



**ORACLE<sup>®</sup>**

## **GUI – MS5 DHQA**

Anissa Lam  
Oct 5, 2010

# Features Delivered for MS5

- Lifecycle Modules support
  - Listing/create/edit/delete Lifecycle modules
  - Support for DAS only and clustering environment
  - Virtual Server doesn't apply to Lifecycle modules
  - All Target related UI visible only when Cluster/Standalone Instance exists
  - Enable Status displayed accordingly
  - Target Tab allowing add/remove/enable/disable specific target



# Features Delivered for MS5

## – Applications Support

- Listing of Sub-components through REST
- Support for Application Versioning
- Viewing Deployment Descriptors, including Web Logics DD
- Virtual Servers Support in cluster env.
- Java Web Start Support



# Features Delivered for MS5

- Web Services Support
  - Endpoint Information
  - Tester
  - wsdl file viewing



# Lifecycle Modules Screen

Listing if only DAS exists.

## Lifecycle Modules

A lifecycle module performs tasks when it is triggered by one or more events in the server's lifecycle. Possible trigger server events are: initialization, startup, ready to service requests, and shutdown. Lifecycle modules are not part of the Java specification, but are an enhancement to the Enterprise Server.

### Lifecycle Modules (1)

| [New...](#) [Delete](#) [Enable](#) [Disable](#)

	Name	Enabled	Load Order
<input type="checkbox"/>	MyLifeCycle	✓	44



# Lifecycle Modules Screen

## New Lifecycle if only DAS exists

### New Lifecycle Module

OK Cancel

A lifecycle module performs tasks when it is triggered by one or more events in the server's lifecycle. Possible trigger server events are: initialization, startup, ready to service requests, and shutdown.

**Name: \***

**Class Name: \***   
Name must contain only alphanumeric, underscore, dash, or dot characters

**Classpath:**   
Can be blank if class is already in server classpath

**Load Order:**   
Order in which lifecycle modules are loaded when the server starts up. Modules with smaller integers are loaded sooner.

**Description:**

**Status:**  **Enabled**

**On Load Failure:**  **Prevent Instance Startup**  
If module load fails, do not start the instance



# Lifecycle Modules Screen

Edit Lifecycle if only DAS exists, No Target Tab

**General**

## Edit Lifecycle Module

Modify an existing lifecycle module.

**Name:** MyLifeCycle

**Class Name: \***   
Name must contain only alphanumeric, underscore, dash, or dot characters

**Classpath:**   
Can be blank if class is already in server classpath

**Load Order:**   
Order in which lifecycle modules are loaded when the server starts up. Modules with smaller integers are loaded sooner.

**Description:**

**Status:**  Enabled

**On Load Failure:**  Prevent Instance Startup  
If module load fails, do not start the instance



# LifeCycle Modules screen

Listing in cluster env.

## LifeCycle Modules

A lifecycle module performs tasks when it is triggered by one or more events in the server's lifecycle. Possible trigger server events are: initialization, startup, ready to service requests, and shutdown. Lifecycle modules are not part of the Java specification, but are an enhancement to the Enterprise Server.

**LifeCycle Modules (1)**

| [New...](#) [Delete](#)

