Troubleshooting the Forms Listener Servlet

An Oracle White Paper
September 2006

ORACLE

Troubleshooting the Forms Listener Servlet

INTRODUCTION

Purpose

The purpose of this document is to give an insight into how the Forms Listener Servlet can aid in the diagnosis and detection of various anomalies and errors commonly seen in a web deployed Oracle Application Server 10*g* Forms application. The paper discusses which files contain diagnostic information about the Forms Listener Servlet and how to effectively interpret this information.

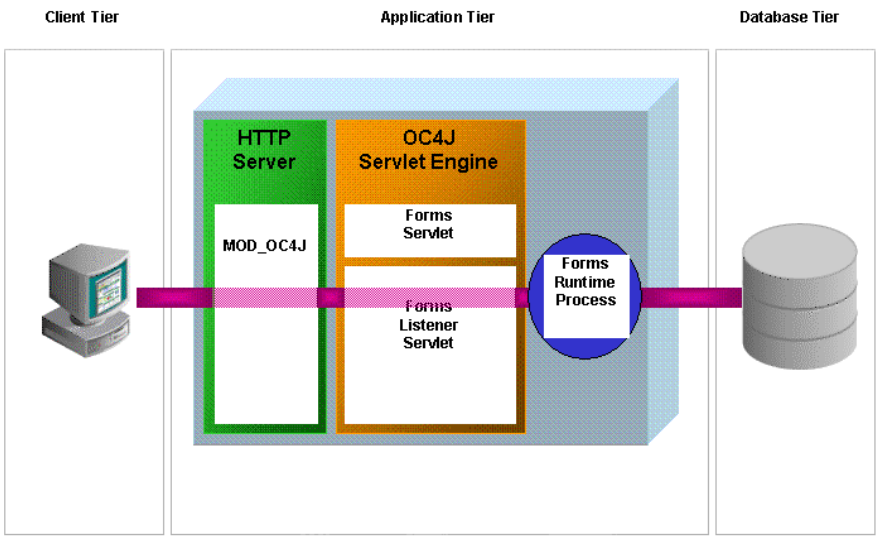
Audience

This document is intended for anyone involved in diagnosing and resolving problems with Oracle Forms10*g*, particularly OracleAS Forms Services (a component of Oracle Application Server 10g). However, it is important to have a good understanding of OracleAS Forms Services architecture and general web deployment processes to derive maximum benefit from the information in this paper.

WHAT IS THE FORMS LISTENER SERVLET?

The Forms Listener Servlet plays a pivotal role in a web deployed Forms application by:

- Creating a dedicated Forms runtime process for each Forms client requesting a Forms session
- Receiving HTTP Get/Post requests from the Forms client, routing these requests to the Forms runtime process and sending the responses back to the client
- Terminating the Forms runtime process when the user exits the Forms application or browser.



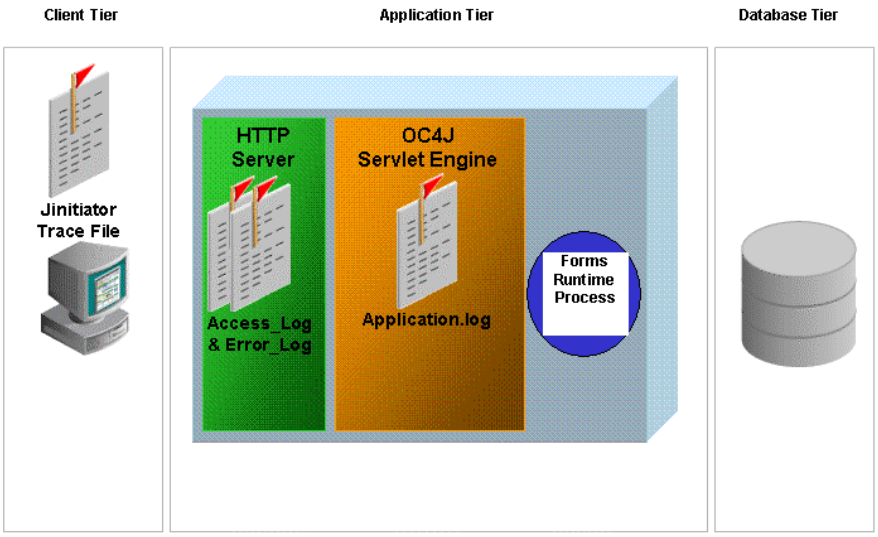
Why do we need to troubleshoot the Forms Listener Servlet?

Depending on the cause of an error, it can be difficult for Forms to determine and report precise error information. For example, the loss of a network connection between the client and the application server would of course mean that the server, if it was aware of the error, is unable to transmit this information to the client (and visa versa). Thus, a web deployed Forms application may throw a number of more generic catchall errors like FRM-92100/FRM-92101

So, it can be a challenge to detect the source of such generic errors and disconnections. Was the cause of the fault with the Forms client or the network or the database or even a bug in the application code? Diagnosing the Forms Listener Servlet can aid in the resolution of such problems and hence it is very useful to understand how to troubleshoot it.

LOG FILES CONTAINING DIAGNOSTIC LISTENER SERVLET INFORMATION

To run a Forms application on the web requires a number of different components to initialize, run and communicate. Each of these components will be separate processes and potentially on different physical locations. As such, trace and diagnostic information can be distributed across a number of different places.



Jinitiator Trace File

Open Jinitiator Control panel by selecting the Windows Start menu -> Settings -> Control Panel -> Jinitiator<version_number>, and then Select "Java Runtime Parameters" under the "Basic" tab.

