**About Spring**

Spring is a great framework for development of Enterprise grade applications. Spring is a light-weight framework for the development of enterprise-ready applications. Spring can be used to configure declarative transaction management, remote access to your logic using RMI or web services, mailing facilities and various options in persisting your data to a database. Spring framework can be used in modular fashion, it allows to use in parts and leave the other components which is not required by the application.

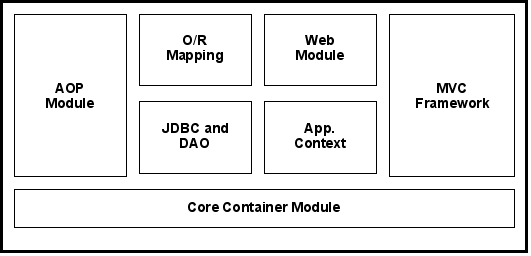
**Features of Spring Framework:**

* **Transaction Management:** Spring framework provides a generic abstraction layer for transaction management. This allowing the developer to add the pluggable transaction managers, and making it easy to demarcate transactions without dealing with low-level issues. Spring's transaction support is not tied to J2EE environments and it can be also used in container less environments.
* **JDBC Exception Handling:** The JDBC abstraction layer of the Spring offers a meaningful exception hierarchy, which simplifies the error handling strategy.
* **Integration with Hibernate(ORM), JDO, and iBATIS:** Spring provides best Integration services with Hibernate, JDO and iBATIS.
* **AOP Framework:** Spring is best AOP framework.
* **MVC Framework:**Spring comes with MVC web application framework, built on core Spring functionality. This framework is highly configurable via strategy interfaces, and accommodates multiple view technologies like JSP, Velocity, Tiles, iText, and POI. But other frameworks can be easily used instead of Spring MVC Framework.

**Modules in the Spring framework are:**

* **Spring AOP:** One of the key components of Spring is the AOP framework. AOP is used in Spring:
  + To provide declarative enterprise services, especially as a replacement for EJB declarative services. The most important such service is declarative transaction management, which builds on Spring's transaction abstraction.
  + To allow users to implement custom aspects, complementing their use of OOP with AOP.
* **Spring ORM:** The ORM package is related to the database access. It provides integration layers for popular object-relational mapping APIs, including JDO, Hibernate and iBatis.
* **Spring Web:** The Spring Web module is part of Spring's web application development stack, which includes Spring MVC.
* **Spring DAO:** The DAO (Data Access Object) support in Spring is primarily for standardizing the data access work using the technologies like JDBC, Hibernate or JDO.
* **Spring Context:** This package builds on the beans package to add support for message sources and for the Observer design pattern, and the ability for application objects to obtain resources using a consistent API.
* **Spring Web MVC:** This is the Module which provides the MVC implementations for the web applications.
* **Spring Core:** The Core package is the most import component of the Spring Framework. This component provides the Dependency Injection features. The BeanFactory provides a factory pattern which separates the dependencies like initialization, creation and access of the objects from your actual program logic.

The following diagram represents the Spring Framework Architecture.

