Community Experience Distilled

Java EE 7 with GlassFish 4 Application Server

A practical guide to install and configure the GlassFish 4 application server and develop Java EE 7 applications to be deployed to this server

David R. Heffelfinger

[PACKT] PUBLISHING open source* community experience distilled

Java EE 7 with GlassFish 4 Application Server

A practical guide to install and configure the GlassFish 4 application server and develop Java EE 7 applications to be deployed to this server

David R. Heffelfinger

[PACKT] open source
community experience distilled
PUBLISHING

BIRMINGHAM - MUMBAI

Java EE 7 with GlassFish 4 Application Server

First published: October 2007

Second Edition: July 2010

Third Edition: March 2014

Production Reference: 1200314

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-78217-688-6

www.packtpub.com

Cover Image by Aniket Sawant (aniket_sawant_photography@hotmail.com)

Credits

Author
David R. Heffelfinger

Reviewers
Stefan Horochovec
Tim Pinet
Chirag Sangani

Acquisition Editors
Subho Gupta
Rubal Kaur

Content Development Editor
Akshay Nair

Technical Editors
Pratik More
Humera Shaikh
Rohit Kumar Singh
Pratish Soman

Copy Editors
Tanvi Gaitonde
Dipti Kapadia
Aditya Nair
Kirti Pai
Stuti Srivastava

Project Coordinator
Amey Sawant

Proofreaders
Maria Gould
Sandra Hopper
Linda Morris

Indexers
Mehreen Deshmukh
Rekha Nair

Graphics
Yuvraj Mannari

Production Coordinator
Aparna Bhagat

Cover Work
Aparna Bhagat

About the Author

David R. Heffelfinger is the Chief Technology Officer at Ensode Technology, LLC, a software consulting firm based in the Greater Washington DC area. He has been architecting, designing, and developing software professionally since 1995. He has been using Java as his primary programming language since 1996. He has worked on many large-scale projects for several clients including the U.S. Department of Homeland Security, Freddie Mac, Fannie Mae, and the U.S. Department of Defense. He has a master's degree in Software Engineering from Southern Methodist University. David is the Editor-in-chief of Ensode.net (`http://www.ensode.net`), a website on Java, Linux, and other technologies. David is a frequent speaker at Java conference such as JavaOne. You can follow David on Twitter, `@ensode`.

About the Reviewers

Stefan Horochovec is from Brazil. He has a graduate degree in Software Engineering and also in Project Management and currently works as a software architect.

Over the past 10 years, he has been dedicated to the development of Enterprise Applications using Java as the backend technology and application servers, such as GlassFish, JBoss, Weblogic, and WildFly.

With regards to frontend, Stefan has worked for 4 years with technologies such as Apache Flex (speaking for three consecutive years at FlexMania, the biggest event on Apache Flex in Latin America), Struts, and JSF. Today, his focus is on projects involving JSF 2 and JavaScript frameworks, with a strong focus on AngularJS.

He has worked with the mobile world for about 6 years, having extensive experience on the Android platform. He was one of the first Android instructors in Brazil and a speaker at the Android conference in Brazil. For about 2 years, he has been working with the HTML-based mobile development using frameworks such as PhoneGap to build enterprise applications.

In 2014, Stefan was invited to join the BlackBerry Elite Member program, which gathers around 100 people worldwide, emphasizing the importance of mobile development, technologies for their development, and using the operating system and BlackBerry devices on the mobile platform.

Stefan also teaches in University courses related to web and Mobile development and is an instructor of in-company courses related to Java, HTML/JS/CSS3, PhoneGap, Git, and Java application servers.

Tim Pinet is a practicing software engineer and web developer currently residing in Ottawa, Canada. From an early age, he was always fascinated with all electronic things and went on to graduate with a bachelor's degree in Engineering in the Software Engineering stream. As Ottawa is a large capital city with a technology sector rich with opportunity, Tim has had the fortune to practice software engineering and systems integration in both private (Computer Associates, Emergis, Telus, Nortel) and public (City of Ottawa) companies and in numerous industries such as transportation and road/weather information systems, healthcare recording, communications and telephony infrastructure, and municipal citizen-centric services and payment handling.

Tim's open source mantra helps him to focus on working for low cost, but high productivity in any environment and has him giving back to projects (such as Apache and SourceForge) and community knowledge bases (such as Stackoverflow and his personal blog). He has brought open source tools to his employers, saving them thousands of dollars and giving them best-practice accelerated development and testing capabilities without giving up dollars or quality.

Loving all things software and web, Tim constantly indulges himself in the newest technologies to better improve service to the end client. He has a vast experience in Java using enterprise technologies, web services, client GUI development, server backend development, database management integration, and SOA services integration. He is a very focused team player and works best in leading teams and architecting solutions.

Chirag Sangani is a computer scientist living in the Seattle area. He obtained his MS from Stanford University, CA, and his B. Tech. from IIT Kanpur, India. He currently works as a software development engineer for Microsoft.

www.PacktPub.com

Support files, eBooks, discount offers and more

You might want to visit `www.PacktPub.com` for support files and downloads related to your book.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at `www.PacktPub.com` and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at `service@packtpub.com` for more details.

At `www.PacktPub.com`, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

`http://PacktLib.PacktPub.com`

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can access, read and search across Packt's entire library of books.

Why Subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print and bookmark content
- On demand and accessible via web browser

Free Access for Packt account holders

If you have an account with Packt at `www.PacktPub.com`, you can use this to access PacktLib today and view nine entirely free books. Simply use your login credentials for immediate access.

Instant Updates on New Packt Books

Get notified! Find out when new books are published by following `@PacktEnterprise` on Twitter, or the *Packt Enterprise* Facebook page.