The first place to look for trace information may be on the client machine. Jinitiator is a Java Virtual Machine (JVM) responsible for running the Forms Java client. Setting the following Java runtime parameters in the Jinitiator Control Panel will enable Jinitiator tracing.

-Djavaplugin.trace=true

-Djavaplugin.trace.option=basic|net|security|ext|liveconnect.

This will produce a trace file, Jinitiator<version>.trace in the User Home Directory. For example, if the User Home Directory is C:\Documents and Settings\username and Jinitiator version1.3.1.9 is used, the file C:\Documents and Settings\username\jinitiator1319.trace will be produced.

Oracle HTTP Server Access_log and Error_log

When a Forms application is running on the Web, the HTTP Server is responsible for the transmission of metadata, between the Forms client and the Forms runtime, as standard HTTP(S) messages.

The HTTP Server is enabled by default to log basic information about all HTTP requests to the access_log file and report errors in the error_log file. It is possible

to increase the logging level on the HTTP Server using the LogLevel directive in httpd.conf. However, the default basic logging information is sufficient to troubleshoot the Forms Listener Servlet and detailed logging is not required

Oracle HTTP Server Access_log and Error_log are found in the following directory on the application server:

<ORACLE_HOME>/Apache/Apache/logs

OC4J_BI_Forms Application.log

Finally, the Forms Listener Servlet is responsible for marshalling the communication between the individual Forms clients and their corresponding Forms runtime processes.

Basic diagnostic information for this component is found in the application.log (or jserv.log in Forms 6i) when the Forms Listener Servlet is used in the default mode. Detailed diagnostic information can be obtained from the application.log by enabling debugging in the Forms Listener Servlet as follows:

- appending the /debug option to serverURL in the formsweb.cfg file

  Eg: serverURL=/forms90/l90servlet/debug

- appending the /debug option to serverURL directly in the URL or use the debug config section already provided in the formsweb.cfg file

  Eg:

http://<server>:port/f90servlet?serverURL=forms90/l90servlet/debug

http://<server>:port/f90servlet?config=debug

Depending on the version of Forms being used, the application.log is found in the following locations:

Oracle 9iAS Rel 1 Forms Services (Forms 6i (6.0.8))

[ORACLE_HOME]/Apache/Jserv/logs/jserv.log

Oracle 9iAS Rel 2 (9.0.2) and Oracle Application Server 10g (9.0.4)

[ORACLE_HOME]/j2ee/OC4J_BI_Forms/application-deployments/forms90app/OC4J_BI_Forms_default_island_[n]/application.log

Oracle 9iDS Rel 2 (9.0.2)

[ORACLE_HOME]/j2ee/Oracle9iDS/application-deployments/forms/application.log

Oracle Developer Suite 10g (9.0.4)

[ORACLE_HOME]/j2ee/DevSuite/application-deployments/forms/application.log

INTERPRETING AND DIAGNOSING INFORMATION IN THE LOG FILES

Of course, once you have identified the log files, you need to be able to interpret and understand the information as a tool to problem resolution.

Jinitiator Trace File

Jinitiator is responsible for executing the Java applet, which renders the Forms user interface (UI), and the communication between the client and the application server.

The Jinitiator trace file contains the following information:

- Success of finding and executing Forms applet classes for UI interaction and communication.
- Useful Java stack dumps, primarily when the error originates from the client.
- The JsessionID, which uniquely identifies a Forms session. The Forms Listener Servlet uses two session tracking mechanisms:
    - Cookies, where the Servlet container sends a cookie to the client. The client returns the cookie to the server upon each HTTP request, thereby associating the session with the cookie.
    - URL rewriting, where the Servlet container appends a session ID to the URL path, for example: http://host[:port]/forms90/l90servlet;jsessionid=a23445bcde89

The Jinitiator trace below shows a typical Java stack trace, where an AppletSecurityException has been raised because the Forms applet has accessed a resource on the client and has violated the security restrictions imposed by the Jinitiator JVM.

```
sun.applet.AppletSecurityException: checkwrite

at sun.applet.AppletSecurity.checkWrite(Compiled Code)

at java.io.FileOutputStream.<init>(FileOutputStream.java:55)
```

```
at java.io.FileWriter.<init>(FileWriter.java:35)
at Test.writeFile(Test.java:18)
at Test.setProperty(Compiled Code)
at
oracle.forms.handler.ComponentItem.setCustomProperty(Compiled
Code)
at oracle.forms.handler.ComponentItem.onUpdate(Compiled Code)
at oracle.forms.handler.JavaContainer.onUpdate(Compiled Code)
at oracle.forms.handler.UICommon.onUpdate(Compiled Code)
at oracle.forms.engine.Runform.onUpdateHandler(Compiled Code)
at oracle.forms.engine.Runform.processMessage(Compiled Code)
at oracle.forms.engine.Runform.processSet(Compiled Code)
at oracle.forms.engine.Runform.onMessageReal(Compiled Code)
at oracle.forms.engine.Runform.onMessage(Compiled Code)
at oracle.forms.engine.Runform.processEventEnd(Compiled Code)
at
oracle.ewt.event.AnyEventMulticaster.processEventEnd(Compiled
Code)
at oracle.ewt.lwAWT.LWComponent.redispatchEvent(Compiled Code)
at oracle.ewt.lwAWT.LWComponent.processEvent(Compiled Code)
at java.awt.Component.dispatchEventImpl(Compiled Code)
at java.awt.Container.dispatchEventImpl(Compiled Code)
at java.awt.Component.dispatchEvent(Compiled Code)
at java.awt.LightweightDispatcher.retargetMouseEvent(Compiled
Code)
at java.awt.LightweightDispatcher.processMouseEvent(Compiled
Code)
at java.awt.LightweightDispatcher.dispatchEvent(Compiled Code)
at java.awt.Container.dispatchEventImpl(Compiled Code)
at java.awt.Component.dispatchEvent(Compiled Code)
at java.awt.EventDispatchThread.run(Compiled Code)
```

