Taming FRM-99999 and Other Forms Server R.6 Issues

Peter Koletzke, Millennia Vision
Howard Fujimoto, Millennia Vision

Deploying your Developer forms on the Web is essentially pretty simple. All you need to do is to recompile the forms and set up the JInitiator plug-in. However, there are web-specific processes required and these are not quite as simple. The Internet Application Server (formerly called Oracle Application Server) listener and Forms Server require extra care and feeding and the path to make these work can be a rocky one. When working with this technology, it is useful to have a set of proven techniques for the installation, configuration, and trouble-shooting of the Forms Server.

A common error you might receive when something goes wrong with the web-deployed Forms environment is the generic FRM-99999. To solve that problem, you first need to be certain that you have an absolutely solid installation of the application server. The very first step you need to go through to tame this problem and all other Forms Server problems is actually a proactive one—ensuring that you have followed the proper steps for installation.

Since this is so important, this paper devotes much space to discussing the essential setup steps. Even if you have properly set up the servers, it is possible that you will obtain error messages and encounter other issues. Therefore, this paper also gives solutions for variations on the FRM-99999 error and other issues.for Forms Server Release 6.0.5 and 6.0.8 (6i).

The "New OAS"

Many of the techniques discussed in this paper were developed on a production project in an OAS (Oracle Application Server) v.4.0.8 and Forms 6.0 (patch 4a) environment. The Internet Application Server (IAS) currently fills the position of application server technology, a post formerly held by OAS. The day after this paper was accepted for presentation at Oracle OpenWorld 2000, the product direction change was announced. The authors felt that presenting their experiences and information on the "old" technology was still useful and appropriate because not all Oracle shops move to the new versions immediately and OAS 4.0.8 with Forms Server 6 will be a viable combination for the near term in many situations.

OAS and IAS and both are different products at the core—Spyglass for OAS and Apache for IAS. Therefore, while IAS and OAS are similar conceptually, you can adjust the methods discussed here to fit the IAS platform if that is your environment. In addition, it is possible to run the Forms Server with non-Oracle listeners. However, that method cannot take advantage of the Forms Server cartridge implementation and this discussion is out of scope for this paper. The paper will concentrate on the cartridge implementation because it produces a more flexible system. One example of this flexibility is that, by using the cartridge implementation, you can specify replaceable parameters on the command line that will alter how the Forms Server acts.

This paper will discuss the Forms Server configuration parameters, peaceful coexistence of the Forms Server and Oracle Portal(now called Oracle Portal), techniques for dynamic and static parameters you can pass into your forms, and the most commonly occurring errors with their solutions.

Setup Notes

Many of the setup steps for the Forms Server are performed on OAS. If OAS is not properly installed and configured for the Forms Server, even a correct configuration of the Forms Server will be to no avail and your system will fail. The discussion assumes that you have successfully installed the OAS software on an application server machine and that the administrative listener (usually port 8888 on UNIX and 9999 on NT) has been started. (The examples in this paper use UNIX.) It also assumes that the C Web cartridge that is used for the Forms Server is installed in OAS.

Most of the configuration steps occur in the OAS administrator's browser application. You load this application using a URL that contains the host name and administrator port, for example: http://myhost:8888. A login dialog will pop up where you enter the administrator's name and password (as they were defined during installation).

Creating Listeners

You will have to create a listener to service the Forms Server. If you have an existing (non-administrative) listener that is used for other application purposes (for example, the PL/SQL Toolkit cartridge), that listener can service the Forms Server as well. Since each listener takes extra resources on the application server and requires setup and administration, you want to limit the number of listeners that you create but, at the same time, provide enough of them to allow debugging and "bouncing" (starting and stopping) without affecting all applications. Our situation used the same listener for both the PL/SQL Toolkit cartridge and the Forms Server.

In the Admin page for your site, drill down to the HTTP Listener node, highlight it and click the Add button (+). The dialog in Figure 1 will appear where you select the node (machine) that the listener will be defined for.

Figure 1: Create listener dialog

The next dialog that appears when you click "Apply" is shown in Figure 2.

Figure 2. Listener node dialog

The only things you fill out here are the listener name (traditionally six characters or less) and the listener port (any free port, for example 8050). Clicking "Submit" will create the listener. You still have to start it by clicking on its name in the HTTP Listeners node and clicking the Start icon (right arrow) in the toolbar.

Note: Any time you change a listener setting be certain that the listener is stopped by clicking the Stop icon (red box) in the toolbar. If you do not stop the listener, you may create orphaned processes that will prevent OAS from being shut down. The underlying reason for this is that the name of the Listener PID file (defined in the Network node under the listener in the Admin page) will be truncated by one or more characters if you apply changes and the listener is running. Shutting down the listener prevents this from occurring.

Set Up the Directories

Once you have set up the listener for the Forms Server, you need to enter the directory names that are associated with the required virtual paths. Open the listener node in the Admin navigator and click on the Directory node as shown in Figure 3.