Table of Contents

Preface

Java Enterprise Edition 7, the latest version of Java EE, adds several new features to the specification. Several existing Java EE APIs have gone through major improvements in this version of the specification; additionally, some brand new APIs have been added to Java EE. This book includes coverage of the latest versions of the most popular Java EE specifications, including JavaServer Faces (JSF), Java Persistence API (JPA), Enterprise JavaBeans (EJB), Contexts and Dependency Injection (CDI), the new Java API for JSON Processing (JSON-P), WebSocket, the completely revamped Java Messaging Service (JMS) API 2.0, the Java API for XML Web Services (JAX-WS) and the Java API for RESTful Web Services (JAX-RS), as well as securing Java EE applications.

The GlassFish application server is the reference implementation for Java EE; it is the first Java EE application server in the market to support Java EE 7. This book covers GlassFish 4.0, the latest version of this powerful open source application server.

What this book covers

Chapter 1, *Getting Started with GlassFish*, explains how to install and configure GlassFish. Deploying Java EE applications through the GlassFish web console are also explained. Finally, basic GlassFish administration tasks such as setting up domains and database connectivity by adding connection pools and data sources are discussed.

Chapter 2, *JavaServer Faces*, covers development of web applications using JSF, including new features such as HTML5-friendly markup and Faces Flows. It also covers how to validate user input using JSF's standard validators and also by creating our own custom validators or by writing validator methods.

Chapter 3, *Object Relational Mapping with JPA*, discusses how to develop code that interacts with a Relational Database Management System (RDBMS) such as Oracle or MySQL through the Java Persistence API.

Chapter 4, *Enterprise JavaBeans*, explains how to develop applications using both session and message-driven beans. Major EJB features such as transaction management, the EJB timer service, and security are covered. The life cycle of the different types of Enterprise JavaBeans are covered, including an explanation of how to have EJB methods automatically invoked by the EJB container at certain points in the life cycle.

Chapter 5, *Contexts and Dependency Injection*, provides an introduction to Contexts and Dependency Injection (CDI). The chapter covers CDI named beans, dependency injection using CDI, and CDI qualifiers.

Chapter 6, *JSON Processing with JSON-P*, covers how to generate and parse JavaScript Object Notation (JSON) data using the new JSON-P API. It also covers both APIs for processing JSON: the Model API and the Streaming API.

Chapter 7, *WebSockets*, explains how to develop web-based applications that feature full duplex communication between the browser and the server as opposed to relying on the traditional HTTP request/response cycle.

Chapter 8, *The Java Message Service*, covers how to set up JMS connection factories, JMS message queues, and JMS message topics in GlassFish using the GlassFish web console. The chapter also discusses how to develop messaging applications using the completely revamped JMS 2.0 API.

Chapter 9, *Securing Java EE Applications*, covers how to secure Java EE applications through provided security realms as well as how to add custom security realms.

Chapter 10, *Web Services with JAX-WS*, covers how to develop web services and web service clients via the JAX-WS API. Web service client code generation using ANT or Maven as a build tool has been explained.

Chapter 11, *Developing RESTful Web Services with JAX-RS*, discusses how to develop RESTful Web services via the Java API for RESTful Web services as well as how to develop RESTful Web service clients via the brand new standard JAX-RS client API. It also explains how to automatically convert data between Java and XML by taking advantage of the Java API for XML Binding (JAXB).

What you need for this book

The following software needs to be installed to follow the material in this book:

- The Java Development Kit (JDK) 1.7 or newer
- GlassFish 4.0
- Maven 3 or newer is needed to build the examples
- A Java IDE such as NetBeans, Eclipse, or IntelliJ IDEA (optional, but recommended).

Who this book is for

This book assumes familiarity with the Java language. The target market for this book is the existing Java developers who wish to learn Java EE and the existing Java EE developers who wish to update their skills to the latest Java EE specification.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "The `@Named` class annotation designates this bean as a CDI named bean."

A block of code is set as follows:

```
if (!emailValidator.isValid(email)) {
  FacesMessage facesMessage = new FacesMessage(htmlInputText.
getLabel()
+ ": email format is not valid");
  throw new ValidatorException(facesMessage);
}
```

When we wish to draw your attention to a particular part of a code block, the relevant lines or items are set in bold:

```
<ejb>
    <ejb-name>CustomerDaoBean</ejb-name>
    <ior-security-config>
        <as-context>
            <auth-method>username_password</auth-method>
```

```
                    <realm>file</realm>
                    <required>true</required>
                </as-context>
            </ior-security-config>
        </ejb>
```

