

# THE GREAT JAVA APPLICATION SERVER DEBATE

TOMCAT, JBOSS, JETTY, GLASSFISH, LIBERTY  
PROFILE, WEBLOGIC & WEBSPHERE



# TABLE OF CONTENTS

## INTRODUCTION

LET THE DEBATES BEGIN... 1-2

## PART I

GETTING STARTED 3-15

## PART II

GOING NINJA 16-30

## PART III

WHICH APP SERVER IS BETTER FOR WHOM? 31-37

## PART IV

THE PAIR OF ELEPHANTS IN THE ROOM (WEBLOGIC AND WEBSHERE) 38-41

## PART V

...AND THE BEST APPLICATION SERVER AWARD GOES TO... 42-43

## TOO LONG, DIDN'T READ (TL;DR)

SUMMARY, CONCLUSION AND A COMIC ;-) 44-46

# INTRODUCTION

## LET THE DEBATES BEGIN...

What type of Java app server should you choose for your next project? Well, that kinda depends on what kind of app you're building, what your needs are, what type of organization you work in, and lots of other factors too. Hence the debate. So perk up your ears, and get ready for a showdown (SPOILER: Jetty doesn't win all!)

## What makes an App Server, Mr. Lebowski?

What makes an application server is a contentious question to many, as the definition is unclear. Wikipedia, the unquestionable source of all knowledge, states:

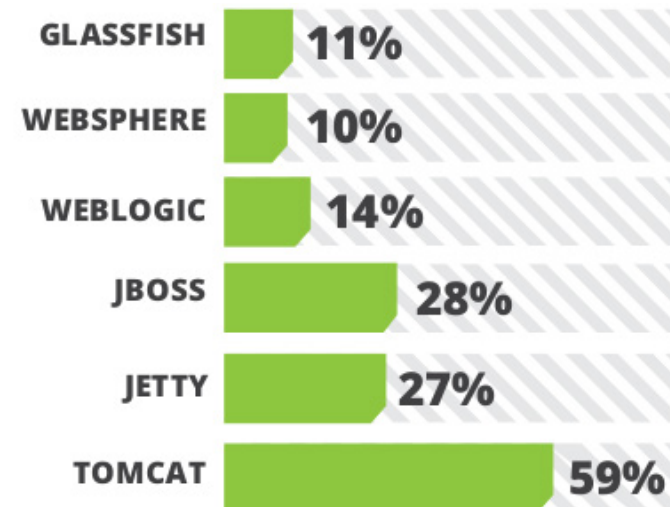


Java Platform, Enterprise Edition or Java EE (was J2EE) defines the core set of API and features of Java Application Servers



Well, sorry Wikipedia, but for the sake of this report, we don't care about pedantic definitions and full JEE implementations, we care more about what a developer wants and uses.

Most developers work on web applications and rarely use all of the bells and whistles that come with the EE specification. In fact many of the application servers available today with only the basic functionality are the most used, as our [Developer Productivity Report](#) section on application servers recently showed:

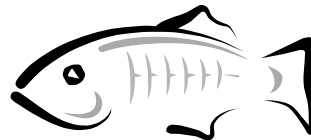
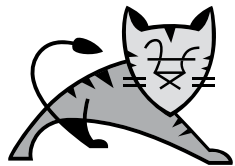


What we're looking at in this report are the real developer concerns and metrics including:

- Ease of download and installation
- Real performance metrics
- How do teams respond to these failures?
- Tooling support
- How long do recoveries take?
- Server Configuration

We're going to compare and contrast many aspects of these servers scoring each and placing each of them based on our findings. We then address the big questions: Which app server is the best? Doesn't someone in a big company have different needs than a hobbyist developer geek working at home? How should we weight what's important to different profiles of people? And so on...

The Application Servers we will be discussing in this report are:




At first, we were going to include IBM WebSphere and Oracle WebLogic into this report, following the same pattern as with the other app servers. But it felt unfair, as these servers are really targeted for large-enterprises and apps in production rather than lightweight development; however, we didn't want to dismiss them completely, so they'll have their own section at the end :-)

# PART I

## GETTING STARTED

When you're getting started with a new technology, things like installation, configuration, tooling support and documentation can affect your experience in a major way. In Part I, we look at starting off with each of the app servers.

This section will compare and contrast each of the Application Servers for each of the following areas ranking them with a score out of 5 in addition to our comments. As we at ZeroTurnaround are all about developer productivity, the unit of measure will be the Rebel - 

- Download and Installation
- Tooling support
- Server Configuration
- Documentation & community

Next, in Part II will look into some of the deeper topics as we go undercover, Ninja style.

## Download & Installation

The first contact is all important as it often gives your initial glimpse of how the rest of the experience might turn out!

### JETTY

Jetty is just 8MB in size! The process is easy:

1. Download installation package
2. Extract the archive
3. You are ready to go!

Jetty has preserved its simplicity in starting: `java -jar start.jar`

**Score:** 

#### Reason:

Smallest download, maven dependency integration, unzip install and good startup scripts.

### TOMCAT

Tomcat is identical in experience to Jetty, although it's another few meg at 12.8.

1. Download installation package
2. Extract the archive
3. You are ready to go!

To start Tomcat, simply run `bin/startup.sh` - easy

**Score:** 

#### Reason:

Small download, maven dependency integration, unzip install and good startup scripts.

## JBOSS

JBoss AS 7.1.1.Final, which was release on March 9th 2012, but still is the latest community edition version, archive is 127Mb.

1. Download installation package
2. Extract the archive
3. You are ready to go!

To start, RedHat have made it simple as well with the bin/standalone.sh script, unless you want to run a clustered environment. Also there are several default configuration files in "standalone/configuration" directory, which allow to turn on and off default clustering and choose between web or full EE profiles, so one can pick the closest to what is needed and tweak it minimally.

**Score:** 

### Reason:

Largest download, unzip install, slower, "standalone" named startup scripts.

## LIBERTY PROFILE

1. Accept Licence agreement
2. Download installation package
3. Extract the archive
4. Accept Licence agreement again - seriously?
5. You are ready to go!

Although you have to accept the license agreement twice, it's not all that bad as most people just click accept anyway :) Does anyone actually read them? The download is 41.4 MB which isn't too bad given the extra features it has over say Tomcat or Jetty

**Score:** 

### Reason:

Reasonable size, license have to be accepted twice, startup scripts need docs :(



## GLASSFISH

There are two options available: GlassFish Open Source Edition and Oracle GlassFish Server. We grabbed the GlassFish Server Open Source Edition installation package, which surprisingly, was not so big, only 53 MBs in size! This is the installation package download and comes in a self-extractable archive. Once downloaded you need to run a large .sh “script” and after a number of steps through a graphical wizard, GlassFish was in place. If you don’t like the idea of an installer, you can also choose the unzip install option which comes in a download around 30MB larger. This gives you the three step experience as most of the others:

1. Download installation package
2. Extract the archive
3. You are ready to go!

Although you have to accept the license agreement twice, it’s not all that bad as most people just click accept anyway :) Does anyone actually read them? The download is 41.4 MB which isn’t too bad given the extra features it has over say Tomcat or Jetty

**Score:** 

### Reason:

No straightforward way to start the server, multiple install methods including unzip install is good, reasonable download size, but still large.

## SCORE SUMMARY



## Tooling support

Developers are as loyal with their IDEs as they are their appservers, so it's important for Application Servers to support the main IDEs so they don't force their developers into using a platform they don't want to, or missing out on Tooling support entirely.

For many, good integration with build tools can be as important as good IDE integration, sometimes more important. The two main build tools used which we will be looking at are maven and ant.

### JETTY

Jetty has an eclipse Wtp plugin ([http://wiki.eclipse.org/Jetty\\_WTP\\_Plugin](http://wiki.eclipse.org/Jetty_WTP_Plugin)) that enables you to run web apps directly from Eclipse. Also, check out [Run Jetty Run](#), another eclipse plugin.

JetBrains also has a listed [plugin for IntelliJ](#) as does NetBeans and can also be leveraged using [Jetty's Maven plugin and NetBeans](#) together.

Jetty's build tool integration is quite well developed. It's has integration with Ant, Maven out the box and can also be integrated with whichever tool you like, with minimal effort.

It is possible to configure your development environment very closely to the one that you are running in production. The maven plugin for Jetty allows you to configure almost everything in the pom.xml file.

The thing that we liked with Maven's Jetty plugin is that Jetty is so tiny. So it won't take long for Maven to download all the internets.

**Score:**



**Reason:**

The Eclipse plugin for earlier versions of Jetty is not great, light server management support.

### TOMCAT

The big three IDEs, Eclipse, IntelliJ IDEA and Netbeans, all have integration support for Tomcat out the box. This means you can import a server into the tooling and deploy projects onto the server. NetBeans offers a distribution which wraps a Tomcat image so that you can deploy as soon as NetBeans installs :o) Eclipse also has the additional download feature which allows you to download and install Tomcat from within your tooling, meaning you don't have to do it via a browser and later import it. Tomcat seems to be very tightly integrated with Ant, and most documentation includes ant examples rather than Maven, although Maven has good integration as well.

**Score:**

**Reason:**

Great IDE support, and download/install through eclipse gives a good experience.

