



the  
**POWER**  
of  
**JAVA™**

**ORACLE®**



JavaOne  
Part of the Oracle and Sun Microsystems

# GlassFish™ Project: An Easy Web Application Development Java™ Server

Jean-Francois Arcand

Staff Engineer  
Sun Microsystems

BOF-2116



# GlassFish.next and the Web Development

The future of Web Development in GlassFish

This presentation highlights some of the features that will improve the web development experience in GlassFish.next

# Agenda

Introduction

Web development: The problems with the current existing Java EE Application Servers.

GlassFish.next: Components on demand

Demo

Q&A

# Agenda

## Introduction

The problems with the current Java EE  
Application Servers

What we want

GlassFish.next: Make it simple

Demo

Q&A

# Introduction

- GlassFish is currently a full fledged Java EE implementation which includes EJB, JMX, JSP, JSF, Servlet, etc.
- For web developer, it means you have much more than you want.
- When you only develop web technologies (JSF, JSP, Servlet), you don't necessarily want to install extra components you never use.

Source: Please add the source of your data here

# Agenda

Introduction

**The problems with the current Java EE  
Application Servers**

What we want

GlassFish.next: Make it simple

Demo

Q&A

# The problems with the current Java EE Application Servers

- Fact(?)  
Java EE web technologies in Application Servers have never had the same impact as other web technologies (PHP, Ruby On Rail).

Source: Please add the source of your data here

# The problems with the current Java EE Application Servers (Cont.)

- Some reasons:
  - Large binaries: from ~10m (Tomcat) to ~62m (GlassFish)
  - Complex configurations: server.xml (Tomcat), domain.xml (GlassFish)
  - Complex layout to learn (war)
  - Complex deployment command (but simplified with autodeploy...until it fails ;-))
  - Deployment errors not very user friendly (ex: XML errors)

Source: Please add the source of your data here

# Agenda

Introduction

The problems with the current Java EE  
Application Servers

**What we want**

GlassFish.next: Make it simple

Demo

Q&A

# Web Development: What we want

- **Easy Setup:** Small binary that I can easily download and install.
- **Easy Configuration:** No configuration! It is all about conventions over configuration. I can easily create the structure for a new application. Then I just drop files at the right place and it all works.

Source: Please add the source of your data here

## Web Development: What we want (Cont.)

- **Easy Database Integration:** Extremely simple database configuration. Simplified access to database through ORM.
- **Easy Development Cycle:** It's termed Zero Deployment Time. Make change to anything, hit refresh in the browser, Voila!

Source: Please add the source of your data here

## Web Development: What we want (Cont.)

- **Easy Testing:** Test framework already embedded in the environment. Automatic generation of tests.
- **Easy Deployment in the Real World:** Flip a switch and your web app is ready to deploy in the real world.

Source: Please add the source of your data here

## Web Development: What we want (Cont.)

- **Easy Growth Path:** Want to handle more complex tasks? Ready to promote your web app to a more powerful environment? Flip a switch and your web app is ready to play with powerful tools and enterprise application environments.

