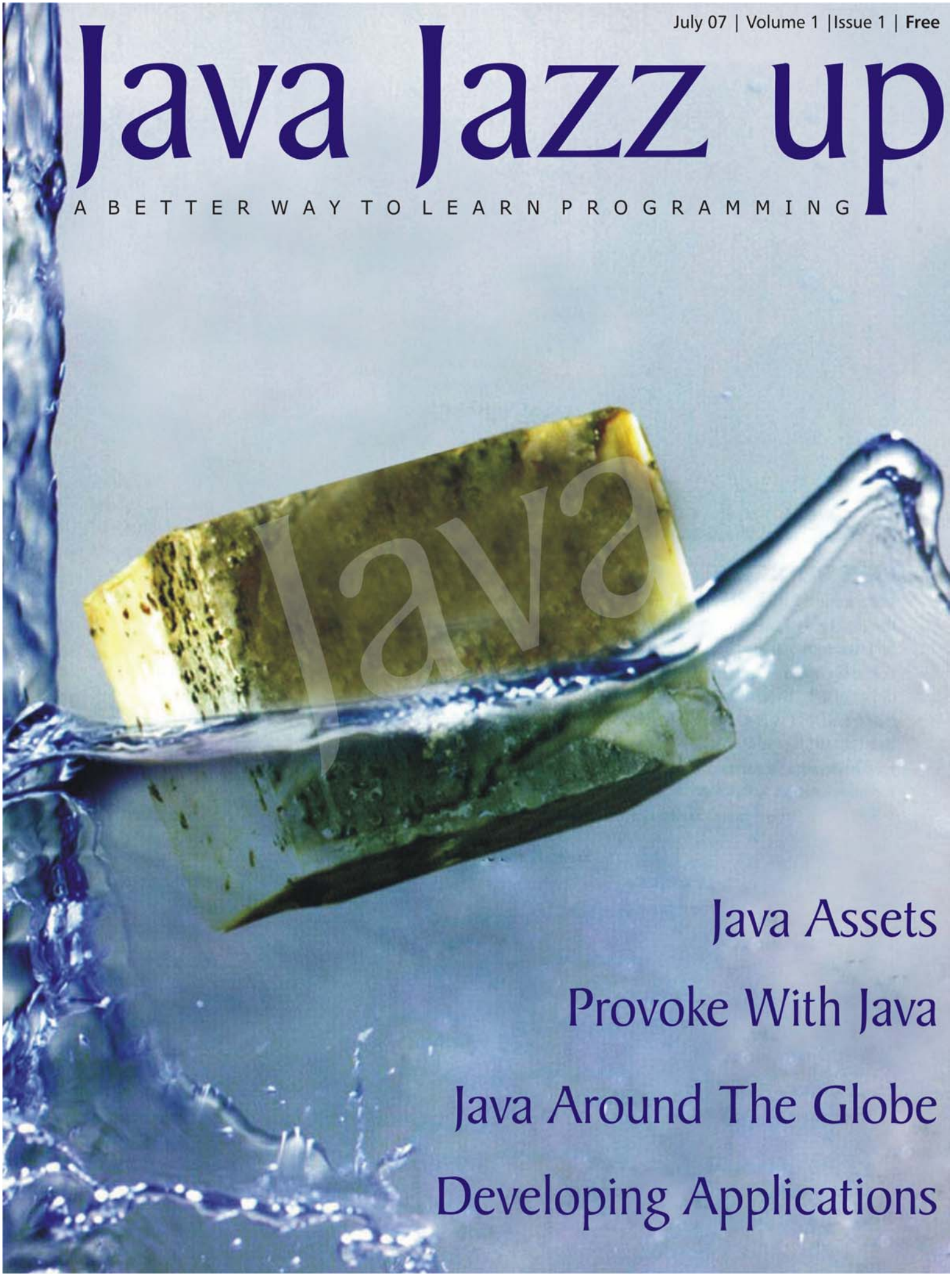
July 07 | Volume 1 |Issue 1 | **Free**

Java Jazz up

A BETTER WAY TO LEARN PROGRAMMING



Java Assets

Provoke With Java

Java Around The Globe

Developing Applications

Java Jazz Up

"It is not how much you do, but how much love you put in the doing"

Editor	Deepak Kumar
Editor-Technical	Ravi Kant Tamana Agarwal
Graphics Designer	Suman Saurabh

Published By

RoseIndia

www.javajazzup.com

Dear Valued Readers,

Java Jazz Up Team is proud to offer you a FREE online magazine on Java Technology. Today Java is the most popular programming language and Java programmer's community is growing with a pace of 10 percent annually around the globe, covering headcounts over 1 Million.

Java Technology has completely changed the way we look at our personal computer, Internet, mobile devices and numerous technological advancements around the globe. Today it is one of the most pervasive technologies in the world that seems ubiquitous in all other related developments. In a sense Java is now in our daily needs whether we are a consumer, a developer or an enterprise owner.

Keeping in mind the latest developments, role of the technology and being forerunner online java resource providers we feel the need to provide you a comprehensive and monthly free newsletter that contains materials on Java.

"Java Jazz Up" as the name suggests is an attempt to add something extra that will certainly excite you to explore new things in Java and at the same time bringing liveliness to your knowledge. This edition offers java technological updates, tutorials and examples written by our developers that are easy to understand and implement specially for a newbie. It is going to cover almost all important aspects of Java technology.

In short it will wrap all facet of java as a leading technology including web/application services, frameworks, design issues, integrating various technologies together, open source tools and a lot beyond, expectations. Additionally you will find interesting articles, news and latest Java updates enriching your java treasure.

Enjoy reading and please let us know how we can serve you better with our coming edition.

Best Wishes,
Java Jazz Up Team

Contents

JAVA AROUND THE GLOBE

Yahoo! Go: smart mobile browsing with java

The new set of Java development tools have been launched by Research In Motion for its BlackBerry handheld. There are lot of new features which include an enhanced Application Programming Interface set, documentation, code samples and applications, new Java Specification Request implementations; moreover features like synchronization tools, an XML generator and a new simulator have been introduced.

Yahoo! to launch its intelligent client Yahoo! Go soon.

Using Java means a better user experience .Yahoo! Go is a Java application which is downloaded onto the handset. It offers a wide range of applications, as well as browsing reformatted websites. Using a Java client means Yahoo! Go can offer a more interesting user interface, based around a carousel of application icons, and interact with locally-stored files and address-book entries, Java APIs permitting.

Accessing locally-stored files means photographs can be uploaded to Flicker, and address-book integration opens up a host of possibilities - but Java applications can be slow to launch and the API implementations aren't always as standard as they ought to be.

BlackBerry Hand helds get Java boost

The Java development kit is available free to developers and enterprise customers as conveyed by the company at its developer Web site.

IBM and BEA Servers Get Spring Support

The Spring Framework is an open source solution for building applications using the Java and Java Enterprise Edition platforms.

Interface21 (structured support from Interface21) has certified the use of its Spring Framework with IBM's WebSphere Application Server.

In addition to its IBM collaboration, Interface 21 has been working with BEA Systems, a provider of service-oriented architecture solutions for enterprises. BEA has incorporated the Spring Framework into parts of its WebLogic Server 10.The collaboration helped BEA produce a Java Enterprise Edition 5-compliant server. Java EE5 products, based on Enterprise JavaBeans 3.0 (EJB 3.0) technology, purportedly simplify matters for developers. For example ESJ 3.0 simplifies how Java objects are mapped to relational databases with a Java Persistence Application Programming Interface.

Aonix has partnered with Fujitsu Ltd for Japanese Real-Time Java Market

Aonix has partnered with Fujitsu Ltd and NTK Aviation America for mutual cooperation in providing products, services, and support for applications related to real-time Java in Japan. On the basis of this relationship, Fujitsu is considering the application of Aonix PERC technology to future airborne computer systems.

PERC Pico technology is claimed to be the most widely used virtual machine technology for mission-critical applications. The PERC Ultra virtual machine offers rich J2SE-based capabilities, and predictable garbage collection, while PERC Pico provides low-level access and small latencies that are often required for "close to the silicon" applications. PERC technologies are said to be more predictable and reliable than other Java solutions, while offering higher productivity and lower lifetime costs compared to C/C++ applications development.

NTK Aviation America Inc will coordinate the group activities as a facilitator and maximize the strength of this Fujitsu-Aonix partnership arrangement.

DDC-I at PHOENIX (USA)

has announced Java software for hard-real-time embedded applications

DDC-I, Compilers and Real-time Operating Systems (RTOS) for Embedded Development:

DDC-I offers an extensive set of software tools designed specifically for developing safety critical real-time embedded applications.

Software developers working on producing safe, real-time Java solutions for the airlines and military now have a new tool available from DDC-I called Scorpion. This plug-in for the Eclipse environment promises the creation of lower latency Java applications than other such tools, according to an announcement issued by DDC-I.

Scorpion supports the current Real-Time Specification for Java (RTSJ) spec. DDC-I promises that its Scorpion tool will support the emerging safety standard currently being developed by the Safety-Critical Java Expert Group, of which DDC-I is a member.

Scorpion supports mixed language development, allowing developers to combine Java with languages such as C, embedded C++ and Ada, according to the announcement. It has a "smart linker" that removes unused objects, which can reduce the code size by "up to 80 percent."

IM+ for Skype Software: BlackBerry, Palm, Symbian, Java Phones...

SHAPE Services announced the availability of three beta versions of IM+ for Skype Software for Java phones, Symbian S60 and Palm OS. These beta versions are intended for people, who want to make cheap mobile calls within Skype ecosystem and to landline numbers. Since Java version of the application works on all Java-enabled phones including smartphones with Symbian OS. Skype on mobile is now available for hundreds of millions users worldwide. SHAPE welcomes everyone to participate in a beta round and experience Skype features on the new platforms.

SHAPE Services works in cross-platform mobile software development since 2002. SHAPE' products keep the leading position in wireless IM and Remote Access markets.

Flurry - a new mobile phone email service for java phones.

Flurry is a fast, free, convenient way to monitor email or RSS feeds on a conventional Java phone

But what if you don't have a high-end mobile phone? A pair of free services, Flurry and Teleflip, let you monitor (and if need be, respond to) email on a wide variety of mass-market handsets. Both are useful, but in different ways.

Flurry requires a Java-capable phone that can access the internet. Flurry officials say the service supports some 700 phone models worldwide, significantly more phones than services from Gmail and Yahoo can support.

Sun Java Studio Enterprise 8.1

Recently Sun released Java Studio Enterprise 8.1. It is build on Net Beans platform and contains the powerful features like application profiling, UML modeling and instant collaboration. It contains the tools required to develop, debug, deploy and tune the enterprise applications, web services and Java EE platform based portal components. It can integrate with runtime environment.

Java Studio Enterprise is developed using the NetBeans platform. NetBean is full-featured and user friendly due towhich it is well known in the open source community. Java Studio Enterprise includes many features, few of them are :

UML: Java Studio Enterprise 8.1 uses the unified markup language to fast develop the enterprise-grade applications.

Developing Collaboration: It enable groups to work together dynamically, and facilitates code optimization to increase the productivity and accelerates the high quality application delivery.

Application Profiling: Built-in load generation capabilities of the web applications enables an optimal eventual user experience, faster transaction through-put and understand and tune performance of applications.

Portlet Builder: It provides a facility to build and deploy the portlets and portal providers within the IDE.

The above illustration shows how the Sun Java Studio Enterprise, comprehensive runtime infrastructure, pre-configured developer IDE, and Sun Java Enterprise System are integrated with each other. Enterprise softwares developed with Java Studio Enterprise IDE are deployable to other application servers based on Java EE specifications.

Apache Solr 1.2 released

Solr is a standalone enterprise text-search engine. It is a high performance search server with a web-services like API. It is based on Lucene (a full text search java based library) that works with XML/HTTP and JSON APIs (JavaScript Object Notation is a lightweight data-interchange format). It have features like hit highlighting, faceted search, caching, replication, and a web administration interface. It is written in Java 5.0, and easily extensible with plugins written in Java. Documents are added to a search collection via XML over HTTP. The collection is queried via HTTP to receive an XML response (or alternately JSON, Python or Ruby text formats). More precisely,

- Solr offers faceted searching, hit highlighting
- Optimized High Volume Web Traffic
- Flexible and Adaptable XML configuration
- Loose schema to define types and fields
- Extensive caching
- Extensible open architecture
- XML/HTTP Interfaces
- Web administration interface
- Index replication
- Advanced Full-Text Search Capabilities
- Extensible Plug-in Architecture
- Scalability - Efficient Replication to other Solr Search Servers

JBoss Tools: Eclipse Plugins for JBoss technology

JBoss Tools is an umbrella project for the JBoss developed plugins that will make it into Red Hat Developer Studio.

JBoss community released the JBoss Tools consisting of Ajax4JSF, Eclipse Plugins including the Exadel Studio product, Richfaces (donated by Exadel to JBoss) along with Hibernate Tools, Drools IDE, JBoss Seam Tools, JBoss Application Server Tools and JBoss jBPM Tools. Exadel plug-ins provide few of the features like:

- Useful for source and two way visual editing of JavaServer Faces and Facelets pages.
- Facilitates to drag and drop between Visual Page Editor windows, component palette and project Navigator.
- It gives assistance to extend code for seam projects and JSF.
- facilitates JBoss Richfaces, JBoss Seam Components and JBoss Ajax4jsf.

SavePoint – A Lightweight Persistence Engine

SavePoint is a first released lightweight Java persistence engine. It is designed to handle the complex and difficult needs of an enterprise application, especially for the ORM frameworks, such as iBATIS and Hibernate.

It is a simple engine in which any explicit object does not require relational mappings. But the mapping information is

directly embedded into the SQL statements. XML files host the SQL statements and contains persistence instructions. The information about the control of execution of SQL statements are contained in persistence instructions.

SavePoint provides support for building a flexible, high-performance persistence with a maintainable architecture. It allows to potentially eliminate the redundant data access objects and work directly with structured domain objects .It allows to control the SQL completely. It supports the difficult and most complex business requirements and maintains flexibility, efficiency and the power of the advanced native SQL. To provide support to a business application, SavePoint is packed with the required features.

XINS 2.0 - A Web Services framework

XINS 2.0 has been recently released. It is an open source framework and used for developing the web services. It supports XML, XML-RPC, Yahoo! JSON, JSON-RPC, SOAP and REST protocols.

XINS generates the following by using the API written in XML:

- Java code for Client-side to support time-out handling, fail-over and load-balancing.
- Web application archives file that must be compatible with servlet containers.
- Test forms to test the applications using the web browser
- WSDL for supporting the SOAP-interoperability.
- Java skeleton for Server-side.
- Unit test code based on JUnit.
- SMD, for Dojo toolkit.
- Stubs for testing purpose.

The new release XINS 2.0 includes the features given below:

- It imports XML and WSDL Schema.
- Imports SMD and WSDL meta functions.
- Provides the examples with REST implementation, using Groovy and Dojo toolkit.
- Support for better integration with GWT, Dojo toolkit and the Spring Framework.
- Support to new protocols like JSON-RPC (1.0 and 1.1) and Yahoo! JSON (with callbacks).

Apache MyFaces Orchestra:

Apache has developed the MyFaces Orchestra. It simplifies the web application development with JPA (Java Persistence API) and other object relational mapping technologies like Hibernate.

No new resources are required other than those used for JPA, like the one while using Springs Templates classes are used to get access to the persistence context. In contrast with Spring framework Apache MyFaces Orchestra introduces the new scope useful for managed beans. For example you can get a “conversation” with synchronized life time with the database session. Which enables you to use “data transfer objects”, use the entities in the view directly. You will feel comfortable to keep things as automatic version checking without thinking about it and gain from the persistence context session cache.

Virtual Ant Beta Released - A GUI for Ant

Virtual Ant is developed by Placid Systems.

Virtual Ant enables to create and edit Ant build scripts through a complete virtualized environment similar to Windows Explorer. It lets you visualize the tasks running on a Virtual File System without affecting your real file system. The actual Ant build script is generated in the background. You can even work with your existing build scripts.

Virtual Ant is particularly useful in creating build scripts where the source code is not complete or compilable, since operations take place on a virtual file system and don’t actually rely on content except when absolutely necessary.

Cooee 1.0 released

Recently Karora, an open source group released a Java based web application UI framework **named as** Cooee**. Cooee builds rich, AJAX based applications without writing any JavaScript and html tags. It allows developers to work entirely against a Swing based Java API.**

New release provides extra functionality that enables developers :

- to set the TabPane width.
- to set the TabPane height.
- to set the the TabPane font.
- to render any component into a TreeTable
- to fire events between tabs in the TabPane for the switching.
- to fix the constructor for the TreeModelEvent.
- to refresh of TreeTables manually.
  OSGI Support

API Updates

Struts 2.0.8 Released

The Apache Project has released Struts 2.0.8 an open source framework for building web applications. Apache Struts 2 is an elegant, extensible framework. It uses Java Servlets and JavaServer Pages (JSP) for creating enterprise-grade web applications. Earlier it was known as WebWork 2. The design of Apache Struts 2 is based on the Model-View-Controller (MVC) design pattern and is streamlined to maintain applications over time. Moreover it has over 60 bug fixes since 2.0.6 and it is simpler to use. After working independently for several years, the WebWork and Struts communities joined forces to create Struts 2.

Java Servlet 3.0 Specification

Sun has submitted Java Specification Request -315, Java Servlet 3.0 Specification , to the Java Community Process.

One of the goals for Java EE 6 is extensibility. Servlet 3.0 specification will work on extensibility / pluggability. Web framework pluggability will be a key driver of the Servlet 3.0 specification. As part of the revision for Java EE 6 JCP would like to revise the Servlet specification to support the Ease of Development (EoD) using the newer language features. Also along with EoD there have been requests for enhancements in other areas as well to support the modern next generation web application development.

Java Caching System 1.3 Released

Java Caching System (JCS) 1.3 has been released by the Apache Jakarta Project. It is useful in speeding up the applications and is highly useful for high read, low put applications. Basically Java Caching System (JCS) is an open source “distributed caching system written in java. It manages cached data of various dynamic natures and sharply drops latency times hence it speed up applications. JCS works on JDK versions 1.3 and up.

The features provided by JCS are:

- Element grouping
- Minimal dependencies
- Region data separation and configuration
- Fine grained element configuration options
- Memory management
- Element event handling
- Remote server chaining (or clustering) and failover
- Remote synchronization
- Disk overflow (and defragmentation)
- Thread pool controls
- UDP Discovery of other caches
- xtensible framework
- Non-blocking “zombie” (balking facade) pattern
- Lateral distribution of elements via HTTP, TCP, or UDP

Apache POI 3.0 Released

POI is a Java library which is used to manipulate few of the file formats based upon Microsoft’s OLE 2 Compound Document format(like Microsoft Office files such as XLS and DOC). Now with POI 3.0, MS Excel files can be written from Java with few more functionalities. This version adds some PowerPoint support, as well as some formula support. Even in near future, Word files will be able to read and write using Java. POI is Java Excel solution as well as Java Word solution.

Collections 3.0 Released

To complement the Java Collections Framework Apache group has released its Commons-Collections Framework with new features in form of Collections 3.0 API. It possesses enhancements, implementations and utilities over the previous releases. New interfaces, new implementations and utility classes have been added These features have been added by the Apache Jakarta Commons Collections Framework component.

Commons-Collections was created to share collection implementations created in various places around Jakarta. Initiallly, there was no clear design or structure to the component. However, with this 3.0 release, a new consistent package structure is defined. Additionallly two new code donations are launched which are primitive collections and event notification collections .

Common Java Compiler Interface 1.0 Released

Common JCI 1.0 is a java component released by Apache Jakarta Project. Java and other JVM hosted languages can be compiled with the help of it. Basically JCI is a java compiler interface which compiles Java or other languages like groovy or javascript to Java. It is well integrated with a FAM (Filesystem Alteration Monitor) that can be used with the JCI compiling/ reloading classloader. All the currently supported compilers (even javac before Java 6.0) feature in-memory compilation. The current implementation supports compilation via the following compilers:

- Groovy

- Rhino
- Javac
- Eclipse
- Janino

Java Speech API 2.0 : JSR 113

Java Specification Request 113, the Java Speech API 2.0 has been posted by the Conversational Computing Corporation as a second proposed final draft.