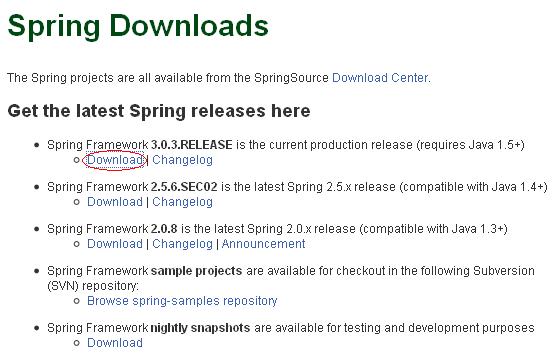


**Installation of Spring Framework**

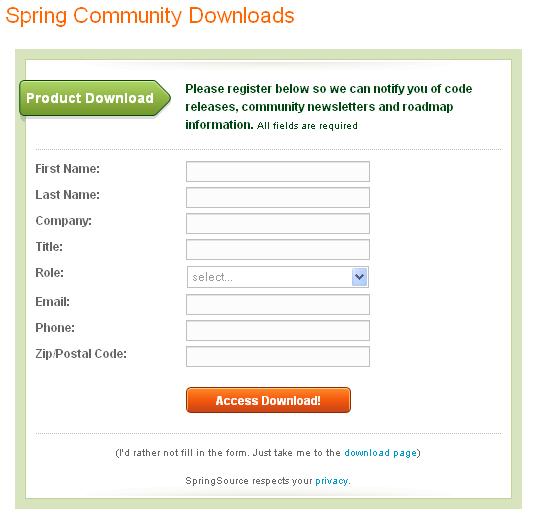
**Spring Framework Install** - A quick tutorial to install Spring Framework on your development environment.

**Downloading Spring Framework:** It is available for downloading straight form its official site. The latest version of this framework for this day is 3.0.3

* Simply go to <http://www.springsource.org/download>
* Choose **spring-framework-3.0.3.RELEASE** Download option as shown on the picture below:



* Fill out this form and click **Access Download** button:



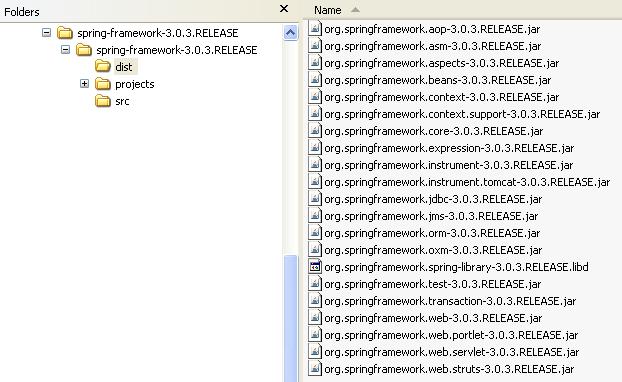
* Choose Spring Framework download with or without documentation, click it with your right mouse button and choose "save as" option. as shown on the picture.



* **Extracting Spring download:** Now extract the downloaded file (spring-framework-3.0.3.RELEASE.zip) into your favorite directory with Winzip/Winrar or an other extracting program u have installed on your computer. Depending upon the system speed it will take around 1 minute.

Once extract go to the extracted folder and you will find many files.

Here is the screen shot of the directory:



The dist directory of the spring framework contains the spring modules (modules directory) library files. We are mostly concern with these jar files. We will copy these jar files into our development environment. These jar files are:

* + org.springframework.aop-3.0.3.RELEASE.jar
  + org.springframework.asm-3.0.3.RELEASE.jar
  + org.springframework.aspects-3.0.3.RELEASE.jar
  + org.springframework.beans-3.0.3.RELEASE.jar
  + org.springframework.context-3.0.3.RELEASE.jar
  + org.springframework.context.support-3.0.3.RELEASE.jar
  + org.springframework.expression-3.0.3.RELEASE.jar
  + org.springframework.instrument.tomcat-3.0.3.RELEASE.jar
  + org.springframework.jms-3.0.3.RELEASE.jar
  + org.springframework.oxm-3.0.3.RELEASE.jar
  + org.springframework.transaction-3.0.3.RELEASE.jar
  + org.springframework.web-3.0.3.RELEASE.jar
  + org.springframework.web.portlet-3.0.3.RELEASE.jar
  + org.springframework.web.servlet-3.0.3.RELEASE.jar
  + org.springframework.web.struts-3.0.3.RELEASE.jar

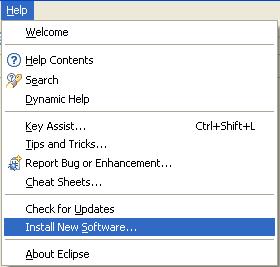
We will be using these libraries as per our requirement in our application.

Another most important directory is "lib", which contains required library files for the application.

