SkillBuilders Modal Page

A Dynamic Action Plug-in for Oracle Application Express (APEX)

Overview

Intro

Modal windows are very popular these days – and for good reason. They can help focus users' attention without disturbing the overall context of the user in the application. The SkillBuilders Modal Page plug-in was designed to display entire APEX pages as modal dialogs. Because the plug-in displays entire APEX pages, it is possible to utilize all of the standard page components within the dialog including processes, computations, dynamic actions, validations, and more.

The SkillBuilders Modal Page plug-in was designed to make creating modal dialogs as declarative and simple as possible. In some cases JavaScript methods have been created to provide programmatic control of the plug-in as well. The plug-in also adds several custom events, fired when the modal is opened or closed, to the Dynamic Action framework in APEX. These events allow other Dynamic Actions to respond in any number of ways, such as refreshing a report or showing a success message. Since version 2, the plug-in has been based on the popular ColorBox plug-in by Jack Moore.

Features at a Glance

1. Displays entire APEX pages in a modal window
2. Automatically closes modal via Success Message detection
3. Built in events allows other Dynamic Actions to respond to the modal

Version History

- 1.0.0 – 8/31/2011
  - Initial release
- 2.0.0 – 4/5/2012
  - Moved to ColorBox over jQuery UI Dialog for the modal display
  - New resize method added, see Advanced Usage > Methods
  - New events added, see Advanced Usage > Events

Feature Requests and Bugs

If you would like to see additional functionality added to the plug-in, or if you have found a bug, please let us know by emailing support@skillbuilders.com.

License

The SkillBuilders Modal Page plug-in is currently available for use in all personal or commercial projects under both MIT and GPL licenses. This means that you can choose the license that best suits your project and use it accordingly. Both licenses have been included with this software.

Legal Disclaimer

The program(s) and/or file(s) are supplied as is. The author disclaims all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any purpose. The author assumes no liability for damages, direct or consequential, which may result from the use of these program(s) and/or file(s).

Installation and Configuration

Installation

With this installation package there is a plug-in installation file named:

- dynamic_action_plugin_com_skillbuilders_modal_page.sql

Navigate to “Shared Components > Plug-ins” and click **Import >**. From there you can follow the menu to upload and install the plug-in using the file above. After installing the plug-in you will be redirected to the plug-in edit screen.

Optional Performance Upgrade

After installing the plug-in you can make it more performant by using files on the file server. If you wish to use the files on the apps server, simply copy the files in the server directory included with this plug-in to a directory on the apps server and change the File Prefix attribute of the plug-in to point to that directory.

Example:
If you copy the contents of "server" to **/images/plugins/apex_modal_2_0**
Then set the "File Prefix" attribute of the plug-in to **#IMAGE_PREFIX#plugins/apex_modal_2_0/**

Configuration

Once installed, this plug-in can be used as a native APEX component. When creating a new dynamic Action, select the “SkillBuilders Modal Page” option for the Action.

See Configuration Settings for details on how the application and component settings affect the plug-in.

Configuration Settings

Application Settings

Application settings are used to configure all instances of a plug-in within an application. These settings are accessed by editing the plug-in within the Shared Components. This plug-in has the following application settings:

Theme

The Theme setting is used to change the look and feel of the modal window. The ColorBox website includes 5 example themes which have been included with this plug-in. Simply select a number from 1 to 5 to use one of these themes. The themes demonstrate the flexibility of the plug-in and can be used

without making any changes. However, a custom option exists that, when selected, exposes two additional fields that allow for a custom CSS file to be defined. It is recommended that you start with an existing theme and make small changes to it until you are comfortable working with the CSS.

Custom CSS Path

The Custom CSS Path setting is used to specify the path to a custom CSS file for the theme. This setting is only displayed when the Theme is set to custom. See Optional Performance Upgrade for additional information.