Any command-line input or output is written as follows:

```
$ ~/GlassFish/glassfish4/bin $ ./asadmin start-domain
Waiting for domain1 to start ........
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "Clicking on the **Next** button moves you to the next screen."

Warnings or important notes appear in a box like this.

Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to `feedback@packtpub.com`, and mention the book title via the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on `www.packtpub.com/authors`.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at `http://www.packtpub.com`. If you purchased this book elsewhere, you can visit `http://www.packtpub.com/support` and register to have the files e-mailed directly to you.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting `http://www.packtpub.com/submit-errata`, selecting your book, clicking on the **errata submission form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from `http://www.packtpub.com/support`.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at `questions@packtpub.com` if you are having a problem with any aspect of the book, and we will do our best to address it.

1
Getting Started with GlassFish

In this chapter, we will discuss how to get started with GlassFish. The following are some of the topics discussed in this chapter:

- An overview of Java EE and GlassFish
- Obtaining GlassFish
- Installing and starting GlassFish
- Explaining the concept of GlassFish domains
- Deploying Java EE applications
- Setting up Database Connectivity

An Overview of Java EE and GlassFish

Java Enterprise Edition (**Java EE**, formerly called **J2EE** or **Java 2 Enterprise Edition**) is a standard set of technologies for server-side Java development. Java EE technologies include **JavaServer Faces** (**JSF**), **Enterprise JavaBeans** (**EJBs**), the **Java Messaging Service** (**JMS**), the **Java Persistence API** (**JPA**), the Java API for WebSocket, **Contexts and Dependency Injection** (**CDI**), the **Java API for XML Web Services** (**JAX-WS**), the **Java API for RESTful Web Services** (**JAX-RS**), and the **Java API for JSON Processing** (**JSON-P**), among others.

Several commercial and open source application servers exist. Java EE application servers allow developers to develop and deploy Java EE-compliant applications, GlassFish being one of them. Other open source Java EE application servers include Red Hat's WildFly (formerly JBoss), the Apache Software Foundation's Geronimo, and ObjectWeb's JOnAS. Commercial application servers include Oracle's WebLogic, IBM's WebSphere, and the Oracle Application Server.

GlassFish is the Java EE 7 reference implementation; as such, it implements the latest Java EE APIs before any other application server in the market. GlassFish is open source and freely available, and is licensed under the **Common Development and Distribution License** (**CDDL**).

You can find out more about the CDDL license at `http://opensource.org/licenses/CDDL-1.0`.

Like all Java EE-compliant application servers, GlassFish provides the necessary libraries to allow us to develop and deploy Java applications compliant with Java EE specifications.

What's new in Java EE 7?

Java EE 7, the latest version of the Java EE specification, includes several improvements and additions to the specification. The following sections list the major improvements to the specifications that are of interest to enterprise application developers:

JavaServer Faces (JSF) 2.2

Java EE 7 includes a new version of the **JavaServer Faces** (**JSF**) specification. JSF 2.2 includes the following notable new features:

- JSF 2.2 features the HTML5 friendly markup, that is, web pages can be written using the standard HTML 5 markup and using JSF-specific attributes on the HTML tags.
- JSF 2.2 also includes Faces Flows, which provides a way to encapsulate related pages with defined entry and exit points.
- Resource library contracts are the third major JSF feature introduced in JSF 2.2. Resource library contracts allow us to easily develop web applications that can have a different look and feel for different users using JSF.

Java Persistence API (JPA) 2.1

JPA was introduced as a standard part of Java EE in version 5 of the specification. JPA replaced entity beans as the standard object relational mapping framework for Java EE. JPA adopted ideas from third-party object relational frameworks such as Hibernate and JDO, and made them a part of the standard.

JPA 2.1 introduces the following new features:

- JPA 2.1 introduces the concept of **Converters**, which allows custom code conversions between values stored in the database and values stored in Java objects. For instance, a common problem when working with database data is that the desired value in Java code differs from the value stored in the database. For example, the values `1` and `0` are commonly stored in the database to denote `true` and `false` respectively. Java has a perfectly good boolean type, so `true` and `false` can be used directly.
- The JPA Criteria API can now perform bulk updates and deletes.
- JPA 2.1 now supports stored procedures.
- JPA 2.1 introduces the `@ConstructorResult` annotation, which allows returning standard Java classes (but not the JPA entities) from native SQL queries.

