

ellucian.

Banner Advancement Web Services Handbook

Release 8.7
December 2013



Banner®, Colleague®, PowerCampus™, and Luminis® are trademarks of Ellucian Company L.P. or its affiliates and are registered in the U.S. and other countries. Ellucian®, Ellucian Advance™, Ellucian Degree Works™, Ellucian Course Signals™, Ellucian SmartCall™, and Ellucian Recruiter™ are trademarks of Ellucian Company L.P. or its affiliates. Other names may be trademarks of their respective owners.

©2009-2013 Ellucian Company L.P. and its affiliates.

Contains confidential and proprietary information of Ellucian and its subsidiaries. Use of these materials is limited to Ellucian licensees, and is subject to the terms and conditions of one or more written license agreements between Ellucian and the licensee in question.

In preparing and providing this publication, Ellucian is not rendering legal, accounting, or other similar professional services. Ellucian makes no claims that an institution's use of this publication or the software for which it is provided will guarantee compliance with applicable federal or state laws, rules, or regulations. Each organization should seek legal, accounting and other similar professional services from competent providers of the organization's own choosing.

Prepared by: Ellucian
4375 Fair Lakes Court
Fairfax, Virginia 22033
United States of America

Revision History

| Publication Date | Summary |
|-------------------------|--|
| December 2013 | New version that supports Banner Advancement 8.7 software. |

Contents

Chapter 1 Introduction..... 5

- What is a Web service? 5**
- What is the GetGivingHistory Web service? 5**
- What is the Banner Advancement Web Services Adapter? 5**
- Contents of this handbook 6**

Chapter 2 Install Banner Advancement Web Services Adapter.....7

- Requirements 7**
- Installation steps 7**
 - Step 1 Create an OC4J instance (optional) 8
 - Step 2 Install the adapter 9
 - Step 3 Define the adapter data source. 11
 - Step 4 Define the Translation Service data source 15
 - Step 5 Configure the security role and user 16
 - Step 6 Enable schema validation (optional) 19
 - Step 7 Configure logging. 21
 - Step 8 Update the WSDL endpoint URL (optional) 22
 - Step 9 Test the Banner Advancement Web services (optional) 23

Chapter 3 GetGivingHistory Web Service25

- Message exchange 25**
 - GivingHistoryRequest 25
 - GivingHistoryResponse 26
- SOAP fault messages. 26**

| | |
|---|-----------|
| Message mapping to Banner | 26 |
| GivingHistoryRequest | 27 |
| GivingHistoryResponse | 27 |
| Intended usage. | 28 |
| Setup requirements | 28 |
| Test cases. | 28 |
| Advancement individual with giving history | 28 |
| Advancement individual without giving history | 30 |
| Invalid advancement individual ID | 30 |

1 Introduction

This chapter introduces Web services, describes the GetGivingHistory Web service, and describes the Banner® Advancement Web Services Adapter.

What is a Web service?

A Web service exposes an application's processing logic to support a service-oriented architecture and to facilitate integration with external systems. A Web service allows an external system or business process to invoke the application's logic without having to understand the application's internal structure.

Web services can be thought of as application programming interfaces (APIs). What distinguishes Web services, however, is their foundation in open, Internet-based standards. This makes them relevant to application integration both within and between organizations. Standards such as XML, SOAP, WSDL, and UDDI provide cross-platform compatibility that does not depend on a single programming language or network transport.

What is the GetGivingHistory Web service?

The GetGivingHistory Web service allows external systems to request giving history information for a donor who is set up in Banner. The Web service returns summary giving information and a list of gifts for the donor.

What is the Banner Advancement Web Services Adapter?

The Banner Advancement Web Services Adapter is a Java-based application that exposes the GetGivingHistory Web service. This exposure makes the Banner Advancement functions available to external systems using the SOAP protocol over HTTP/HTTPS. External systems interact with the Web service, which in turn is supported by Banner APIs. This layered approach provides an insulating buffer between external systems and Banner. External systems do not interact with Banner directly, but rather exchange XML messages with the exposed Web service.

The Banner Advancement Web Services Adapter supports the synchronous, request/reply message exchange pattern as follows:

1. The external system requests a service of Banner by sending an XML message to the Web service endpoint exposed by the adapter. The message contains the information required for Banner to service the request.
2. The Banner Advancement Web Services Adapter invokes the appropriate Banner API.
3. The Banner API performs the necessary Banner processing logic.
4. One of the following occurs:
 - 4.1. If the action is completed successfully, the API provides a response message, which the adapter forwards to the external system.
 - 4.2. If the action is not completed successfully, the adapter sends an error message (called a SOAP fault) to the external system.

Contents of this handbook

This handbook provides the steps that systems administrators use to install the Banner Advancement Web Services Adapter.

This handbook also includes the following information that integration analysts and business analysts use to understand the GetGivingHistory Web service:

- General description
- Message exchange pattern and specific messages used
- Mapping between message elements/attributes and Banner columns
- Intended usage
- Setup requirements
- Test cases

2 Install Banner Advancement Web Services Adapter



The Banner® Advancement Web Services Adapter exposes the GetGivingHistory Web service. This chapter gives instructions for installing the adapter on Oracle Application Server 10.1.2.x.

