

Applaud Web Services

Implementation Guide

Releases 11i & 12

Part No. ASSRV-01

September 2011

Employee Directory, Release 11i & R12
Part No. ASSRV-01

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

The Programs may provide links to Web sites and access to content, products, and services from third parties.

Copyright © 2011, Applaud Solutions UK Ltd

Contents

AN OVERVIEW OF APPLAUD WEB SERVICES	7
Introduction	7
Types of Web Services	7
Web Service Repository.....	7
QUERY-STYLE SERVICES	9
Anatomy of a Query-Style service	9
‘Query’ Example	9
Understanding the results of a Query-style Web Service Call	10
Web Services returning an error	11
Customizing a Query Service	11
Customization Guidelines.....	12
Customization APIs	13
Disclaimer and Warning.....	13
UPDATE-STYLE SERVICES	15
Anatomy of an Update-Style service	15
‘Update’ Example	15
Customizing a Update Service	15
Customization APIs	16
Disclaimer and Warning.....	16
DEBUGGING APPLAUD WEB SERVICES.....	17

Turning on Debug	17
Getting Debug Output.....	17
WEB SERVICE SECURITY	19
Apps Initialise	19
Service Level Security and Functions	19
Multiple GUEST accounts.....	20
Security over the internet	20
DEVELOP WEB PAGES USING OUR WEB SERVICES	21
Calling Web Services from a Web Page	21

Intended Audience

Welcome to the Applaud Web Services Guide.

The manual is intended to instruct functional experts and super users on how to the web service framework that is shared by various products in the Applaud Solutions portfolio.

This guide assumes you have a working knowledge of the following:

- The principles and customary practices of your business area.
- Standard request submission in Oracle Applications.
- The Oracle E-Business Suite user interfaces.
- Setting up menus, functions, request sets and other common Oracle E-Business Suite features

To learn more about the above, read the Oracle Applications User's Guide. If you have never used Oracle Applications or are not comfortable with the above concepts, we suggest you engage Applaud Solutions or your consultancy partner to assist you in your implementation

Do Not Use Database Tools to Modify Oracle Applications Data

It is **STRONGLY RECOMMENDED** that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Applications data unless otherwise instructed.

Oracle provides powerful tools you can use to create, store, change, retrieve, and maintain information in an Oracle database. But if you use Oracle tools such as SQL*Plus to modify Applications data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle Applications tables are interrelated, any change you make using an Oracle Applications form can update many tables at once. But when you modify Oracle Applications data using anything other than Oracle Applications, you may change a row in one table without making corresponding changes in related tables. If your tables get out of synchronization with each other, you risk retrieving erroneous information and you risk unpredictable results throughout Oracle Applications.

This product delivers new objects prefixed by 'XXAS' and code to support the solution. The same guidance applies to all data residing in these tables.

An overview of Applaud Web Services

Introduction

Many of our solutions, including all of our mobile solutions, connect to your Oracle HR instance using XML Web Services. Applaud Solutions deliver a Service Oriented Architecture built upon powerful features within your existing Oracle infrastructure. XML Web Services retrieve data from the Oracle database and send data changes back to the Oracle database when the user makes updates.

Applaud Solutions register all services in a Service Repository, which can be customized by technical developers when the out-of-the-box configuration options don't satisfy your requirements.

Types of Web Services

In general, Applaud Web Services can be categorized into two types:

1. **Query.** Those that retrieve data from Oracle
2. **Create/Update/Delete.** Those that transact changes.

Web Service Repository

All Applaud Web Services are registered in the Service Repository together with their Service Descriptor. The Service Descriptor is an XML File for each web service that describes the operations it will perform.

You can view the contents of the Service Repository by querying back rows on the table `XXAS_COM_SERVICE_REPOSITORY`, which lists all Web Services and their Service Descriptors.

If you wish to change any Web Service, the first step is to query back the Web Service in question and examine its Service Descriptor.

The following table lists a selection of the Web Services delivered by some of our products. For a complete list, please query the `XXAS_COM_SERVICE_REPOSITORY` table on your database instance.

