Spring Framework

tutorialspoint

SIMPLY EASY LEARNING

www.tutorialspoint.com

https://www.facebook.com/tutorialspointindia

https://twitter.com/tutorialspoint

About the Tutorial

Spring framework is an open source Java platform that provides comprehensive infrastructure support for developing robust Java applications very easily and very rapidly.

Spring framework was initially written by Rod Johnson and was first released under the Apache 2.0 license in June 2003.

This tutorial has been written based on Spring Framework version 4.1.6 released in Mar 2015.

Audience

This tutorial is designed for Java programmers with a need to understand the Spring framework in detail along with its architecture and actual usage. This tutorial will bring you at an intermediate level of expertise, from where you can take yourself to higher levels of expertise.

Prerequisites

Before proceeding with this tutorial, you should have a good understanding of Java programming language. A basic understanding of Eclipse IDE is also required because all the examples have been compiled using Eclipse IDE.

Questions and Answers

Spring Questions and Answers has been designed with a special intention of helping students and professionals preparing for various **Certification Exams** and **Job Interviews**. This section provides a useful collection of sample Interview Questions and Multiple Choice Questions (MCQs) and their answers with appropriate explanations - **Study Spring Questions and Answers**

Disclaimer & Copyright

Table of Contents

1. Spring – Overview

Spring is the most popular application development framework for enterprise Java. Millions of developers around the world use Spring Framework to create high performing, easily testable, and reusable code.

Spring framework is an open source Java platform. It was initially written by Rod Johnson and was first released under the Apache 2.0 license in June 2003.

Spring is lightweight when it comes to size and transparency. The basic version of Spring framework is around 2MB.

The core features of the Spring Framework can be used in developing any Java application, but there are extensions for building web applications on top of the Java EE platform. Spring framework targets to make J2EE development easier to use and promotes good programming practices by enabling a POJO-based programming model.

Benefits of Using the Spring Framework

Following is the list of few of the great benefits of using Spring Framework:

- Spring enables developers to develop enterprise-class applications using POJOs. The benefit of using only POJOs is that you do not need an EJB container product such as an application server but you have the option of using only a robust servlet container such as Tomcat or some commercial product.

- Spring is organized in a modular fashion. Even though the number of packages and classes are substantial, you have to worry only about the ones you need and ignore the rest.

- Spring does not reinvent the wheel, instead it truly makes use of some of the existing technologies like several ORM frameworks, logging frameworks, JEE, Quartz and JDK timers, and other view technologies.

- Testing an application written with Spring is simple because environment-dependent code is moved into this framework. Furthermore, by using JavaBean-style POJOs, it becomes easier to use dependency injection for injecting test data.

- Spring's web framework is a well-designed web MVC framework, which provides a great alternative to web frameworks such as Struts or other over-engineered or less popular web frameworks.

- Spring provides a convenient API to translate technology-specific exceptions (thrown by JDBC, Hibernate, or JDO, for example) into consistent, unchecked exceptions.

- Lightweight IoC containers tend to be lightweight, especially when compared to EJB containers, for example. This is beneficial for developing and deploying applications on computers with limited memory and CPU resources.

- Spring provides a consistent transaction management interface that can scale down to a local transaction (using a single database, for example) and scale up to global transactions (using JTA, for example).

Dependency Injection (DI)

The technology that Spring is most identified with is the **Dependency Injection (DI)** flavor of Inversion of Control. The **Inversion of Control** (**IoC**) is a general concept, and it can be expressed in many different ways. Dependency Injection is merely one concrete example of Inversion of Control.

When writing a complex Java application, application classes should be as independent as possible of other Java classes to increase the possibility to reuse these classes and to test them independently of other classes while unit testing. Dependency Injection helps in gluing these classes together and at the same time keeping them independent.

What is dependency injection exactly? Let's look at these two words separately. Here the dependency part translates into an association between two classes. For example, class A is dependent of class B. Now, let's look at the second part, injection. All this means is, class B will get injected into class A by the IoC.

