Rapid Application Development of Oracle Web Systems

There are many ways to implement a web-enabled Oracle database using complex tools such as XML and PHP. However, these are not easy tools for deploying complex Oracle web systems, and Oracle Application Express (HTML-DB) opens up a whole new world for Rapid Application Development (RAD).

This book excerpt (http://www.rampant-books.com/book_2005_2_html_db.htm) provides step-by-step examples for getting started building web enabled applications. Oracle Application Express (Apex, or HTML-DB) is Oracle's latest tool for fast deployment of web-based Oracle systems, and this excerpt will get you started, fast.

This is an excerpt from the bestselling book "**Easy HTML-DB Oracle Application Express: Create Dynamic Web Pages with OAE**" (http://www.rampant-books.com/book_2005_2_html_db.htm) by Michael Cunningham and Kent Crotty.

Introduction to Apex (HTML-DB)

This chapter serves two main purposes. The first is to introduce the HTML DB development environment, and the second is to provide a reference for topics that are not specifically mentioned in other chapters.

The Application Builder is complex, and this book will not cover it all. The book is written for the beginning to intermediate level HTML DB developer, so the most common things any user should know when starting out will be covered.

Instead of taking up space in the individual chapters describing the HTML DB pages used to edit the attributes of various components, they will be included in this chapter. When applicable, more detailed explanations will be included in the other chapters.

Access to the Application Builder is gained from the Workspace home page. While on the workspace home page, click on the Application Builder icon. This will bring up the Application Builder home page as shown in Figure 6.1.

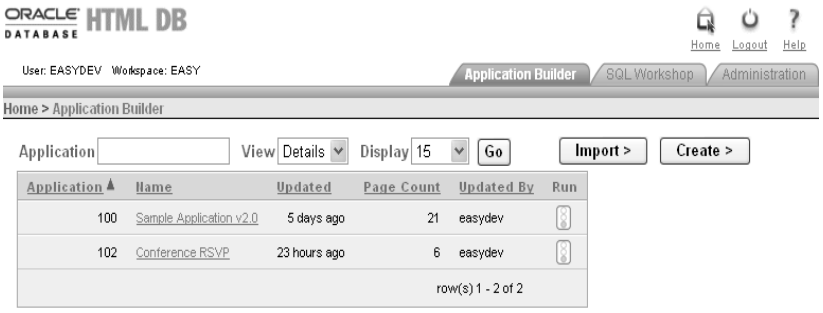

Figure 6.1: *Application Builder home page.*

The screenshot in Figure 6.1 shows the applications in the EASY workspace. The Application display shows properties of the existing applications such as Application ID, Name, last time the

application was updated and by whom, and the number of web pages in the application. This page can be used to Import an application from an export file, such as the application included in the on-line code depot. The importing of applications will be covered in another chapter on deployment. Invoking the Create Application wizard is done by clicking the Create button on the Application Builder home page.

Editing an Application

From the Application Builder home page, click on the Conference RSVP application. This will bring up the Application's home page, shown in Figure 6.2.

There are several options available from the Application's home page. The Pages region shows a list of the pages in the application as well as some key attributes of each page, such as page id, the last time and by whom it was updated, and whether the page is locked for development. The application can be run by clicking on the Run Application icon or run a single page by clicking on the run stoplight icon .

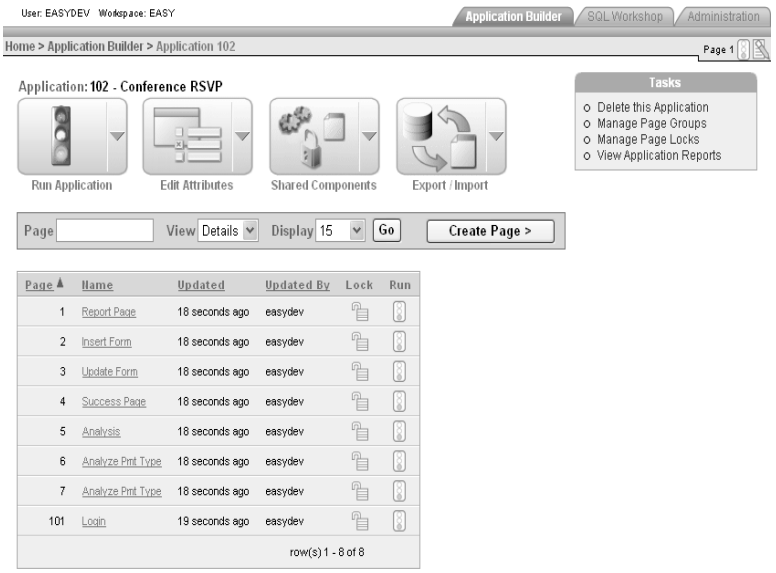

Figure 6.2: *Application Home page for Conference RSVP.*

Run Application

This icon can be used to run the application just as if it were being entered in the browser as a URL. This simulates what a visitor to the web application would encounter upon entering the application.

Edit Attributes

This is used to modify global Application Attributes and is described in greater detail in the Application Attributes section later in this chapter.

Shared Components

Shared Components are parts of an application intended for use on many pages of an application. They are reusable components. Many of the shared components can participate in what is called Publish and Subscribe. For example, a List of Values (LOV) such as Products could be defined in one application and published to other applications. If the LOV were to be modified, the changes can be published to all the subscribing applications. There is also more detail for this option later in the Shared Components section.

Export/Import

This function is used to export the entire application making it easy to deploy to another HTML DB database, or to export various pieces of the application such as Themes, Pages and User Interface Defaults.

Tasks

Many pages in the HTML DB development environment include additional links in the Tasks region in the upper right area of the application home page. These are usually tasks that are not used as often as the tasks available as icons. The tasks available from the Application home page are shown in Table 6.1.

TASK	DESCRIPTION
Delete this Application	This is obvious and will allow the deletion of the application.
Manage Page Groups	Allows the creation of Page Groups. Page Groups are used as a way to group similar pages together. Some examples are billing, customer maintenance, etc.
Manage Page Locks	This task allows application pages to be locked or unlocked. Locking a page prevents other developers from opening the page for making modifications.
View Application Reports	Navigation to a page that provides links to the many application reports available. There are reports for Shared Components, Page Components, Activity and Cross Application.

Table 6.1: *Application home page tasks*

Standard Attributes

The Application Attributes are where the global properties are set for the application. Navigate to the Edit Application Attributes page by clicking on the down arrow in the Edit Attributes icon and then click on the Edit Definition menu item. The standard attributes are explained in the following sections.

Name

This is where the name of the application is set. It is also the name displayed in the Application Builder home page.

Application Alias

The text entered here can be used in the URL to the application. The application alias can be used in lieu of the application id. For example: if the application id is 102 as in the Conference RSVP application, the application alias can be set to HELLOWORLD. Either of the following can be used on the URL's to get to the application.

```
http://localhost:7777/pls/hdb20/f?p=102
http://localhost:7777/pls/hdb20/f?p=helloworld
```


Use the Application Alias cautiously. These should be unique in the HTML DB engine. If the same alias name is used more than once, it may confuse the HTML DB engine.

