OraFormsFaces™

Release Notes v3.1.5

COMMIT CONSULTING

October 2010

Authored by: Wilfred van der Deijl

OraFormsFaces Release Notes

Primary Author: Wilfred van der Deijl

Contents

Compatibility

Oracle Forms Compatibility

This matrix shows Oracle Forms versions that are compatible with OraFormsFaces versions.

	OraFormsFaces 3.1	OraFormsFaces 3.0	OraFormsFaces 2.3
Oracle Forms 6i (6.0.8.x)	If Needed note 1	If Needed note 1	If Needed note 1
Oracle Forms 9i (9.0.2.x)	If Needed note 1	If Needed note 1	If Needed note 1
Oracle Forms 10gR1 (9.0.4.x)	If Needed note 1,2	If Needed note 1,2	Yes note 2,4
Oracle Forms 10gR2 (10.1.2.0.2 with Focus Super Merge Patch)	Yes note 4,5	Yes note 4,5	Yes note 4
Oracle Forms 10gR2 (10.1.2.2+)	Yes note 4	Yes note 4	Yes note 4
Oracle Forms 11 (11.x)	Yes note 4	Yes note 4	Preliminary note 3

Note: Information in this matrix might undergo changes and you should always cross check this information with the latest information at http://www.commit-consulting.com/oraformsfaces/. Note that the release notes of a specific version can be updated after releasing that version.

Notes:

1. If support for this Forms version is needed it can most likely be added. OraFormsFaces does not ship with FMB, OLB, and PLL files for this Forms version but it is likely OraFormsFaces would just work if these files are back ported to this Forms version. Support for this Forms version could be added to OraFormsFaces upon customer request, but also see note 2.
2. Oracle has desupported this version of Oracle Forms. We strongly advise you to use a supported version of Oracle Forms and install the latest patch sets. If you do decide to use a desupported version of Oracle Forms, be sure to use the latest patch set to minimize the chances of running into Forms bugs.
3. Commit Consulting was intensively involved in Oracle Forms version 11 beta testing. As a result preliminary support for Forms 11 was added to OraFormsFaces. Changes in the final version of Oracle Forms 11 might require additional changes to OraFormsFaces.
4. You are advised to install the latest patchset for your Forms version and install the latest "Forms Focus Super Merge" patch to resolve any focus related issues. Check Oracle Support Note 730581.1 for the most up-to-date information on this Focus Super Merge patch for your platform and version. These patches can and should be applied both on your Oracle Application Server and your Oracle Developer Suite installations.
5. The base release 10.1.2.0.2 of Oracle Forms suffers from bug 4502472 and requires the installation of the 10.1.2.2.0 (or later) patch set or the installation of the latest "Forms Focus Super Merge" patch as described in the previous note.

Client Platform Support

The matrix below shows the client operating systems and web browsers that can be used with this version of OraFormsFaces. Be sure to check if the used operating system and web browser is also supported by Oracle to run Oracle Forms. You can get this information from the Oracle Client Platform Support statements at http://www.oracle.com/technology/products/forms/

Client Operating System	Browser	Supported
Windows	Internet Explorer 6.0	Not any more note 2
Windows	Internet Explorer 7.0, Internet Explorer 8.0	Yes
Windows	Firefox 3.0	Not any more note 2
Windows	Firefox 3.5, Firefox 3.6	Yes
Apple Mac OS X		Probably note 1
Linux	Firefox 3.0,	Not any more note 2
Linux	Firefox 3.5, Firefox 3.6	Yes

Notes:

1. OraFormsFaces is not tested with Apple Max OS X. However it is very likely that an application using OraFormsFaces will run on Apple Mac OS X as long as the configuration is capable of running Oracle Forms.
2. Support for this browser has ended or will soon be ended by its vendor. OraFormsFaces used to be compatible with this older browser, but we no longer test and certify with this old browser. It is likely OraFormsFaces will still work but if you encounter any issues we will ask you to reproduce them in a supported version of the browser.