With the Java Speech API (JSAPI 2.0) the speech technology can be incorporated by the developers into user interfaces. This is useful for their Java programming language applets (or MIDlets) and applications. It is based on J2ME platform. The API can be scaled in several ways which may look large at the first glance. This API provides a platform to support command and control recognizers. The engines such as Recognition and Synthesis engines used may be small or large depending on their capabilities. For instance,

Recognition engines may provide full support for command and control or provide more limited support through specialized built-in grammars.
Synthesis engines may support full text-to-speech capabilities or audio sequencing only.
It also works well on J2SE. To support this compatibility as well as to aid the developer with familiar programming methods, few small building blocks have been included in the API.

The uses of JSAPI 2.0 are:

- Application switching
- E-mail interaction including both reading and manipulation
- Calendar management and interaction
- Accessibility including screen reading
- System alerts including low memory or low fuel
- Car navigation system
- Games
- Learning
- Data entry

JSAPI 2.0 is especially well suited for downloadable applications, opening rich new interaction possibilities on devices with limited size yet unlimited potential. Capabilities for applications include the ability to

- easily switch between applications,
- prioritize applications, especially between trusted and untrusted applications, and
- support multimodal interaction

Google's Java-Checkout-API : new version

Google aims to organize the world's information and make it universally accessible and useful. Google provides around 32 different Google APIs for free. Few of the E-commerce centric APIs are Checkout API, Order Processing API etc.

Google Checkout lets the customers buy items quickly and securely using a Google username and password. It can be used by E-commerce sites for customers to checkout to charge customer's credit cards, track orders through a fulfillment process and receive order payments in its bank account. As such, Google Checkout touches each step of the customer's shopping experience, beginning with the customer's search for an item and continuing through the order checkout and fulfillment processes.

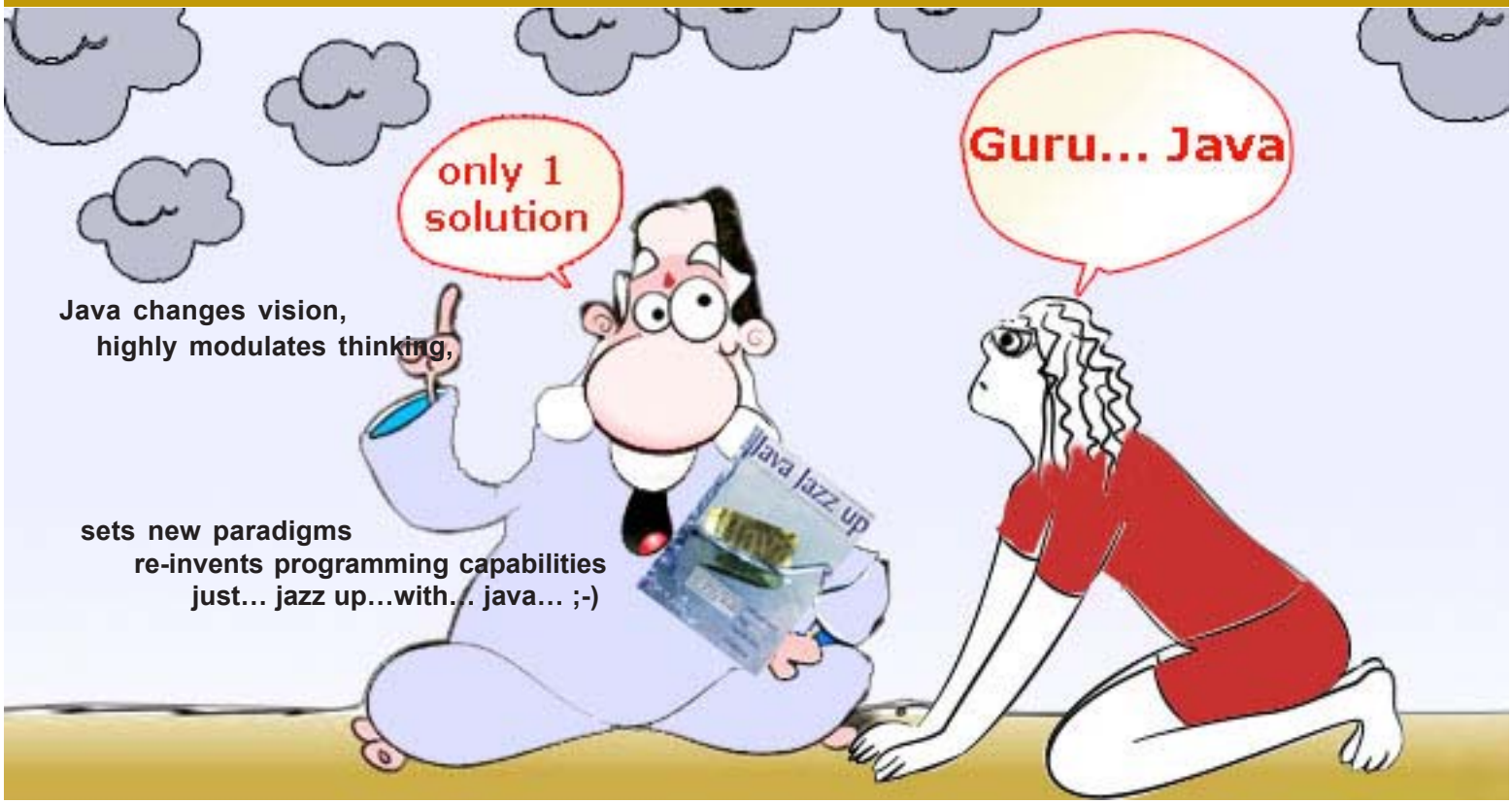
The reference implementation of Java Checkout sample code uses the org.w3c.* classes to provide a fully functional implementation of the Java Checkout Api interfaces. It is designed to be used as a library which can be dropped into your existing development environment. It depends on the JavaCheckoutApi.

It provides an implementation of the outgoing APIs only - **Checkout API** and **Order Processing API**. Typically, the logic required for the incoming APIs is heavily merchant-specific and therefore does not belong in a library. The web app and J2EE app examples contain default implementations. In future releases, Google will extend the API and reference implementations to include reference implementations of the incoming classes, exposing a strongly typed API.

The features of this new version of the Java Checkout sample code are as follow:

- Provide clear customization points to avoid the user to have non-standard requirements.
- Clearly defined API for the developer by avoiding the use of static helper style classes.
- For the 80% of merchants who don't want to customize or build, ship the library as a single JAR + Javadoc.
- Allow only core JDK types on the public API interfaces therefore independent of JDK version.
- Make the .NET and Java sample code APIs similar, to reduce support overhead.
- Shipped as a library which can be dropped into a project. Compatible with the widest possible range of JDK versions (1.3.x, 1.4.x, 1.5.x)

Java Developers Desk



Java is the essential ingredient of the digital experience for hundreds of millions of people in all walks of life, all over the planet.

Java software powers the onboard computers in toys, cars, planes, rockets, and even the NASA Mars Rover. It brings interactivity to the Internet, real-time graphics to television, instant imaging to cameras, and multi-player games to mobile phones and desktop PCs. It connects the largest enterprises and smallest businesses to their employees, customers, and data. And it secures the vast majority of electronic transactions in retail, finance, government, science, and medicine. In short, Java technology goes everywhere you go.

If you're looking for a way to do something that's never been done before, you've come to the right place…

Jazz-up with java, learn to develop the most thrilling, efficient, fast, secure,compatible, and reliable software.

Be platform independent, have control to the widest range of gadgets with java from mobile phones, computers, washing machines, refrigerators, satellites, cars...

Ensure quality through java as it is least prone to errors due to its superb error handling mechanism.

Be International and go global with java as it is highly suited to internet environment.

Be a real life programmer with java as it is Object Oriented.

Develop applications in all spheres with java

- stand alone applications like games, desktop publications
- Enterprise applications dealing with finance, defense ,insurance, banks
- Embedded applications used in automated systems, robots , flying machines
- Micro Device applications used in cell phones, PDAs, TV set-top boxes
- Java is always a fun to develop with. Just Imagine and get it done in JAVA.

Nut-Bolts of Java:

a) Java SE - Java SE (Java Standard Edition) provides tools and API's to create diverse applications. Applications developed with Java SE are supported by every operating system, including Linux, Macintosh, Solaris, and Windows.

b) Java EE - Java Enterprise Edition specifications based on the foundation framework of the standard edition. Java Enterprise Edition are the specifications needed to service the multi-tiered environment, to support the enterprise class service oriented architecture (SOA) and a lot

c) Java ME - Java Micro Edition is an accumulation of Java APIs used to develop micro-devices applications like mobile phones, PDAs, TV set-top boxes, game programming. The platform of micro edition generally consists of an easy user interface, a robust security model and a wide variety of built-in networks for running Java based application.

A collections framework is a unified architecture for representing and manipulating collections that reduces programming efforts. It provides

High-performance and high-quality implementations of useful data structures and algorithms, thus it frees developers to concentrate on the important parts of a program i.e. quality and performance rather than worrying about the low-level issues required to make it work.

Introduction to Java Collections API

Java Collections API supports, a set of data structures that includes ArrayList, Map, HashSet, etc. A collection is simply an object that groups multiple elements into a single unit. It can be thought as a container that is used to store, retrieve, manipulate, and communicate aggregate data.

It groups data items and allows accommodating duplicate elements. It may consist of both ordered and unordered elements.

Earlier versions of the Java (pre-1.2) included generalized collection implementations like vector, hashtable, and array but not the collections framework. Hence the new version of the Java platform contains the collections framework.

Let's explore the collections framework

Collections Framework

A collections framework is used to represent and manipulate collections. Basically it is a unified architecture that consists of:

1. Interfaces: These are abstract data types that represent collections. With the help of interfaces we can manipulate collections without worrying about their implementation details. A hierarchy is generally formed by interfaces in object-oriented languages.

2. Implementations: They are the reusable data structures with the concrete implementations of the collection interfaces.

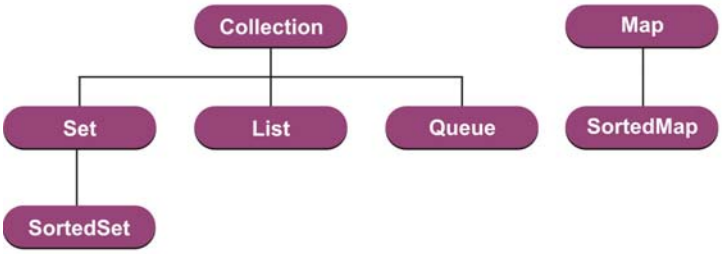
3. Algorithms: Algorithms are used to perform computations, such as searching and sorting, on objects that implement collection interfaces. They provide reusable functionality i.e. the same method can be used with different implementations of collection (appropriate) interface. Hence they are said to be polymorphic.

4. General-purpose Implementations: These are general implementations of the collection interfaces at the primary levels.

5. Infrastructure: They are the interfaces that provide essential support for the collection interfaces.

6. Array Utilities: Provides classical utility functions for arrays of primitives and reference objects. This utility was added to the Java platform as a part of the Collections Framework.

The figure below shows the hierarchy of collection interface.



Advantages of collections framework

1 A programmer need not to learn multiple ad hoc collection APIs.
2 It provides a standard interface for collections that fosters software reuse along with the algorithms to manipulate them.
3 It provides ready-made data structures and algorithms that reduce programming effort by eliminating the need to write them from scratch.
4 It provides high-performance implementations of useful data structures and algorithms that increase the performance.
5 Helps in establishing a common language to pass collections back and forth that provides interoperability between unrelated APIs.

Introduction to Java 6.0 Collections API

Some of the new collections APIs have been introduced in Java 6.0. These are:

1. Deque: It is used to represent a double ended queue. This collection helps to add or remove elements at both the ends. Deque implementation can be used as a stack (Last in first out) or as a queue (First in First Out). There are methods in Deque used for insertion, retrieval and removal of elements. All the Deque methods exist in two forms, one method returns status or special value for each operation and the other one throws exception if it fails in an operation.

2. BlockingDeque: It is similar to Deque but with added functionality. When we try to insert an element in a BlockingDeque and if the blockingDeque is already full then the element waits till the space becomes available to

insert an element. There are four operations defined for each method specified in the BlockingDeque.

- •Method may throw exception.
- •Method may wait for a given time if times-out.
- •Method that blocks may wait indefinitely for space to be available.
- •Method returns special value.

3. NavigableSet : It is used to return the closest matches of elements. For instance if it is needed to retrieve an element, which is immediately greater than, or lower than element 20 or if we want to retrieve all elements greater than or lower than 35 from sorted set elements [10,20,35,5] we can use NavigableSet. **ConcurrentSkipListSet** is one of the class that implements NavigableSet.

4. NavigableMap: Unlike the NavigableSet (return values), NaviagableMap methods is used to return the key-value pair. **ConcurrentSkipListMap** is the one of the class which implements NaviagableMap

New classes have been introduced in Java 6.0 collections APIs. These are:

1. ArrayDeque: ArrayDeque is a class that implements Deque. If used as a stack or linked list, it performs much faster. It is neither thread safe nor does it has the capacity restrictions.

2. LinkedBlockingDeque: It implements BlockingDeque. Maximum capacity can be specified by using BlockingDeque interface. However the maximum capacity will be Integer.MAX_VALUE if not specified.

3. ConcurrentSkipListSet: ConcurrentSkipListSet is one of the class that implements NavigableSet. It is used to return the closest matches of elements.

4. ConcurrentSkipListMap: ConcurrentSkipListMap is one of the class which implements NaviagableMap. In NavigableSet, methods are used to return values.

5. AbstractMap.SimpleEntry: The key-value pair of a single entry in a Map is held by the instance of this class. Moreover it is a static nested class which is nested inside an abstractMap class.

6. AbstractMap.SimpleImmutableEntry: This class is similar to AbstractMap.SimpleEntry class however it has one difference that it throws the exception UnsupportedOperationException when we try to set an improper value.

Modified Classes in Java 6.0 collections APIs

- LinkedList
- TreeSet
- TreeMap
- Collections

Some existing classes have been modified by implementing the new interfaces. **TreeSet class** is have implemented the NavigableSet, **LinkedList** class is modified to implement Deque, similarly **TreeMap** is modified to implement NavigableMap etc. Some of the new methods like **newSetFromMap** and **asLifoQueue** have been added to Collections 2.

Bi-directional traversal has become easier with java 6.0 collection APIs.

Let's see the features of the NavigableMap using an example.

In the example below the NavigableMap methods are used to return the key-value pair element from the specified collection object location like retrieving first key-value pair, retrieving last key-value pair, retrieving the greatest key strictly less than the given key, retrieving a key-value associated with the least key strictly greater than the given key, removing the first key-value pair, removing the last key-value pair etc..

```
import java.util.*;
import java.util.concurrent.*;
public class NavigableMapExample
{
public static void main(String[] args)
{
System.out.println("Navigable Map Example!\n");
NavigableMap <Integer, String>navMap =
new ConcurrentSkipListMap<Integer, String>();
navMap.put(1, "January");
navMap.put(2, "February");
navMap.put(3, "March");
navMap.put(4, "April");
navMap.put(5, "May");
navMap.put(6, "June");
navMap.put(7, "July");
navMap.put(8, "August");
navMap.put(9, "September");
navMap.put(10, "October");
navMap.put(11, "November");
```

```
navMap.put(12, “December”); //Displaying all data
System.out.println(“Data in the navigable map: “ +
navMap.descendingMap()+”\n”);   //Retrieving first data
System.out.print(“First data: “ + navMap.firstEntry()+”\n”);
                                                    //Retrieving last data

System.out.print(“Last data: “ + navMap.lastEntry()+”\n\n”);
//Retrieving the nearest less than or equal to the given key
System.out.print(“Nearest less than or equal to the given key:
“ + navMap.floorEntry(5)+”\n”);
//Retrieving the greatest key strictly less than the given key
System.out.println(“Retrieving the greatest key strictly less
than the given key: “ + navMap.lowerEntry(3));
//Retrieving a key-value associated with the least key strictly
greater than the given key
System.out.println(“Retriving data from navigable map greater
than the given key: “ + navMap.higherEntry(5)+”\n”);
//Removing first
System.out.println(“Removing First: “ +
navMap.pollFirstEntry());              //Removing last
System.out.println(“Removing Last: “ +
navMap.pollLastEntry()+”\n”);       //Displaying all data
System.out.println(“Now data: “ + navMap.descendingMap());
}}
```

Output:

```
C:\Javabasics\collection>javac NavigableMapExample.java

C:\Javabasics\collection>java NavigableMapExample

Navigable Map Example!Data in the navigable map:
{
12=December, 11=November, 10=October, 9=September,
8=August, 7=July, 6=June, 5=May, 4=April, 3=March, 2=Feb-
ruary, 1=January
}

First data: 1=January
Last data: 12=December

Nearest less than or equal to the given key: 5=May

Retrieving the greatest key strictly less than the given key:
2=February

Retriving data from navigable map greater than the given key:
6=June

Removing First: 1=January
Removing Last: 12=December

Now data: {11=November, 10=October, 9=September, 8=Au-
gust, 7=July, 6=June, 5=May, 4=April, 3=March, 2=February}

C:\Javabasics\collection>
```

Now, let’s see the features of the NaviagableSet **using an example.**

In the example below the NavigableSet method is used to sort the elements in ascending order, descending order and to retrieve the element which is immediately greater than or equal to 35 etc by returning the closest matches of elements for the given elements in the collection.

```
import java.util.*;
import java.util.concurrent.*;
public class NavigableSetExample {
public static void main(String[] args) {
System.out.println(“Navigable set Example!\n”);
NavigableSet <Integer>nSet = new
ConcurrentSkipListSet<Integer>();
nSet.add(10);
nSet.add(20);
nSet.add(50);
nSet.add(30);
nSet.add(100);
nSet.add(80);// Returns an iterator over the elements
in navigable set, in ascending order.
Iterator iterator = nSet.iterator();
System.out.print(“Ascending order navigable set: “);
//Ascending order list
while (iterator.hasNext()){
System.out.print(iterator.next() + “ “);
}
System.out.println(); //Descending order list
System.out.println(“Descending order navigable set: “
+ nSet.descendingSet() + “\n”);
//Greater than or equal to the given element
System.out.println(“Least element in Navigable set
greater than or equal to 35: “ + nSet.ceiling(35));
//Less than or equal to the given element
System.out.println(“Greatest element in Navigable set
less than or equal to 35: “ + nSet.floor(35) + “\n”);
//Viewing the portion of navigable set whose elements
are strictly less than the given element
System.out.println(“Navigable set whose elements are
strictly less than ‘40’: “ + nSet.headSet(40));
//Viewing the portion of navigable set whose elements
are greater than or equal to the given element
System.out.println(“Navigable set whose elements are
greater than or equal to ‘40’: “ + nSet.tailSet(40) +
“\n”);
//Removing first element from navigable set
System.out.println(“Remove element:
“+nSet.pollFirst());
//After removing the first element, now get navigable
set
System.out.println(“Now navigable set: “ +
nSet.descendingSet() + “\n”);
//Removing last element from navigable set
System.out.println(“Remove element: “ +
```

```
nSet.pollLast());
//After removing the last element, now get navigable
set
System.out.println("Now navigable set: " +
nSet.descendingSet());   }}
```

Output:

```
C:\javac>javac NavigableSetExample.java

C:\javac>java NavigableSetExample
Navigable set Example!

Ascending order navigable set: 10 20 30 50 80 100
Descending order navigable set: [100, 80, 50, 30, 20,
10]

Least element in Navigable set greater than or equal
to 35: 50
Greatest element in Navigable set less than or equal
to 35: 30

Navigable set whose elements are strictly less than
'40': [10, 20, 30]
Navigable set whose elements are greater than or
equal to '40': [50, 80, 100]

Remove element: 10
Now navigable set: [100, 80, 50, 30, 20]

Remove element: 100
Now navigable set: [80, 50, 30, 20]

C:\javac>
```

JSF- Java Server Faces

"Smart way to develop Web Applications"

It is a natural phenomenon to think about learning and adopting new technologies while there exists some well-established and popular ones. It is the scenario prevailing with web development landscape. Recently JSF has evolved as a new emerging and robust web technology.

JSF is developed by JCP (Java Community Process), a community of web application experts from different groups like Jakarta Struts, Oracle, Sun, IBM and ATG etc. They all collectively worked together to extract the best of existing technologies in order to simplify various web application development issues. Their collective effort brought a new technology named Java Server Faces (JSF). It is the best combination of features derived from all the available frameworks.