Figure 3. Click on the Directory node.
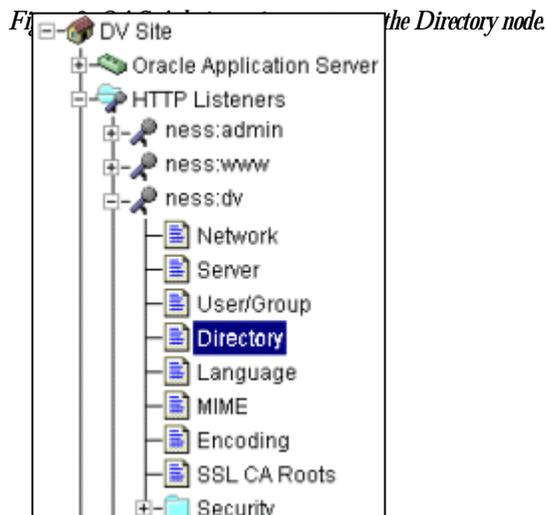
Tip: When you navigate to any listener node in the Admin page for the first time, the Admin login dialog may appear. Just enter the username and password that you used to log into the Admin page.

The right side of the screen will display the directories and their associated virtual directories. A virtual directory is used by the application server to locate files in its file system. This area is for defining the mappings of real operating system directories to virtual directories. Entries for the listener that you will use for the Forms Server will be defined as the samples in Table 1.

File-System Directory	Flag	Virtual Directory	Description
/oracle/dbs/OAS/ows/4.0/doc/	NR	/	Default directory used for the OAS documentation files.
/oracle/dbs/dev60/oracle/product/	NR	/forms60code/ (6.0) /forms60java/ (6i)	The location of the Java applet files that are downloaded to the client.
/home/appuser/	NR	/reg/	The location of the registry.dat or default.dat files.
/tmp/	NR	/tmpdir/ (6.0) /dev60temp/ (6i)	A directory used to write temporary files. This directory needs to be accessible in read and write mode to the user account that is running the Forms Server.
/home/appuser/webhtml/prod/	NR	/webhtml/ (6.0) /dev60html/ (6i)	The location of the startup HTML file.
/home/appuser/webicons/	NR	/webicons/	The location of the icon (GIF) files used for buttons in the form and the splash screen graphic.
/oracle/dbs/dev60/oracle/product/	NR	/webjars/	The location of the Java Archive files that are downloaded to the client.
/home/appuser/webtemp/	NR	/webtemp/	The location of the temporary compressed graphics (JPEG) files that the Forms Server creates when it needs to render images on the form.
/oracle/dbs/jinit/	NR	/jinitiator/	Location of the JInitiator executable for downloading to the client.
/oracle/dbs/tools/web60/temp/	NR	/dev60cgi/	Location of the CGI executables

Table 1. Sample directory settings for a listener

In the examples above, ORACLE_HOME is set to /oracle/dbs. The actual directory names will vary depending on your installation, but the virtual directory names may be used as represented in this table.

Note: The virtual directory names may be somewhat different for Forms R.6i. Consult the Forms online help system for the actual virtual directory names if you are using R.6i.

If specifying particular usage for the physical files, use the *Flag* column as follows: the first character in this field specifies whether CGI programs may run from the specified virtual directory. Possible values follow:

- **N** (No CGI) - Does not allow CGI programs to run from this directory.
- **C** (CGI) - Allows CGI programs to run from this directory.

- **S** (Servlets) – All OS Java Servlets to run from this directory.

- **W** (WinCGI) - Allows WinCGI programs to run from this directory. Used for Win NT only.

The second character specifies whether subdirectories of the specified file-system directory should be mapped recursively—that is, whether the directory tree rooted at the specified file-system directory should be accessible through the specified virtual directory. Possible values follow:

- **R** (Recursive) - Map subdirectories recursively.

- **N** (Non-recursive) - Do not map subdirectories recursively.

If you do not want to make the subdirectories of the mapped file-system directory accessible to clients, set the directory mapping to non-recursive.

Caution: All of the entries above must start and end with a forward slash (/). This is required by the environment to preserve it concatenation and lookup consistency. On NT, this is replace with a backward slash (\). In either case it is required, before and after each entry. Failure to provide the trailing slash on the virtual directory results in some very obscure errors at variable replacement time.

Adding an Application

Once you have configured the listener, you can add an OAS application to the site. An OAS application is associated with a particular cartridge such as the C Web cartridge used for the Forms Server. To create an application, select the Applications node in the Admin navigator and click the Add icon in the toolbar. You will see the dialog in Figure 4.

Figure 4. Adding an application

After selecting the Application Type as C Web, you click the Apply button. Another dialog will appear with the following fields:

- **Application Name** - This is the single-word name you will use within the OAS admin. The user will not see this name, but you should use a name that designates the use, for example: Web_Forms_Prod.

- **Display Name** - This is the name that will be displayed in the OAS Admin screens. This can be multiple words, for example: Production Web Forms.

- **Application Version** - Enter 1.0 or some other number that is meaningful to your operation.

Clicking Apply in this dialog will (hopefully) display the Success page. At this point, you have a choice to add a cartridge or stop. Click the button labeled "Add Cartridge to this Application." The cartridge identifies a part of the URL that the user will use to reach the application.

Adding a Cartridge

Since you selected the C Web cartridge type in the application set up, the Add C Web Cartridge dialog will pop up as Figure 5 shows.