Requirements

The Banner Advancement Web Services Adapter requires the following:

- Oracle Application Server (OAS) 10.1.2.x and Java 1.4
- Banner Translation Service data source



Note

The adapter does *not* require the Banner Translation Service. The adapter, however, does require the definition of a Translation Service data source. This chapter includes a step for creating the Translation Service data source. If you are deploying the adapter into the same OC4J container with other Banner Web service adapters, then the data source should already be defined. ■

- Patch p1-5j4fyo_alu80100 or Banner 8.2 (contains objects that create an API in Banner Advancement and expose it as a Web service)

Installation steps

The Banner Advancement Web Services Adapter is packaged as a J2EE compatible enterprise archive file named `advancement_services.ear`.

Use the following steps to install the adapter:

- [Step 1, “Create an OC4J instance \(optional\)”](#)
- [Step 2, “Install the adapter”](#)
- [Step 3, “Define the adapter data source”](#)
- [Step 4, “Define the Translation Service data source”](#)

- [Step 5, “Configure the security role and user”](#)
- [Step 6, “Enable schema validation \(optional\)”](#)
- [Step 7, “Configure logging”](#)
- [Step 8, “Update the WSDL endpoint URL \(optional\)”](#)
- [Step 9, “Test the Banner Advancement Web services \(optional\)”](#)

Important notes:

- The installation steps in this chapter are based on Oracle Application Server 10.1.2.x and Java 1.4. If you are using another version, refer to your application server documentation for instructions specific to that application server.
- You should dedicate one OC4J instance to the Banner Web Services Adapters so that you can separately administer the Web services environment. You can install the Banner Advancement Web Services Adapter in this instance with other adapters, or you can install the Banner Advancement Web Services Adapter in a separate instance. Consult with your database administrator to determine which method is preferable for your institution.

Step 1 Create an OC4J instance (optional)

If you are deploying the Banner Advancement Web Services Adapter to an existing OC4J instance used for Banner Web services, you can skip this step. Otherwise, use the following steps to create a new OC4J instance for the adapter.

1. Connect to the Oracle Enterprise Manager. The Home page is displayed.

The screenshot shows the Oracle Enterprise Manager Home page. At the top, there are navigation tabs: Home, J2EE Applications, Ports, Infrastructure, and Backup/Recovery. Below this, there are three main sections: General, CPU Usage, and Memory Usage.

General: Shows a status icon (a red circle with a white exclamation mark) and buttons for Stop All, Restart All, and Start All. The status is Down. Host is octopus.sungardhe.com, Version is 10.1.2.0.2, Installation Type is Forms and Reports Services, and Oracle Home is /u01/app/oracle/10gASfr/10.1.2.

CPU Usage: A pie chart showing 2% Application Server, 89% Idle, and 9% Other.

Memory Usage: A pie chart showing 33% Application Server (1,039MB), 2% Free (75MB), and 65% Other (2,077MB).

System Components: A table with buttons for Start, Stop, Restart, and Delete OC4J Instance. It also has buttons for Enable/Disable Components and Create OC4J Instance. The table lists several components with their status, start times, CPU usage, and memory usage.

| Select Name | Status | Start Time | CPU Usage (%) | Memory Usage (MB) |
|--------------------------------------|--------|------------------------|---------------|-------------------|
| <input type="checkbox"/> bannerOH | ↑ | Jan 8, 2011 8:26:54 AM | 0.00 | 60.46 |
| <input type="checkbox"/> BEIS_8_1_2 | ↑ | Jan 8, 2011 8:26:54 AM | 0.00 | 60.52 |
| <input type="checkbox"/> BEIS_8.1 | ↓ | Unavailable | Unavailable | Unavailable |
| <input type="checkbox"/> Forms | ↑ | Jan 8, 2011 8:26:54 AM | 0.00 | 0.00 |
| <input type="checkbox"/> home | ↑ | Jan 8, 2011 8:26:54 AM | 0.00 | 59.32 |
| <input type="checkbox"/> HTTP_Server | ↑ | Jan 8, 2011 8:26:55 AM | 0.73 | 207.11 |

- Click **Create OC4J Instance**. The Create OC4J Instance page is displayed.

Create OC4J Instance

Enter the name of the OC4J instance you wish to create. Cancel Create

* OC4J Instance Name

Cancel Create

- Enter the name of the new instance in the **OC4J Instance Name** field. You can enter any meaningful name for the new instance.
- Click **Create**. The instance is created, and a confirmation message is displayed.
- Click **OK**. The Home page is redisplayed with the new instance in the System Components list.

Step 2 Install the adapter

Before beginning this step, you must understand the concepts published by Oracle regarding the deployment of ear files.

Use the following steps to deploy the adapter to the Oracle Application Server.

- Connect to the Oracle Enterprise Manager. The Home page is displayed.

The screenshot shows the Oracle Enterprise Manager Home page. At the top, there are navigation tabs: Home, J2EE Applications, Ports, Infrastructure, and Backup/Recovery. Below the navigation, there are three main sections: General, CPU Usage, and Memory Usage.

General section shows a status of **Down** with buttons for Stop All, Restart All, and Start All. It lists the host as octopus.sungardhe.com, version 10.1.2.0.2, installation type as Forms and Reports Services, and Oracle Home as /u01/app/oracle/10gASfr/10.1.2.

