



# Monitoring in GlassFish V3

17<sup>th</sup> Aug 2009

**Prashanth Abbagani**  
**Byron Nevins**



# Agenda

- Probe Annotations
- Writing Probes in Java
- Providing Probes in XML
- Scripting Demo
- DTrace Demo
- Summary

# Probe Annotations

- @ProbeProvider
  - > @ProbeProvider(
    - moduleProviderName="glassfish",*
    - moduleName="web",*
    - probeProviderName="web-module"*)
- @Probe
  - > @Probe(*name="webModuleStartedEvent"*)
- @ProbeParam
  - > @ProbeParam(*"appName"*) *String appName,*

```
@Probe(name="webModuleStartedEvent")
public void webModuleStartedEvent(
  @ProbeParam("appName") String appName,
  @ProbeParam("hostName") String hostName) {}
```

# Defining Providers - Java

## Defining the Provider Class

```
@ProbeProvider(providername="glassfish"  
    , modulename="web")  
  
Class JspProbeProvider {  
  
    ...  
  
    @Probe(name="jspLoadedEvent")  
    public void jspLoadedEvent(  
        @ProbeParam("jsp") String jsp,  
        @ProbeParam("hostName")  
        String hostName) { }  
  
    ....  
}
```

## Emitting the probe

```
Class JspProbeEmitterImpl.java{  
  
    ...  
  
    public JspProbeEmitterImpl(){  
        JspProbeProvider jspProvider =  
            new JspProbeProvider();  
        .....  
    }  
  
    public jspLoaded (String jsp, String vsId){  
        jspProvider.jspLoadedEvent(jsp, vsId);  
    }  
  
    ...  
}
```

# Packaging the Java Provider

- Put an entry in META-INF/MANIFEST.MF of your module

*probe-provider-class-names* : *<provider-class-name[,provider-class-name]>*

## Pom.xml

*<configuration>*

*<archive>*

*<manifestEntries>*

*<probe-provider-class-names>*org.glassfish.web.admin.monitor.RequestProbeProvider

*</probe-provider-class-names>*

*.....*

# Demo – Probes from Java

- Code walkthrough
- Adding the ProbeProvider class
- `asadmin list-probes`

# Defining Providers using XML

## XML as the Provider

```
<probe-providers
<probe-provider moduleProviderName="glassfish" moduleName="web"
  probeProviderName="jsp" class="com.sun.enterprise.web.jsp.JspProbeEmitterImpl">
  <probe name="jspLoadedEvent">
    <method>jspLoaded</method>
    <probe-param type="String" name="appName" />
    <probe-param type="String" name="hostName"/>
    <return-param type="void" />
  </probe>
</probe-provider>
</probe-providers>
```

# Packaging the XML Provider

- Put an entry in META-INF/MANIFEST.MF of your module

*probe-provider-xml-file-names* : *<provider-xml-file-name[,provider-xml-file-name]>*

## Pom.xml

*<configuration>*

*<archive>*

*<manifestEntries>*

*<probe-provider-xml-file-names>gfprobe-provider.xml</probe-provider-xml-file-names>*

*. . . . .*

# Demo – Probes from XML

- Adding the ProbeProvider from XML
- `asadmin list-probes`

# Demo - Client-Scripting

- Writing a javascript – listens to the probes
- Executing the script using asadmin
- Changing the script on the fly

# Future

- XML extension mechanism
- Ability to expose Probes in a deployed App
- Support for other scripting languages
  - > Java, Groovy, JRuby etc.,

# Listeners

- Listens to Probe events emitted by Provider
- Collects statistics based on the probe events
- Exposes stats through CLI/GUI/REST/MBeans
- Documentation
  - > Writeup
  - > Guidelines for writing ProbeProviders and StatsProviders

# Dtrace - Demo

- End-to-end monitoring on Solaris platforms

# Summary

- Easy and Quick
- Powerful
- Fun



# Monitoring in GlassFish V3

17<sup>th</sup> Aug 2009

**Prashanth Abbagani**  
**Byron Nevins**