The Jinitiator trace below shows the JsessionID and information for the successful loading of jar files.

```
Oracle JInitiator: Version 1.3.1.17

Using JRE version 1.3.1.17-internal Java HotSpot(TM) Client VM

User home directory = C:\Documents and Settings\kprakash

Proxy Configuration: Manual ConfigurationProxy: www-
proxy..com:80Proxy Overrides:
a.us.oracle.com,b.uk.oracle.com,*.oracle.com,*.ora.com,*.oracle
.com,*.oraclel.com,<local>

JAR cache enabled
```

```
     Location: C:\Documents and Settings\kprakash\Oracle Jar
Cache

     Maximum size: 50 MB

     Compression level: 0----------------------------------------
--------------

c:   clear console window

f:   finalize objects on finalization queue

g:   garbage collect

h:   display this help message

l:   dump classloader list

m:   print memory usage

q:   hide console

s:   dump system properties

t:   dump thread list

x:   clear classloader cache

0-5: set trace level to <n>

----------------------------------------------------
```

The first three entries show the loading for the f90all_jinit.jar, webutil.jar and Jacob.jar files This is then followed by an indication of the Jsession ID.

```
Loading http://mymachine-pc2.uk.oracle.com:8889/forms90/java/f90all_jinit.jarfrom JAR cache

Loading http://mymachine-pc2.uk.oracle.com:8889/forms90/java/webutil.jar from JAR cache

Loading http://mymachine-pc2.uk.oracle.com:8889/forms90/java/jacob.jar from JAR cache

proxyHost=nullproxyPort=0connectMode=HTTP, native.Forms Applet version is : 9.0.4.0RegisterWebUtil - Loading Webutil Version 1.0.4 BetaOpening http://mymachine-pc2.uk.oracle.com:8889/forms90/l90servlet;jsessionid=8a03960422b943572eabdb5d47e8ac1ab294c4adb43b

Connecting http://mymachine-pc2.uk.oracle.com:8889/forms90/l90servlet;jsessionid=8a03960422b943572eabdb5d47e8ac1ab294c4adb43b with no proxyConnecting http://mymachine-pc2.uk.oracle.com:8889/forms90/l90servlet;jsessionid=8a03960422b943572eabdb5d47e8ac1ab294c4adb43b with cookie "ORA_UCM_VER=%2FMP%2F8i_tgrf_%2Cnp_i_qf%3Emp_ajc%2CamkMP%2F8g%5Drepd%5D*ln%5Dg%5Dod%3Ckn%5D_ha*_kiMP%2F8pckmrcGnMP%2F8naikpaEl;
ORA_UCM_INFO=3~000271477666646089252278168700 92~Kavitha~Prakash~kavitha.prakash@oracle.com~GBR~en~39~41~1~~1;
ORA_UCM_SRVC=3*OTN~1~0~//~null~*OPN~1~0~//~SE1%3ASE1%3ASE1%3ASE1%3ASE1%3ASE1%3ASE1%3A~*EMP~1~0~/34/~null~*GMO~1~0~//~null"
```

Oracle HTTP Server Access_log and Error_log

Get Requests are used to download the Forms Applet, establish a Forms Listener Servlet Session and connect to a Forms Runtime Process. Post Requests are then subsequently used to exchange information between the Forms client Applet and the Forms Runtime Process via the Forms Listener

The Oracle HTTP Server access_log file contains all Get and Post requests from the Forms client to the application server. All Get/Post Requests for a particular Forms session can be tracked using the unique JsessionID.

A few example Get/Post Requests in the access_log are shown below:

```
123.321.41.24 - - [20/Jun/2003:09:16:29 +0200] "GET /ecap/forms90java/f90all.cab HTTP/1.1" 304 - "-" "Mozilla/4.0 (compatible; MSIE 5.0; Win32)"\ 0
```

The above GET request attempts to download f90all.cab and receives a http 304 "Not Modified"(***HTTP/1.1" 304***) response from the Server. This tells the client it can use the cached copy of f90all.cab.

```
123.321.41.24 - - [20/Jun/2003:09:16:30 +0200] "GET
/ecap/forms90java/java/awt/KeyboardFocusManager.class HTTP/1.1"
404 347 "-" "Mozilla/4.0 (compatible; MSIE 5.0; Win32
```

The above GET request attempts to download KeyboardFocusManager.class and receives a http 404 "File Not Found" (***HTTP/1.1" 404***) response from the Server

```
123.321.41.24 - - 20/Jun/2003:09:16:35 +0200] "POST
/ecap/jecap/oracle.forms.servlet.ListenerServlet?JServSessionId
jecap=7jf415vfr1.nkbKq79ycALJmQ5Go6XNr3COc30Mbu-- HTTP/1.1" 200
2 "-" "Mozilla/4.0 (compatible; MSIE 5.0; Win32)"\ 0
```

The above POST request sends information to the Server using the unique session ID ***7jf415vfr1.nkbKq79ycALJmQ5Go6XNr3COc30Mbu*** *and* receives a http 200 "success" response and 2 bytes of data **(*HTTP/1.1" 200 2)***. This could be the result of an action such as the user tabbing between two fields.