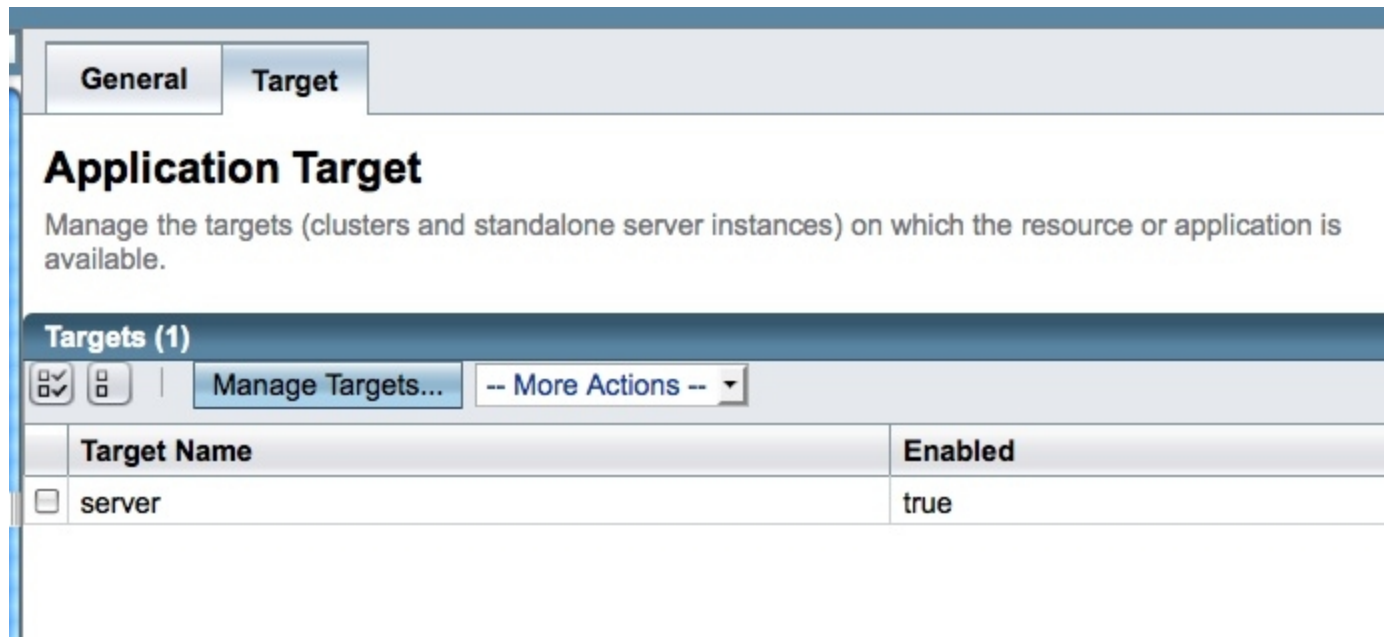
	Name	Status	Load Order
<input type="checkbox"/>	MyLifeCycle	Enabled on All Targets	44





# LifeCycle Modules screen

Edit in Cluster env. Target Tab exists, no VS action.



The screenshot displays the 'Application Target' configuration interface. At the top, there are two tabs: 'General' and 'Target', with 'Target' being the active tab. Below the tabs, the title 'Application Target' is followed by a descriptive text: 'Manage the targets (clusters and standalone server instances) on which the resource or application is available.' Below this is a section titled 'Targets (1)' which contains a table. The table has two columns: 'Target Name' and 'Enabled'. There is one row in the table with the target name 'server' and the status 'true'. Above the table, there are several controls: a checkmark icon, a refresh icon, a 'Manage Targets...' button, and a '-- More Actions --' dropdown menu.

Target Name	Enabled
server	true



# Web Services Screens

'webservices' listed as one of Engines

## Applications

Applications can be enterprise or web applications, or various kinds of modules.

### Deployed Applications (1)

	Name	Status	Engines	Action
<input type="checkbox"/>	TestWebJAX-WS20	Enabled on All Targets	webservices, web	Launch   Redeploy   Reload



# Web Services Screens

'Sub Components table allows View Endpoint', maybe more than 1 endpoint in an app.

**Edit Application** Save Cancel

Modify an existing application or module.

**Name:** TestWebJAX-WS20

**Status:** Enabled on All Targets

**Context Root:**   
Path relative to server's base URL

**Description:**

**Location:** \${com.sun.aas.instanceRootURI}/applications/TestWebJAX-WS20/

**Libraries:**

Modules and Components (4)				
Module Name	Engines	Component Name	Type	Action
TestWebJAX-WS20	[web, webservices]	-----	-----	Launch
TestWebJAX-WS20		jsp	Servlet	
TestWebJAX-WS20		default	Servlet	
TestWebJAX-WS20		NewService	Servlet	View Endpoint



# Web Services Screens

Information displayed when click on View Endpoint.

## Web Service Endpoint Information Back

View details about a web service endpoint.

<b>Application Name:</b>	TestWebJAX-WS20
<b>Tester:</b>	/TestWebJAX-WS20/NewServiceService?Tester
<b>WSDL:</b>	/TestWebJAX-WS20/NewServiceService?wsdl
<b>Endpoint Name:</b>	NewService
<b>Service Name:</b>	NewServiceService
<b>Port Name:</b>	NewServicePort
<b>Deployment Type:</b>	109
<b>Implementation Type:</b>	SERVLET
<b>Implementation Class Name:</b>	service.NewService
<b>Endpoint Address URI:</b>	/TestWebJAX-WS20/NewServiceService
<b>Namespace:</b>	http://service/



# Web Services Screens

All possible test links is displayed.

