Extend EBS Using Applications Express

John Peters
JRPJR, Inc.

Abstract

Few people know about Oracle Applications Express (APEX) an actual free Oracle Tool included with your Oracle DB Licenses. How many times have you heard of an easy to use and free Oracle product? Attend this session and see how I have used APEX to extend Oracle EBS with fast development reporting applications and easier to use data entry screens. I will cover how easy it is to integrate APEX into an EBS env, the process for creating reports and how to build easy to use data entry screens that call Oracle API's.

Background

Oracle Applications Express (APEX) was first named Oracle HTML DB when it was introduced back in 2004. It is currently up to version 4.2.4 which was release 12/2013 (a version 5 release is currently in Beta release). This is part of the Oracle DB and best of all it is a Free tool, it included in your Oracle DB License. How many times do you get Free tools from Oracle!

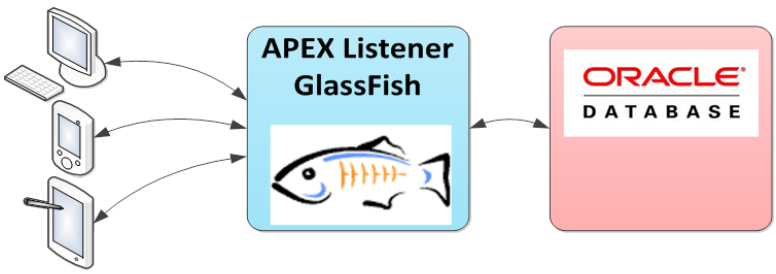
The use of APEX is accepted and encouraged by the Oracle E-Business Suite (EBS) Applications Technology Group (ATG) as a lightweight development tool for developing extensions in:
- https://blogs.oracle.com/stevenChan/entry/new_whitepaper_upgrading_your_customizations
- Extending Oracle E-Business Suite Release 12 using Oracle Application Express (Doc ID 1306563.1)
- How to Integrate APEX with Oracle e-Business Suite and Set Up Authentication (Doc ID 373604.1)

EBS Users should consider this tool for uses like:
- Interactive Reporting (users can alter the filter condition and columns displayed)
- Simple data entry and update screens

If you find yourself using a tool like Access, SQL*Server, or even Excel Spreadsheet Datasets you should consider APEX.

Architecture

APEX is installed and runs in the Database. The development is performed in a browser. The runtime user interface is displayed in a browser. The communication between the users browser and the database is accomplished using a Web Listener, typically the APEX Listener with Glassfish.



This browser centric user interface (UI) allows APEX to be accessed from any device that supports a browser. This allows an APEX application to be displayed on a PC, Tablet or Smart Phone. APEX automatically identifies the device and then switches to UI conventions that a common on that device. As an example: - iOS has certain conventions for date and time List of Value's

- PC's have a different convention for data and time List of Value's
These subtle differences in UI experience do not require any additional programming/development effort. This flexibility allows an application that is written for use in an office environment, to be shifted to a factory floor or field use without modification.

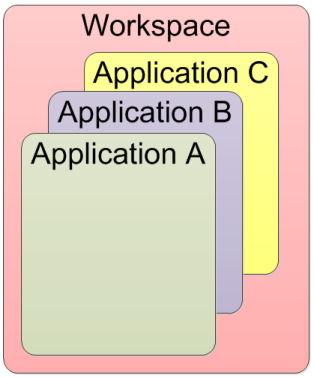
APEX requires a constant connection to the database. Just like any other browser based application, the browser must be connected to some web server to provide the data. For this reason mobile APEX applications must have a Network or VPN connection from the device back to the web listener and database. With the proliferation of cellphone networks this requirement for mobile access becomes a minimal issue.

Development

Development is done in an APEX Workspace. The lowest level of access that Developers are granted to in APEX is a Workspace. So if you have different development teams you want to segregate you should create different Workspaces.

The Workspace defines the database user connection that APEX will use to access the database. You must grant access to the database objects you want APEX to utilize to this database user. So if you have different database access restrictions you might want a different Workspace (example normal EBS reporting might be different from HR reporting).