Jserv and OC4J are J2EE containers. They are used in the application server to, amongst other things, run Java servlets such as the Forms Listener Servlet. Jserv is used in iAS Release 1 (1.0.2.x) and OC4J is used in the later releases.

The Oracle HTTP Server error_log typically reports errors encountered whilst accessing files from the Server or while interacting with OC4J or Jserv. For example, the following error in the error_log file indicates a problem downloading the default.dat file. It is worth noting that such errors will not be logged in the error_log file if the LogLevel directive in the httpd.conf file is set to "Critical".

```
[Tue Sep 23 09:54:18 2003] [error] [client 148.87.19.50] File
does not exist:


/private/ias/iAS902/forms90/java/oracle/forms/registry/default.
dat

.

java.io.FileNotFoundException: null_1455 (Permission denied)

at java.io.FileOutputStream.open(Native Method)

at java.io.FileOutputStream.<init>(FileOutputStream.java:102)
```

```
at java.io.FileOutputStream.<init>(Compiled Code)

at oracle.forms.servlet.RunformProcess.listProcesses(Compiled
Code)

at oracle.forms.servlet.RunformProcess.addProcess(Unknown
Source)

at oracle.forms.servlet.RunformProcess.connect(Compiled Code)

at oracle.forms.servlet.RunformProcess.dataToRunform(Compiled
Code)

at oracle.forms.servlet.RunformSession.dataToRunform(Unknown
Source)

at oracle.forms.servlet.ListenerServlet.doPost(Unknown Source)

at javax.servlet.http.HttpServlet.service(HttpServlet.java:521)

at javax.servlet.http.HttpServlet.service(HttpServlet.java:588)

at
org.apache.jserv.JServConnection.processRequest(JServConnection
.java:435)

at
org.apache.jserv.JServConnection.run(JServConnection.java:290)

at java.lang.Thread.run(Compiled Code)

Problem opening process list file null_1455 - process list
disabled
```

OC4J_BI_Forms Application.log

The OC4J_BI_Forms application.log (Jserv.log in 9iAS Release 1) contains Get/Post requests from the Forms client to the Application Server, only when the Forms Listener Servlet is run in debug mode. Each Get/Post request reported in the application.log is more informative than its counterpart in the access_log. A Get/Post request in the access_log contains basic information like the URL of the request, the time of the request and the HTTP response code. A Get/Post request in the application.log includes the various HTTP headers associated with the request, which provide detailed information like the number of bytes sent by the request, the Jinitiator version used on the client, the session tracking method used and so on. The application.log also provides other useful information about the Forms runtime session and reports errors encountered by the OC4J JVM.

Get Request

Get Requests are used to download the Forms applet, establish a Forms Listener Servlet Session and connect to a Forms runtime process

The Get request in the application.log file provides the following information:

- Host Name of the Client PC
- IP address of the Client PC
- The JsessionID
- Session Tracking Method - Cookies or URL Rewriting

For example, from the Get Request shown below, it can be discovered that the host name of the client pc is `kavitha-uk`, the client IP address is `23.1.151.120`, the JsessionID is `3116db55b1154b1c85dcc8f81078228d` and URL rewriting is used for session tracking (see "not from cookie").

```
29/06/04 09:53 forms90web: GET request received, cmd=getinfo,
qstring=ifcmd=getinfo&ifhost=kavitha-uk&ifip=123.1.151.120

29/06/04 09:53 forms90web: Existing servlet session, id =
3116db55b1154b1c85dcc8f81078228d, not from cookie

29/06/04 09:53 forms90web: Creating new Runtime Process using
default executable

29/06/04 09:53 forms90web: startProcess: executing ifweb90
server webfile=HTTP-0

29/06/04 09:53 forms90web: Getting stdin, stdout and stderr of
child process

29/06/04 09:53 forms90web: New server process created

29/06/04 09:53 forms90web: Forms session <2> started for
kavitha-uk ( 123.1.151.120 )
```

Post Request

In a request/response protocol like HTTP, the Post Requests used to exchange information between the Forms client applet and the Forms runtime process via the forms Listener

The post request in the application.log file includes several HTTP headers that provide very useful information. Proxy servers or load balancers between the client and the application server may also add their respective header information to the Post Request. A few common HTTP headers are described below:

PRAGMA

The pragma header is a sequential number for a given forms session and informs the Forms runtime process of the type of request that is coming from the Forms applet and the order in which to process these requests

COOKIE

The cookie header provides the cookie used to track the session.

HOST

The host header provides the application server host name.

CONTENT-LENGTH

This header provides the number of bytes sent from the Forms client.

ACCEPT-LANGUAGE:

This header provides the language used by the client browser.

USER-AGENT

The user agent header provides the Jinitiator version used by the client.

IFSESSION

The IFSESSION header is set to CLOSE and indicates a request to close down the forms session.

CONTENT-TYPE

The content type header contains a MIME content type, describing the format of the data sent from the client.

Apart from the information in the various HTTP headers, the post request also gives the following additional information:

- Process ID of the Forms runtime process for a given Forms session
- JsessionID
- Session tracking method – Cookies or URL Rewriting
- Port number used by the Listener Servlet to communicate with the Forms runtime process.

