  
 J - Revenue from a "Time project billing rule" rule  
 U - Time billed but not recognized  
 T - Time not billed  
 <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 be a small number Dated by the date field  
 <slip\_projection\_id/>Id of the slip\_projection that was used for an as billed rule  
 <project\_billing\_rule\_id/>Id of the project billing rule that created this projection.  
 <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  
 <booking\_type\_id/>Id of the booking type if this was generated from bookings.  
 <revenue\_stage\_id/>Id of the revenue\_stage. This will always be 'no revenue stage' 0 for revenue projections.  
 <transaction\_id/>For internal user only.  
 <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.  
 <name/>The name of the slip. This field is never populated. It is used only to satisfy subtotalling by slip in summary reports.  
 <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.
<originating_id/>For use with split slips feature. If set, the
slip.id of the originating slip for this split portion.
<repeat_id/>The id of the associated repeating event.
<vehicle_id/>The id of the associated vehicle.
<cost_includes_tax/>A 1/0 field indicating whether the cost
includes the tax.
<exported/>Date and time the record was marked as exported.
</RevenueProjection>

```

This datatype supports the read [XML Commands](#).

**Note:** This datatype cannot be read while projections are running. This is because the table data may be incomplete until the job completes. Error 606 will be returned if a read is attempted while projects are running.

## Revenue\_recognition\_rule

Use the Revenue\_recognition\_rule datatype to specify revenue recognition rules.

```

<Revenue_recognition_rule>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <user_filter/> CSV list of users to limit the rule to.
  <purchase_how/> How purchases should be recognized: M - mark up/
down on billed purchases or B - billed purchases.
  <percent_complete/> The calculated percent complete value if a
type P transaction.
  <percent/> The percentage value for a fixed fee percent trigger.
  <end_milestone/> ID of the ending milestone (project_task).
  <recognition_type/> What we are recognizing: R - revenue,
C - cost, or O - other.
  <marked_as_ready/> Trigger recognition when a task (id in phase)
is marked as ready to recognize.
  <break_by_user/> Break out the transactions by user. Currently
only implemented for the incurred rules.
  <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.
  <timetype_filter/> CSV list of timetypes to limit the rule to.
  <expense_how/> How expenses should be recognized: M - mark up/
down on billed expenses, B - billed expenses, or I - incurred
expenses.
  <active/> A 1/0 field indicating whether this is an active rule.
  <name/> Name of the rule.
  <categoryid/> The ID of the associated category.
  <start_milestone/> ID of the starting milestone (project_task).
  <end_date/> End date of the rule.
  <customerid/> The ID of the associated customer.
  <agreementid/> ID of the associated agreement.

```

```

<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
<asb_exclude_slip_type/> CSV list of the slip types to exclude
from the as billed rule.
<customerpo_id/> ID of the associated customerpo.
<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 or B - % complete of planned hours
for a phase or task (the task id is held in the phase field).
<item_filter/> CSV list of items to limit the rule to.
<project_task_filter/> CSV list of tasks to limit the rule to.
<product_filter/> CSV list of products to limit the rule to.
<slip_stage_filter/> CSV list of slip_stage ID to limit a type A
rule to.
<repeatid/> The ID of the associated repeating event.
<currency/> Currency for the money fields in the record.
<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.
<acct_code/> Optional accounting system code for integration
with external accounting systems.
<start_date/> Start date of the rule.
<projectid/> The ID of the associated project.
<amount/> The amount. If we have multiple amounts, the values are
held in the revenue_recognition_rule_amount table.
<extra_data/> Holds extra data fields associated with the rule.
<notes/> Notes associated with this revenue recognition rule.
<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
<accounting_period_id/> The ID of the associated accounting
period.
<cost_centerid/> The ID of the associated cost center.
<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.
<project_billing_rule_filter/> CSV list of project billing rule
id's to limit a type T rule to.
<category_lid/> 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.
<assigned_user/> The user to assign to fixed fee recognition.
</Revenue_recognition_rule>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Revenue\_recognition\_rule\_amount

Use the Revenue\_recognition\_rule\_amount datatype to specify multiple amounts for a recognition rule.

```

<Revenue_recognition_rule_amount>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <revenue_recognition_rule_id/> The ID of the associated rule.
  <customerpo_id/> The ID of the associated customerpo.
  <recognition_type/> Recognition type: R - revenue, C - cost, or
  0 - other.
  <updated/> Time the record was last updated or modified.
  <currency/> Currency for the money fields in the record.
  <category_id/> The ID of the associated category.
  <amount/> The amount.
  <acct_code/> Optional accounting system code for integration
  with external accounting systems.
  <agreement_id/> The ID of the associated agreement.
  <cost_center_id/> The ID of the associated category.
  <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.
</Revenue_recognition_rule_amount>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Revenue\_recognition\_transaction

Use the Revenue\_recognition\_transaction datatype to specify revenue recognition transactions. This is a record of a single transaction created when revenue recognition was run for a particular project.

```

<Revenue_recognition_transaction>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.

```

<percent\_complete/> The calculated percent complete value if it is a type P transaction.  
 <revenue\_recognition\_ruleid/> The ID of the associated rule.  
 <userid/> The ID of the associated user.  
 <date/> The date of the transaction.  
 <taskid/> The ID of the associated task.  
 <customerpo\_id/> The ID of the associated customerpo.  
 <recognition\_type/> Recognition type: R - revenue, C - cost, or 0 - other.  
 <updated/> Time the record was last updated or modified.  
 <slipid/> The ID of the associated slip.  
 <currency/> Currency for the money fields in the record.  
 <customerid/> The ID of the associated customer.  
 <ticketid/> The ID of the associated ticket.  
 <project\_taskid/> The ID of the associated project task.  
 <projectid/> The ID of the associated project.  
 <total/> The amount of the transaction. Dated by the date field.  
 <categoryid/> The ID of the associated category.  
 <notes/> Notes associated with this revenue recognition transaction.  
 <acct\_code/> Optional accounting system code for integration with external accounting systems.  
 <type/> The type of the transaction. Matches the type field in revenue\_recognition\_rule.  
 <agreementid/> The ID of the associated agreement.  
 <acct\_date/> The accounting period date of the transaction.  
 <cost\_center\_id/> The ID of the associated cost center.  
 <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.  
 <project\_billing\_ruleid/> The ID of the associated billing rule.  
 <job\_codeid/> The ID of the associated job code.  
 <rate/> The hourly rate for a T type. Dated by the date field.  
 <decimal\_hours/> The number of decimal hours.  
 <hour/> The number of hours for a T type.  
 <minute/> The number of minutes for a T type.  
 <revenue\_containerid/> The ID of the associated revenue\_container once posted.  
 <revenue\_stageid/> The ID of the associated revenue stage.  
 <originatingid/> The ID of the originating revenue\_recognition\_transaction for this revenue\_recognition\_transaction.  
 <offsetsid/> The ID of the revenue\_recognition\_transaction which this revenue\_recognition\_transaction offsets.  
 <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.  
 <portfolio\_projectid/> The ID of the associated portfolio project.  
 <agreement\_externalid/> Import-only field.

```

<category_externalid/> Import-only field.
<customer_externalid/> Import-only field.
<project_externalid/> Import-only field.
<projecttask_externalid/> Import-only field.
<user_externalid/> Import-only field.
</Revenue_recognition_transaction>

```

This datatype supports the read, add, and modify [XML Commands](#).

## RevenueStage

Use the RevenueStage datatype to specify revenue recognition transaction stage information. Index the attributes and use them to filter revenue recognition transactions.

```

<RevenueStage>
  <id/> Unique ID. Automatically assigned by the system.
  <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
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
</RevenueStage>

```

This datatype supports the read [XML Commands](#).

## Role

Use the Role datatype to specify role information.

```

<Role>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <name/> The name of this role.
  <notes/> Notes associated with this role.
  <default_role/> A 1/0 field indicating whether this is the
  default new user role.
  <admin_role/> A 1/0 field indicating whether this is the chief
  administrator role, with full rights.
  <external_id/> If the record was imported from an external
  system, you store the unique external record ID here.
  <updated/> Time the record was last updated or modified.
</Role>

```

This datatype supports the read [XML Commands](#).

## Schedulebyday

Use the Schedulebyday datatype to retrieve the day-by-day representation of users' work schedules.

```

<Schedulebyday>
  <id/> Unique ID. Automatically assigned by the system.
  <date/> The date.
  <hours/> The number of schedule hours on this date for this user,
  including exceptions.
  <user_id/> The id of the associated user.
  <base_hours/> The number of base hours on this date for this
  user.
  <target_hours/> The number of target hours for this user on this
  date. Target_utilization.percentage * hours.
  <target_base_hours/> The number of target base hours for this
  user on this date Target_utilization.percentage * base_hours.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
</Schedulebyday>

```

This datatype supports the read [XML Commands](#).

## Scheduleexception

Use the Scheduleexception datatype to describe changes to the default work schedule for a company or user.

```

<Scheduleexception>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <workhours/> The number of hours per day during this daterange.
  This overrides any workhours on each day of either the account
  schedule or the account/user schedule.
  <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.
  <name/> The exception name and description, e.g. New Years Day.
  <exception_type/> The type of exception. R - Date range of the
  exception.
  <startdate/> The start date for the exception.
  <enddate/> The end date for the exception.
  <updated/> Time the record was last updated or modified.
  <workscheduleid/> The ID of the corresponding work schedule.
  <timetypeid/> The ID of the associated time type.
  <schedule_request_itemid/> The ID of the schedule change item
  from a schedule request.
</Scheduleexception>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Schedulerequest

Use the Schedulerequest datatype to specify schedule request details.

```

<Schedulerequest>
  <id/> Unique ID. Automatically assigned by the system.

```

```

<created/> Time the record was created.
<number/> The schedule request number that increments by 1.
<userid/> The ID of the user creating the schedule request.
<startdate/> The start date of the schedule request.
<date/> The date of the schedule request creation.
<attachmentid/> If non-zero, the attachment record associated
with this schedule request.
<enddate/> The end date of the schedule request.
<approval_status/> The approval status of the schedule request:
0 - open, P - pending approval, A - approved, or R - rejected.
<updated/> Time the record was last updated or modified.
<date_approved/> The date the schedule request was approved.
<date_submitted/> The date the schedule request was submitted.
<customerid/> The ID of the associated customer.
<timetype/> The time type of this schedule request: R - regular
time or P - personal time.
<timetypeid/> The ID of the associated time type.
<project_taskid/> The ID of the associated project task.
<projectid/> The ID of the associated project.
<categoryid/> The ID of the associated category.
<notes/> Notes to print on the schedule request.
<externalid/> If the record was imported from an external system
you store the unique external record ID here.
<description/> Description or purpose for the schedule request.
<prefix/> A static alphanumeric schedule request number prefix.
<name/> The name of the schedule request (Prefix + number).
</Schedulerequest>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Schedulerequest\_item

Use the Schedulerequest\_item datatype to specify information for multiple schedule request items.

```

<Schedulerequest_item>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <hours/> The number of hours for this schedule request item.
  <date/> The date of the schedule request item.
  <userid/> The ID of the associated user.
  <timetypeid/> The ID of the associated time type.
  <name/> The schedule request item name. It is the same as the
schedule request description.
  <request_reference_number/> Unique reference number within
schedule request.
  <schedule_requestid/> The ID of the associated schedule request.
  <updated/> Time the record was last updated or modified.
  <customerid/> The ID of the associated customer.
  <project_taskid/> The ID of the associated project task.
  <projectid/> The ID of the associated project.
  <categoryid/> The ID of the associated category.
  <externalid/> If the record was imported from an external system
you store the unique external record ID here.
</Schedulerequest_item>

```



This datatype supports the read, modify, and delete [XML Commands](#).

## Slip

Use the Slip datatype to specify slip information. A slip is an individual timebill or an individual charge to a customer. Multiple slips are aggregated into an invoice.