---

## Web Service Test Links

If the server or listener is not running, the link may not work. In such case, check the status of the server instance. After launching the web service test form, use the browser's Back button to return to this screen

**Application Name:** TestWebJAX-WS20

**Links:** [http://localhost:\\${HTTP\\_LISTENER\\_PORT}/TestWebJAX-WS20/NewServiceService?Tester](http://localhost:${HTTP_LISTENER_PORT}/TestWebJAX-WS20/NewServiceService?Tester)  
[https://localhost:\\${HTTP\\_SSL\\_LISTENER\\_PORT}/TestWebJAX-WS20/NewServiceService?Tester](https://localhost:${HTTP_SSL_LISTENER_PORT}/TestWebJAX-WS20/NewServiceService?Tester)  
<http://localhost:8080/TestWebJAX-WS20/NewServiceService?Tester>  
<https://localhost:8181/TestWebJAX-WS20/NewServiceService?Tester>

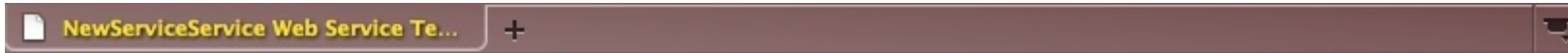
---

Close



# Web Services Screens

When click on one of the test link



## NewServiceService Web Service Tester

This form will allow you to test your web service implementation ([WSDL File](#))

To invoke an operation, fill the method parameter(s) input boxes and click on the button labeled with the method name.

### Methods :

```
public abstract java.lang.String service.NewService.sayHello(java.lang.String)
```

sayHello (  )



# Web Services Screens

All links for viewing WSDL

---

## View WSDL

If the server or listener is not running, the link may not work. In such case, check the status of the server instance. After launching the web service test form, use the browser's Back button to return to this screen

**Application Name:** TestWebJAX-WS20

**Links:** [http://localhost:\\${HTTP\\_LISTENER\\_PORT}/TestWebJAX-WS20/NewServiceService?wsdl](http://localhost:${HTTP_LISTENER_PORT}/TestWebJAX-WS20/NewServiceService?wsdl)  
[https://localhost:\\${HTTP\\_SSL\\_LISTENER\\_PORT}/TestWebJAX-WS20/NewServiceService?wsdl](https://localhost:${HTTP_SSL_LISTENER_PORT}/TestWebJAX-WS20/NewServiceService?wsdl)  
<http://localhost:8080/TestWebJAX-WS20/NewServiceService?wsdl>  
<https://localhost:8181/TestWebJAX-WS20/NewServiceService?wsdl>

---

Close





# Web Services Screens

click on one of the WSDL links to view the file



```
http://localhost:8080/TestWebJAX-WS20/NewServiceService?wsdl +
This XML file does not appear to have any style information associated with it. The document tree is shown below.
- <!--
  Published by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.2.2-hudson-
  -->
- <!--
  Generated by JAX-WS RI at http://jax-ws.dev.java.net. RI's version is JAX-WS RI 2.2.2-hudson-
  -->
- <definitions targetNamespace="http://service/" name="NewServiceService">
  - <types>
    - <xsd:schema>
      <xsd:import namespace="http://service/" schemaLocation="http://localhost:8080/TestWebJAX-
        WS20/NewServiceService?xsd=1"/>
      </xsd:schema>
    </types>
  - <message name="sayHello">
    <part name="parameters" element="tns:sayHello"/>
  </message>
  - <message name="sayHelloResponse">
    <part name="parameters" element="tns:sayHelloResponse"/>
  </message>
  - <portType name="NewService">
    - <operation name="sayHello">
      <input wsam:Action="http://service/NewService/sayHelloRequest" message="tns:sayHello"/>
      <output wsam:Action="http://service/NewService/sayHelloResponse"/>
    </operation>
  </portType>
</definitions>
```





# Support of viewing WebLogic DD

General **Descriptor** Target

## Module Descriptors

View module descriptors for the application.

Deployment Descriptors (6)		
Application Name:	Subcomponent Name	Descriptor File Name
ss_wl		META-INF/application.xml
ss_wl		META-INF/weblogic-application.xml
ss_wl	stateless-simpleEjb.jar	META-INF/ejb-jar.xml
ss_wl	stateless-simpleEjb.jar	META-INF/sun-ejb-jar.xml
ss_wl	stateless-simple.war	WEB-INF/sun-web.xml
ss_wl	stateless-simple.war	WEB-INF/web.xml



# Support of viewing WebLogic DD

## Deployment Descriptor

[Back](#)

A deployment descriptor is an XML file that describes how a Java EE application or module should be deployed. Each deployment descriptor has a corresponding Document Type Definition (DTD) file or schema (XSD) file. Enterprise Server supports Java EE Standard Descriptors, GlassFish Server Descriptors, and WebLogic Descriptors.