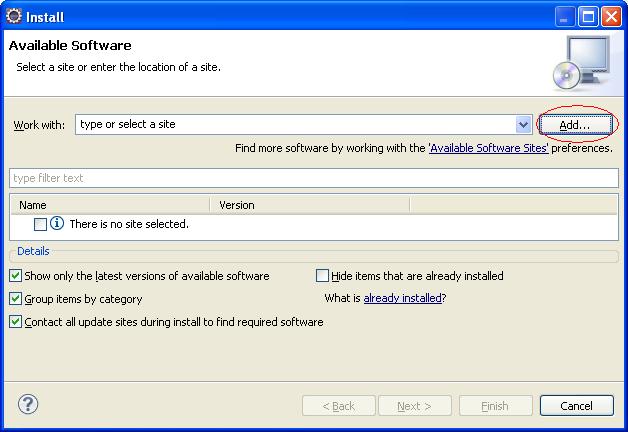
**Spring IDE Tutorial**

Spring IDE is an eclipse plug-in that helps in developing Spring Application. First we will see how to install the Spring IDE and later we will create our first Spring project using it. I am using Eclipse 3.5.0 version to demonstrate this.

* To install Spring IDE, Go to **Help** -> **Install New Software** as shown in the picture.



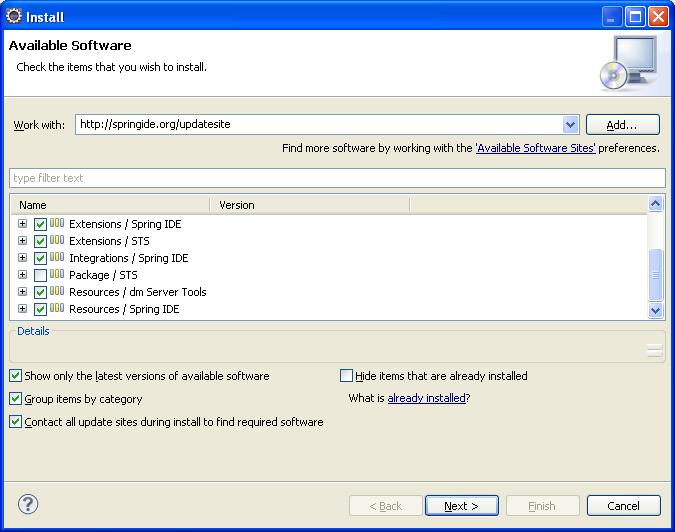
* Click the "**Add**" button as shown in the picture.



* Enter Name and copy this url "http://springide.org/updatesite" and paste it in place of Location in the Add Site popup as shown in the picture.



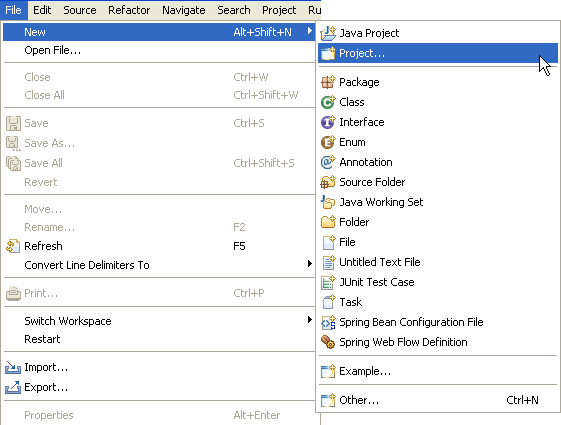
* Select all the Spring IDE features and click Install as shown in the picture.



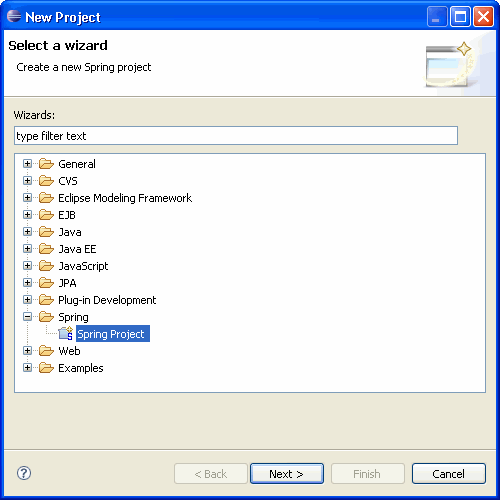
Once the installation is complete you are done. Now let's see how to create the hello world example using the Spring IDE.