```
<Slip>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <hour/> The number of hours for a T slip.
  <date/> The date of the billing slip.
  <unitm/> The unit of measure for an E or P slip or the rate
description for an O slip.
  <rate/> The hourly rate for a T slip. Dated by the date field.
  <slip_stageid/> The ID of the associated slip stage.
  <project_billing_ruleid/> The ID of the associated project
billing rule.
  <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.
  <tax_location_name/> The name of the tax location
  <sold_to_contactid/> The ID of the contact sold to.
  <description/> The description of the billing slip.
  <total/> The total value of the slip. Dated by the date field.
  <categoryid/> The ID of the associated category. If this is set,
the slip is based on this category.
  <timer_start/> The starting time of the timer.
  <minute/> The number of minutes for a T slip.
  <customerid/> The ID of the associated customer.
  <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.
  <agreementid/> The ID of the associated agreement.
  <total_tax/> The total tax paid. Dated by the date field.
  <customerpoid/> The ID of the associated customerpo.
  <userid/> The ID of the associated user.
  <invoiceid/> The ID of the associated invoice once billed.
  <currency/> Currency for the money fields in the record
  <city/> The slip city or location.
  <decimal_hours/> The number of decimal hours for a T slip.
  <payment_typeid/> The ID of the associated payment type.
  <total_with_tax/> A 1/0 field indicating whether the cost
includes the tax.
  <shipping_contactid/> The ID of the associated shipping contact.
  <itemid/> 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.
  <timetypeid/> The ID of the associated time type.
  <quantity/> The quantity for an E, O, or P slip.
  <billing_contactid/> The ID of the associated billing contact.
  <projectid/> The ID of the associated project.
  <projecttaskid/> The ID of the task within the associated
```

```

project.
<productid/> The ID of the associated product.
<notes/> Notes associated with the slip.
<cost_centerid/> The ID of the associated cost center.
<acct_date/> The accounting period date of the slip.
<projecttask_type_id/> The ID of the projecttask_type of the
associated projecttask.
<job_code_id/> The ID of the associated job code.
<payroll_type_id/> The ID of the associated payroll type.
<ref_slipid/> For credit/rebill, ID of the original slip id.
<portfolio_projectid/> The ID of the associated portfolio
project.
<originating_id/>For use with split slips feature. If set, the
slip.id of the originating slip for this split portion.
<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.
<gl_code/> The fixed code 1234455454.
<skip_recognition/> A "1/0" field indicating if this record
should be recognized. Used for split charges which were already
recognized.
</Slip>

```

This datatype supports the read, add, and modify [XML Commands](#).



**Note:** If not all portfolio\_projectids 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 your Professional Services Consultant or create a support ticket. See [Troubleshooting](#) for instructions on creating a support ticket.

## SlipProjection

Use the SlipProjection datatype to hold slips created from a projected billing run. This datatype contains many of the slip datatype fields with addition fields.

```

<SlipProjection>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <hour/> The number of hours for a T slip.
  <date/> The date of the billing slip.
  <unitm/> The unit of measure for an E or P slip or the rate
description for an O slip.
  <rate/> The hourly rate for a T slip. Dated by the date field.
  <slip_stageid/> The ID of the associated slip stage.
  <project_billing_ruleid/> The ID of the associated project
billing rule.

```

```

<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.
<sold_to_contactid/> The ID of the contact sold to.
<description/> The description of the billing slip.
<total/> The total value of the slip. Dated by the date field.
<categoryid/> The ID of the associated category. If this is set,
the slip is based on this category.
<timer_start/> The starting time of the timer.
<minute/> The number of minutes for a T slip.
<customerid/> The ID of the associated customer.
<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.
<agreementid/> The ID of the associated agreement.
<customerpoid/> The ID of the associated customerpo.
<userid/> The ID of the associated user.
<invoiceid/> The ID of the associated invoice once billed.
<currency/> Currency for the money fields in the record
<city/> The slip city or location.
<decimal_hours/> The number of decimal hours for a T slip.
<payment_typeid/> The ID of the associated payment type.
<shipping_contactid/> The ID of the associated shipping contact.
<itemid/> 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.
<timetypeid/> The ID of the associated time type.
<quantity/> The quantity for an E, O, or P slip.
<billing_contactid/> The ID of the associated billing contact.
<projectid/> The ID of the associated project.
<projecttaskid/> The ID of the task within the associated
project.
<productid/> The ID of the associated product.
<notes/> Notes associated with the slip.
<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.
<booking_typeid/> ID of the booking type if this was generated
from bookings.
<transactionid/> For internal use only.
<projecttask_typeid/> The ID of the project task type.
<cost_centerid/> The ID of the associated cost center.
<acct_date/> The accounting period date of the task.
<job_codeid/> The ID of the associated job code.
</SlipProjection>

```

This datatype supports the read [XML Commands](#).

## Slipstage

Use the Slipstage datatype to specify the various stages a slip can be in.

```
<Slipstage>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <exclude_from_invoicing/> Exclude slips of this stage from
  invoicing.
  <notes/> Notes associated with this slip stage.
  <name/> The name of the stage.
  <updated/> Time the record was last updated or modified.
  <position/> The position of the stage.
  <enable_slip_tab/> Display slips of this stage in a separate
  tab.
</Slipstage>
```

This datatype supports the read, add, and modify [XML Commands](#).

## TagGroup

Use the TagGroup datatype to specify user entity tags for users, customers, or projects.

```
<TagGroup>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <active/> A 1/0 field indicating whether the record is active.
  <name/> Name of the tag group.
  <entity_type/> The tag group type: U - user, C - customer, or P -
  project.
  <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.
</TagGroup>
```

This datatype supports the read [XML Commands](#).

## TagGroupAttribute

Use the TagGroupAttribute datatype to specify attributes associated with user entity tags for users, customers, or projects.

```
<TagGroupAttribute>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <active/> A 1/0 field indicating whether the record is active.
  <updated/> Time the record was last updated or modified.
  <name/> Name of the tag group attribute.
  <tag_groupid/> The ID of the tag group this attribute is in.
</TagGroupAttribute>
```

This datatype supports the read [XML Commands](#).

## TargetUtilization

Use the TargetUtilization datatype to specify target utilization values for a user.

```
<TargetUtilization>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <user_id/> The ID of the associated user.
  <start_date/> The start date for the target utilization.
  <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.
  <percentage/> Target utilization for this user as a percentage.
  For example, 75.30.
  <dirty/> A 2/1/0 field: 0 - Clean, 1 - Dirty, and 2 - Inprogress.
</TargetUtilization>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Task

Use the Task datatype to specify task information. A task is a single time entry in a timesheet grid.

```
<Task>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <projecttask_typeid/> The ID of the projecttask_type of the
  associated project_task.
  <userid/> The ID of the associated user.
  <date/> The date of the task.
  <decimal_hours/> The number of decimal hours for the task.
  <cost_centerid/> The ID of the associated cost center.
  <slipid/> The ID of the associated slip if this task was
  billed.
  <hours/> The number of hours for the task.
  <timetypeid/> The ID of the associated time type.
  <minutes/> The number of minutes for the task.
  <projectid/> The ID of the associated project.
  <description/> Description of the task.
  <categoryid/> The ID of the associated category.
  <projecttaskid/> ID of the task within the associated
  project.
  <timesheetid/> The ID of the associated timesheet.
  <notes/> Notes associated with this task.
  <customerid/> The ID of the associated customer.
  <payroll_typeid/> The ID of the associated payroll type.
  <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.
<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.
<acct_date/> The accounting period date of the task.
<category_lid/> 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.
<thin_client_id/> Used by thin clients to reconcile
imported records.
</Task>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).



**Note:** 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, please contact OpenAir Support and request the "API to Respect Time Entry Date for Currency Conversion in Loaded Costs" feature.

## TaskTimecard

Use the TaskTimecard datatype to specify tasks associated with timecards.

```

<TaskTimecard>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <projecttask_typeid/> The ID of the project task type.
  <project_phaseid/> The ID of the project phase.
  <userid/> The ID of the associated user.
  <date/> The date of the task timecard.
  <decimal_hours/> The number of decimal hours for the task
timecard.
  <cost_centerid/> The ID of the associated cost center.
  <slipid/> The ID of the associated slip.
  <hours/> The number of hours for the task timecard.
  <timetypeid/> The ID of the associated time type.
  <minutes/> The number of minutes for the task timecard.
  <projectid/> The ID of the associated project.
  <description/> The description of the task timecard.
  <categoryid/> The ID of the associated category.
  <projecttaskid/> The ID of the task within the assoc. project.
  <timesheetid/> The ID of the associated timesheet.
  <notes/> Notes associated with this task timecard.

```

```

<time_cardid/> The ID of the associated timecard.
<customerid/> The ID of the associated customer.
<payroll_typeid/> The ID of the associated payroll 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.
</TaskTimecard>

```

This datatype supports the read [XML Commands](#).

## TaxLocation

Use the TaxLocation datatype to specify tax location information.

```

<TaxLocation>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <hst_rate/> The HST rate. This is used instead of GST and PST in
  some locations.
  <federal_rate/> The federal tax rate.
  <name/> The name for the estimate adjustment.
  <updated/> Time the record was last updated or modified.
  <acct_code_federal/> GL accounting code for the federal entries.
  <tax_method/> The tax method: G - GST and PST, H - HST, or F -
  Federal and State.
  <state_rate/> The state tax rate.
  <acct_code_pst/> GL accounting code for the PST entries.
  <acct_code_state/> GL accounting code for the state entries.
  <active/> A 1/0 field specifying if the location is active.
  <acct_code_gst/> GL accounting code for the GST entries.
  <pst_rate/> The PST rate.
  <acct_code_hst/> GL accounting code for the HST entries.
  <notes/> Notes associated with this tax location.
  <gst_rate/> The GST rate.
</TaxLocation>

```

This datatype supports the read, add, and modify [XML Commands](#).

## TaxRate

Use the TaxRate datatype to specify tax rate information.

```

<TaxRate>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <pst/> The PST tax. Dated by the date field.
  <date/> The date (used for currency conversions).
  <notes/> Notes associated with this tax rate.
  <updated/> Time the record was last updated or modified.
  <federal/> The federal tax. Dated by the date field.
  <tax_locationid/> The ID of the associated tax location.

```

```

<state/> The state tax. Dated by the date field.
<currency/> Currency for the money fields in the record.
<hst/> The hst tax. Dated by the date field.
<slipid/> The ID of the associated slip.
<ticketid/> The ID of the associated ticket.
<gst/> The GST tax. Dated by the date field.
<purchase_itemid/> The ID of the associated purchase order item.
</TaxRate>

```

This datatype supports the read, add, and modify [XML Commands](#).

## Term

Use the Term datatype to specify term information. Terms are the customizable terminology your company uses. For example, in a doctor's office, a customer might be called a patient.

```

<Term>
  <name/> The name for the term.
  <display/> Display the term as.
</Term>

```

This datatype supports the read [XML Commands](#).

**Note:** When you use read, you can see the original or default term as well as the term that is currently being used.

## Ticket

Use the Ticket datatype to specify ticket information. A ticket, also known as a receipt, is an individual expense item for an expense report. An expense report can contain multiple tickets.