**Application Name:** ss\_wl

**Module Name:**

**Descriptor File Name:** META-INF/weblogic-application.xml

---

```
<?xml version="1.0" encoding="UTF-8"?>

<weblogic-application xmlns="http://xmlns.oracle.com/weblogic/weblogic-application"
<application-param>
<param-name>webapp.encoding.default</param-name>
<param-value>UTF8</param-value>
</application-param>
  <application-param>
<param-name>foo</param-name>
<param-value>bar</param-value>
</application-param>
</weblogic-application>
```



# Manage Virtual Server from the Target Screen

**Application Target**  
Manage the targets (clusters and standalone server instances) on which the resource or application is available.

**Targets (2)**

Manage Targets... -- More Actions --

	Target Name	Enabled	LB Enabled	Virtual Servers
<input type="checkbox"/>	clusterABC	true	true	<a href="#">Manage Virtual Servers</a>
<input type="checkbox"/>	server	true	true	<a href="#">Manage Virtual Servers</a>



# Java Web Start Support

- check the 'java web start' checkbox to enable it
- checkbox is enabled by default is specified as AppClient jar
- checkbox is disabled by default if packaged as ear.

The screenshot shows the 'New Application' dialog box in Oracle JDeveloper. The 'Location' section has two radio buttons: 'Packaged File to Be Uploaded to the Server' (selected) and 'Local Packaged File or Directory That Is Accessible from the Enterprise Server'. The first option has a text field containing '/Users/anilam/DeployApp/AppClient/showArgsApp.ear' and a 'Browse...' button. The second option has a text field and 'Browse Files...' and 'Browse Folders...' buttons. The 'Type' dropdown is set to 'Enterprise Application'. The 'Application Name' field contains 'showArgsApp'. The 'Status' checkbox is checked and labeled 'Enabled'. The 'Precompile JSPs' checkbox is unchecked and labeled 'Enabled'. The 'Run Verifier' checkbox is unchecked and labeled 'Enabled'. The 'Force Redeploy' checkbox is unchecked, with the text 'Force redeployment if this application is already deployed' below it. The 'Java Web Start' checkbox is checked and labeled 'Enabled'. A legend in the top right corner indicates that an asterisk (\*) denotes a required field.



# Java Web Start Support

- Just like web app, a Launch link is provided.

**Edit Application** Save Cancel

Modify an existing application or module.

**Name:** showArgsApp

**Status:** Enabled on All Targets

**Java Web Start:**  Enabled

**Description:**

**Location:** \${com.sun.aas.instanceRootURI}/applications/showArgsApp/

**Libraries:**

Modules and Components (2)				
Module Name	Engines	Component Name	Type	Action
showArgs-client.jar	[appclient]	-----	-----	Launch
client2/showArgs-client2.jar	[appclient]	-----	-----	Launch



# Java Web Start Support

- Links provided for all deployed target

## Application Client Launch Page

Launch

Back

If the server or listener is not running, links may not work. In such case, check the status of the server instance.

**Application:** showArgsApp

**Module:** showArgs-client.jar

**Links:**

- [http://localhost:\\${HTTP\\_LISTENER\\_PORT}/showArgsApp/showArgs-client](http://localhost:${HTTP_LISTENER_PORT}/showArgsApp/showArgs-client)
- [https://localhost:\\${HTTP\\_SSL\\_LISTENER\\_PORT}/showArgsApp/showArgs-client](https://localhost:${HTTP_SSL_LISTENER_PORT}/showArgsApp/showArgs-client)
- <http://localhost:8080/showArgsApp/showArgs-client>
- <https://localhost:8181/showArgsApp/showArgs-client>

