

7 Best Practices – APEX Side

7.1 Create a new application via the wizard

To create a new application, use the wizard as follows:

- Go to Application Builder, then click the Create button.
- Choose your type of application and click "Next"
- Enter the "application name", click "Create application"

Note: for the application name, use by default the same as the project name.

7.2 Create a global page for each application

The global page will carry the breadcrumb region or any other region/item that we want to appear on multiple pages.

7.3 Enable feedback for every application

Allow your users to send feedback about your application from within the application. This way they can easily forward you any issues found in the application.

All feedback entries are stored in the *Team Development* section of the APEX environment.

Team Development is a standard APEX functionality that enables team members to share information about features and bugs. This section is reserved for users in APEX with a specific *Developer* or *Team Development* role.

7.4 Give an alias to every application

Within APEX, your application is referred to by a unique ID. However, best is to give your application a specific and meaningful alias. This alias can as well be used within the URL of your browser.

Example: `http://hostname/apex/f?p=203` can be replaced by `http://hostname/apex/f?p=portal`, which is more meaningful and easier to remember.

7.5 Enable Session State Protection

Session State Protection is a security setting that can be used to prevent what is called "URL tampering".

URL tampering is the abuse of a malicious user that modifies URL parameters to access data that they are not entitled to access.

Example: you have a Master Detail setup for the EMP table (i.e a report with edit links to a form) each link will look something similar to:

http://hostname/apex/f?p=211:11:14916146169058:::P11_EMPNO:7698.

If you have restricted the report to show only employees of your current department then you can easily change the EMPNO id and change it to another employee of another department. Even though the intention of the application was to “restrict” users to only edit people within their own department. A malicious or curious user could start editing all the employees.

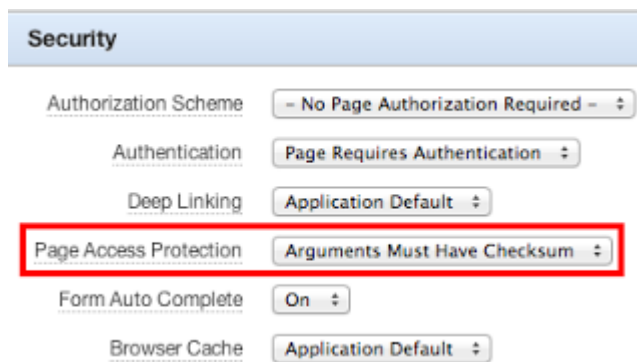
SSP prevents URL tampering by applying a checksum to the end of the URL. The new URL looks than similar to:

http://hostname/apex/f?p=211:11:14916146169058:::P11_EMPNO:7698&cs=3358AAB32787C2A2B294CE934D50FD0C3.

When a user tries to modify the EMPNO id, an error will be generated by APEX because the URL won't have the correct checksum.

Applying SSP

To add SSP at the page level (technically called Page Access Protection at the page level) edit the page and scroll down to the Security section. Select Arguments Must Have Checksum to set the Page Access Protection option.



The screenshot shows the 'Security' section of the APEX page editor. It contains several settings with dropdown menus:

- Authorization Scheme: - No Page Authorization Required -
- Authentication: Page Requires Authentication
- Deep Linking: Application Default
- Page Access Protection: Arguments Must Have Checksum** (This row is highlighted with a red rectangle)
- Form Auto Complete: On
- Browser Cache: Application Default

Once that is added you can modify the page items you want SSP to be applied to. Edit a page item and scroll down to the Security section. Select an option for Session State Protection (note: click on the help link to find the differences between the various options; we usually use Checksum Required - Session Level).

Security

Authorization Scheme: - No Authorization Required -

Session State Protection: Checksum Required - Session Level

Store value encrypted in session state: No

Restricted Characters: - All Characters Allowed -

Tip: To modify a page and all its page items at the same time go to Shared Components > Session State Protection > Page Item and click on a page link.