```

<Ticket>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <date/> The date of the ticket.
  <unitm/> The unit of measure.
  <reference_number/> Unique reference number within envelope.
  Used to cross-reference digital receipts with paper receipts.
  <currency_total_tax_paid/> The tax paid in the foreign currency
  if this is a foreign currency receipt.
  <tax_rateid/> The ID of the associated tax rate.
  <currency_rate/> The conversion rate if this is a foreign
  currency receipt.
  <total_no_tax/> The total paid before tax added. Dated by the
  date field.
  <project_taskid/> The ID of the associated project task.
  <cost/> The cost per unit of measure. Dated by the date field.
  <tax_location_name/> The name of the tax location.
  <non_billable/> If set to 1 this is not billable.
  <description/> The description of the ticket.
  <total/> The total value of the ticket. Dated by the date field.

```



```

<categoryid/> The ID of the associated category.
<customerid/> The ID of the associated customer.
<paymethod/> Payment method now comes from payment_type table.
Keep for backwards compatibility.
<userid/> The ID of the associated user.
<status/> The status of the ticket: R - reimbursable or N - nonreimbursable.
<currency/> Currency for the money fields in the record.
<city/> The ticket city or location.
<cost_centerid/> The ID of the associated cost center.
<slipid/> The ID of the associated slip.
<currency_cost/> The cost per unit of measure in the foreign
currency if this is a foreign currency receipt.
<payment_typeid/> The ID into the payment type field for the
payment method.
<currency_symbol/> The currency for foreign currency receipts.
    <tax_location_id/> The ID of the associated tax location.
<itemid/> The ID of the associated item.
<envelopeid/> The ID of the associated envelope.
<quantity/> The quantity.
<projectid/> The ID of the associated project.
<total_tax_paid/> The tax paid. Dated by the date field.
<vendorid/> The ID of the associated vendor.
<missing_receipt/> If set to 1, the paper receipt is missing for
this ticket.
<acct_date/> The accounting period date of the ticket.
<notes/> Notes associated with the ticket.
<attachmentid/> If non-zero, the attachment record associated
with this ticket.
<currency_exchange_intolerance/> A 1/0 field indicating if the
record is within the specified foreign currency tolerance as
defined in database data definitions.
<externalid/> If the record was imported from an external system
you store the unique external
record ID here.
<projecttask_typeid/> The ID of the associated projecttask_type.
Only if project_task_id switch is on.
<thin_client_id/> Used by thin clients to reconcile imported
records.
<user_locationid/> The location ID for this user.
</Ticket>

```

This datatype supports the read, add, modify, and delete [XML Commands](#).



**Note:** There is an OpenAir internal switch that prohibits you from editing the following fields created through the American Express receipt import wizard: date, quantity, cost, currency, payment\_typeid, and total. However, in the event that editing is necessary, you can request that the following switch be temporarily disabled: Do not allow editing of receipts with an American Express transaction number. To enable or disable the internal switch, speak with your Professional Services Consultant or create a support ticket. See [Troubleshooting](#) for instructions on creating a support ticket.

## Timecard

Use the Timecard datatype to specify timecard information.

```

<Timecard>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <time_start/> Time they started working.
  <hours/> Hours worked.
  <notes/> Notes associated with the timecard.
  <updated/> Time the record was last updated or modified.
  <userid/> The ID of the associated user.
  <date/> The date of the time card.
  <break_end/> Time they ended the break.
  <break_start/> Time they started the break.
  <timesheetid/> The ID of the associated timesheet.
  <time_end/> Time they stopped working.
</Timecard>

```

This datatype supports the read [XML Commands](#).

## Timesheet

Use the Timesheet datatype to specify timesheet information.

```

<Timesheet>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <userid/> The ID of the associated user.
  <status/> The status of the timesheet: O - Open, S - Submitted, A
  - Approved, R - Rejected, or X - Archived.
  <default_payrolltypeid/> The default payroll type ID this
  timesheet is associated with.
  <default_timetypeid/> The default time type ID this timesheet is
  associated with.
  <name/> The name of the timesheet.
  <default_customerid/> The default customer ID this timesheet is
  associated with. All new task entries get this default value.
  <submitted/> The date the timesheet was submitted.
  <total/> The total number of hours in the timesheet.
  <default_categoryid/> The default category ID this timesheet is
  associated with. All new task entries get this default value.
  <ends/> The ending date of the timesheet.
  <starts/> The starting date of the timesheet.
  <approved/> The date the timesheet was approved.
  <notes/> Notes related to this timesheet.
  <default_projectid/> The default project ID this timesheet is
  associated with.
  <acct_date/> The accounting period date of the task.
  <thin_client_id/> Used by thin clients to reconcile imported
  records.
  <history/> History of events that occurred to the TimeSheet.
  <approved_by/> Empty value kept for backwards compatibility.
  <duration/> The duration of the timesheet:
  W - Weekly


```

```

D - Daily
M - Monthly
B - Bi-weekly
S - Semi-monthly
<default_projecttaskid/> The default task id this timesheet is
associated with.
All new task entries get this default value.
<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.
</Timesheet>

```

This datatype supports the read, add, modify, submit, and delete [XML Commands](#).

 **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

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 your Professional Services Consultant or create a support ticket. See [Troubleshooting](#) for instructions on creating a support ticket.

## Timetype

Use the Timetype datatype to specify information for time types such as regular time, overtime, sick time.

```

<Timetype>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <notes/> Notes associated with this time type.
  <active/> A 1/0 field indicating whether this is time type is
  active.
  <name/> The name of the time type.
  <updated/> Time the record was last updated or modified.
  <externalid/> If the record was imported from an external system
  you store the unique external record ID here.
  <payroll_code/> The payroll code for this time type.
  <cost_centerid/> The ID of the associated cost center.
  <code/> Optional accounting system code for integration with
  external accounting systems.
  <picklist_label/> Label as shown on form picklist.

```

```
</Timetype>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Todo

Use the Todo datatype to specify information about something that needs to be done.

```
<Todo>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <priority/> Todo priority (1 - 9).
  <contactid/> The ID of the associated contact.
  <name/> The name or description of the to do item.
  <updated/> Time the record was last updated or modified.
  <due/> Date and time the task is due.
  <userid/> The ID of the associated user.
  <dealid/> The ID of the associated deal.
  <status/> Todo status: N - Not Started, C - Completed, W -
  Waiting, D - Deferred, or A - Active.
  <notes/> Notes associated with the to do item.
  <customerid/> The ID of the associated customer.
  <createdbyid/> The ID of the user who created the to do item.
  <finished/> Date and time the task was finished.
  <start/> Date and time the task is to be started.
</Todo>
```

This datatype supports the read [XML Commands](#).

## Uprate

Use the Uprate datatype to specify information about user and project rate combinations.

```
<Uprate>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <userid/> The ID of the associated user.
  <customerid/> The ID of the associated customer.
  <notes/> Notes associated with the user project rate (uprate).
  <updated/> Time the record was last updated or modified.
  <duration/> Billing rate: H - hourly or D - Daily.
  <projectid/> The ID of the associated project.
  <categoryid/> The ID of the associated category.
  <currency/> Currency for the money fields in the record.
  <rate/> The billing rate.
  <project_billing_ruleid/> If project billing rules are used,
  this is the ID of the associated project billing rule.
  <job_codeid/> The ID of the associated job code.
</Uprate>
```

This datatype supports the read, add, and modify [XML Commands](#).

**Note:** Uprate is used in conjunction with the <Company><rate\_from/></Company> setting. If <rate\_from/> is set to up, then project billing rates will come from the individual users associated to project tasks.

## User

Use the User datatype to specify user information.

```
<User>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <project_access_nodes/> Comma delimited list of hierarchy node
  IDs for project level access control.
  <addr/> The user's address.
  <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
  and 2 - approver is the manager's manager.
  <rate/> The hourly billing rate
  <br_approvalprocess/> The approvalprocess_id of the
  booking_request approval process. This field is mutually
  exclusive with br_approver.
  <password/> The user's password.
  <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
  and 2 - approver is the manager's manager.
  <sr_approver/> The user ID of the schedule request approver if
  this is a single approver process. This field is mutually
  exclusive with sr_approvalprocess. 1 - approver is the manager
  and 2 - approver is the manager's manager.
  <departmentid/> The ID of the associated department.
  <tb_filter_set/> The ID of the optional filter set for the
  Invoices module.
  <name/> The name used for display in lists. This is
  programmatically generated if not entered.
  <hierarchy_node_ids/> The IDs of the associated hierarchy nodes.
  <po_approvalprocess/> The approvalprocess_id of the purchase
  order approval process. This field is mutually exclusive with
  po_approver.
  <rm_filter_set/> The ID of the optional filter set for the
  Resources module.
  <hint/> Password hint.
  <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
  and 2 - approver is the manager's manager.
  <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
  and 2 - approver is the manager's manager.
  <az_approvalprocess/> The approvalprocess_id of the expense
```

authorization approval process. This field is mutually exclusive with az\_approver.

<pm\_filter\_set/> The ID of the optional filter set for the Projects module.

<currency/> The currency for money fields.

<cost\_centerid/> The ID of the associated cost center.

<locked/> A 1/0 field indicating if this user is locked.

<filterset\_stamp/> A unique string which changes when the primary filter set changes for the user.

<job\_codeid/> The ID of the current job code this user belongs to.

<payroll\_code/> The payroll code for this user.

<report\_filter\_set/> The ID of the optional filter set for Reporting.

<km\_filter\_set/> The ID of the optional filter set for the Workspaces module.

<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 and 2 - approver is the manager's manager.

<role\_id/> The ID of the associated role.

<dr\_approvalprocess/> The approvalprocess\_id of the deal\_booking\_request approval process. This field is mutually exclusive with br\_approver.

<te\_approvalprocess/> The approvalprocess\_id of the expense report approval process. This field is mutually exclusive with te\_approver.

<ta\_approvalprocess/> The approvalprocess\_id of the timesheet approval process. This field is mutually exclusive with ta\_approver.

<filterset\_ids/> A comma separated list of filter set IDs this record should be part of.

<active/> A 1/0 field indicating where this is designated as an active user.

<externalid/> If the record was imported from an external system, you store the unique external record ID here.

<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 and 2 - approver is the manager's manager.

<generic/> A 1/0 field indicating whether this is a generic resource.

<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 and 2 - approver is the manager's manager.

<pb\_approvalprocess/> The approvalprocess\_id of the proposals approval process. This field is mutually exclusive with pb\_approver.

<type/> Legacy field.

<workscheduleid/> The ID of the associated user workschedule.

<po\_filter\_set/> The ID of the optional filter set for the Purchases module.

<primary\_filter\_set/> The ID of the primary filter set for this user.

<user\_locationid/> The location ID for this user.  
 <account\_workscheduleid/> The ID of the associated user account workschedule.  
 <om\_filter\_set/> The ID of the optional filter set for the Opportunities module.  
 <ssn/> The users's social security number.  
 <acct\_code/> Optional accounting system code for integration with external accounting systems.  
 <ma\_filter\_set/> The ID of the optional filter set for the My Account module.  
 <te\_filter\_set/> The ID of the optional filter set for the Expenses module.  
 <sr\_approvalprocess/> The approvalprocess\_id of the schedule\_request approval process. This field is mutually exclusive with sr\_approver.  
 <nickname/> The users nickname. This must be unique.  
 <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 and 2 - approver is the manager's manager.  
 <logintime/> The date and time of the user's last login.  
 <pr\_approvalprocess/> The approvalprocess\_id of the purchase request approval process. This field is mutually exclusive with pr\_approver.  
 <line\_managerid/> The ID of this user's line manager (will actually be another user\_id).  
 <week\_starts/> The day the week starts for this user: 0 - Monday or 6 - Sunday.  
 <ta\_filter\_set/> The ID of the optional filter set for the Timesheets module.  
 <flags/> A collection of Switch values.  
 <update\_workschedule/> A 1/0 field indicating an update to the user's workschedule.  
 <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.  
 <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 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.  
 <update\_tag/> Set this field to 1 to enable automatic updating of user entity tags.

<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.

<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\_id/> The ID of the tag group for the new tag.

<tag\_group\_attribute\_id/> The ID of the tag group attribute that is being assigned to the new tag.

<update\_cost/> Set this field to 1 to enable automatic updating of user loaded cost.

<cost\_start\_date> Start date for the new loaded cost. If blank, the new cost will assume the current date as it's start date.