Version

The version field provides a place to set a version that can be used in page templates to indicate the version on every page of the application if the user chooses. To add the version to a page template, use the #APP_VERSION# substitution string in the templates.

Image Prefix

This is set to /i/ by default and represents the same setting as is in the dads.conf file as presented in the section of this book on Database Access Descriptor (DAD). The value here must be the same as in the DAD file.

Logging

This allows the activity to be recorded in activity logs. Each page view will be logged allowing the workspace administrators to monitor the application activity. The activity reports will be covered in the chapter on advanced HTML DB administration.

Parsing Schema

The parsing schema is very important to the application. All SQL statements and PL/SQL are executed as a specific database user known as the parsing schema. This is also the schema that will own the tables, indexes, PL/SQL, etc. that will be created through the SQL Workshop. Although this can be modified for the application, EASYHDB will be used for the parsing schema throughout the book. The EASYHDB parsing schema was created when the easy workspace was created earlier in this book.

Status

This option is used to set the current status and availability of the application. This can be handy when in a development mode for an application or when it is necessary to make an application unavailable during maintenance. The options are shown in Table 6.2 below.

OPTION	DESCRIPTION
Available	The application is available to administrators, developers and other users; however, the developer toolbar will not be displayed on the web pages when they are executed.
Available with Edit Links	The application is available to all and the developer toolbar will be displayed when logged into the Application Builder as an administrator or developer.
Available to Developers Only	Only administrators and developers will be allowed to run the application. Also, a built in authentication scheme must be used. Authentication schemes are covered elsewhere in this book.
Restricted Access	With this option, the Restrict to comma separated user list, where the status must equal Restricted Access, text area is used to build a list of users allowed to use the application.
Unavailable	This option will not allow the application to be executed. The text entered in the Message for unavailable application text area will displayed on the screen to any user attempting to use the application.
Unavailable (Status Shown with PL/SQL)	Use this option to be allowed to write custom PL/SQL that will build html code to display to the user.
Unavailable (Redirect to URL)	This will allow the user to enter a URL to redirect the user to another web site. Make sure to use the full URL including the leading http:// or risk a failure of the redirection action.

Table 6.2: *Status options and their descriptions*

Build Status

There are two settings for build status. The Run Application Only status will allow the application to be executed, but the application will not be displayed on the Application Builder home page. Leaving this setting at the default, Run and Build Application, will allow it to be executed and displayed in the report of applications on the workspace home page.

Global Notification

This is used to display a global, or system wide, notification on pages of an application. A good use of this would be if there was planned maintenance on the weekend. A message to that effect could be displayed during the days prior to the maintenance alerting the users of the application maintenance. The label has #GLOBAL_NOTIFICATION# shown in the text. This is referred to as a substitution string and is very important. A later chapter will cover page templates and how these templates use substitution strings. This particular substitution string would be used in a page template to put the global notification in a strategic location on the page.

Substitutions

HTML DB has several built in substitution strings. This is an area where users can define their own. They can then be used throughout the application. A good use of this may be for a copyright string. One goal might be to create a copyright notification string that reads "Copyright 2005, HTML DB Book. All rights reserved." For this, MSG_COPYRIGHT could be entered for the Substitution String and the copyright notice in the Substitution Value. Then to display the copyright notice in an application it would be referenced with &MSG_COPYRIGHT. including the trailing period. If it is ever necessary to change the copyright notice, this is where it would be changed, and it would instantly be reflected everywhere it is used in the application.

Logo

The built-in page templates are defined with an area in the upper left corner to display a logo, an image file, if the use of one is desirable. This is exactly how the HTML DB development environment displays the image in the upper left corner of the browser. The text used to do this for HTML DB environment is shown below.

The Logo Image Attributes page item is used to set other properties of the <img> tag.

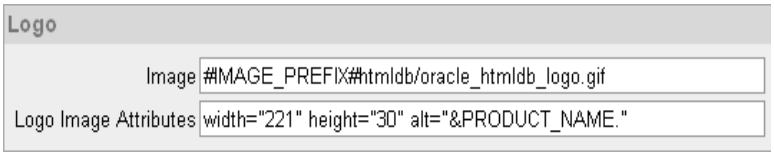

Figure 6.3: *Logo attributes.*

The #IMAGE_PREFIX# text is a substitution string and is substituted with the text /i/ as mentioned earlier in the Image Prefix section. The example above would be translated to:

```
/i/htmldb/oracle_htmldb_logo.gif
```

Security Attributes

Navigate to the Security Attributes page by clicking on the down arrow in the Edit Attributes icon from the Application home page and then click on the Edit Security menu item. Security attributes are explained in the following sections.

Home Link

This indicates the home page of the application. When the Run icon in the application home page is clicked, this is where HTML DB gets the information to know which page is the home page. It is also used in the case where someone types in a URL such as http://localhost:7777/pls/hdb20/f?p=102. The URL tells HTML DB which application ID to run, which is 102 in this example. Then HTML DB uses the value in Home Link to know which application page has been indicated as the home page.

Public User

This is the database user HTML DB uses for logging into the database. It must match the PlsqlDatabaseUsername parameter in the dads.conf file. Looking at the Oracle sessions will reveal a connection to the database with this name as the logged in user.

Authorization Scheme

This option is used to control access to the application. Authorization is different than Authentication. While authentication will confirm who is logged in and that they are permitted to log in, authorization sets the privileges for the user. Custom authorization schemes can be created for an application and can be used at the application level, page level, or down to the component level for each of the page items. If a user does not pass authorization, the page component(s) will not be displayed.

Parsing Schema

This indicates the schema owner for all the database objects the application will use. Each application can use one schema. If there is a need to use multiple schemas, it is usually done by creating a separate application and, if necessary, building navigation to go between the applications. The list of available Parsing Schemas is controlled in HTML DB Administration Services.

Session State Protection

This feature can be used to prevent savvy users from trying different parameters in the URL to gain access to application pages or data to which they should not have access. Session State Protection is explained in the chapter on Advanced Security.

Virtual Private Database (VPD)

This is an advanced feature and can be used to limit the records a user can see. It can be handy for applications where the same SQL select statement would return a different result set depending on who is logged in, such as for different regional salespersons.

Application Pages

Now that the application attributes have been covered, it is time to look at the attributes of an application page. This section will explain many aspects about page attributes, how they are rendered, and attributes of the various components on application pages.

Page Definition

On the Page Definition for an application page, as shown in Figure 6.4, there are three regions.

Page Rendering: This is the definition of the page components, such as regions, buttons, and items, and the logic controls, such as computations and processes, that will be used in the rendering of the application page.

Page Processing: This region is where programming logic is defined. These are referred to as logic controls in HTML DB. Logic controls are executed when the application page is submitted to the HTML DB engine.

Shared Components: This is a list of the shared components used on the current application page. This region can also be used to create new shared components or edit existing ones by clicking on the links for the various shared components.