Figure 5. Add C Web Cartridge dialog

Fill in the fields as follows:

- **Cartridge Name** - The name you use to identify the cartridge internally to OAS. For example, Web_Forms_Prod_Cart.
- **Display Name** - The name that will display in the OAS Admin screens, such as Production Web Forms Cartridge.
- **Cartridge Shared Object** - This is the name of the library that contains the Forms executable. On UNIX, the value will be something like: /oracle/dbs/dev60/oracle/product/6.0/lib/f60webc.so
- **Cartridge Entry Point** - The name of the function in that executable: form_entry.
- **Virtual Path** - The name of the directory that will be used in the URL to identify this cartridge in the application, for example, prodforms.

Cartridge Parameters

Once you have set up the cartridge, you can enter the specific cartridge parameters required for your application. One of the features of the cartridge implementation of the Forms Server is the ability to pass parameters into the form using the URL. This ability requires a special formatting of the startup HTML file that is the file used to start the first form in the session. Flexible parameter passing through the startup HTML file is diagrammed in Figure 6.
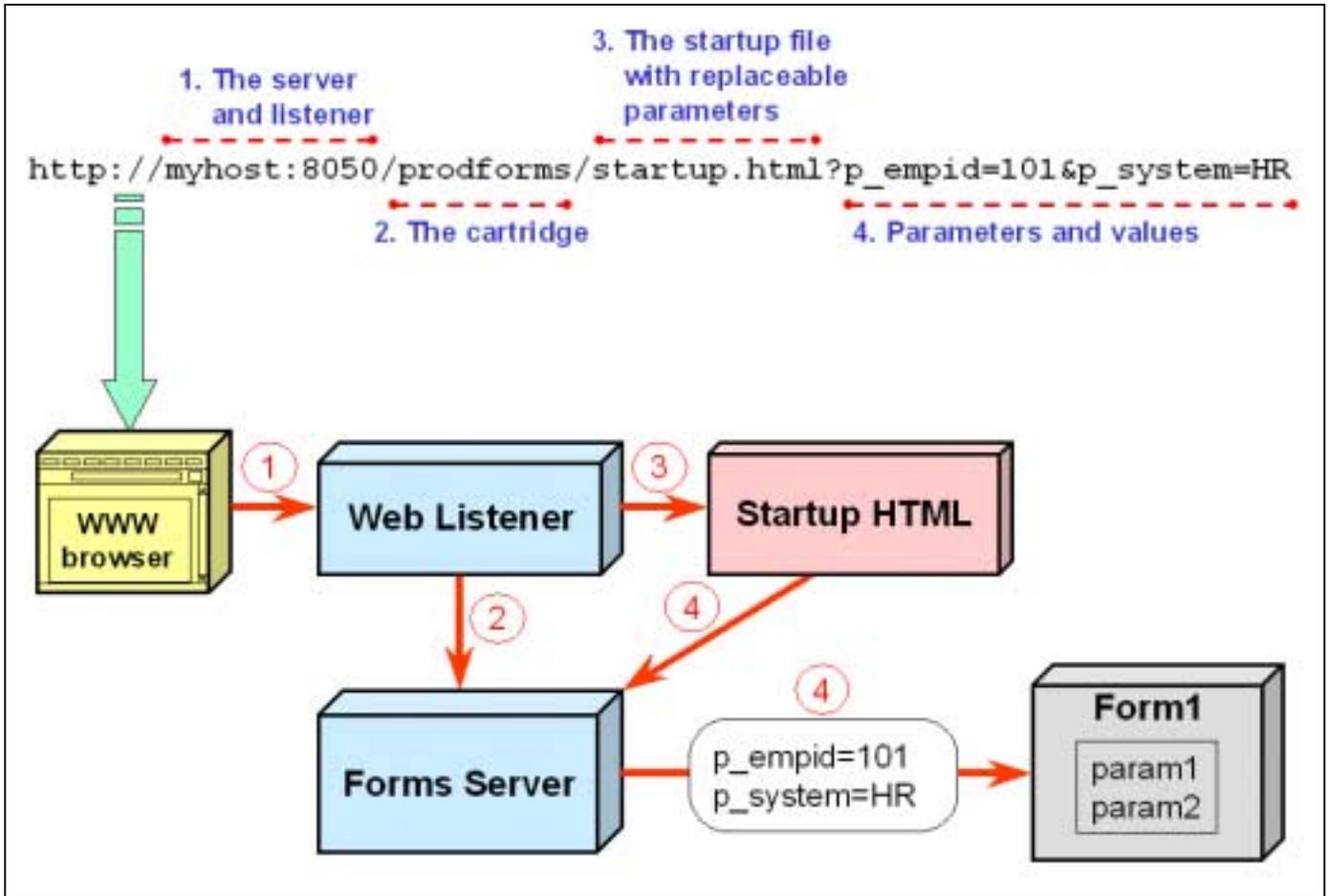
Figure 6. Passing parameters from the URL to a form using the Forms Server cartridge

The parameters need to be set up in the OAS Cartridge Parameters node in the Cartridges - Configuration node under a specific application and cartridge within that application. This navigation path is shown in Figure 7.
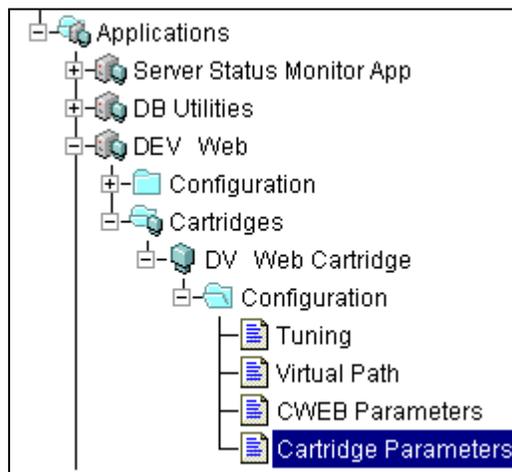

Figure 7. Cartridge Parameters node

Once you click on the Cartridges Parameters node, a screen will appear on the right where you enter and modify the parameters and default values. Table 2 lists some sample parameters and their values. Appendix A shows a sample startup HTML that contains some of the replaceable parameters used in this table.