CPU Usage section shows a pie chart with the following data:

| | |
|--------------------|-----|
| Application Server | 1% |
| Idle | 85% |
| Other | 14% |

Memory Usage section shows a pie chart with the following data:

| | | |
|--------------------|-----|---------|
| Application Server | 32% | 1,032MB |
| Free | 2% | 79MB |
| Other | 66% | 2,080MB |

System Components section shows a table with buttons for Start, Stop, Restart, and Delete OC4J Instance. The table lists the following components:

| Select Name | Status | Start Time | CPU Usage (%) | Memory Usage (MB) |
|-------------------------------------|--------|-------------------------|---------------|-------------------|
| <input type="checkbox"/> bannerOH | ↑ | Jan 8, 2011 8:26:53 AM | 0.00 | 60.46 |
| <input type="checkbox"/> BEIS_8_1_2 | ↑ | Jan 8, 2011 8:26:54 AM | 0.00 | 60.52 |
| <input type="checkbox"/> BEIS_8.1 | ↓ | Unavailable | Unavailable | Unavailable |
| <input type="checkbox"/> BWS | ↑ | Jan 19, 2011 6:56:19 AM | 0.00 | 115.18 |

- Click the name of the OC4J instance that will host the adapter. The **Home** tab for the selected instance is displayed.

OC4J: BWS

Home Applications Administration

General

Status **Up**
Start Time **Jan 19, 2011 6:56:11 AM**
Virtual Machines **1**

JDBC Usage

Open JDBC Connections **1**
Total JDBC Connections **1**
Active Transactions **0**
Transaction Commits **0**
Transaction Rollbacks **0**

Status

CPU Usage (%) **0.00**
Memory Usage (MB) **115.18**
Heap Usage (MB) **8.57**

Response - Servlets and JSPs

Active Sessions **0**
Active Requests **1**
Request Processing Time (seconds) **0.001**
Requests per Second **0.09**

Response - EJBs

Active EJB Methods **0**
Method Execution Time (seconds) **0.00**
Method Execution Rate (per second) **0.00**

Related Link: [All Metrics](#)

Home Applications Administration

3. Select the **Applications** tab. A list of deployed applications is displayed.

OC4J: BWS

Home Applications Administration

Default Application Name **default**
Default Application Path **application.xml**

Deployed Applications

| Select | Name | Path | Parent Application | Active Requests | Request Processing Time (seconds) | Active EJB Methods |
|----------------------------------|--------------------|--|--------------------|-----------------|-----------------------------------|--------------------|
| <input checked="" type="radio"/> | Campuscard | ../applications/Campuscard.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> | housing | ../applications/housing.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> | pci | ../applications/pci.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> | TranslationService | ../applications/TranslationService.ear | default | 0 | 0.00 | 0 |

Home Applications Administration

4. Click **Deploy EAR File**. The Deploy Application page is displayed.

Deploy Application

For a J2EE application to be successfully deployed on the OC4J container, the application has to be assembled correctly as an Enterprise Archive (ear) file, with all the needed application and module deployment descriptors. The OC4J container generates default OC4J specific deployment descriptors when the application is deployed. If you have custom OC4J specific deployment descriptors that you wish to use, you need to include these in the ear file.

J2EE Application

* Application Name

Parent Application

5. Select the file to be uploaded:

- 5.1. In the **J2EE Application** field, click **Browse** and navigate to the `advancement_services.ear` file.

- 5.2. Select the file and click **Open**.
6. Enter the **Application Name** (for example, *Advancement_Services*).
7. Click **Continue**. The Deploy Application: URL Mapping for Web Modules page is displayed, showing the configurable part of the default URL mapping that is defined in the ear file.

Deploy Application: URL Mapping for Web Modules

A web module needs to be mapped to an URL pattern in the default web site before it can be accessed. The following table lists all the web modules found in your application. Specify the URL mapping for each of these modules.

| Name | URL Mapping |
|------|---|
| | <input type="text" value="/advancement"/> |

8. (Optional) If multiple adapters will be deployed on the same server in different OC4J instances, use the following steps to change the URL mapping for the adapter that you are currently installing. This provides a unique URL for the instance and helps differentiate the various adapters.

- 8.1. Change the entry in the **URL Mapping** field by adding descriptive text to the default string (for example, `/advancement/test` or `/advancement/prod`). Adding descriptive text to the default string, rather than changing the entire default string, is preferable.

- 8.2. Note the new context root:

- 8.3. Make sure this new context root is used in any subsequent steps that refer to the default URL.

9. Click **Finish**. The Deploy Application: Review page is displayed.

Deploy Application: Review

| | |
|----------------------------|--------------------------|
| Ear File to Deploy | advancement_services.ear |
| Deployment Destination | Instance BWS |
| URLs Mapped to Application | /advancement |

TIP The HTTP listener will be restarted after deployment, to pick up the new web module mappings.

10. Click **Deploy**. The file is uploaded and a confirmation page is displayed.
11. Click **OK**. The **Applications** tab is displayed with the deployed application.

Step 3 Define the adapter data source

A data source provides the connection properties to the Banner database. By default, the adapter needs a data source with lookup name `jdbc/bannerws`. If you previously

installed a Banner Web Services Adapter in the OC4J instance, then the data source should already exist and you can skip this step. Otherwise, use the following steps to define the data source.

1. On the **Applications** tab, select the Banner Advancement Web Services Adapter.

OC4J: BWS

Home Applications Administration

Default Application Name default
Default Application Path application.xml

Deployed Applications Deploy EAR file Deploy WAR file

Edit Undeploy Redeploy

| Select Name | Path | Parent Application | Active Requests | Request Processing Time (seconds) | Active EJB Methods |
|---|--|--------------------|-----------------|-----------------------------------|--------------------|
| <input checked="" type="radio"/> Advancement_Services | ../applications/Advancement_Services.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> Campuscard | ../applications/Campuscard.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> housing | ../applications/housing.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> pci | ../applications/pci.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> TranslationService | ../applications/TranslationService.ear | default | 0 | 0.00 | 0 |

Home Applications Administration

2. Select the **Administration** tab. A list of options is displayed.

OC4J: BWS

Home Applications Administration

Instance Properties
Server Properties
Website Properties
JSP Container Properties
Replication Properties
Advanced Properties

Application Defaults
Data Sources
Security
JMS Providers
Global Web Module

Related Links
ADF Business Components

OC4J Inheritance
OC4J applications have a hierarchical parent-child relationship to facilitate administration through inheritance. A child application inherits certain attributes from its parent application such as principals and JNDI objects including data sources, JMS providers and EJBs. When an OC4J application is deployed, you specify the parent application. The Default Application is the top of the parent hierarchy.