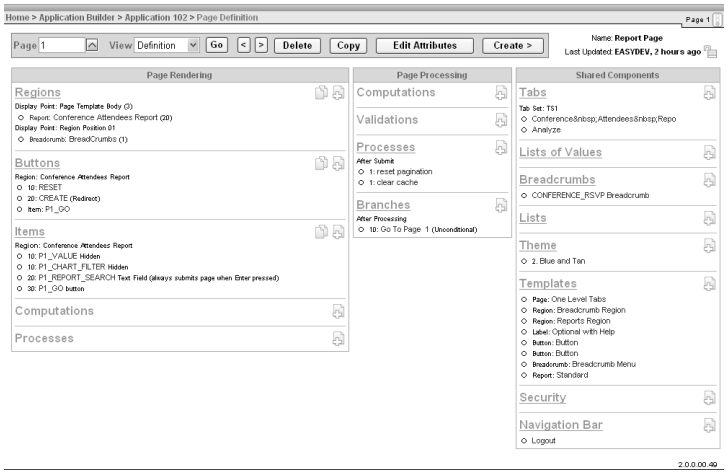

Figure 6.4: *Page Definition for an application page.*

Regions

Regions act as a container for other page components. There are many types of regions such as reports, forms, charts, calendars, web services, TREE's, etc. The components can be a report, text fields, buttons, etc. Regions can then be positioned on the web page, and all the page items in the region will be positioned with it. Regions can also be displayed on the screen or hidden from view based on conditions set by the user.

The following attributes are part of the Region Definition page. The Region Definition page is accessed by clicking on the region link on the applications Page Definition page. The link is shown being selected in Figure 6.5.

Figure 6.5: *Regions section.*

Name

Title

This is the text that will be displayed at the top of the region. This page item supports more that just straight text. A page item can be added to the Title, and it can be made dynamic. A good use

for this would be for a policy number. Entering the text Policy - &P1_POLICY_ID.would produce something like Policy - A756123 when the page is rendered.

Another common thing that HTML DB uses in its own development environment is to enter an anchor tag in the Title. For example, it has as the Title a value of Conditions. This serves as a bookmark and allows the development environment to provide bookmark links at the top of pages so it is possible to quickly scroll to that part of the page.

Type

This will dictate how HTML DB will render the region. Types will be covered in greater detail throughout the book, but a brief description is provided here to serve as a point of reference.

The region type is set to instruct HTML DB what type of text is entered in the Region Source text area. That way HTML DB knows how to treat the text when the page is being rendered. Table 6.3 shows the source types used to indicate how a region should be displayed or populated. …

SOURCE TYPE	DESCRIPTION
HTML Text	Allows the user to enter standard HTML syntax to be rendered in the region. Substitution strings, including page item values, can be inserted into the HTML code by using substitution string syntax such as &APP_USER. The HTML Text type is also used as the region type when a form page is created.
HTML Text (with shortcuts)	Allows the same as HTML Text, but also allows the inclusion of HTML Shortcuts. Shortcuts are pieces of reusable code.
HTML Text (escape special characters)	Allows the same as HTML Text but will render special characters in their original form. As an example: if the user wanted to display the text " in the region. This is the HTML syntax to display a single quote ('). If the region type HTML Text is used, it would be rendered as ('), but setting the region type to HTML Text (escape special characters), the text will display as *"*
SQL Query	This type of region type is the most common type used when creating a report. The HTML DB engine will execute the SQLselect statement in the region source to populate the report. Even though the Report Wizard will provide a step-by-step guide to build a report, this is more common. Many times an experience programmer will have someone providing them with an SQL select statement they would like to see displayed on a web page and this is the type of report region created when I use the SQL Report wizard.

SOURCE TYPE	DESCRIPTION
SQL Query (Structured Query)	This type is assigned to the region when creating a report using the Report Wizard. Regions of this type are not normally created manually.
SQL Query (PL/SQL function body returning SQL Query	This region type can be used to write a PL/SQL function that dynamically builds an SQL select statement and returns the string from the function. HTML DB will then use that string as the select statement to populate the report. This is a powerful feature and can be used by experienced SQL programmers for flexibility.
SQL Query (updateable report)	This type of region is for tabular forms. A tabular form allows the editing of multiple rows for a table at a time. This type of form is best built using the built-in Tabular Form Wizard.
PL/SQL (anonymous block)	This type of region is best used when using the HTML API packages such as the HTF and HTP packages. An example of this would be to put owa_util.print_cgi_env; into the Region source to display the CGI environment in the region.
Chart (HTML) chart	This is a simple chart provided for in the HTML language. This type of chart is rendered using HTML table tags.
Chart (SVG)	This is a more powerful type of chart using the Adobe SVG (Scalable Vector Graphics) standard.
Web Service Result	This type of region permits the use of services provided by other sources. For example: Federal Express provides web services by which one can use to track shipping orders.
Easy Calendar	This is the region type assigned when the calendar was created using the Easy Calendar wizard. This wizard will walk the user through a selection process to pick the table and columns to use for the population of the calendar
Calendar	This is similar to the Easy Calendar except the user will be asked to provide the SQL statement used to populate the calendar.
TREE	A tree provides a page component to represent multi-level information such as a directory tree for a file system, or for District, Store, and Department type of drill down information.

▪ SOURCE TYPE	▪ DESCRIPTION
▪ URL	▪ Using this type of region, a URL can be entered and the results of the URL will be rendered inside the region.

Table 6.3: Source types for application page regions.

User Interface

Template

This defines the template that will be used to determine the look of the region.

Display Point

Display points are explained in more detail in a later chapter of this book.

Region HTML table cell attributes

This attribute is only used when there are multiple regions displayed side by side. For example, if there are three regions lined up from left to right instead of top to bottom, regions are across multiple columns. In that case, this attribute can be used to add additional attributes to the region, such as aligning them vertically to the top of the row. This would be accomplished by entering valign="top" for this attribute. To have them display right next to each other, a width would have to be entered on the region to the left that is small enough to allow the region on the right room to move over. To accomplish this, enter width="200" for this attribute.

Source

This is the source code used to populate the region. Depending on whether the user is viewing a report based on a Structured Query, an SQL Query, a Chart, a Calendar, etc. this region will display differently. For example, a Structured Query is not editable except by using the Query Definition tab. If the report is based on an SQL Query, the source region will provide a text area for editing the SQL statement for the report. This is assuming the region type is SQL Query or a similar type as explained above in the Name section. For charts, a Build Query button is provided to guide users through building the SQL statement used to populate the chart.

Conditional Display

The tutorials in this book will provide ample exposure to the user of this attribute, so there will be plenty of opportunities to gain experience with it. There are far too many condition types to explain in detail, but some of the more commonly used conditions are explained in order to provide an understanding of when and where it might be desirable to use them.

The conditional display is used to prevent the rendering of a particular region or page item. Conditions are meant to return a TRUE or FALSE value. TRUE is the default and if returned, the region or page item is rendered. A return value of FALSE will prevent the rendering. Table 6.4 shows the most popular condition types for the conditional display.