Parameter	Sample Value	Notes
code	oracle.forms.engine.Main	Entry point for starting the middle-tier runform environment. Value must be as specified as shown here.
codebase	/forms60code/	Virtual path location of the runform environment code.
baseHTML	/home/appuser/webhtml/prod/espStart.htm	Path and name of the startup HTML file, if this parameter is specified, then invocation of the application can be made with just the virtual path. The baseHTML file will be used by default.
HTMLdelimiter	$	Character which denotes variables in the HTML file. This value must be $, the % value specified in the OAS examples does not work in this UNIX environment.
archive	/webjars/f60all.jar	Location of the jarfiles, specified with the virtual pathname.
width	800	Screen width for the initial MDI window.
height	600	Screen height for the initial MDI window.
serverApp	default	Specified with value "default", other values are available, reference the Oracle documentation for more information.
serverPort	8050	The port that the Forms server is listening, reference
module	app_logon login_db=prod	Name of the form to be executed, the .fmx suffix is optional. Any extra parameters that are required may be concatenated at the end of this parameter value using the name=value syntax. Alternatively, you can set up separate parameters explicitly in this area.
separateFrame	false	Indicates whether or not the form will be displayed in the current browser window or a separate frame.
pluginspage	http://myhost:8050/webhtml/jinit_download.html	Browser page that allows the user to download Jinitiator, required plugin to run webforms.
lookAndFeel	generic	Allows for the switching between a generic and Oracle look and feel (the value is "Oracle" in that case). Colors, icons, etc will be manipulated by this parameter's value.

Parameter	Sample Value	Notes
apptype	application/x-jinit-applet;version=1.1.7.18	Entry for the browser to recognize the correct mime type.
jinitloc	http://myhost.company.com:8050/webtemp/jinit11718.exe#Version=1,1,7,18	Physical location of the Jinitiator executable, in URL format. Requested downloads of the program will occur from this location. Be sure you have specified the entire host name (myhost) and domain name (company.com) in the URL.
classid	clsid:020F6116-407B-11D3-A3BB-00C04FA32518	Class Id value for this version of Jinitiator. At startup time, this value is interrogated which allows the user to be informed of newer versions of the software. Reference the Windows registry for more information.
title	My Application on the Web	The "MDI" window title that will appear in the browser window's title bar.
bg		Specifies whether a background is to be applied.
colorScheme	teal	The color scheme that is used when the look and feel is set to the Oracle style.

Table 2. Cartridge parameters

Note: Parameter names are case sensitive and must match exactly what has been specified in the HTML file.

Replaceable Parameters

This startup HTML file (see Appendix A for a sample) contains a number of replaceable parameters designated with the "$" characters which are set up in OAS as described above. A replaceable parameter is one for which the user supplies a value on the URL "command line." If the user does not specify a value for the parameter, the default value that is set up in the OAS Cartridge Parameters node is used. This concept allows flexible passing of nearly everything such as the name of the form that will be started and the width, height, and any other parameter you need to set up. See the online Forms documentation for a list of other parameters that the Forms Server can use.

When moving a system from development to production, you need to examine the list of replaceable parameters carefully because users should not be able to change anything that will compromise security of your application or cause unexpected side effects.

When the URL contains the replaceable parameter (as shown in Figure 6) and its value, the Forms Server processes the value if the parameter is a meaningful one (such as height and width). If the parameter is not meaningful to the Forms Server, it will be passed to the form. In this case, there must be a parameter set up in the Form Builder navigator for the incoming parameter value.

Forms 6i Startup HTML File

The Forms 6i install loads two types of files that are used when starting up the first form. These are loaded into the forms60/admin/server directory. The first type of file is the startup file which has two manifestations. BASE.HTM (used for appletviewer or native browser applets) and BASEJINI.HTM (used when JInitiator is used to display the form) contain values that affect the startup, some of which are specified as replaceable parameters. These files work the same way as the STARTUP.HTML file described above, but they work with the second type of file, the configuration file, to fill in the values of the various startup parameters that can be used in this environment. The configuration file, called FORMSWEB.CFG, contains the same type of parameters. Values that are assigned to the parameters in this file are used for the replaceable parameters in the startup files.

The sample HTM files that are used should be complete enough that you would not need to change them. Oracle recommends that you make changes to the parameter values in the configuration file. This will allow you to keep different configuration files for different applications and customize the parameters for each application.

There is more information on the contents and workings of these files in Chapter 5: Configuring the Forms Server of the document called *Deploying Forms Applications to the Web with Forms Server* document (Help-Manuals in the Form Builder).

Environment Variables

Before you start up the Forms Server process, you need to set some environment variables to define where the process should look for certain files. Table 3 lists the most common variables. Set these in a login script (.profile or its equivalent for your particular shell) for a UNIX server or in the registry for an NT server.

