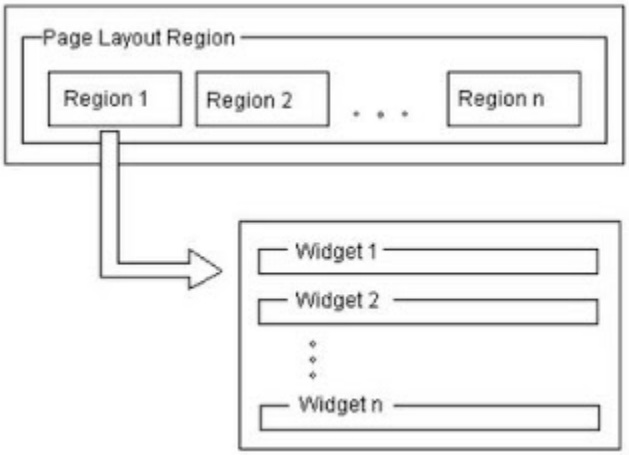
OA Frame Work Page Structure

Basics of Page Rendering
For understanding the process of page rendering, it is very important to know the structure of an OAF page.

Structure of an OA Framework Page
An OA Framework page has a collection of web beans or widgets like text boxes, buttons, etc. Every OAF page has containers for maintaining these widgets known as regions. And there always exists a parent region of all the regions known as the page layout region. This page layout region is one for every OAF page and is also the point where the Application Module is linked to the page.
Every region has a property of containing other regions and web beans. A controller object may be defined for every region that exists on the page. The following diagram highlights the structure of an OA Page:-



Processing Page Requests
The main OA Framework Page processing class is the oracle.apps.fnd.framework.webui.OAPageBean. This class uses the page name to determine which root AM is needed, so that AM can be fetched from the AM pool. The page layout region of the page actually holds the details of the AM to which the page is attached. Further, the chosen AM picks a JDBC connection, from the JDBC Connection Pool, to establish a transaction context for the page. Now depending on the valid login and request parameters, it is decided whether the request is a POST or a GET.
During both kinds of requests, the OAPageBean calls the code in the controller to instantiate each of the beans, along with setting of their appropriate properties. Certain methods which are not defined in the controller are also called for instantiating complex web beans.

GET Request

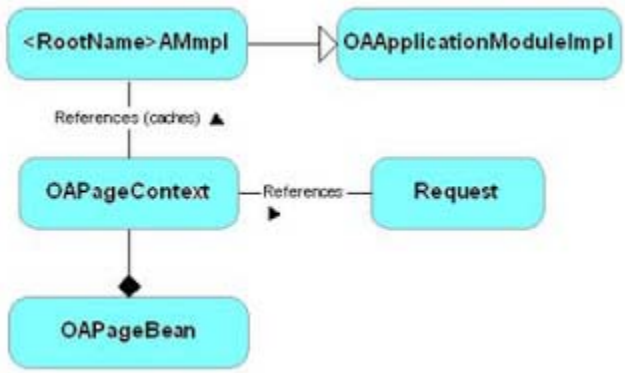
1. When a request for a page is issued for the first time, the page's declarative metadata definition is read to create the web bean hierarchy.
2. The OAPageBean class calls the page initialization code for the page in the following manner:-
a. Calling the processRequest() function of the controller of the page layout region.
b. Calling prepareForRendering() for complex web beans
c. Recursive calls to processRequest() for each of the beans in the hierarchy (if the associated controller objects are present).
3. When this page processing completes, the OAPageBean hands this processed XML to the UIX which renders it in the form of HTML on the browser.

POST Request

1. When a submit command is fired on the page, the POST request processing starts with recreation of the web bean hierarchy if one is not present.
2. A call to processFormData() is made to write the data to the model, and this method is called recursively on all the web beans recursively.
3. All the attribute and entry level validations are undertaken during calls to processFormData() and if any errors exist, appropriate exceptions are fired.
4. A call to processFormRequest() of the controller is made recursively for all the web beans.
5. V When this page processing completes, the OAPageBean hands this processed XML to the UIX which renders it in the form of HTML on the browser.

The OAPageContext

The Page Context class is one of most important components of the entire machinery, and plays the part of the binding glue to all the above mentioned components. The following class diagram explains the role of the OAPageContext class in the machinery.



The instance of this OAPageContext class is passed to each of the response processing methods in the controller to perform the following tasks:-

Accessing request parameters
One of the methods provided by the OAPageContext class, namely getParameter(), allows us to extract values of the web beans entered at the form level and use it inside the controller code for processing.

Getting access to the root AM
The contoller has to call the appropriate method of the Application Module for execution of the right database activity like calling database procedures, etc. Access to the instance of the AM is given to the controller using the OAPageContext. By firing the getRootApplicationModule() method on the OAPageContext, a reference to the AM is provided to the controller in its local instance of the AM.

Navigation and forwarding
Navigating across pages in an application is a necessary feature. This functionality is given life by the OAPageContext. The setForwardURL() method effects the feature of page navigation. The basic input parameters to the page include the server path of the target page, session level parameters to the page and a flag to determine whether the AM should be maintained for the next page.

Accessing application context information
Session level information, which needs to remain available across pages, constitutes the application context information. This is also stored and transferred by the OAPageContext. The information may either be inherent attribute values like user ID, employee details or user-defined values put in the session by controller or AM code.