Within a Workspace you will have one or more Applications. An Application is used to hold many common pages together. When you migrate your APEX development from a Dev instance to Test or Prod you do this at the Application Level. So it is important to ensure you have enough Application granularity that can develop and test in migrate able units. On the flip side too many Applications require users to have different login screens and home pages they have to navigate through. While SSO eases this it does not completely eliminate it.



APEX development lends itself to Rapid Application Development paradigm. It is very easy to use the prebuild Templates and Wizards to stand up an application quickly and get user feedback. You can then quickly implement changes and get it back to user to review and test.

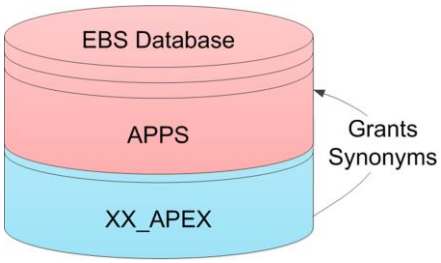
APEX includes an SQL Workshop tool that allows you to explore database objects and build queries. There are also hooks in SQL*Developer to export queries directly to APEX.

APEX also has many controls that were set to default values by the templates that allow you to further enhance the user experience. In addition, since it is browser based, you jump out to tools like Javascript to enhance the UI components. Keep in mind that as you use these extra tools and features you might limit the browsers (and platforms) that support them. You should try to keep things as simple as possible.

There are many Pre-Built Packaged Applications available that you can download and explore. These not only provide you a good starting point for your application development, but they also show how others have accomplished certain functionality. There are also many good tutorials and reference information on the Oracle APEX site:
http://apex.oracle.com

Normally you will install APEX in your EBS Instance. But if you wanting to explore APEX you can signup for a free hosted APEX Workspace at:
http://apex.oracle.com/pls/apex/f?p=4700:1:0

Typically you will grant privileges to your APEX DB User associated with your Workspace. In an EBS environment this should not be the APPS user, but a new database user, possibly XX_APEX that has access to only the object you need.



Example APEX Applications

The accompanying presentation for this paper has three example applications. The first is a pure reporting environment linked to an EBS Instance. This provides some examples of a Framework (which will be discussed later), Report Organization, Report Drill Down, Report Filtering and Report Download to Excel.

The second example is a reporting environment that also allows some limited update of EBS data. This is accomplished using the TCA API's to add and update Customer Contact records. In addition, the TCA API's were used to enable Customer Addresses in different Operating Units. This example also shows how detailed the reporting environment can get providing a full 360 degree view of the customer, including: Customer Sites, Contacts, Sales Orders, Invoices, Service Contracts, etc. This client also took the output of the certain BI Publisher documents, like Invoices, and uploaded those into the DB as a BLOB. This allowed APEX to put up a link on the report to display the AR Invoice and once clicked it opens up the actual PDF document that sent to the customer.

The third example is a simplified data entry application. This was an extension to Order Management to support an industrial equipment rental business. In this example APEX was used instead of Forms or OA Framework pages to provide a simple uncluttered screen to the users. Also in this environment the initial users were office personnel, with this eventually being transferred out to yard personnel working on hardened tablets and eventually to field sales users. Show how you can write the application once and then choose the deployment later.

If you have questions about these example applications please send me an email at john.peters@jrpjr.com.