Variable	Description
FORMS60_MAPPING	The webtemp directory specified in the virtual directories screen described above. This is a virtual directory, not a physical directory, so the value will be something like "/webtemp/".
FORMS60_OUTPUT	The webtemp directory specified in the virtual directories screen described above. This is an actual directory name, such as "/home/appuser/temp".
FORMS60_REPFORMAT	If Forms is calling Reports, this specifies the output style of the report, for example, HTML.
FORMS60_TIMEOUT	The length of time in minutes until timeout occurs between 3 and 1440 (1 day).
UI_ICON	This is a list of directories (delimited by a semi-colon on NT and a colon on UNIX) that designates where the Forms Server will find the icon files used for buttons, menus, and windows. Use a value such as /home/appuser/webicons:$UI_ICON
FORMS60_PATH	The location of the .FMX Forms executable files. This is also a path (list of directories). Each application should have its own directory and different Forms path. This means that there should be a different listener for each application because there is a different Forms Server process running with the different path.

Table 3. Environment variables

More on REGISTRY.DAT

This file (located in the ORACLE_HOME/forms60/java/oracle/forms/registry directory) supplies some Forms Server parameters some of which are used by the runtime and some of which are not. The location of icons, behavior of the Oracle look and feel mode, and graphics types are all found in this file.

You can copy this file and customize it for a particular application. In the startup HTML file, supply a line such as "<PARAM NAME="serverApp" VALUE="/appfile/myapp">" where "myapp" refers to a copy of REGISTRY.DAT called MYAPP.DAT and "appfile" refers to a virtual directory that you have already mapped to a real directory in the directories screen of OAS. The specified .DAT file will be used to override similar settings found in the REGISTRY.DAT file. This allows you to have a common REGISTRY.DAT file for the default settings to serve a number of applications and still have overrides available through the additional .DAT file.

The URL

Once you have defined the application and the cartridge, you have set the complete virtual path that will be used to call the first form. The URL that the user uses to connect to the startup HTML file will be something like the following:

```
http://ness.com:8050/prodforms/prodstart.html
```

In this example, ness.com:8050 indicate the host machine and listener port number. prodforms indicates the application name and prodstart.html is the startup HTML file. It is possible to eliminate the HTML file name by defining it as the starting point for the cartridge. You can do this by entering the name of the HTML file in the OAS Admin configuration screen for Network (under the specific listener's node); the field is called "Initial File."

Forms Server Setup Wizard

Although you can also use the Forms Server setup wizard to configure the environment, using the wizard does not give you an idea of which properties and values are being set. You need to install the Forms Server on the application server from the install media.

SQL*Net/Net8

You also have to install SQL*Net or Net8 (depending on the database version) on your application server so you can connect to the database. This is a standard net install and you will be able to test whether it works by using the tnsping utility from the command line. Be sure to install under the user account that you use to start the Forms Server. This will set up the proper environment variables so that this account can connect to the database. The TNSNAMES.ORA file should contain a listing for the database you need to access from Forms. This is also a Net8 install task. If you are taking advantage of the Oracle Name Server, the TNSNAMES file will not require specific entries for the database, but you will need the SQLNET.ORA file set up to make requests to the names resolution process.

Starting and Stopping the Forms Server

You can start or stop the Forms Server from the command line using ifsrv60 (on NT) or f60ctl (on UNIX). Be sure to set any environment variables that are required (see discussion above) before starting the Forms Server. A sample call might be:

```
f60ctl start port=9000 pool=1
```

The port parameter indicates the port number that the Forms Server listener will use. While this can be any free port, the convention is to use a number in the 9000s. The pool parameter specifies that the Forms Server will maintain one active spare connection so it can accept a request from a user without the overhead of starting a new connection. You can tune this for your particular application needs.

Starting or stopping the OAS listeners is done from the online Admin interface by clicking on a listener node and clicking the Stop or Start icon, respectively. If all listeners need to be affected, you can select All on the HTTP Listeners node and click Start or Stop icons. If the entire OAS session needs to be restarted, you have to go to the command line and issue the following:

```
owsctl stop
```
then

```
owsctl start
```
With Forms 6i, the integration of Oracle Enterprise Manager (OEM) with the Forms Server means that you can start and stop the Forms Server listener using OEM. *Deploying Forms Applications to the Web with Forms Server,* in the online documentation, addresses this integration in Chapter 13.

Oracle Portal Setup

Oracle Portal can be set up to use the C Web cartridge of OAS and can co-exist with v.2.2. You install a listener and C Web application in the OAS Admin screens in the same way as you do for the Forms cartridge. An additional step that is required is to create a Database Access Descriptor (DAD) using the Oracle Portal site maintenance screens. It is important to check that there is an additional environment variable set for the Oracle Portal DAD configuration file websvr.app. This file is located in the WEBDB_HOME/listener/cfg directory. The WV_GATEWAY_CFG environment variable should contain the entire path and filename for this file if Oracle Portalis to work properly.