The matrix below shows which Java Runtime Environment can be used on the client with OraFormsFaces and thus Oracle Forms. Be sure to check if the used JRE is also supported by Oracle to run Oracle Forms. You can get this information from the Oracle Client Platform Support statements at http://www.oracle.com/technology/products/forms/

Client Java Runtime Environment	Supported
Oracle JInitiator	No note 1
Sun JRE 1.4.2_06 or above	Yes note 2
Sun JRE 1.5.0_06 through 1.5.0_13	Yes note 2
Sun JRE 1.5.0_14 through 1.5.0_18	No note 2,3
Sun JRE 1.5.0_19 or above	Yes note 2
Sun JRE 1.6.0_04 or above	Yes

Notes:

1. OraFormsFaces uses a JRE feature called Legacy Lifecycle to only start the Oracle Forms applet once and re-use it on all pages. This is a feature that was introduced in Sun JRE 1.4.2 and which is not available in Oracle JInitiator (which is based on Sun JRE 1.3). Without this feature OraFormsFaces could probably gotten to work but it would start a fresh Oracle Forms applet on each page.
2. Sun has marked this Java release family as end-of-life (EOL). We strongly advise you to use the latest supported version of Sun JRE. If you do decide to use an EOL version of Sun JRE, at least be sure to use the latest patch release to minimize the chances of running into bugs and security issues.
3. A bug was introduced in Sun JRE 1.5.0_14 which makes applets fail to execute JavaScript when the applet is configured to be reused on subsequent web pages. OraFormsFaces relies on these two technologies and hence cannot run with Sun JRE 1.5.0_14 through 1.5.0_18. This issue has been filed as Sun bug 6603064 and has been fixed in Sun JRE 1.5.0_19.

Java Development Tools Support

This matrix shows which development tools can be used with OraFormsFaces to build JSF applications that use the OraFormsFaces components.

	OraFormsFaces 3.1	OraFormsFaces 3.0	OraFormsFaces 2.3
Oracle JDeveloper 10.1.2.x	No note 1	No note 1	No note 1
Oracle JDeveloper 10.1.3.x	Yes	Yes	Yes
Oracle JDeveloper 11.x	Yes note 5	Yes note 5	Preliminary note 2,3,5
Other IDEs	Probably note 4	Probably note 4	Probably note 4

Note: Information in this matrix might undergo changes and you should always cross check this information with the latest information at http://www.commit-consulting.com/oraformsfaces/. Note that the release notes of a specific version can be updated after releasing that version.

Notes:

1. OraFormsFaces uses the open standard JavaServer Faces technology. Oracle JDeveloper 10.1.2.x does not support JSF development but uses it predecessor Oracle UIX.
2. Due to issue FS#90 it is not possible to use OraFormsFaces in an ADF container that is added to the page using partial-page-rendering.
3. When using ADF Data Controls, you cannot drag attributes and methods from a data control and drop them as an OraFormsFaces component onto your page automatically creating the necessary bindings. You can still drag-and-drop the OraFormsFaces components from your component palette and bind them manually. This issue has been fixed in newer OraFormsFaces version.
4. Oracle JDeveloper is our primary development platform, since it is the most likely choice for OraFormsFaces and Oracle Forms users. Nonetheless, OraFormsFaces adheres to the industry JSF standard which means you should be able to use any JSF compliant Java development environment with OraFormsFaces. If you do encounter issues, please report them so we can see if they can be fixed.
5. Early ADF 11 versions caused popup elements like dialog boxes and pull down menu's to render behind the OraFormsFaces applet and not on top of it. This is Oracle ADF bug 7608635 which has been fixed in later ADF 11 versions. We have created a workaround for early ADF 11 versions. See OraFormsFaces issue FS#89 for a more detailed description and workaround.

Oracle JHeadstart Support

OraFormsFaces and Oracle JHeadstart can be used together to build hybrid applications. Oracle JHeadstart can be used to model and generate JSF applications and can generate pages that use OraFormsFaces components to include Oracle Forms. The matrix below shows which versions of JHeadstart are compatible with which versions of OraFormsFaces.