JSF follows the Model-View-Controller (MVC) design pattern. It is simple to develop a well-designed application with JSF framework and easy to maintain than an application developed with JSP and Servlet. JSF can be thought as a toolbox containing the readymade components. Just fetch and use the components any number of times in a page and get the result-oriented actions. JSF made nesting of components possible.

JSF is a robust component framework and based on event driven programming model. It offers a set of UI components, extensible architecture, supports multiple client devices etc. The UI (user interface) created using JSF technology runs on server and output is shown to the client. Developers can focus on UI components, events handling, backing beans and on their interactions rather than worrying about request, response and markup. JSF hides complexities to enable developers to focus on their specific tasks. JSF is a vendor neutral technology and its implementations must comply with JSF specification.

Why JSF: Let's explore the necessities initiating the JSF technology. There are reasons behind the development of new framework in spite of existence of numerous web technologies like JSP, Servlets, and others. Few of the problems faced with these technologies are:

Directly working with HTTP Request and Response

Using these technologies programmers directly work with HTTP request and response objects and manipulate the data.

Tedious and repetitive coding

Technologies like JSP forces a programmer to do a lot of tedious and repetitive coding.

Non-availability of IDE

Non availability of IDE is another major drawback, which affects the programmer's productivity and hence it increases the overall project cost.

JSF simplifies the development of web application. Some points highly advocates JSF :

- JSF provides standard, reusable components for creating user interfaces for web applications.
- JSF provides tag libraries for accessing and manipulating the components.
- It automatically saves form data and repopulates the form when it is displayed at client side.
- JSF encapsulates the event handling and component rends logic from programmers, programmers just use the custom components.
- JSF is a specification and vendors can develop the implementations for JSF.

There are many GUIs available these days to simplify the development of web application based on JSF framework.

JSF Components: JSF components mainly includes:

I Set of APIs to represent and manage state of components for server side validation, event handling, page navigation, data conversion etc.

II JSP custom tag library in order to create UI components in a view page.

JSF changes all that by giving intuitive framework to the developers. Furthermore, JSF is a specification and many vendors are developing their own implementations. Both free and commercial implementations of JSF are available these days. You can choose any one of them based on your requirement and budget.

Now a days software vendors are developing IDEs to develop JSF based applications, which is another good news for the learners of JSF framework. Once you are familiar with the core concepts of JSF you can kick start the development of software projects using any IDE available in the market. Such changes in the programming world make the life of programmer much easier.

JSF Features:

JSF is rich featured that make it much popular than the existing frameworks. Some of the JSF features are :

- **I** JSF is standard web user interface framework for Java.
- **II** Built on top of Servlet API.
- **III** JSF is a component framework
- **IV** UI components are stored on the server.
- **V** Easy use of third party components.
- **VI** Event driven programming model.
- **VII** Events generated by user are handled on the server.
- **VIII** Navigation handling.
- **IX** It can automatically synchronize UI components.
- **X** JSF supports multiple client devices.
- **XI** JSF has extensible architecture.
- **XII** International language support.
- **XIII** Extensive tool support (Sun, Oracle, IBM etc.).
- **XIV** Rapid application development approach.

How JSF Fits For Web Applications?

JSF best suits the java web development environment because of following reasons described below:

Easy creation of UI:

It makes easier to create complex UI for an application using JSF tags. Its APIs are layered directly at the top of served APIs that enable us to use presentation technology other than JSP, creating your own custom components and rendering output for various client devices.

Capacity to handle complexities of UI management:

It handles cleanly the complexities of UI management like input validation, component-state management, page navigation, and event handling.

Clean separation between presentation and logic:

One of the greatest advantages of JSF is its ability to clearly separate behavior and presentation with in an application. JSF is based on the Model View Controller (MVC) architecture.

Shorter development cycle:

The separation between logic and presentation enables a wide range of users (from web-page designers to component developers) to focus on their own task only, resulting in division of labor and shorter development cycle.

Standard Java Framework:

JSF is a Java standard, which is being developed by Java Community Process (JCP). Several prominent tool vendors are members of the group and are committed to provide ease to use and productive development environments for Java Server Faces.

An extensible architecture:

JSF architecture has been designed to be extensible. Extensible means additional functionality can be given on the top of JSF core i.e. we can customize the functionality. JSF UI components are customizable and reusable elements. You can extend standard components and create your own complex components like stylish calendar, menu bar etc.

Support for multiple client devices:

Component developers can extend the component classes to generate their own component tag libraries to support specific client. JSF's flexible and extensible architecture allows developers to do so.

Flexible rendering model:

Flexible rendering features separates the functionality and view of the component. So we can create multiple renderers and give them different functionality to get different appearance of the same component for the same client or different.

Support for Internationalization:

Java has excellent support for internationalization. It allows you to localize messages with user specific locale. A locale is a combination of a country, a language, and a variant code. Java Server Faces adopts this property and lets you specify which locale your application needs to support. So you can display messages in different languages.

Robust tool support:

There are several standard tool vendors like Sun Java Studio Creator which provide robust tools that take advantages of JSF to create server side UI easily.

Eclipse IDE

Integrated Development Environment (IDE)

Generally, a java programmer starts programming with a notepad. To compile and run a program, a programmer uses javac and java commands at the command prompt window. Notepad doesn't help a programmer to track the improper java syntax while programming. This makes programming a bit uncomfortable. Hence, Integrated Development Environments (IDEs) are developed to provide best solution to ease the programmer's task.

IDE is a smart editor that equips a programmer with features like editing, compiling, deploying, debugging etc.. IDE abstracts the programming complexities and reduces the applications development time hence it enhances the productivity. IDE may provide services itself or enables using plug-ins developed and provided by third party. A programmer can choose IDEs according to the set of tools and features required.

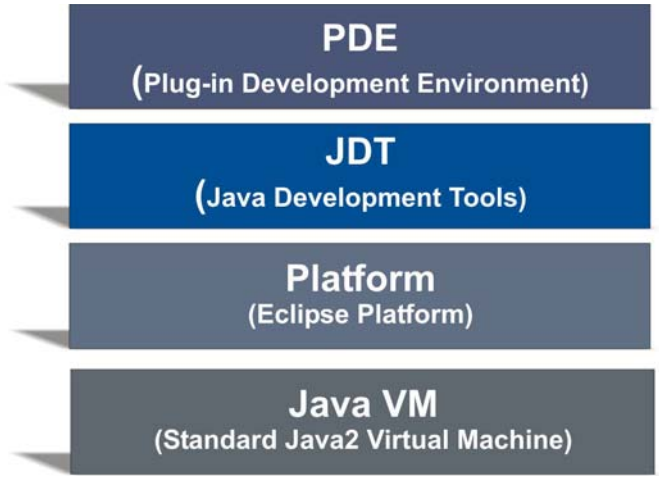
IDEs have become very powerful as RAD (Rapid Application Development) solutions. Most of the available IDEs understand java and its structure that enables to develop a better code. There are many IDEs in use like **Eclipse, JDeveloper, JBuilder, NetBeans, Sun Java Studio Creator** and many more. **Sun** has provided **NetBeans, Sun Studio, Java Studio Enterprise (JSE), Java Studio Creator (JSC)**.

An IDE may include:

- **Wizard** that generate program for particular technology.
- Support for automatic **unit testing**.
- Wizard to import and export UML data into XMI format. This enables the programmer to interchange UML data between different applications.
- Generation of **jars, wars, ears** at deployment time.
- Support for custom **Plug-In**.
- Support for **debugging** the code.
- Plug-In for different application servers.
- Memory profiler and Memory leakage detection tools

Starting with Eclipse IDE

This is **free** and **open source, extensible** IDE written in Java. This is one of the most popular IDE used among Java community. Eclipse **Plug-Ins** are available for **C/C++,Cobol, PHP, JSP/Servlet, J2EE** and many more. These Plug-Ins can be used with the same IDE at the same time. They all have their own debuggers and integrated IDE options. It includes Java development tools and uses its own compiler that compiles instantly as you type. Eclipse is supported well on **Windows 98, Windows XP, Windows ME, Windows 2000, Windows ME, Linux, Solaris** and other systems as well.



Eclipse Architecture

Working with Eclipse:

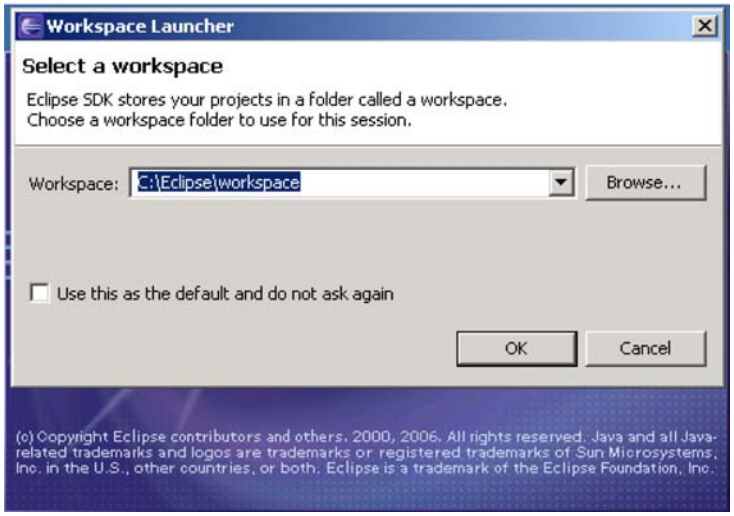
To use Eclipse in your system, it is required to download its appropriate copy along with java runtime environment. Downloading and installation steps has been summarized below:

Downloading Eclipse: Eclipse can be obtained from eclipse's home page **http://www.eclipse.org/downloads/index.php**. Download the latest release of eclipse as **.zip** file named as **"eclipse-SDK-3.2.2-win32.zip"**.

Installing Eclipse: Unzip the downloaded file into the directory of your choice (**C:\eclipsewin32**).This creates another folder named **eclipse** containing all the unzipped files and folders. This is first installation step for Eclipse.

Initializing Eclipse: Running eclipse requires following steps:

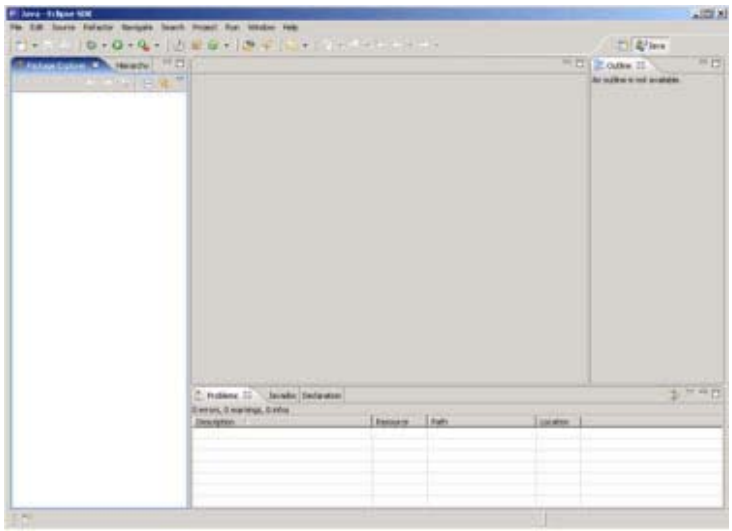
1. Double Click the **"eclipse.exe"** file in the **eclipse** folder.
2. Now it will ask to specify a **workspace** to use (**C:\Eclipse\workspace**). Press **OK** to continue.
3. If Eclipse has been opened first time with this workspace then a **welcome screen** is opened. Rolling over the icons provides you help facilities. Click on **arrow like icon** on the **right side** of the screen.



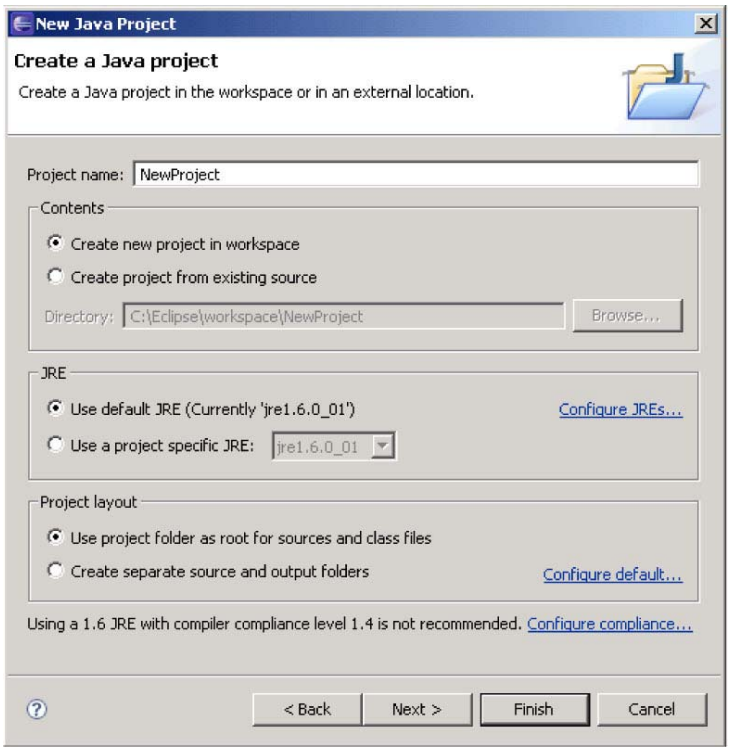
4. Clicking on the **arrow** like icon opens a **blank eclipse window**.

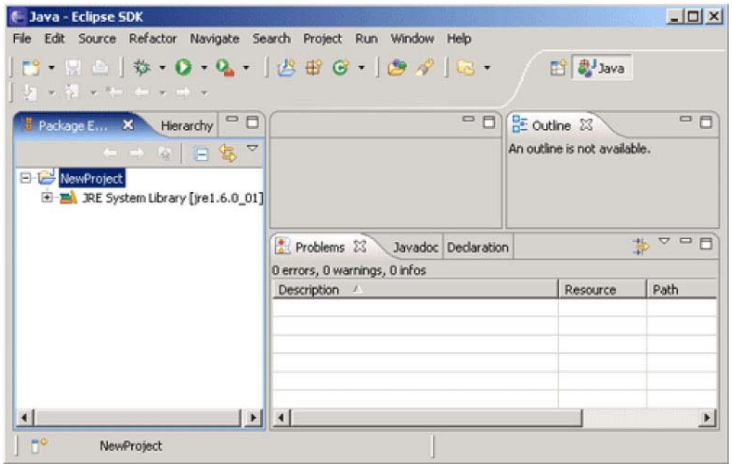
Creating a Project:

1. Now, go and click to **file->new->project** icon.



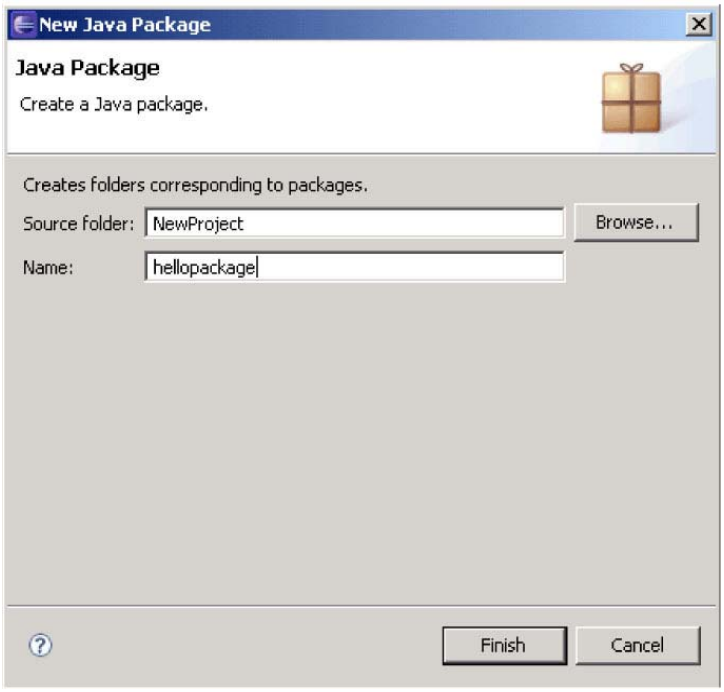
Click on **"Java Project"** and give a project name of your choice **("NewProject")** and click **Finish**. This project will appear in the **"Package Explorer"** area on the left panel. Some libraries are added automatically in the project. You can check it expanding the view by clicking on the **+** sign.





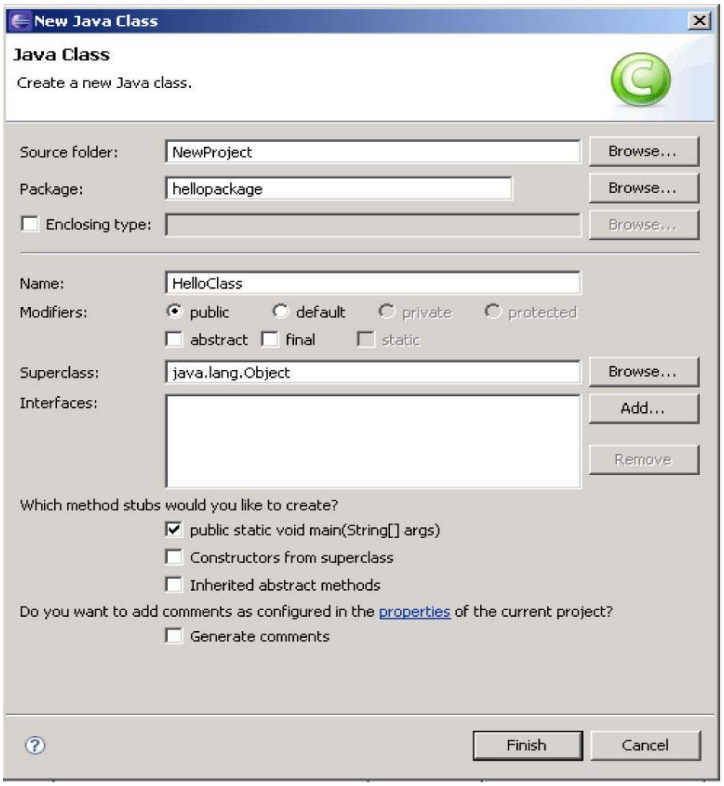
2. Right click on the project name **NewProject** symbol and click on the **new->package**. Specify the name of your choice for the package **("hellopackage")** and click **Finish**. This appears in the **"Package Explorer"** area on the left under the project name **NewProject**.

3. Right click on the package name **hellopackage**



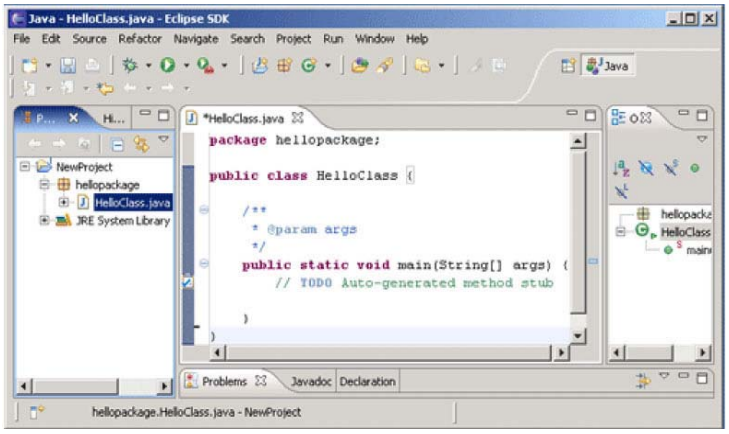
and click on the **new->class**. Specify the name of your choice for the class **("HelloClass")**. Unclick **Inherited abstract methods**, and click **public static void main(String[] args).** Click radio button labeled **public** to select modifier for a simple program and click **Finish**.

4. Now a window opens containing a java program with the **main()** method. This program will be according to the information provided before.