Source: Please add the source of your data here

# Agenda

Introduction

The problems with the current Java EE  
Application Servers

What we want

**GlassFish.next: Make it simple**

Demo

Q&A

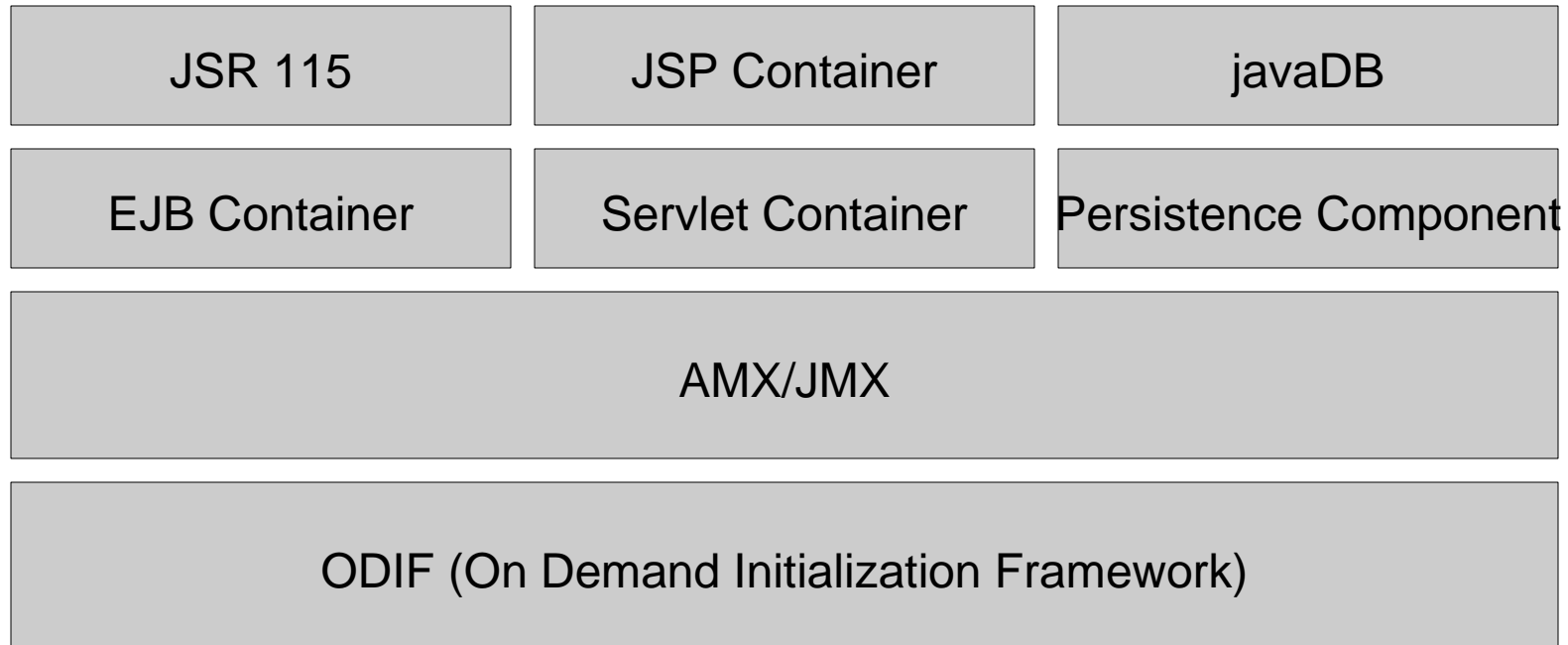
# GlassFish.next: Make it simple

- Reduce the bundle size:
  - Small footprint, Components and Containers are installed on demand.
- Improve deployment experience
  - Dynamic class loading to avoid deployment cycle
  - Simplify web app layout (no WEB-INF, no war\*\*)
  - Ability to map a resource to the containers it needs.
- Let's see what we have now, and what we can do.

Source: Please add the source of your data here

\*\* Already supported in GlassFish

# What we have now...



[1][http://weblogs.java.net/blog/binod/archive/2005/09/lazy\\_initializa.htm](http://weblogs.java.net/blog/binod/archive/2005/09/lazy_initializa.htm)

Source | Please add the source of your data here

## What we have now... (Cont.)

- When you start GlassFish, you are mainly starting ODIF and AMX
- When you open a connection to a Container, the Container is started:
  - Invoking <http://localhost:4848> will start the Servlet and JSP Container.
  - Invoking an EJB will start the EJB Container
- But If you never use EJB, why do you have the Container installed but not started?

Source: Please add the source of your data here

## GlassFish.next: What we want

- When you start GlassFish, you only start ODIF without any Components or Containers installed.
- A special ODIF Service, called PlugNow is started and listens for requests for Components and Containers.
- Make it very simple to add Components or Containers on demand.