Dependency injection can happen in the way of passing parameters to the constructor or by post-construction using setter methods. As Dependency Injection is the heart of Spring Framework, we will explain this concept in a separate chapter with relevant example.

Aspect Oriented Programming (AOP)

One of the key components of Spring is the **Aspect Oriented Programming (AOP)** framework. The functions that span multiple points of an application are called **cross-cutting concerns** and these cross-cutting concerns are conceptually separate from the application's business logic. There are various common good examples of aspects including logging, declarative transactions, security, caching, etc.

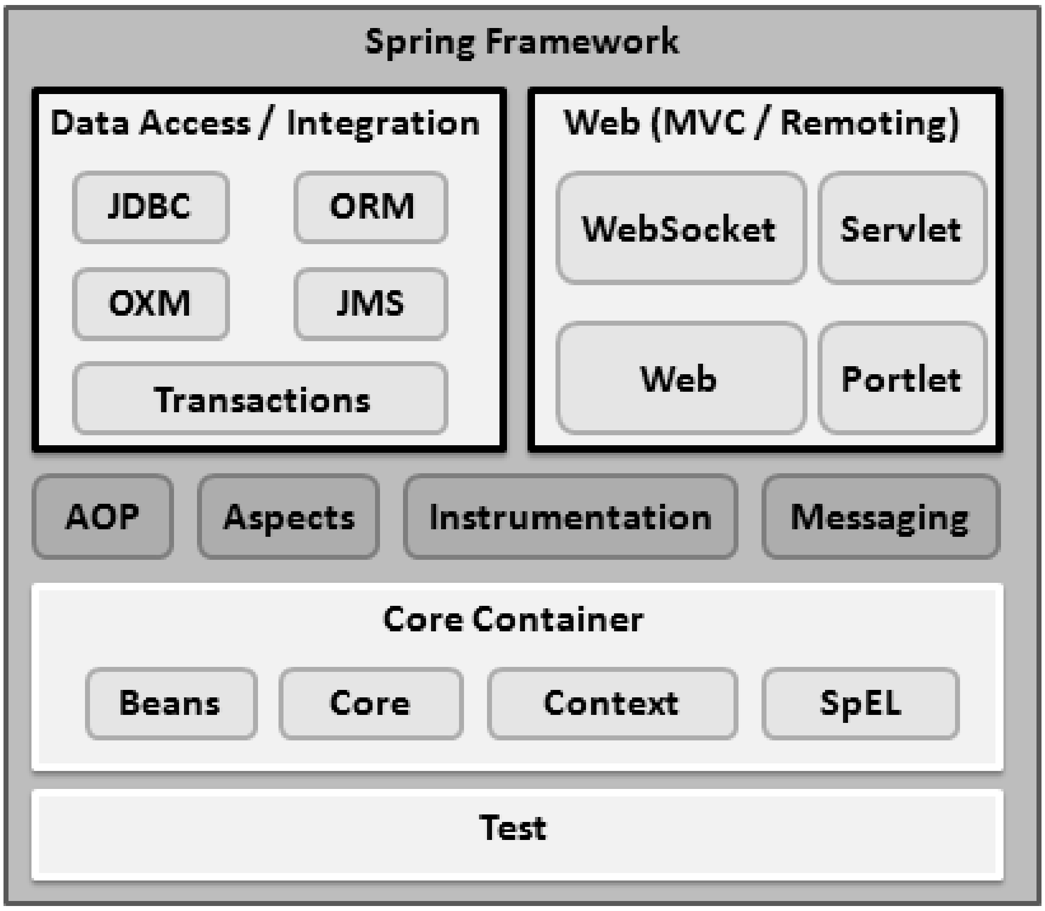
The key unit of modularity in OOP is the class, whereas in AOP the unit of modularity is the aspect. DI helps you decouple your application objects from each other, while AOP helps you decouple cross-cutting concerns from the objects that they affect.