## JBOSS

[jboss-tools](#), an umbrella project for multiple eclipse plugins, includes among other things JBoss Developer Studio which has a server adaptor for JBoss Servers and allows you to manage jbosses from your IDE. IDEA has a closed source “JBoss Integration” plugin to manage servers, and NetBeans has integration and ability to manage AS 7-th since version 7.2.1.

There’s a [maven plugin](#) to manage your jboss instances. Its functionality includes starting and stopping, deploying and redeploying apps, managing resources and executing commands like through a jboss CLI.

**Score:** 

### Reason:

Good cross IDE support, able to make light server config changes, good maven support

## LIBERTY PROFILE

The Liberty Profile tools, named IBM WebSphere Application Server Developer Tools for Eclipse (a product name that almost needs punctuation) is only supported on the Eclipse platform. If you’re an IntelliJ IDEA or NetBeans fan, you’re out of luck. The tools let you download and install the whole server easily, and provides feature rich development for the programming models which the Liberty Profile supports. It also provides rich editor support for server config changes in a similar style to a deployment descriptor editor which you’ve likely seen/used. Overall the tooling is good, on Eclipse.

The Liberty profile also includes a Maven plugin which provides allows you to drive some actions to a server, You can install, start, stop, package and create server using predefined goals. Driving these is fairly easy:

```
mvn liberty:package-server -DserverHome=/path/to/server_
home -DserverName=[server_name] -DpackageFile=/path/to/
packaged server file location
```

**Score:** 

### Reason:

Great eclipse support and rich server config changes. Good Maven support, but lacking in cross IDE support.

## GLASSFISH

GlassFish has plugins available for all major IDEs these days. But the problem with the plugin is that the information is scattered around. The plugin for NetBeans is already bundled with the IDE, as you'd expect given the vendor behind it, but others need to find the correct website to install the plugin. Fortunately, GlassFish Tools are available in the Eclipse Marketplace for Eclipse users - this makes Eclipse user's lives a lot easier.

IntelliJ IDEA users are lucky too - they have the plugin already bundled with the IDE.

GlassFish has also [plugin for Maven](#) and [Ant](#) which are quite well developed, enabling functionalities like deploy, undeploy, start and stop the server.

**Score:** 

### Reason:

Info about where to find plugins is poor, great IDE support once the plugins were found (particularly with NetBeans and Eclipse).

## SCORE SUMMARY



# Server Configuration

## JETTY

You can make your own XML based configuration file and pass it along with the startup command:

```
java -jar start.jar /path/to/your-jetty-conf.xml
```

Or those who are impatient, can just start up their instance with JVM arg settings passing in additional config. Also, it is possible to pass multiple configuration files to Jetty like configuration for HTTP and HTTPS, which is very useful for sharing snippets of config around a team.

Jetty's XML based configuration is reflection-based. This means that all options in XML are actually corresponding to the Java class fields.

The biggest downside with this reflection-based (not so well documented) approach is that you need to understand how Jetty works under the covers and learn some of its internals. But hey - if you know your server well, you'll never run into anomalies and won't let the server to go out of control, right?

**Score:** 

### Reason:

Able to create Jetty config but need to know Jetty internals, restarts often required, JVM args to override config is a nice feature, config is small.

## TOMCAT

Configuration in Tomcat is scattered across a number of files in the tomcat/conf directory but mainly resides in the server.xml file. This file can be modular to allow reuse and sharing across a development team. If you wanted to make a quick change, perhaps one that needed to be undone on next restart for a test, the best way is to add a system property when starting the JVM up as system properties override xml configuration. The default configuration file is quite verbose, but this is actually mostly comment lines with 'how-to's. If you look at the active lines of xml, it's actually a very small file. Every server.xml file will require you to recycle the server as the configuration files are only checked by the core runtime during server startup.

**Score:** 

### Reason:

Restarts required for config changes, scattered across multiple files, small file, easy to update, nice examples in comments.

## JBOSS

The domain model is quite understandable and straightforward, so getting the setup of your dreams is easy. Here is how one would enable ssl connections on some arbitrary port. Locate the following part of the **standalone/configuration/standalone.xml** and add a connector element for "https" (bold line in the following table).

```
<subsystem xmlns="urn:jboss:domain:web:1.1" default-  
virtual-server="default-host" native="false">  
  <connector name="http" protocol="HTTP/1.1" scheme="http"  
socket-binding="http" />  
  <connector name="https" protocol="HTTP/1.1" scheme="https"  
socket-binding="https" enable-lookups="false"  
secure="true"></connector>  
  <virtual-server name="default-host" enable-welcome-  
root="true">  
    <alias name="localhost" />  
    <alias name="example.com" />  
  </virtual-server>  
</subsystem>
```

This will enable connector, but if you also want to change the port where https binds, change "`<socket-binding name="https" port="8443" />`" element to have the desired value.

The whole default config is just 15KB (300 lines) of xml, located in a very predictable location. It also contains a couple of larger sample configuration for clustering. Changes are pretty static, configuration files are read at the start and isn't reread automatically. However, a CLI command:

```
[jboss-as-7.1.3.Final]$ bin/jboss-cli.sh --connect :reload
```

This will do the thing for you without restarting the server.

Score: 

### Reason:

Single file, 300 lines of xml :( Easy to config, well structured "subsystems" configuration, CLI reloads rather than restarts (manual process), several examples out-of-the-box.

## LIBERTY PROFILE

The Liberty Profile has a single config file called server.xml. This file can include other config files if desired, so that the config can be logically separated or shared across a team. The default config file is tiny and the most interesting part is the feature manager:

```
<featureManager>
  <feature>jsp-2.2</feature>
</featureManager>
```

This allows a user to construct their application server with the features they want to run, so initially, only the jsp-2.2 feature is enabled and running. Oh and by the way, you can change and update the server.xml and see the changes reflected in the running server. Neat!

**Score:** 

### Reason:

Top class config model, dynamic updates mean there's no need to restart, small file, simple config.

## GLASSFISH

Everything that is needed to manage and maintain a GlassFish server is put into the asadmin utility. It is even possible to manage remote servers with it, which is cool!

One thing that asadmin does well at is scripting. Yeah – it is possible to write various asadmin scripts for managing your deployments and servers. These scripts can be executed inside asadmin as if you are running regular shell scripts.

For those who are still real fans of XML files, it is possible to configure everything in the domain.xml file too:

```
<network-listener port="8800" protocol="http-listener-1"
transport="tcp" name="http-listener-1" thread-
pool="http-thread-pool"></network-listener>
```

Some configuration changes require that you restart the DAS or GlassFish Server instances for the changes to take effect. Other changes are applied dynamically without requiring that the DAS or instances be restarted.

You can determine whether a domain or instance requires a restart from asadmin

```
asadmin> list-domains
domain1 running, restart required to apply configuration
changes Command list-domains executed successfully.
```

```
asadmin> list-instances pmd-11
pmd-11 running; requires restart
Command list-instances executed successfully.
```

With dynamic configuration, changes take effect while the DAS or instance is running. The following configuration changes of developer interest do not require restart:

- Adding or deleting add-on components
- Adding or removing JDBC, JMS, and connector resources and pools (Exception: Some connection pool properties affect applications.)

- Changing a system property that is not referenced by a JVM option or a port
- Changing logging levels
- Enabling and disabling resources and applications
- Deploying, undeploying, and redeploying applications

**Score:**     

**Reason:**

Full JEE and OSGi support. Neat!

## SCORE SUMMARY





## Documentation & Community

### JETTY

Jetty team has done a lot of work recently on improving and structurizing their docs. Currently there is a “Jetty Documentation Hub” available, which is easy to read and navigate. It features docs for different Jetty versions. There are different mailing lists for Jetty developers users available. No official forums that we could find, though.

To get some professional and commercial support you could find a third party company that can provide this. Unfortunately, Jetty the website does not have a list of such companies available.

**Score:** 

**Reason:**

There is no list of “real experts” or companies, who can provide support, provided.

### TOMCAT

The documentation for Tomcat is very good (particularly for new users) as is supported by a vast community. This same community is where the Tomcat support comes from, both as descriptive help as well as code changes and bug fixes. Most people will be comfortable with this, but if you need more of a guarantee, there are vendor support contracts available, which very often include Tomcat committers.

**Score:** 

**Reason:**

Great, vibrant community, established over years. Responsive, docs are great, site is easy to use, support via the community, 3rd party vendors available for more guaranteed support.

### JBoss

The JBoss community is one of the best things one can think of regarding JBoss AS7. They have a myriad of projects under their wing and they work together well. If not, almost everything can be found on forums or discussion groups. If everything else fails you can read the documentation, which is quite good.

**Score:** 

**Reason:**

Slower releases, but docs are good, large community.

### LIBERTY PROFILE

Documentation can mostly be found on the new community site, WASdev.net. It’s run by the development team so isn’t full of the usual fluff. There is also the traditional info center which provides more formal documentation for the product. You can get support from the WASdev forum which is reasonably active, and again, you’re talking with the development team. This is best can do support, without guarantees. If you choose to pay the big bucks you’ll also get the full IBM support where you can afford to scream at IBM and still get help :o)

**Score:** 

**Reason:**

Good quality docs & sample code, small community, but growing good response times on forum, often very corporate answers as not much can be said about futures, slow bug fixes.

## GLASSFISH

Crawling through the docs, forums and mailing list gives you a secure feeling. The forum is quite active, there are many mailing lists specializing on different parts and areas of GlassFish and the documentation looks pretty good.