Web Service Name	Description
QueryTaggedPeople	List all people with a particular tag
QueryPerson	Get name, email and availability details for a person

QueryPersonTags	Get the tags for a particular person.
QueryPersonQualifications	Get qualifications details for a given person
QueryPersonPeers	Get the details of the people that report into the same manager as a particular person
QueryPeopleByName	Get a list of all the people who match a particular search string
QueryAssignmentHierarchy	Get person details back in a hierarchical structure
QueryPersonManagers	Get the manager for a person (multiple return values if you have multiple assignments)
QueryPersonImage	Retrieve the photo for a person
QueryPersonExperience	Get previous experience for a person
QueryPersonDRs	Get a list of the direct reports for a person
QueryPersonContactDetails	Get the phone numbers and phone types for a person
QueryPersonAssignments	Retrieve assignment details for a person: Organization, Position, Job, Supervisor, Location and length of service.
DeletePersonTag	Remove a tag from a person
DeletePersonExperience	Remove experience from a person
UpdatePersonImage	Upload a person's photo
UpdatePersonExperience	Update their Previous Experience
UpdatePersonAboutMe	Update what a person has written about themselves
CreatePersonTag	Create a new tag for a person
CreatePersonExperience	Create some new experience for a person

The structure of a Web Service varies according to whether it is a Query service or a Create/Update/Delete service.

Query-Style Services

If the out of the box Web Services do not satisfy your requirements, they can be customized by technical developers. There are certain restrictions; you cannot override the securing functions, nor can you create brand new services, for example.

But you are allowed to change the SQL that retrieves the data from the database, which delivers a very powerful way to tailor the viewable data that your users see.

This section explores the different components of a query-style service and details how to extend them.

Anatomy of a Query-Style service

'Query' Example

The code below shows the XML Service Descriptor for the Web Service **QueryPersonByName**. This is a simple service that retrieves person_id, name, and job details for a given person.

The following column can be seen in the 'SEEDED_DESCRIPTOR' column for this service:

```
<ServiceDescriptor serviceType="QUERY" securingFunction="XXAS_SRV_Q_PER_NAME_PERM">
  <Parameters>
    <Parameter name="pSearchText" type="VARCHAR2"/>
  </Parameters>
  <SQL pageSize="25">
    SELECT papf.person_id
      ,papf.global_name short_name
      ,nvl(pos.name, nvl(pj.name, hou.name)) job_title
      /*,nvl2(img.image_id, 'Y', 'N') has_image */
    FROM   per_all_people_f papf
      ,per_all_assignments_f paaf
      ,hr_all_organization_units hou
      ,per_jobs pj
      ,hr_all_positions_f pos
      ,per_images img
    WHERE  papf.person_id = paaf.person_id
    AND    ((papf.current_employee_flag IS NOT NULL AND
      papf.current_employee_flag = 'Y' AND
      paaf.assignment_type = 'E') OR
      (papf.current_npw_flag IS NOT NULL AND
      papf.current_npw_flag = 'Y' AND
      paaf.assignment_type = 'C'))
    AND    paaf.primary_flag = 'Y'
    AND    trunc(sysdate) BETWEEN
      papf.effective_start_date AND papf.effective_end_date
    AND    trunc(sysdate) BETWEEN
      paaf.effective_start_date AND paaf.effective_end_date
    AND    paaf.organization_id = hou.organization_id
    AND    paaf.job_id = pj.job_id(+)
    AND    paaf.position_id = pos.position_id(+)
    AND    trunc(sysdate) BETWEEN
```

```

pos.effective_start_date(+) AND pos.effective_end_date(+)
AND papf.person_id = img.parent_id(+)
AND img.table_name(+) = 'PER_PEOPLE_F'
AND papf.person_id IN
(SELECT column_value
FROM TABLE(xxas_per_search_pkg.people_by_name_tbl
(:pSearchText, trunc(sysdate))))
ORDER BY 2, 3
</SQL>
</ServiceDescriptor>