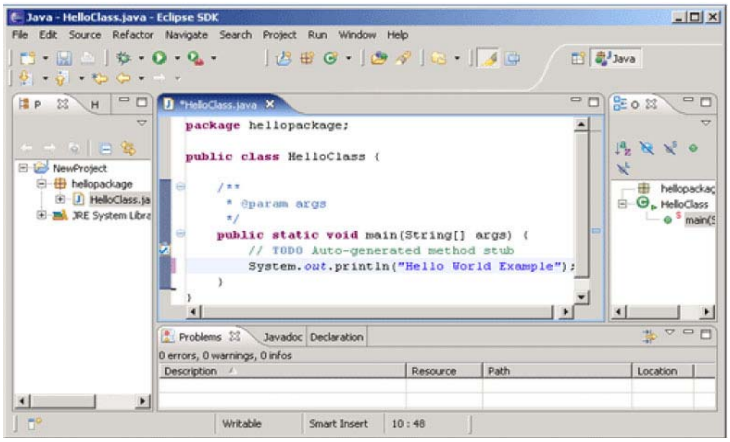


5. You can now modify the program according to your requirement.

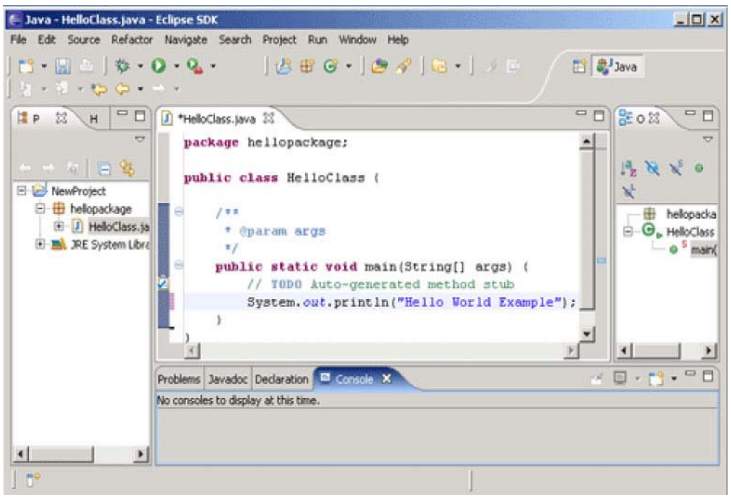
6. Click **Window->Show View->Console**. A **console window** will open at the bottom. If a **Console tab** is already there, just **click** on it.



7. Eclipse compiles the program as we type. If there is no any **red mark** in the window of the program, it means program is syntactically correct. Save this file

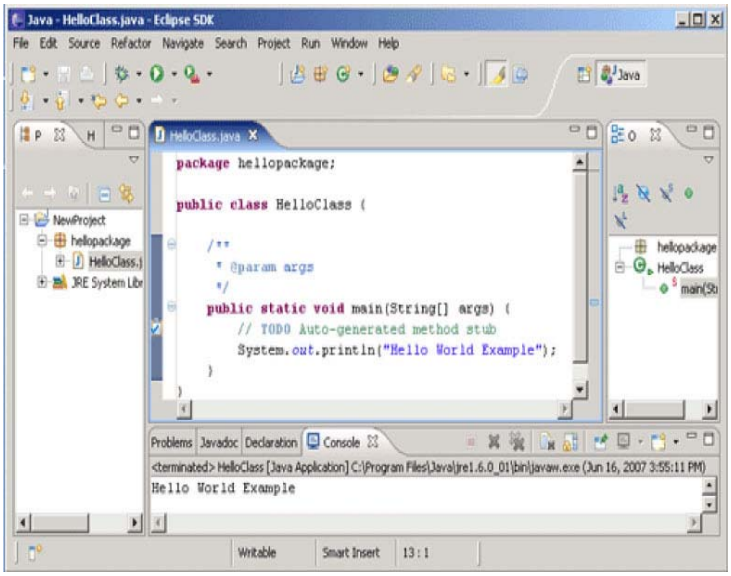


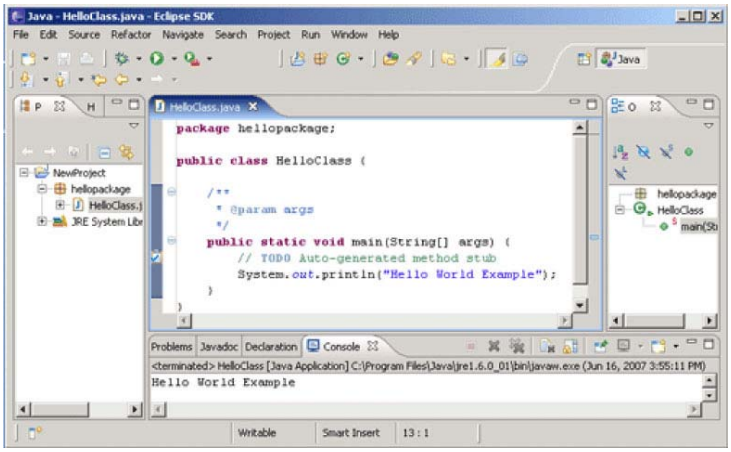
clicking on **File->Save** or press **Ctrl+s** in the keyboard. If there is any error then you will see **red circle** on the **left**

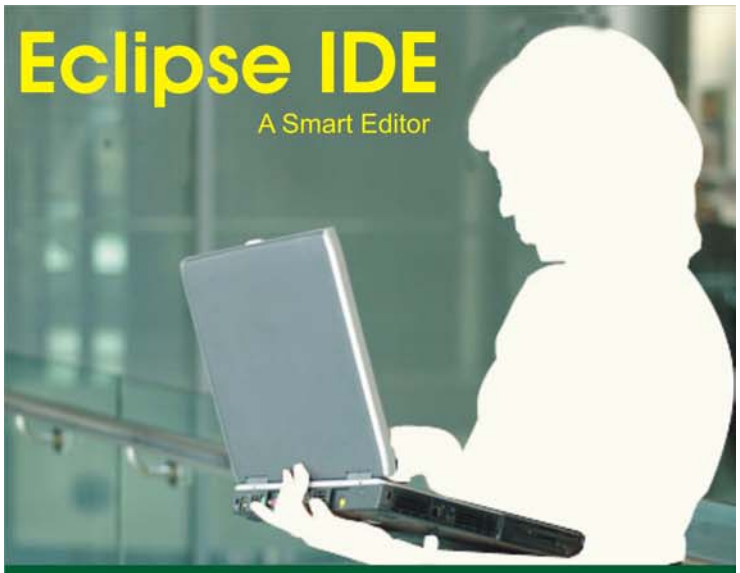


of the window of the program. Click on the **Problems tab** to see errors. Fix these errors and save the program.

8. Click **Run->Run as->Java application**. The **output** will appear in the **Console window** below.







Generally, a java programmer starts programming with a **Notepad**

IDE is a smart editor that equips a programmer with features like editing, compiling, deploying, debugging etc.. IDE abstracts the programming complexities and reduces the applications development time hence it enhances the productivity. IDE may provide services itself or enables using plug-ins developed and provided by third party. A programmer can choose IDEs according to the set of tools and features required.

Spring

Spring is an open-source application framework, introduced and developed in 2004. The main ideas were suggested by an experienced J2EE architect, Rod Johnson

The Spring framework is a MVC Architecture for building Web applications. Its pluggable MVC architecture provides options for the developers to use the built-in Spring Web framework or an existing Web framework such as Struts, Saveria etc. Spring accommodates multiple view technologies such as JSP technology, Velocity, Tiles etc. Even, Spring framework is highly configurable because of its strategy interfaces.

Spring Salient Features

Spring stresses the OO design issues rather than focusing on the implementation aspects of a technology, such as J2EE. And above all, the application code does not depend on the Spring APIs unlike the EJB that force to use JNDI and the Struts that force to extend Action. Spring promotes the easy implementation of JEE technology. It aims to simplify the JEE development and testing. Spring's functionality can be used in any JEE server.

Spring focuses on interface-oriented programming rather than keeping it class-centric. Spring reduces the complexity cost of using interfaces to zero.

Unlike java, Spring framework avoids the overuse of the checked exceptions by allowing a user not to catch exceptions.

Testability is an essential feature provided in Spring that helps to test the user code very easily.

Spring does not attempt to do everything itself but supports the best breed of technologies, For instance... it supports several persistence technologies like JDO, Hibernate and OJB as these ORM tools are highly capable.

Spring offers a great way of configuring applications through JavaBeans. Programmers find it easy to work with these standard JavaBeans naming convention.

Spring Flexibility issues that distinguishes it from other JEE Frameworks

Spring focuses on the reusability of business and data access objects (by not keeping them specific to any of the JEE services and environments like web, EJB etc). It keeps them loosely coupled.

In addition to classic OOP, Spring uses AOP(Aspect-Oriented Programming) which is a very recent and useful paradigm to promote the separation of concerns within a system. Hence it makes Spring highly flexible.
Spring's goal is to emerge as an entire Application Framework. Other popular frameworks like **Struts, Tapestry, JSF** etc., are very good web tier frameworks but when we use these framework, we have to provide additional framework to deal with enterprise tier that integrates well with these framework. Spring tries to alleviate this problem by providing a comprehensive framework, which includes a **core bean container, an MVC framework, an AOP integration framework, a JDBC integration framework and an EJB integration framework.**

Sometimes it is not possible to completely switch to a different framework. Spring does not force to migrate completely rather it allows to perfectly integrate with the existing technologies. Spring provides various ready-made adapters for various hot web-tier and presentation technologies. For example, there exists a variety of technologies in the web-tier like STRUTS, JSF, MVC PATTERN, WEB-WORK, TAPESTRY, FREEMARKER, JSP etc. Developers are often puzzled and confused about the relative merits and demerits of all these. Once they choose a technology and start implementing and later want to change over to another technology, it is very difficult. Spring offers various modules supporting a different technology, it just requires to change the configuraion file. With this approach, it is even possible for a development team to try and test a given task in all the above forms and see the effect and performance before deciding the choice. Spring offers its own version of MVC architecture. It also offers adapters for Struts.

Spring provides proxying for RMI (special remoting technologies like Burlap) JAX-RPC & web-service. Spring makes use of Acegi, an open-source Security framework and provides declarative security through its configuration file.

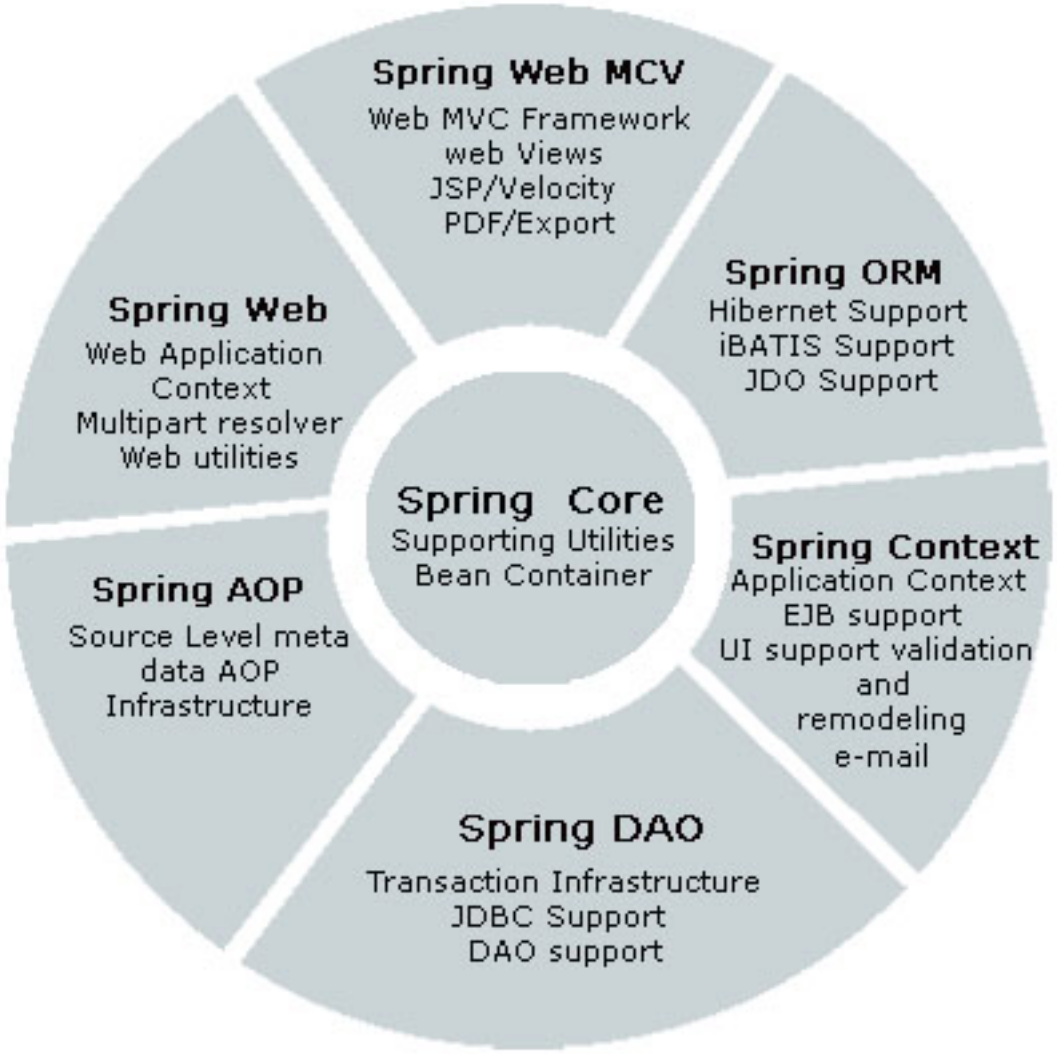
Spring provides abstraction layers for JDBC (which simplifies the error handling strategy to a great extent) and transaction management (allows to add the pluggable transaction managers and easily demarcate the transactions without dealing with low-level issues). Spring's transaction support is not tied to JEE environments and it can also be used in container-less environments.

Spring MVC separates the roles of the controller, the model object, the dispatcher, and the handler object, which makes them easier to customize. This MVC framework is designed around a DispatcherServlet that dis-

patches requests to handlers, with configurable handler mappings, view resolution like features. The default handler is a very simple Controller interface with method, ModelAndView handleRequest(request, response). Spring provides a controller hierarchy that can be sub-classed. For Example if some application needs to process user entry forms you can inherit AbstractFormController or if Application needs to process a multi-page entry into a single form you can inherit AbstractWizardFormController class of framework.

Spring Architecture

Spring's layered architecture provides a lot of flexibility and functionality that is build on the lower levels. Spring is a complete JEE framework and has a well-organized architecture consisting of seven modules:



1. Spring Core package

The Core package is the most import component of the Spring Framework. It's primary component is the 'BeanFactory', an implementation of the Factory pattern. It separates the dependencies issues like initialization, creation and access of the objects from the actual program logic. The BeanFactory applies the IOC pattern to separate an application's configuration and dependency specification from the actual application code.

2. Spring Context package

This package uses observer design pattern and provides an ability to obtain resources (for application objects through a consistent API) and it adds support for message sources. The spring context is a configuration file that provides context information to the spring framework. The spring context supplies enterprise services such as JNDI access, EJB integration, e-mail, internalization, validation, and scheduling functionality

3. Spring DAO package

The DAO (Data Access Object) package is primarily used for standardizing the data access work using the technologies like JDBC, Hibernate or JDO. The Spring's JDBC and DAO abstraction layer offers a meaningful

exception hierarchy for managing the database connection, exception handling and error messages thrown by different database vendors. The exception hierarchy simplifies error handling and greatly reduces the amount of code that we need to write, such as opening and closing connections.

4. Spring ORM package

The ORM package is related to the database access support. It provides integration layers for popular object-relational mapping APIs, including JDO, OJB, Hibernate and iBatis SQL Maps.

5. Spring AOP package

One of the key components of Spring is the AOP package. AOP is used in Spring:

I. To provide declarative enterprise services, especially as a replacement for EJB declarative services. The most important such service is declarative transaction management, which is build on Spring's transaction abstraction layer.

II. To allow users to implement custom aspects, complementing their use of OOP with AOP. The Spring AOP module also introduces metadata programming to Spring. This is used to add annotation to the source code that instructs Spring on where and how to apply aspects.

6. Spring Web package

The Spring Web module is a part of Springs web application development stack, which includes Spring's MVC. The Web context module is build at the top of application context module which provides contexts for Web-based applications. As a result, the Spring framework supports integration with Jakarta Struts, JSF and web works. The Web module also eases the tasks of handling multipart requests and binding request parameters to domain objects.

7. Spring Web MVC package

The MVC framework is a full-featured MVC implementation for building Web applications. The MVC framework is highly configurable via strategy interfaces and accommodates numerous view technologies including JSP, Velocity, Tiles and the generation of PDF and Excel Files.

The following figure represents the Spring Framework Architecture

Main Pillars of Spring :

I. An AOP Framework
II. An Inversion of Control Container
III. A Service Abstraction Layer

These together enable to write powerful, scalable applications using the Plain Old Java Objects (POJOs).

I. AOP: Aspect Oriented programming

AOP decomposes a system into concerns, instead of objects. It deals with "aspects" that cross-cut across the code which could be difficult or impossible to modularize with OOP.

Aspect Oriented programming is a new programming technique that promotes separation of concerns within the systems. The core construct of AOP is the aspect, which encapsulates behaviors affecting multiple classes into reusable modules. Systems are composed of several components each responsible for a specific piece of functionality. Irrespective of the core functionality of a program, the system services like logging, transaction management, security etc., must be included in the program. These system services are commonly referred to as 'cross-cutting concerns' as they tend to cut across multiple components in a system.

AOP makes it possible to modularize and separate these services and then apply them declaratively to the components and we can focus on our own specific concerns. In spring, aspects are wired into objects in the spring XML file in the same way as JavaBean. This process is known as 'Weaving'.

In a typical object-oriented development approach you might implement logging functionality by putting logger statements in all your methods and Java classes. In an AOP approach you would instead modularize the logging services and apply them declaratively to the components that required logging. The advantage, of course, is that the Java class doesn't need to know about the existence of the logging service or concern itself with any related code. As a result, application code written using Spring AOP is loosely coupled.

II. IoC: Inversion Of Control

"The IoC pattern enables better software design that facilitates reuse, loose coupling, and easy testing of software components."

The basic concept of the Inversion of Control pattern (also known as dependency injection) is that you do not create your objects but describe how they should be

created. You avoid connecting components and services together in your code. Instead a configuration file is used to describe the services needed by a component . In the Spring framework, the IOC container is responsible for looking it all up.

IoC: salient features

1. Eliminates lookup code from within your application.
2. Allows pluggablity and hot swapping.
3. Promotes good OO design.
4. Enables the reuse of existing code.
5. Makes your application extremely testable.

In a typical IOC scenario, the container creates all the objects, wires them together by setting the necessary properties. It also determines when methods will be invoked. The three implementation pattern types for IOC are listed in the table below.

Type 1: Services need to implement a dedicated interface through which they are provided with an object from which they can look up dependencies (for example, additional needed services).

Type 2: Dependencies are assigned through JavaBeans properties (for example, setter methods).

Type 3: Dependencies are provided as constructor parameters and are not exposed as JavaBeans properties.

The Spring framework uses the Type 2 and Type 3 implementations for its IOC container.

IOC is a broad concept, its two main types are:

1. Dependency Lookup: Type 1 IoC The container provides callbacks to components and a lookup context. The managed objects are responsible for their other lookups. This is the EJB Approach. The Inversion of Control is limited to the Container involved callback methods that the code can use to obtain resources. Here JNDI is needed to look up other EJBs and resources. Because of this reason EJB is not branded as 'IOC framework'. There are some problems in this implementation. The class needs a application server environment as it is dependent on JNDI and it is hard to test as we need to provide a dummy JNDI contest for testing purpose.

2. Dependency Injection: Type 2 / Type 3 IoC In this application objects are not responsible to looking up resources they depend on. Instead IoC container configures the object externalizing resource lookup from application code into the container. That is, dependencies are injected into objects. Thus lookups are completely removed from application objects and it can be used outside the container also.

Here, the objects can be populated via Setter Injection (Java-Beans properties) or Constructor Injection (constructor arguments). Each method has its own advantage and disadvantage.