▪ CONDITION TYPE	▪ DESCRIPTION

▪ CONDITION TYPE	▪ DESCRIPTION
▪ -No Condition -	▪ The Default. This will cause the component to always be rendered.
▪ Exists (SQL query returns at least one row.	▪ This is very common for use when it is desirable to only display a component if a record exists in the database. A common use would be for verifying the user is an administrator.
▪ NOT Exists (SQL query returns no rows)	▪ This is the reverse of the previous item. If a row were to be returned by the query, it would result in a FALSE answer and the component would not be rendered.
▪ SQL Expression	▪ This is similar to the WHERE clause in an SQL stat ▪ A good use for this is when none of the other condi types meets your need. For example: you may want to have condition that the page must be equal to 101 and it is not in friendly mode. This is similar to having two conditions work to return a true of false value.
▪ Request = Expression1	▪ This condition type is mostly used during page processing on conditions, validations, and processes. For example, a computation does not have a condition where you can specify the button from a select list. So, for computations you would have to use this type of a condition.
▪ Value of Item in expression 1 Is NULL	▪ This type of condition is commonly used on buttons such as a Create or Insert. Let's say you have a page item on a form named P100_CUSTOMER_ID. If that page item is NULL then it would indicate this is a new customer and you would want the Create or Insert buttons to be rendered. This is exactly how the HTML DB wizards setup a form created by a wizard.
▪ Value of Item in expression 1 Is NOT NULL	▪ This type of condition is commonly used on buttons such as a Update or Delete. Let's say you have a page item on a form named P100_CUSTOMER_ID. If the page item has a value then the customer already exists and you would want the Update and Delete buttons to be rendered. This is exactly how the HTML DB wizards setup a form created by a wizard.
▪ Current Page Is Contained Within Expression 1	▪ For this condition you provide a comma delimited list of application page IDs. This type of condition would be used to display Lists or Tabs only for a select list of pages.
▪ Current page is NOT in Printer Friendly Mode	▪ This condition type is often used on the application page referred to as Page 0 (zero). It is common to display a banner at the top of the page using an after header region. You would use this condition to prevent the region from being rendered when the user put it into the Printer Friendly mode.

▪ CONDITION TYPE	▪ DESCRIPTION
▪ User is Authenticated	▪ In the event there is a page that is public and does not require authentication. However, that page is also used by authenticated users. You have one region on the page that you only want to be displayed if the user is authenticated while the rest of the regions can display in either condition. For that one region you could use this type of condition. I've used this type of region on a page so I could see auditing information, but it would not be displayed to the public.
▪ Never	▪ This will prevent the component from being rendered. It's also used to prevent a branch, computation, validation, or process from being executed. I've most often used this computation to prevent certain processes from running while I test an application page.

Table 6.4: *Conditional display types*

Header and Footer

The Header and Footer region is used to modify the look of a region by allowing the user to provide HTML code that will be inserted upon the region being rendered. A common use for this feature within HTML DB is to add descriptive text to the top of a region. For example, the copy region wizard in HTML DB displays helpful text at the top of the regions and this is done by using the Header and Footer.

Authorization

Custom authorization schemes can be created within HTML DB. An authorization scheme may be something like IS_MANAGER or CAN_EDIT_PRODUCTS. These authorization schemes can be created in Shared Components → Authorization Schemes. The authorization scheme either evaluates to TRUE or FALSE. If it evaluates to TRUE, the page component will be rendered. If it evaluates to FALSE, the component will not be rendered.

A common use for this would be for something such as create or edit buttons. If the user has authorization to perform edits, the buttons would be displayed. If not, the buttons would be unavailable.

Customization

Although not a very technical description, this feature is really cool! The developer wants to provide a page with ten regions, which could be reports, forms, or any other type of region, on the page, but some users may only be interested in any combination of three or four regions. Instead of creating several pages to satisfy each application user, all of the regions can be placed on a single page and it can be made customizable. What this feature will do is add a Customize link on the page thereby allowing the user to display a popup with the names of all the regions. The user can then select the regions they want to see and unselect the regions they do not want to see.

In each of the regions, the developer can decide if the region should participate in the customization popup window. There may be one region to be displayed in any situation; whereas, the rest of the regions can participate in user customization.

Customized Option Name

By default, the name to be shown in the popup window will be the Region Title. The Customized Option Name is used to provide an alternate name in the popup Customize Page.

This requires the developer to be using an authentication scheme where each user has a separate login to the HTML DB application. HTML DB must be able to distinguish between users so it knows which regions to display next time the user renders the page.

Configuration

This region allows a Build Option to be selected. A Build Option is used to enable or disable functionality within an application. They are created in Shared Components → Build Options. When a build option is created, it is given a status of INCLUDE or EXCLUDE. If the status is INCLUDE, the components with that build option will be rendered by HTML DB as normal. If the status is EXCLUDE, HTML DB will ignore them during rendering. A build option can be assigned to application pages, regions, page items, page processing, and other page components to include them or exclude them from the rendering process. For example, there may be a build named Version 1.1 that is set to EXCLUDE. This build option may be assigned to page components that are not ready to be rendered. Once testing is completed, the version 1.1 build option could be set to INCLUDE and the page components would be rendered in the application.

Comments

This is a good location to keep notes about the report region. This area could be used to keep track of changes made or perhaps plans to be made in the future.

Page Events View

The Page Events View is accessed by selecting the Events option in the View select list as shown in Figure 6.6 and pressing the Go button.

Page Events are an extremely valuable tool to the HTML DB newcomer. It shows a detailed view of when each page component is being rendered by the HTML DB engine. Figure 6.6 shows page three of application 102, the Conference RSVP application being displayed in the page events view. In order, the After Header processes are executed, the Breadcrumb menu is rendered, and then the Conference Attendee region and page items are rendered. Understanding that one event must occur before another can save hours of frustration. This is one of the tricky things to get the hang of when learning HTML DB.

Page rendering events and page processing events are explained in greater detail in a later chapter of this book.

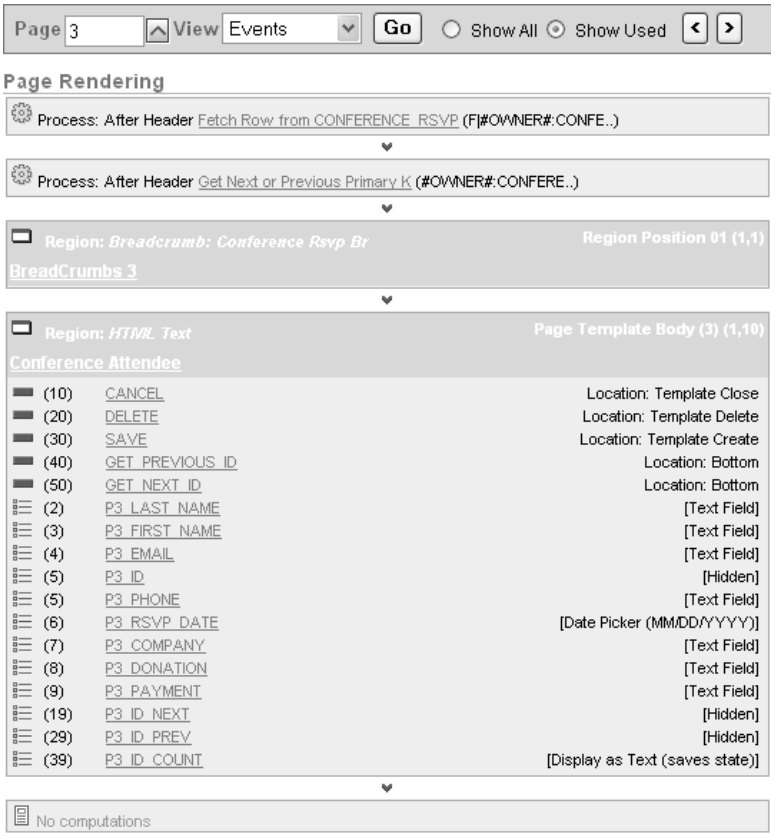

Figure 6.6: *Page rendering events view.*

Page Objects View

This view is used to see the database objects on the application page.