**Score:** 

### Reason:

good docs, awkward to navigate to, some docs are outdated, good sized community, mailing lists, active forums.

## SCORE SUMMARY



# PART II

## GOING NINJA

In this section, we go ninja and look at real performance metrics, features & UI, and how much does it cost, if anything?

Hopefully at this point, you didn't just realize that you have a budget of 0 and you were totally sold on WebLogic ;)

So far we've looked at the kind of topics that are important when getting started and learning about application servers. Now we look a little deeper into each application server including how the app servers perform with real developer tasks.

- Real performance metrics
- Features & Open Standards compliance
- Administration & Management/UI
- Cost \$\$\$/Licensing

Keep reading, we're almost at the part where we tell you which app server we think is the best of the best!

## Real performance metrics

Did you know that a 2011 Lamborghini Gallardo LP570-4 Superleggera can travel 0-60 mph in 2.8 seconds but a 2011 Dodge Viper Hennessey Venom GT can only do the same in 2.9 seconds?

Those Viper drivers must get sooo bored of waiting to get to 60! We don't care whether they're wrapped in bacon or not, 0.1 seconds isn't going to change the world, and the average driver isn't going to notice or care. To them they both accelerate stupid-uber-fast.

Similarly, a developer isn't going to choose an application server because it starts up 0.2 seconds faster, or even half a second, as the overall developer experience will be the same. We will therefore score servers based on the experience they will give the user in this section.

Firstly, the applications we use for our performance tests are the tried-and-true Spring Petclinic application that we show JRebel demos on, and a Jenkins application. The PetClinic application is a webapp which uses Servlets, JSPs and a whole bunch of Java POJOs. It also has Hibernate, JPA and JDBC integration code along with the HypersonicSQL DB. The application is just under 20MB in size. Jenkins doesn't need an introduction, as it's now a well known CI framework most people love and use. The web application itself is just under 55MB in size, containing the spring framework and a huge bunch of web artifacts.

## TEST MACHINE

Tests were all run on the same MacBook Air laptop. Spec: 8GB RAM (1600 MHz DDR3), 2GHz Intel Core i7 CPU. The HDD is a 512 GB SSD, OSX 10.8.2. The JRE used is:

```
3. bash
Simons-MacBook-Air:~ maples$ java -version
java version "1.7.0_21"
Java(TM) SE Runtime Environment (build 1.7.0_21-b12)
Java HotSpot(TM) 64-Bit Server VM (build 23.21-b01, mixed mode)
Simons-MacBook-Air:~ maples$
```

## STARTUP/RESTART TIMES

Let's start by looking at the startup and restart times. We ran tests against an empty server, a server with the petclinic application deployed and a server with the jenkins app installed. Here are our results.

	LP	TC	GF	JETTY	JB	AVG
Empty Server Startup time	2	2	4.5	4	3	3.1
Server Startup time w/ petclinic app installed	2.5	6	9	12	7	7.1
Server Restart time with petclinic installed	4.5	8	13	13.5	10.5	9.9
Server Startup time with jenkins app installed	4	9	10	12	15	10
Server Restart time with jenkins installed	6	12	19	14.5	21	14.5

*\*\*Note all figures are in seconds\*\**

You'll notice that the Liberty Profile and Tomcat server are twice as fast as GlassFish and (somewhat surprisingly) Jetty. All the servers are pretty much in and around an acceptable start time. When we start with an application it gets interesting, as the Liberty Profile holds it's time very low, while the other application servers increase dramatically. You can see from the highlighted data that the Liberty Profile has start times anywhere from 2-4 times faster than it's competitors.

	<b>LP</b>	<b>TC</b>	<b>GF</b>	<b>JETTY</b>	<b>JB</b>
Empty Server Startup time	<b>2</b>	2	+2.5	+2	+1
Server Startup time w/ petclinic app installed	<b>2.5</b>	+3.5	+6.5	+8.5	+4.5
Server Restart time with petclinic installed	<b>4.5</b>	+3.5	+8.5	+9	+6
Server Startup time with jenkins app installed	<b>4</b>	+5	+6	+8	+11
Server Restart time with jenkins installed	<b>6</b>	+6	+13	+8.5	+15

If we see how each application server compares against the best in category the clear winner here is the Liberty Profile and very interestingly and surprisingly Jetty performs underwhelmingly, as does Glassfish. JBoss keeps under the radar until the jenkins application exposes it slightly during startup time, being 11-15 seconds slower than the Liberty Profile.

## DEPLOY TIME

Let's take a look at the deploy times, in seconds, of the petclinic and jenkins applications on each of our servers.

	<b>LP</b>	<b>TC</b>	<b>GF</b>	<b>JETTY</b>	<b>JB</b>	<b>AVG</b>
Petclinic app deploy time	<b>1</b>	5	5	6	3	4
Jenkins app deploy time	<b>2</b>	8	8	8	13	7.8

We can see drastic differences between the Liberty Profile and our other servers again, with it outperforming the field by up to 6 times.

	<b>LP</b>	<b>TC</b>	<b>GF</b>	<b>JETTY</b>	<b>JB</b>
Petclinic app deploy time	<b>1</b>	+4	+4	+5	+2
Jenkins app deploy time	<b>2</b>	+6	+6	+6	+6

We can see the Liberty Profile is way ahead of the chasing pack, while JBoss again shows it doesn't like the Jenkins application. Otherwise, results are fairly similar across the other servers

So what is the Liberty Profile doing that the other servers are failing to do? Or rather, what is the Liberty Profile not doing... When we take a look at the application initialization times, we get a better picture of what might be happening.

## APPLICATION INITIALIZATION

	LP	TC	GF	JETTY	JB	AVG
Petclinic Initialization	8	1.5	2	1.5	1.5	2.9
Jenkins Initialization	21	2,5	4	3	2	6.5

Most servers here give a respectable initialization time, while the Liberty Profile shows itself to have a lazy initialization model. It has clearly pushed a lot of its startup/deploy work into the later stages when used for the first time.

	LP	TC	GF	JETTY	JB
Petclinic Initialization	+6.5	1.5	+0.5	1.5	1.5
Jenkins Initialization	+19	+0.5	+2	+1	2

Looking against the averages, we can see JBoss, Tomcat performing very well, along with Jetty and GlassFish close behind, but the Liberty Profile really does stand out here with a painful initialization time of 14.5 seconds \*over\* the average! Ouch! Clearly our application servers are doing different things at different times, so let's these metrics together to see the bigger picture:

	LP	TC	GF	JETTY	JB	AVG
Startup with app & invoke petclinic	10.5	7.5	11	12.5	8.5	10
Startup, deploy & invoke petclinic	11	8.5	11.5	11.5	7.5	10
Startup with app & invoke jenkins	25	11.5	14	15	17	16.5
Startup, deploy & invoke jenkins	25	12.5	16.5	15	18	17.4

Here we can see that the petclinic application isn't so different across the board once everything has been added together. Tomcat, Glassfish and Jetty have comparable times between the two applications, as the difference is only a few seconds, whereas the Liberty Profile and JBoss both take twice as long with the jenkins app than they do with the petclinic application. The notable numbers here though are from the Liberty Profile, taking 25 seconds for the tasks, which is a big difference to Tomcat's 11-12 seconds.

	LP	TC	GF	JETTY	JB
Startup with app & invoke petclinic	+3	<b>7.5</b>	+3.5	+5	+1
Startup, deploy & invoke petclinic	+3.5	+1	+4	+4	<b>7.5</b>
Startup with app & invoke jenkins	13.5	<b>11.5</b>	+2.5	+3.5	+5.5
Startup, deploy & invoke jenkins	12.5	<b>12.5</b>	+4	+2.5	+5.5

JBoss numbers don't look too bad for the jenkins app, but the Liberty Profile clearly has room for improvement. Tomcat is the big winner in this category though, outshining the other servers in almost every task.

## SCORE SUMMARY





## Features & Open Standards compliance

### JETTY

As Jetty itself is largely just a container, it misses many components that would be needed when running more serious apps. There is a Jetty distribution called "Hightide", that provides components and features that are needed to host apps requiring JNDI, JMX, annotations or JEE integration.

Jetty 9 does not yet offer a distribution for Hightide but for those who want to use Hightide, there is a distribution based on Jetty 8 available now: <http://repo1.maven.org/maven2/org/mortbay/jetty/jetty-hightide/>.

Jetty Hightide includes JNDI, JMX, annotations and also JEE integrations. But even with all those awesome features it remains lightweight - Hightide is only 24 megabytes in size. Jetty also offers the option to embed itself into the web application. This creates an executable webapp and is mostly achievable due to the size of the Jetty codebase.

**Score:** 

**Reason:**

Includes WebSockets, Jetty Continuations, Servlet 3.1 spec (almost ready) but not much else.

### TOMCAT

Tomcat is low on features, which often forces developers to craft their own feature set on top of the Tomcat base. But there is new hope in an Apache project called TomEE, which does all of the integration work described out the box, providing an *"all-Apache stack aimed at Java EE 6 Web Profile certification where Tomcat is top dog"*. The last part is key, as the flavour of this project is very much to keep the Tomcat look and feel, experience,

performance as close as possible while providing the extra functionality some Tomcat users would like out the box.

The TomEE Web Profile (35.9MB download) most notably includes, CDI, EJB, JPA, JSF, JTA and Bean Validation support among others. The TomEE+ (55.2 MB download) edition contains JAX-RS, JAX-WS, JMS and Connector support on top of the TomEE Web Profile..