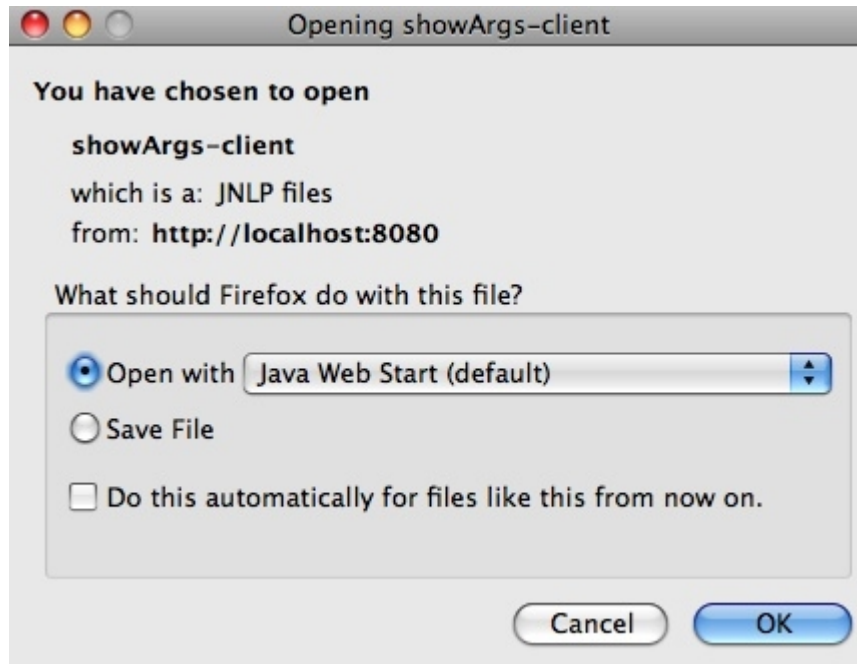
**Arguments:**

Arguments to append to the URL for launching the application; for example, `arg=first&arg=second`



# Java Web Start Support

- After entering the arg and press Launch





# OSGi Deployment

- check the osgi checkbox during deployment
- no DD for OSGI

## Deploy Applications or Modules

OK Cancel

Specify the location of the application or module to deploy. An application can be in a packaged file or specified as a directory.

\* Indicates required field

Location:  **Packaged File to Be Uploaded to the Server**

/Users/anilam/DeployApp/OSGI/monitoring-scripting-client.jar

Browse...

**Local Packaged File or Directory That Is Accessible from the Enterprise Server**

Browse Files...

Browse Folders...

Type: \* Other

Application Name: \* monitoring-scripting-client

Status:  Enabled

Precompile JSPs:  Enabled

Run Verifier:  Enabled

Force Redeploy:   
Force redeployment if this application is already deployed

OSGi Type:   
The component is packaged as an OSGi Alliance bundle.

Libraries:





# Applications table

## Applications

Applications can be enterprise or web applications, or various kinds of modules.

Deployed Applications (4)				
<input checked="" type="checkbox"/>	<input type="checkbox"/>	Deploy...	Undeploy	Filter: <input type="text"/>
	Name	Status	Engines	Action
<input type="checkbox"/>	showArgsApp	Enabled on All Targets	ear, appclient	Redeploy   Reload
<input type="checkbox"/>	monitoring-scripting-client	Disabled on All Targets		Redeploy   Reload
<input type="checkbox"/>	ss_wl	Enabled on All Targets	ear, ejb, web	Redeploy   Reload
<input type="checkbox"/>	TestWebJAX-WS20	Enabled on All Targets	webservices, web	Launch   Redeploy   Reload



# Developer Tests

- Admin Console Devtests at
  - v3/admingui/devtests
- Running GUI Devtests
  - cd v3/admingui/devtests
  - mvn install to run all the devtests
  - mvn -Dtest=<TEST\_CLASSNAME> test
    - eg mvn -Dtest=ClusterTest test



# Hudson Job

- setup at
  - <http://hudson.sfbay.sun.com/view/GFv3/job/v3-admingui-devt>
- Currently offline, need to revive it.



# Know Issues

- Web Service Test Links port number is not resolved.
  - issue# 13796 under Siraj
- In AppClient launch links page, always select the first link regardless what is choosen.
  - Issue#13803
- Application table doesn't show 'osgi' engine due to deployment issue# 13800



# Documents

- One Pager
  - <http://wikis.sun.com/display/GlassFish/GF3.1AdminConsoleC>
- Project Plan
  - <http://wikis.sun.com/display/GlassFish/GlassFish3.1AdminCo>
- GUI Plugin Modules
  - <http://wikis.sun.com/download/attachments/209655113/admir>
- GUI MS5 deliveries
  - <http://wikis.sun.com/display/GlassFish/GUI-MS5-DHQA>