The Page Objects View is accessed by selecting the Objects option in the View select list as shown in Figure 6.6 and pressing the Go button.

Page History View

This view is used to see the history of changes made to page components on the application page. It will show the changes made and by which developer. It is also possible to see the history of when the page was locked and unlocked.

Page Export View

This view provides another way of exporting an application page. This option has generated a lot of questions on the HTML DB forum. The export implies that it would be possible to import the page into any other application, but that is not the case. An exported page can only be imported back into the same application id that exported the page. The value of exporting an application is so modifications can be made to the application, and if something goes wrong, the page can be re-imported to get back to where you started.

Page Groups View

This view will show the other pages that belong to the same group as the current page.

Page References View

This view displays the other application components that reference this application page. Figure 6.7 shows that a breadcrumb menu, two branches on pages one and three, and a tab have links to this page.

Application	From Component	Relation	To Page
102	CONFERENCE_RSVP Breadcrumb	Breadcrumb	1
	Page 1	Branch	1
	Page 3	Branch	1
	Conference Attendees Report	Tab	1
		row(s) 1 - 4 of 4	

Figure 6.7: *Other pages that reference the current application page.*

Page Rendering

Page rendering is the action taken by the HTML DB engine to render the application page in a browser. The Events view of an application page shows the Page Rendering execution order, shown previously in Figure 6.6. The events view shows the order in which page components such as regions, buttons, items, computations, and processes are executed. Page rendering is also covered later in this book in the section on Page Processing.

Page Level Computations

Computations can either be processed while the page is being rendered or while the page is being submitted. A page level computation can be used to set the current date and time of a page item.

Page Level Processes

Processes can either be executed while the page is being rendered or while the page is being submitted. There is a significant difference with each. Page level processes are considered for execution as the page is being rendered. This is handy for executing a page process that is used to populate the page items on the page.

Session State

To the seasoned developer, session state can best be defined in terms of global variables. Session state is the storing of page item values in the database. Their values can be referenced from any page in an application. Since they are stored in a database table and are specific to each HTML DB session, their values can be referenced in stored procedures.

A big benefit any experienced web developer will notice immediately is that session state is managed by HTML DB. Nothing special like saving them in hidden page values has to be done.

Just being able to reference session state from a stored procedure will show how valuable it is to have HTML DB provide this ability.

Session state for page items are tied to a page so all the page items can be set to NULL values by clearing the cache for a page. It is also possible to set all the page items in the entire application to NULL by clearing the cache for the application.

Viewing the values in session state can be done by using the Session link on the developer toolbar. It is also possible to view session state for the whole workspace by logging in as an administrator and navigating to HTML DB Workspace Administration → Manage Services → Session State

Navigating to the HTML DB Workspace Administration is accomplished by clicking on the Administration tab.

Reference Types for Page Items

While using HTML DB, it will be necessary to reference the session state for page items. There are several ways, or reference types, for referring to the page item depending on the type of code the HTML DB engine is executing. For the most part, page items in SQL are referenced using the Bind Variable syntax and when editing page items, their values are set using the Substitution String syntax.

Bind Variable: Syntax :P100_CUSTOMER_ID

This syntax is used within the SQL statements for page processes, page computations, and page validations.

Substitution String: Syntax &P100_CUSTOMERID.

When a page is rendered, the HTML DB engine will substitute the text of the substitution string &P100_CUSTOMER_ID. with the value of the session state, such as the value 231. The syntax of the substitution string includes a leading ampersand (&) and a trailing period (.). The substitution strings can be used in region titles, page items, URL's, and other locations.

PL/SQL: Syntax V('P100_CUSTOMER')

This is used to reference the session state of page items from a PL/SQL stored procedure. When the page is submitted, the session state is stored in HTML DB tables for each of the page items. The session state can then be referenced by using the V and NV functions. The V function is for alphanumeric values and the NV function will return the value as a number.

Template Substitution: Syntax #P100_CUSTOMER_ID#

This syntax is primarily used in templates to reference attributes set on attributes pages such as application attributes, page definition, etc. This syntax is also used on reports to reference the values returned for columns in an SQL select statement. One example is to use this syntax in a report column link to refer to the column in the SQL statement.

Preferences

Preferences comprise a persistent session state that is maintained between user sessions. That is, when a user logs out and logs back in, the Preferences can be re-read and used. They are like variables that last forever between uses of the application.

In HTML DB, the developer may have chosen to modify the default view from Icons to Details, so HTML DB saved that selection as a user preference. HTML DB then uses the stored user preference to know how to display the screen the next time the developer logs in to do development.

Preferences are user specific. If public authentication, Page Is Public, is being used for web pages, all users are authenticated with the same user preferences. Since public pages do not require a valid user login id, HTML DB uses the username of NOBODY; therefore, since everyone using public pages is known to HTML DB as nobody, the preferences are shared among all web users. If the previous visitor to the web page sorted the report by the Payment Type column, the next user to visit would see the same sorting due to the preference being reused. In this case, the developer may want to create a page process that will reset the preferences as the page is being rendered. That can be done by creating a Session State page process with the type set to Reset Preferences (remove all preferences for current user). This can be handy on report pages where data can be sorted. In the description above, the preferences are being reset before loading a report page to restore the default sorting of how the report was designed. The section in this book on HTML DB APIs provides more information on the htmldb_util.remove_sort_preferences procedure.

HTML DB URL Format

Understanding the URL used in HTML DB will help improve a developer's efficiency. The URL has several distinct parts. For reference look at the sample URL that follows.

```
http://localhost:7777/pls/hdb20/f?p=102:3:7646151020298818363::NO::P3_ID:28
```

http://localhost:7777/pls/hdb20 is the address used to navigate to HTML DB.

The rest of the URL is actually a call to a stored procedure within the database named F. The parameter being passed is a colon delimited string to the P parameter in the procedure. The values in the colon delimited string have a specific position they must be presented to the procedure. These positions are referred to as arguments. The arguments are shown here.

```
f?p=App:Page:Session:Request:Debug:ClearCache:itemNames:itemValues:PrinterFriendly
```

App: This is the application ID or the alias of the application.

Page: The page ID to render or the alias name of the page.

Session: A numeric value indicating the session ID. The session ID is needed so HTML DB knows which session state to reference.

Request: When buttons are pressed, they set the REQUEST, normally to the name of the button. The next page can then reference the REQUEST to know the name of the button the user pressed.

Debug: Set this value to YES to display debug information in the page being rendered.

ClearCache: This will clear the cache for an application page. To clear the cache for multiple pages, a comma delimited string of page ID's can be provided. This is a URL argument to remember. It will be used quite often during development and testing. It is argument six in the colon delimited string. To clear cache for the entire application, enter the value APP for this argument.

itemNames: This argument is used to set the session state of page items. Multiple page items can be set here by using a comma delimited list. Page items are global so even though the goal is to navigate to page 100, the session state can be set for items on other pages or application level items.

itemValues: This is partner to the itemNames argument. These are the values used to set the session state for the items listed in the itemNames argument.