These features make use of Tomcat's underlying infrastructure, for example, servlets now get access to JPA and Transactions, EJBs get access to Tomcat provided Security. Any Tomcat provided resources, say from a context.xml, can be looked up or injected by any managed component in the system. Neat!

Like Jetty, Tomcat isn't feature rich enough to be Web Profile compliant, but what it does have follows the usual web specifications. Typically you bolster what you need on top of Tomcat, and this path typically includes OpenEJB and OpenJPA. But it's really up to you how and what you choose to add.

**Score:** 

**Reason:**

Fairly bare bones features, most people use Tomcat as a base.

### TomEE

**Score:** 

**Reason:**

Completes the Web Profile with Open Source components, all tested together. Ace!

## JBoss

JBoss AS7 is fully EE6 compatible out of the box, we mean Full Profile: EJB, CDI, JPA, JSF, Jax-RS, Bean validation etc. It supports OSGi 4.2, 4.3 is in the works.

From an Open Source compatibility point of view, JBoss 7 is a fully Java EE compatible. Not much can be said here: JAXB, EJB, CDI, anything you throw on it will be handled gracefully. At the same time it is compliant with OSGi version 4.2 and allows you to take the best of both worlds.

**Score:** 

**Reason:**

Full EE6 profile, OSGi support. Neat!

## Liberty Profile

Version 8.5 doesn't quite support the Web Profile as it lacks EJB and CDI support, but does have OSGi application support if that is your favoured technology. The latest beta of the Liberty Profile, however, does contain more support including EJB, CDI, WebServices and really interestingly product extensions, which allow users to create their own 'feature's in the server.xml file. This is due for release in June 2013

IBM is pretty hot on compliance and standards so it's no surprise to see the Liberty Profile following the same path. Although version 8.5 of the Liberty Profile is not compliant with Web Profile, the latest Beta is (due for release June 2013). The Liberty Profile also supports standardised OSGi applications and allows users to create bundles and applications they can deploy and run on the server.

**Score:** 

**Reason:**

Almost Web Profile support (available in the new beta), OSGi support, ability to write your own features coming!

## GLASSFISH

This is an easy section for GlassFish as it is the reference implementation for Java EE. It's therefore a fully compliant Java EE 6 server and again, this includes the full profile: EJB, CDI, JPA, JSF, Jax-RS, Bean validation etc. It also supports OSGi.

From the support of Open Source, similar to JBoss, GlassFish is a fully Java EE compatible server, so... JAXB, EJB, CDI, anything you throw on it will also be handled gracefully.

**Score:** 

### Reason:

Full JEE and OSGi support. Neat!

## SCORE SUMMARY



## Administration & Management/UI

### JETTY

Probably one big drawback is the lack of a management console that you may be used to using with a server like Tomcat (Tomcat Manager) or JBoss AS. With that being said, it may be hard to manage big production environments with Jetty, unless you use an awesome tool like LiveRebel [<http://zeroturnaround.com/software/liverebel/download/>], of course!

**Score:** 

**Reason:**

Management only possible via scripts/changing xml directly.

### TOMCAT

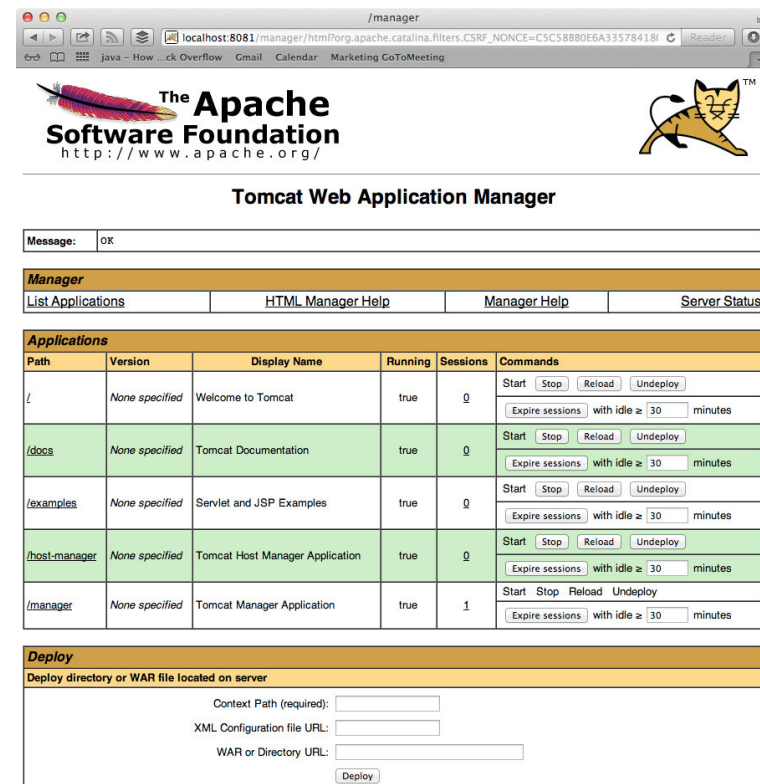
Tomcat provides an administrative webapp called the manager app out the box. You'll need to configure the conf/tomcatusers.xml file to grant access to roles first as access is gated. This is a nice feature as it provides different roles for different levels of admins.

The manager app isn't flashy or anything special, but let's you do what you need, from application deployments and changes to listing OS and JVM properties.

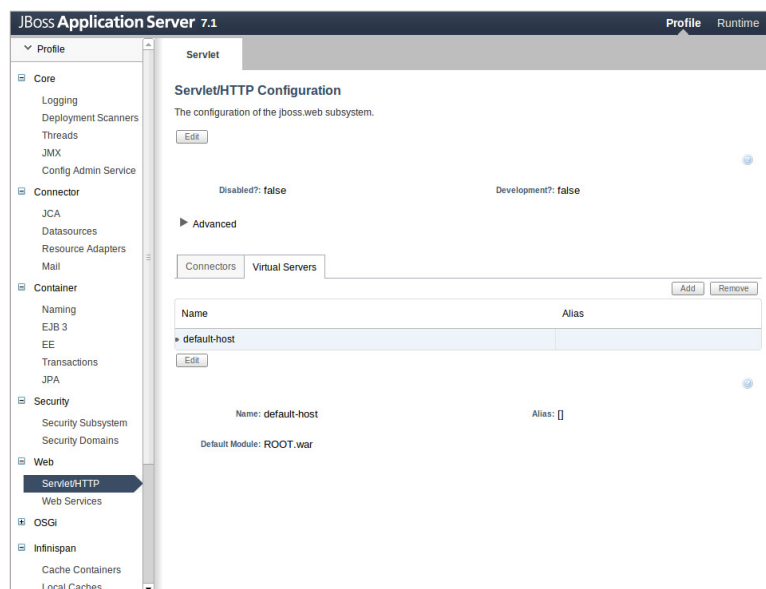
**Score:** 

**Reason:**

Simple, limited functionality, 1980's style console, need to configure xml to add a user before you can even use the manager console.



## JBOSS



The Web console looks ok, much nicer than for example Tomcat's manager application. It allows you to configure data sources, webservices, OSGi, jvm parameters, JPA with transactions and much more. Pretty good choice of things to fiddle with and working with the console application itself is pleasant and fast.

If JBoss cannot reload the changes that you did to it in the web-console, it'll greet you with a message saying that, so you won't wonder why new configuration is not applied.

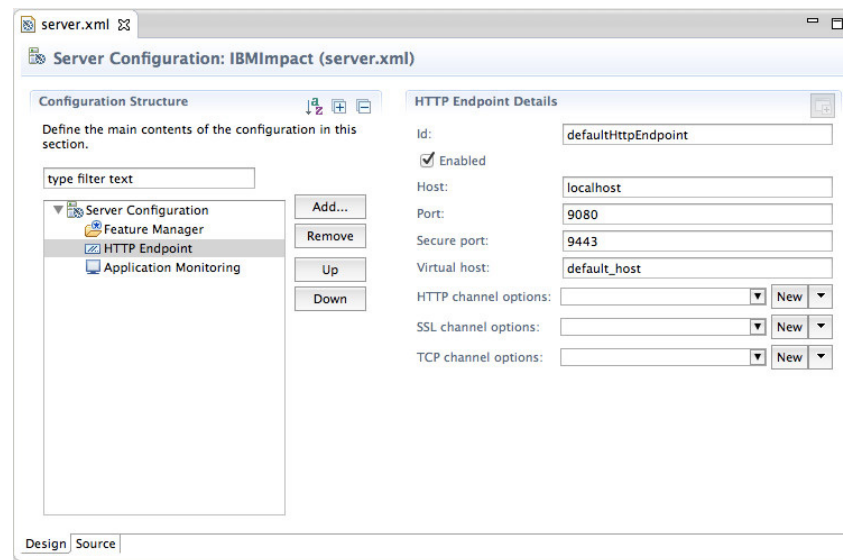
**Score:**

### Reason:

Enterprise quality administration, reload/restart information, need to configure users before use, extensive feature configuration, modern.

## LIBERTY PROFILE

This part sucks a bit. The Liberty Profile does not have an administrative console to update server config or install applications etc so you're stuck with the eclipse editor to update the server.xml.