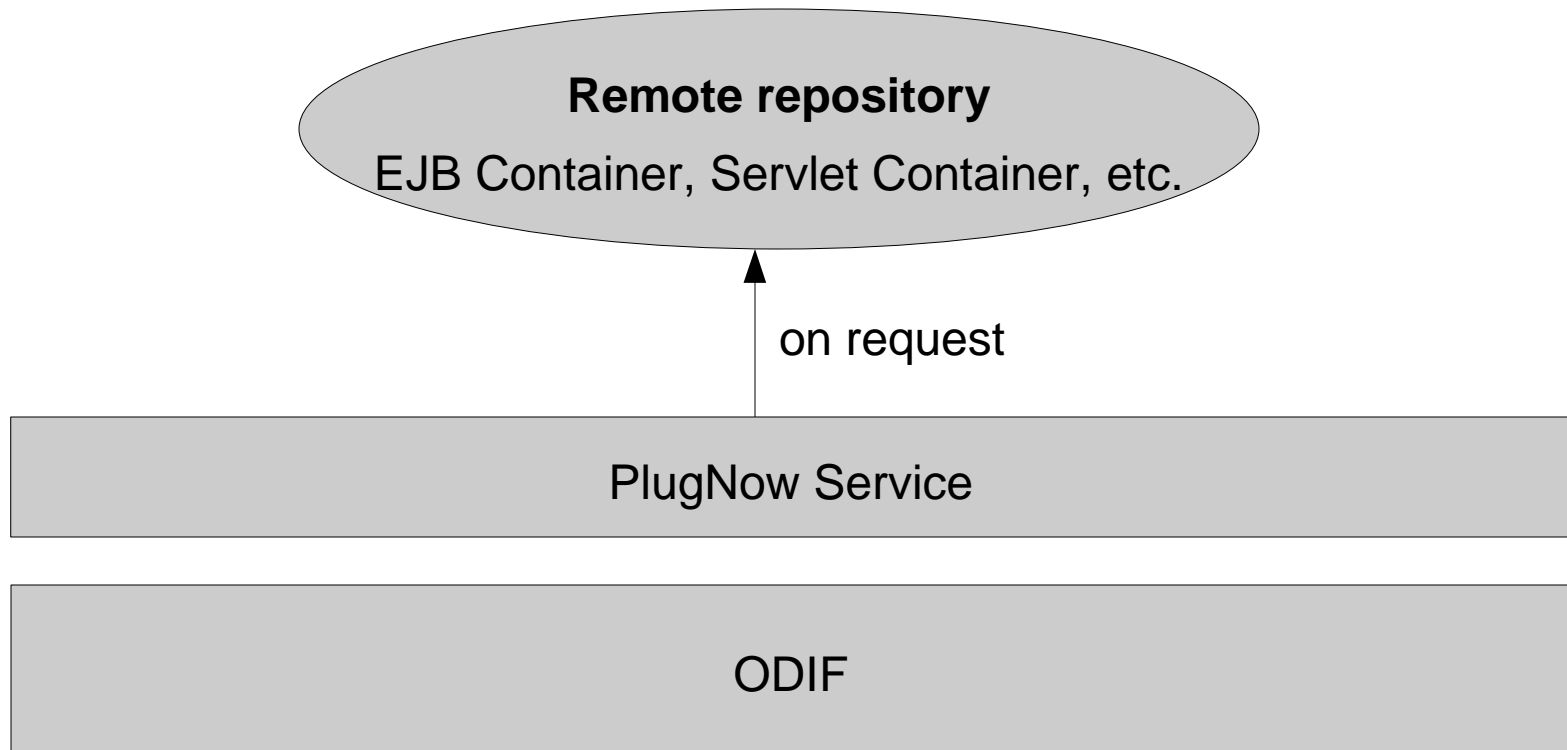
Source: Please add the source of your data here

# GlassFish.next: What we want (Cont.)

- Four Easy steps:
  - Download GlassFish's ODIF ( ~4m )
  - Start PlugNow Service
  - Based on your need, install
    - JSF Support
    - Servlet
    - Servlet and JSP
    - JSF with Persistence Support
    - EJB

Source: Please add the source of your data here

# GlassFish.next: What we want (Cont.)



Source: Please add the source of your data here

# Goals of PlugNow Service

- Download the appropriate Components or Containers from a local or remote repository.
- Make sure there is no class loader issue between the existing Components and Containers.
- Make is simple and fast for developer who wants to add their own Component or Containers.
- Maintains Java EE Compatibility.