PrinterFriendly: To render the page in a mode that is better for printing, set this argument to YES. It is then possible to set the conditional display to Current page is NOT in Printer Fiendly Mode. The page component would then not be rendered if the PrintFriendly argument is set to YES.

Shared Components

There is an icon on the Application Home page to access shared components. Shared Components are used by many pages in an application. In some cases, using the publish and subscribe feature of HTML DB, these shared components can actually be used by other applications.

Logic: Application Level Components

Most documentation of Shared Components will not mention Application Level Components. That is because, technically, they are not defined in HTML DB as a shared component. However, an application level component can be used anywhere within the application. Application Level Components consist of those described in Table 6.5.

COMPONENT	DESCRIPTION
Application Items	Application Level Items are similar to hidden page level items. They have session state and are not visible page components. One difference is they are not named with the page prefix (P100_). Instead, they will be named with the application prefix F102 for app id 102, or another prefix such as G_ can be chosen. This has been used to indicate Global.
Application Processes	These are globally defined processes that can execute for every page in the application. It could be handy for inserting a record into a table to indicate a page is being viewed. Another great use for an Application Process is the ability to define a process here and call it on the appropriate pages by creating an On Demand process on an application page.

COMPONENT	DESCRIPTION
Application Computations	An Application Computation would be setup if there is a computation the developer wants to execute for every page in the application or if they simply wanted it to execute when the user creates a new session, which is when the user first accesses an application page. The On New Instance computation point is the one that executes when the user first accesses the application and is commonly used to read browser cookies.

Table 6.5: *Application level components*

Web Services

The building of web services is covered later in this book in the section on Creating Other Components with HTML DB.

Build Option

Using build options can disable certain functionality. This is done by creating a build option and setting the page component attribute called build option. Figure 6.8 depicts that a build option named Version 1.1 has been created. It will be turned on, and at a later time, set to INCLUDE when the new features are ready for the public. Another use for this is if there are problems discovered with the new features after the application is in production, the developer can simply change the build option back to EXCLUDE and HTML DB will not render the new features. If used cautiously, this feature could be used to back-out changes in the middle of the day.

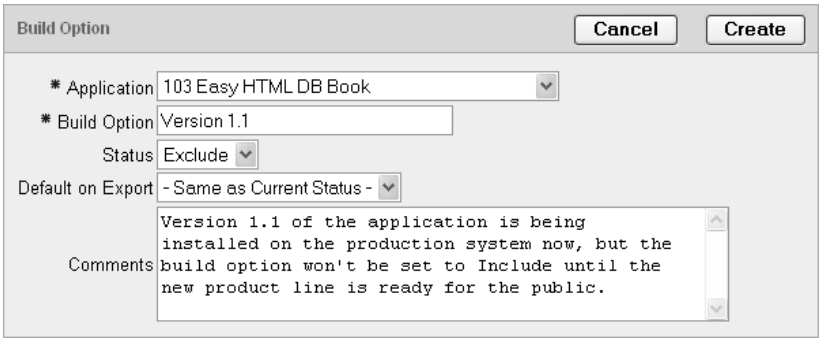

Figure 6.8: *Creating a build option.*

Security

Security will be covered in detail later in this book in the chapter on HTML DB Administration II.

Navigation

The components responsible for HTML DB navigation are covered in detail in a later chapter in this book.

User Interface

The user interface section includes Themes, Templates, User Interface Defaults, Lists of Values (LOV's), and Shortcuts.

Themes and Templates are covered in a later chapter of this book.

User Interface Defaults

User Interface Defaults provide the ability to set default properties for tables and views and columns. These defaults are used by the wizards when creating reports and forms. Not only does this feature provide a consistent name for the components placed on an application page, but it also saves time because they are defined ahead of time.

To create a user interface default, navigate to the Shared Components page and click on the User Interface Defaults link. This will bring up a list of the tables in the parsing schema. An experienced developer would encourage new developers to switch the view to Details by selecting the Details item in the View select list and clicking the Go button.

1. Click on the DEMO_ORDERS table link.
2. On the next page, click on the Create User Interface Defaults button.

   The next page is the Table Defaults page shown in Figure 6.9. It shows several attributes for each column. When a report or a form is created and columns are added to that report or form, they will obtain the defaults set here. The order in which the column will be displayed can be established by setting the sequence and whether or not the column should be included.

Column Name	Label	Include In Reports	Report Sequence ▲	Searchable
ORDER_ITEM_ID	Order Item Id	✓	1	-
ORDER_ID	Order Id	✓	2	-
PRODUCT_ID	Product Id	✓	3	-
UNIT_PRICE	Unit Price	✓	4	-
QUANTITY	Quantity	✓	5	-

Group By	Aggregate By	Include In Forms	Form Sequence	Required
-	-	✓	1	✓
-	-	✓	2	✓
-	-	✓	3	✓
-	-	✓	4	✓
-	-	✓	5	✓

Figure 6.9: *Table Defaults.*

3. Click on the UNIT_PRICE column link to edit the default properties.
4. There are several properties that can be set for the UNIT_PRICE column. Some of them are:

Label: This is the default label, or column heading, for the page item when it is added to a page.

Mask: For dates and numbers, this is valuable because the formatting can be set here once and it will not have to be done every time a report or form is created.

Display As: For dates, the developer may want the default to be a Date Picker. This is also really helpful for large text fields where it is desirable to default this to Text Area.

Default Value: This is another valuable property to set for a date. An example of this is setting the default to today's date by entering the following text in this property.

```
( select to_char( sysdate, :PICK_DATE_FORMAT_MASK ) from dual )
```

Help Text: If the Help Text feature is used in an application, this can be an incredible time saver.

List of Values: If the column is a select list and uses a List of Values, it can be defined by using the List of Values tab.

5. For the UNIT_PRICE column, make the following selections:

   Report Defaults → Mask: &PICK_DATE_FORMAT_MASK.

   Tabular Form Default → Display As: Date Picker (use application format mask).

   Form Defaults → Mask: &PICK_DATE_FORMAT_MASK.

   Default Value: Use the query shown above in the Default Value bullet point.

6. Click the Apply Changes button.

The authors of this book believe that user interface defaults are an underutilized feature in HTML DB. To not use this feature should be considered cruel and unusual punishment to the developer, because they will find themselves making the same attribute changes to page item after page item to keep things consistent.

Lists of Values (LOV)

A LOV is a reusable list that can be built dynamically with a SQL select statement or with a static set of values typed in by the developer. Examples of static LOVs are:

Gender: Male, Female

Days of Week: Monday, Tuesday, Wednesday, etc…

The values in dynamic LOV's are populated using a SQL select statement such as:

```
select description d, manufacturer_id r
from   manufacturer
order by 1;
```

LOV's can be used in forms, in tabular form columns, in reports using Display as Text based on LOV, and select lists. More explanation on creating and using LOV's is included later in this book.