The AOP module of Spring Framework provides an aspect-oriented programming implementation allowing you to define method-interceptors and pointcuts to cleanly decouple code that implements functionality that should be separated. We will discuss more about Spring AOP concepts in a separate chapter.

2. Spring – Architecture

Spring could potentially be a one-stop shop for all your enterprise applications. However, Spring is modular, allowing you to pick and choose which modules are applicable to you, without having to bring in the rest. The following section provides details about all the modules available in Spring Framework.

The Spring Framework provides about 20 modules which can be used based on an application requirement.



Core Container

The Core Container consists of the Core, Beans, Context, and Expression Language modules the details of which are as follows:

- The **Core** module provides the fundamental parts of the framework, including the IoC and Dependency Injection features.

- The **Bean** module provides BeanFactory, which is a sophisticated implementation of the factory pattern.

- The **Context** module builds on the solid base provided by the Core and Beans modules and it is a medium to access any objects defined and configured. The ApplicationContext interface is the focal point of the Context module.

- The **SpEL** module provides a powerful expression language for querying and manipulating an object graph at runtime.

Data Access/Integration

The Data Access/Integration layer consists of the JDBC, ORM, OXM, JMS and Transaction modules whose detail is as follows:

- The **JDBC** module provides a JDBC-abstraction layer that removes the need for tedious JDBC related coding.

- The **ORM** module provides integration layers for popular object-relational mapping APIs, including JPA, JDO, Hibernate, and iBatis.

- The **OXM** module provides an abstraction layer that supports Object/XML mapping implementations for JAXB, Castor, XMLBeans, JiBX and XStream.

- The Java Messaging Service (**JMS**) module contains features for producing and consuming messages.

- The **Transaction** module supports programmatic and declarative transaction management for classes that implement special interfaces and for all your POJOs.

Web

The Web layer consists of the Web, Web-MVC, Web-Socket, and Web-Portlet modules the details of which are as follows:

- The **Web** module provides basic web-oriented integration features such as multipart file-upload functionality and the initialization of the IoC container using servlet listeners and a web-oriented application context.

- The **Web-MVC** module contains Spring's Model-View-Controller (MVC) implementation for web applications.

- The **Web-Socket** module provides support for WebSocket-based, two-way communication between the client and the server in web applications.

- The **Web-Portlet** module provides the MVC implementation to be used in a portlet environment and mirrors the functionality of Web-Servlet module.

Miscellaneous

There are few other important modules like AOP, Aspects, Instrumentation, Web and Test modules the details of which are as follows:

- The **AOP** module provides an aspect-oriented programming implementation allowing you to define method-interceptors and pointcuts to cleanly decouple code that implements functionality that should be separated.

- The **Aspects** module provides integration with AspectJ, which is again a powerful and mature AOP framework.

- The **Instrumentation** module provides class instrumentation support and class loader implementations to be used in certain application servers.

- The **Messaging** module provides support for STOMP as the WebSocket sub-protocol to use in applications. It also supports an annotation programming model for routing and processing STOMP messages from WebSocket clients.

- The **Test** module supports the testing of Spring components with JUnit or TestNG frameworks.

3. Spring – Environment Setup

This chapter will guide you on how to prepare a development environment to start your work with Spring Framework. It will also teach you how to set up JDK, Tomcat and Eclipse on your machine before you set up Spring Framework.

Step 1 – Set Up Java Development Kit (JDK)

You can download the latest version of SDK from Oracle's Java site: Java SE Downloads. You will find instructions for installing JDK in downloaded files, follow the given instructions to install and configure the setup. Finally set PATH and JAVA_HOME environment variables to refer to the directory that contains java and javac, typically java_install_dir/bin and java_install_dir respectively.

If you are running Windows and have installed the JDK in C:\jdk1.6.0_15, you would have to put the following line in your C:\autoexec.bat file.

```
set PATH=C:\jdk1.6.0_15\bin;%PATH%

set JAVA_HOME=C:\jdk1.6.0_15
```

Alternatively, on Windows NT/2000/XP, you will have to right-click on My Computer, select Properties -> Advanced -> Environment Variables. Then, you will have to update the PATH value and click the OK button.