For example, from the post request below, we know that the browser language is English, host is test-machine, Jsession ID is ***27f6412da05c426ab47db4ae77636113***, cookies are not used to track the session (meaning that URL rewriting is used to identify the session), **jinitiator 1.3.1.17** is used, the process ID of the f90webm process is ***474***, the Listener Servlet uses port ***2791*** to talk to the Forms runtime process and the client sends **8** bytes to the Forms runtime process and receives **8** bytes back.

```
Got POST request, length HTTP request headers:
```

```
ACCEPT-LANGUAGE: en

PRAGMA 1

CONTENT-TYPE: application/x-www-form-urlencoded
ACCEPT:text/html, image/gif,   image/jpeg, *; q=.2, */*; q=.2

USER-AGENT: Java1.3.1.17-internal

HOST:test-machine:8888

CONTENT-LENGTH: 8

CONNECTION: Keep-Alive

Existing servlet session, id =
27f6412da05c426ab47db4ae77636113, not from cookie

Forms session <4> runtime process id = 474 Port number is 2791
RunformProcess.connect(): connected after 1 attempts Connected
to ifweb process at port 2791 Forms session <4>: request
processed in 1.032 sec. Received 8 bytes, returned 8 bytes.
```

Post requests can be broadly classified as follows, depending on the value of the Pragma Header:

Positive Pragma Header

A post request with a positive pragma and a non-zero content length is used to perform a normal forms operation like executing a built-in or performing a database operation like executing a query or stored procedure.

Example: Post Request with Positive Pragma Header

```
Got POST request, length = 20
HTTP request headers:
PRAGMA: 9
CONTENT-TYPE: application/octet-stream

USER-AGENT: Java1.3.1.17-internal
HOST: mymachine.domain:8889

ACCEPT: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
CONNECTION: keep-alive

      CONTENT-LENGTH: 20
```

```
Existing servlet session, id =
8a03931b22b97634765bbfc84cffa15ccf8f70d1919c, not from cookie
forms90web: Forms session <2>: request processed in 0.000 sec.
Received 20 bytes, returned 7 bytes.
```

Negative Pragma Header

A post request with a negative pragma is used for a zero content-length retry request to check whether the Forms runtime process has completed the previous request. When a long running request (like running a query) is submitted to the Forms runtime process, the Listener Servlet waits for a default time of one second for the completion of the request. The default time of one second can be configured using a Listener Servlet parameter called maxBlockTime. If the query is not completed, the Listener Servlet sends a "busy" response to the client requesting the client to retry. The client subsequently sends a zero content-length "retry" request with a negative pragma to check if the previous request has been completed. This "busy" response and "retry" request dialogue will continue until the Forms Runtime Process has completed the query.

Example: Post Request with a negative pragma header

```
Got POST request, length = 0
HTTP request headers:
PRAGMA: -21
CONTENT-TYPE: application/octet-stream

USER-AGENT: Java1.3.1.17-internal
HOST: mymachine.domain:8889

ACCEPT: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
CONNECTION: keep-alive

CONTENT-LENGTH: 0

Existing servlet session, id =
27f6412da05c426ab47db4ae77636113, not from cookie

Forms session <4>: request processed in 1.003 sec. Received 0
bytes, Returned 0 bytes

Read timed out. Sending Response Pending with retry time = 200

Forms session <4>: request processed in 1.003 sec. Received 0
bytes,  returned 0 bytes.

Read timed out. Sending Response Pending with retry time = 200
```

Duplicate Positive Pragma Header

A post request with a duplicate pragma is used for the terminal zero content-length close request to shut down the Forms session. This post request includes an additional IFESSION header with value "Close".

Example: Post Request with a duplicate pragma header

```
Got POST request, length = 16

HTTP request headers

PRAGMA: 13

CONTENT-TYPE: application/octet-stream

USER-AGENT: Java1.3.1.17-internal

HOST: ukp16646.uk.oracle.com:8889

ACCEPT: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2

CONNECTION: keep-alive

CONTENT-LENGTH: 16

Existing servlet session, id =
8a03931b22b97634765bbfc84cffa15ccf8f70d1919c, not from cookie

Forms session <2>: request processed in 0.041 sec. Received 16
bytes, returned 90 bytes.


Got POST request, length = 0

HTTP request headers

PRAGMA: 13

CONTENT-TYPE: application/octet-stream

IFSESSION: close

USER-AGENT: Java1.3.1.17-internal

HOST: ukp16646.uk.oracle.com:8889

ACCEPT: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2

CONNECTION: keep-alive

CONTENT-LENGTH: 0
```

```
Existing servlet session, id =

             8a03931b22b97634765bbfc84cffa15ccf8f70d1919c, not
from cookie

Got close Forms Session request, forwarding 0 length request to
runtime

Forms session <2> ended

Total duration of network exchanges: 0.471

Total number of network exchanges: 12 (0 "long" ones over 1.000
sec)

Average time for one network exchange (excluding long ones):
0.039

Total bytes: sent 5,635, received 655

RunformSession.valueUnbound(): stopping server process

RunformSession.stop(): Invalidating servlet session
```

DETECTING DIFFERENT TYPES OF ERRORS

Using the diagnostic information provided in the various trace and log files discussed earlier, we can effectively detect the source or cause for various different errors and problems.

Client Errors

Client Errors occur due to problems with the Forms Java client applet. These errors can be detected as follows:

The Jinitiator trace file will usually contain useful stack dumps.

The application.log file will give information about the behaviour of the Forms runtime process after the error. If the client applet can respond to the heart beat from the server, even after the error, the Forms runtime process will not terminate, leading to a deadlock or orphaned process. The Forms Runtime Process may also terminate gracefully after the FORMS90_TIMEOUT period.