Shortcuts

Shortcuts enable the creation of reusable HTML or PL/SQL code. For an idea of how this is used, the select list in the Conditional Display region for a button is shown in Figure 6.10. There

is a button displayed after the select list followed by a list of items shown directly below the select list. In the HTML DB development environment, this was done using a shortcut. The name of the shortcut is then placed in the Post Element Text of the select list item. This example shows that HTML DB is actually written in HTML DB.

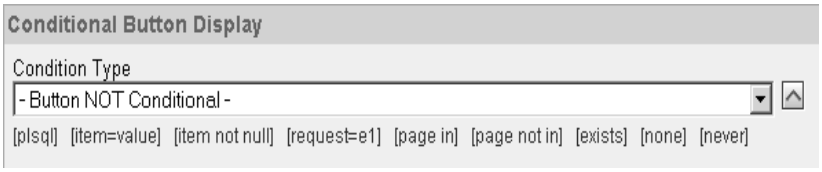

Figure 6.10: *Shortcut use in HTML DB Development Environment*

As an exercise to demonstrate how to create and use the developer's own shortcut, a page item that displays a date page item will be created. A link will be provided on the side of the page item that will allow the user to click on it and set the item to today's date.

This is defined in the following four step process:

1. Create a blank application page named Shortcuts. In the Page Attributes, add the following text to the HTML Header section. This is using the functions.js script provided as part of HTML DB. It provides the source code for the setValue() function used later in the shortcut. What is being accomplished here is borrowing some code provided with the installation of HTML DB.

   ```
   <script src="#IMAGE_PREFIX#javascript/functions.js"
   type="text/javascript"></script>
   ```

2. Create a shortcut from scratch by going to Shared Components →Shortcuts.
   - On the Creation Method page, select From Scratch and click Next.
   - On the Shortcut Attributes page:
     - Name: SET_TODAYS_DATE.
     - Type: PL/SQL Function Body.
     - Shortcut: Enter the text from below. The file is in the Code Depot.
   - Click Create.

1. plsql_shortcut.sql

```
declare
   r  varchar2(32767) := null;
   d  varchar2(11) := null;
begin
   d := to_char( sysdate, :PICK_DATE_FORMAT_MASK );
   r := '<a href="javascript:setValue(''#CURRENT_ITEM_NAME#'','''
        || d || ''');">';
   r := r || '<img src="#IMAGE_PREFIX#r_blue_arrow.gif"
         width="16" height="16"></a>';
   return r;
end;
```

In the Listing above, make sure there are not any extra lines or characters after the final semi-colon (end;). This will cause a PL/SQL error.

Not only is the setValue JavaScript function provided by HTML DB going to be used in this example, the above script references a gif image named r_blue_arrow.gif, which is provided with HTML DB. Take note of the date format. It is possible to change the date format to the developer's preferred format. Another thing that can be done is the use of a substitution string for the date format to keep the date format consistent throughout the application. One such substitution string would be PICK_DATE_FORMAT_MASK. If so, the statement would look something like this:

```
d := to_char( sysdate, :PICK_DATE_FORMAT_MASK );.
```

1. When using the PICK_DATE_FORMAT_MASK as mentioned above, the page item would be created as a Date Picker (with application format mask).

2. Create an HTML region on the page and then a Date Picker on the page. This way the formats will always stay consistent whether the Date Picker icon or the Shortcut icon is used.

3. After the page item is created, edit the new page item and add the text SET_TODAYS_DATE to Post Element Text (include the double quotes), as shown in Figure 6.11.

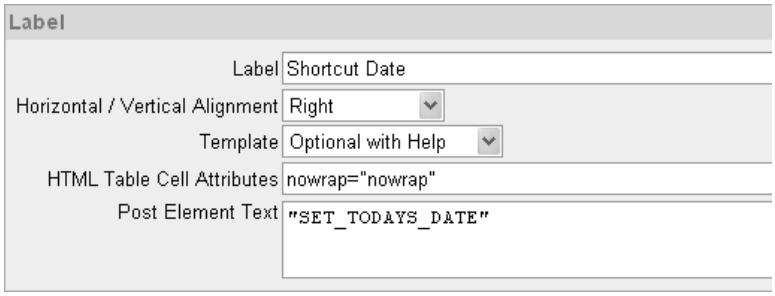

Figure 6.11: *Using a Shortcut in the Post Element Text.*

Be careful: Shortcuts are case-sensitive. It is recommended that developers use the same case as the HTML DB development team, which is UPPERCASE.

Apply Changes to the page item and run the page. It should look similar to Figure 6.12. What has happened with the SET_TODAYS_DATE shortcut is HTML DB has executed the PL/SQL block and returned the following HTML code.

```
<a href="javascript:setValue('P2910_SHORTCUT_DATE','01/24/2006');">
<img src="/i/r_blue_arrow.gif" width="16" height="16"></a>
```

It is this HTML code that was rendered as the Post Element Text for the page item. It creates an anchor tag and an image tag displaying a blue arrow. Clicking on the blue arrow executes the JavaScript in the anchor tag.

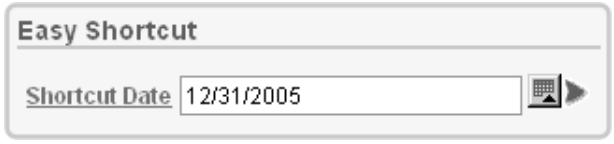

Figure 6.12: *Using a Shortcut to Set the Date.*

The example above used the PICK_DATE_FORMAT_MASK substitution string. This is a best practice method of keeping all dates consistent throughout an application.

This is not the extent of the usefulness of shortcuts. The sample in Figure 6.10 is created with a shortcut that is an elaborate PL/SQL program returning a string of HTML source code.

Files

The files in this shared component are stored in the HTML DB repository. From the repository, they can be referenced in HTML code as described below.

Cascading Style Sheets and Static Files from the information on JavaScript are covered in a later chapter of this book.

Images

Images can be stored in the HTML DB repository and rendered in an application page. The process and explanation of adding an image to the repository follows:

4. Navigate to the Shared Components page.
5. Click on the Images link.
6. Click on the Create button.
7. On the Create Image page:
   - Application: Select the application this image will be used in.

   There are two ways to add the image to the repository. It can either be associated with an application or with the workspace. Adding it to the workspace is done by selecting the No Application Associated option. If the image is added to the workspace, it can be referenced by all applications in the workspace. If it is added to a particular application, it can only be referenced by that application.

   - Upload New Image: Browse for the file name to upload.
   - Notes: Enter notes if desired.
   - Click Upload.
8. Upon completion, the program will return to the Images page showing a list of images in the repository.

Referencing Images

When images are referenced from the repository, it is done by using substitution strings. The two options are:

- #WORKSPACE_IMAGES#: Use this substitution string if the image is either associated with an application other than the developer's own or associated with the workspace using the No Application Associated option.
- #APP_IMAGES#: Use this substitution string if the image is loaded into the repository and associated with the current application.

Examples

The image can be rendered into a region of type PL/SQL by adding the following code for the region source:

```
htp.img( '#WORKSPACE_IMAGES#EasyHTMLDB.gif' );

htp.img( '#APP_IMAGES#EasyHTMLDB.gif' );
```

To use a loaded image as the logo for the application, it can be set in the Application Attributes page as shown in Figure 6.13.