	OraFormsFaces 3.1	OraFormsFaces 3.0	OraFormsFaces 2.3
JHeadstart 10.1.3.2.x	Manual note 3	Manual note 3	Manual note 3
JHeadstart 10.1.3.3.81	Yes note 1	Yes note 1	Yes note 1
JHeadstart 11.x	Yes note 1	Yes note 1	Yes note 1

Notes:

1. This version of JHeadstart ships with the appropriate templates for using this version of OraFormsFaces.
2. You'll need to replace the OraFormsFaces template file that ships with JHeadstart with a template file appropriate for this version of OraFormsFaces. Most developers should be able to accomplish this themselves or they can contact us for an updated template file.
3. This could be made to work by changing some of the metadata files of JHeadstart. Newer versions of JHeadstart come with OraFormsFaces support out-of-the-box, but this can be added to previous versions as well without changes to binary JHeadstart files.

- Fixed bug with FormParameter component when the value was null
- Added support for having many FormParameters or FormParameters with long content. Previous version would throw an error if the internal JSON representation exceeded 2000 characters.
- Many enhancements to the JDeveloper extension:
  - A new installation wizard has been added to configure a Forms Server or Forms Developer Suite installation for the use of OraFormsFaces. This replaces the manual steps required in previous releases. The wizard is automatically invoked after installing OraFormsFaces and can be run from the JDeveloper Tools menu.
  - Added a documentation index page that is automatically shown after installing the OraFormsFaces JDeveloper extension and which can be accessed from the JDeveloper Help menu.
  - The documentation index page has links to open the release notes and developer's guide that are bundled with the JDeveloper extension
  - The documentation index page has links to open the Fusion Demo Application
  - An entry was added to the JDeveloper Help menu in JDeveloper 11g to open the Fusion Demo Application.

What is new in version 3.0.7

Version 3.0.7 contains no major new functionality. It merely fixes a number of issues with the 3.0 series of release candidates to make a final version:

- OraFormsFaces no longer tries to install its own JSON library if the browser has native JSON support. This prevents warnings or JavaScript errors in some browsers and can have a performance benefit.
- Fixed an issue where the loading image would not be removed if the user manually closed all forms windows in the previous usage of the applet instance.
- Fixed an issue when recycling an applet that has multiple forms open where one of the forms has a modal document window.
- Ensure older version of the JavaScript and/or CSS file are not cached by the browser when downgrading OraFormsFaces.
- Fixed an issue when recycling an applet that has a modal dialog window open when the user left or reloaded the page.
- The OraFormsFaces JSF JAR file, Forms JAR file and Forms PLL file now check if they are from the same version to prevent incompatibilities when only a partial upgrade has been performed.
- Improved applet startup time when using Internet Explorer with the next-gen JRE plugin architecture (available since 1.6.0_10) or when using Firefox 3.6.x.

What is new in version 3.0.5 (Release Candidate)

Version 3.0.5 is the third release candidate with a single fix for an incompatibility between OraFormsFaces' method for using a custom Java class to determine the Oracle Forms database credentials and Oracle ADF 11.1.1.2.

What is new in version 3.0.4 (Release Candidate)

Version 3.0.4 is the second release candidate with a number of fixes:

- The command line tools to attach libraries, subclass object groups or compile Oracle Forms files will now fail if attached libraries or subclassed objects cannot be found due to an incorrect FORMS_PATH setting. Previous versions would remove the attached library or subclassed objects without notice.

- The formsweb.cfg file for Oracle Forms 10gR2 no longer uses the "swan" color scheme as this undocumented color scheme was introduced with a patch set and is not available in the base 10.1.2.0.2 release.
- Redrawing the applet after initialization has been changed to prevent a series of network roundtrips between the client and the Forms server.
- Handling has been improved if offCustom.shouldFormClose returns false indicating no (more) forms should be closed or restarted upon reusing a suspended applet. Previous versions could sometimes keep the loading image up or show (ignorable) errors in the client Java console.

What is new in version 3.0.3 (Release Candidate)

Below are the fixed bugs and new features of OraFormsFaces version 3.0.3 which was released in December 2009. Previous 3.0.x versions were only technology preview releases and were not officially released for production use.