If you face any problems while installation, please follow the [link](http://tutorials4u.net/forum/showthread.php?tid=1&pid=2)

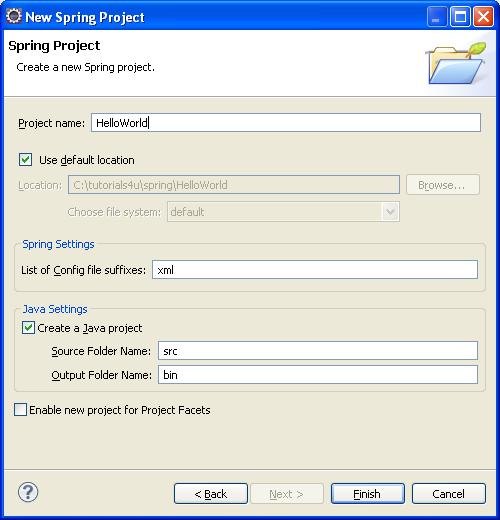
* First create a Spring project, go to **File** -> **New** -> **Project**.



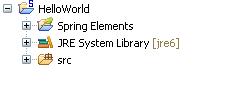
* Select Spring Project and click Next.



* Enter the project name and click Finish.



The "S" in the upper right corner indictes it is a Spring Project.



* Right click the src package and create a new package "com.tutorials4u.helloworld".
* Right click on the "com.tutorials4u.helloworld" package, Create a new java class with the name, "HelloWorldBean"

01.package com.tutorials4u.helloworld;

02.

03.public class HelloWorldBean {

04.

05.    private String message;

06.

07.    public void setMessage(String message) {

08.        this.message = message;

09.    }

10.

11.    public void display()

12.    {

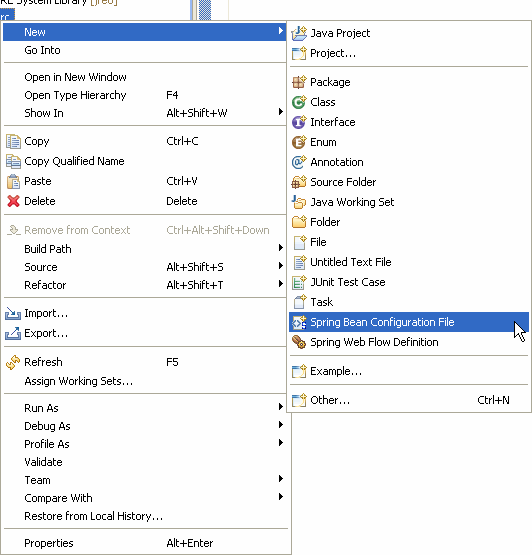
13.        System.out.println(message);

14.    }

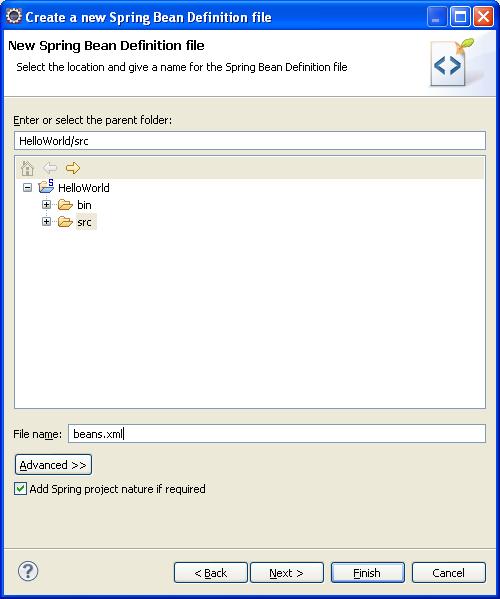
15.}

The HelloWorld class has a message property and its value is set using the setMessage() method. This is called setter injection. Instead of directly hard coding the message, we inject it through an external configuration file. The design pattern used here is called Dependency Injection design pattern and it is explained in detail in the next example. ([Spring Dependency Injection](http://www.tutorials4u.net/spring-tutorial/spring_ioc.html))The HelloWorldBean class also has a display() method to display the message.