Custom CSS Filename

The Custom CSS Filename setting is used to specify the name of the file that contains custom CSS for the theme. Only the name of the file should be included with this setting as it is assumed the extension will be ".css". This setting is only displayed when the Theme is set to custom. See Optional Performance Upgrade for additional information.

Transition

The Transition setting can be used to apply some effects to the opening and closing of the modal dialog.

Overlay Opacity

The Overlay Opacity setting is used to adjust the darkness of the overlay behind the modal dialog. Specify a number between 0 and 1 where 0 is transparent and 1 is completely obscured. The default value of .5 is in the middle and allows the users to see the screen behind but focuses their attention on the modal dialog.

Scrolling

The Scrolling setting is used to specify whether or not the modal can include scroll bars across the top and bottom of the modal to allow users to see all of the content in the page. When set to No, any overflow cotent will be hidden from view.

This setting must be used in conjunction with the Height and Width settings in the component settings. A later version of this plug-in will move this setting to the component settings.

Draggable

The Draggable setting is used to specify whether or not the user should be able to move the modal dialog around on the screen.

Initial Height

The Initial Height setting is used to specify the initial height (in pixels) of the modal page when it opens but before the main content loads.

Initial Width

The Initial Width setting is used to specify the initial width (in pixels) of the modal page when it opens but before the main content loads.

Component Settings

Component settings are used to configure an individual instance of a plug-in within an application. These settings are accessed by editing the component as you would a native APEX component. This plug-in has the following component settings:

Dialog Title

The Dialog Title setting is used to specify the title displayed in the modal page.

URL Location

The URL Location setting is used to specify whether the URL for the modal page is static or dynamic. Dynamic values are passed through an attribute of the triggering element.

Static URL

The Static URL setting is used to specify a static URL for the modal page.

Attribute Name

The Attribute Name setting is used to specify which attribute of the triggering element contains the URL for the modal page. This is often the href attribute of anchor elements.

Auto-close On Element Selector

The Auto-close On Element Selector setting is used to specify a jQuery selector used to close the modal page automatically. The selector is executed when the modal page is loaded. If the selector selects anything, the modal page will close and the Auto Close event will be triggered. Typically this is only used for modal pages that are submitted for processing. The success message is used to auto-close the modal page.

Dialog Height/Width Mode

The Dialog Height/Width Mode setting is used to control the size of the modal page. The default value of "Auto" will try to automatically determine the appropriate size of the page based on the content. However, if you need more control, this attribute allows you set the height and width as a percentage of the total window size or by a fixed number of pixels.

Height

The Height setting is used to set the height of the modal page. This value can be used to specify a percentage of the total screen or specific number of pixels which is determined by the Dialog Height/Width Mode setting.

Width

The Width setting is used to set the width of the modal page. This value can be used to specify a percentage of the total screen or specific number of pixels which is determined by the Dialog Height/Width Mode setting.

Modal Page ID

The Modal Page ID setting is used to associate an identifier, such as "create-customer-page", with the modal page that has been opened. When the modal page closes, this identifier will be passed back with the event object so that the closing of one modal page can be differentiated from another. The value can be accessed from the data object of "this" in a Dynamic Action with: **this.data.modalPageId**. If not all modal dialogs on a page have a Modal Page ID specified, you will first need to test if the modalPageId exists.

Advanced Usage

Methods

Resize

The resize method can be used to resize the modal dialog programmatically. The method accepts an optional options object that can be used to explicitly set any of 4 values: height, width, innerHeight, or innerWidth. To access the method from within the child page, see the third example.

```
$(document).apex_modal_page('resize');

$(document).apex_modal_page('resize', {height: 500, width: 600});

parent.$(parent.document).apex_modal_page('resize');
```

Close

The close method can be used to close the modal dialog programmatically. The method accepts an optional parameter which can be anything. That value can then be accessed in Dynamic Actions that respond to any of the close events of the plug-in. The syntax to access the value is: **this.data.modalPageCloseValue**. To access the method from within the child page, see the third example.

```
$(document).apex_modal_page('close');

$(document).apex_modal_page('close', someVariable);

parent.$(parent.document).apex_modal_page('close');
```