Set Page and Item Protection

Application: 211 - Securing APEX - Demo

Session State Protection: Enabled

Page: 11

Name: Edit Emp

Page Access Protection: Arguments Must Have Checksum

Display Item Type: ☒ Data Entry Items ☐ Display-Only Items

Item Name	Region	Sequence	Label	Item Session State Protection
P11_EMPNO	Edit Emp	10	Empno	Checksum Required - Session Level
P11_ENAME	Edit Emp	20	Ename	Unrestricted
P11_JOB	Edit Emp	30	Job	Unrestricted
P11_MGR	Edit Emp	40	Mgr	Unrestricted
P11_HIREDATE	Edit Emp	50	Hiredate	Unrestricted
P11_SAL	Edit Emp	60	Sal	Unrestricted
P11_COMM	Edit Emp	70	Comm	Unrestricted
P11_DEPTNO	Edit Emp	80	Deptno	Unrestricted

7.6 Choice of APEX Theme

Theme 42 – the universal theme - is a responsive theme that is by standard included into Oracle APEX. A responsive theme allows for automatic adaptation depending on the screen size and as a result it can be used on a variety of devices (tablet computer, smartphones,...).

7.7 Create a LOV for every table

For every table holding data of a specific type, create a List of Values (LOV). This will not only ease the life of the users, but as well assure no invalid values can be entered by the user.

7.8 Naming conventions for Applications Items

In Shared Components, you find a section Application Items. These are items that can be accessed cross page in the entire application. Typically we create an application item for the user_id, customer_id, role_id or any other high level item.

The naming conventions for Application Items is AI_<item name>

7.9 Run the Advisor once a day

At least once a day the APEX Advisor (found in the *Utilities* section or on page level) should be ran to check if the Application doesn't have any errors or warnings based on best practices of the APEX development team wrote.

7.10 Run the APEX R&D Advisor once a day

Additional to the Advisor that comes with APEX, some checks specific for your own project are made available by APEX R&D to validate Example: naming conventions, components set to condition never etc.

A complete list of those checks can be found in the APEX Developer Portal (a paid for product by APEX R&D).

7.11 Usage of comments

Use Page comments to specify what's happening on the page or why specific (technical) options are used.

Use Region, Item, Process etc. comments to indicate what's happening in case things are not self explanatory.

7.12 Working in teams

The APEX Development Environment enables working in teams. Therefore follow some guidelines when working with more than one developer on the same application.

Lock the page and enter a comment how long you believe you will lock the page and what you are doing.

Share your knowledge by using Shared Components and the Publish/Subscribe method.

7.13 Use of bind variables

In Computations, Processes, PL/SQL Code, SQL in reports etc. always use bind variables. Never use a substitution string.

BAD (substitution string):

```
select empno
from emp
where ename like '%&P1_SEARCH.%)'
```

GOOD (bind variable):

```
select empno
from emp
where ename like '%' || :P1_SEARCH || '%'
```

7.14 Use of Page Items

You can reference page and application items in different ways:

- :APP_USER
 - SQL & PL/SQL Region within APEX
- &APP_USER.
 - HTML Regions, Templates, Before and After Header Region, ...
- v('APP_USER')
 - PL/SQL Packages, procedures and functions

7.15 Use of Page Groups

To categorize pages, you can use Page Groups, it makes it easy to identify pages.

7.16 Coding Practices

Following coding practices are tried to be followed as much as possible:

- All code in packages
- Instrument heavily
- Add comments where necessary
- Procedure in PKG shouldn't take more than one page
- Reusable code (if possible)
- Javascript? Look for Dynamic Action and/or Plugin
- Use of external files on the webserver
- Make use of ALL the features in APEX and stay as close as possible to the existing features, with every new release more useful features are in, so reviewing code after the introduction of a new version is highly recommended.
- If APEX needs to be extend, make use of plugins or use the main building blocks APEX was build with (JQuery, PL/SQL, HTML5, CSS3, ...)

Following specific features (not an exhaustive list) can help to build consistent applications:

- Team Development
- Application and Page Groups
- Shared Components
- Publish and Subscribe to Master App

- Multiple Apps with same Authentication
- Application Properties
- Substitutions - BASE_DIR
- Version
- Lock Pages
- Comments on items
- Instance (Public) / Workspace Theme
- User Interface Defaults
- Plugins
- Shortcuts
- Advisor

7.17 One APEX workspace vs multiple workspaces

When creating multiple APEX applications you can either add them into one APEX workspace or multiple APEX workspaces.