There is a web app called the Liberty Profile Admin UI Tech Preview, which is a beta download. We had a quick play but it only has extremely basic functionality and has a very mobile app look and feel which doesn't really feel appropriate for browser access. You are able to update server config via the eclipse editor, but this doesn't help everyone.

**Score:**

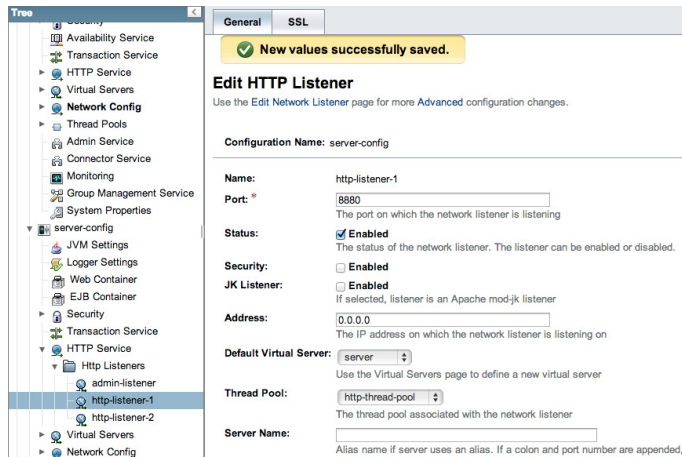
### Reason:

Limited support in a poor admin tech preview, wrong ui format for the browser, UI in Eclipse tools is much much better, but requires the use of Eclipse, not good for remote servers.

## GLASSFISH

GlassFish provides two ways to administer your server or server cluster. You can either use the asadmin utility or it's web-based admin console, that can be reached at port 4848 by default, which is a friendly and easy to use tool with which to make your server changes.

To modify the HTTP port value we need to update a network listener. GlassFish uses network listeners for it's connections with the outside world and thus, also HTTP connections are handled by network listeners called http-listener. The easiest way to modify the HTTP port is through the Web-based admin console:



Wow, that was easy! Despite it being big and enterprise looking, the GlassFish server has a highly modular architecture enabling you to configure it mostly without restarts! We saved the new HTTP port value and now we have our new HTTP port working.

It is possible to create different configurations and apply them on different servers or server clusters. I'd say that this has been done in a very DevOps friendly way.

**Score:**

### Reason:

Clean admin console, not overly complex, extensive feature configuration, maybe a little aged, but couldn't drop from full marks as it's all there.

## SCORE SUMMARY



## Cost \$\$\$ / Licensing

All of the servers we have looked at are free for use in a development environment. This doesn't come as a surprise, but cost is more than just an initial outlay in dollars. It should also include the support you get when things go wrong, from the vendor and/or community, the time and effort spent in infrastructure, including supplementing the features on the application server, testing and future upgrades.

Here are some interesting costs of each of the servers (minimum prices calculated with minimal settings):

SERVER	LICENSE	PRODUCTION	DEVELOPMENT	COMMERCIAL SUPPORT
Jetty	Apache 2.0 / EPL	Free	Free	No
Tomcat	Apache 2.0	Free	Free	No
JBoss AS	GNU LGPL	Free	Free	Yes (EAP)
JBoss EAP	GNU LGPL	\$\$\$	Free	Yes
GlassFish OSE	CDDL 1.1	Free	Free	Yes (Oracle GlassFish Server)
Oracle GlassFish Server	Commercial	\$\$\$	Free	Yes
WAS Liberty Profile	Commercial	\$\$\$	Free	Yes

\$\$\$ - These servers charge for use in production, but it's not the kind of thing you pick up from the grocery store so often has a complex mind-aching pricing structure. Where possible a guide price has been given, but for those with environment specific questions we've linked you to the product specific pricing pages, rather than put misleading numbers down.

## JETTY

As Jetty is very lightweight, anything which you glue to the side of it will be maintained by the framework team you own. Hightide delivers more of the feature you require, in Jetty 9.

Jetty 9 (as well as 7 and 8) is dual licensed under the [Apache License 2.0](#) and [Eclipse Public License 1.0](#). Jetty is free for commercial use and distribution under the terms of either license, with exceptions listed in the [NOTICE](#) file.

**Score:** 

### Reason:

Great license, and it's free! What more could you ask for!

## TOMCAT

Your maintenance of the server will typically be higher as you'll own the deployment which will likely include frameworks of functionality which Tomcat does not include out the box. This is the hidden cost. TomEE is a project out to fix this and will bring the maintenance cost right down. The community support helps here although it is still down to you to perform your own maintenance.

Tomcat is distributed under the very nice Apache 2 license and as a result is embedded in TomEE!, SpringSource tc server and Tcat. It can also be pulled apart and glued back together as has been done with Geronimo and JBoss!

**Score:** 

### Reason:

Awesome Apache 2 license and it costs as much as a compliment - Well done Tomcat :o)

## JBOSS

JBoss AS is an open source product, licensed under LGPL, so it's free to run in development and production. As it's a tested full EE6 profile, many things will work better than with a self-composed soup of libs to support EE stack. And if you like even more stable releases and having a support you can go with an EAP distribution. JBoss EAP is priced on a number of CPUs per year.

**Score:** 

### Reason:

Free for development server, but if you want official support it will come at a price.

## LIBERTY PROFILE

As everything is packed, delivered and tested within IBM, the maintenance is very low. Also if you choose to buy and deploy into production, you'll have the support to lean on if things go wrong.

Having read the license, end to end, during install, twice, you'll already know that there is no license available for using the Liberty Profile outside of development for your hobby usage. So if you wanted to run an app server for a home project, you're not licensed to do so with the Liberty Profile :o( However, this shouldn't count for much in this review as we're only looking at it from a developers point of view.

**Score:** 

### Reason:

License isn't flexible enough to use in a hobby style outside of development for non-commercial, but fine for developers, free to use in development. Tomcat



## GLASSFISH (OPEN SOURCE EDITION)

GlassFish Open Source Edition has CDDL license, so it's free to run in development and production. CDDL license enables [more freedom](#) than GPL and BSD licenses, and in this way it is not risky to use GlassFish in serious projects.

If you want more support and quick bug fixes, it is possible to move on to Oracle GlassFish Server which has better management tools. As GlassFish contains a full JEE implementation there isn't so much maintenance to do in the same way you might need to with Tomcat or Jetty.

**Score:** 

### Reason:

Almost everything packaged in the box with a friendly license and it's freeeeee!

## SCORE SUMMARY



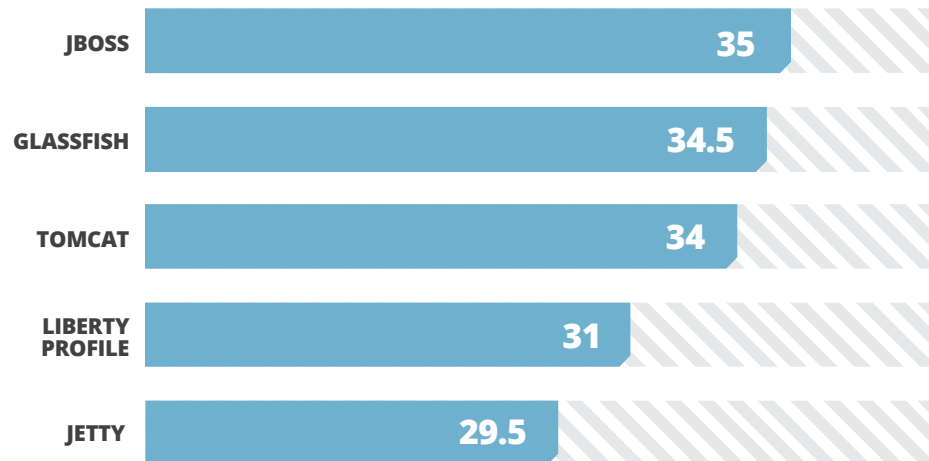
# PART III

## WHICH APP SERVER IS BETTER FOR WHOM?

Now we come to the fun part! Let's collate all our results and make sweeping statements about which Application Server you should all be using, mocking those who are not using what we consider to be their best option... Kind of...

## The Results

"How should we answer the million dollar question - Which is better? We could just add up each of the scores to get a combined total like this:



And say that (in our opinion) JBoss is the best Application Server out there for developers...

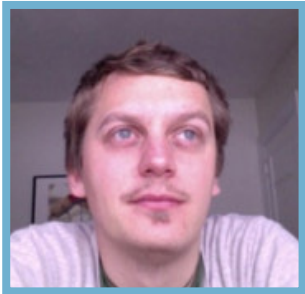
...but that isn't realistic of the situation, as many people would rank performance higher than say installation. So we've created a number of developer profiles and ranked each area with a high->low priority and weighted each of the areas we've looked at to see which application server comes out top of the pile. Is this Scientific? Well, no, but it's fun to see the results :o)

Let's go through our App Developer Profiles one by one and give some example preferences to particular application server features followed by some Application Server recommendations.

Note that areas being assigned High, Medium or Low results in the attached score being scaled by 2, 1 and 0 respectively.