> **Note**: We experienced a situation where the wdbsvr.app file is not correctly updated. The workaround is to "hand" edit the file. The site will not work correctly without the environment variable being set and correctly edited. The following is a sample of what the DAD definition should look like in this file.
>
> ```
> [DAD_app]
> connect_string   =  PRODDB
> password   =  APP_public
> username   =  APP_public
> default_page   =  home_page.html
> document_table   =  APP.wwv_document
> document_path   =  docs
> document_proc   =  APP.wwv_testdoc.process_download
> ;name_prefix   =
> ;always_describe =
> ;after_proc =
> ;befor_proc =
> reuse   =  Yes
> connmax   =  4
> ```

Troubleshooting WebDB

One of the main troubleshooting techniques for Oracle Portalis to examine the log files for error messages. There is a file called wrb.log in the ORAWEB_HOME/admin/SITE_NAME/log directory (where SITE_NAME is the name of your site). This directory also contains a number of log files with names such as wdbctx_9999.log (where 9999 is a unique number). Another file, xlf.log, shows the HTTP requests that are made to the server and this may prove useful in troubleshooting.

Troubleshooting OAS and the Forms Server

Even though you may follow the recommended guidelines, the FRM-99999 error message might pop up. This is the most common error because it is a generic one that has a number of meanings. This section examines some of the known problems and their workarounds or solutions.

Java Console

You can turn on a message box that displays the output of various steps the Forms Server and Java applet go through when loading a form. When the Java Console is turned on, you will see a message window as in Figure 8.
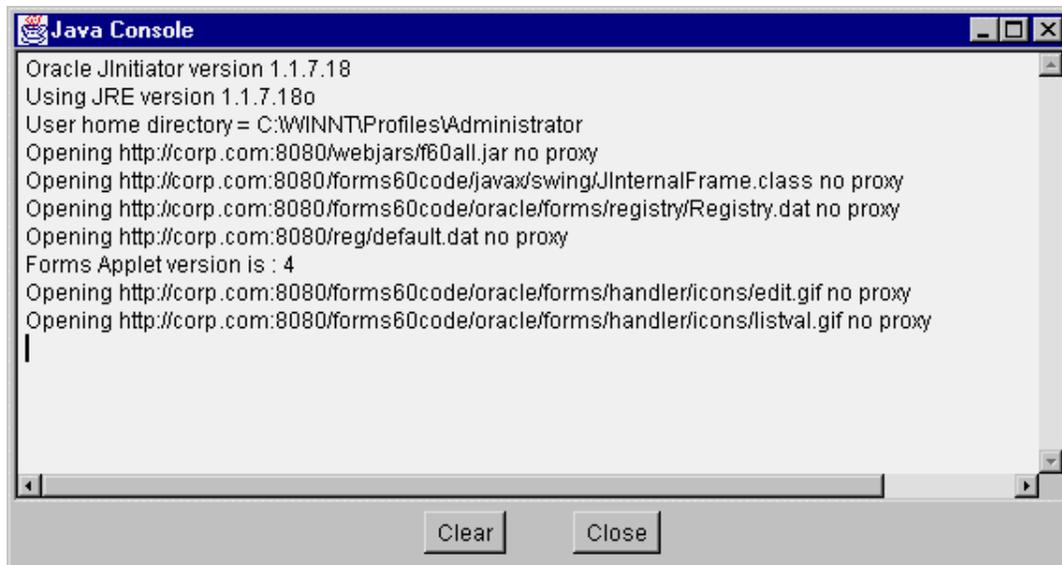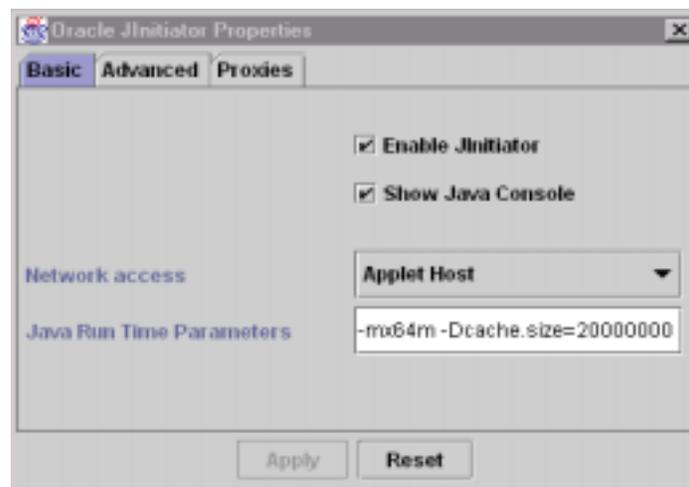
Figure 8. The Java Console

You can turn this console on using the "Oracle JInitiator Control Panel" application that is installed into your Windows start menu when you install JInitiator. Running this application displays the dialog as Figure 9 shows.

Figure 9. The JInitiator Control Panel

Checking the checkbox for "Show Java Console" will display the console when you start the next form from the browser. While you will still have to do some digging to determine the cause of the problem, this message window can supply error message numbers to check and a sense of what is working and what is failing.

Miscellaneous Techniques

The following is a compiled list of problems encountered and their solutions. Some problems relate to OAS and were discovered while setting up the Forms Server. This list of techniques is in addition to the other tips mentioned throughout this paper.

A PL/SQL Toolkit Problem

- After the initial installation of OAS, the PLSQL Toolkit installer was not functioning correctly. The behavior exhibited was that the pop-up window would appear, but remain blank. Further examination of the problem showed that a spawned process on UNIX had become an orphaned runaway (discovered using ps –ef | grep OAS). The UNIX owner of OAS must kill this process. The problem was isolated to a missing environment

variable for TNS_ADMIN and the binary location of srvmgr was missing from the PATH concatenation. These items need to be confirmed as a post-installation step from the original software install.