Workspaces are logical containers within APEX that provide functional security. Workspaces can be associated with one or more database schemas. Each schema associated with an APEX workspace can be used to parse SQL and PL/SQL requests.

One workspace:

- Advantages
 - Easy to (publish &) subscribe to Framework applications
 - Only one workspace to login too to see all APEX applications
- Disadvantages
 - All developers see all APEX applications, in case an external developer comes in, no clear way to hide APEX applications in the same workspace
 - Potential to query the wrong table as multiple Oracle schema's are linked to the same workspace. On the application level the primary parsing schema can be set too, which should solve at least 90% of the cases.

Multiple workspaces:

- Advantages
 - Every workspace is linked to one Oracle database schema = Parsing schema
 - Best security as developers will only see the applications of the workspace they have access to
- Disadvantages
 - Harder to publish and subscribe as you would need to copy the master application in every workspace. When an update is done in the master application this has to be imported again into all the different workspaces

7.18 Build for performance and scale

It's important that the developer tests his code on performance and makes use of best practices in this area. For example the use of bind variables is important, working in sets in PL/SQL is another aspect, using analytical functions instead of building your own PL/SQL routine etc.

Using external files on the webserver will also generate less database hits. Things can get cached and compressed etc.

Typically we try to build pages that take less than 0.5 seconds (CPU time). Debug information gives information about timings, other tools like YSlow can help to track performance as well.

We can build application performance reports into the application so an administrator/developer can easily see how the performance of the app is going. We also try to be pro-active and recognize trends early on.

7.19 Build to be secure

A custom authentication will be used that allows Single Sign On in the office and custom login (user table) on a ship or when the Single Sign On is not accessible.

Authorization schemes will be setup so a person see only what he is allowed to see.

We will look at the current Oracle database roles and transform them into meaningful authorization schemes. Those authorization schemes will be linked to different components in APEX.

It's also important that some best practices are used in an APEX application so no SQL injection, URL tampering or Cross site scripting is introduced.

Use of Session State Protection (found in Shared Components) needs to be used, as well as the use of bind variables wherever possible. Extra attention on having the right authorization schemes and the use of validations is required too.

7.20 Build to be reusable

As much as possible reusable components will be used. For example PL/SQL packages to do the error handling (wrapper around existing APEX API) and Plugins to extend APEX.

7.21 Build to understand

Be able to turn on logging and debugging in the application at every moment in time is very important to see what is going on behind the scenes. There are different ways to do this (see presentation).

7.22 Consistent and user-friendly Lay-out

A theme will be created which will be used by every developer. The look and feel will be consistent within the application and cross applications. By using the playground different look and feels will be tested and once one is picked it will be reused wherever possible. We work together with a professional designer to get a clean and good looking UI. This design will be translated into an APEX theme, preferable using the latest guidelines and insights in web-design. At the moment responsive web-design and grid layouts is the way forward, so the application fits nicely on different screen sizes (for example on a tablet (iPad)). Also scalable graphics (vectors/fonts) are preferred that scale in size without noticing the pixels.

Custom themes are hard to start with, that's why you could start from a theme on the internet e.g. wrapbootstrap.com

7.23 Extending the developer practices

During every release a code review will be done. All the items that pop-up will be included in the document so other developers can learn from it as well.

Training can be scheduled whenever needed.

7.24 Useful Plugins

Select2, Dropzone

7.25 Template/Starter application

Oracle APEX 5.2 comes with a Blueprint option when you create a new application. It is great to start a new application as it will include many features e.g. global page, administrator pages, performance pages etc.

7.26 Mobile vs Responsive/Adaptive

Although you can create native mobile applications in APEX, which is using JQuery Mobile behind the scenes, we recommend using Universal theme for your application. It's responsive and you only have one code base. So far JQuery Mobile has not been updated in APEX, but it might change in the future.