Java API for RESTful Web Services (JAX-RS) 2.0

JAX-RS is a Java API for developing RESTful web services. RESTful web services use the **Representational State Transfer** (**REST**) architecture. Java EE 6 adopted JAX-RS as an official part of the Java EE specification.

JAX-RS 2.0 includes the following new features:

- JAX-RS 2.0 introduces a new client-side API. While previous versions of JAX-RS made it easy to develop RESTful web services, each implementation defined its own proprietary client-side API.
- Extension points, method filters, and entity interceptors are also introduced in JAX-RS 2.0. These features allow **Aspect Oriented Programming** (**AOP**) when developing RESTful web services.
- JAX-RS 2.0 also introduces asynchronous processing both on the server side and as part of the client API.

Java Message Service (JMS) 2.0

The **Java Message Service** (**JMS**) API has been completely revamped in Java EE 7. Previous versions of JMS required lots of boilerplate code; with the new revamped JMS 2.0 API, this is no longer the case.

Java API for JSON Processing (JSON-P) 1.0

JSON-P is a brand new API introduced in Java EE 7. JSON-P allows us to parse and generate **JSON** (**JavaScript Object Notation**) strings.

Java API for WebSocket 1.0

Traditional web applications use a request-response model, that is, a client (typically a web browser) requests resources and the server provides a response. In this model, communication is always initiated by the client.

WebSockets were introduced as part of the HTML5 specification; they provide full-duplex communication between the client and the server.

GlassFish advantages

With so many options in Java EE application servers, why choose GlassFish? Besides the obvious advantage of GlassFish being available free of charge, it offers the following benefits:

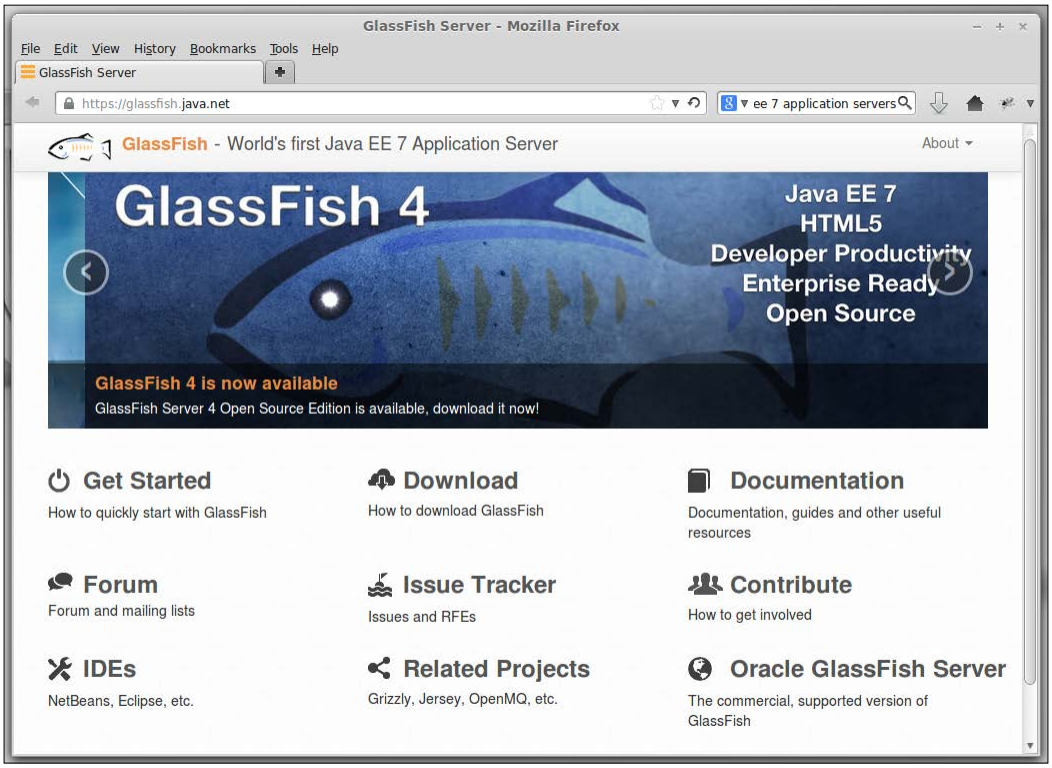
- **Java EE reference implementation**: GlassFish is the Java EE reference implementation. What this means is that other application servers may use GlassFish to make sure their product complies with the specification. GlassFish could theoretically be used to debug other application servers. If an application deployed under another application server is not behaving properly, but it does behave properly when deployed under GlassFish, then more than likely the improper behavior is due to a bug in the other application server.
- **Supports the latest versions of the Java EE specification**: Since GlassFish is the reference Java EE specification, it tends to implement the latest specifications before any other application server in the market. As a matter of fact, at the time of writing, GlassFish is the only Java EE application server in the market that supports the complete Java EE 7 specification.

Obtaining GlassFish

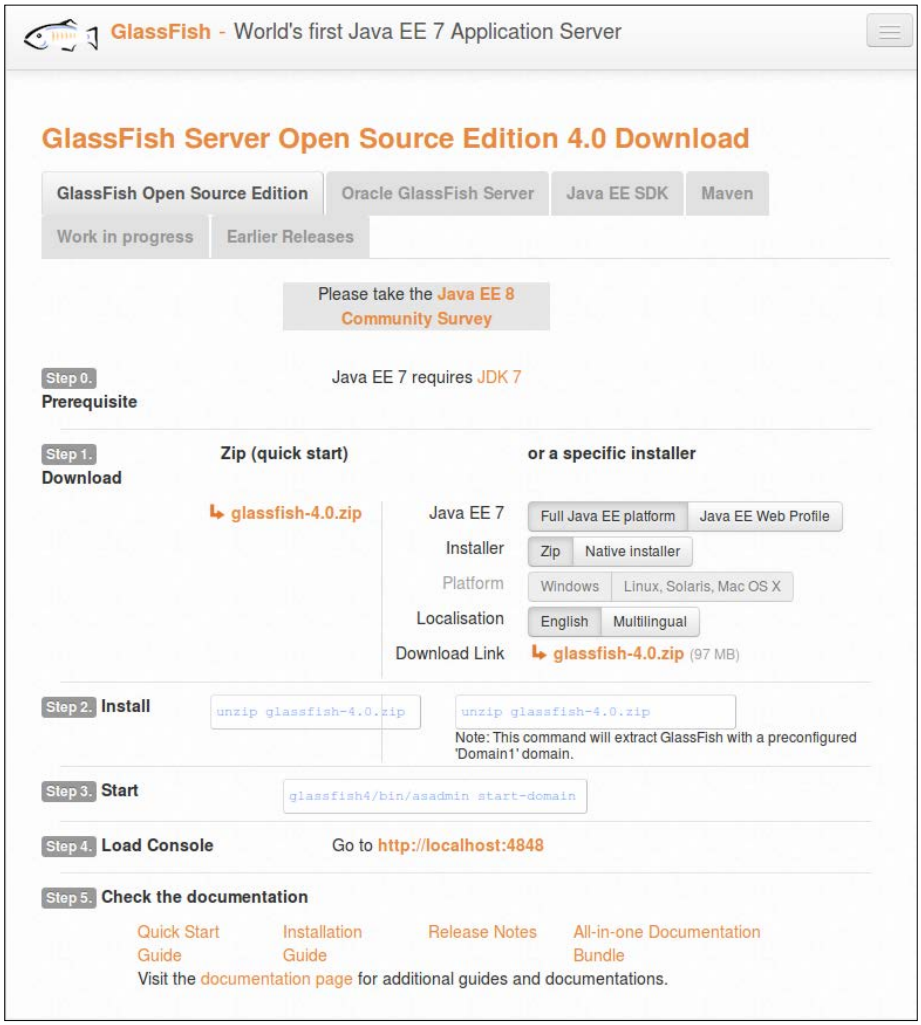
GlassFish can be downloaded at `https://glassfish.java.net`.

GlassFish 4.0 is also bundled with the NetBeans IDE version 7.4 or newer.

Once there, you will see a window as shown in the following screenshot:



Clicking on the **Download** link takes us to a wizard page that provides several options to download GlassFish as shown in the following screenshot:



The download page has several options; we can get the full Java EE platform or the web profile. We can also download GlassFish as a compressed ZIP file or as a native installer for the operating system of our choice.

To be able to follow all of the examples in this book, we need to download the full Java EE platform version of GlassFish. We will download the compressed ZIP file version since the instructions to install it are very similar across any operating system; feel free to download a platform-specific installer if you prefer.

Installing GlassFish

We will use the ZIP installer to illustrate the installation process. This installation process works under all major operating systems.

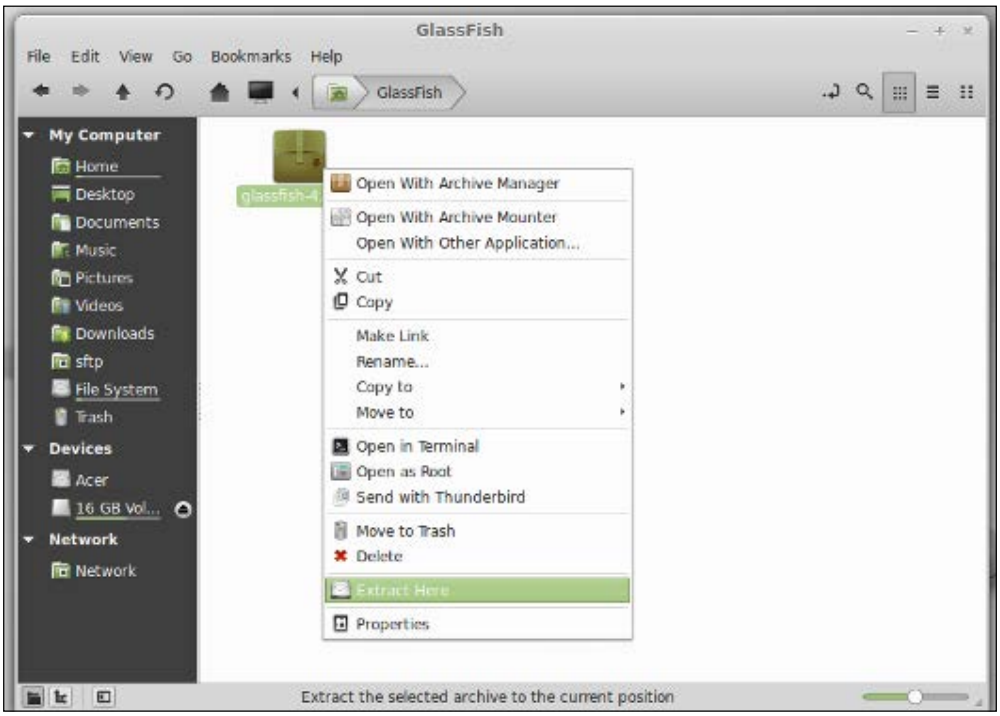
Installing GlassFish is an easy process; however, GlassFish assumes that some dependencies are present in your system.

GlassFish dependencies

In order to install GlassFish 4, a recent version of the **Java Development Kit** (**JDK**) must be installed on your workstation (JDK 1.7 or newer required), and the Java executable file must be in your system PATH. The latest JDK can be downloaded at `http://www.oracle.com/technetwork/java/javase/downloads/index.html`. Please refer to the JDK installation instructions for your particular platform at `http://docs.oracle.com/javase/7/docs/webnotes/install/index.html`.

Performing the installation

Once JDK has been installed, the GlassFish installation can begin by simply extracting the download compressed file as shown in the following screenshot:



All modern operating systems including Linux, Windows, and Mac OS X include out-of-the-box support to extract compressed ZIP files; consult your operating system documentation for details.

After extracting the ZIP file, a new directory named `glassfish4` will be created. This new directory contains our GlassFish installation.

Starting GlassFish

To start GlassFish from the command line, change your directory to `[glassfish installation directory]/glassfish4/bin` and execute the following command:

```
./asadmin start-domain domain1
```

The preceding command, and most commands shown in this chapter, assume a Unix or Unix-like operating system such as Linux or Mac OS. For Windows systems, the initial `./` is not necessary.

A few short seconds after executing the preceding command, we should see a message similar to the following at the bottom of the terminal:

```
$ ~/GlassFish/glassfish4/bin $ ./asadmin start-domain
Waiting for domain1 to start ........
Successfully started the domain : domain1
domain  Location: /home/heffel/GlassFish/glassfish4/glassfish/domains/
domain1
Log File: /home/heffel/GlassFish/glassfish4/glassfish/domains/domain1/
logs/server.log
Admin Port: 4848
Command start-domain executed successfully.
```

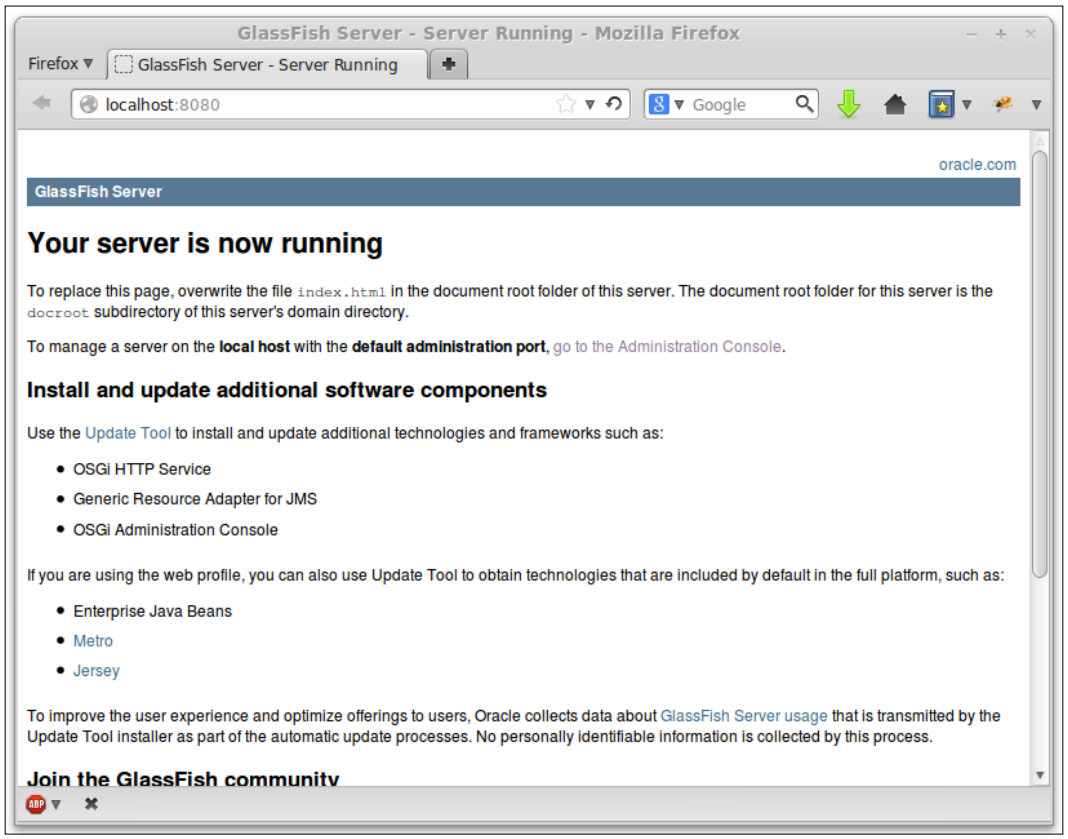
Downloading the example code

You can download the sample code files for all the Packt books that you have purchased from your account at `http://www.packtpub.com`. If you purchased this book elsewhere, you can visit `http://www.packtpub.com/support` and register to have the files e-mailed directly to you.

We can then open a browser window and type the following URL in the browser's location text field:

```
http://localhost:8080
```

If everything goes well, we should see a page indicating that your GlassFish server is now running as shown in the following screenshot:



Getting Help

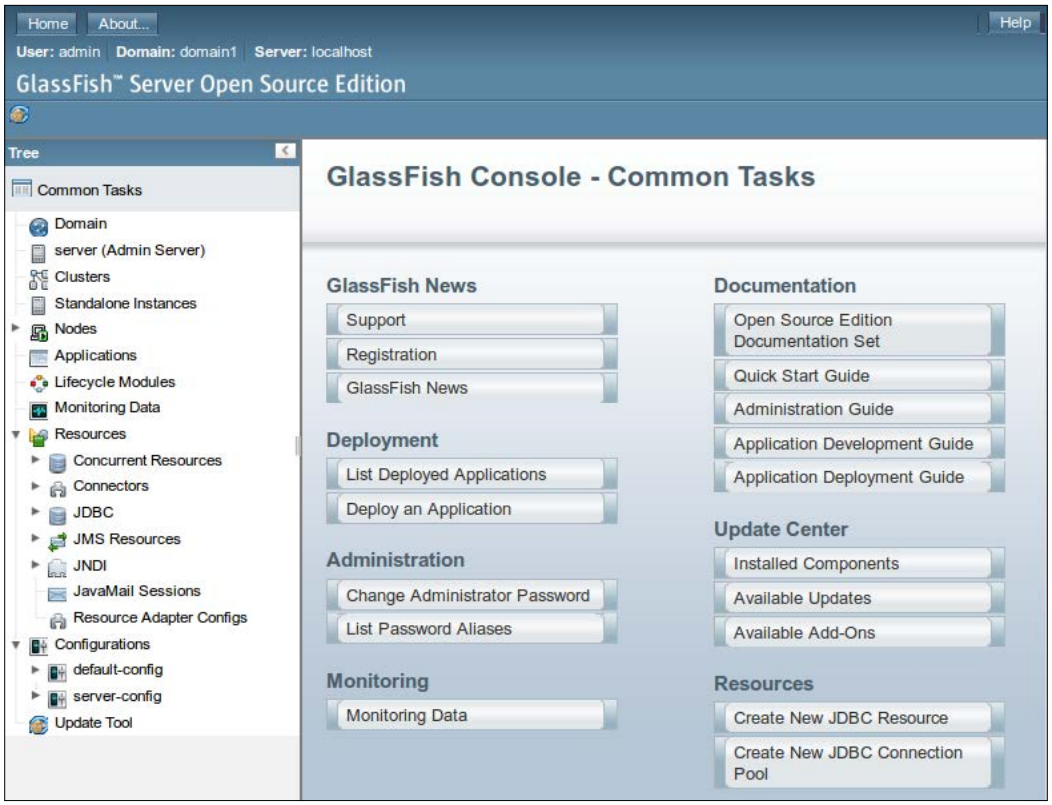
If any of the preceding steps fail or for help with GlassFish in general, a great resource is the GlassFish forum at `https://www.java.net/forums/glassfish/glassfish`.

Deploying our first Java EE application

To further confirm that our GlassFish installation is running properly, we will deploy a **WAR** (**Web ARchive**) file and make sure the file deploys and executes properly. Before moving on, please download the file `simpleapp.war` from this book's web site at `www.packtpub.com`.

Deploying an application through the Web Console

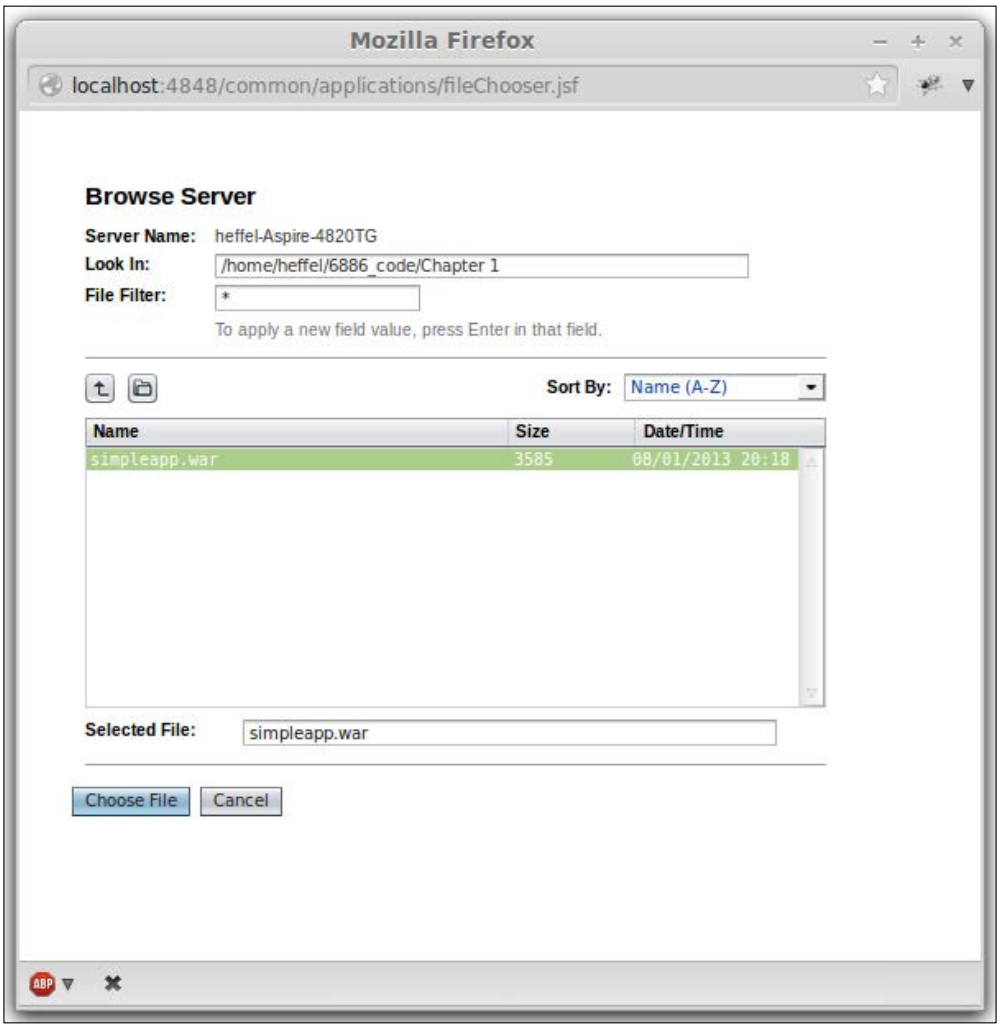
To deploy `simpleapp.war`, open a browser and navigate to `http://localhost:4848`. You should be greeted with the default GlassFish server administration page as shown in the following screenshot:



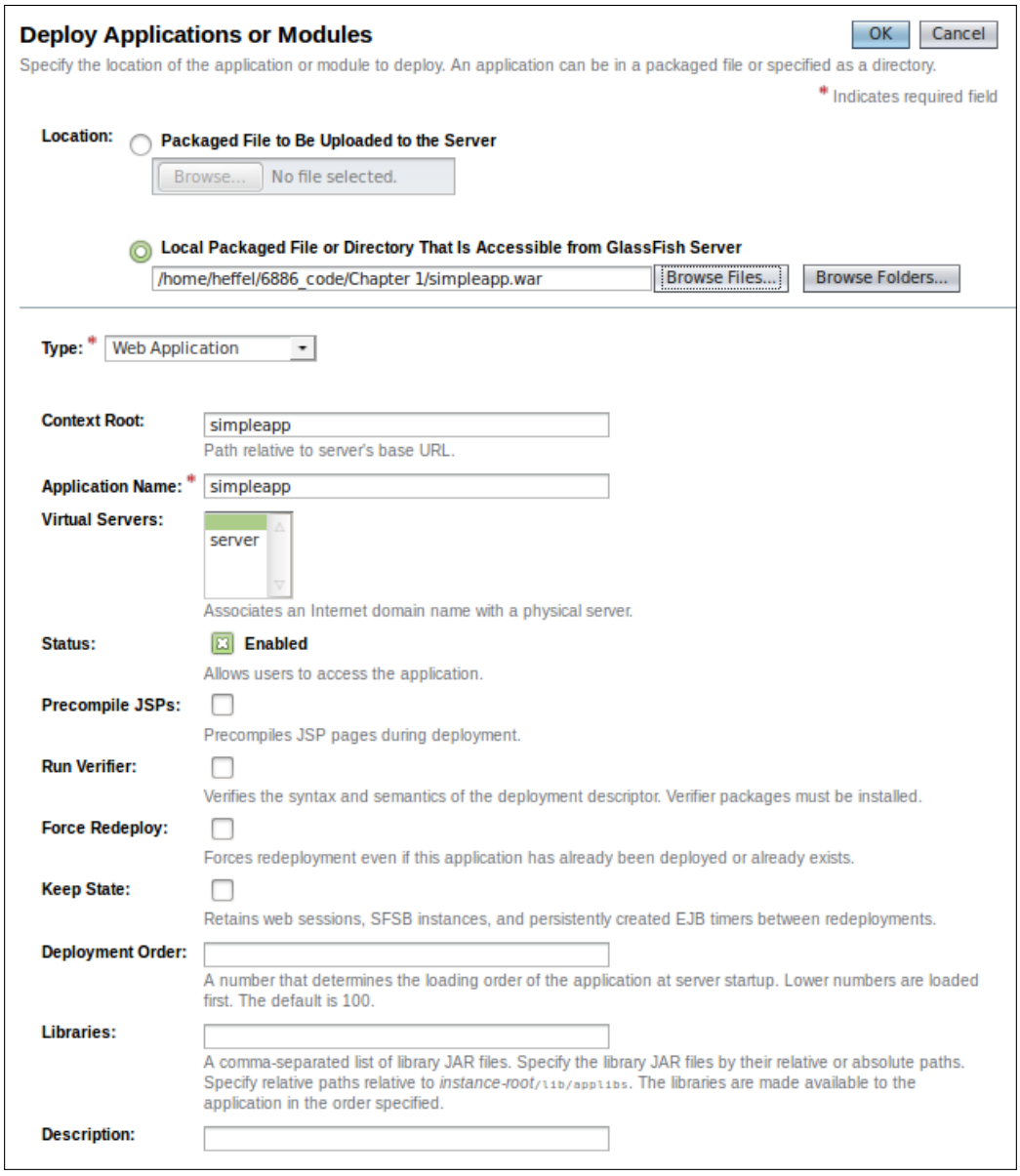
By default, GlassFish is installed in development mode. In this mode, it is not necessary to enter a username and password to access the GlassFish web console. In production environments, it is highly advisable to configure the web console so that it is password protected.

At this point, we should click on the **Deploy an Application** item under the **Deployment** section on the main screen.

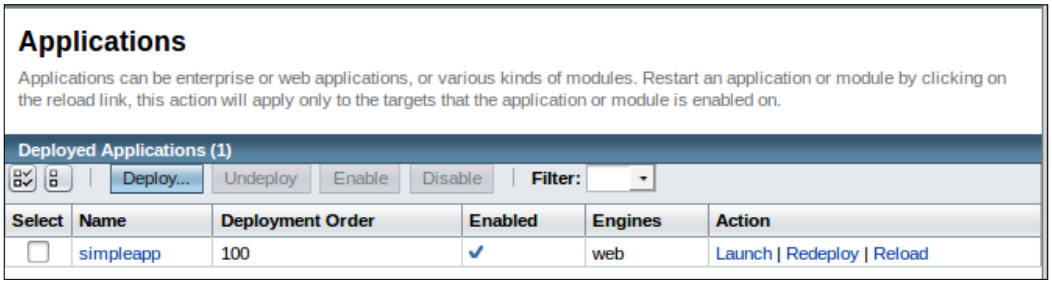
To deploy our application, we should select the **Local Packaged File or Directory That is Accessible from GlassFish Server** radio button and either type the path to our WAR file or select it by clicking on the **Browse Files...** button. Once this is done, you will see a window as shown in the following screenshot:



After we have selected our WAR file, a number of input fields that allow us to specify several options are shown. For our purposes, all defaults are fine. We can simply click on the **OK** button at the top right of the page as shown in the following screenshot:



Once we deploy our application, the GlassFish web console displays the **Applications** window, with our application listed as one of the deployed applications as shown in the following screenshot:



To execute the `simpleapp` application, type the following URL in the browser's location text field:

```
http://localhost:8080/simpleapp/simpleservlet
```

The resulting page should look like the following screenshot:

That's it! We have successfully deployed our first Java EE application.

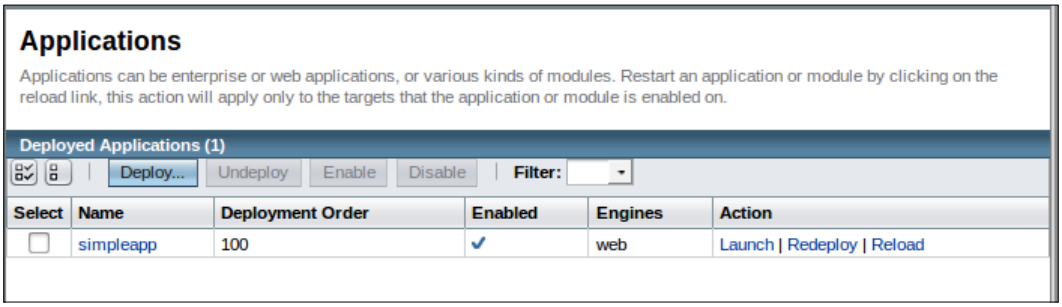
Undeploying an application through the GlassFish Admin Console

To undeploy the application we just deployed, log in to the GlassFish Admin Console by typing the following URL in the browser:

```
http://localhost:4848
```

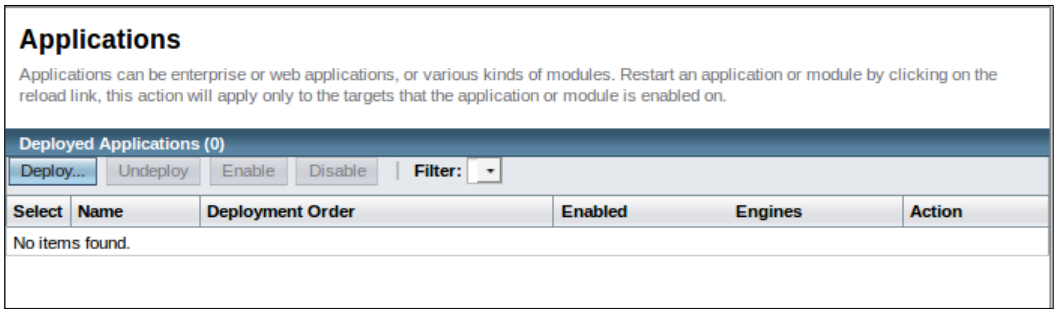
Then, either click on the **Applications** menu item in the navigation pane on the left, or click on the **List Deployed Applications** item on the administration console's home page.

Either way should take us to the application management page as shown in the following screenshot:



The application can be undeployed simply by selecting the checkbox next to the `simpleapp` name from the list of deployed applications and clicking on the **Undeploy** button above the list of deployed applications.

Once our application has been undeployed, it is no longer shown on the application management page as shown in the following screenshot:



Deploying an application through the command line

There are two ways in which an application can be deployed through the command line—it can be done either by copying the artifact we want to deploy to an `autodeploy` directory, or by using GlassFish's `asadmin` command-line utility.