OAS Admin Manager Help

- As with most screens presented in the right frame, there is a ? icon in the upper right hand corner. Pressing the icon supplies context-sensitive help for the fields on the frame.

OAS Manager -Applications: PL/SQL

- The state of an application can be misleading, the green and red flags indicate active connections versus no active connections. This is different than other screens which denote a started versus not started state. Also note that the green arrow start button is missing from the top of the frame. It is not displayed because applications cannot be started.

- The Virtual Path entry will be used in the URL and must not have a trailing slash (/).

OAS Manager - Applications: C Web

- The Virtual Path entry will be used in the URL and must not have a trailing slash (/).

- Oracle Portaldoes currently not use cartridge parameters.

Troubleshooting Tips

In general, OAS is fairly trouble free. On those occasions where clients are having difficulty there are number of items that can be checked to determine the problem. The first task is to determine if the problem is in any of the components of the application, this would include: OAS, Dev6 Forms Server, Oracle database instance, and the hardware. If there are error messages being reported, this is the place to start.

Tips for Network Testing

- Use the TCP/IP ping utility to check the health of the network, using the following command:

  ```
  ping -s myhost.corp.com 1472 -5
  ```
  The -s specifies that a large packet size will be tested; 5 indicates that the test will be made five times; 1472 indicates the size of the tested data transmission. The output from this command will indicate how well the network is transmitting data.

- Use the utility netstat to determine how the network is set up. Issuing a command line command for netstat -I, you will see a list of various measurements. If the collisions (shown in the Collis column) are high (greater than 1 to 2% of the total packets reported in the Opkts column), the network is busy. If the packets queued (in the Queue column) are not zero, you have data that cannot be transmitted. Both conditions may require extra work and help from the network administrator to resolve.

Tips for OAS

- Access the OAS Manager and determine the state of the primary OAS objects. Do this by expanding the site and then highlighting "Oracle Application Servers", "HTTP Listeners" and "Applications" in turn. Identify required processes in a stopped state (red flag) through this action. As stated in this document, the "red flag" state is not necessarily a problem for objects such as applications, but for other objects it is.

- Use the Monitor icon (magnifying glass) to determine object usage. Through this built-in tool, object usage can be seen, dynamically. This will indicate whether new connections are being made.

- Determine the workload on the backend server, by using any of the tools available. This would include SQL*Plus on the instance and top or iostat on the UNIX box. These will show you which processes are active and consuming memory or a large portion of CPU time.

- If the problem is determined to be OAS, then reload the administrator (using the command line) and all site services (using the OAS Admin manager).

Tips for the Forms Server

- Access the appropriate UNIX box and determine the state of the Forms listener. Do this by issuing "ps –ef | grep f60" from the UNIX command line. If the listener port is not running then start it using the procedures outlined above. Otherwise, stop and start the listener and Forms Server if the process seems to be hung.

- With Forms 6i, the FRM-99999 error now contains more details on the exact problem. For example, if the error is FRM-99999, Error 1412, you have programmatically tried to set the position of the scrollbar on a block that has no scrollbar (using the SET_BLOCK_PROPERTY built-in). If you tried to get the scrollbar position information from a block that had no scrollbar, you would receive an error 1413 with the 99999. The release notes (relnotef.pdf) contains a more extensive list of error messages that are associated with FRM-99999.

- If you have problems with the Forms Server finding the toolbar icons, check the directory listing for webicons as mentioned in the OAS setup notes above. Also, check the icons directory listed in the registry.dat (in the FORMS60 directory on the server) or default.dat (also located on the application server). There should be an entry for "default.icons.iconpath=/webhtml/icons/" or something similar.

- To test whether the JAR files are being cached, enter the following parameter in the Java Console runtime parameters field:

  ```
  -Dcache.verbose
  ```

  This will add messages that are specific to the Forms Server to the Java Console window (described above). If you want to capture the output of this process to a file, use the following parameter in the Java Comsole:

  ```
  -Dcache.logfile mylog.txt
  ```

  This will output the results of the Java Console into mylog.txt. In the Advanced tab of the control panel, you can check the checkbox for "Enable Debug" to get additional messages.

- Other logging facilities can be activated by providing a parameter to the Forms Server processes when you start them on the command line. The parameter is "log=*logfile*" where *logfile* is the name of the file that you want to have messages stored in. This technique can be used for the server (f60svrm or ifsrv60) and the control program (f60ctl).

- There may be stack trace files generated on the client machine when an error occurs. Look for files with an .RPT extension. The names of those files will indicate which program caused the error (for example netscape.rpt means that the Netscape browser had an error).

- Broken images (images not rendering properly) may be due to the FORMS60_MAPPING environment variable not being set properly. This is mentioned in the Environment Variables section above, but it is important that you use the proper slashes ("/" for a virtual directory) and place them on both sides of the virtual directories.

Conclusion

Taming the FRM-99999 and other problems is largely a matter of properly setting up the OAS and Forms Servers. While the steps may seem to be extensive, you only need to think about them once. When troubleshooting a problem with the Forms Server you want to first look at the messages that the Java Console provides to see if it leads you towards the actual problem. In addition, you need to be able to stop and start all web components—OAS listeners and Forms listener. With these techniques as well as those in the documents mentioned in the Extra Reading section below, you should be able to successfully set up and run your own web-deployed Forms environment. Best of Luck!