<cost\_end\_date/> End date for the new loaded cost. If left blank, the new cost will have no end date.

<cost/> New cost value.

<cost\_currency/> Currency of the cost.

<cost\_lc\_level/> If multiple loaded cost levels are enabled, use this field to hold the level of the loaded cost.

<timezone/> The user's timezone.

<book\_assign\_stamp/> Internal hash key.

<code/> The acct\_code.

<external\_id/> The unique external record ID if the record was imported from an external system.

<password\_forced\_change/> A 1/0 field indicating whether the password must change at next login.

<ta\_approver\_externalid/> Import-only field.

<user\_location\_externalid/> Import-only field.

<job\_code\_externalid/> Import-only field.

<role\_externalid/> Import-only field.

<pm\_filter\_set\_externalid/> Import-only field.

<tb\_filter\_set\_externalid/> Import-only field.

<pr\_approver\_externalid/> Import-only field.

<po\_approver\_externalid/> Import-only field.

<cost\_center\_externalid/> Import-only field.

<rm\_filter\_set\_externalid/> Import-only field.

<sr\_approver\_externalid/> Import-only field.

<te\_approver\_externalid/> Import-only field.

<om\_filter\_set\_externalid/> Import-only field.

<pb\_approver\_externalid/> Import-only field.

<ma\_filter\_set\_externalid/> Import-only field.

<km\_filter\_set\_externalid/> Import-only field.

<line\_manager\_externalid/> Import-only field.

<report\_filter\_set\_externalid/> Import-only field.

<dr\_approver\_externalid/> Import-only field.

<ta\_filter\_set\_externalid/> Import-only field.

<account\_workschedule\_externalid/> Import-only field.

<po\_filter\_set\_externalid/> Import-only field.

<te\_filter\_set\_externalid/> Import-only field.

<br\_approver\_externalid/> Import-only field.

<primary\_filter\_set\_externalid/> Import-only field.

<az\_approver\_externalid/> Import-only field.

<department\_externalid/> Import-only field.

<rm\_approver/>The user\_id of the booking approver if this is a single user approver process.



```

This field is mutually exclusive with rm_approvalprocess
If -1 then the approver is the manager
If -2 then the approver is the manager's manager
<rm_approvalprocess/>The approvalprocess_id of the booking
approval process. This field is mutually exclusive with
rm_approver.
<picklist_label/> Label as shown on form picklist.
<cv_attachment_id/> The ID of the user's latest CV.
</User>

```

This datatype supports the read, CreateUser, modify, and delete [XML Commands](#).

**Note:** In order to return a generic user in a read command, add a generic attribute to the read request. See [generic](#) in [Attributes](#).

**Note:** When using the <Modify/> command, the <flags/> section of the <User/> record will be ignored unless the current authorized user is an administrator. To modify flags for a particular user record, submit the <flags/> portion in the <Modify type "User"> command. Refer to the following example:

```

<Modify type="User">
  <User>
    <id>1</id>
    <flags>
      <Flag>
        <name>flag_name</name>
        <setting>X</setting>
      </Flag>
    </flags>
  </User>
</Modify>

```

**Note:** The <Delete/> command supports the "User" datatype, so you can delete user records from an account. You cannot delete a user record from the system if there are existing transactions that are associated with it. Refer to the following example:

```

<Delete type="User">
  <User>
    <id>1</id>
  </User>
</Delete>

```

## Set User Workschedule

Refer to [UserWorkschedule](#) to read user workschedule information.

### To set the user workschedule while updating or creating users:

1. Set the update\_workschedule field of the user datatype 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 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 User datatype:

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:

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`.

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.

### Refer to the following example for subsequent imports:

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.

1. Set `update_tag=1`.
2. Set `tag_start_date=blank`.
3. Set `tag_end_date=blank`.

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 its 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.

## UserLocation

Use the UserLocation datatype to specify user location information.

```
<UserLocation>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The name of the user location.
  <external_id/> The unique external record ID if the record was
  imported from an external system.
  <acct_code/> Optional accounting system code for integration
  with external accounting systems.
  <notes/> Notes associated with this user location.
  <active/> A 1/0 field indicating whether the record is active.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
</UserLocation>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## UserWorkschedule

Use the UserWorkschedule datatype to retrieve information about user-specific and company-wide work schedules.

```

<UserWorkschedule>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The company-wide schedule name for company schedules or
  user's first and last name for user schedules.
  <userid/> ID of the user if this is a users work schedule. Blank
  - if there is a company work schedule
  <use_this_schedule/> Can be blank 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 blank, then
  this is a company schedule. If blank then the user (with userid
  above) is using the company schedule indicated by
  account_workscheduleid.
  <account_workscheduleid/> The ID of the company workschedule to
  use when userid is not blank.
  <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.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
</UserWorkschedule>

```

This datatype supports the read [XML Commands](#).

## Vendor

Use the Vendor datatype to specify vendor information.

```

<Vendor>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <addr/> The vendor's address.
  <terms/> Standard payment terms for the vendor.
  <purchaseorder_text/> Text to display on every purchase order.
  <currency/> Currency for the money fields in the record. Also the
  default currency when a purchase order is created.
  <web/> Vendor's Web address.
  <code/> Optional accounting system code for integration with
  external accounting systems.
  <attention/> To whom purchase orders should be sent.
  <name/> The vendor name. This shows up on all the vendor pop-up
  windows in the application.
  <active/> A 1/0 field indicating where this is designated as an
  active vendor 1/0.
  <purchaseprder_email_text/> Extra text to include in emails
  announcing purchase orders.
  <externalid/> If the record was imported from an external
  system, store the unique external record ID here.
  <tax_locationid/> The ID of the associated tax location.
  <notes/> Notes associated with this vendor.
  <picklist_label/> Label as shown on form picklist.

```

```
</Vendor>
```

This datatype supports the read, add, modify, and delete [XML Commands](#).

## Viewfilter

Use the Viewfilter datatype to filter lists or calendars.

```
<Viewfilter>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <userid/> The user who created this filter.
  <name/> The internal name of the list or calendar this filter is
    applied to.
  <label/> The name given to this filter. It appears in the Filter:
    drop-down list.
  <action/> The filter action.
  <fields/> Comma delimited list of fields.
  <match_all/> A 1/0 field. 1 = if all rules met. 0 = if rules must
    be met.
</Viewfilter>
```

This datatype supports the read [XML Commands](#).

## Viewfilterrule

Use the Viewfilterrule datatype to specify the individual rules for a particular viewfilter.

```
<Viewfilterrule>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
  <viewfilterid/> The viewfilter to which this rule belongs.
  <field/> The field or column to be compared.
  <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.
  <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.
  <value/> The value the field is compared to.
  <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.)
</Viewfilterrule>
```

This datatype supports the read [XML Commands](#).

## Workspace

Use the Workspace datatype to specify workspace information.

```
<Workspace>
  <id/> Unique ID. Automatically assigned by the system.
  <name/> The workspace name.
  <description/> The description of the workspace.
  <date/> The date of the workspace.
  <userid/> The user id of the workspace owner.
  <notes/> Notes.
  <open/> A "1/0" field indicating whether this workspace is open.
  <allow_guests/> A "1/0" field indicating whether guests can be
  subscribed to this.
  <global/> A "1/0" field indicating 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
  <created/> Time the record was created.
  <updated/> Time the record was last updated or modified.
</Workspace>
```

This datatype supports the read, add, and modify [XML Commands](#).

## Workspacelink

Use the Workspacelink datatype to specify workspace associations with other records.

```
<Workspacelink>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
  <recordid/> The table ID the workspace is associated with.
  <url/> The URL of external link.
  <external/> A 1/0 field indicating if the record is an external
  link.
  <updated/> Time the record was last updated or modified.
  <workspaceid/> The ID of the associated workspace.
</Workspacelink>
```

This datatype supports the read, add, and modify [XML Commands](#).

## Workspaceuser

Use the Workspaceuser datatype to specify workspace user permission information.

```
<Workspaceuser>
  <id/> Unique ID. Automatically assigned by the system.
  <created/> Time the record was created.
```

```
<userid/> The ID of the associated user.  
<access/> The access permissions for the user: R - Read-only, W -  
Read/write, or A - Administrator.  
<workspaceid/> The ID of the associated workspace.  
<project_group_id/> The ID of the project group if the user was  
assigned as part of a project group.  
<updated/> Time the record was last updated or modified.  
</Workspaceuser>
```

This datatype supports the read, add, and modify [XML Commands](#).

# Setting Application Switches Via the API

Certain Company and User switches can be set via the API. Switches are settings that customize the application. They do not affect actual record data.

Switches are set using the Flag XML datatype. [Company](#) and [User](#) datatypes have a <flags/> field that supports the [Flag](#) datatype and contains company and user switches and settings. These switch fields correspond to the switch fields in the User and Company tables in the OpenAir database.

Switches set at the Company level affect the entire company account. Switches set at the User level affect only a particular user.

The XML datatype structure for a switch is as follows:

```
<Company> (or <User>)
  <flags>
    <Flag>
      <name>X</name>
      <setting>X</setting>
    </Flag>
  </flags>
</Company> (or </User>)
```

where:

- <name/> is the name of the switch.
- <setting/> is the value to which it is set.



# Customizing the Application

There are many options available that allow you to customize OpenAir to meet the needs of your company. You can access these options or switches using the XML API if the switch is enabled for either the account or the user. To obtain a list of switches supported by the system, please contact the OpenAir Support Department and open a support ticket. See [Troubleshooting](#).

The switches determine the functionality, terminology, and appearance of the accounts in your offer code. For example, you could disable entire modules so that they are not visible in your accounts. Or, you could set an option for one particular user.

Keep in mind that these options or switches are specific to offer codes, not namespaces, and affect all accounts in an offer code. Your namespace may have one or more offer codes associated with it, depending on your particular setup. (See [Connecting to the API](#).) If you have multiple namespace/key combinations, different accounts within the same namespace could have different settings if they are associated with different offer codes.

**Note:** These switches set default values for accounts in offer codes. Some, but not all, can be overridden on a per-account or per-user basis, meaning that account Administrators can change them. Feel free to contact the OpenAir Support Department or your account representative if you have any questions.

The following summarizes each of the Switch Groups in the OpenAir Switches Dictionary.

Switch Group	Description
Approval Options	In this section, there is the option to enable the submit/approve process for Proposals and the project-level approvals for timesheets and expense reports for your entire namespace. All accounts in your namespace will have the submit/approve process enabled by default. Account Administrators can later disable these options for their particular accounts.
Batch Export Options	In this section, there are options that affect the data of batch export files. Export functionality is found on the My Account > Exchange > Import/Export page.
Company-Specific Options	In this section, you will find switches created for very specific requirements.
Dashboard Options	In this section, there are display options for the My Account > Dashboard tab. You can set the option to display dashboard items with values of zero as the default for all accounts in your namespace. For example, "No Open TimeBills" may display. This setting can be changed on a per-user basis as well. You can also set a default text message of your choice to be displayed on the Dashboard tab of all accounts. Account Administrators can change this message for their accounts.
Data Entry Options	In this section, there are options to enable the Accounting code and External ID fields on the New/Edit User form. In addition, you can limit the total size of all document attachments in the accounts in your namespace and the number of items displayed in the smart drop-down list boxes.
Display Options	In this section, there are display options for fields in the application.
Email Options	In this section, there are options to hide the default URLs that appear in approval notification emails for Expenses, Timesheets, and Proposals,

Switch Group	Description
	and replace them with your own. There are also options to replace the default text.
Entity Creation	In this section, there are options that disallow the creation of certain account entities such as clients, users, services, and projects.
Entity Deletion	In this section, there are options to disallow the deletion of certain account entities such as clients, users, services, and projects.
Entity Listing	In this section, there are options that hide account entities such as clients, users, services, and projects.
Entity Manipulation	In this section, there are options to disable the de-activation of certain account entities.
Expenses Options	In this section, there are options for envelopes. The option to disable the reimbursement feature can be re-enabled on a per-account basis, but only through the OpenAir Support Department or your account representative.
Feature Prevention — Account Creation	In this section, there are options to disable the automatic creation of certain default account items such as expense items and time types. If items are disabled, accounts might not have default expense items such as mileage, copies, or airfare.
Feature Prevention — Field Edit	In this section, there are options to hide various fields in the application, including certain tax-related, user-related, and company-related fields.
Feature Prevention — Tabs	In this section, there are options to hide various tabs (and therefore pages) in the application, as well as options to hide certain batch import/export links from the Exchange tab.
FilterSet Options	In this section, there are options to enable and require filter sets.
Guest Options	In this section, there are options to enable guest viewing for modules.
Invoices Options	In this section, there is an option that disables the payment feature for invoices if another accounting system is performing the function. The feature can be re-enabled on a per-account basis, but only through the OpenAir Support Department or your account representative. Other options include disabling certain editing and reporting functions.
Module Availability	In this section, there are options to disable certain import/export features and disable access to any of the modules in the application for all the accounts in your namespace.
Module Selections	In this section, there are options to disable access to any of the modules in the application for all accounts in your namespace. However, functionality remains for these options to be re-enabled on a per-account, per-role, or per-user basis.
OpenAir Billing	In this section, there is the option to hide the My Charges tab in the application for all accounts in your namespace.
Opportunities Options	In this section, there are options for the Opportunities module.
Optional Features	In this section, there are options to enable access to certain features for all accounts in your namespace, including the VAT feature and the Vehicle feature. The VAT feature setting can be changed on a per-account basis, but only by the OpenAir Support Department or your account representative. The Vehicle feature setting can be changed by account Administrators for their particular accounts.

Switch Group	Description
Page Layout	In this section, there are options that determine the default appearance of the pages in the application for all accounts in your namespace. Several of the options have to do with the display of OpenAir and partner-specific logos and banners.
Password Options	In this section, there are password options.
Print Settings	In this section, there are default print settings for accounts in your namespace. For example, you can set PDF text format and page size (Letter, A4).
Projects Options	In this section, there are options for the Projects module.
Purchases Options	In this section, there are purchases-related options such as user privileges for submitted and approved purchase requests and POs.
Regional Settings	In this section, there are options to set the default date format, currency, and number format for all accounts in your namespace. Account Administrators can change these settings in their particular accounts.
Reporting Options	In this section, there are options to hide personal information on reports, show projected billing, enable date and timestamp, enable date filters in various reports, hide the drill-down reports tab, disable FTE forecast summary report values, show internal ids on detail reports, and enable multi-currency reporting.
Resources Options	In this section, there are options for the Resources module such as email notifications of booking changes.
Security Options	In this section, there are options that set the amount of security used for pages in the application for all accounts in your namespace. There are options to determine the amount of SSL Encryption that will be used, to set a timeout for pages, and to disallow URL sharing. Account Administrators can change these settings in their particular accounts.
Signers Options	In this section, there is the option to enable the signers feature in the Expenses and Timesheets modules, as well as the option requiring that all sign-offs be complete before an expense report or timesheet can be approved. Account Administrators can change these settings in their accounts.
Terminology	In this section, there are options to set the default terminology to be used for all accounts in your namespace. Terminology for module and account entities can be changed. Account Administrators can change these default settings in their particular accounts.
Time Settings	In this section, there are options to set the defaults for several time settings including the time zone and whether Daylight Saving Time is being used. Account Administrators may change these settings for their particular accounts, and in the case of the time zone and Daylight Saving Time settings, also for particular users in their accounts.
Timesheet Options	In this section, you will find the option to enter default text that will appear on the Submit for approval page in the Timesheets module. This text is typically used for a legal disclaimer of some sort. You will also find the option to allow approvers to edit timesheets that have been submitted.
User Proxying	In this section, there are options to disable the proxy user feature for all accounts in your namespace. If this feature is disabled, the Account > Users [User ID] > Proxy link is not available.

Switch Group	Description
Vat Settings	In this section, there is the option to set the default VAT rate for your all accounts in your namespace.
Workspace Options	In this section, there are options for certain elements within the Workspaces module.

# Other Features

Other features that are helpful in using the XML API include limiting records returned with filters, hints, and the use of IDs.

## Filters

There are several ways to filter the number of records you get back from a request. You can use an additional filter attribute when you make a Read command, you can use the user method of the Read command, use the project method of the Read command, or use the Filter datatype. Each is explained below.

## Additional Filter Attribute with Read Command

**Example 1** — to return all tickets that are in open envelopes:

```
<Read type="Ticket" method="all" filter="open-envelopes"/>
```

**Example 2** — to return a list of all approved envelopes pending reimbursement:

```
<Read type="Envelope" method="equal to" filter="nonreimbursed-envelopes">
  <Envelope>
    <status>A</status>
  </Envelope>
</Read>
```

There are a variety of additional filters you can use to limit the number of records you get back from a request. Take a few minutes and review the [Attributes](#) listed under the [Read, all](#) command. Examples are also provided that use different methods and filter attributes.

## Read, user Command

```
<Read obtype="ObjectName" method="user">
  <User>
    <id>X</id>
  </User>
</Read>
```

Returned: A list of "ObjectName" XML records that have a <userid> field equal to X (see above). Returns a failure message if "ObjectName" is a type that doesn't have a <userid> field.

## Read, project Command

```
<Read type="datatype" method="project">
  <Project>
```

```
<id>X</id>
</Project>
</Read>
```

Returned: A list of records that have a <projectid> field equal to X (see above). Make sure that the datatype used is a type that has a <projectid> field.

## Filter datatype

Example —


```
<Filter type="customer">
  <id/> the customer ID
</Filter>
```

Currently, the only list that can be filtered with this datatype is the Customer list, using the type customer as an attribute of the <Filter/> datatype. The only Read method that is supported is [Read, all](#).

## Hints

Hints appear at the bottom of OpenAir application pages. To add hints to an application page, use the following html comment tags:

```
<!--BEGIN HINT --> The hint goes here. <!--END HINT -->
```

 **Note:** Since this feature involves altering html pages, this is not a feature we generally recommend. Be cautious if you use it.

## IDs

If you need to change (modify or delete) records in an account, you must make sure you use the record IDs in your request. If you do not have the IDs, you must first request the list of IDs that you need and then parse the XML for the records desired.

For example, in order to inactivate a user record, you can read all user records in order to obtain their IDs:

```
<Read type="User" method="all"/>
```

Then you can use the ID to inactivate a particular user record:

```
<<Modify type="User">
  <User>
    <id>X</id>
    <active>0</active>
  </User>
</Modify>
```

## Remaining Limit

A daily rate limit is enforced in the OpenAir XML API. See [Limits](#) for more information on the usage limits.


To find out your remaining daily rate limit you can make the following request:

```
<Read type="RateLimit" method="all" limit="1">
</Read>
```

You will receive a response similar to the following:

```
<Read status = "0">
<RateLimit><remain_24h_error>99949</remain_24h_error></RateLimit>
</Read>
```

In this example, 99949 is the remaining limit.

 **Note:** Making this request will use up one from your daily limit.

# Code Examples

Three levels of code examples are provided: basic, intermediate, and advanced.

## Basic Example

Here is a basic example that will connect to the server using company name 'a', user name 'b' and password 'c'. It will then ask for the time and disconnect.

First we'll do it in the easier-to-read indented style:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<request API_ver="1.0" client="test app" client_ver="1.1"
namespace="rightnamespace" key="0123456789">
  <Auth>
    <Login>
      <company>a</company>
      <user>b</user>
      <password>c</password>
    </Login>
  </Auth>
  <Time>
  </Time>
</request>
```

Here is how it might look in an actual application:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?><request
API_version="1.0" client="test app"><request><Auth><Login><company>a
</company><user>b</user> <password>c </password></Login></
Auth><Time></Time> </request>
```

The server response would be:

```
<response>
  <Auth status="0"/>
  <Time status="0">
  <Date>
    <day>13</day>
    <month>2</month>
    <year>2000</year>
    <hour>23</hour>
    <minute>59</minute>
    <second>01</second>
  </Date>
  </Time>
</response>
```

Here is how it might look in an actual application:

```
<response><Auth status="0"/><Time status="0"><Date><day>13</
day><month>2 </month><year>2000</year><hour>23 </hour><minute>59</
```



```
minute><second>01 </second></Date></Time></response>
```

**Note:** The command we issued was 'Time' and the data we got back was in the 'Time' tag. Date is capitalized here because it is the name of an XML structure, as defined in XML Datatypes for *Date*. Note also that since the 'Auth' is only returning success or failure, it uses the empty tag syntax of XML. It could have returned instead `<Auth status="0"></Auth>` which is equivalent.

## Intermediate Example

For our intermediate example, we will fetch some actual information from the OA site. We will request 1000 Invoices in the system for this user starting at index 0. Subsequent requests can return more Invoices in the account. (The index value would need to be modified.)

Here is our request:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<request API_ver="1.0" client="test app" client_ver="1.1"
namespace="rightnamespace" key="0123456789">
  <Auth>
    <Login>
      <company>a</company>
      <user>b</user>
      <password>c</password>
    </Login>
  </Auth>
  <Read type="Invoice" limit="0,1000" method="all"/>
</request>
```

Here is the response:

```
<response>
  <Auth status="0"/>
  <Read status="0">
    <Invoice>
      <id>1</id>
      <number>234</number>
      <customerid>204</customerid>
      <total>99.00</total>
      <tax>0.00</tax>
      <balance>80.00</balance>
      <draw>1</draw>
      <credit/>
      <credit_reason/>
      <terms/>
      <emailed>
        <Date>
          <day>13</day>
          <month>2</month>
          <year>2000</year>
          <hour>0</hour>
          <minute>0</minute>
          <second>0</second>
        </Date>
      </emailed>
    </Invoice>
  </Read>
</response>
```

```

        </emailed>
    </Invoice>
    <Invoice>
        <id>4</id>
        <number>983</number>
        <customerid>204</customerid>
        <total>12.00</total>
        <tax>0.00</tax>
        <balance>50.00</balance>
        <draw>1</draw>
        <credit/>
        <credit_reason/>
        <terms/>
        <emailed/>
    </Invoice>
</Read>
</response>

```

## Advanced Example

This example (next two connects) shows a more useful example of the account creation and the API in general. We try to add a user to an account that doesn't exist, and then look up the error code returned.

Here is our request:

```

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<request API_ver="1.0" client="test app" client_ver="1.1"
namespace="rightnamespace" key="0123456789">
    <Auth>
        <user>admin user name</user>
        <company>barbaz</company>
        <password>mypassword</password>
    </Auth>
    <CreateUser>
        <Company>
            <nickname>barbaz</nickname>
        </Company>
        <User>
            <name>admin user name</name>
            <password>passwd</password>
            <taapprover>16</taapprover>
            <teapprover>16</teapprover>
            <addr>
                <Address>
                    <first>Bah</first>
                    <last>Foo</last>
                    <phone>123-345-6789</phone>
                </Address>
            </addr>
        </User>
    </CreateUser>
</request>

```

Here is the response:

```
<response>
  <CreateUser status="201"/>
</response>
```

To find out what status 201 means, we reconnect.

Here is our request:

```
<request>
  <Read type="Error" method="equal to">
    <Error>
      <code>201</code>
    </Error>
  </Read>
</request>
```

Here is the response:

```
<response>
  <Read status="0">
    <Error>
      <code>201</code>
      <text>Company does not exist</text>
      <comment>Create the company first, then add the user</comment>
    </Error>
  </Read>
</response>
```

# Appendix A Error Code Listing

The API returns error codes that you can use to help you identify problems with your operations. You can either refer to the tables that follow for specific [Error Codes](#) or, since an error is also a valid datatype in the XML data set, you can use the API to query the text translation. The response you receive depends of the type of error. Generally, you receive a response with a non-0 status code, which in some cases may also include the <errors> element that contains one or more textual error messages, concatenated in its inner text. See the [Note](#) under [Parser Success with Failed Commands](#). Also see the datatype for [Error](#) and refer to the following [Error Responses](#).