JSF Determining Forms Database Credentials

OraFormsFaces already supported a number of ways to specify the database credentials for the Oracle Forms database session. You could use fixed credentials (specified with the userid parameter in formsweb.cfg), Oracle Single Sign-On (where each user has its own credentials stored in Oracle Internet Directory) or prompting the user for her credentials.

With OraFormsFaces v3.0 an additional method is introduced that allows you to implement a simple Java class in the JSF application that determines the database credentials to be used. It can use the session state of the JSF application (with things like the logged in user) to determine the database credentials, for instance by looking them up in some secure store. The rest is handled by OraFormsFaces. It will encrypt these credentials before passing them on to the Forms server, where the OFF_LAND.FMX form of OraFormsFaces contains a trigger that decrypts this information and uses this to logon to the database. This gives you the possibility to use individual database accounts for all users without the need to setup Oracle Single Sign On.

Better Applet Recycling

OraFormsFaces tries to use a single Forms applet instance for all your web pages in a single session. This means the applet is suspended when leaving a page and is later re-used on a subsequent page. Newer browser and Java versions made significant changes to the way and applet is suspended and resumed. To ensure compatibility the mechanism in OraFormsFaces to detect applet suspends and resumes has been completely rewritten.

OraFormsFaces now also detects if your Forms application is using multiple forms (using CALL_FORM or OPEN_FORM). If detected, OraFormsFaces will exit all open forms in the correct order when resuming an applet on a subsequent page before opening the requested form. OraFormsFaces will now also detect any open modal dialog in Forms when the applet is resumed and will close this dialog automatically. It also prevents any pending messages from the previous applet instance to throw a modal dialog.

JavaScript API Enhancements

OraFormsFaces now uses the open source jQuery framework for all client side JavaScript code. The jQuery object is also available for your custom code through OraFormsFaces.jq. OraFormsFaces also introduces a JavaScript class model consisting of OraFormsFaces, OraFormsFacesForm, OraFormsFacesParameter and OraFormsFacesCommand classes representing the JSF components and their properties. The Developer's Guide has full documentation of this new API.

JSON (JavaScript Object Notation) is now used throughout OraFormsFaces. This includes Java classes at the JSF server, JavaScript objects at the client and a PL/SQL JSON package at the Forms server. This allows us and you

as a developer to pass complex objects and collections between these environments automatically handling special characters and Unicode encoding.

OraFormsFaces always had the option of raising a Forms event from JavaScript. You could pass an event name and payload and could implement your own handling in PL/SQL. With version 3.0 we also introduce the option to raise an event and wait for a value to be returned. In your custom PL/SQL event handling code you can return a value which is relayed to the calling JavaScript. This allows for things like querying the form-status for pending changes from JavaScript perhaps preventing navigation.

The Development Controls have been rewritten based on the jQuery framework allowing for much more enhanced controls.

PL/SQL Enhancements

The separate PL/SQL PLL library files have been merged into a single file to allow the different packages to call each other without getting circular references between the PLL files.

A new function offGlobal.isOraFormsFaces has been introduced. This returns true if running in an OraFormsFaces environment, otherwise false. This can be used if a single FMB/FMX file needs to be used both in a legacy Forms environment and in OraFormsFaces while needing slightly different functionality in both environments. This new function allows you to determine if the form is running in OraFormsFaces and act accordingly.

Java/ADF/JSF Enhancements

All JSF components are now compatibly with both ADF version 10 and ADF version 11.

The internal implementation of the JSF components itself has been changed so that all OraFormsFaces components now have proper getter and setter methods for all properties. This allows for programmatic manipulation of the components at runtime. All properties now fully support EL expressions and we have ensured they are compatible with JDK 1.4.2 for deployment to older application servers.

JDeveloper Integration

OraFormsFaces has now been bundled into an official JDeveloper extension. This eases the JDeveloper part of the installation especially for supporting drag-and-drop of ADF data controls as OraFormsFaces components into the JSF editor.

OraFormsFaces can be downloaded as a JDeveloper extension through our own site. It is the intention to get OraFormsFaces listed on the Open Source and Partners Extensions update center at OTN. This will make the installation even simpler since this update center is already configured in every JDeveloper installation.