Note: Here are the raw results, combined, in one place:

	LP	TC	GF	JETTY	JB
Download and Installation	4	5	3.5	5	4
Tooling support	4	4.5	4.5	3.5	4.5
Server Configuration	5	4	3	4	4
Real performance metrics	3.5	5	4.5	4	4.5
Features & Open Standards compliance	4	3	5	3	5
Documentation & community	3.5	5	4	3	4.5
Administration & Management/UI	3	2.5	5	2	4.5
Cost \$\$\$/Licensing	4	5	5	5	4
Total	31	34	34.5	29.5	35



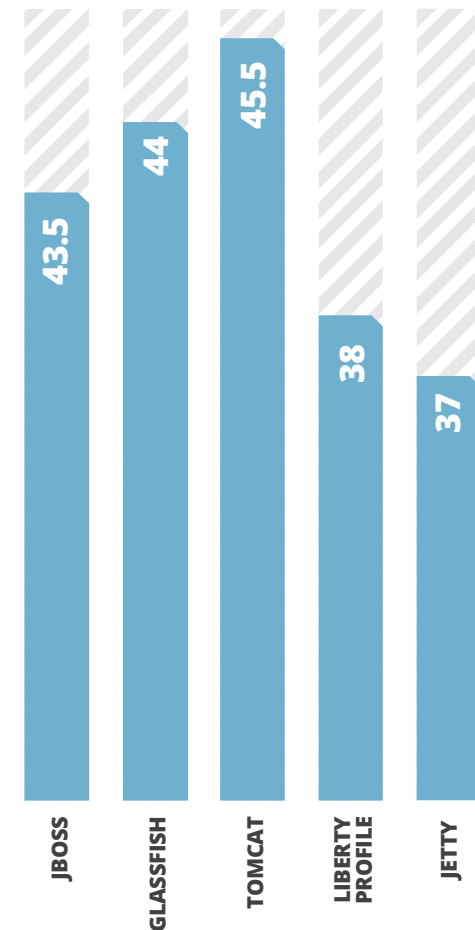
## LJUBOMIR MARIC

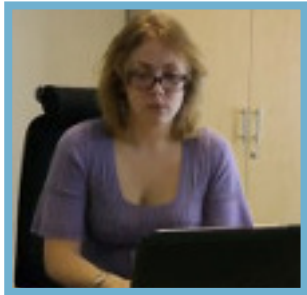
LJ is a hobbyist developer. He likes to play Minecraft and grow root vegetables. He programs in many languages outside of his day job as a Java app developer and has many projects outside of work he enjoys working on, for which he needs application server support. LJ isn't new to application servers, but isn't an expert.

LJ says "This isn't my day job, so I don't want to get bored. Performance is important as I don't have uber powerful kit at home, It also needs to be free, as this my hobby project, not an investment. Community and Docs are also important to me as I don't have a team I can talk to when I need help."

Here are LJ's Priorities and resultant App Server scores:

Priority		LP	TC	GF	JETTY	JB
↓	Download and Installation	4	0	0	0	0
↑	Tooling support	4	9	9	7	9
↔	Server Configuration	5	4	3	4	3
↑	Real performance metrics	3.5	10	9	8	9
↓	Features & Open Standards compliance	4	0	0	0	0
↑	Documentation & community	3.5	10	8	6	8
↔	Administration & Management/UI	3	2.5	5	2	5
↑	Cost \$\$\$/Licensing	4	10	10	10	10
	<b>Total</b>	<b>38</b>	<b>45.5</b>	<b>44</b>	<b>37</b>	<b>43.5</b>





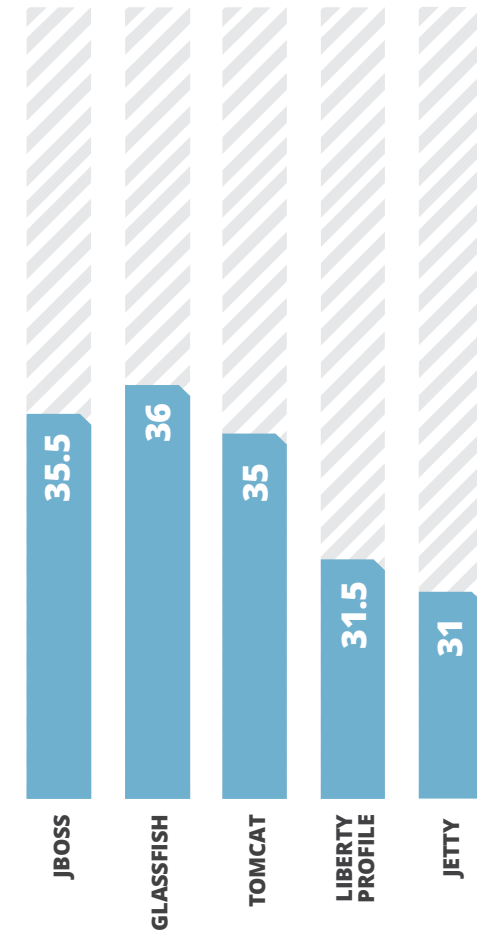
## HEDI PIHLAMÄGI

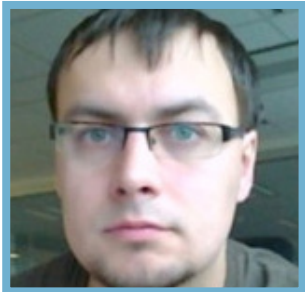
Hedi is a CTO and developer for a newly formed startup. She loves walking and playing roleplay games. She once had dreams of being a [popstar](#), but has given that up for a world record attempt for watching the star wars trilogy more than anyone else. She is up to 4335 viewings and still going strong. The company Hedi works for only have 4 employees, all of which are technical. The startup doesn't have any budget and care mostly about getting results quickly.

Hedi says "We're a startup and every second of every minute counts for us. We need to get as much out as we can for as little investment. Cost and Features and Performance are always important to us. We're on the bleeding edge so need a server that can keep up. Open Standards is also important. If we want to switch vendors at any time, we should be able to. We're a bunch of smart folks, we can work the rest out for ourselves"

Here are Hedi's Priorities and resultant App Server scores:

Priority		LP	TC	GF	JETTY	JB
↓	Download and Installation	0	0	0	0	0
↓	Tooling support	0	0	0	0	0
↔	Server Configuration	5	4	3	4	4
↑	Real performance metrics	7	10	9	8	9
↑	Features & Open Standards compliance	8	6	10	6	10
↔	Documentation & community	3.5	5	4	3	4.5
↓	Administration & Management/UI	0	0	0	0	0
↑	Cost \$\$\$/Licensing	8	10	10	10	8
	<b>Total</b>	<b>31.5</b>	<b>35</b>	<b>31.5</b>	<b>31</b>	<b>35.5</b>





## ANDRES LUUK

Andres has just completed his Computer Science degree at University and loves to play soccer and go out drinking with friends. He dislikes bacon and is ashamed of it. He has just started working for a medium sized company and has started a project where he needs to develop an application to run on an app server, but isn't clear about what he needs.

Andres says "I need support and help more than anything as I'm trying to learn as much as I can. Throwing me in at the deep end is no good. I care about docs, community and ease of use more than anything. Plus, we're not a big company so we try to keep things fairly cheap"

Here are Andres' Priorities and resultant App Server scores:

Priority		LP	TC	GF	JETTY	JB
↓	Download and Installation	0	0	0	0	0
↑	Tooling support	8	9	9	7	9
↑	Server Configuration	10	8	6	8	8
↔	Real performance metrics	3.5	5	4.5	4	4.5
↔	Features & Open Standards compliance	4	3	5	3	5
↑	Documentation & community	7	10	9	6	9
↔	Administration & Management/UI	8	2.5	4.5	2	4.5
↑	Cost \$\$\$/Licensing	4	10	8	10	8
	<b>Total</b>	<b>43.5</b>	<b>47.5</b>	<b>47.5</b>	<b>40</b>	<b>48</b>





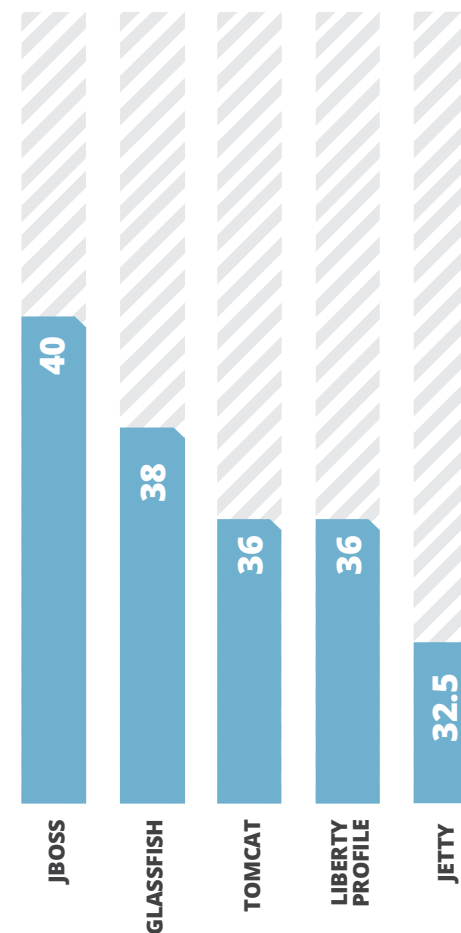
## ARNEL PÄLLO

Arnel is a developer working for a large corporate. He owns six dogs and is afraid his life might go the same way as the movie 'Big'. Arnel believes in Bigfoot and once camped out for 2 months trying to catch a glimpse. Arnel loves being a developer and is part of a large team which produce many applications each with hundreds of components and large footprints.