Home Applications Administration

3. Select **Data Sources** from the Application Defaults section. The Data Sources page is displayed showing the current data sources for the instance.

Data Sources

This table contains all the data sources configured for this application. Each data source is bound to the specified JNDI location.

Create

Select a Data Source and... Edit Create Like Delete

| Select Name | JNDI Location | Class | JDBC Driver | Monitor Performance |
|--------------------------------|-------------------|--|---------------------------------|---------------------|
| <input type="radio"/> OracleDS | jdbc/OracleCoreDS | com.evermind.sql.DriverManagerDataSource | oracle.jdbc.driver.OracleDriver | |

4. Click **Create**. The Create Data Source page is displayed with the following sections:

- General
- Datasource Username and Password
- JNDI Locations
- Connection Attributes
- Properties - not needed.

5. Enter the following information in the General section:

General

| | |
|---------------------|--|
| * Name | bannerwsDS |
| Description | |
| * Data Source Class | com.evermind.sql.DriverManagerDataSource |
| JDBC URL | jdbc:oracle:thin:@maldevs10:1521:s10b80 |
| JDBC Driver | oracle.jdbc.driver.OracleDriver <small>This field is required if you are using a generic Orion Data Source Class.</small> |
| Schema | |

Name *bannerwsDS*
(This is an example. Enter the name of your choice.)

Data Source Class *com.evermind.sql.DriverManagerDataSource*

JDBC URL *jdbc:oracle:thin:@host:port:SID* where
host = database host
port = database listener port (usually 1521 or 1549)
SID = database instance

JDBC Driver *oracle.jdbc.driver.OracleDriver*

6. Enter the following information in the Datasource Username and Password section:

Datasource Username and Password

Cleartext passwords may pose a security risk, especially if the permissions on the data-sources.xml configuration file allows it to be read by any user. You can specify an indirect password to avoid this risk. An indirect password is used to do a look up in the User Manager to get the password.

| | |
|---|---|
| Username | integmgr |
| <input checked="" type="radio"/> Use Cleartext Password | Password ***** |
| <input type="radio"/> Use Indirect Password | Indirect Password <small>example: Scott, customers/Scott</small> |

Username *integmgr*

Use Cleartext Password Select the **Use Cleartext Password** button and provide a password for the `integmgr` Oracle user.

7. Enter the following information in the JNDI Locations section:

JNDI Locations

For an emulated Data Source, please specify all three location attributes. It is recommended that you reference the EJB Location attribute in your code to look up this Data Source. For a non-emulated Data Source, the location attribute is all that is needed.

| | |
|----------------------------|---|
| * Location | <input type="text" value="jdbc/bannerws"/> |
| Transactional(XA) Location | <input type="text" value="jdbc/xa/bannerwsXA"/> |
| EJB Location | <input type="text" value="jdbc/bannerws"/> |

For emulated data sources, retrieve the data source using the JNDI value in this field.

Location *jdbc/bannerws*

Transactional(XA) Location *jdbc/xa/bannerwsXA*

EJB Location *jdbc/bannerws*

8. Enter the following information in the Connection Attributes section:

Connection Attributes

| | |
|---|---------------------------------|
| Connection Retry Interval (seconds) | <input type="text" value="1"/> |
| Max Connection Attempts | <input type="text" value=""/> |
| Cached Connection Inactivity Timeout (seconds) | <input type="text" value="30"/> |
| <small>The following attributes only apply if you are using pooled data sources</small> | |
| Maximum Open Connections | <input type="text" value="5"/> |
| Minimum Open Connections | <input type="text" value="1"/> |
| Wait For Free Connection Timeout (seconds) | <input type="text" value=""/> |

Connection Retry Interval (seconds) *1*

Cached Connection Inactivity Timeout (seconds) *30*

Maximum Open Connections Recommended value is 5. Depending on the usage, this value can be tuned.

Minimum Open Connections Recommended value is 1.

9. Click **Create**. The required data source is created in `data-sources.xml` in the `<IAS_HOME>/j2ee/<OC4J instance>/config` directory. A message confirms the action and asks if you want to restart the server.
10. Click **No**. The Data Sources page is redisplayed with the new data source.

Step 4 Define the Translation Service data source

By default, the adapter needs a data source with lookup name `jdbc/transsvc`. If you previously installed the data source in the OC4J instance, then you can skip this step. Otherwise, use the following steps to define the data source.

1. On the Data Sources page, select the data source that you just created.

Data Sources

This table contains all the data sources configured for this application. Each data source is bound to the specified JNDI location.

| Select Name | JNDI Location | Class | JDBC Driver | Monitor Performance |
|--|-------------------|--|---------------------------------|---------------------|
| <input checked="" type="radio"/> bannerwDS | jdbc/bannerws | com.evermind.sql.DriverManagerDataSource | oracle.jdbc.driver.OracleDriver | |
| <input type="radio"/> OracleDS | jdbc/OracleCoreDS | com.evermind.sql.DriverManagerDataSource | oracle.jdbc.driver.OracleDriver | |

2. Click **Create Like**.
3. Enter the following information in the General section, keeping the default values in all other fields:

Name *Transsvc*
(This is an example. Enter the name of your choice.)