About the Authors

Peter Koletzke is a Technical Director and Principal Instructor for Millennia Vision. Peter is Executive Vice President and Web Initiatives Director for the International Oracle Users Group-Americas and is a frequent speaker at various Oracle users group conferences where he has won numerous awards such as Pinnacle Publishing's Technical Excellence, ODTUG Editor's Choice, and ECO/SEOUC Oracle Designer Award. Peter co-authored, with

Dr. Paul Dorsey, the Oracle Press books *Oracle Designer Handbook, Second Edition*, and *Oracle Developer Advanced Forms and Reports*. http://ourworld.compuserve.com/homepages/Peter_Koletzke

Howard Fujimoto is a Senior Technical Director and Chief Architect for Millennia Vision. With over 20 years of IT experience, Howard now specializes in Oracle database server, application server, and technology architecture. He is a contributor to the Oracle Press book *Oracle Developer Advanced Forms and Reports*.

Millennia Vision, Redwood Shores, CA, a Full Service Provider (FSP), that delivers e-business solutions in e-time designed for dot-com, Fortune 1000 and high-growth companies. With over five years proven experience, Millennia Vision provides a single source for Business Modeling, Integrated e-Services, ASP and Strategic Business Process Outsourcing. http://www.mvsn.com

Other Reading

Note: the web locations of the Oracle white papers may change over time. If they are not available in the locations indicated, you can navigate to the Products page and select Developer or Forms Server. This will display a list of available documents and you should be able to match the titles below with an existing document.

Oracle Forms Server Troubleshooting, An Oracle Technical White Paper, March 2000, found on TechNet, http://technet.oracle.com/products/forms/pdf/275198.pdf

Developer Server Troubleshooting Tips, Oracle bulletin, January 2000, on TechNet; http://technet.oracle.com/products/developer/pdf/developertips.pdf

Oracle Forms Server Frequently Asked Questions, Technical White Paper, on TechNet, http://technet.oracle.com/products/forms/pdf/256628.pdf

Deploying Forms Applications to the Web with Forms Server, Chapter 15: Troubleshooting Solutions and Chapter 5: Configuring the Forms Server, Forms online documentation

Forms Server and Forms Runtime Logging, Metalink document 62664.1, by Duncan Mills, Oracle Corp. Search MetaLink for this file number

Forms Troubleshooting and Diagnostic Techniques, Nick Triggs, Oracle Corp., Published in ODTUG 2000 Conference Proceedings (www.odtug.com)

Tips for Configuring Oracle Developer Server for Forms 6 and 6i Under WebDB, Gene Schneider, ODTUG Technical Journal, June 2000 issue, www.odtug.com (members-only area)

Appendix A: Startup HTML Files and Parameter Passing

The following is a sample startup HTML file that you can use as a sample. See the Forms online documentation for other examples.

```
<HTML>
  <HEAD>
    <TITLE>$title$</TITLE>
  </HEAD>
  <BODY>
    <OBJECT classid="$classid$"
          WIDTH=$width$
          HEIGHT=$height$
          codebase="$jinitloc$">
    <PARAM NAME="CODE"          VALUE="oracle.forms.engine.Main">
    <PARAM NAME="CODEBASE"      VALUE="/forms60code/">
    <PARAM NAME="ARCHIVE"       VALUE="/webjars/f60all.jar">
    <PARAM NAME="type"          VALUE="$apptype$">
    <PARAM NAME="serverPort"    VALUE="9000">
    <PARAM NAME="serverArgs"    VALUE="Module=app_logon login_db=prod">
    <PARAM NAME="splashScreen"  VALUE="/webicons/co_logo.gif">
    <PARAM NAME="lookAndFeel"   VALUE="$lookAndFeel$">
    <PARAM NAME="colorScheme"   VALUE="$colorScheme$">
    <PARAM NAME="background"    VALUE="$bg$">
    <PARAM NAME="registryPath"  VALUE="/reg/">
    <PARAM NAME="separateFrame" VALUE="$separateFrame$">
    <PARAM NAME="serverApp"     VALUE="default">
    <COMMENT>
      <EMBED type="$apptype$"
            java_CODE="oracle.forms.engine.Main"
            java_CODEBASE="/forms60code/"
            java_ARCHIVE="/webjars/f60all.jar"
            WIDTH=$width$
            HEIGHT=$height$
            serverPort="9000"
            serverArgs="Module=esp_logon login_db=prod"
            splashScreen="/webicons/co_logo.gif"
            lookAndFeel="$lookAndFeel$"
            colorScheme="$colorScheme$"
            background="$bg$"
            separateFrame="$separateFrame$"
            serverApp="default"
            registryPath="/reg/"
            pluginspage="$pluginspage$">
    <NOEMBED>
    </COMMENT>
    </NOEMBED>
    </EMBED>
  </OBJECT>
  </BODY>
</HTML>
```

The file contains two sections, each of which has similar parameters. The first section, starting with "<PARAM NAME="CODE"," is used by a Netscape browser on the client. The second section, starting with "<EMBED type="$apptype$"," is used by an Internet Explorer browser. Each type of browser ignores the instructions for the other so that one startup file can service both browser types.