```

Document	Attribute	Able to Modify?	Description
ServiceDescriptor	ServiceType	N	Set to QUERY or UPDATE
	securingFunction	N	Each Web Service is secured against an Oracle Form Function. Users granted access to this Function can call the service. If the user does not have access to that function, they cannot use the service. In this way, services can be secured in the same way as regular menu options. For more information, see the section on Web Service Security. You cannot update the name of the securing function.
Parameter	name	N	The name of the Parameter passed into the Web Service
	Type	N	The Type of Parameter passed into the Web Service; can be DATE, VARCHAR2, NUMBER or BLOB. Dates are always passed in the format YYYY-MM-DD and datetimes are passed as YYYY-MM-DD HH24:MI:SS.
SQL	pageSize	Y	The number results returned in one query to the database. The default setting is using 25. Example, if you are using the 'QueryPerson' service and enter the parameter 'Jon%', you may get 200+ results in a large organization. To improve performance, only bring back the first 25 (as specified by the pageSize) and then execute the query again to retrieve more.
	Text	Y	This is the SQL that is executed on the database when the Web Service is called. This is what you will most commonly be required to modify.

Understanding the results of a Query-style Web Service Call

Here is an example of what is returned when the Web Service QueryPersonByName is called with a search parameter of 'Erickson':

```

= <Output rsltCode="SUCCESS" rsltMsg="Success" pageSize="25"
  rowCount="4">
= <ROWSET>
= <ROW>
= <PERSON_ID>267</PERSON_ID>
  <SHORT_NAME>Erickson, Barry</SHORT_NAME>
  <JOB_TITLE>VP330.VP Human Resources</JOB_TITLE>
</ROW>

```

```

- <ROW>
  <PERSON_ID>29398</PERSON_ID>
  <SHORT_NAME>Erickson, Bobby</SHORT_NAME>
  <JOB_TITLE>IMA100.Imarketing Administrator</JOB_TITLE>
</ROW>
- <ROW>
  <PERSON_ID>10206</PERSON_ID>
  <SHORT_NAME>Erickson, Edward</SHORT_NAME>
  <JOB_TITLE>128621.Personnel Assistant..01-JAN-1951</JOB_TITLE>
</ROW>
- <ROW>
  <PERSON_ID>4807</PERSON_ID>
  <SHORT_NAME>Erickson, Peter</SHORT_NAME>
  <JOB_TITLE>220300.ASSOCIATE PROFESSOR OF HISTORY</JOB_TITLE>
</ROW>
</ROWSET>
</Output>

```

Document	Attribute	Description
<Output>	rsltCode	Set to SUCCESS or FAILURE
	rsltMsg	Set to Success or displays the error message returned from the database.
	pageSize	Shows the maximum number of possible rows returned. This should correspond to the pageSize defined against the Web Service
	rowCount	The actual number of rows returned
<ROWSET>		Marks the start of the number of rows returned
<ROW>		Marks the start of one row. Within each row, there will be same number of return results.

Using pPageNum

If you are running a query that will return many results, then the pageSize parameter limits the number of results returned. To retrieve further results, you must specify a pPageNum parameter to return you the next set of results.

For example, if a service pageSize is 10 and the rowCount is 100, the first query will only return 10 results, leaving 90 results remaining. However, if you call the service again with pPageNum=2 then the next set of 10 results (i.e., results 11-20) will be returned. In this way, you can continue to retrieve results in a performant manner.

Web Services returning an error

When a Web Service is called, it may return no results or an error may occur. In this case, the <OUTPUT> document is returned with no accompanying Rows. Here is an example of what is returned when a Query returns No Data.

```
<Output rsltCode="NODATA" rsltMsg="No data found" rowCount="0" />
```

Customizing a Query Service

Customizing a Query Service allows you to do one or both of two things:

1. Change the SQL that is used to retrieve information from the Oracle database.
2. Change the page size of the SQL query. In general, this will simply limit the maximum number of results returned. However, some components use a paged result set that allows the user to scroll through multiple 'pages' of results. An example can be seen in the search results of Employee Directory. In the latter scenario, this is the maximum number of results for one single page.