Setter Injection: Normally in all the java beans, we use setter and getter method to set and get the value of property as follows

```
public class nameBean {
String name;
public void setName(String a) {
name = a;
}
public String getName() {
return name;
}
}
```

We create an instance of the bean 'nameBean' (say bean1) and set property as bean1.setName("amit"); Here in setter injection, we set the property 'name' by using the <property> subelement of <bean> tag in spring configuration file as shown below.

```
<bean id="bean1" class="nameBean">
<property name="name" >
<value>amit</value>
</property>
</bean>
```

The subelement <value> sets the 'name' property by calling the set method as setName("amit"); This process is called setter injection.

```
constructor injection : For constructor injection, we use
constructor with parameters as shown below,
public class nameBean {
String name;
public nameBean (String a) {
name = a;
}
}
```

We will set the property 'name' while creating an instance of the bean 'nameBean' as nameBean bean1 = new nameBean("amit");
Here we use the <constructor-arg> element to set the the property by constructor injection as

```
<bean id="bean1" class="nameBean ">
<constructor-arg>
<value>Bean Value</value>
</constructor-arg>
</bean>
```

To set properties that reference other beans <ref>, subelement of <property> is used as shown below,

```
<bean id=”bean1" class=”bean11">
<property name=”game”>
<ref bean=”bean2"/>
</property>
</bean>
<bean id=”bean2" class=”bean22" />
```

The IOC container

The mainstream JEE involves heavyweight containers to develop applications. So exploring alternatives and coming up with creative ideas have evolved a lot of open source java communities. In the Java community there's been a rush of lightweight containers that help to assemble components from different projects into a cohesive application. Several open source projects, including Spring, PicoContainer, and HiveMind use the IoC pattern to develop lightweight JEE Containers. The container manages the life cycle and configuration of application objects.

Spring should not, however, be confused with traditionally heavyweight EJB containers, which are often large. The Spring actually comes with two distinct containers:

Bean Factories-defined by “org.springframework. beans. factory.BeanFactory” are the simplest containers, providing support for dependency injection.

Application contexts - defined by “org.springframework. context.Application Context” provides application framework services.

BEAN FACTORY:

Bean factory is an implementation of the factory design pattern. Its function is to create and dispense beans. As the bean factory knows about many objects within an application, it is able to create association between collaborating objects as they are instantiated. This removes the burden of configuration from the bean and the client.

BEAN FACTORY supports two object modes.

Singleton mode provides a shared instance of the object with a particular name, which can be retrieved on lookup. Singleton is the default and most often used object mode. It is ideal for stateless service objects.

Prototype mode ensures that each retrieval will result in the creation of an independent object. Prototype mode would be best used in a case where each user needed to have his or her own object.

The bean factory concept is the foundation of Spring as an IOC container. IOC moves the responsibility for making things happen into the framework and away from application code. The Spring framework uses JavaBean properties and configuration data to figure out which dependencies must be set.

There are several implementation of BeanFactory like **“org.springframework.beans.factory.xml.XmlBeanFactory”** which loads its beans based on the definition contained in an XML file.

APPLICATION CONTEXT:

The Application Context is spring's more advanced container. Like ‘BeanFactory’ it can load bean definitions, wire beans together and dispense beans upon request. Additonally, It also provides:

1. A means for resolving text messages, including support for internationalization ie. i18n messages
2. A generic way to load file resources.
3. Events to notify beans that are registered as listeners.

Because of additional functionality, ‘Application Context’ is preferred over a BeanFactory. BeanFactory is used for simple applications and when the resource is scarce like mobile devices.

III Service Abstraction Layers

Spring provides consistent integration with various standard and 3rd party APIs through its various Service abstraction layers. Few of them are:

1. Transaction Management abstraction for JTA, JDBC, others
2. Data Access abstraction for JDBC, Hibernate, JDO, TopLink, iBatis
3. Abstraction for Emailing
4. Remoting abstraction layers for EJB, Web Services, RMI, Hessian/Burlap

Benefits of the Service Abstraction Layers in spring framework:

1. There is no implicit contract with JNDI, etc.
2. It separates the user from the underlying APIs.
3. It enhances the reusability to a great extent.
4. Spring abstractions always consist of inter faces.
5. Testing is kept simpler than ever before.
6. For data access, Spring uses a generic transaction infrastructure and DAO exception hierarchy that is common across all sup ported platforms.

Design Pattern

"Pattern" word suggests a series of events occurring in a definite order.

Many a times, you get an easy way to tackle a recurring problem (which has been faced earlier, by people frequently). This solving technique gradually becomes a pattern to tackle that particular problem.

In a broad spectrum, it can be said that a pattern describes a proven solution to a recurring problem. Pattern gives emphasis on the context and the forces causing the problem, and it invents the consequences and impacts of the solution to the problem (where context is the environment, surroundings, situation, or interrelated conditions within which the problem exists).

What is a design pattern?

The design patterns are language-independent strategies used to solve common object-oriented design issues (problems). When you design a problem, you should know some common solutions. It makes communication between team players easier, effective and result oriented.

A good software-developer should know at least some popular solutions to the coding problems. These solutions prove efficient and effective to the new breed of developers as they are gradually developed by the experienced developers. Such solutions are described as so-called design patterns.

Learning design patterns speeds up a developer experience in accumulating OOA/OOD concepts. Once a developer grasp them, he would be benefited from them to be a master of designing and developing. Furthermore, he will use these terms to communicate with his fellows or assessors more effectively. Design pattern is a indispensable part of Java and J2ee technologies.

Learning the design patterns is a multiple step process:

1. Individual acceptance

2. Group/Mass recognition
3. Internalization

Why Use Patterns?

· **They have been proven.** Patterns reflect the experience, knowledge and engineering insights of developers who have successfully used these patterns in their own work.
· **They are reusable.** Patterns provide a ready-made solution that can be adapted to different problems without the need to reinvent the wheel on a project-by-project basis.
· **They are expressive.** Patterns provide a common vocabulary of solutions that can express large solutions succinctly. They provide a design vocabulary and reusable artifacts. Once described, any level engineer can use the pattern.
Many programmers don't know design patterns even with many years of experience. However as an Object-Oriented programmer, they have to know them well, especially for a new Java programmer. Actually, when you solve a coding problem, you use a design pattern rather you may not know a popular name to describe it. It is always beneficial to learn from the experiences of the past to solve the coding problems and use them in your projects are a best way.

To keep pace with the new developments, it is imperative that you are not wasting time in maintaining designs that have poor architecture or code that was poorly written. Just use the Design Patterns as they involved a highly experienced engineering in recognizing, collaborating and refining a design pattern. Design pattern is just a recommendation and it is up to the engineer or architect, to apply a pattern appropriately to his scenario.

A design pattern does not apply that only one solution exists for a problem neither it necessarily imposes as the best solution in all cases. Patterns merely provide a best-practice approach to a particular problem, learned through the countless experiences of the programming community. A pattern often has several variations. Each programmer must determine if and when to apply a particular pattern. Often, the programming environment in use influences the pattern choice. Not all programming environments and languages support all design patterns. What may be easy to create in one environment or language may be extremely difficult in another

Relationship among Design patterns?

Patterns have relationships and work together to form a weave. That is why, it is found that more familiar you are with the different patterns, better you determine their interactions. Generally, to build a system, you may need many patterns to fit together. Patterns can also be used to design frameworks. Different designer may use different patterns to solve the same problem. Usually:

I Some patterns naturally fit together
II One pattern may lead to another
III Some patterns are similar and alternative

IV Patterns are discoverable and documentable
V Patterns are not methods or framework
VI Patterns give you hint to solve a problem effectively

Categorizing Design Patterns:

Patterns focus on different types of problems. Related patterns are grouped together and assigned a type. This helps programmers to identify similar types and further it simplifies the process of comparing similar patterns to find the appropriate one(s) to utilize.

I Classification of Design Patterns in general spectrum:

1 Creational Patterns
2 Structural Patterns
3 Behavioral Patterns

II JEE-specific Design Patterns

1 Presentation tier patterns
2 Business tier patterns
3 Data Access tier patterns

1. Creational Patterns

This design category is all about the class instantiation. **Creational** pattern uses class-creation patterns and object-creational patterns. The class-creation pattern uses inheritance effectively in the instantiation process while object-creation pattern uses delegation to get the job done. All the creational patterns define the best possible way to instantiate an object. They describes the best way to create the object instances. Many times, nature of the created object changes according to the nature of the program. In such scenarios, we use patterns to give this a more general and flexible approach, for example a object instance can be created using a new operator, which is a hard coding methodology (hard coding should be the last option to go with).
There are five types of Creational Patterns.

I Factory Pattern
II Abstract Factory Pattern
III Builder Pattern
IV Prototype Pattern
V Singleton Pattern

2. Structural Patterns

Structural Patterns describe how objects and classes can be combined to form larger structures. The difference between class patterns and object patterns is that class patterns describe abstraction using inheritance and describe how it can be used to provide more useful program interface. Object patterns, on other hand, describe how objects can be associated and composed to form larger, more complex structures. There are seven structural patterns described. They are as follows:

I Adapter Pattern
II Bridge Pattern
III Composite Pattern
IV Decorator Pattern
V Facade Pattern
VI Flyweight Pattern
VII Proxy Pattern

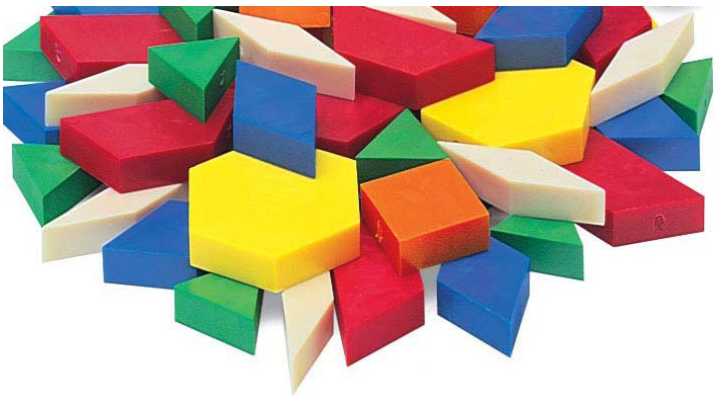
3. Behavioral Patterns

Behavioral patterns are those patterns which are specifically concerned with communication (interaction) between the objects. The interactions between the objects should be such that they are talking to each other and still are loosely coupled. The loose coupling is the key to n-tier architectures. In this, the implementations and the client should be loosely coupled in order to avoid hard-coding and dependencies.
The behavioral patterns are:

- Chain of Responsibility Pattern
- Command Pattern
- Interpreter Pattern
- Iterator Pattern
- Mediator Pattern
- Momento Pattern
- Observer Pattern
- State Pattern
- Strategy Pattern
- Template Pattern
- Visitor Pattern

II JEE-specific Design patterns:

The term JEE is tossed around a lot because it covers the



widest range of applications development in the enterprise and distributed environments. Infact, the JEE modules and environment is still growing with a rapid pace. So, it is important to take advantage of the most efficient and effective strategies while re-factoring the existing projects and developing newer ones. These efficient and effective strategies in JEE scenario are known as the JEE-specific **Design patterns.**

JEE-specific design patterns identify the minimal set of known problems that application architecture should solve. These patterns are based on the experiences of the JEE community (involved with JEE development) to solve the problems.

Classification of JEE Design Patterns :

1. Business Tier Patterns

These business tier patterns tackle problems occurring in an application resulting from the presentation tier accessing distributed business services, network performances degradation due to multiple calls between client and server, memory impact due to retrieval of a large list of data, and so on. The patterns demonstrated here focus on and solve design problems occurring in the middle tier of a J2EE application.

- Session Facade Design Pattern
- Service Locator Design Pattern
- Value List Handler Pattern

2. Presentation Tier Patterns

The presentation tier patterns deal with the common problems occurring in the presentation layer such as - view management and navigation, processing of dynamic business data, efficiently accessing the read-only data, and so on. The patterns under this category focus on and solve design problems occurring in the presentation tier of a J2EE application for example:
Fast Lane Reader Pattern

3. Data Access Tier Patterns:

These data access tier patterns tackles best practices for an application accessing the database or the underlying persistence layer from the business tier. The patterns demonstrated here focus on and solve design problems occurring in the data tier of a J2EE application.
Data Access Object Design Pattern

A word of CAUTION:

Patterns are not a magical stuff for a problem identified and a pattern applied does not mean that you will have a perfect solution, for that scenario. Patterns are just a way to introduce clarity into your system architecture. Patterns allow the possibility of building a better system meeting the intended business requirements, performing well, is maintainable, and is delivered on time. Above all it keeps us engineering in business…..

Few of the familiar Design Patterns in Java

1 Interface

Objects define their interaction with the outside world through the methods that they expose. These methods form the object's interface with the outside world, for example, the buttons on the front of your television set, are the interface between you and the encapsulated electrical television components.

An interface pattern encapsulates a coherent set of services and attributes, without explicitly binding the functionality to a particular object or a code. In Java an interface is a programming construct, similar to an abstract class, that allows you to specify zero or more method signatures without providing the implementations of those methods.
Implementing an interface allows a class to become more formal about the behavior it promises to provide. Interfaces form a contract between the class and the outside world, and this contract is enforced at build time by the compiler. If your class claims to implement an interface, all methods defined by that interface must appear in its source code before the class will successfully compile.

2 Abstract

Abstract pattern can be used to create an abstract class

which misses definitions for one or more methods. Thus it restricts the creation of an object of that class. You must first create a subclass and provide definitions for the abstract methods. Unlike interfaces, abstract classes may implement some of the methods. Though you can't instantiate an abstract class, you can invoke its static methods. Abstract classes form an interesting and useful middle ground between interfaces and classes.

3 Delegation

The Delegation and Interface patterns are often used together. Delegation is used to avoid Inheritance. Delegation is a way of extending and reusing a class by writing another class with additional functionality that uses instances of the original class to provide the original functionality.

4 Marker Interface

The Marker Interface pattern uses interfaces declaring no methods or variables to indicate semantic attributes of a class. It works particularly well with utility classes that determine something about objects without assuming that they are an instance of any particular class.

5 Immutable

The Immutable pattern reduces the overhead of concurrent access to an object and increases the robustness of objects that share references to the same object. It accomplishes this by not allowing an object's state information to change after it is constructed. The Immutable pattern also avoids the need to synchronize multiple threads of execution that share an object.

6 Single Threaded Execution

The Single Threaded Execution pattern is the pattern most frequently used to coordinate access by multiple threads to a shared object. The Immutable object pattern can be used to avoid the need for the Single Threaded Execution pattern or any other kind of access coordination.

7 Proxy

The Proxy pattern forces method calls to an object to occur indirectly through a proxy object that acts as a surrogate for the other object, delegating method calls to that object. Classes for proxy objects are declared in a way that usually eliminates client object's awareness that they are dealing with a proxy. Proxy is a very general pattern that occurs in many other patterns, but never by itself in its pure form.

Tomcat Web Server

How to install and Configure the Tomcat Server

Tomcat is an open source web server developed by Apache Group. Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies. The Java Servlet and JavaServer Pages specifications are developed by Sun under the Java Community Process. Web Servers like Apache Tomcat support only web components while an application server supports web components as well as business components (BEAs Weblogic, is one of the popular application server).

To develop a web application with jsp/servlet install any web server like JRun, Tomcat etc to run your application.

Here we are illustrating the guidelines to install and configure the Tomcat 6.0, as a stand-alone web server. In this tutorial we are covering the Tomcat version 6.0.10. Sometime Apache Tomcat is referred, as the Jakarta Tomcat so don't confuse between the two.

Here we are illustrating the installation process only for Microsoft Windows. Installing Tomcat on Windows can be done easily using the Windows installer.

Steps involved in installation and configuration process for Tomcat 6.0.10 are illustrated below:

Step 1: **Installation of JDK:** Installation of Tomcat 6.0.10 requires the JVM compatible with Java 1.5 (Java 5) and Java 1.6 (Java 6) so don't forget to install the needed JDK on your system (if not installed) and to set the classpath.

Step 2: Setting the class path variable for JDK: **There are two methods to set the classpath.**

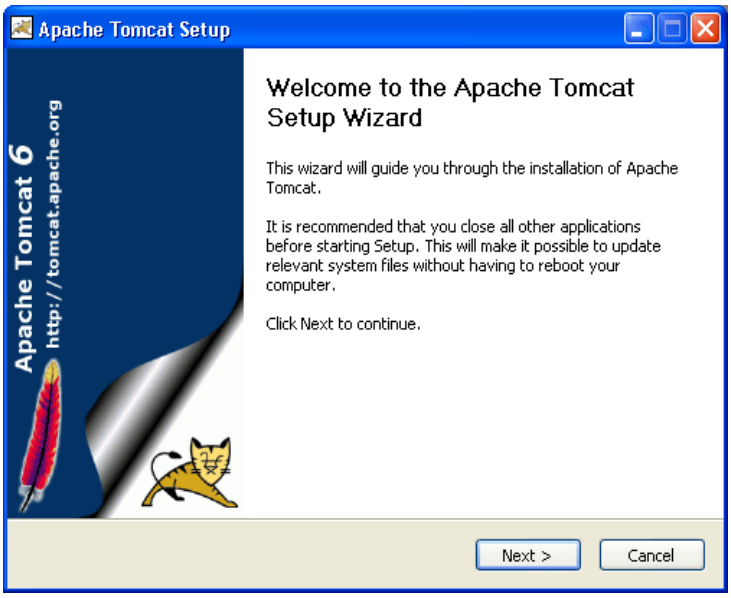
1. Set the class path using the following command.

   set PATH="C:\Program Files\Java\jdk1.5.0_08\bin";%PATH%

2. The other way of setting the class path variable is:
   First right click on the My Computer->properties->advance->Environment Variables->path.
   Set bin directory path of JDK in the path variable.

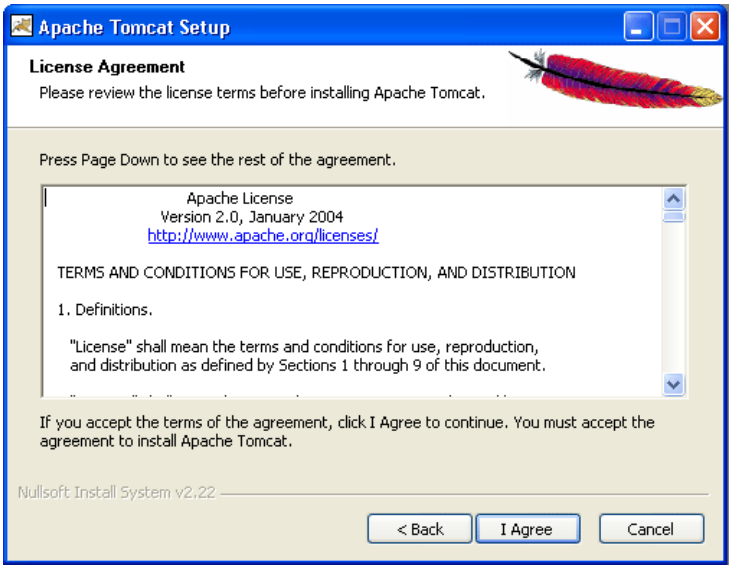
Step 3: Now it's time to shift on to the installation process of Tomcat 6.0.10. It takes various steps for installing and configuring the Tomcat 6.0.

For Windows, Tomcat comes in two forms: .zip file and the Windows installer (.exe file). Here we are exploring the installation process by using the .exe file. The directory C:\apache-tomcat-6.0.10 is the common installation directory as it is pre-specified C:\ as the top-level directory. First unpack the zipped file and simply execute the .exe file.

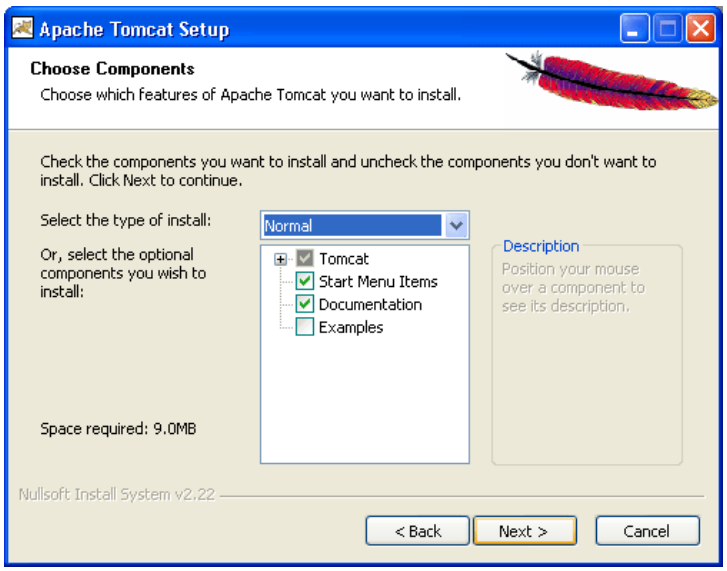


The above shown screen shot is the first one shown in the installation process.