Forms 11 Support

This version of OraFormsFaces introduces full support of Oracle Forms 11. It will leverage the native JavaScript API from Oracle Forms removing the need for a pluggable java component in each form. However, OraFormsFaces still needs a WHEN-CUSTOM-JAVASCRIPT-EVENT trigger in each form. The existing OraFormsFaces Object Library (OLB) has been changed to only contain this single trigger for Forms 11. This means that when you are upgrading to Oracle Forms 11, only a recompile of all your forms should be necessary as it will pick up on the newer and slimmer version of OraFormsFaces.olb at compile time.

Better Browser Integration

Sun introduced a new architecture for the Java Runtime Environment to plug into a browser. This new architecture became the default in JRE 1.6.0_10 and later. It is a radical change in how the JRE interacts with the

browser and required quite a bit of work in OraFormsFaces. The new architecture also introduced some new applet parameters which have been added to the Forms configuration files that ship with OraFormsFaces.

Internet Explorer 6 and 7 warned about non-encrypted content on an OraFormsFaces page when using an https connection. This message was wrong since it is triggered by an empty iframe on the page. This has been resolved an OraFormsFaces can now be used with an https connection without any warnings.

Older version of Internet Explorer and/or older version of Sun JRE also suffered from an issue where the applet would sometimes shift vertically misplacing the applet. This has been resolved by forcing the browser to repaint the applet when resuming from the legacy lifecycle cache.

OraFormsFaces now ensures a single applet instance is tied to a specific HTTP web session. If a user logs out of the web application and then logs in again, a new applet instance will be created as the state of the previous applet is likely linked to the state of the initial HTTP web session.

The OraFormsFaces JAR file is now signed with a code signing certificate to enhance security.

Disabling Parts of OraFormsFaces for Bug Hunting

OraFormsFaces includes many enhancements to the Oracle Forms applet itself and complex JavaScript interacting with the Oracle Forms applet. There's always a chance that these extensions or interactions cause issues or can trigger existing bugs in Oracle Forms. To diagnose these cases, many parameters have been added to disable parts of OraFormsFaces. The fine grained possibility to disable parts of OraFormsFaces offers a simple way to diagnose if OraFormsFaces is the cause of any occurring issues. These parameters might disable vital parts of OraFormsFaces and are only intended for diagnostics and are not intended to be used in a production environment.

and More...

Many more (smaller) improvements and bug fixes are included in this release of OraFormsFaces.

What is new in version 2.3.2

Below are the fixed bugs of OraFormsFaces version 2.3.2 which was released in February 2009 as a bugfix release.

Fixed Bugs

- FS#91 – Timed out ADF 11 session gives NullPointerException

What is new in version 2.3.1

Below are the fixed bugs of OraFormsFaces version 2.3.1 which was released in October 2008 as a bugfix release.

Fixed Bugs

- FS#83 – java.lang.NoSuchMethodError: com.commit_consulting.oraformsfaces.extension.FormsApplet.getJDKVersion

What is new in version 2.3

Below are the fixed bugs and new features of OraFormsFaces version 2.3 which was released in September 2008.

New Auto-Sizing and Auto-Clipping Features

In previous OraFormsFaces versions the page developer had to specify the width and height of the Forms applet. If you wanted to clip parts of the Forms applet (like the menu, toolbar, status bar, or others) you would have to

specify the amount of clipping in pixels as well. It could be a tedious task to find the correct values for these properties.

Version 2.3 of OraFormsFaces adds a new Auto-Sizing feature that no longer requires the page developer to set the width and height. Instead, the Forms applet will see how large the actual Oracle Forms window is and will adjust the applet size to it at runtime.

A similar feature has been introduced for clipping, where you can specify if you want to clip the menu, toolbar, status bar, window-title, or any combination. The OraFormsFaces applet will figure out the required number of pixels to clip and will do this automatically.

Added Loading Splash

When the Forms applet is starting up, the display could change or flicker a number of times. This could be distracting to the user and didn't look very professional. With the new auto-sizing and auto-clipping features this got even worse as the applet would resize and shift at runtime.

So, with this new version of OraFormsFaces the applet is covered with a white area and an animated image while the applet is starting up. Once the startup has completed, the animated image is removed and the final applet shows. This looks much more professional. OraFormsFaces ships with a number of predefined animated images, but you can also specify your own custom image.