No other parts of the Service can be modified. For example, you cannot change the bind parameters or securing function.

Customization Guidelines

Care must be taken when customizing a query service: if any SQL fails to run that service will be inoperable until the problem is fixed. Here are some guidelines to help ensure a smooth user experience:

- Bind parameter names *are* case-sensitive. Take care to use the exact bind parameter names defined in the seeded Service Descriptor.
- Do not write additional bind parameters in the SQL statement – you are limited to using those already defined in the Service Descriptor.
- You do not have to use all Bind Parameters. Any parameters you don't specify are not bound at runtime.
- Do not wrap date parameters in a `to_date()` function. Date conversion is automatically performed by the service invocation framework.
- You cannot change the SELECT List: all columns in the seeded descriptor must be specified in your custom SQL statement with exactly the same column aliases.
- Where column aliases are not currently specified but you choose to select a different column instead, alias the column to the seeded column name. For example, if the seeded descriptor selects `job.job_name` and you change this to `pos.position_name` you should use '`pos.position_name job_name`'.
- Columns that are currently returned as a Number or Date (rather than a Varchar2) must continue to be returned as a Number or Date. Do not wrap these in a `to_char()` function.
- Some columns, typically primary and foreign keys or key bits of information such as a person's name require a value. Do not return NULL for these columns.
- Take care to avoid common SQL errors, such as ORA-01422 ('Exact Fetch returns more than requested number of rows').

- Write SQL that is tuned and performance tested. Many of the existing services are highly tuned; poor performance drastically impacts the user experience.
- Once you have modified a service, you will need to commit. The customization APIs listed below do not commit.

Customization APIs

Applaud Solutions provide you with a number of PL/SQL APIs to customize query services.

ⓘ Important: Never make direct updates to the `XXAS_COM_SERVICE_REPOSITORY` table.

`xxas_com_sr_pkg.customize_query_service(p_service_name, p_sql, p_page_size)`

This API allows you to change the SQL or the page size of a delivered query service.

Specify the service to change via `p_service_name`; `p_sql` parameter to change the SQL; `p_page_size` to limit the maximum number of results.

`xxas_com_sr_pkg.customize_hier_query_service(p_service_name, p_sql)`

Hierarchy-based queries are rarely used. In general, these are only used when complicated hierarchical XML is required.

Set the `p_sql` parameter to change the advanced SQL. The concept of a page size is not relevant for hierarchical XML queries.

`xxas_com_sr_pkg.clear_customization(p_service_name)`

This API clears an existing customization and reverts the service back to its seeded Service Descriptor.

Disclaimer and Warning

ⓘ Important: Applaud Solutions support the ability to modify Web Services but do not support any changes you choose to make. Nor can Applaud Support validate or verify any changes you do make. Changing Query-style Web Services in this way can cause our products to malfunction.

If you raise a support call with Applaud Solutions and are using modified Web Services, your support representative will firstly ask you to revert back to the original, shipped Web Service. If your problem does not persist after reverting back to the seeded descriptor, Applaud Solutions cannot offer further support.

Modification of Web Services is done at your own risk and should only be attempted by capable and experienced technical developers.

Update-Style Services

Update services are those which change data on the database via create, update or delete functions. Our update-style Web Services always call the Oracle supported APIs, where they are available.

Like the Query-style services, there are certain restrictions on what you can change; however, you can alter the services to make calls to your own APIs or wrapper procedures, if the standard Oracle APIs are not suitable.

Anatomy of an Update-Style service

‘Update’ Example

The code below shows the XML Service Descriptor for the Web Service **UpdatePicture**. This is a simple service that updates a person’s photo in PER_IMAGES.