Application Framework

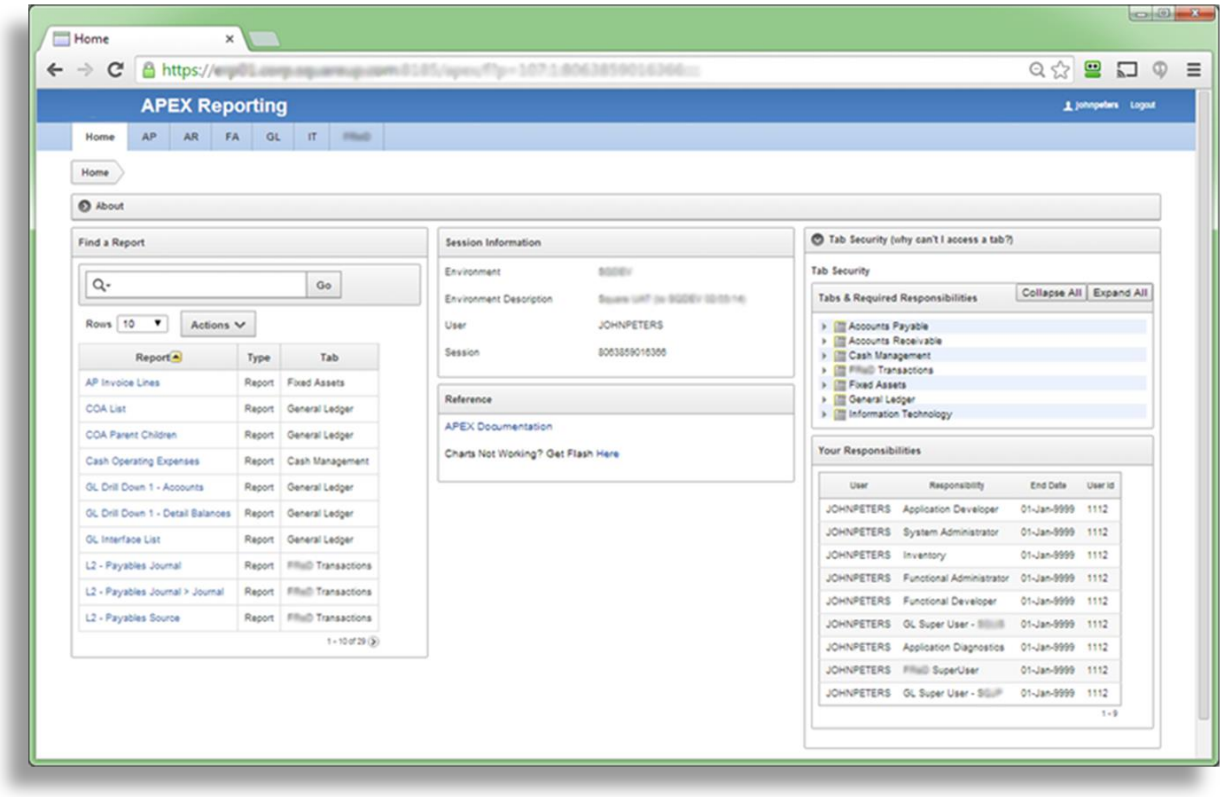
APEX is just a development tool. You have to develop an Application Framework that provides for:
Authentication, Verifying who a user is when they login to your APEX Application
Authorization, Determining what that specific user is able to see and do in your APEX Application

On the Authentication front we can use EBS Authentication to ensure the person is who they say they are. This uses the standard EBS Userid/Passwords. You can review two methods for performing Authentication with EBS environments in:
- How to Integrate APEX with Oracle eBusiness Suite and Set Up Authentication (Doc ID 373604.1)
- Extending Oracle E-Business Suite Release 12 using Oracle Application Express (APEX) (Doc ID 1306563.1)
The second paper is the preferred method to doing EBS Authentication.

You can also integrate with EBS supported SSO mechanisms. There are many papers on this topic but you should start with this one reference page:
- Master Note for Oracle Application Express (APEX) Authentication (Doc ID 1094413.1)

For Authorization it makes sense to leverage EBS User Responsibilities to determine what access they should be granted to APEX pages.

Both Authentication and Authorization require development to implement. The screen shot below is an example of a client's Framework in an APEX EBS Reporting Environment.



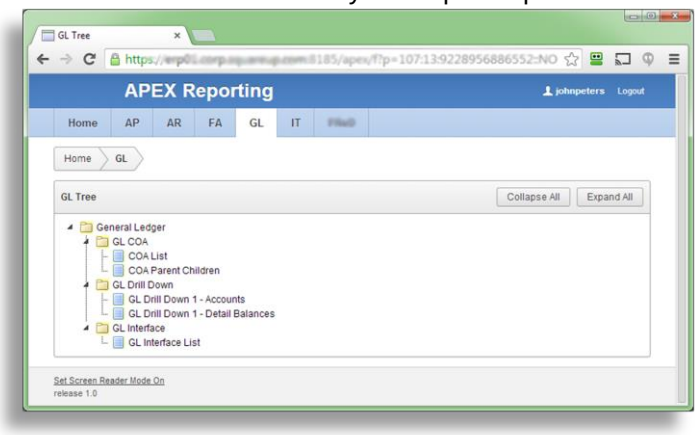
In the picture above these regions perform the following functions:

Top Light Blue Bar - Has tabs by reporting functional area. This allows us to link EBS Responsibilities to a reporting functional area. We can then link specific APEX pages to those reporting functional areas.