4. Enter the following information in the JNDI Locations section:

Location *jdbc/transsvc*

Transactional(XA) Location *jdbc/xa/transsvc*

EJB Location *jdbc/transsvc*

5. Click **Create**. The required data source is created in `data-sources.xml` in the `<IAS_HOME>/j2ee/<OC4J instance>/config` directory. A message confirms the action and asks if you want to restart the server.
6. Click **Yes**. The server is restarted, and a confirmation message is displayed.

7. Click **OK**. The Data Sources page is redisplayed with the new data source.

Step 5 Configure the security role and user

Before beginning this step, refer to the security configuration for your version of the Oracle Application Server.

Use the following steps to add the `bannerws` role and an administrative user to the adapter. This role and user protect the defined endpoint.

1. Navigate to the **Applications** tab for the instance where the adapter is installed.
2. Select the adapter for which security roles and users are to be defined.

OC4J: BWS

Home Applications Administration

Default Application Name default
Default Application Path application.xml

Deployed Applications Deploy EAR file Deploy WAR file

Edit Undeploy Redeploy

| Select Name | Path | Parent Application | Active Requests | Request Processing Time (seconds) | Active EJB Methods |
|---|--|--------------------|-----------------|-----------------------------------|--------------------|
| <input checked="" type="radio"/> Advancement_Services | ../applications/Advancement_Services.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> Campuscard | ../applications/Campuscard.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> housing | ../applications/housing.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> pci | ../applications/pci.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> TranslationService | ../applications/TranslationService.ear | default | 0 | 0.00 | 0 |

Home Applications Administration

3. Click **Edit**. The Application configuration page is displayed.

General

Status **Not Loaded**

Auto Start **true**

Parent Application **default**

Response - Servlets and JSPs

Active Sessions **0**

Active Requests **0**

Request Processing Time (seconds) **0.00**

Requests per Second **0.00**

Response - EJBs

Active EJB Methods **0**

Method Execution Time (seconds) **0.00**

Method Execution Rate (per second) **0.00**

Web Modules

| Name | Path | Active Requests | Request Processing Time (seconds) | Active Sessions |
|-----------------|---------------------|-----------------|-----------------------------------|-----------------|
| advancement_web | advancement_web.war | 0 | 0.00 | 0 |

EJB Modules

| Name | Path | Active EJB Methods | Method Execution Time (seconds) |
|----------------------|------|--------------------|---------------------------------|
| No EJB modules found | | | |

Administration

Properties

General

Advanced Properties

Resources

Data Sources

JMS Providers

Security

Security

- Select **Security** from the Administration section. The Security page is displayed.

Security Page Refreshed Ja

Principals

User Manager Name **JAZNUserManager**

User Manager Class **oracle.security.jazn.oc4j.JAZNUserManager**

Groups

Select Name

No groups found using the specified User Manager

Users

| Select Name | Group Memberships |
|---|-------------------|
| No users found using the specified User Manager | |

Security Roles

| Select Name | Assigned Users | Assigned Groups |
|-------------|----------------|-----------------|
| • bannerws | | |

- Click **Add Group** in the Groups section. The Security: Add Group page is displayed.

- Enter the following information:

Security: Add Group

Name

Description

TIP Remote EJB clients require the RMI login permission in order to be able to access objects on the OC4J server.

Grant the RMI Login Permission.

Grant the Administration Permission.

Name *jazn.com/advancement_ws_clients*

Description Security access group for the adapter

7. Click **OK**. The Security page is redisplayed with the new group.
8. Click **Add User** in the Users section. The Security: Add User page is displayed.
9. Enter the following information in the General section:

| General | |
|------------------|---------------------------------|
| Name | jazn.com/bannerws |
| Description | Banner Advancement Adapter User |
| Password | •••••••• |
| Confirm Password | •••••••• |

Name *jazn.com/bannerws*
(This is an example. Enter the name of your choice.)

Description *Banner Advancement Adapter User*

Password Password for the user being created.

Confirm Password Confirmation of the password for the user being created.

10. Assign this new user to the *jazn.com/advancement_ws_clients* group by checking the corresponding check box in the Group Memberships section.

| Group Memberships | |
|--|---------------------------------|
| Select All Select None | |
| Select Group Name | |
| <input checked="" type="checkbox"/> | jazn.com/advancement_ws_clients |

11. Click **OK**. The Security page is redisplayed with the new user.
12. Verify that the new user is assigned to the *jazn.com/ advancement_ws_clients* group.
13. Click **Map Role to Principals** in the Security Roles section. The Role page is displayed.

Map Role To Groups

Select All | Select None

Select Group Name

| | |
|-------------------------------------|---------------------------------|
| <input checked="" type="checkbox"/> | jazn.com/advancement_ws_clients |
|-------------------------------------|---------------------------------|

Map Role To Users

Select All | Select None

Select User Name

| | |
|--------------------------|-------------------|
| <input type="checkbox"/> | jazn.com/bannerws |
|--------------------------|-------------------|

Revert Apply

14. Select the `jazn.com/advancement_ws_clients` group name.
15. Click **Apply**. A confirmation page is displayed.
16. Click **OK**. The Role page is redisplayed.
17. Navigate to the Security page and verify that the group is mapped to the role.
18. Navigate to the Home page.
19. Click **Restart** to restart the server.