Events

Auto Close

The Auto Close event fires when the modal dialog is closed after the page has been submitted and the Auto Close element is found in the next page that loads. When the Auto Close event fires it exposes the Auto Close element to in Dynamic Actions as **this.data.$modalPageCloseObject**. The $modalPageCloseObject is a jQuery object which allows a great deal of flexibility. For example, the success message captured from the modal dialog could be displayed on the parent page with the following code (or something similar):

```
$('#messages')
   .hide()
   .empty()
   .append(this.data.$modalPageCloseObject)
   .slideDown('slow');
```

Another option would be to pass a JSON string as the success message so that it could be used more programmatically by the parent page. The following example shows how this could be done. Of course it would need to be adjusted depending on your success message template (found in the page template) and the embedded JSON string.

```
var successMessage = this.data.$modalPageCloseObject;
var custObjText = successMessage.find('div.uMessageText').text();
var custObj = $.parseJSON(custObjText);

$s('PX_PARENT_PAGE_ITEM', custObj.someProp);
```

Manual Close

The Manual Close event fires when the the modal dialog is closed by either the end user or programmatically by the close method. If the close method was used it's possible that an optional value was passed to the close method which is exposed in Dynamic Actions as **this.data.modalPageCloseValue**.

Close

The Close event fires whenever the modal dialog is closed, whether automatically or manually.

Start Open

The Start Open event fires when the modal dialog has started opening but has not yet loaded its contents.

End Open

The End Open event fires after the modal dialog has finished loading its contents.

About SkillBuilders and the Authors

About SkillBuilders

SkillBuilders is known for excellent IT training and consulting. Our instructors are always industry-tested experts and outstanding teachers who have set an unsurpassed standard of excellence. SkillBuilders' roots can be traced to 1985 when our founder, Dave Anderson, embarked on his career as an independent IT consultant, instructor and author. Dave and his colleagues built a small, energetic and growing company, headquartered in South Kingstown, RI.

About the Authors

Dan McGhan

Dan is a Senior Developer and Instructor with SkillBuilders. He suffers from Compulsive Programing Disorder which is believed to be linked to his balding. Having started his development career in the land of MySQL and PHP, he was only too happy to have stumbled upon APEX. Since then, he's dedicated his programming efforts to learning more about Oracle and web based technologies in general.

Dan is an Oracle Application Express Certified Expert, an Oracle PL/SQL Developer Certified Associate, as well as an Oracle ACE. In addition to his "day job", he is one of the top contributors to the APEX forum, maintains his own Oracle and APEX blog, and is a regular presenter at various events and user group meetings including ODTUG Kaleidoscope and APEXposed, the New York, New England, and Suncoast Oracle User Groups. His most recent addiction, as you may have guessed, is developing plug-ins for APEX.

When not programming, Dan may be found studying languages other than those used for development, notably Spanish. He's also been sited at various venues dancing Salsa with his wife, Sonia, and even enjoying an occasional cigar, a time when Sonia prefers not to be around.

Jason Lyle

Jason graduated in 2006 from Bowling Green State University with a degree in computer science. For him, coding is more of a puzzle to be solved rather than work to be done. He has dabbled in PHP, MySQL, Java, C++, TCL/TK, web technologies, PL/SQL (quite obviously), and anything else he can get his hands on. Right now, Jason is focusing most of his time on Oracle Databases and APEX applications.

Outside of the world of code, Jason enjoys trying almost anything. However, mostly he enjoys snow skiing, snowboarding, white water rafting, canoeing, kayaking, volleyball, cooking, reading sci-fi or

fantasy books, and traveling. In his international travels, he has only made it to a couple countries in South America, but hopes to keep increasing his list of places visited. He also hopes to learn to surf, water ski, and wake board.