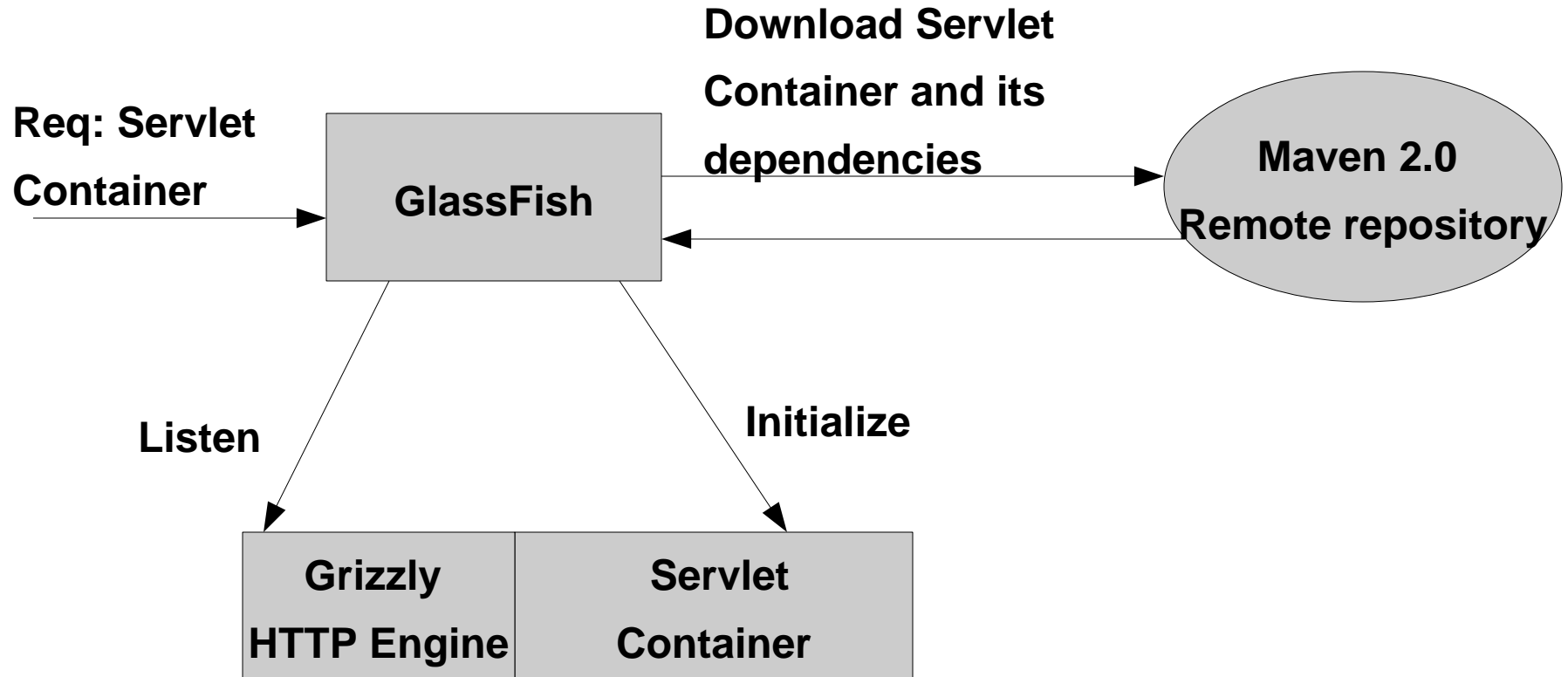
Source: Please add the source of your data here

# PlugNow Service: Architecture

- Grizzly
  - Simple Web Server that listens for requests for Components and Containers.
- A new Class Loader that isolates Container classes.
- Embedded Maven 2.0
  - Use Maven 2.0's dependency resolution mechanism to download Components and Containers from Maven 2.0 local or remote repository.

Source: Please add the source of your data here

# PlugNow In Action



Source: Please add the source of your data here

# Adding a Component or Container to PlugNow

- Three simple steps:
  - Have your component/container implement a simple PlugNow interface
  - Write your Maven 2.0 Project Object Model (POM), and deploy your component/container to a Maven repository.
  - Register your PlugNow implementation

Source: Please add the source of your data here

## PlugNow Architecture:

- Implement the PlugNow simple interface to instantiate, start and stop the component/container

```
initPlugNow(PlugNowContext)  
startPlugNow(PlugNowContext)  
stopPlugNow(PlugNowContext)
```

- Register the component with PlugNow

```
servlet=org.glassfish.plugin.ServletPlugNow
```

Source: Please add the source of your data here

# PlugNow Architecture:

- Write the Maven 2.0 POM:

```
<dependencies>
  <dependency>
    <groupId>glassfish.next</groupId>
    <artifactId>servlet</artifactId>
    <version>2.5.0</version>
  </dependency>
  <dependency>
    <groupId>glassfish.next</groupId>
    <artifactId>jasper</artifactId>
    <version>2.1.0</version>
  </dependency>
</dependencies>
```

Source: Please add the source of your data here

# DEMO

GlassFish.next: PlugNow demo

# Summary

- Web development experience: still improvements to be made.
- GlassFish.next: Component based approach. Install only what you need.
  - Try to leverage the buzz around Maven 2.0 repository.
  - Make deployment fast and simple

# Q&A

Optional Speaker Names Here

*Instructions:*

*(Delete this red box before  
submitting your slides)*

*Use this slide to mark the  
beginning of the Question  
& Answer section of your  
presentation.*



the  
**POWER**  
of  
**JAVA™**

**ORACLE®**



JavaOne  
Part of the Oracle and Sun Microsystems

# GlassFish™ Project: an Easy Web Application Development Java™ Server

Jean-Francois Arcand

Staff Engineer  
Sun Microsystems

BOF-2116