Step 6 Enable schema validation (optional)

Validating XML request and response messages for each Web service invocation degrades system performance. For this reason, schema validation is disabled by default. To enable schema validation, you must set system property `BANNERWS_SCHEMA_VALIDATION` with a value of `true` for the OC4J instance where the adapter is installed. Use the following steps to enable schema validation.

1. Navigate to the **Applications** tab for the instance where the adapter is installed.
2. Select the Banner Advancement Web Services Adapter.

OC4J: BWS

Home Applications Administration

Default Application Name default
Default Application Path application.xml

Deployed Applications Deploy EAR file Deploy WAR file

Edit Undeploy Redeploy

| Select | Name | Path | Parent Application | Active Requests | Request Processing Time (seconds) | Active EJB Methods |
|----------------------------------|----------------------|--|--------------------|-----------------|-----------------------------------|--------------------|
| <input checked="" type="radio"/> | Advancement_Services | ../applications/Advancement_Services.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> | Campuscard | ../applications/Campuscard.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> | housing | ../applications/housing.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> | pci | ../applications/pci.ear | default | 0 | 0.00 | 0 |
| <input type="radio"/> | TranslationService | ../applications/TranslationService.ear | default | 0 | 0.00 | 0 |

Home Applications Administration

3. Select the **Administration** tab. A list of options is displayed.

OC4J: BWS

Home Applications Administration

Instance Properties

- Server Properties**
- Website Properties
- JSP Container Properties
- Replication Properties
- Advanced Properties

Application Defaults

- Data Sources
- Security
- JMS Providers
- Global Web Module

Related Links

- ADF Business Components

OC4J Inheritance

OC4J applications have a hierarchical parent-child relationship to facilitate administration through inheritance. A child application inherits certain attributes from its parent application such as principals and JNDI objects including data sources, JMS providers and EJBs. When an OC4J application is deployed, you specify the parent application. The Default Application is the top of the parent hierarchy.

Home Applications Administration

4. Select **Server Properties** from the Instance Properties section. The Server Properties page is displayed.
5. Scroll to the Command Line Options section. The corresponding value for **Java Options** might look similar to this:

```
-Xrs -server -Djava.security.policy=$ORACLE_HOME/j2ee/  
home/config/java2.policy -Djava.awt.headless=true
```

6. Add the following system property at the end of **Java Options**:

```
-DBANNERWS_SCHEMA_VALIDATION=true
```

Command Line Options

| | |
|-----------------|---|
| Java Executable | |
| OC4J Options | -properties |
| Java Options | -va.awt.headless=true -DBANNERWS_SCHEMA_VALIDATION=true |

[Related Links](#) [Tracing Properties](#)

7. Click **Apply**. A message confirms the action and asks if you want to restart the server.
8. Click **Yes** to restart the server.

Step 7 Configure logging

The Banner Advancement Web Services Adapter uses Apache's log4j to log the activities performed by the application at runtime. Log4j uses a properties file to establish specific runtime options. The following options should be reviewed and modified as appropriate:

- Location of the log files - The default location is <IAS_HOME>/j2ee/home/log. This location should be changed to the OC4J instance where the Banner Advancement Web Services Adapter is installed.
- Logging level - The default level is *INFO*, resulting in limited information (*INFO*, *WARNING*, and *ERROR* level statements) being stored in log files. To provide detailed logging for initial operation, you should change the logging level to *DEBUG*.

Note

To prevent the log file from getting too big, you should change the logging level back to *INFO* after initial operation is completed. ■

Use the following steps to modify the logging options as appropriate:

1. Navigate to <IAS_HOME>/j2ee/<OC4J instance>/applications/<Web services adapter name>/advancement_web/WEB-INF/classes.
2. Edit log4j.properties as follows:

| Property | Original Value | New Value |
|-------------------------------|----------------------------|---|
| log4j.appender.out.File | log/ advancement_ws.log | ../<OC4J instance>/ log/ advancement_ws.log |
| log4j.category.com.sungardsct | INFO | DEBUG |

3. Restart the OC4J instance for the changes to take effect.

Step 8 Update the WSDL endpoint URL (optional)

SOAP-based Web services are described via a WSDL (Web Services Definition Language) document. WSDL is an XML format for describing network services as endpoints. A WSDL document contains information about the messages that can be exchanged with the Web service, the network protocol supported, and the location of the service expressed as a URL.

The WSDL for the Banner Advancement Web Services Adapter (`banner_giving_history_service.wsdl`) is provided in patch `p1-5j4fyo_alu80100`. This file can be used to generate client-side code for accessing the Web service. Third-party testing tools, such as soapUI (www.soapui.org), can also use this file to generate XML-based test cases and test suites for testing the Web service.

If you plan to use the WSDL document for development or testing, you must update the document to reflect the actual location of the Web service (that is, the service endpoint). Once updated, the file can be transferred to a local computer or shared network location to be used, as needed, by development and test tools that require the WSDL document. The WSDL document should *not* be deployed to the application server.

Use the following steps to configure the WSDL document so it references the proper endpoint URL.

1. Locate the `banner_giving_history_service.wsdl` file provided in the `p1-5j4fyo_alu80100` patch download.
2. Open the file with a text editor or XML editor.
3. Locate the `<service>` tag in the WSDL document. The `<soap:address location>` attribute under this tag contains the following tokenized dummy URL string:

```
http://@@@hostname@@@:@@@port@@@/advancement/v8_2
```

4. Modify the dummy URL to reflect the correct host name and port for your installation.
5. If you changed the default URL string (`/advancement`) during installation, make the corresponding change in the dummy URL.
6. Save the file.