## Error Responses

The following illustrates the error responses you may receive based on the XML request.

### Parser Failure

If the XML request in its entirety is malformed and the parser on the server failed to load the request XML document, you will get the following response.

```
<response status="1">Badly formed XML, parsing aborted</response>
```

### Parser Success with Validation Error on Request

If the XML request was parsed successfully, but there was a specific validation error on a specific request in the request stack, the response has the following structure. For more information, see the Code Examples chapter for [Advanced Example](#).

```
<response>
  <Add status="1">An error description</Add>
</response>
```

### Parser Success with Failed Commands

If you requested a stack of commands and some fail while others succeed, you will get a mixed response. The error description is either the body of the response or command element.

```
<response>
  <CreateUser status="201"/>
  <Add status="0">....data....</Add>
  <Add status="1"/>....data....</Add>
  ...
  etc.
  ...
</response>
```



**Note:** In some cases, in addition to the non-0 error code returned in a status attribute, there is an additional textual description of error(s) in the <errors> child node under the failed request element. See the example that follows.

```
<response>
```

```

<Add status="123">
  <errors>Description of errors</errors>
</Add>
</response>

```

## Error Codes

Error codes are broken out by their type and you can search for one using the error code number. Refer to the following tables.

### 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

Error Code	Short Message	More Information
204	Namespace error	Users must be created in the same namespace as the company
205	Workschedule error	Invalid account workschedule 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

## Auth Errors

Error Code	Short Message	More Information
401	Auth failed : No such company/ user/pass	The combination of usernick/companynick/ password doesn't exist
402	Old TB login	Internal TB error
403	No Company name supplied	N/A
404	No User name supplied	N/A
405	No User Password supplied	N/A
406	Invalid Company name	N/A
408	Bad Password	N/A
409	Account Canceled	This account has been canceled
410	Inactive user	This user has been made inactive by their administrator
411	Account conflict, contact customer service	There is a problem with the account. Contacting customer service will allow you to use the account again
412	Wrong namespace for account	This account is not associated with the namespace provided
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
416	User locked	This user has been locked, please contact your Account Administrator

Error Code	Short Message	More Information
417	Restricted IP address	Access is not allow from this 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 maybe required
420	Authentication failed	If the problem keeps reoccurring, please contact your identity 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

## 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. Please 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	Please contact your OpenAir administrator to request more space

Error Code	Short Message	More Information
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

Error Code	Short Message	More Information
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



Error Code	Short Message	More Information
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	<p>This error is returned under the following conditions:</p> <ul style="list-style-type: none"> <li>- The timesheet cannot be modified, it has been submitted for approval.</li> <li>- The timesheet has status (A/X - Approved/ Archived) and internal switch allowing editing of approved timesheets is not enabled.</li> <li>- Timesheet is not in Open Period and user's role doesn't allow editing of timesheets outside of Open Periods.</li> <li>- Timesheet has status S (Submitted) and is modified by the submitter, while internal switch doesn't allow editing of submitted timesheets by owner.</li> <li>- Timesheet has been exported and internal switch disallowing modification of exported timesheets is turned on.</li> <li>- When a user who does not own a full Account Administrator role attempts to modify timesheet of another user via API.</li> </ul>
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

Error Code	Short Message	More Information
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
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	Purchaseorderid 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, please contact OpenAir 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

Error Code	Short Message	More Information
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
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

Error Code	Short Message	More Information
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
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.
1200	Condition not met	Command wasn't executed because condition wasn't met. Returning the record from DB.

Error Code	Short Message	More Information
1400	Missing start_end_month_ts flag	Please specify a valid start_end_month_ts flag for the Timesheet.
1401	Invalid associated_tmtd	Specified associated_tmtd is invalid, please consult documentation.
1402	Non-overlapping timesheet	You cannot specify associated_tmtd nor start_end_month_ts flag for non-overlapping timesheets.
1403	Cannot modify timesheet with associated_tmtd	You cannot modify specific field of associated timesheets, please consult documentation.

## 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
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

Error Code	Short Message	More Information
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
931	Duplicate project assignment	A user can only be assigned to a project once
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

Error Code	Short Message	More Information
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 be modified	This ResourceAttachment cannot be edited
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

## 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

## Filterset Errors

Error Code	Short Message	More Information
1300	Invalid filter set specified	Please specify a valid filter set

## 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



## Appendix B Simple client (in perl)

This is an extremely simple client that will demonstrate a few basic exchanges to/from the API server. You must change the 'YOURNAME' text to the name of the subdomain you have been assigned (probably your company's name). This example requires the libwww-perl modules to run. They can be found at [www.cpan.org](http://www.cpan.org) if you don't have them already. This client does not use SSL and isn't robust enough for a production environment. SSL with perl is possible, but requires you install several other modules, and was not included for that reason.

### Example:

```
#!/usr/bin/perl
use LWP::UserAgent;
use ::Request::Common;

$ua = new LWP::UserAgent;
$req = HTTP::Request->new( 'PUT' );
$req->url( https://www.openair.com/api.pl );
$content = '<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<request API_version="1.0" client="test app" client_ver="1.1"
namespace="yourco" key="0123456789">';

$test = prompt( "a) create new account\n".
                "b) universal login\n"
                "c) fetch an error code explanation\n" );

$test = lc( $test );

if ( $test ne 'a' && $test ne 'b' && $test ne 'c' ) {
    print "You entered $test, run me again and use a, b, or c\n";
    exit(0);
}

if ( $test eq 'c' ) {
    $code = prompt( "Enter the error code to look up\n" );
    $content .= "<Read type='Error' method='equal to'>".
               "<Error><code>$code</code></Error></Read></request>";
}
else {
    $company = prompt( "Enter company nickname :\n" );
    $user = prompt( "Enter user nickname:\n" );
    $pass = prompt( "Enter password\n" );

    if ( $test eq 'a' ) {
        $email = prompt("Enter email address:\n" );

        $content .= "<CreateAccount>";

        $content .= "<Company><nickname>$company</nickname></Company>".
                   "<User><nickname>$user</nickname><password>$pass</password>".
                   "<addr><Address><email>$email</email></Address></addr></User>";
    }
}
```

```

        $content .= "</CreateAccount>";
    }
    elsif ( $test eq 'b' ) {
        $content .= "<RemoteAuth>".
            "<Login><company>$company</company><user>$user</user><password>$pass
        </password></Login>".
            "</RemoteAuth>";
    }
    $content .= "</request>";
}
print "--- I'm going to send this to the server --\n";
print $content."\n";
print "--- Here is the response ---\n";

$req->content( $content );
$response = $ua->request( $req );

if ( $response->is_success )
{
    print $response->content;
}
else
{
    print $response->status_line;
}

sub prompt {
    my $text = shift;
    print $text;
    my $answer = <>;
    chomp( $answer );
    return $answer;
}

```

# Appendix C OpenAir Data Dictionary

The complete OpenAir Data Dictionary can be found at the following URL:

[http://www.openair.com/database/single\\_user.html](http://www.openair.com/database/single_user.html)

The Customer Table is presented below to show how the XML datatype structure matches field names supported by the API. Each XML Datatype is comprised of field names and descriptions. Refer to [XML Datatypes](#).

**Note:** An "X" in between XML tags shows where you would put the information itself. Remember, the XML datatypes are displayed in an indented style for readability. In your actual application, you would just use a continuous string without new lines or formatting.

## Customer Table

Name	Type	Index	Description	XML Datatype
id	Integer Auto-Increment	P	Unique ID. Automatically assigned by the system.	<pre>&lt;Customer&gt;   &lt;id&gt;X&lt;/id&gt; &lt;/Customer&gt;</pre>
primary_contact_id	Integer	Y	Unique primary contact id.	Unsupported
billing_contact_id	Integer	Y	Unique billing contact id.	<pre>&lt;Customer&gt;   &lt;billing_contact_id&gt;X &lt;/   billing_contact_id&gt; &lt;/Customer&gt;</pre>
acct_code	Varchar(75)		Optional accounting system code for integration with external accounting systems.	<pre>&lt;Customer&gt;   &lt;code&gt;X&lt;/code&gt; &lt;/Customer&gt;</pre>
external_id	Varchar(75)	Y	The place to store an external record ID if the record was imported from an external system.	<pre>&lt;Customer&gt;   &lt;externalid&gt;X&lt;/externalid&gt; &lt;/Customer&gt;</pre>
name	Varchar(75)	Y	The "nickname" used for display in pop-up windows and lists. The system will generate a name if this field is blank.	<pre>&lt;Customer&gt;   &lt;name&gt;X&lt;/name&gt; &lt;/Customer&gt;</pre>
company	Varchar(70)		The name of the company.	<pre>&lt;Customer&gt;   &lt;company&gt;X&lt;/company&gt; &lt;/Customer&gt;</pre>
last	Varchar(50)		The contact's last name.	<pre>&lt;Customer&gt;   &lt;contactaddr&gt;   &lt;Address&gt;</pre>

Name	Type	Index	Description	XML Datatype
				<pre> &lt;last&gt;X&lt;/last&gt; &lt;/Address&gt; &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>
first	Varchar(50)		The contact's first name.	<pre> &lt;Customer&gt;   &lt;contactaddr&gt;     &lt;Address&gt;       &lt;first&gt;X&lt;/first&gt;     &lt;/Address&gt;   &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>
salutation	Varchar(50)		The contact's salutation.	<pre> &lt;Customer&gt;   &lt;contactaddr&gt;     &lt;Address&gt;       &lt;salutation&gt;X&lt;/salutation&gt;     &lt;/Address&gt;   &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>
email	Varchar(50)		The contact's email address.	<pre> &lt;Customer&gt;   &lt;contactaddr&gt;     &lt;Address&gt;       &lt;email&gt;X&lt;/email&gt;     &lt;/Address&gt;   &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>
address1	Varchar(50)		Address line 1.	<pre> &lt;Customer&gt;   &lt;contactaddr&gt;     &lt;Address&gt;       &lt;addr1&gt;X&lt;/addr1&gt;     &lt;/Address&gt;   &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>
address2	Varchar(50)		Address line 2.	<pre> &lt;Customer&gt;   &lt;contactaddr&gt;     &lt;Address&gt;       &lt;addr2&gt;X&lt;/addr2&gt;     &lt;/Address&gt;   &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>
city	Varchar(50)		The city.	<pre> &lt;Customer&gt;   &lt;contactaddr&gt;     &lt;Address&gt;       &lt;city&gt;X&lt;/city&gt;     &lt;/Address&gt;   &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>
state	Varchar(25)		The State/Province.	<pre> &lt;Customer&gt;   &lt;contactaddr&gt;     &lt;Address&gt;       &lt;state&gt;X&lt;/state&gt;     &lt;/Address&gt;   &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>