Find a Report region - This allows the user to quickly find a report rather than having to navigate all of the tabs searching for it. This is handy as your report inventory grows.

Tab Security - This shows the users what reporting functional areas are accessible with which responsibilities and which responsibilities the user has. This answers that age old question of 'Why can't I access a Tab?'.

So as the user selects tabs they see reports specific to that reporting functional area.



Again this APEX Framework must be developed. There is nothing out of the box to handle this directly, but there are many examples in the sample applications that you can use as a starting point.

Installation

You should install APEX into your EBS Database. This is very important because query across a database link can be sub-optimal. Also being in a different DB complicates development and the use of the SQL Workbench within APEX.

You will need to install a Web Listener, and I have always used the APEX Listener with Glassfish.

You can follow the install steps in the Oracle document:
Oracle® Application Express, Installation Guide, Release 4.2, E35123-05

While doing the initial install I would leave the SSO Integration for a later step.

You can also integrate with EBS Menus. You must set the profile option 'FND: APEX URL' to point to the APEX Listener. Then for each APEX page you can create an EBS Form Function of Type JSP to call the APEX page. These Form Functions can be linked to an EBS Menu and Responsibility. For additional details take a look at:
Extending Oracle E-Business Suite Release 12 using Oracle Application Express (APEX) (Doc ID 1306563.1)

Development Steps

There are copious tutorials and example for APEX development on Oracle's Web Site:
- http://apex.oracle.com
- Oracle® Database, 2 Day + Application Express Developer's Guide, Release 4.2, E35122-05
- Each APEX install has sample pre-packaged applications so you can examine "How they did that?".
- Extending Oracle E-Business Suite Release 12 using Oracle Application Express (APEX) (Doc ID 1306563.1) has a great step by step example
Based on this I will only cover the high level steps in this paper.

1) Determine the general Page Flow
Since APEX is essentially generating Web pages, albeit some are interactive reports, or data entry screens, they are still just Web pages. You need to determine the overall flow of your application from login page, through the framework, to the individual reports, some with drill downs, all the way to the logout. It is important to do this so your reporting framework has a logic flow and does not seem to have evolved in a haphazard structure.