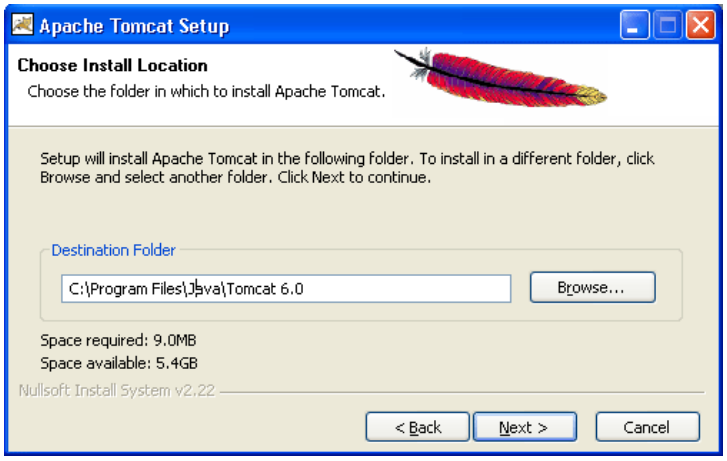
Just click on the Next button to proceed the installation process.



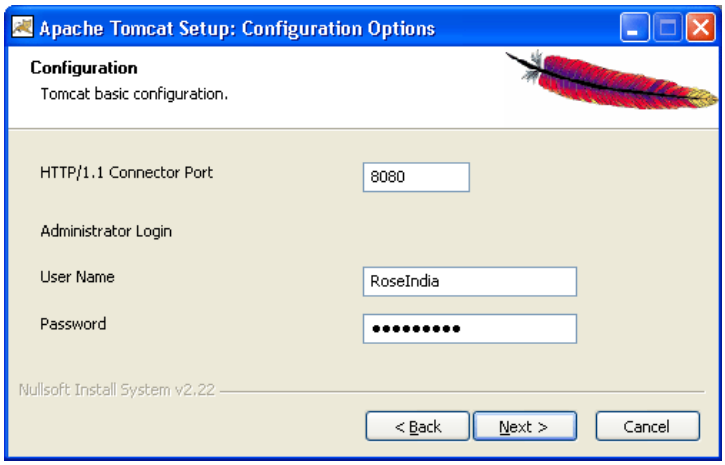
Click **"I Agree"** button to continue the installation process.



Click next to go with the default components chosen.



Choose the location for the Tomcat files as per your convenience. You can also choose the default location.



Now choose the port number on which you want to run the tomcat server. Tomcat uses the port number 8080 as its default value. But Most of the people change the port number to 80 because in this case the user is not required to specify the port number at request time.

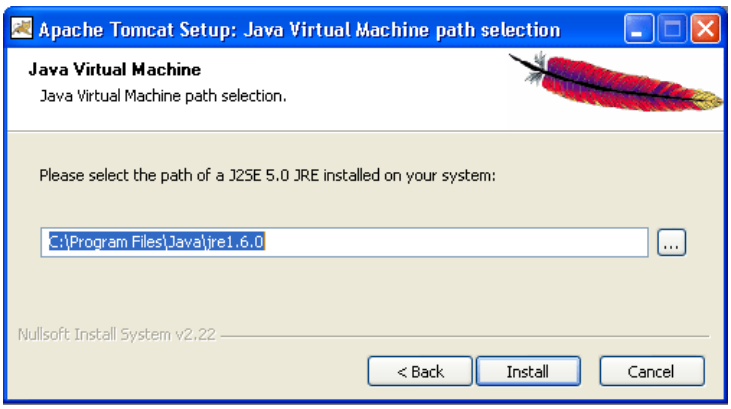
But we are using here the default port number as 8080. Choose the user name and password as per your convenience. We can change the port number even the installation process is over. For that, go to the specified location as " Tomcat 6.0 \conf \server.xml ". Within the server.xml file choose "Connector" tag and change the port number.

e.g While using the port number 8080, give the following request in the address bar as:

Default Port: http//localhost:8080/index.jsp

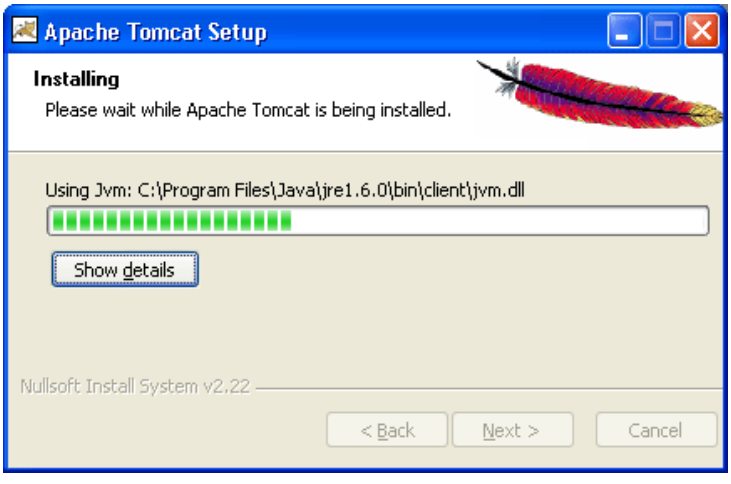
In case of port number number 80 just type the string illustrated below in the address bar:

New Port: http://localhost/index.jsp



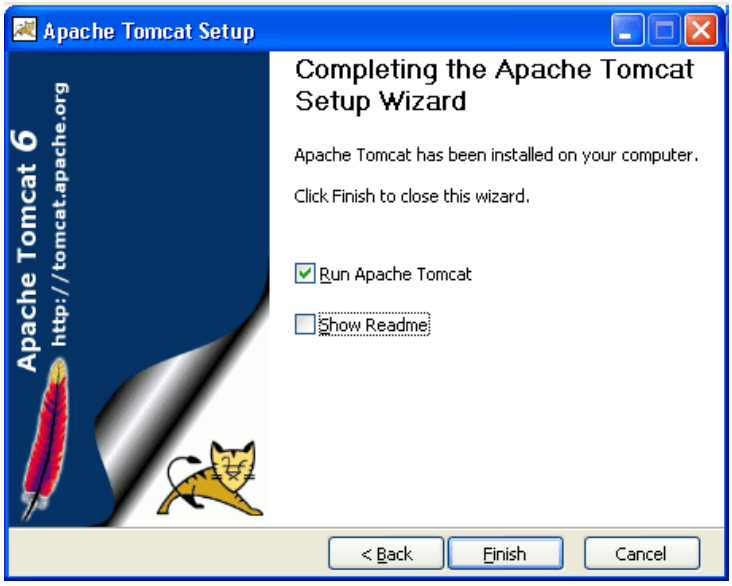
Note that we do no need to specify any port number in the URL.

Now click on the Next button to proceed the installation process.

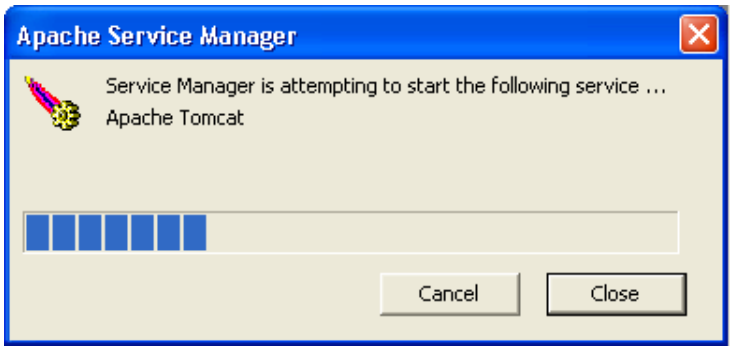


The installation process shows the above screen as the next window. This window asks for the location of the installed Java Virtual Machine. Browse the location of the

JRE folder and click on the Install button. This will install the Apache tomcat at the specified location.



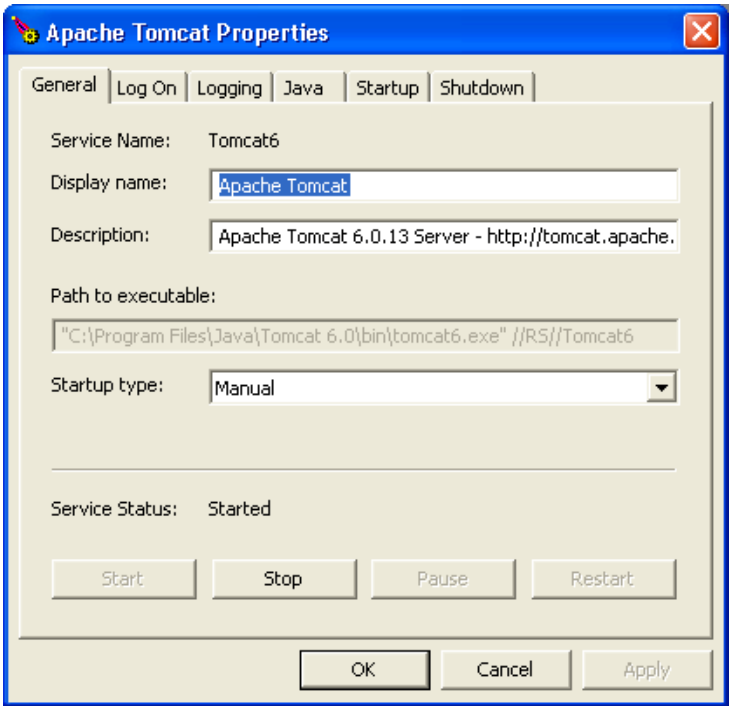
To get the information about installer click on the **"Show details"** button.



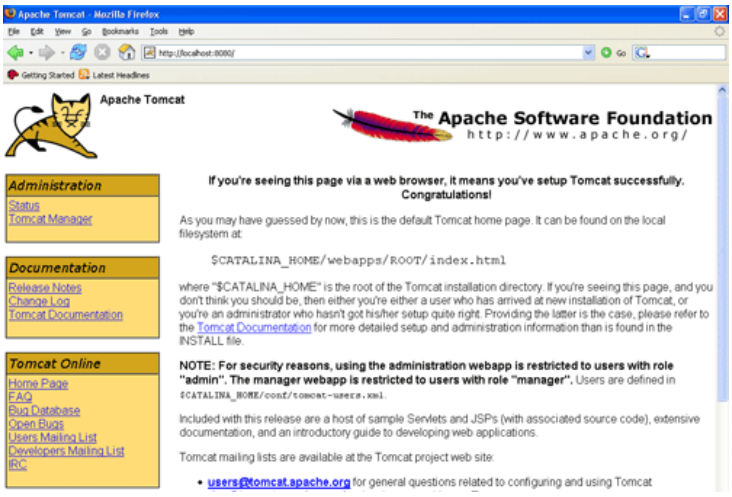
After completion of installation process it will display the window like the above one.

On clicking at Finish button, a window like the above one will display a message printed on the window given below.

After successfully installing, a shortcut icon to start the tomcat server appears in the icon tray of the task bar as shown above. Double clicking the icon, displays the window of Apache Manager for Tomcat. It will show the "Startup type" as manual since we have changed the destination folder for tomcat during the installation



process. Now we can configure the other options like "Display name" and "Description" .We can also start, stop



and restart the service from here.

If installation process completes successfully then a window as shown below will appear.

Now, set the environment variable for tomcat:

Step 4: Setting the JAVA_HOME Variable: Purpose of setting the environment variable JAVA_HOME is to specify the location of the java run time environment needed to support the Tomcat else Tomcat server does not run. This variable contains the path of JDK installation directory. Note that it should not contain the path up to bin folder.

Set JAVA_HOME=C: \Program Files\Java\jdk1.5.0_08

Here, we have taken the URI path according to our installation convention

For Windows XP, Go through the following steps:

Start menu->Control Panel->System->Advanced tab->Environment Variables->New->set the Variable Name as JAVA_HOME and Variable Value as C:\Program Files\Java\jdk1.6.0 and then click on all the three ok buttons one by one. It will set the JDK path.

For Windows 2000 and NT, follow these steps:

Start->Settings->Control Panel->System->Environment Variable->New->set the Variable Name as JAVA_HOME and Variable Value as C:\Program Files\Java\jdk1.6.0 and then click on all the three ok button one by one. It will set the JDK path

Now, **Start the Tomcat Server:** Start the tomcat server from the bin folder of Tomcat 6.0 directory by double clicking the "tomcat6.exe " file. You can also create a shortcut of this .exe file at your desktop.

Stop the Tomcat Server: Stop the server by pressing the "**Ctrl + c**" keys.

Web Services

A web service is a collection of protocols and standards used for exchanging data between applications over the web. Web services enable applications written with different programming languages supported over varying platforms (different hardware, software, database, or network platforms) to exchange data, very conveniently.

Web services are a new breed of web technologies that offer a platform and language-neutral approach. They are so cool that a web service written in Java and running on BEA WebLogic server can access a web service written in C and running on a Microsoft IIS. All without a concern about, how each web service is actually implemented.
Web Services have revolutionized the way; the Business to Business and Business to Consumer services are provided because they provide multiple functions, ranging from a simple web request to a complicated business process.

A web service makes it easily available over the web and uses a standardized XML messaging system as its core mechanism. XML is used to encode all communications into a web service. For example, a client invokes a web service by sending an XML message, then waits for a corresponding XML response. Because all communication is in XML, web services are not tied to any one operating system or programming language. So Java can talk with Perl and windows applications can talk with Unix applications

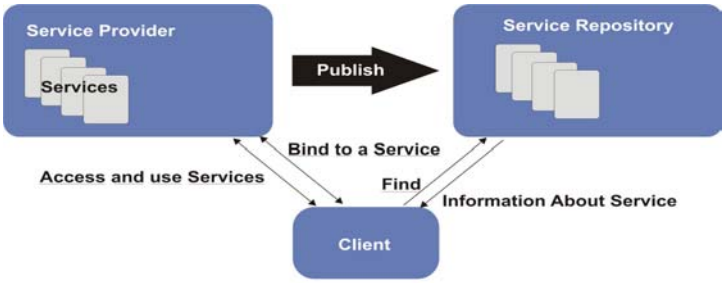
Web Services: Distributed Computing environment

Web Services have gradually evolved from the basic Remote Procedure Calls. But unlike RPCs they use XML as a core ingredient for communication. Web Service concerns about three specific things:

- A mechanism to find and register a service.
- A transport mechanism to access a service.
- A mechanism for two parties to communicate and define the input-output parameters for a service.

What are Web Services

These properties, when coupled together, provide a new form of Distributed Computing Environment. It requires standard definition mechanisms, lookup services, and transport definitions using protocols like SOAP. These all are provided without concern for the underlying implementation mechanism.



Distributed Computing Environment: web scenario

Service provider: First and foremost a service provider is required to host any number of exposed Web Services.

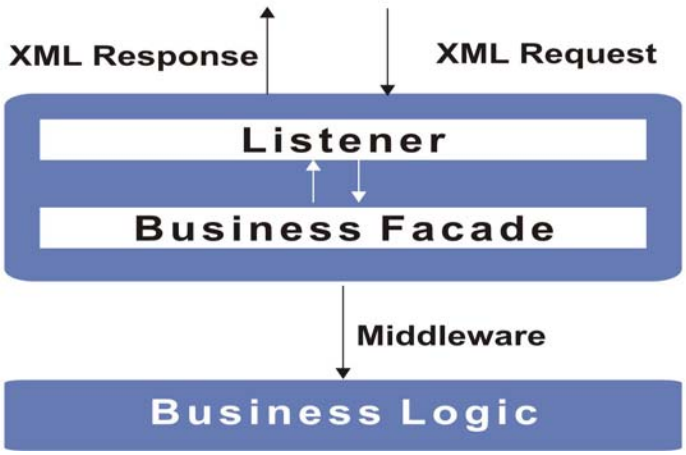
Service repository: Secondly, a centralized service repository is required for publishing information that clients can use to find information about published Web Services.
And finally, various mechanisms are required to locate and access the services.

Let's look at the figure in general and understand what's going on.

- Firstly, a service provider publishes a service to an external repository. Once such a service has been exported to a registry, it can then be used by a client.
- Client look- ups a service from a repository and receives information about a service. Information, such as the format of the procedure calls and the address of a service provider, would normally be provided, amongst other details.
- At last, the client is bind to the underlying services and then accesses that service (functionality the service provides).

Web Services: Core specifications

XML: a key player: Web Services at its core uses a fundamental set of functionality based on XML (it is a wonderful technology, and it has really found a home in Web Services). XML provides a user-defined meta language that represents data and the sets of validating rules. XML has become an excellent vehicle for packing data over the distributed network such that both ends of the pipe easily exchange and understand the data.



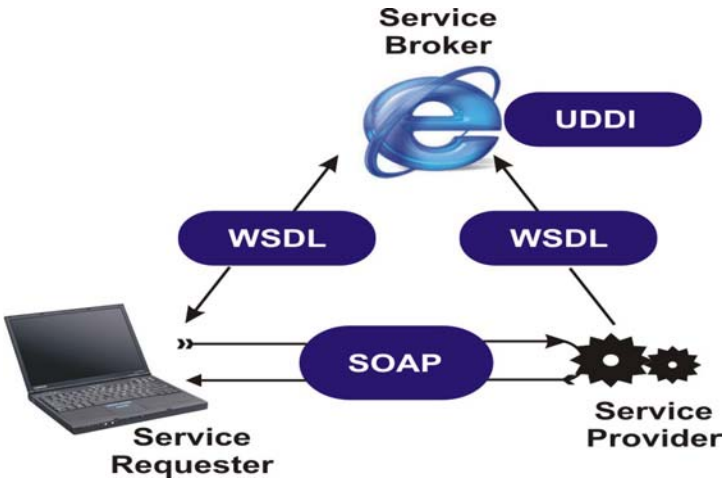
XML provides a meta language in which you can write specialized languages to express complex interactions between clients and services. Behind the facade of a web server, the XML message gets converted to a middleware request and the results are converted back to XML.

The specifications that define **Web services** are intentionally modular, and as a result there is not a single document that contains them all. Additionally, there is neither a single, nor a stable set of specifications. There are a few "core" specifications that are supplemented by different groups (as the circumstances and choice of technology dictate) like XML , SOAP have come from **W3C** while WSDL and UDDI are provided by **OASIS.**

The web services platform can be thought of as:

HTTP + **SOAP** (XML-based protocol) + **WSDL** (XML-based format) **+ UDDI.** Where

- HTTP is a ubiquitous protocol supported everywhere on the Internet.
- SOAP is needed for remote procedure calls. It is an XML-based format, that provide" bindings" with underlying network protocols like HTTP, HTTPs, SMTP, XMPP etc. A SOAP call is packaged as the body of an HTTP request.
- WSDL is an expression of service characteristics that allows service interfaces to be described, along with the details of their bindings to specific protocols. WSDL is used to describe what a web service can do, where it resides, and how to invoke it.
- UDDI acts as the trader, directory service etc. It is required for publishing and discovering metadata about Web services, that enables applications to find Web services, either at design time or runtime.



Web Services model

Examples of few web services

Google's Web Service - access the Google search engine

- Amazon's Web Service - access Amazon's product information
- SalCentral - WSDL / SOAP Web services search engine

Description of the Web Services Standards:

I WSDL (Web Services Description Language)

At the early stages, defining the standard behavior of the both ends of the pipe over the web using traditional RPC was highly problematic. It immensely required some fresh changes to introduce. Hence, Web Services Description Language (WSDL for short and often pronounced *wisdel*) took initiative to deal with such issues.

WSDL uses XML to define the interfaces to the existing or to the new application or anywhere in between. WSDL describes what a web service can do, where it resides, and how to invoke it. It defines the syntax, the semantics, and all the various administrative aspects of a web services remote procedure call.

WSDL defines services as a collection of network endpoints (ports). In WSDL the abstract definition of endpoints and messages is separated from their concrete network deployment and data format bindings. This allows reuse of abstract definitions of messages (which are abstract descriptions of the data being exchanged) and port types (which are abstract collections of operations). The concrete protocol and data format specifications for a particular port type constitute a reusable binding. A port is defined by associating a network address with a reusable binding and a collection of ports define a service.

A WSDL document uses the following elements to define network services:

- **Types** - a container for data type definitions using some type system (such as XSD).
- **Message** - an abstract, typed definition of the data being communicated.
- **Operation** - an abstract description of an action supported by the service.