Name	Type	Index	Description	XML Datatype
zip	Varchar(15)		The Zip/Postal Code.	<pre> &lt;Customer&gt;   &lt;contactaddr&gt;     &lt;Address&gt;       &lt;zip&gt;X&lt;/zip&gt;     &lt;/Address&gt;   &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>
country	Varchar(30)		The country.	<pre> &lt;Customer&gt;   &lt;contactaddr&gt;     &lt;Address&gt;       &lt;country&gt;X&lt;/country&gt;     &lt;/Address&gt;   &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>
phone	Varchar(30)		The contact's phone number.	<pre> &lt;Customer&gt;   &lt;contactaddr&gt;     &lt;Address&gt;       &lt;phone&gt;X&lt;/phone&gt;     &lt;/Address&gt;   &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>
fax	Varchar(50)		The contact's fax number.	<pre> &lt;Customer&gt;   &lt;contactaddr&gt;     &lt;Address&gt;       &lt;fax&gt;X&lt;/fax&gt;     &lt;/Address&gt;   &lt;/contactaddr&gt; &lt;/Customer&gt; </pre>
b_last	Varchar(50)		The billing contact's last name. OBSOLETE - USE FOR BACKWARDS COMPATIBILITY ONLY	<pre> &lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;       &lt;last&gt;X&lt;/last&gt;     &lt;/Address&gt;   &lt;/billingaddr&gt; &lt;/Customer&gt; </pre>
b_first	Varchar(50)		The billing contact's first name. OBSOLETE - USE FOR BACKWARDS COMPATIBILITY ONLY	<pre> &lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;       &lt;first&gt;X&lt;/first&gt;     &lt;/Address&gt;   &lt;/billingaddr&gt; &lt;/Customer&gt; </pre>
b_salutation	Varchar(50)		The billing contact's salutation. OBSOLETE - USE FOR BACKWARDS COMPATIBILITY ONLY	<pre> &lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;       &lt;salutation&gt;X&lt;/salutation&gt;     &lt;/Address&gt;   &lt;/billingaddr&gt; &lt;/Customer&gt; </pre>
b_email	Varchar(50)		The billing contact's email address. OBSOLETE - USE FOR BACKWARDS	<pre> &lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;       &lt;email&gt;X&lt;/email&gt;     &lt;/Address&gt;   &lt;/billingaddr&gt; &lt;/Customer&gt; </pre>

Name	Type	Index	Description	XML Datatype
			COMPATIBILITY ONLY	<pre>&lt;/billingaddr&gt; &lt;/Customer&gt;</pre>
b_address1	Varchar(50)		Address line 1. OBSOLETE - USE FOR BACKWARDS COMPATIBILITY ONLY	<pre>&lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;       &lt;addr1&gt;X&lt;/addr1&gt;     &lt;/Address&gt;   &lt;/billingaddr&gt; &lt;/Customer&gt;</pre>
b_address2	Varchar(50)		Address line 2. OBSOLETE - USE FOR BACKWARDS COMPATIBILITY ONLY	<pre>&lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;       &lt;addr2&gt;X&lt;/addr2&gt;     &lt;/Address&gt;   &lt;/billingaddr&gt; &lt;/Customer&gt;</pre>
b_city	Varchar(50)		The city. OBSOLETE - USE FOR BACKWARDS COMPATIBILITY ONLY	<pre>&lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;       &lt;city&gt;X&lt;/city&gt;     &lt;/Address&gt;   &lt;/billingaddr&gt; &lt;/Customer&gt;</pre>
b_state	Varchar(25)		The state. OBSOLETE- USE FOR BACKWARDS COMPATIBILITY ONLY	<pre>&lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;       &lt;state&gt;X&lt;/state&gt;     &lt;/Address&gt;   &lt;/billingaddr&gt; &lt;/Customer&gt;</pre>
b_zip	Varchar(15)		The ZIP/Postal Code. OBSOLETE - USE FOR BACKWARDS COMPATIBILITY ONLY	<pre>&lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;       &lt;zip&gt;X&lt;/zip&gt;     &lt;/Address&gt;   &lt;/billingaddr&gt; &lt;/Customer&gt;</pre>
b_country	Varchar(30)		The country. OBSOLETE - USE FOR BACKWARDS COMPATIBILITY ONLY	<pre>&lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;       &lt;country&gt;X&lt;/country&gt;     &lt;/Address&gt;   &lt;/billingaddr&gt; &lt;/Customer&gt;</pre>
b_phone	Varchar(30)		The billing contact's phone number. OBSOLETE - USE FOR BACKWARDS COMPATIBILITY ONLY	<pre>&lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;       &lt;phone&gt;X&lt;/phone&gt;     &lt;/Address&gt;   &lt;/billingaddr&gt; &lt;/Customer&gt;</pre>
b_fax	Varchar(50)		The billing contact's fax number. OBSOLETE - USE	<pre>&lt;Customer&gt;   &lt;billingaddr&gt;     &lt;Address&gt;</pre>

Name	Type	Index	Description	XML Datatype
			FOR BACKWARDS COMPATIBILITY ONLY	<pre>&lt;fax&gt;X&lt;/fax&gt; &lt;/Address&gt; &lt;/billingaddr&gt; &lt;/Customer&gt;</pre>
billing_code	Char(2)	Y	The customer billing code. It is used in bulk invoicing.	Unsupported
rate	Decimal(12,2)		The hourly billing rate.	<pre>&lt;Customer&gt; &lt;rate&gt;X&lt;/rate&gt; &lt;/Customer&gt;</pre>
invoice_text	Varchar(100)		Text to display on every invoice.	<pre>&lt;Customer&gt; &lt;invoice_text&gt;X&lt;/ invoice_text&gt; &lt;/Customer&gt;</pre>
invoice_email_text	Text		Extra text to include in emails announcing invoices.	Unsupported
invoice_prefix	Varchar(10)		Text with which to start every invoice.	Unsupported
notes	Text		Notes.	<pre>&lt;Customer&gt; &lt;notes&gt;X&lt;/notes&gt; &lt;/Customer&gt;</pre>
terms	Varchar(30)		Standard payment terms for the customer. A textual description, like "Net 30."	<pre>&lt;Customer&gt; &lt;terms&gt;X&lt;/terms&gt; &lt;/Customer&gt;</pre>
active	Char(1)	Y	A "1/0" field indicating whether the customer is active.	<pre>&lt;Customer&gt; &lt;active&gt;X&lt;/active&gt; &lt;/Customer&gt;</pre>
type	Char(1)	Y	A "C/P" field indication whether this is a Customer or a Prospect.	<pre>&lt;Customer&gt; &lt;type&gt;X&lt;/type&gt; &lt;/Customer&gt;</pre>
statements	Char(1)		A "1/0" field indicating whether the customer can view statements.	<pre>&lt;Customer&gt; &lt;statements&gt;X&lt;/statements&gt; &lt;/Customer&gt;</pre>
deleted	Char(1)	Y	A "1/0" field indicating whether the record has been deleted.	Unsupported
created	Datetime		Time the record was created.	<pre>&lt;Customer&gt; &lt;createtime&gt; &lt;Date&gt; &lt;day&gt;XX&lt;/day&gt; &lt;month&gt;XX&lt;/month&gt;</pre>

Name	Type	Index	Description	XML Datatype
				<pre>&lt;year&gt;XXXX&lt;/year&gt; &lt;minute&gt;XX&lt;/minute&gt; &lt;second&gt;XX&lt;/second&gt; &lt;/Date&gt; &lt;/createtime&gt; &lt;/Customer&gt;</pre>
updated	Timestamp		The time the record was last modified.	<pre>&lt;Customer&gt;   &lt;updatetime&gt;     &lt;Date&gt;       &lt;day&gt;XX&lt;/day&gt;       &lt;month&gt;XX&lt;/month&gt;       &lt;year&gt;XXXX&lt;/year&gt;       &lt;minute&gt;XX&lt;/minute&gt;       &lt;second&gt;XX&lt;/second&gt;     &lt;/Date&gt;   &lt;/updatetime&gt; &lt;/Customer&gt;</pre>



# Appendix D Best Practices

Before you begin using OpenAir XML 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 XML 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 XML 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 XML API](#).

- 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. 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 [Connecting to the API](#) 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 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 command 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 a code example, see code [Example 5](#) in [Read, all](#) for code to request records that are newer than a specified date.

## 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.

See code [Read, all](#) in [Read, all](#) for code to request not-yet exported records, filter by not-exported, as well as code to mark returned Slip records as being exported.

## 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 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 [Troubleshooting](#) for information on getting help from OpenAir 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.

# Troubleshooting


If you are experiencing difficulties or would like additional information, please create a support case and submit it through your OpenAir account. Use a support ticket to request API access.

## To create a support case:

1. Log in to your OpenAir account and select **Support** from the User Center menu.
2. Click on the **Go to SuiteAnswers** button.
3. From the **SuiteAnswers** site home page, click **Contact Support Online**.
4. Enter your question keywords and click **Search**.

 **Note:** If you do not have a question, i.e. you need a switch enabled, just click Search.

5. Very often the answer to your question will be displayed. If you still want to create a support case click **Continue to Create Case**.
6. Fill out the **Create Case** form and then click the **Submit**. You will receive an email confirmation with **Your OpenAir Customer Care #**.

 **Note:** An asterisk \* displays after required fields.

Our support staff and engineers will work with you to find a solution to your problem.

# New Features

## Features for April 14, 2018

The following datatypes and fields were exposed:

Datatype	Fields Exposed
ProjecttaskEstimate	id, project_task_id, user_id, timesheet_id, hours, date_changed, changed_by, created, updated

## Features for October 14, 2017

- Added [ModifyOnCondition](#).
- Added **order** attribute to [Read](#) command.

The following datatypes and fields were exposed:

Datatype	Fields Exposed
Proxy	id, user_id, proxy_id, own, role_id, expiration, created, updated
Resourceprofile_type	id, name, description, type, related_table, related_id, active, external_id, created, updated
ResourceAttachment	id, userid, attachment_id, type, latest_attachment_id, created, updated
User	cv_attachment_id
AccountingPeriod	id, name, start_date, end_date, period_date_how, period_date, current_period, default_period, notes, active, created, updated

- 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 [Category\\_1](#). (interim change)
- Enabled delete support for [Category\\_2](#). (interim change)
- Enabled delete support for [Category\\_3](#). (interim change)
- Enabled delete support for [Category\\_4](#). (interim change)
- Enabled delete support for [Category\\_5](#). (interim change)
- Enabled delete support for [Costcenter](#). (interim change)
- Enabled delete support for [Request\\_item](#). (interim change)

The following datatypes and fields were exposed:

Datatype	Fields Exposed
Purchase_item	project_taskid
Project	main_contactid
ExpensePolicy	id, customerid, projectid, description, deleted, created, updated, audit, all_items_allowed
ExpensePolicyItem	id, expense_policyid, itemid, price_max, price_fixed, currency, deleted, created, updated, audit
AttributeDescription	id, resourceprofile_typeid, attributeid, description, deleted, created, updated, audit
Attachment	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 **projecttask** datatype. See [Projecttask](#).
- Added note to clarify **Task loaded\_cost** and **project\_loaded\_cost** default functionality, and corrected their descriptions.

## Features for October 15, 2016

- Added **Unapprove** support for Envelopes, Invoices, and Timesheets.
- Error code and related information was added for Project Budget error codes 945 and 946. See [Error Codes](#).

## Expose Datatypes and Fields

The following datatypes and fields were exposed:

Datatype	Fields Exposed
ProjectBudgetGroup	approval_status, budget_by, calculated_total, cf_opt, cf_pes, created, currency, customerid, date, date_approved, date_archived, date_submitted, funding_total, id, internal_total, labor_subcategory, name, notes, parentid, profitability, projectid, setting, total, total_calculated_billing, total_calculated_cost, total_expected_billing, total_expected_cost, total_from_funding, unassigned_task, updated, userid, version
ProjectBudgetRule	category, categoryid, created, currency, customerid, date, end_date, id, imported, itemid, job_codeid, notes, period, productid, profitability, project_budget_groupid, project_taskid, projectid, quantity, quantity_best, quantity_most_likely, quantity_worst, rate, start_date, total, total_best, total_most_likely, total_worst, updated
ProjectBudgetTransaction	category, categoryid, created, currency, customerid, date, id, itemid, job_codeid, productid, project_budget_groupid, project_budget_ruleid, project_taskid, projectid, quantity, quantity_best, quantity_most_likely, quantity_worst, total, total_best, total_most_likely, total_worst, updated
ApprovalLine	id, approvalid, status, timesheetid, envelopeid, proposalid, purchaserequestid, purchaseorderid, authorizationid,

Datatype	Fields Exposed
	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, deleted, created, updated, audit, delay_to, delay_action

## Features for April 16, 2016

- It is no longer possible to rename, change, or delete a custom field which is being used by an active script. This prevents unintended script problems.