Added preliminary support for Oracle Forms 11

Commit Consulting is closely involved in Oracle Forms 11 beta testing. As a result, OraFormsFaces has already been prepared for the new version of Oracle Forms. Forms 11 will have a native JavaScript API, something that older versions of Forms lack. OraFormsFaces has been adopted to use the Forms native JavaScript API in version 11 and continue to use the OraFormsFaces JavaScript API in previous Oracle Forms versions. This ensures a smooth transition from Oracle Forms 10g to Oracle Forms 11g.

Better Detection of Misconfigured or Unavailable Forms Server

OraFormsFaces now has better detection of a misconfigured or unavailable Forms server. The Forms Servlet URL has to be specified in the web application configuration and the Forms server also has to be configured for OraFormsFaces. This is error prone and configuration errors typically only resulted in an icon indicating a JavaScript error. That is not very informative or clear to an end user. The new version of OraFormsFaces detects if the Forms server cannot be reached but it also detects if the Forms server can be reached but is not configured properly to return the OraFormsFaces JavaScript. In both cases an alert dialog box will be shown to the user including the URL that should return a valid JavaScript. This way it is at least clear that the Forms startup failed and it is easy to check the URL in a browser to see why the request is failing or not returning a JavaScript.

Enhanced Development Controls

Previous versions of OraFormsFaces already featured Development Controls, but they have been greatly enhanced in this version. It is now possible to change all properties one can set at design time. Since you are changing the properties at runtime in an actual web browser, you instantly see the result. Once you are satisfied you can go to the design time in JDeveloper and set the component properties accordingly. This can save numerous roundtrips of changing the JSP page, running, and changing it again.

Check for Incompatible Sun JRE Versions

A bug exists in Sun JRE 1.5.0_14 and above that prevents and applet from executing JavaScript once that applet is being re-used on a subsequent page. This is exactly what OraFormsFaces relies on. Fixing a bug in Sun JRE is beyond our control, but this version of OraFormsFaces at least checks if you are using an incompatible version and will show an appropriate error. Previous version would just fail with unexplained error messages.

Improved Internet Explorer Stability

The integration between Microsoft Internet Explorer and Sun JRE is very fragile. A very large number of bugs, fixes, and workarounds exist. One of the issues OraFormsFaces users could run into was a freezing/hanging Internet Explorer when the Oracle Forms applet was first started in a session. The likelihood of this occurring has been drastically reduced by slightly delaying the applet startup until Internet Explorer has had a chance to render the rest of the page. The number of milliseconds for this delay is configurable as specific environments might benefit from higher/lower values. We hope the default value as used by OraFormsFaces will prove to be sufficient for most environments.

Increased Client Performance

A number of changes have been made to increase client side performance. The required JavaScript and CSS files have been drastically reduced in size. These files are now cached in the web browser client so they don't get requested again for each and every page as was the case in previous OraFormsFaces versions.

Moved Context Parameters to Environment Entries

In the past, the OraFormsFaces license key and Oracle Forms servlet URL had to be specified as context parameters in the web.xml file of your JSF project. With this new version, OraFormsFaces will try to retrieve these settings from Environment Entries first. These are also specified in the web.xml file but have the benefit that their values can be set or overwritten during or after deployment of the WAR/EAR file.

This means the application server administrator can set the Forms servlet URL which makes it possible to have testing, staging, and production versions of a web application pointing to different Forms servlet URLs without the need to create different WAR/EAR files for these environments. It also means you can keep the license key private and not include it in the source files and have the system administrator enter the license key at or after deployment.

Deployable to Oracle Application Server 10.1.2

OraFormsFaces had some unwanted dependencies on JDK 1.5 specific classes which prohibited OraFormsFaces application from being deployed on J2EE 1.4 application servers and Java containers such as Oracle Application Server 10.1.2. With this version of OraFormsFaces all code is now compatible with JDK 1.4 so OraFormsFaces applications can be deployed to Oracle Application Server 10.1.2 which could be the same server as used to run Oracle Forms. The OraFormsFaces Developer's Guide has detailed instructions on how to accomplish this.