- **Port Type** - an abstract set of operations supported by one or more endpoints.
- **Binding** - a concrete protocol and data format specification for a particular port type.

- **Port** - a single endpoint defined as a combination of a binding and a network address.

- **Service** - a collection of related endpoints.

In simple words, WSDL is a template which shows how

services should be described and bound by clients.

II. Universal Disc overy, Description and Integration (UDDI)

In order to use a service, a client must first locate that service and retrieve information to use the service.

The Universal Discover, Description and Integration or UDDI specification, defines a number of lookup services that allow clients to look up and retrieve the required information to access a Web Service. UDDI provides a mechanism for clients to dynamically find other web services over the web. Using a UDDI interface, businesses can dynamically connect to services provided by external business partners.

UDDI actually provides three specific services:

- **Traditional White** Pages for looking up a Web Service by name.
- **Traditional Yellow** Pages for looking up a Web Service by topic.
- **Green Pages** for more generic searches based on the characteristics of a Web Service.

Vendors who provide UDDI services typically operate a service known as a UDDI Business Registry, or UBR, which can be accessed to both publish and request information about a given Web Service.

III. Connecting It All Together With the Simple Object Access Protocol (SOAP)

We have analyzed few things like defining, describing and publishing a web service. Now we see the mechanism to access a web service once we've found it.

Web Services become accessible through the SOAP protocols. SOAP defines a way to perform remote procedure calls (RPCs) using HTTP as the underlying communication protocol. Typically, a SOAP call is packed as a body of an HTTP request. SOAP is simple, easy to implement, and well supported in the industry. Even a UDDI registry is accessed through well-defined SOAP calls.

SOAP supports both synchronous and asynchronous call semantics i.e. standard RPC as well as message-based. It can also be used with a variety of protocols other than HTTP.

A SOAP message is an ordinary XML document containing the following elements:

- A required Envelope element that identifies the XML document as a SOAP message
- An optional Header element that contains header information
- A required Body element that contains call and response information
- An optional Fault element that provides information about errors that occurred while processing the message

AN INTERACTIVE WEB SERVICE WORKING WITH JAVA:

Google

The Google Web APIs service package contains

- GoogleSearch.wsdl - the WSDL description
- googleapi.jar - a client java API library

and provides three operations for the Google database:

- doGoogleSearch
- doGetCachedPage
- doSpellingSuggestion

Developing Web Services with Java

The few essential tools for Java Web Service development:

- Apache **AXIS** (Apache Extensible Interaction System)
  - **I** a Java-based implementation of SOAP+WSDL
  - **II** largely allows the programmer to forget these technologies
  - **III** typically used together with Tomcat
  - **IV** Highly recommended!

- Sun's **Java WSDP** (Web Services Developer Pack)
  - **I** support for SOAP, WSDL, UDDI, ...
  - **II** JAX-RPC maps SOAP/WSDL to RMI (Java Remote Method Invocations)
  - **III** Java API for XML Pack (the so-called JAX Pack)
  - **IV** the JSP Standard Tag Libraries (JSTL) API
  - **V** Java Secure Socket Extension (JSSE) API
  - **VI** Jakarta Tomcat Web container
  - **VII** the Ant build tool
  - **VIII** the UDDI-based registry server

- Open-Source Java EE 5-compliant application server : **Project GlassFish**

  - **I** Will feature a major transition from Java WSDP.
  - **II** Java WSDP Pack will no longer be developed as a discrete release vehicle for the Web services and XML technologies.
  - **III** will provide latest state of development will be supported that WSDP provides
    IBM's alpha Works's **EETK** (Emerging Technologies Toolkit)
  - **IV** Support for SOAP, WSDL, UDDI and much more...

E-Commerce : Shopping Cart

E-Commerce – role of a Shopping Cart

Life has become so easy, no need to rush to the markets. Now shopping is just a click away. Well you got it right; we are talking about e-shopping carts.

Software is used to create an online "storefront," or E-Commerce Web site. It acts as a virtual shopping cart, keeping track of the items that visitors have ordered and allowing them to add or remove items. When a visitor decides to "check out" (purchase the items online) the software sends all order information to the merchant.

Shopping Cart software acts as an online store's catalog and ordering process. By using shopping cart on the web customers can purchase multiple items with a single payment. Not only this, they can browse the entire selection, and view a consolidated list of all their items before purchasing. Basically, a shopping cart is the interface between a company's Web site and its deeper infrastructure. This allows customers to select merchandise; review their selection, make necessary modifications or additions; and finally purchase the merchandise.

Companies integrate shopping carts as these are sold as independent pieces of software into their own unique online solution. There are number of shopping carts available like LaGarde, RomanCart, AgoraCart etc. For instance the PayPal Shopping Cart is a low-cost way for you to accept credit card and bank account payments, and can be fully integrated with your website in a few easy steps.

Features of a shopping cart:

- Easy-to-use store management tools, saves time and money.
- Customizable for your business needs.
- Easy to implement - no CGI scripting necessary.
- Improve buyer experience - with customizable buttons and secure payments happy customers becomes a frequent visitor.
- Customizable HTML store home page.
- Order forms are in HTML for ease of customization.
- Upload and store static html pages for product info, store policies, etc.
- Dynamically generated product pages from store's database.

Developing a Shopping Cart Application

We are providing relevant details to develop an e-commerce application (a shopping cart) with **Struts, Hibernate** and **Spring**.

Features provided in the shopping cart

Customers Registration: Existing user can login to the shopping cart by signing in using their username and password. Moreover they can update their profile such as address, contact number, Email-id etc.

Product catalog: Product catalog displays the list of items. It helps the customers to quickly browse the details of the products available to buy.

Adding items to the cart: This feature provides a mechanism to add multiple products to a cart. Moreover options are provided to choose quantity of a particular item.

Updating Quantity: This feature enables to update the quantity of a product..

Remove Items/Product: Remove feature enables to remove a selected item from the shopping cart.

Checkout /Place order Form: After selecting the multiple items, customers can place an order by filling the **Proceed to Checkout** form. In this form customer has to fill personal as well as his credit card information.

Administrative Features

Catalog management: This feature enables to add and edit categories and subcategories. Here, the administrator can add the number of products to the Catalog Listing by providing the product details like Category ID, Category Name and Parent ID etc. Following options can be opted.

Add/Edit Categories or Subcategories

Add/Edit Product

Check orders: Here, the administrator check the orders placed by the customers with the details such as Order Id, Customer, Email, Address, Phone number, Ordered products, Order total, Order time and Quantity.

Configuration: Here, the administrator can update the following relevant information.

- Store Name
- Store URL
- Notification mail-id
- Currency Symbol & Code
- About Us content
- Shipping and Delivery content
- Changing password

Technical Features

- **Free Java source code.** Java Source code is provided free with the shopping cart application.

- **Choice of web server and a database:** This e-commerce application has been tested on Apache Tomcat 5.0 and 5.5, and is conformant with other JEE Web containers like Resin and JRun. Databases like MySQL, PostgreSQL, MS SQL Server, and Oracle are all supported out-of-the-box.

- **Integration of Struts, Spring and Hibernate: Struts is configured with** Hibernate persistence framework that handles all of the application's database access. Spring Framework is used to manage java beans for example within the application, ShoppingCartDAO object is managed through spring framework. It is also acting as transaction manager for Hibernate SessionFactory to manage the database transactions.

Application Architecture

I Shopping Cart follows a strict Model, Controller, View (MVC) architecture to organize its classes and files.

II Controller employs two packages to control the model and the view sections. One package contains classes performing different actions and other one contains classes acting as various action forms.

III The Model is organized into different packages where classes are devoted to interact with the database and to process business logic.

CONTROLLER LAYER

The Web Controller layer provides the mechanisms that control the routing of each web request through the system (the "C" in MVC). Its role is to entertain each browser request, check its validity, invoke the classes from the Business layer needed to process a request, to handle certain processings (such as recording informa-

tion in the browser session), and finally to route the processed request to the appropriate JSP in the Presentation layer.

The Controller employs two packages to control the model and the view sections i.e. roseindia.web.struts.form (contains varous Action Forms) and **roseindia.web.struts.action** (contains varous Action Classes).

I The package **roseindia.web.struts.form** contains 10 Action Forms

II The package **roseindia.web.struts.action** contains 16 Action Classes

MODEL SECTION

The Business layer is responsible for processing the application's business logic. The Business layer is responsible for managing a cart. It creates a representation of a new cart object needed to choose, remove and update the various items in the cart, and then communicate with the Data Access layer to store a representation of the cart in the database.
The Data Access layer is responsible for writing and retrieving data to and from the database, and returning the results to the Business layer

The Model is organized into six packages where classes are devoted to interact with the database and to process business logic.

1 The **roseIndia.dao** package contains the ShoppingCartDAOImpl class implementing the ShoppingCartDAO interface. ShoppingCartDAOImpl class deals with all database related concern of administrator and customer sections of the application**.**

2. Package **roseindia.dao.hibernate** contains 7 java classes and 7 hibernate mapping xml documents (*.hbm.xml). These classes and xml documents does java-Object relational mapping with the underlying database**.**

3. The **package roseindia.services** contains a single class ServiceFinder needed to retrieve various integrated services from Spring Application context.

4 The package **roseindia.web.cart** is the heart of the whole application. It contains 2 main classes which enable to add, update and remove items into a shopping cart.

5 **The package roseindia.web.common contains various helper and utility classes needed to process the business logic**.

The Presentation Layer

After the Web Controller layer routes the request to the Business layer and retrieves the results of the processing, it then forwards the request to the Presentation layer, which is responsible for generating the HTML sent to the user's browser.

Getting Shopping Cart Code

We are maintaining the code and documentation of the shopping cart at http://www.roseindia.net/shoppingcart/. Here, you can download the latest version of the shopping cart application.
Basically, a shopping cart is the interface between a company's Web site and its deeper infrastructure. This allows customers to select merchandise; review their selection, make necessary modifications or additions; and finally purchase the merchandise.

Companies integrate shopping carts as these are sold as independent pieces of software into their own unique online solution. There are number of shopping carts available like LaGarde, RomanCart, AgoraCart etc. For instance the PayPal Shopping Cart is a low-cost way for you to accept credit card and bank account payments, and can be fully integrated with your website in a few easy steps.

Features of a shopping cart:

I Easy-to-use store management tools, saves time and money.
II Customizable for your business needs.
III Easy to implement - no CGI scripting necessary.
IV Improve buyer experience - with customizable buttons and secure payments happy customers becomes a frequent visitor.
V Customizable HTML store home page.
VI Order forms are in HTML for ease of customization.
VII Upload and store static html pages for product info, store policies, etc.
VIII Dynamically generated product pages from store's database.

Developing a Shopping Cart Application

We are providing relevant details to develop an e- commerce application (a shopping cart) with **Struts, Hibernate** and **Spring**.

Features provided in the shopping cart

- **Customers Registration:** Existing user can login to the shopping cart by signing in using their username and password. Moreover they can update their profile such

as address, contact number, Email-id etc.

- **Product catalog:** Product catalog displays the list of items. It helps the customers to quickly browse the details of the products available to buy.

- **Adding items to the cart:** This feature provides a mechanism to add multiple products to a cart. Moreover options are provided to choose quantity of a particular item.

- **Updating Quantity: T**his feature enables to update the quantity of a product..

- **Remove Items/Product:** Remove feature enables to remove a selected item from the shopping cart.

- **Checkout /Place order Form:** After selecting the multiple items, customers can place an order by filling the **Proceed to Checkout** form. In this form customer has to fill personal as well as his credit card information.

Administrative Features

- **Catalog management: T**his feature enables to add and edit categories and subcategories. Here, the administrator can add the number of products to the Catalog Listing by providing the product details like Category ID, Category Name and Parent ID etc. Following options can be opted.

- Add/Edit Categories or Subcategories
- **Add/Edit Product**

- **Check orders:** Here, the administrator check the orders placed by the customers with the details such as Order Id, Customer, Email, Address, Phone number, Ordered products, Order total, Order time and Quantity.

- **Configuration:** Here, the administrator can update the following relevant information.

  i Store Name
  ii Store URL
  iii Notification mail-id
  iv Currency Symbol & Code
  v About Us content
  vi Shipping and Delivery content
  vii Changing password

Technical Features

- **Free Java source code.** Java Source code is provided free with the shopping cart application.

- **Choice of web server and a database:** This e-commerce application has been tested on Apache Tomcat 5.0 and 5.5, and is conformant with other JEE Web containers like Resin and JRun. Databases like MySQL, PostgreSQL, MS SQL Server, and Oracle are all supported out-of-the-box.

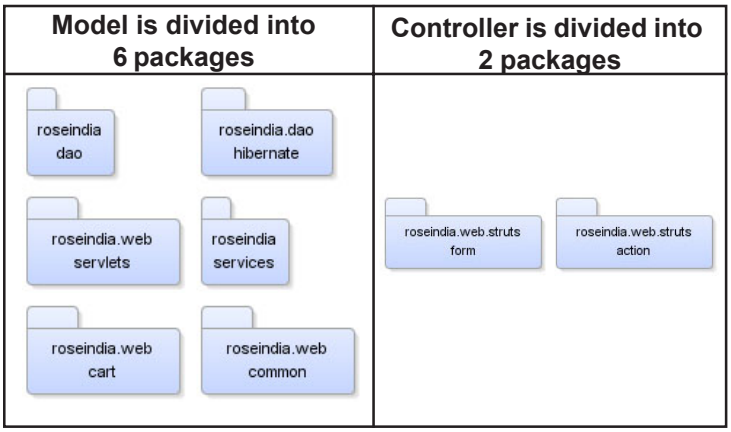
- **Integration of Struts, Spring and Hibernate: Struts is configured with** Hibernate persistence framework that handles all of the application's database access. Spring Framework is used to manage java beans for example within the application, ShoppingCartDAO object is managed through spring framework. It is also acting as transaction manager for Hibernate SessionFactory to manage the database transactions.

Application Architecture

Shopping Cart follows a strict Model, Controller, View (MVC) architecture to organize its classes and files.

Controller employs two packages to control the model and the view sections. One package contains classes performing different actions and other one contains classes acting as various action forms.

The Model is organized into different packages where classes are devoted to interact with the database and to process business logic.

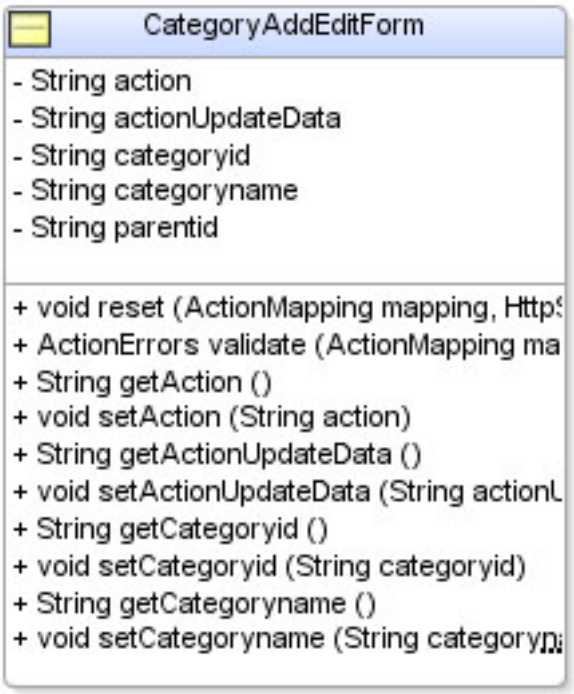


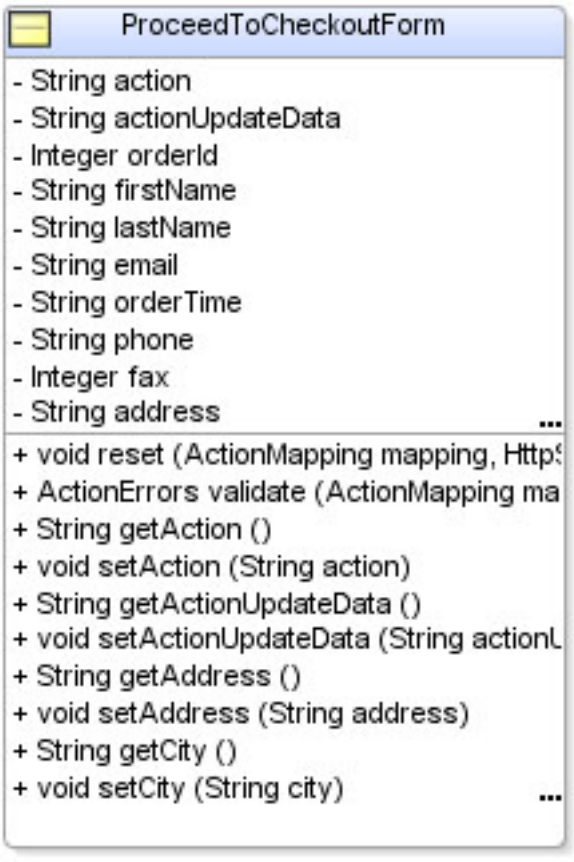
CONTROLLER LAYER

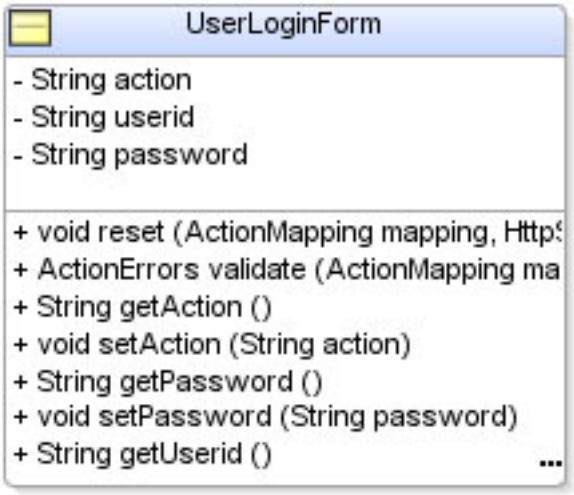
The Web Controller layer provides the mechanisms that control the routing of each web request through the system (the "C" in MVC). Its role is to entertain each browser request, check its validity, invoke the classes from the Business layer needed to process a request, to handle certain processings (such as recording information in the browser session), and finally to route the processed request to the appropriate JSP in the Presentation layer.

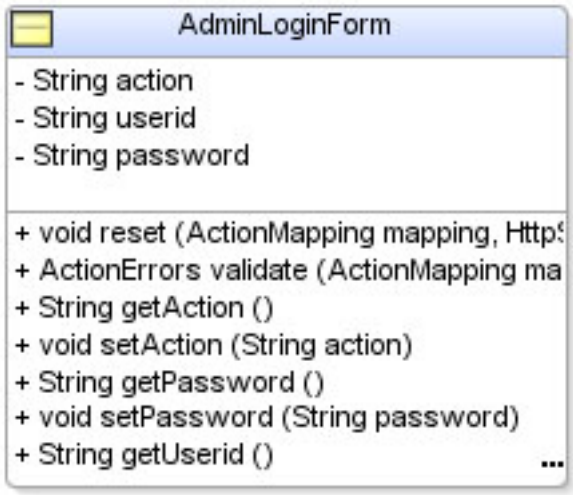
The Controller employs two packages to control the model and the view sections i.e. **roseindia.web.struts.form** (contains varous Action Forms) and **roseindia.web.struts.action** (contains varous Action Classes).