## Features for October 17, 2015

- Error code and related information was added for MakeURL error code 943. Project names must be unique by customer

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Paymenttype	default_status, default_payment_type
Slip	skip_recognition

### Changes to Existing Functionality

[Approve](#), [Reject](#), and [Submit](#) commands can now be performed for Invoices.

## Features for April 18, 2015

The following datatypes were added: [Booking\\_request](#)

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Project	rate_cardid



Datatype	Fields Exposed
Agreement, BookingType, Category, Category_1, Category_2, Category_3, Category_4, Category_5, Contact, Costcenter, Customer, Customerpo, Department, Item, Payrolltype, Project, Projectstage, Projecttask_type, Timetype, User, Vendor	picklist_label

## Features for October 18, 2014

The following datatypes were added: [ItemToUserLocation](#) and [UserLocation](#).

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Booking	source_booking_id
Projectbillingrule	assigned_user
Ticket	user_locationid

## Features for May 17, 2014

The following datatypes were added: [ResourceRequest](#), [ResourceRequestQueue](#), [ResourceSearch](#), [Workspace](#).

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Attachment	ownerid, is_a_folder, owner_type, name

## Features for February 15, 2014

The following datatypes were added: [ResourceRequest](#), [ResourceRequestQueue](#), [ResourceSearch](#), [Workspace](#).

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Address	contact_id

## Features for November 16, 2013

- The following datatype was added: [BillingSplit](#)

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Approvalprocess	externalid
LoadedCost	externalid
BookingByDay	userid
Projectbillingtransaction	customerpoid, cost_centerid, timetypeid, customerid, agreementid, payroll_typeid
SlipProjection	projecttask_typeid, cost_centerid, acct_date, job_codeid
Address	id
Invoice	submitted, approved
Contact	exported
Contact	userid, audit

## Features for August 17, 2013

- Added restriction on reading [RevenueProjection](#). This datatype cannot be read while projections are running. Added error code 606 to report this condition.
- Added [PendingBooking](#) and [ProjectAssignmentProfile](#) datatypes.

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Projectassignment	project_assignment_profile_id, pending_booking_id, booking_id
Booking	project_assignment_profile_id
User	rm_approver, rm_approvalprocess

Datatype	Fields Exposed
Project	rm_approver, rm_approvalprocess

## Features for May 18, 2013

- The following datatypes were added: [BookingByDay](#) and [RevenueProjection](#).
- Added ability to determine [Remaining Limit](#).

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Projectbillingtransaction	slip_stage_id
Slip	originating_id

## Features for March 16, 2013

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Booking	date_approved, date_submitted, approval_status

## Features for January 19, 2013

- Custom fields associated with [Budget](#) may be requested using the [Read, custom equal to](#) command.

## Features for November 17, 2012

- Provide support for "Require use of expense type price on receipts" option for Android devices.

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Item	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.
- 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 Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Booking	notify_owner
Projectbillingrule	exclude_non_billable_task
Revenue_recognition_transaction	portfolio_projectid
Slip	portfolio_projectid

## Features for May 12, 2012

- Expanded the definition of the **limit** attribute on the [Read](#) command.
- Added a reference for an internal switch to [ForexInput](#). You can have an internal switch enabled in your account for user defined reporting currencies.

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Project	portfolio_projectid, is_portfolio_project

## Features for March 17, 2012

- Add or modify custom fields inline in a single request with other native fields. To enable this behavior, supply the "enable\_custom" attribute in your add or modify request and set it to 1. See

Reading Custom Field Values Inline with Native Fields and Adding/Modifying Records with Inline Custom Field Values.

- 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: [Envelope](#), [Invoice](#), [Purchase\\_item](#), [Slip](#), [Ticket](#), and [Timesheet](#).
- Error code and related information was added for Custom Field error code 1106. One or more inline custom fields failed to be updated.
- Error code and related information was added for MakeURL error code 941. Reject User 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 [Timesheet](#).

## Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Customer	created, updated, billing_code

## Features for January 21, 2012

- The following datatype was added: [Schedulebyday](#). Custom fields associated with [Schedulebyday](#) may be requested using the [Read, custom equal to](#) command.
- Custom fields associated with [Purchaseorder](#) may now also be requested using the [Read, custom equal to](#) command.
- Custom fields associated with [Request\\_item](#) may now also be requested using the [Read, custom equal to](#) command.

## Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
<a href="#">Schedulebyday</a>	id, date, user_id, hours, base_hours, target_hours, target_base_hours, created, updated

## Features for November 19, 2011

- The following datatype was added: [RevenueStage](#)

## Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Booking	locationid
RevenueStage	id, name, revenue_stage_type, created, updated
Revenue_recognition_transaction	is_from_open_stage

## Features for September 17, 2011

- Added an Error Responses section to [Appendix A Error Code Listing](#). It includes errors you may receive for XML requests when there is Parser Failure, Parser Success with Validation Error on Request, and Parser Success with Failed Commands.
- The following datatype was added: [UserWorkschedule](#).
- XML API handles all existing task rounding rules.
- Error code and related information was added for Add/Modify error code 882.

## Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Invoice	credit_rebill_status, original_invoiceid
Project	rv_approver, rv_approvalprocess
Projectbillingrule	category_1id, category_2id, category_3id, category_4id, category_5id
RevenueContainer	project_billing_rule_filter, category_1id, category_2id, category_3id, category_4id, category_5id
Revenue_recognition_rule_amount	category_1id, category_2id, category_3id, category_4id, category_5id
Slip	ref_slipid
TaskTimecard	category_1id, category_2id, category_3id, category_4id, category_5id
UserWorkschedule	id, name, userid, use_this_schedule, account_workscheduleid, workdays, workhours, created, updated



**Important:** New Features for July 16, 2011

- The following arguments /options were added to the MakeURL command: view-invoice, dashboard-project, grid-timesheet, report-timesheet.

- Custom fields associated with [Payment](#) may now also be requested using the [Read, custom equal to](#) command.
- Custom fields associated with [User](#) may now be returned for regular as well as generic users.
- Error code and related information was added for API Login error code 556.

## Features for May 14, 2011

- The following datatype was added: RevenueContainer. Enabled support for read including CustField 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 Contact update when value was not provided.
- Added parentid field to Attachment.
- Error codes and related information were added for Add/Modify error codes 880 and 881.

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
<a href="#">Attachment</a>	parentid
<a href="#">Projecttask</a>	default_category_1, default_category_2, default_category_3, default_category_4, default_category_5
<a href="#">RevenueContainer</a>	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

- TargetUtilization records can now be added for inactive users.
- When a new customer or client 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 Projectassign and Projecttaskassign.
- 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 Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
CustField	never_copy
Task	category_1id, category_2id, category_3id, category_4id, category_5id

## Features for January 22, 2011

- Enabled custom equal to support for Paymenttype.
- Enabled modify and delete support for Attachment.
- Enabled delete support for Booking.
- Error codes and related information were added for Add/Modify error codes 878 - 879 and Custom Field error code 1105.

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Revenue_recognition_rule_amount	cost_center_id
Task	acct_date
Ticket	externalid
Timesheet	acct_date

## Features for November 20, 2010

- The following datatype was added: Projectgroup.
- Enabled add, modify, and delete support for Agreement\_to\_project.
- Enabled delete support for Entitytag.
- 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 Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Projectassign	job_codeid
Projectbillingrule	daily_rate_multiplier and job_code_filter
Projectbillingtransaction	job_codeid



Datatype	Fields Exposed
Projectgroup	id, attributes, assigned_users, created, updated, name, notes, and active
Projecttaskassign	job_codeid
RevenueContainer	asb_which_slips
Uprate	job_codeid

## Features for September 18, 2010

- The following datatypes were added: Agreement\_to\_project and IssueStatus
- The following filter was added to [Attributes](#): approved-revenue-recognition-transactions.

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Agreement_to_project	agreementid, attribute, customerid, projectid, active, created, and updated
Booking	job_code_id
Customer	sold_to_contact_id
IssueStatus	id, name, attribute, active, created, and updated
Revenue_recognition_transaction	project_billing_rule_id, job_code_id, rate, decimal_hours, hour, minute, revenue_containerid, revenue_stageid, originatingid, and offsetsid
Slip	projecttask_type_id, job_code_id, and payroll_type_id

## Features for July 17, 2010

- The following datatypes were added: Category\_1, Category\_2, Category\_3, Category\_4, and Category\_5.
- Enabled custom equal to support for Revenue\_recognition\_transaction.

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Booking	starttime and endtime

Datatype	Fields Exposed
Category_1	id, name, code, externalid, active, created, updated, and notes
Category_2	id, name, code, externalid, active, created, updated, and notes
Category_3	id, name, code, externalid, active, created, updated, and notes
Category_4	id, name, code, externalid, active, created, updated, and notes
Category_5	id, name, code, externalid, active, created, updated, and notes
Revenue_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 Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Agreement	acct_date
Customerpo	acct_date
Project	attachmentid

## Features for March 20, 2010

- 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.
- Attachment fields are now returned as part of an [Add](#) request.
- Programming Fixes, Checks, and Validations
  - Enabled “custom equal to” command for Fulfillment datatype.
  - Established imported and exported as required fields on import for ImportExport.
  - Updating ProjectBillingRule does not require that cost\_centerid to be populated.
  - Added Add and Modify commands to Schedulerequest.

### Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Actualcost	id, name, userid, date, period, currency, cost, cost_typeid, is_accrual, externalid, notes, created, updated
Attachment	attachmentid
Costcategory	id, name, active, notes, created, updated, externalid
Costtype	id, name, active, notes, created, updated, externalid
Envelope	currency_exchange_intolerance
Repeat	id, frequency, every, end, occur_number, how_end, exclude_dow, created, updated
RevenueContainer	cost_centerid
Ticket	attachmentid, currency_exchange_intolerance

## Features for January 23, 2010

- Added a new Report command handler for the following attribute types: Envelope, Timesheet, and Report. Refer to [Report](#) command.
  - email\_report: when equal to 1, a report executes and sends an email with a PDF attachment to the session user.
  - relatedid for type ="Report". ID of a saved report.
  - relatedid for type ="Timesheet". ID of a timesheet.
  - relatedid for type ="Envelope". ID of an envelope.
- Programming Fixes, Checks, and Validations
  - Enabled read support for Report.
  - Enabled modify and add support for Hierarchy.
  - Enabled modify, add, and delete support for HierarchyNode.
  - Fixed issue where user workschedule was not being set when user is created through XML.
  - 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 CreatorUser logic to more closely follow UI logic when setting user.name field.

## Expose Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Booking	owner_id

Datatype	Fields Exposed
Company	workscheduleid (read-only field)
Hierarchy	externalid
HierarchyNode	available_as_column, externalid, primary_dropdown_filter, primary_user_filterset
Report	id, userid, name, type, thin_client_context, date_created, email_report, relatedid, created, updated

## Features for November 21, 2009

- Set User Workschedule - Added the ability to set the user workschedule via the User datatype or during user account creation. See [Set User Workschedule](#).
- Programming Fixes, Checks, and Validations
  - Check for valid project and customer on slip add.
  - Made sure duplicate import\_export records are never created, specific to add calls.
  - Modified [CreateUser](#) 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 Datatypes and Fields

The following fields were exposed.

Datatype	Fields Exposed
Envelope	attachmentid
Project	pm_approver_1, pm_approver_2, pm_approver_3, payroll_type_filter
Resourceprofile	externalid
Resourceprofile_type	externalid
User	update_workschedule, is_user_schedule, workschedule_workdays, workschedule_workhours