Example

Updating the dummy URL to reflect a deployment to a server named `OAS_Test` that listens on port 443 would have the following result:

```
http://oas_test.somewhere.edu:443/advancement/v8_2
```

If the default URL string was changed from `/advancement` to `/advancement/bws_test`, the URL would also need to be updated as follows:

```
http://oas_test.somewhere.edu:443/advancement/bws_test/v8_2
```

With these modifications in place, testing and development tools can use the WSDL document to generate code for accessing the specific URL.

Step 9 Test the Banner Advancement Web services (optional)

Once the adapter is installed and the appropriate WSDL file is modified, you should test the exposed Banner Advancement Web service. This step is optional, but highly recommended. The degree of testing depends on the data and tools that your institution uses. For this reason, this document provides testing guidelines rather than specific steps.

Ellucian highly recommends the use of an interactive testing tool that provides visibility into Web service request and response messages. Such a tool helps functional and technical users understand the options that are available for a Banner Web service and their effect on the content of response messages. You can create schema-compliant XML messages for your specific environment, quickly inspect the results, modify Web service settings in Banner (if necessary), and execute the same request to see the resulting differences.

Several commercial and open source Web service testing tools provide these capabilities. One open source tool is eviware soapUI (www.soapui.org). This tool uses a WSDL document and associated XML schema as input to generate compliant request message stubs to which data can be added for calling the associated service. Response messages are displayed in an adjacent window for easy visibility. The tool also allows individual service requests to be tied together to form larger test suites.

Manually creating Web service requests and visually inspecting responses provide practical insight into the data returned by a Web service under specific conditions. In addition, issues with the underlying Banner configuration for a Web service are noticed more readily.

Refer to [“Test cases” on page 28](#) for sample test cases for the GetGivingHistory Web service.



3 GetGivingHistory Web Service

The GetGivingHistory Web service allows external systems to request giving history information for a donor who is set up in Banner®. The Web service returns summary giving information and a list of individual gifts for the donor. The results are filtered as follows:

- The Outstanding Pledges total includes those pledges that are allowed to be displayed on the Web. The **Web Indicator** defined for each pledge type on the Pledge Type Validation Form (ATVPLDG) determines which pledges are allowed to be displayed on the Web.
- Details are displayed for those gifts that are allowed to be displayed on the Web. The **Web Indicator** defined for each gift type on the Gift/Payment Type Validation Form (ATVGIFT) determines which gifts are allowed to be displayed on the Web.

This Web service supports integration with iModules Encompass, but can be used for other purposes.

Message exchange

The GetGivingHistory Web service is a request/reply Web service that exchanges the following messages using the SOAP protocol over HTTP:

- GivingHistoryRequest
- GivingHistoryResponse

GivingHistoryRequest

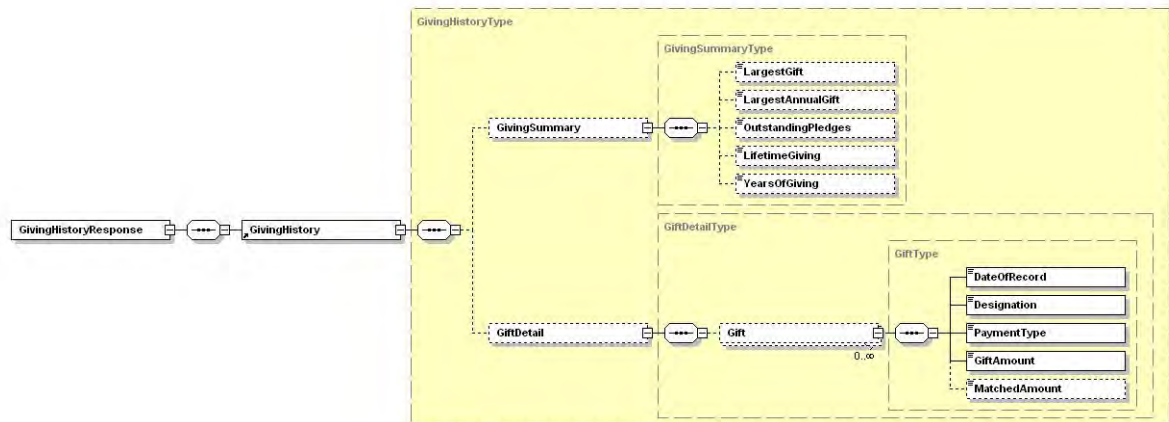
An external system uses the GivingHistoryRequest message to request giving history information from Banner. The only input parameter is the `ConstituentID`.

The following diagram shows the structure of the GivingHistoryRequest message schema.



GivingHistoryResponse

If the `ConstituentID` is found, the `GetGivingHistory` Web service returns giving history in the `GivingHistoryResponse` message. This message contains both summary and detailed information. The following diagram shows the structure of the `GivingHistoryResponse` message schema.



If the `ConstituentID` is found but no giving history exists, the Web service returns an empty `GivingHistory` element.

SOAP fault messages

If a valid response cannot be created as a `GivingHistoryResponse` message, a SOAP fault message is returned. The following situations might cause a SOAP fault message:

- The `ConstituentID` provided in the `GivingHistoryRequest` message is not a valid Banner PIDM (`SPRIDEN_PIDM`).
- A `ConstituentID` is not supplied in the `GivingHistoryRequest` message.
- A network, database, or other technical issue occurs.

Message mapping to Banner

The following tables provide a mapping between the message elements/attributes and Banner columns. The left vertical lines represent the nesting of the attributes inside the elements. Elements can nest inside other elements as well.