The following column can be seen in the ‘SEEDDED_DESCRIPTOR’ column for this service:

```
<ServiceDescriptor serviceType="UPDATE"
securingFunction="XXAS_SRV_U_PER_IMAGE_PERM">
  <Parameters>
    <Parameter name="pPersonId" type="NUMBER"/>
    <Parameter name="pImage" type="BLOB"/>
  </Parameters>
  <PLSQLProcedure>
xxas_per_image_pkg.update_image
  </PLSQLProcedure>
</ServiceDescriptor>
```

The various attributes of an update-style service have the same purpose as for the query-style services. The one difference is the section marked ‘PLSQLProcedure’. In this section, a reference is made to a package procedure. This procedure must take the same parameters as specified in the <Parameters> section.

Customizing a Update Service

Customizing a Update Service allows you to change the PL/SQL procedure that is used to update information on the Oracle database.

No other parts of the Service can be modified. For example, you cannot change the bind parameters or securing function.

Customization APIs

Applaud Solutions provide you with a number of PL/SQL APIs to customize query services.

ⓘ Important: Never make direct updates to the `XXAS_COM_SERVICE_REPOSITORY` table.

`xxas_com_sr_pkg.customize_update_service(p_service_name, p_plsql_procedure)`

This API allows you to change the PL/SQL package that the update service calls.

Specify the service to change via `p_service_name`; `p_plsql` parameter to change the procedure name. This parameter should provide the name in the format:

`<package_name>.<procedure_name>`

`xxas_com_sr_pkg.clear_customization(p_service_name)`

This API clears an existing customization and reverts the service back to its seeded Service Descriptor.

Disclaimer and Warning

ⓘ Important: Applaud Solutions support the ability to modify Web Services but do not support any changes you choose to make. Nor can Applaud Support validate or verify any changes you do make. Changing Update-style Web Services in this way can cause data corruption.

If you raise a support call with Applaud Solutions and are using modified Web Services, your support representative will firstly ask you to revert back to the original, shipped Web Service. If your problem does not persist after reverting back to the seeded descriptor, Applaud Solutions cannot offer further support.

Modification of Web Services is done at your own risk and should only be attempted by capable and experienced technical developers.

Debugging Applaud Web Services

Extending our Web Services requires technical development. To assist in this development, we provide a comprehensive logging facility to allow for rapid debug and analysis. We also require logs to be provided when dealing with more complex support queries.

Turning on Debug

If an unexpected or unhandled error occurs during invocation of a customized Web Service, the user will most likely see a generic, user-friendly message explaining that a problem has occurred.

Our Web Service based solutions use FND Logging to capture details of any problems or exceptions that occur. During development and testing, it is recommended that you set the following profile options:

- Set **FND: Debug Log Enabled** to **Yes** at **Site Level**
- Set **FND: Debug Log Level** to **Error** at **Site Level**

Note: to turn debug 'off', revert the profile values back to what they were originally.

Next, clear the cache using the Functional Administrator responsibility using the following navigation path:

Functional Administrator > Core Services > Caching Framework > Global Configuration > Clear All Cache

This will ensure that any unexpected problems that occur during service invocation, together with the users context at the time of the error, are logged in `fnd_log_messages`.

Getting Debug Output

With Debug turned 'on', perform the actions you wish to investigate. For product error, reproduce the error to create debug output.

To retrieve the debug output, run the following query:

```
SELECT *  
FROM    fnd_log_messages
```

```
WHERE    module like 'xxas%'
ORDER BY log_sequence desc
```

Web Service Security

When tasked with implementing Web Service based solutions, particularly mobile solutions, security is often the number one concern of IT teams. Our solutions take the best practice approach, recommended by Oracle and make use of standard tools like function security. All our services perform an 'apps initialise' to simulate a user logging one with secure credentials and all our mobile solutions are able to work across HTTPS.

Apps Initialise

All services perform an apps.initialise when called, using the standard Oracle username and password credentials. Solutions which operate in a 'read-only' mode still connect to the database via your GUEST account.

When a web service queries or updates, the database, it does so under a new session that has been authenticated. This allows Web Services to adhere to regular e-Business Suite Security attached to that user.

Service Level Security and Functions

Each Web Service has a 'securing function'. For example, the Web Service QueryPerson has the securing function XXAS_SRV_Q_PER_PERM. The Service UpdatePicture has the securing function XXAS_SRV_U_PER_IMAGE_PERM.