Example: The following Java stack trace in the Jinitiator trace file indicates an error in the Forms Applet caused by a mouse event on a modal window (like an Alert box)

```
Exception occurred during event dispatching:

java.lang.ArrayIndexOutOfBoundsException: -1 < 0

at java.util.Vector.elementAt(Unknown Source)

at
oracle.ewt.lwAWT.lwWindow.SystemModalStyle.grabMouseEvent(Unkno
wn Source)

at
oracle.ewt.lwAWT.lwWindow.LWWindow$MouseGrab.grabMouseEvent(Unk
nown Source)

at
oracle.ewt.lwAWT.lwWindow.LWWindow$MouseGrab.mouseExited(Unknow
n Source)

at java.awt.AWTEventMulticaster.mouseExited(Unknown Source)

at
oracle.ewt.event.tracking.GlassMouseGrabProvider.processMouseGr
abs(Unknown Source)

at

oracle.ewt.event.tracking.GlassMouseGrabProvider$Disp._redispat
chEvent(Unknown Source)

at

oracle.ewt.event.tracking.GlassMouseGrabProvider$Disp._redispat
chEvent(Unknown Source)

at
oracle.ewt.event.tracking.GlassMouseGrabProvider$Disp._checkTar
get(Unknown Source)

at
oracle.ewt.event.tracking.GlassMouseGrabProvider$Disp.mouseClic
ked(Unknown Source)

at java.awt.Component.processMouseEvent(Unknown Source)

at oracle.ewt.lwAWT.LWComponent.processMouseEvent(Unknown
Source)

at java.awt.Component.processEvent(Unknown Source)

at java.awt.Container.processEvent(Unknown Source)

at oracle.ewt.lwAWT.LWComponent.processEventImpl(Unknown
Source)

at oracle.ewt.lwAWT.LWComponent.redispatchEvent(Unknown Source)
```

```
at oracle.ewt.lwAWT.LWComponent.processEvent(Unknown Source)
at java.awt.Component.dispatchEventImpl(Unknown Source)
at java.awt.Container.dispatchEventImpl(Unknown Source)
at java.awt.Component.dispatchEvent(Unknown Source)
at java.awt.LightweightDispatcher.retargetMouseEvent(Unknown
Source)
at java.awt.LightweightDispatcher.processMouseEvent(Unknown
Source)
at java.awt.LightweightDispatcher.dispatchEvent(Unknown Source)
at java.awt.Container.dispatchEventImpl(Unknown Source)
at java.awt.Component.dispatchEvent(Unknown Source)
at java.awt.EventQueue.dispatchEvent(Unknown Source)
at
java.awt.EventDispatchThread.pumpOneEventForHierarchy(Unknown
Source)
at java.awt.EventDispatchThread.pumpEventsForHierarchy(Unknown
Source)
at java.awt.EventDispatchThread.pumpEvents(Unknown Source)
at java.awt.EventDispatchThread.run(Unknown Source)
```

Network Errors

Network errors occur due to network problems like latency or loss of packets or anomalies caused by various network devices like routers, proxy servers, firewalls and load balancers. These errors can be detected as follows:

- The access_log and application.log files will show longer time intervals between the Get/Post Requests
- The application.log file may contain duplicate posts requests with identical pragma header numbers, indicating that the client is resending requests due to network packet losses. Under normal circumstances, only the last Post request, terminating a Forms session, should contain a duplicate pragma header number.

Network tracing tools and sniffers can be used to further diagnose the cause for these errors. Simple tracing utilities like ethereal (http://www.ethereal.com) and iehttpheaders(http://www.blunck.info/iehttpheaders.html) can be installed on the client PC to trace the network traffic from/to the browser client.

- The application.log may contain post requests without vital http headers like PRAGMA or CONTENT-LENGTH, indicating that a network device has effectively manipulated the requests from the client.

Example:

The access_log and application.log below show two post requests with identical **pragma 45** headers and a time interval of 2 minutes between the requests.

ACCESS_LOG

```
123.321.41.24 - - [20/Jun/2003:07:45:54 +0200] "POST
/ecap/jecap/oracle.forms.servlet.ListenerServlet/debug?JServSes
sionIdjecap=y99r25ugm1.nkbKq79ycALJmQ5Go6XNr3COc30MbK--
HTTP/1.1" 200 147 "-" "Mozilla/4.0 (compatible; MSIE 6.0;
Win32)"\ 0


123.321.41.24 - - [20/Jun/2003:07:47:55 +0200] "POST
/ecap/jecap/oracle.forms.servlet.ListenerServlet/debug?JServSes
sionIdjecap=y99r25ugm1.nkbKq79ycALJmQ5Go6XNr3COc30MbK--
HTTP/1.1" 200 64200 "-" "Mozilla/4.0 (compatible; MSIE 6.0;
Win32)"\ 0
```