GivingHistoryRequest

| Element/Attribute | Database Mapping |
|----------------------|------------------|
| GivingHistoryRequest | |
| ConstituentID | SPRIDEN_PIDM |

GivingHistoryResponse

| Element/Attribute | Database Mapping |
|-----------------------|-------------------------------------|
| GivingHistoryResponse | |
| GivingHistory | |
| GivingSummary | |
| LargestGift | APBGHIS_HIGH_GIFT_AMT |
| LargestAnnualGift | Calculated |
| OutstandingPledges | Calculated |
| LifetimeGiving | Calculated |
| YearsOfGiving | Calculated |
| GiftDetail | |
| Gift | |
| DateOfRecord | AGBGIFT_GIFT_DATE |
| Designation | AGRGDES_DESG |
| PaymentType | ATVGIFT_DESC (AGBGIFT_GIFT_CODE) |
| GiftAmount | AGRGDES_AMT |
| MatchedAmount | SUM(AGRMDES_AMT) |

Intended usage

Partner applications such as iModules Encompass use the GetGivingHistory Web service to display giving history to donors in a portal-based environment.

Setup requirements

The following setups determine which pledge and gift information is displayed to a donor:

- The **Web Indicator** flag defined for each pledge type on the Pledge Type Validation Form (ATVPLDG) determines which pledges are included in the Outstanding Pledges total that is displayed to a donor. If the indicator is selected for a type code, the Web service includes pledges with that type code in the total. If the indicator is not selected for a type code, the Web service excludes pledges with that type code from the total.
- The **Web Indicator** flag defined for each gift type on the Gift/Payment Type Validation Form (ATVGIFT) determines which gift details are displayed to a donor. If the indicator is selected for a type code, the Web service displays details for gifts with that type code. If the indicator is not selected for a type code, the Web service hides details for gifts with that type code.

Test cases

After the Banner Advancement Web Services Adapter is installed, you should test the exposed GetGivingHistory Web service. Testing is optional but highly recommended. The degree of testing depends on the data and tools that your institution uses.

The GetGivingHistory Web service has one input parameter, the `SPRIDEN_PIDM` of the advancement individual or advancement organization. In the Web service request, the XML tag is titled `ConstituentID`.

The following tests can help you understand the Web service. These tests are examples only. They do not reflect actual data in your database.

Advancement individual with giving history

Test GetGivingHistory using the `SPRIDEN_PIDM` of an advancement individual with known giving history.

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:advancement:messages">
  <soapenv:Header/>
  <soapenv:Body><urn:GivingHistoryRequest>
    <urn:ConstituentID>301</urn:ConstituentID>
  </urn:GivingHistoryRequest>
</soapenv:Body>
</soapenv:Envelope>

```

The response message should look similar to the following:

```

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/
soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <SOAP-ENV:Body>
    <GivingHistoryResponse
xmlns="urn:sungardhe:enterprise:advancement:messages">
      <GivingHistory>
        <GivingSummary>
          <LargestGift>0</LargestGift>
          <LargestAnnualGift>0</LargestAnnualGift>
          <OutstandingPledges>937.5</OutstandingPledges>
          <LifetimeGiving>1250</LifetimeGiving>
          <YearsOfGiving>1</YearsOfGiving>
        </GivingSummary>
        <GiftDetail>
          <Gift>
            <DateOfRecord>2005-10-10</DateOfRecord>
            <Designation>Allen Family Scholarship</
Designation>
            <PaymentType>Check</PaymentType>
            <GiftAmount>312.5</GiftAmount>
            <MatchedAmount>0</MatchedAmount>
          </Gift>
          <Gift>
            <DateOfRecord>2005-10-10</DateOfRecord>
            <Designation>University Campaign Fund</
Designation>
            <PaymentType>Check</PaymentType>
            <GiftAmount>312.5</GiftAmount>
            <MatchedAmount>0</MatchedAmount>
          </Gift>

```

```

        </GiftDetail>
    </GivingHistory>
</GivingHistoryResponse>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Advancement individual without giving history

Test `GetGivingHistory` using the `SPRIDEN_PIDM` of an advancement individual without giving history.

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:advancement:messages">
    <soapenv:Header/>
    <soapenv:Body><urn:GivingHistoryRequest>
        <urn:ConstituentID>50005</urn:ConstituentID>
    </urn:GivingHistoryRequest>
    </soapenv:Body>
</soapenv:Envelope>

```

The response message should look similar to the following:

```

<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/
soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
    <SOAP-ENV:Body>
        <GivingHistoryResponse
xmlns="urn:sungardhe:enterprise:advancement:messages">
            <GivingHistory/>
        </GivingHistoryResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>

```

Invalid advancement individual ID

Test `GetGivingHistory` using an invalid `SPRIDEN_PIDM`.

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/
soap/envelope/"
xmlns:urn="urn:sungardhe:enterprise:advancement:messages">
    <soapenv:Header/>
    <soapenv:Body><urn:GivingHistoryRequest>

```

```
        <urn:ConstituentID>303</urn:ConstituentID>
    </urn:GivingHistoryRequest>
</soapenv:Body>
</soapenv:Envelope>
```

The response message should look similar to the following:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/
soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <SOAP-ENV:Body>
    <SOAP-ENV:Fault>
      <faultcode>Client</faultcode>
      <faultstring>*ERROR* Invalid Constituent ID.
ORA-06512</faultstring>
      <faultactor>/advancement/v8_2</faultactor>
    </SOAP-ENV:Fault>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