Convenience Batch Files

Added three new time saving batch files to quickly subclass the OraFormsFaces object group to a large set of Forms in one go, attach the OraFormsFaces PLL library to a large set of Forms in one go, and compile a large set of Forms in one go.

Detailed JavaScript Tracing

All OraFormsFaces JavaScript is now instrumented to be able to provide detailed tracing information in combination with the Mozilla Firefox plug-in Firebug. See the Developer's Guide for instructions.

Added Support for Custom PL/SQL Events from JavaScript

Any event that is sent to Forms from JavaScript that is not automatically recognized and handled by OraFormsFaces is now forwarded to an empty stub procedure that can be changed by the developer. This allows for more advanced integrations where UI elements can send events that are specific to your situation and environment.

Enhanced License Key Handling

The handling of license keys has been improved. The actual license key is in a simpler format which makes it less error prone to install it. The key handling has some enhanced features as well.

JavaScript errors are caught. and printed

OraFormsFaces allows you to execute a snippet of JavaScript from within Forms PL/SQL. This is also heavily used by OraFormsFaces internally. When an error occurs during this JavaScript execution the error is now caught and printed to the client Java console.

Fixed Bugs

- FS#65 – JavaScript API breaks after legacy-lifecycle restore
- FS#66 – Installation instructions do not include OLB file
- FS#68 – CustomerDemo.fmb depends on obsolete off_tpl.fmb
- FS#69 – Context Root and Application Name for demo applications
- FS#70 – NullPointerException during applet startup on Linux client
- FS#71 – Invoke the JavaScript API automatically to prevent security delay on first user action
- FS#72 – Forms open in separate frame if default config has separateFrame=true
- FS#79 – FRM-40010 Cannot read form off_land.fmx. Some installations would not use the WorkingDirectory or FORMS_PATH if set specific for the OraFormsFaces config section in the formsweb.cfg resulting in an error about the off_land.fmx form not being found.

What is new in version 2.2

Below are the fixed bugs and new features of OraFormsFaces version 2.2 which was released in March 2008.

OraFormsFaces Object Library to replace the template form

In previous versions OraFormsFaces came with a template form off_tpl.fmb. This has been replaced with an object library file oraformsfaces.olb. This allowed for greater flexibility and better allows us to add new features to the object group in future releases.

The object group in the object library file now also contains a dummy window and canvas. This makes it easier to subclass the object group into one of your forms. In previous versions you would have to change the order of blocks and set the canvas property of the OraFormsFaces data block. With the object group from the object library file this is no longer necessary. You can just drag-and-drop the object group from the object library to your form and don't need to make any changes after that.

Added support for Oracle Forms 10gR1 (9.0.4)

Support for Oracle Forms 10gR1 has been added to this release. The OraFormsFaces ZIP file now comes with files for both Oracle Forms 10gR2 (10.1.2) and Forms 10gR1 (9.0.4). Files for Oracle Forms 9i can be made available upon request. Oracle Forms 6i files are harder to create and are currently not available. If you have a demand for Forms 6i files, please contact us and we can see if it is possible to downgrade the existing files to Forms 6i for you.

Added compatibility with Oracle 10gR2 patch set 3 (10.1.2.3)

Oracle made a change to one of the internal java methods of the Forms applet in 10gR2 patch set 3 which broke OraFormsFaces.

Upgrading Instructions

In general, upgrading is the same as doing a fresh install. You can follow the steps from the Developer's Guide for doing a fresh install. However, there might be some version specific notes which are explained later in this section. In general you need to consider the following remarks when doing a fresh install as part of an upgrade:

- If you do not intend to use the previous version of OraFormsFaces anymore, you can remove the library definitions from your JDeveloper installation. Go to the "Tools" menu, then "Manage Libraries". On the "Libraries" tab remove OraFormsFaces and on the "JSP Tag Libraries" remove OraFormsFaces as well.
- When modifying the faces_creator_configuration.xml file as described in "Add Drag-and-Drop Support for Data Controls in JDeveloper", you'll need to replace the information from the previous OraFormsFaces install with the information from the new version.