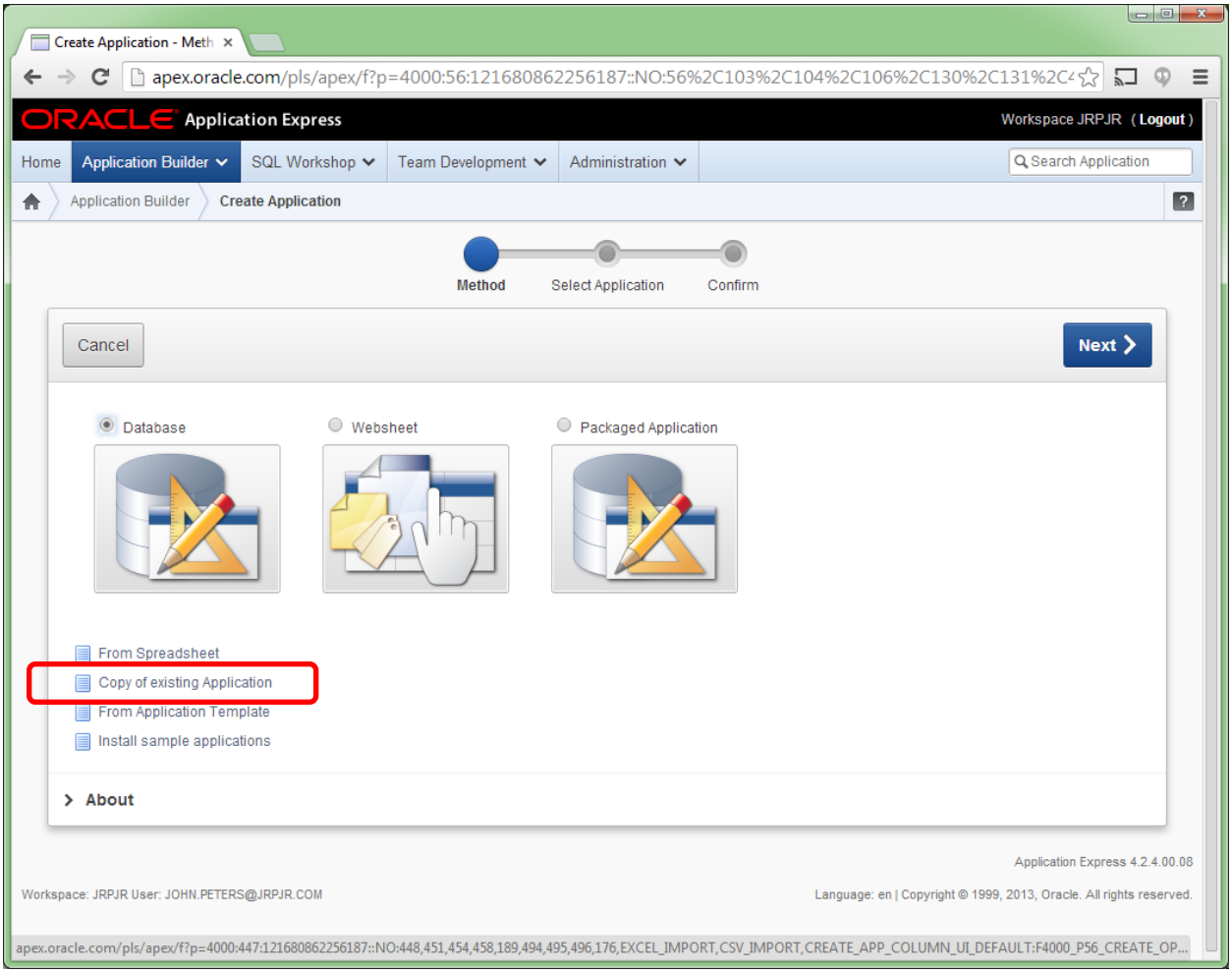
2) Create Views in APPS, Grant Select, Create Synonyms
Join the various EBS tables together in Views in the APPS schema. Then all you need to grant to XX_APEX is select on the View along with a Synonym for the view. This keeps you XX_APEX schema cleaner and easier to work with. It also allows you to develop and debug your EBS joins in the APPS schema (in a non-PROD instance of course) which is easier.

If your development includes updating EBS data, you have two options:
- Create a custom interface table you insert data into and process from in PL/SQL
- Call an EBS API but in this case you will need to grant execute on the PL/SQL code

3) Create your APEX Pages
In the APEX Application Builder you can use the provided Templates and Wizards to step you through the creation of the APEX Page. This provides the basic structure of the APEX page. You will fine tune it in the next steps.

4) Optionally edit the APEX Page
APEX provides a great deal of control on the way the finished page behaves. You can tweak these settings and then immediately test if you are getting the desired results.

One other development tip is that you can Copy an existing Application to a new Application. This is very handy if you are unsure of the change you are about to make and you don't want to totally trash your existing application.



Migration

Once you have completed the development of your Application you can easily migrate the entire Application between instances. APEX provides an Export function that extracts all of the APEX MetaData and any supporting components to an SQL script from the Source Instance. On the Target Instance log into the APEX Application Builder and Import the SQL file. This will load all of the APEX MetaData along with any supporting components into the Target Instance. You should resist the temptation to change things in the Target instance, and instead always go back to the original Source instance to make your changes to prevent getting branched development and possibly loosing development work.

Do not attempt to run the SQL script in a tool like TOAD, SQL*Developer or SQL*Plus. The APEX Application Builder prepares an SQL session in a manner that is required for the script to run properly. There are ways to run the SQL from SQL*Plus but that is beyond the scope of this paper and I would not recommend it.

While Importing your application there are two important Steps/Options:
1) I would typically delete the existing application on the Target Instance prior to importing, just to ensure that everything is loaded properly.
2) During the Import there is an option to either:
- Auto Assign New Application ID
- Reuse Application ID from Export File
- Change Application ID

You should typically choose the Reuse option so that your underlying Application ID remains the same. This is especially important if you have integrated APEX into EBS Form Functions, which will contain the Application ID.

Conclusion

Hopefully this paper has provided a cursory insight into the user of APEX to extend EBS.