Arnel says "I work in a team of 150 devs just on this project. First and foremost scripting and automation is very important and sharing across a team. So anything to do with installs, config, etc are important. We almost have a community in our department, so that's not a big issue. Performance and features are very important though as our application is so large, we can't afford to waste time watching it. Cost? I'll let the company worry about that. If I suggest it's right, we'll just budget for it."

Here are Arnel's Priorities and resultant App Server scores:

Priority		LP	TC	GF	JETTY	JB
↔	Download and Installation	4	5	3.5	5	4
↔	Tooling support	4	4.5	4.5	3.5	4.5
↑	Server Configuration	10	8	6	8	6
↑	Real performance metrics	7	10	9	8	9
↑	Features & Open Standards compliance	8	6	10	6	10
↓	Documentation & community	0	0	0	0	0
↔	Administration & Management/UI	3	2.5	5	2	5
↓	Cost \$\$\$/Licensing	0	0	0	0	0
	<b>Total</b>	<b>36</b>	<b>36</b>	<b>38</b>	<b>32.5</b>	<b>40</b>





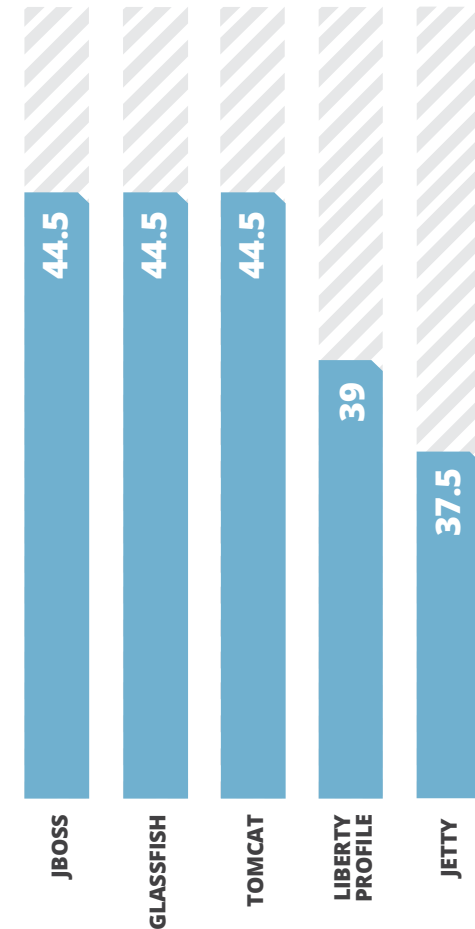
## OLEG ŠELAJEV

Oleg is devoted to Open Source development. He dislikes big corporations and IP and invests more time than is healthy into both boardgames and cross stitch but it looking into a way to combine the two. Oleg loves being part of a vibrant community and believes many hands make light work.

Oleg says “The most important things to me are the Open Standards stack and support, as well as the community. The ability to port my code onto another stack if I want to is very important so I don’t want vendor lock in. Equally, I want to choose my own environment, rather than use what I’m told to use. I’d rather not pay for something the community and I can create, so cost is also important.”

Here are Oleg’s Priorities and resultant App Server scores:

Priority		LP	TC	GF	JETTY	JB
↓	Download and Installation	0	0	0	0	0
↔	Tooling support	4	4.5	4.5	3.5	4.5
↔	Server Configuration	5	4	3	4	4
↑	Real performance metrics	7	10	9	8	9
↑	Features & Open Standards compliance	8	6	10	6	10
↑	Documentation & community	7	10	8	6	9
↓	Administration & Management/UI	0	0	0	0	0
↑	Cost \$\$\$/Licensing	8	10	10	10	8
	<b>Total</b>	<b>39</b>	<b>44.5</b>	<b>44.5</b>	<b>37.5</b>	<b>44.5</b>





# PART IV

## THE PAIR OF ELEPHANTS IN THE ROOM (WEBLOGIC AND WEBSPHERE)

WebSphere and WebLogic servers are a common choice for large scale production environments. They offer High Availability, Scalability and other production like QoS features in abundance, but this doesn't really mean a lot to a developer who is just trying to code their feature request, or fix a bug outside of the production environment.

## Production Focused Application Servers

After considerable debate, we concluded that it's not really fair to compare these application servers to others which have a strong developer focus, so we'll look at these separately. Our developer has a number of options when coding for a production environment which will be using WebSphere or WebLogic:

1. Bite the bullet and use the development distribution of the full WebSphere or WebLogic offerings on their machine
2. Configure the application server up on another, potentially shared machine among devs, and code in your IDE for deployment and test on the remote server
3. Use a more lightweight offering from the same vendor
4. Use an entirely different application server during development and test and later migrate to WebSphere/WebLogic in a staging environment before pushing it to production

Each have their pain, complications and risk. Lets take a look at the last two, first.

## Use a more lightweight offering from the same vendor

IBM and Oracle both know their full enterprise offerings suck in a development environment, if nothing else, purely from a size and resource point of view. They also know developers have the option of using other application servers for other vendors in development and then porting them to production. Neither IBM or Oracle want you to use Tomcat or Jetty in development, so if you wanted a much more lightweight development runtime, they'd want you to use the Liberty Profile or GlassFish if you had to.

The Liberty Profile uses the same containers and frameworks as the full WebSphere Application Server profiles. This means it *should* be easier when moving applications from the Liberty Profile up to the full profile of WebSphere. There are a couple of pain points though, feature support and config. WebSphere hosts a huge number of frameworks and programming models which the Liberty Profile only touches upon in version 8.5 but improves on in it's most recent Beta and release in June 2013. Still, if you're using functionality existing in the full profile which doesn't exist in the Liberty Profile, your app just won't run in the lighter environment. Secondly, there isn't a nice path for mapping config changes you make to the Liberty Profile in your development environment up to the full profile of WebSphere in your production environment, as the config model is so vastly different.

GlassFish is a separate codebase entirely from WebLogic, so you won't get the same cosy feeling as you would with the Liberty Profile relationship with WebSphere. But as with both pairings, you can feel assured that it isn't an unusual path that has never been trodden, so you can have a level of confidence when migrating.

## Use an entirely different application server during development

This is a common scenario, particularly with developers favouring Tomcat and Jetty in their development environment and later migrating up to WebSphere or WebLogic. Should work right? After all, if everyone follows the same spec, what could go wrong? Unless of course specifications were written vaguely by architects who haven't written code in 20 years, leading to different vendors interpreting the spec differently and providing different, non-interoperable implementations, while remaining compliant to the spec and not wanting to change... nah, that'll never happen

\*awkward silence\*.

## What are we left with?

The other options essentially mean the same thing. As a developer we need to write code to run directly on WebSphere or WebLogic, which will inevitably include config changes, setup, server management and possibly download/install. Whether it's due to process, proprietary functionality or masochism, this may be the hand we're dealt. So let's see what they both have to offer.

## Download / Installation

If you hope to read:

1. Download installation package
2. Extract the archive
3. You are ready to go!

You're going to be disappointed! The experience you'd expect (rightly or wrongly) when downloading enterprise software is exactly what you get here. Both Oracle and IBM require you to have an account with them giving them large amounts of information and opting out of all the newsletters

and subscriptions. Once you get to the actual download site you may be pleasantly surprised or enraged depending on your choice.

Oracle offers a number of options, including the >1GB packages with full Enterprise bells and whistles including installer etc. It also provides a lighter weight, unzip & script install for developers at 183MB! Wow, too good to be true? Let's download it and see :)

### Installers with Oracle WebLogic Server, Oracle Coherence and Oracle Enterprise Pack

- ↓ [Linux x86 with 32-bit JVM](#) (1.2 GB)
- ↓ [Windows x86 with 32-bit JVM](#) (1.2 GB)
- ↓ [Mac OS X with 32-bit JVM](#) (1.4 GB)

### Installers with Oracle WebLogic Server and Oracle Coherence:

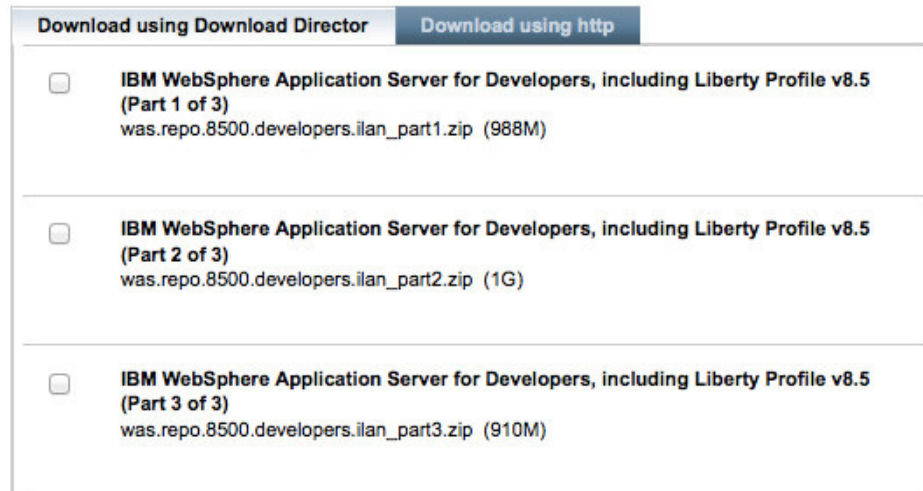
- ↓ [Linux x86 with 32-bit JVM](#) (811 MB)
- ↓ [Windows x86 with 32-bit JVM](#) (800 MB)
- ↓ [Generic](#) (997 MB)

### Zip distribution with Oracle WebLogic Server only and intended for WebLogic Server development only.

- ↓ [Zip distribution for Mac OSX, Windows and Linux](#) (183 MB) | [readme](#)
- ↓ [Supplemental zip distribution](#) (105 MB) | [readme](#)

Once downloaded and unzipped a README.txt file includes all the necessary steps to get up and running. Once you run through these, which only takes a few minutes, you are indeed left with a running WebLogic server. That's very impressive for what many would (without knowing) consider to be a large server that would be the bottleneck in any development environment. We're also left with scripts which we can now just run... oh... it has a slight problem finding Java on MAC OSX Lion, but nothing that a [google search](#) couldn't help with!