On Unix (Solaris, Linux, etc.), if the SDK is installed in /usr/local/jdk1.6.0_15 and you use the C shell, you will have to put the following into your .cshrc file.

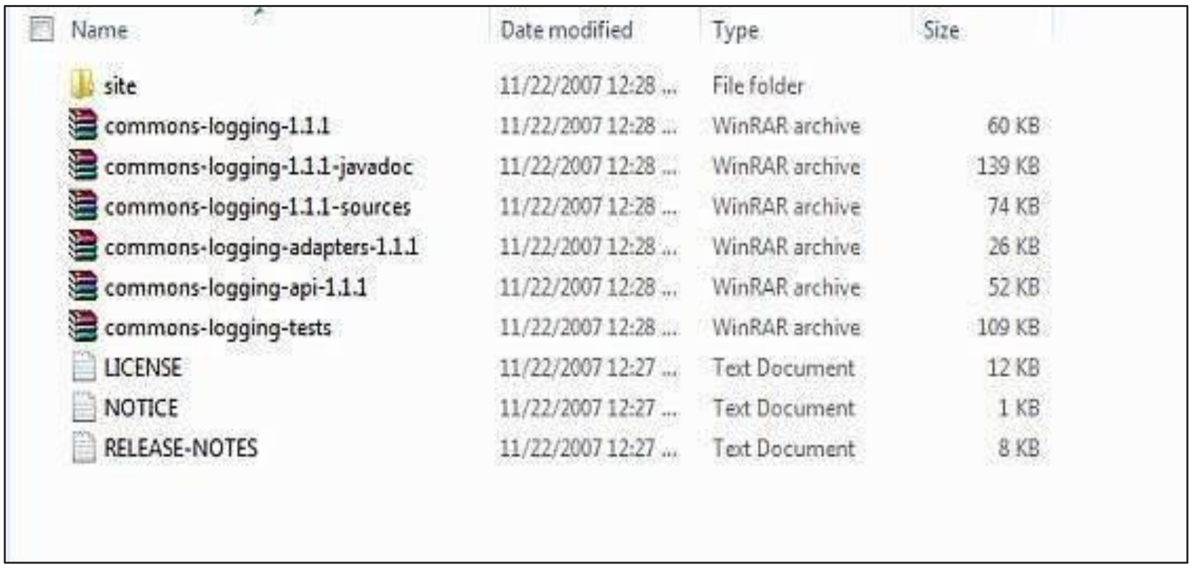
```
setenv PATH /usr/local/jdk1.6.0_15/bin:$PATH

setenv JAVA_HOME /usr/local/jdk1.6.0_15
```

Alternatively, if you use an Integrated Development Environment (IDE) like Borland JBuilder, Eclipse, IntelliJ IDEA, or Sun ONE Studio, you will have to compile and run a simple program to confirm that the IDE knows where you have installed Java. Otherwise, you will have to carry out a proper setup as given in the document of the IDE.

Step 2 — Install Apache Common Logging API

You can download the latest version of Apache Commons Logging API from http://commons.apache.org/logging/. Once you download the installation, unpack the binary distribution into a convenient location. For example, in C:\commons-logging-1.1.1 on Windows, or /usr/local/commons-logging-1.1.1 on Linux/Unix. This directory will have the following jar files and other supporting documents, etc.



Name	Date modified	Type	Size
site	11/22/2007 12:28 ...	File folder	
commons-logging-1.1.1	11/22/2007 12:28 ...	WinRAR archive	60 KB
commons-logging-1.1.1-javadoc	11/22/2007 12:28 ...	WinRAR archive	139 KB
commons-logging-1.1.1-sources	11/22/2007 12:28 ...	WinRAR archive	74 KB
commons-logging-adapters-1.1.1	11/22/2007 12:28 ...	WinRAR archive	26 KB
commons-logging-api-1.1.1	11/22/2007 12:28 ...	WinRAR archive	52 KB
commons-logging-tests	11/22/2007 12:28 ...	WinRAR archive	109 KB
LICENSE	11/22/2007 12:27 ...	Text Document	12 KB
NOTICE	11/22/2007 12:27 ...	Text Document	1 KB
RELEASE-NOTES	11/22/2007 12:27 ...	Text Document	8 KB

Make sure you set your CLASSPATH variable on this directory properly otherwise you will face a problem while running your application.