APPLICATION.LOG

```
Got POST request, length = 2

HTTP request headers:

        user-agent: Mozilla/4.0 (compatible; MSIE 6.0; Win32)

        x-forwarded-host: sso.domain.co

        accept-language: de

[20/06/2003 07:45:54:938 GMT+01:00]
**********************************************

x-forwarded-for: 123.111.22.13
```

```
connection: close

[20/06/2003 07:45:54:939 GMT+01:00] Got POST request, length =
2

 x-forwarded-server: sso.domain.co

content-length: 2

[20/06/2003 07:45:54:939 GMT+01:00] HTTP request headers:

cookie: JSESSIONID=FLLNAFLBOCBN;
JServSessionIdjecap=y99r25ugm1.nkbKq79ycALJmQ5Go6XNr3COc30MbK--

content-type: application/octet-stream

[20/06/2003 07:45:54:939 GMT+01:00]

user-agent: Mozilla/4.0 (compatible; MSIE 6.0; Win32)

accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2

authorization: Basic ZGwyMGJ1bjpsaWZlYm9vazIx

[20/06/2003 07:45:54:939 GMT+01:00]         x-forwarded-host:
sso.domain.co

via: 1.0 sso.domain.co

host: 123.321.37.46:8080

[20/06/2003 07:45:54:939 GMT+01:00]

accept-language: de

pragma: 45

Existing servlet session, id =
y99r25ugm1.nkbKq79ycALJmQ5Go6XNr3COc30MbK--, from cookie

[20/06/2003 07:45:54:939 GMT+01:00]         x-forwarded-for:
123.111.22.13

[20/06/2003 07:45:54:939 GMT+01:00]         connection: close

[20/06/2003 07:45:54:939 GMT+01:00]         x-forwarded-server:
sso.domain.co

[20/06/2003 07:45:54:939 GMT+01:00]         content-length: 2

[20/06/2003 07:45:54:939 GMT+01:00]         cookie:
JSESSIONID=FLLNAFLBOCBN;
JServSessionIdjecap=y99r25ugm1.nkbKq79ycALJmQ5Go6XNr3COc30MbK--
```

[20/06/2003 07:45:54:939 GMT+01:00] content-type: application/octet-stream

[20/06/2003 07:45:54:939 GMT+01:00] accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2

[20/06/2003 07:45:54:939 GMT+01:00] authorization: Basic ZGwyMGJ1bjpsaWZlYm9vazIx

[20/06/2003 07:45:54:940 GMT+01:00] via: 1.0 sso.domain.co

[20/06/2003 07:45:54:940 GMT+01:00] host: 123.321.37.46:8080

[20/06/2003 07:45:54:940 GMT+01:00] **pragma: 45**

[20/06/2003 07:45:54:940 GMT+01:00] Existing servlet session, id = y99r25ugm1.nkbKq79ycALJmQ5Go6XNr3COc30MbK--, from cookie

Forms session <1>: request processed in 0.006 sec. Received 2 bytes, returned 147 bytes.

[20/06/2003 07:45:54:946 GMT+01:00] Forms session <1>: request processed in 0.006 sec. Received 2 bytes, returned 147 bytes.

Got POST request, length = 2

HTTP request headers:

user-agent: Mozilla/4.0 (compatible; MSIE 6.0; Win32)

x-forwarded-host: sso.domain.co

accept-language: de

x-forwarded-for: 123.111.22.13[20/06/2003 07:47:55:450 GMT+01:00] **********

connection: close

x-forwarded-server: sso.domain.co[20/06/2003 07:47:55:450 GMT+01:00] Got POST request, length = 2

content-length: 2

[20/06/2003 07:47:55:451 GMT+01:00] HTTP request headers:

cookie: JSESSIONID=FLLNAFLBOCBN; JServSessionIdjecap=y99r25ugm1.nkbKq79ycALJmQ5Go6XNr3COc30MbK—

content-type: application/octet-stream

```
[20/06/2003 07:47:55:451 GMT+01:00]         user-agent:
accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Mozilla/4.0 (compatible; MSIE 6.0; Win32)
authorization: Basic ZGwyMGJ1bjpsaWZlYm9vazIx

[20/06/2003 07:47:55:451 GMT+01:00]         x-forwarded-host:
sso.domain.co

via: 1.0 sso.domain.co

host: 123.321.37.46:8080

[20/06/2003 07:47:55:451 GMT+01:00]         accept-language: de

pragma: 45

Existing servlet session, id =
y99r25ugm1.nkbKq79ycALJmQ5Go6XNr3COc30MbK--, from
cookie[20/06/2003 07:47:55:451 GMT+01:00]         x-forwarded-
for: 123.111.22.13


[20/06/2003 07:47:55:451 GMT+01:00]         connection: close

[20/06/2003 07:47:55:451 GMT+01:00]         x-forwarded-server:
sso.domain.co

[20/06/2003 07:47:55:452 GMT+01:00]         content-length: 2

[20/06/2003 07:47:55:452 GMT+01:00]         cookie:
JSESSIONID=FLLNAFLBOCBN;
JServSessionIdjecap=y99r25ugm1.nkbKq79ycALJmQ5Go6XNr3COc30MbK--

[20/06/2003 07:47:55:452 GMT+01:00]         content-type:
application/octet-stream
```

PL/SQL Errors

The Forms runtime process whilst performing database operations, file input/output, executing PL/SQL code etc., may raise PL/SQL errors. These errors, sometimes give useful FRM/ORA messages. However, they may also give more generic error messages like FRM-92101/ FRM-92102. These errors can be detected as follows:

An FRM-93000 error is usually seen in the application.log file when the Listener Servlet is unable to communicate with the Forms runtime process as a result of the error.

The Jinitiator trace file will often show the following exception:

```
oracle.forms.net.ConnectionException:Forms session <7>
aborted:unable to  communicate with runtime process
```

PL/SQL errors can be further diagnosed using:

- Sqlnet tracing
- Forms Runtime Diagnostics
- Forms Trace

Example:

The Jinitiator trace file shows that the client has lost communication with the Forms runtime process. The FRM-93000 error in the application.log file confirms that the Forms Listener Servlet session is not able to communicate with the Forms runtime process.