Let's try with WebSphere now. We get all the same navigation woes trying to find the right place to download the server from, in fact, it's harder. Once we get to the download site, we're faced with our developer download option:



At first sight, it suggests you have to download a sizable 988MB package to develop with and you groan, then the "Part 1 of 3" bit hits you, followed by 1G and 910MB. The difference between the developer edition and the production ready edition is simply the licensing terms. The bytecode is the same for both offerings which means you get to play with the server in all its glory. Still, let's download and install, but this is where we had to swap development machines, as WebSphere is not supported on Mac - eek. Quite a large population of devs use macs, IBM!

Since version 8, WebSphere has been installed using the Installation Manager which provides value as a tool which orchestrates the installations of your product fix packs and i-fixes. As a developer, it's mostly overhead which isn't particularly useful and is just another step that you need to

go through to get to the end result of having an installation to code your application against.

In terms of features, both WebSphere and WebLogic both always try to keep ahead of the game by implementing the latest JEE spec available. Both provide very similar administration consoles which are ok if you know your way around them, but if you don't and are unsure exactly what you want, you may find yourself deep in questions and options which you don't know the answers to. If you want to change your configuration, your best bet is probably scripting or the admin console, as particularly with WebSphere, the config can get quite hairy.

Tooling support is largely based around a specific IDE. For IBM and WebSphere, it's eclipse and RAD, while for Oracle and WebLogic, it's JDeveloper. Tools matched with the application server are mostly useful as the programming models are also matched, and the ability to manage your server through the tools can be useful.

# PART V

## ...AND THE BEST APPLICATION SERVER AWARD GOES TO...

In case you were wondering, we did finally decide that one application server among those tested proved to win over the others...



# JBoss<sup>TM</sup> AS 7 (aka WildFly) WINS THE AWARD!

In our largely scientific and flawless comparison, JBoss, Tomcat and Glassfish did very well across the board in our developer profiling exercise, taking 3, 2, and 2 wins (including joint wins) in total.

Jetty was the surprising server for us that struggled throughout the developer profile exercise. It's low scores on Features, Admin and Community made it struggle. If as an experienced developer you know what you want from Jetty and are just developing webapps, it's still a very good choice. It did ok in the performance tests and has such a small footprint. It's a bit mean comparing Jetty to the larger, more complete, application servers such as JBoss. Jetty is really for the web developer who doesn't need all the extra bells and whistles and want to get some hardcore coding done in an uber-lightweight environment. It's so lightweight you can even embed the Jetty runtime inside your app - that rocks!

The Liberty Profile Suffered from the Admin, Performance and Community. The additional tooling support which allows you to configure the server.xml saved it from dropping lower, while the difference in app initialization and request times stood it out against the other servers. The community is still quite small in comparison to the other servers, but is growing. Let's remember that this is the first lightweight release of WebSphere from the full profile and impressively it's already competing with existing lightweight containers. With another release of the Liberty Profile due imminently (June 2013), it's certainly one to watch going forward.

Tomcat stuttered on Features and Admin during our tests, while it was the clear winner in community and performance. With TomEE bringing the extra features Tomcat lacks, which many developers bolster on the side, it is an up and coming server which deserves attention. Tomcat fared very well in our performance tests and clearly has a huge community backing and support which has made it one of the most successful developer platforms available today.

GlassFish really only suffered on the download/Installation and Server configuration sections. It performed well on the performance tests and have a full JEE feature set as well as having a first class admin environment. Tooling is very good across the board and well worth considering for future projects requiring an app server of all varieties.

If we had to pick a winner, which we really should and will, it would be JBoss. The only application server in the group whose score never dropped below a 4, and interestingly the received the joint fewest top marks of 5 in the categories. JBoss consistently performs very well in each category which is why it also shines in the developer profiles exercise. Yes it is a bigger download and larger in memory consumption than most, but performance wise for a developer, it doesn't show.

**TOO LONG,  
DIDN'T READ (TL;DR)**

SUMMARY, CONCLUSION, FAREWELL  
AND A COMIC

## Summary of Findings and a Goodbye Comic ;-)

### Download and Installation

All Application Servers see the need for lowering the barriers to access and provide easy download and installation procedures.

### Tooling support

Most favoured Eclipse, but Tomcat, JBoss and GlassFish all had excellent support for the big 3 IDEs.

### Server Configuration

The Liberty Profile won here with its new dynamic model of hot reloading both the runtime components and config. Jetty and Tomcat were also strong.

### Documentation & community

Tomcat has a big, vibrant and established community that beats the competition hands down. Docs are also very good.

### Real performance metrics

Tomcat outperformed the others in our developer oriented performance tests. The Liberty Profile suffered long initialization times, while the others chased Tomcat.

### Features & Open Standards compliance

JBoss and GlassFish rule the feature war, both being JEE 6 compliant and supporting OSGi applications. Jetty and Tomcat are happier remaining as lightweight as possible as web containers, while the Liberty Profile, looks to be growing each year.

### Administration & Management/UI

JBoss and GlassFish both provide top class admin consoles, while Jetty doesn't ship an ounce of UI. The Liberty Profile and Tomcat do provide glimpses of admin but not far enough.

### Cost \$\$\$/Licensing

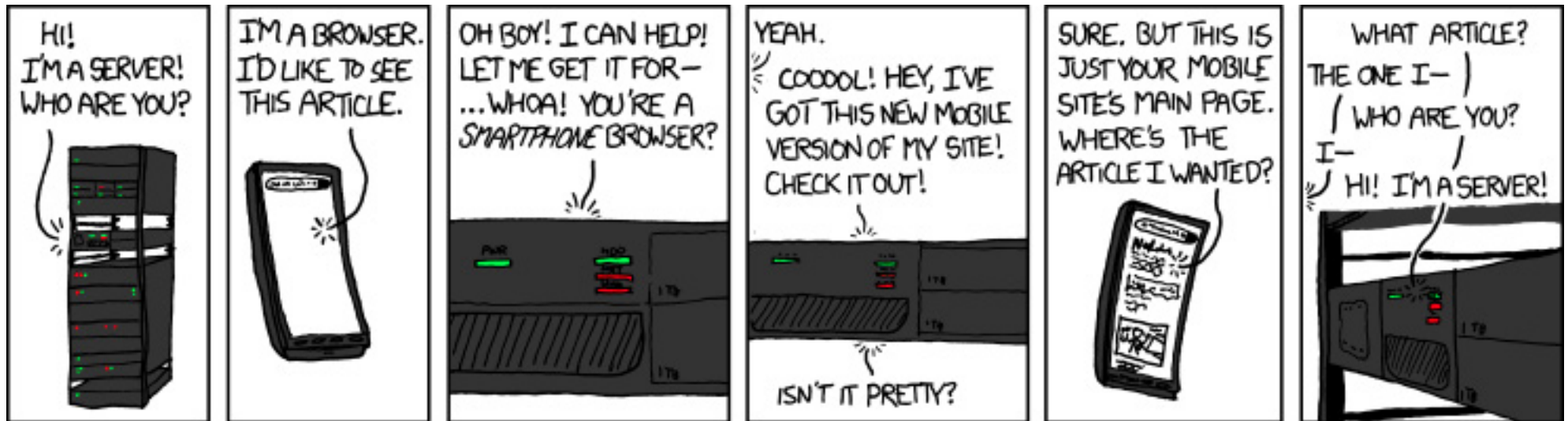
All servers are free for the developer - WIN! The Liberty Profile is a little more restrictive when it comes to licenses.

**WebSphere and WebLogic both prove to be 'enterprise' from download all the way to production.**

**THE GREAT DEBATE IS OVER!**  
**JBOSS WINS,**  
**FOLLOWED BY GLASSFISH**  
**& TOMCAT.**  
**WIN FOR OSS!**



## Goodbye comic



the comic: <http://xkcd.com/869/>



Rebellabs is the research & content  
division of ZeroTurnaround

Contact Us

Twitter: @RebelLabs

Web: <http://zeroturnaround.com/rebellabs>

Email: [labs@zeroturnaround.com](mailto:labs@zeroturnaround.com)

#### Estonia

Ülikooli 2, 5th floor  
Tartu, Estonia, 51003  
Phone: +372 740 4533

#### USA

545 Boylston St., 4th flr.  
Boston, MA, USA, 02116  
Phone: 1(857)277-1199

#### Czech Republic

Osadní 35 - Building B  
Prague, Czech Republic 170 00  
Phone: +372 740 4533

#### This report is brought to you by:

Simon Maple, Sigmar Muuga, Oleg Šelajev,  
Ryan St. James & Oliver White