Step 3 – Set Up Eclipse IDE

All the examples in this tutorial have been written using Eclipse IDE. So we would suggest you should have the latest version of Eclipse installed on your machine.

To install Eclipse IDE, download the latest Eclipse binaries from http://www.eclipse.org/downloads/. Once you download the installation, unpack the binary distribution into a convenient location. For example, in C:\eclipse on Windows, or /usr/local/eclipse on Linux/Unix and finally set PATH variable appropriately.

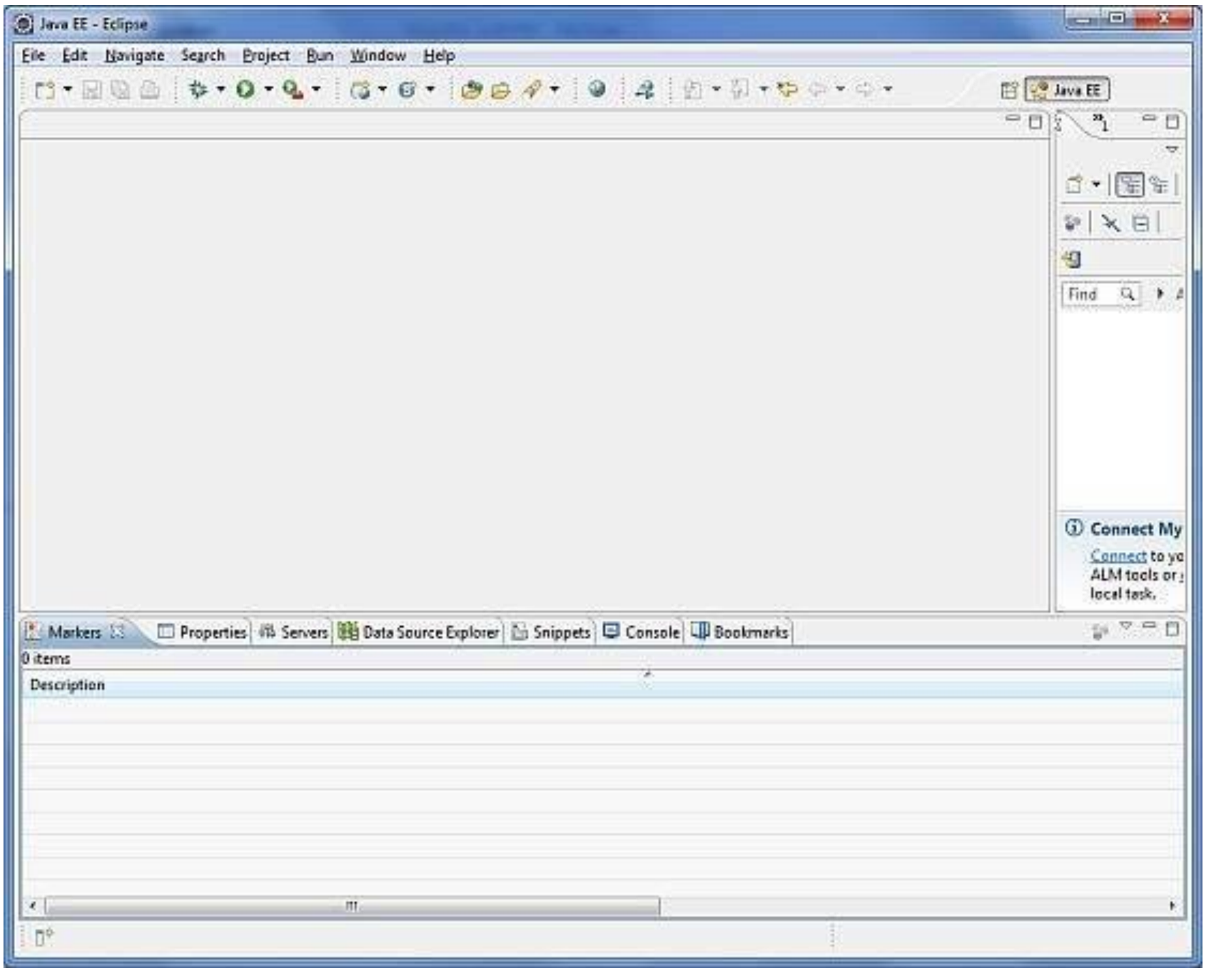
Eclipse can be started by executing the following commands on Windows machine, or you can simply double-click on eclipse.exe

```
%C:\eclipse\eclipse.exe
```