JINITIATOR TRACE FILE

```
 Java Exception

   oracle.forms.net.ConnectionException:Forms session <7>
aborted:unable tocommunicate with runtime process

  at java.lang.Throwable.<init>(Compiled Code)

  at java.lang.Exception.<init>(Compiled Code)

  at java.ioException.<init>(IOException.java:45)

  at oracle.forms.netConnectionException.<init>(Unknown Source)

  at
oracle.forms.netConnectionException.createConnectionException(C
ompiled

  Code)

  at oracle.forms.net.HTTPNStream.getResponse(Compiled Code)

  at oracle.forms.net.HTTPNStream.doFlush(Compiled Code)

  at oracle.forms.net.HTTPNstream.flush(Compiled Code)

  at java.io.DataOutputStream.flush(Compiled Code)
```

```
 at oracle.forms.net.StreamMessageWrite.run(Compiled Code)
.
```

APPLICATION.LOG

```
[17/05/2002 22:32:48:060 JST] Got POST request, length = 8
[17/05/2002 22:32:48:060 JST] HTTP request headers:
[17/05/2002 22:32:48:060 JST]         content-type:
application/x-www-form-urlencoded
@ [17/05/2002 22:32:48:060 JST]         host:
tsoeda04.jp.oracle.com
[17/05/2002 22:32:48:060 JST]         accept: text/html,
image/gif, image/jpeg, *; q=.2, */*; q=.2
[17/05/2002 22:32:48:060 JST]         user-agent: Java1.1.8.16
[17/05/2002 22:32:48:060 JST]         pragma: 7188
[17/05/2002 22:32:48:060 JST]         content-length: 8
[17/05/2002 22:32:48:060 JST]         connection: keep-alive
[17/05/2002 22:32:48:060 JST] Existing servlet session, id =
5eozm0ffd1.JS4, not from cookie
[17/05/2002 22:32:48:060 JST] writing data into runform
[17/05/2002 22:32:48:070 JST] Forms session <7> aborted: unable
to communicate with runtime process.
[17/05/2002 22:32:48:511 JST] Forms session <7> exception stack
trace:
java.io.IOException: FRM-93000: Unexpected internal error.
Details : Invalid or absent Content-Length from runform
      at
oracle.forms.servlet.ListenerServlet.forwardResponseFromRunform
(Unknown
Source)
      at java.lang.Exception.<init>(Exception.java, Compiled
Code)
      at java.io.IOException.<init>(IOException.java, Compiled
Code)
```

```
      at
oracle.forms.servlet.ListenerServlet.forwardResponseFromRunform
(Unknown
Source)
      at oracle.forms.servlet.ListenerServlet.doPost(Unknown
Source)
      at
javax.servlet.http.HttpServlet.service(HttpServlet.java,
Compiled Code)
      at
javax.servlet.http.HttpServlet.service(HttpServlet.java,
Compiled Code)
      at
org.apache.jserv.JServConnection.processRequest(JServConnection
.java,
Compiled Code)
      at
org.apache.jserv.JServConnection.run(JServConnection.java,
Compiled Code)
      at java.lang.Thread.run(Thread.java, Compiled Code)
```

JVM Errors

The JVM container in which the Forms Listener Servlet is running (OC4J or Jserv) may also be responsible for raising errors. For example, the JVM may exceed the maximum heap size limit or operating system limits like file descriptors. The OC4J session timeout may also occur to terminate idle Listener Servlet sessions. Useful java stack dumps in the application.log file can help detect these errors.

Example:

The following java stack dump in the application.log file indicates a session timeout

```
[17/05/2002 22:32:48:060 JST] Got POST request, length = 8
[17/05/2002 22:32:48:060 JST] HTTP request headers:
```

```
[17/05/2002 22:32:48:060 JST]         content-type:
application/x-www-form-urlencoded
@ [17/05/2002 22:32:48:060 JST]         host:
tsoeda04.jp.oracle.com
[17/05/2002 22:32:48:060 JST]         accept: text/html,
image/gif,
image/jpeg, *; q=.2, */*; q=.2
[17/05/2002 22:32:48:060 JST]         user-agent: Java1.1.8.16
[17/05/2002 22:32:48:060 JST]         pragma: 7188
[17/05/2002 22:32:48:060 JST]         content-length: 8
[17/05/2002 22:32:48:060 JST]         connection: keep-alive
[17/05/2002 22:32:48:060 JST] Existing servlet session, id =
5eozm0ffd1.JS4,
not from cookie
[17/05/2002 22:32:48:060 JST] writing data into runform
[17/05/2002 22:32:48:070 JST] Forms session <7> aborted: unable
to communicate
with runtime process.
[17/05/2002 22:32:48:511 JST] Forms session <7> exception stack
trace:


java.io.InterruptedIOException: Read timed out
      at java.net.SocketInputStream.socketRead(Native Method)
      at
java.net.SocketInputStream.read(SocketInputStream.java:90)
      at
java.io.BufferedInputStream.fill(BufferedInputStream.java:190)
      at
java.io.BufferedInputStream.read(BufferedInputStream.java,
Compiled Code)
```

SUMMARY

Like any enterprise application, the development and management of an Oracle Forms application will require a continuous cycle of testing and debugging and

monitoring. With the disparate nature of a web infrastructure, diagnostic information can often be difficult to gather and interpret. The Forms Listener Servlet provides very useful diagnostic information, which is easy to interpret, and can help us to effectively detect various difficult and intermittent problems occurring in a web deployed Oracle Application Server 10*g* Forms Application.

Troublshooting the Forms Listener Servelet
September 2006
Author: Kavitha Prakash
Contributing Authors:Grant Ronald

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
www.oracle.com