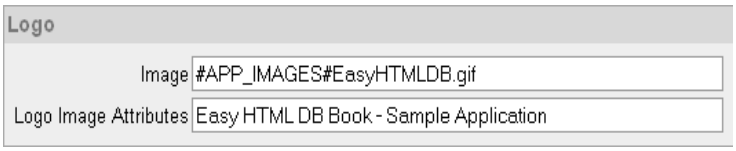

Figure 6.13: *Using substitution strings.*

There is additional information about working with images in a later section of this book.

Export/Import

Export/Import is a utility to export various pieces of an application. Using this utility, the following application pieces can be exported: an application; an application page; CSS files; Image Files; other Files (i.e. JavaScript); Themes; and, User Interface Defaults. The export/import utilities are used for backing up data or for deploying an application to another location. Application Deployment is covered in more detail later in this book and more information about export/import is included there.

Developer Toolbar

The developer toolbar is displayed at the bottom of web pages when a developer or administrator is logged into the HTML DB development environment. It provides quick links to several common tasks.

- Edit Application: Goes straight to the Application's home page described earlier in the chapter.
- Edit Page 1: The page id (1) will depend on which page you are on. This will take you to the Page Definition screen.
- Create: Runs the Create wizard allowing the developer to select whether they want to create a New Page, new Region on the current page, new Control on the current page, or a new Shared Component.
- Session: It is difficult to express enough how important this link is to an HTML DB developer. Clicking this link will display a popup window showing the current status session state. Session state will become very important as new developers dive into the world of developing HTML DB applications.
- Debug: This is another very important link. It causes the current web page to be re-posted with debug information shown inline. Some of the valuable information displayed is the

amount of time each step took to process while rendering the page. There is a section on debugging later in the book.

- Show Edit Links: This link will display a little icon next to each of the components on the page. Clicking on the icon will display a popup window allowing the developer to change the properties of the component without leaving the current page. This is a very handy productivity feature in HTML DB.

Page Zero

Page Zero is a special application page that must be deliberately created by the developer. It is not created by default when using the Create Application wizards.

The intended purpose of Page Zero is that any items created on Page Zero will be rendered on every other page of the application. The only restriction is Page Zero will only render the following types of components:

Regions

Buttons

Items

An example use for Page Zero would be for a banner, footer, or a sidebar table of contents the developer wants displayed on every page of their application.

In the HTML DB development environment Page Zero is used to display the main tab list you see at the top of every page in the Application Builder, SQL Workshop, and Administration applications.

Creating a Page Zero

9. From the application builder page click the Create Page button.
10. Choose Blank Page, click Next.
11. On the Page Attributes page enter 0 in the Page field, click Next.
12. On the Page Name page, enter Page Zero for Name, click Next.
13. On the Tabs page, select the No option, click Next.
14. Click Finish.

There, now the application will have a Page Zero. Any item placed on this page will be rendered on every other page in the application. Viewing the page definition of Page Zero will display a page as shown below.

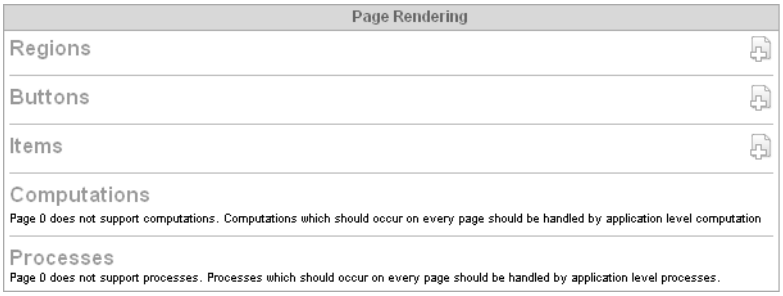

Figure 6.14: *Page Definition for a Page Zero application page.*

Even though figure 6.14 shows regions for defining Computations and Processes, they are not processed by the HTML DB engine. Also, any JavaScript placed in Page Zero attributes such as in the HTML Header area, are not processed by the HTML DB engine.

Easy HTML DB Book application

During the course of this book, there are several exercises that can be built and used for later reference. So far, the Conference RSVP application has been created and the following exercises will create an application named Easy HTML DB Book. This application will be used to build the pages in the exercises in the following chapters.

The code depot also includes an application named Easy Samples that is similar to the Easy HTML DB Book application. It has all the pages the author built while working on the book and documenting the exercises. This application can be installed and is a great source for looking at how to do perform a certain task in HTML DB.

1. Navigate to the Application Builder home page and click on the Create application link as shown in Figure 6.9.

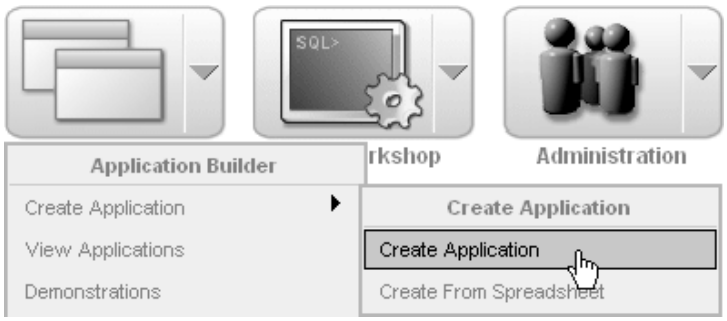

Figure 6.9: *Create a new Application.*

2. On the Name page:
   - Name: Easy HTML DB Book.
   - Application: 103 If ID 103 is not available, another ID should be used.
   - Create Application: Choose the From scratch option.

- Schema: User the EASYHDB schema already created. When using a workspace on the hosted Oracle site, use the schema available.
- Click Next.

3. On the Pages page:
   - Page Name: Enter Home as the page name.
   - Click the Add Page button. The application must have at least one application, and this is the starting point.
   - Click Next.
4. On the Tabs page, choose the No Tabs option and click Next.
5. On the Shared Components page, choose No and click Next.
6. On the Attributes page, leave the defaults and click Next.
7. On the User Interface page, choose Theme 2 and click Next. In truth, any theme can be chosen, but the figures used throughout the book are using Theme 2.
8. On the Confirm page click the Create button.

That completes the building of the Easy HTML DB Book application.

Conclusion

This chapter described many of the attributes that can be set for an application, application pages, and page level components. It should serve as a place of reference when it becomes necessary to understand what each attribute is providing.

The topic of Session State was also covered. The session state can be thought of as HTML DB's way of managing variables. The examples showed how to create the Easy HTML DB Book application that will be used in the exercises throughout the book.

The next chapter will begin with the most common type of application page used in HTML DB; the Report. An HTLM DB report is a graphical representation of an SQL Select statement and has many built in features.

and shows working examples of HTML DB database access from the simple web page population to the complex database transaction enforcing multi-user concurrency.

This book is unique because it describes all HTML DB Application Express concepts in an easy to understand language and contains step-by-step working examples of various HTML DB features. The HTML-DB Application Express book shows the when, where, why, and how of using each feature to build and deploy your web applications. It is designed for any Oracle professional including DBA's, application developers, Network Engineers and IT managers.