Eclipse can be started by executing the following commands on Unix (Solaris, Linux, etc.) machine:

```
$/usr/local/eclipse/eclipse
```

After a successful startup, if everything is fine then it should display the following result.



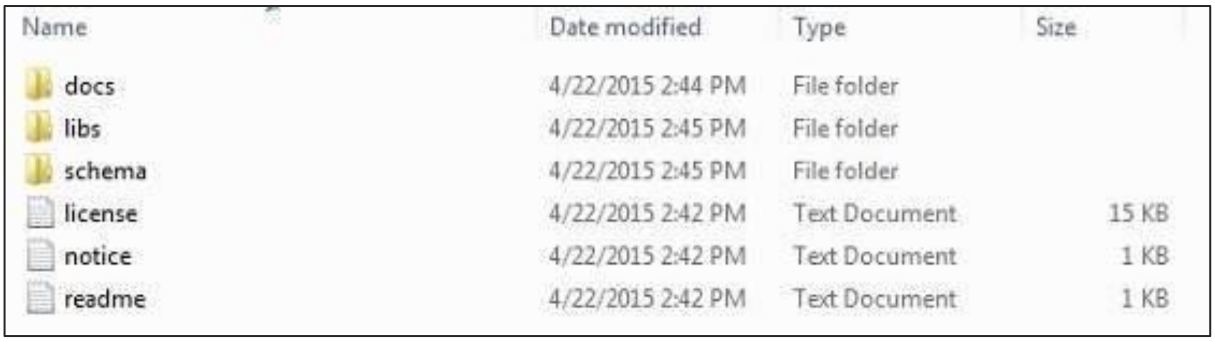
Step 4 – Set Up Spring Framework Libraries

Now if everything is fine, then you can proceed to set up your Spring framework. Following are the simple steps to download and install the framework on your machine.

- Make a choice whether you want to install Spring on Windows or Unix, and then proceed to the next step to download .zip file for Windows and .tz file for Unix.

- Download the latest version of Spring framework binaries from http://repo.spring.io/release/org/springframework/spring.

- At the time of developing this tutorial, **spring-framework-4.1.6.RELEASE-dist.zip** was downloaded on Windows machine. After the downloaded file was unzipped, it gives the following directory structure inside E:\spring.



Name	Date modified	Type	Size
docs	4/22/2015 2:44 PM	File folder	
libs	4/22/2015 2:45 PM	File folder	
schema	4/22/2015 2:45 PM	File folder	
license	4/22/2015 2:42 PM	Text Document	15 KB
notice	4/22/2015 2:42 PM	Text Document	1 KB
readme	4/22/2015 2:42 PM	Text Document	1 KB

You will find all the Spring libraries in the directory **E:\spring\libs**. Make sure you set your CLASSPATH variable on this directory properly otherwise you will face a problem while running your application. If you are using Eclipse, then it is not required to set CLASSPATH because all the setting will be done through Eclipse.

Once you are done with this last step, you are ready to proceed to your first Spring Example in the next chapter.