Upgrading a JDeveloper Workspace and Project

After completing the installation steps for your new OraFormsFaces version, you'll need to upgrade your JDeveloper workspaces and projects as well. To make sure these workspaces use the new OraFormsFaces version, follow these steps:

- Right-click the ViewController project and select Properties
- Go the the "JSP Tag Libraries" node of the navigation tree. Then remove OraFormsFaces from the list of JSP Tag Libraries. JDeveloper probably asks if the OraFormsFaces library should also be removed from the project. Select "Yes".
- If JDeveloper did not ask to remove the library, navigate to the "Libraries" node of the navigation tree and remove the OraFormsFaces library manually.
- Close the Project Properties dialog box by pressing OK. This finalizes the new settings and actually removes the library from the project.
- Again, right click the ViewController project and select Properties
- Go to the "JSP Tag Libraries" node of the navigation tree.
- Click the Add button and add the new version of the OraFormsFaces tag libraries to the project.
- Press OK to close the dialog box. This will make the changes final and also add the OraFormsFaces java library to the project.

Upgrading Oracle Forms

After installing a new version of OraFormsFaces, be sure to recompile all of your forms. Your forms will have a subclassed object group from the OraFormsFaces object library. With each new version of OraFormsFaces, this object library might change. By recompiling your Forms you ensure that these changes are also embedded in the compiled FMX files.

Upgrade Notes when upgrading from any version to 3.1.x

Some previous OraFormsFaces versions required manual deinstallation during upgrades. With version 3.1 a new installation wizard has been introduced that handles this for you. See the Developer's Guide for detailed instructions and for alternative manual instructions.

As with any OraFormsFaces upgrade you also have to upgrade your existing JSF projects and recompile all your Forms modules to incorporate changes from the OraFormsFaces Object Library. See the Developer's Guide for instructions on how to compile all FMB files in one action.

Upgrade Notes when upgrading 2.3.x to 3.0.x

OraFormsFaces 3.0.x is the first version that is installed as a JDeveloper Plug-In. This means that all manual registrations of previous versions have to be removed before installing version 3.0.x. See the installation instructions in the Developer's Guide that include instructions for de-installing older versions.

As with any OraFormsFaces upgrade you also have to recompile all your Forms modules to incorporate changes from the OraFormsFaces Object Library. See the Developer's Guide for instructions on how to compile all FMB files in one action.

Upgrade Notes when upgrading 2.3.1 to 2.3.2

There are no specific instructions for upgrading OraFormsFaces version 2.3.1 to version 2.3.2, other than the remarks at the beginning of this chapter. If you are upgrading from an earlier version of OraFormsFaces to version 2.3.2, be sure to read the remarks for all versions between your old and new version.

Upgrade Notes when upgrading 2.3 to 2.3.1

There are no specific instructions for upgrading OraFormsFaces version 2.3 to version 2.3.1, other than the remarks at the beginning of this chapter. If you are upgrading from an earlier version of OraFormsFaces to version 2.3.1, be sure to read the remarks for all versions between your old and new version.

Upgrade Notes when upgrading 2.2 to 2.3

There are no specific instructions for upgrading OraFormsFaces version 2.2 to version 2.3, other than the remarks at the beginning of this chapter. If you are upgrading from an earlier version of OraFormsFaces to version 2.3, be sure to read the remarks for all versions between your old and new version.

Upgrade Notes when upgrading 2.1 to 2.2

Besides the normal upgrade instructions as explained before, there are some specific remarks to consider when upgrading to OraFormsFaces 2.2:

- Prior to OraFormsFaces version 2.2, OraFormsFaces used a template form to subclass some objects to your forms. In version 2.2 this has been replaced with an object library. This means you will have to go over all your forms and remove the subclassed objects from the template form. Next, follow the steps from "Preparing the Oracle Forms modules" in the Developer's Guide to subclass the objects from the object library.
  We apologize for the inconvenience this may cause you as you need to go over all your forms. Subclassing the object group from an object library is more powerful and the new object group is easier to use. It includes a dummy window, so there is no need for you to set the canvas property of the OraFormsFaces data block after subclassing it to your form.