All securing functions have the name format XXAS_SRV_<Q for Query or U for Update>_<Descriptor>.

These functions are regular Form Functions that you can view in the System Administration->Application->Function form.

These functions are then granted to users (and, possible, the GUEST user). Users have to be granted a function for them to be to access it. The Quick Start guide for some products, like Employee Directory or the Self Service Org Chart, instruct you to grant all Functions used by that solution to All Users, including GUEST.

You can introduce a finer level of security by removing permissions for particular functions. This is useful if there are some feature you wish to disable or you want to control who can access the Services externally. For example, if you did not want all users to be able to upload their picture, you could remove their access to the function XXAS_SRV_PER_IMAGE_PERM.

Multiple GUEST accounts

If you have a multi-tenant environment or use the Oracle e-Business Suite in a Shared Service Center, then you may require separate 'GUEST' accounts for solutions that can be deployed to your intranet, such as Employee Directory. You may also require different GUEST accounts for each Business Group within your organization.

To achieve this you must set up multiple GUEST user accounts. You may use any naming convention, but we recommend using the name format 'GUEST_<descriptor>'.

When you deploy a solution in this way, you specify which account to use; this is the account under which an apps.initialise will be performed. The method used to specify which account to use will vary by solution and will be documented in the relevant implementation guide.

Security over the internet

Applaud Web Services are used by all our Mobile Solutions. Depending on the existing infrastructure, additional preparations may be needed to ensure secure communication over private and public networks. Applaud strongly recommends that one of the following security options is deployed:

1. Use Secure HTTPS if the Oracle E-Business Suite environment is exposed outside your organization's firewall. This is essential, even if it is exposed through a demilitarized zone (DMZ), because passwords are transmitted using Basic Access Authentication which is only secure when used in combination with Secure HTTPS. This will protect passwords and allow private information to be communicated securely over a public network.
2. Use your phone's VPN and Wi-Fi capabilities to provide secure access to Oracle E-Business Suite environments that are only available inside your organization's firewall. Users can manually configure VPN or Wi-Fi settings, or you can leverage the phone's enterprise features to remotely configure settings, ensure compliance with corporate policies, and rollout features such as VPN On Demand which makes VPN connections transparent to the user.

Develop Web Pages using our Web Services

Your license agreement with Applaud Solutions permits you to make use of any of the Web Services we deliver into XXAS_COM_SERVICE_REPOSITORY. You may use the web services within your own organization, independently of our software. For example, you might choose to use a web service to obtain data from your Oracle instance to interface into a 3rd party system, or to display information within your corporate intranet.

Calling Web Services from a Web Page

To call our Web Services from any of your internal web pages or system, you need to make a URL call with the following structure:

```
<Internal e-Business Suite URL: port number>/OA_HTML/xxas/<Applaud  
JSP>?pSrvName=<service_name>?pParameter1=<parameter 1  
value>.....?pParameterN=<parameter>
```

Internal e-Business Suite URL: port number

This should just be the URL you normally access to login to the e-Business Suite.

Example: <http://ebs.yourcompany.com:8000/>

Applaud JSP

The Applaud JSP handles the routing of the Web Service request to the database and is delivered & installed into the \$OA_HTML directory by the same patch that delivers the Web Services. There are two JSPs used:

- XxasXmlGetBlob.jsp – used to return binary values, such as Photos, documents, etc
- XxasXmlSrv.jsp – used for character string based return values

Service details

Each service is defined in the table XXAS_COM_SRV_REPOSITORY. Set the parameter pSrvName to the name of the service being called, e.g., QueryPerson and then list out all parameters required to call the service, e.g., pPersonId=5435. When passing dates, use the format YYYY-MM-DD. When passing times, use YYYY-MM-DD HH24:MI:SS.

Example

The following example will execute the QueryPersonByName service:

http://prod.yourcompany.com:8000/OA_HTML/xxas/XsasXmlSrv.jsp?pSrvName=QueryPeopleByName&pSearchText=erickson&pEffectiveDate=2011-09-23