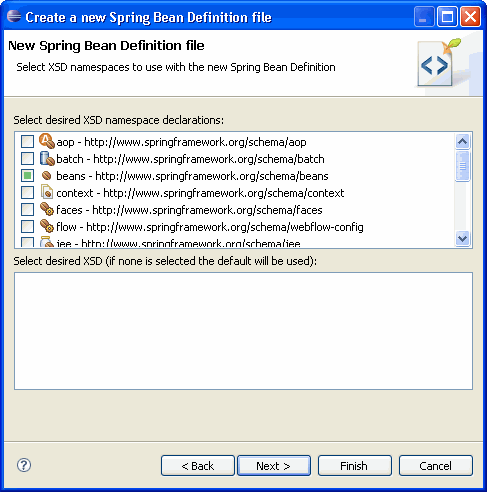
Now we have created the HelloWorldBean bean class, the next step is to add an entry for this in the bean configuration file. The bean configuration file is used to configure the beans in the Spring IoC container. To create a new bean configuration file right click the **src** folder and select **New -> Spring Bean Configuration File**.



Enter the bean name and click Next.



Select the beans option and click Finish.



Now the Spring configuration file is created. Add the following code to created an entry for the HelloWorld bean class.

01.<?xml version="1.0" encoding="UTF-8"?>

02.<beans xmlns="http://www.springframework.org/schema/beans"

03.xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

04.xsi:schemaLocation=" http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">

05.

06.    <bean id="helloWorldBean" class="com.tutorials4u.helloworld.HelloWorldBean">

07.        <property name="message" value="Hello World!"></property>

08.    </bean>

09.

10.</beans>

The id attribute of the bean element is used to give a logical name to the bean and the class attribute specifies the fully qualified class name of the bean. The property element within the bean element is used to set the property value. Here we set the message property to "Hello World!".

If you want to display a differnt message, the only change you need to do is, to change the value of the message in the bean configuration file. This is one of the main benefits of using the Dependency Injection design pattern, this makes the code loosely coupled.

To dispaly the message, create the following HelloWorld class.

01.package com.tutorials4u.helloworld;

02.

03.import org.springframework.context.ApplicationContext;

04.import org.springframework.context.support. ClassPathXmlApplicationContext;

05.

06.public class HelloWorld {

07.

08.    public static void main(String[] args) {

09.        ApplicationContext context = new ClassPathXmlApplicationContext("beans.xml");

10.        HelloWorldBean helloWorldBean = (HelloWorldBean) context.getBean("helloWorldBean");

11.        helloWorldBean.display();

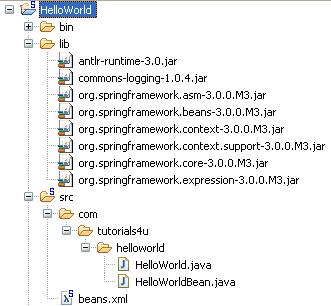
12.    }

13.

14.}

First we instantiate the Spring IoC container with the bean configuration file beans.xml. We use the getBean() method to retrive the helloWorld bean from the application context and call the display() method to display the message in the console.

The figure shows the final directory structure of the hello world example.



Add the following jar files to the classpath.

1.antlr-runtime-3.0

2.commons-logging-1.0.4

3.org.springframework.asm-3.0.0.M3

4.org.springframework.beans-3.0.0.M3

5.org.springframework.context-3.0.0.M3

6.org.springframework.context.support-3.0.0.M3

7.org.springframework.core-3.0.0.M3

8.org.springframework.expression-3.0.0.M3

To execute the example run the **HelloWorld** file. The "Hello World!" message gets printed on the console.

**Output:**