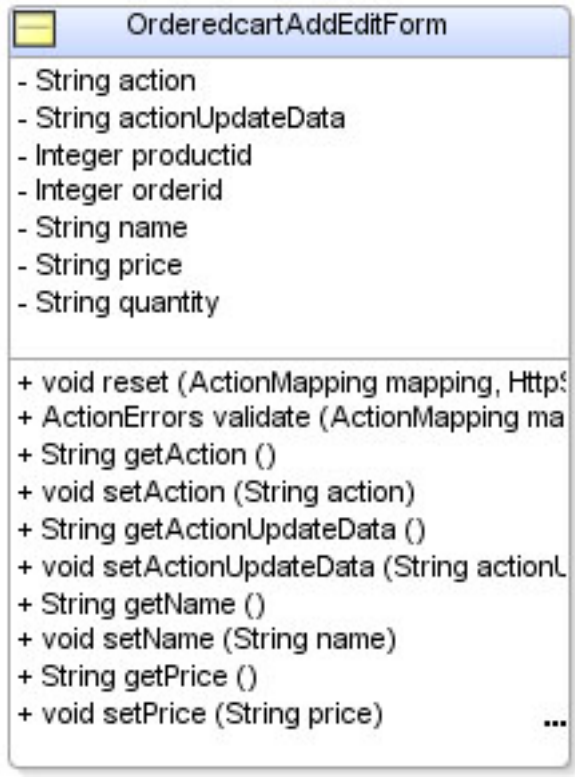
I. The package **roseindia.web.struts.form** contains 10 Action Forms



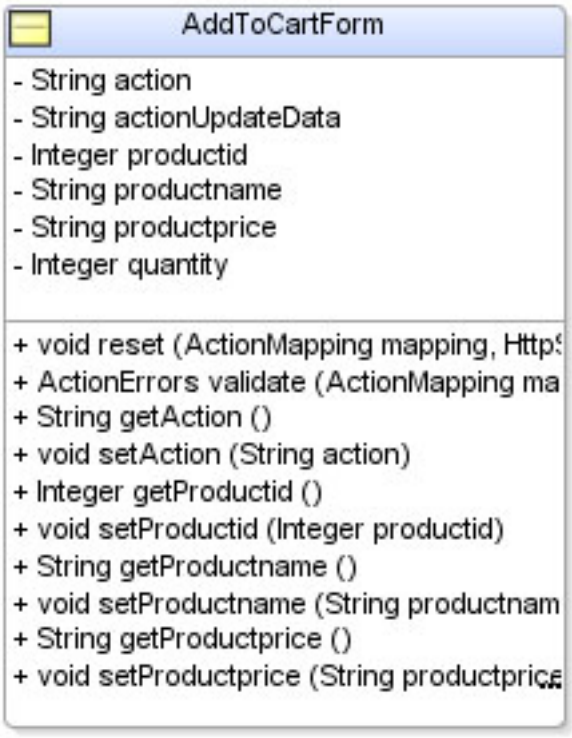


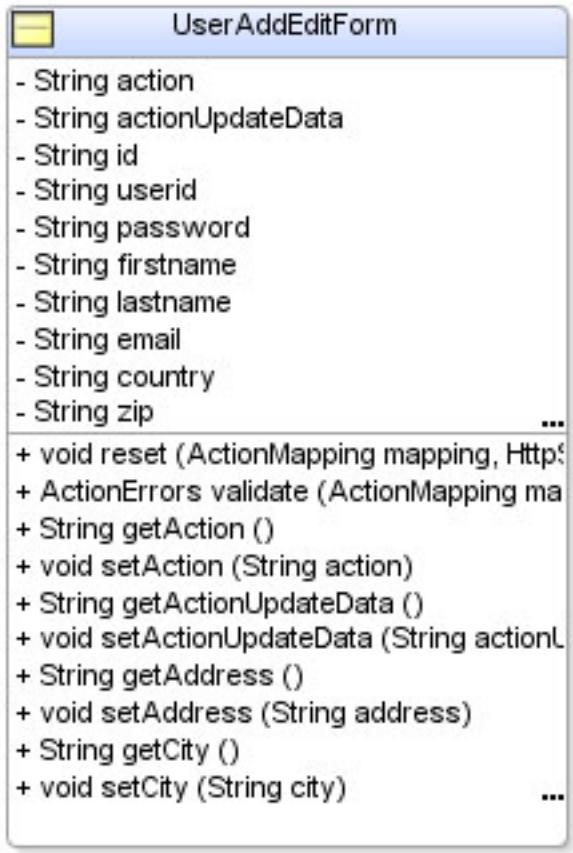




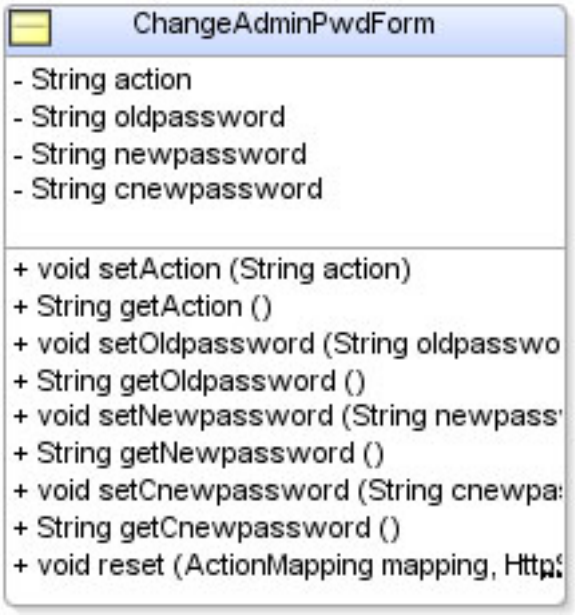



ConfigAddEditForm
- String action
- String actionUpdateData
- Integer id
- String storename
- String storeurl
- String ordernoticeemail
- String cursymbol
- String curcode
- String aboutustext
- String shipdeltext
+ void reset (ActionMapping mapping, Http
+ ActionErrors validate (ActionMapping ma
+ String getAboutustext ()
+ void setAboutustext (String aboutustext)
+ String getAction ()
+ void setAction (String action)
+ String getActionUpdateData ()
+ void setActionUpdateData (String actionL
+ String getCurcode ()
+ void setCurcode (String curcode) ...

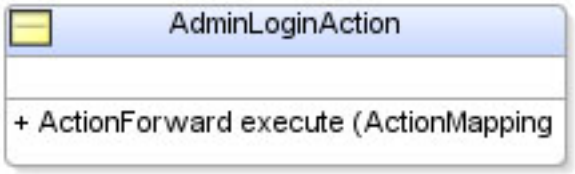

AddToCartForm
- String action
- String actionUpdateData
- Integer productid
- String productname
- String productprice
- Integer quantity
+ void reset (ActionMapping mapping, Http
+ ActionErrors validate (ActionMapping ma
+ String getAction ()
+ void setAction (String action)
+ Integer getProductid ()
+ void setProductid (Integer productid)
+ String getProductname ()
+ void setProductname (String productnam
+ String getProductprice ()
+ void setProductprice (String productpric

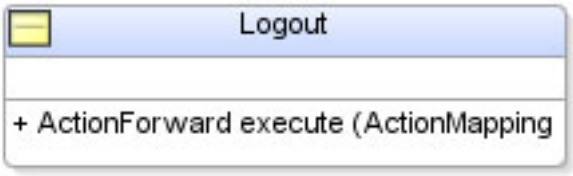

UserAddEditForm
- String action
- String actionUpdateData
- String id
- String userid
- String password
- String firstname
- String lastname
- String email
- String country
- String zip ...
+ void reset (ActionMapping mapping, Http
+ ActionErrors validate (ActionMapping ma
+ String getAction ()
+ void setAction (String action)
+ String getActionUpdateData ()
+ void setActionUpdateData (String actionL
+ String getAddress ()
+ void setAddress (String address)
+ String getCity ()
+ void setCity (String city) ...

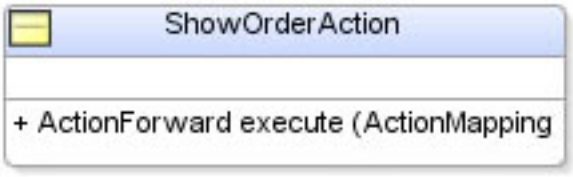

ProductAddEditForm
- String action
- String actionUpdateData
- String productid
- String categoryid
- String productname
- String productprice
- String listprice
- String imagename
- FormFile theFile
- String quantity ...
+ void reset (ActionMapping mapping, Http
+ ActionErrors validate (ActionMapping ma
+ String getAction ()
+ void setAction (String action)
+ String getActionUpdateData ()
+ void setActionUpdateData (String actionL
+ String getBriefdisc ()
+ void setBriefdisc (String briefdisc)
+ String getCategoryid ()
+ void setCategoryid (String categoryid) ...

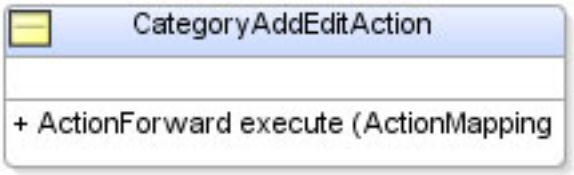


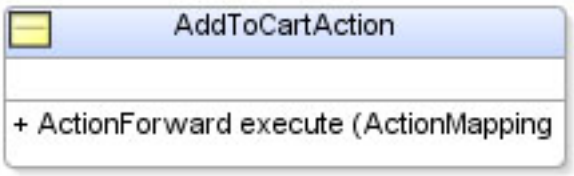
II. The package **roseindia.web.struts.action** contains 16 Action Classes

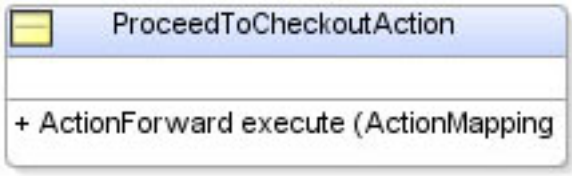


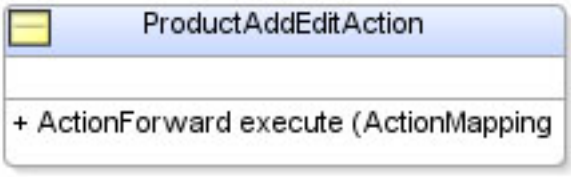


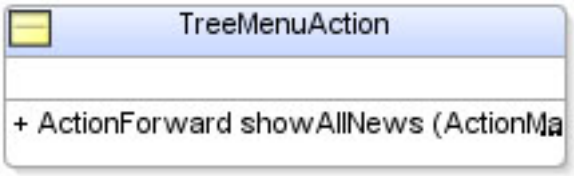


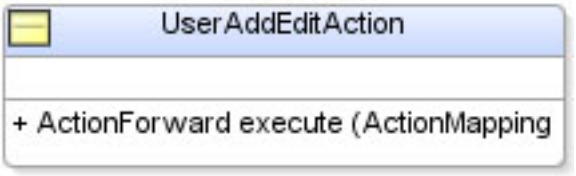


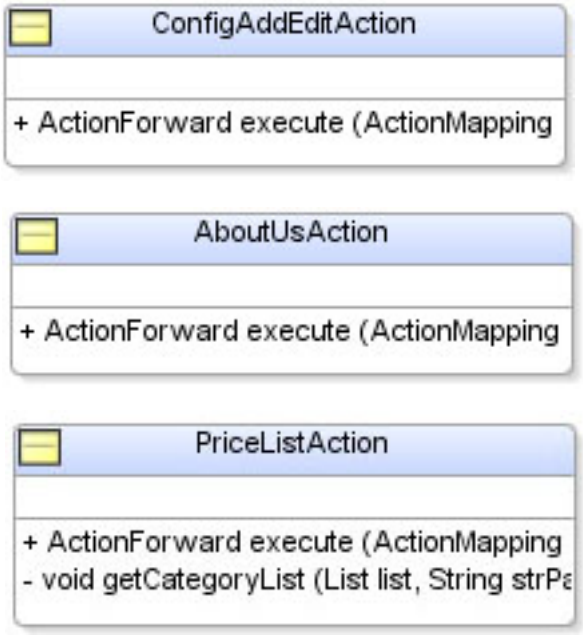












MODEL SECTION

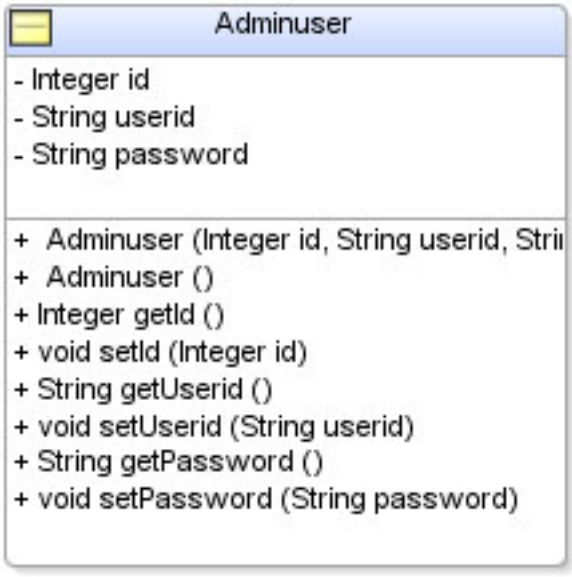
The Business layer is responsible for processing the application's business logic. The Business layer is responsible for managing a cart. It creates a representation of a new cart object needed to choose, remove and update the various items in the cart, and then communicate with the Data Access layer to store a representation of the cart in the database.

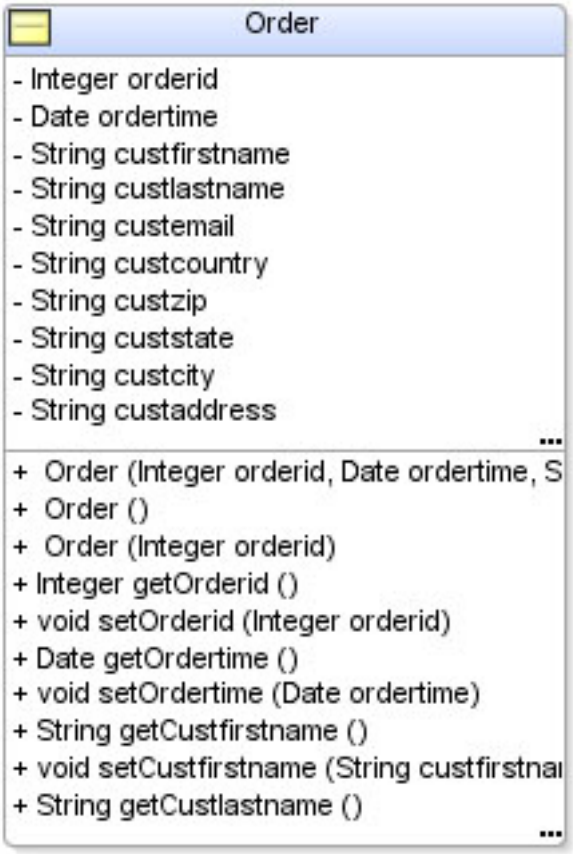
The Data Access layer is responsible for writing and retrieving data to and from the database, and returning the results to the Business layer

The Model is organized into six packages where classes are devoted to interact with the database and to process business logic.

1. The **roseIndia.dao** package contains the ShoppingCartDAOImpl class implementing the ShoppingCartDAO interface. ShoppingCartDAOImpl class deals with all database related concern of administrator and customer sections of the application.

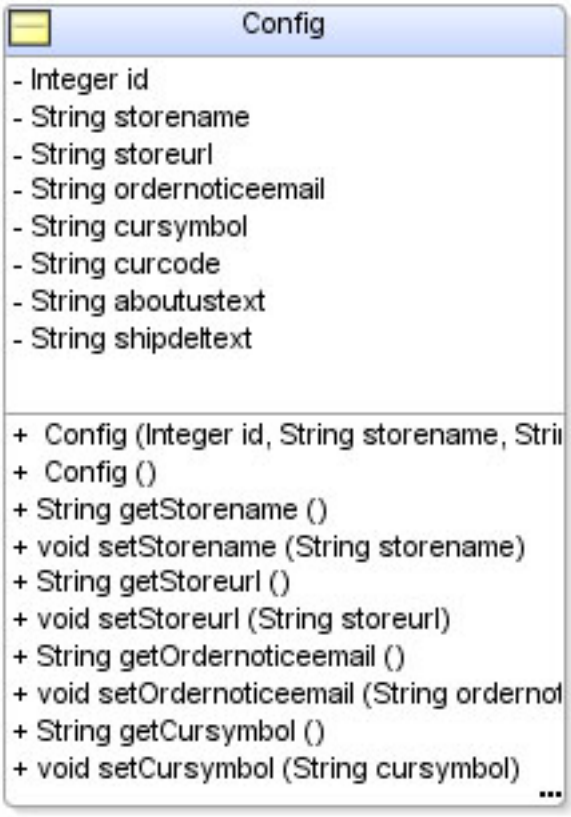
2. Package **roseindia.dao.hibernate** contains 7 java classes and 7 hibernate mapping xml documents (*.hbm.xml). These classes and xml documents does java-Object relational mapping with the underlying database.

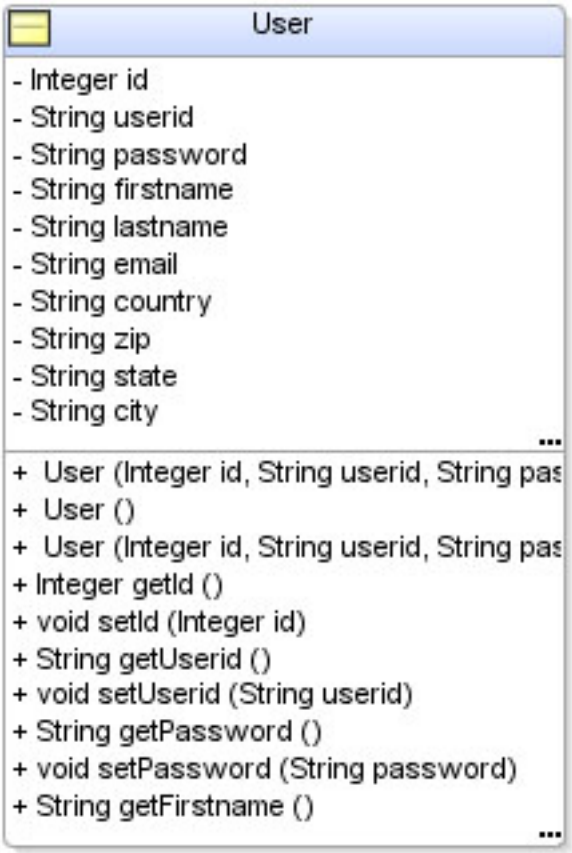


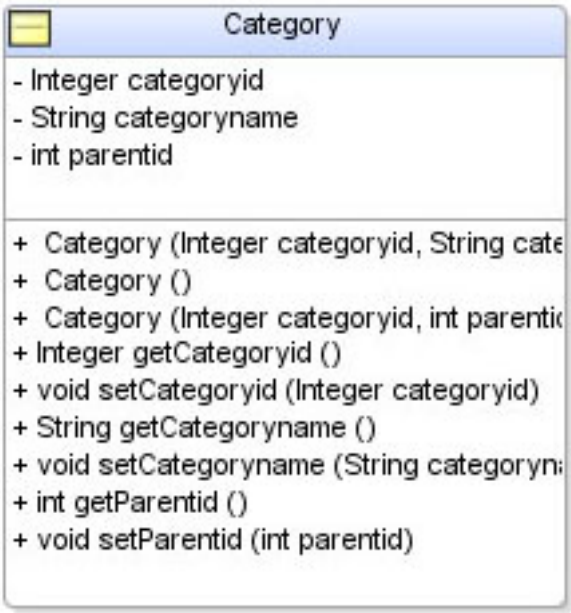

Order
- Integer orderid
- Date ordertime
- String custfirstname
- String custlastname
- String custemail
- String custcountry
- String custzip
- String custstate
- String custcity
- String custaddress
+ Order (Integer orderid, Date ordertime, S
+ Order ()
+ Order (Integer orderid)
+ Integer getOrderid ()
+ void setOrderid (Integer orderid)
+ Date getOrdertime ()
+ void setOrdertime (Date ordertime)
+ String getCustfirstname ()
+ void setCustfirstname (String custfirstnai
+ String getCustlastname ()


Product
- Integer productid
- int categoryid
- String productname
- String productprice
- String listprice
- String imagename
- String quantity
- String description
- String briefdisc
+ Product (Integer productid, int categoryic
+ Product ()
+ Product (Integer productid, int categoryic
+ Integer getProductid ()
+ void setProductid (Integer productid)
+ int getCategoryid ()
+ void setCategoryid (int categoryid)
+ String getProductname ()
+ void setProductname (String productnam
+ String getProductprice ()


Orderedcart
- Integer productid
- Integer orderid
- String name
- Float price
- Integer quantity
+ Orderedcart (Integer productid, Integer c
+ Orderedcart ()
+ Orderedcart (Integer productid, Integer c
+ String getName ()
+ void setName (String name)
+ Float getPrice ()
+ void setPrice (Float price)
+ Integer getQuantity ()
+ void setQuantity (Integer quantity)
+ Integer getOrderid ()

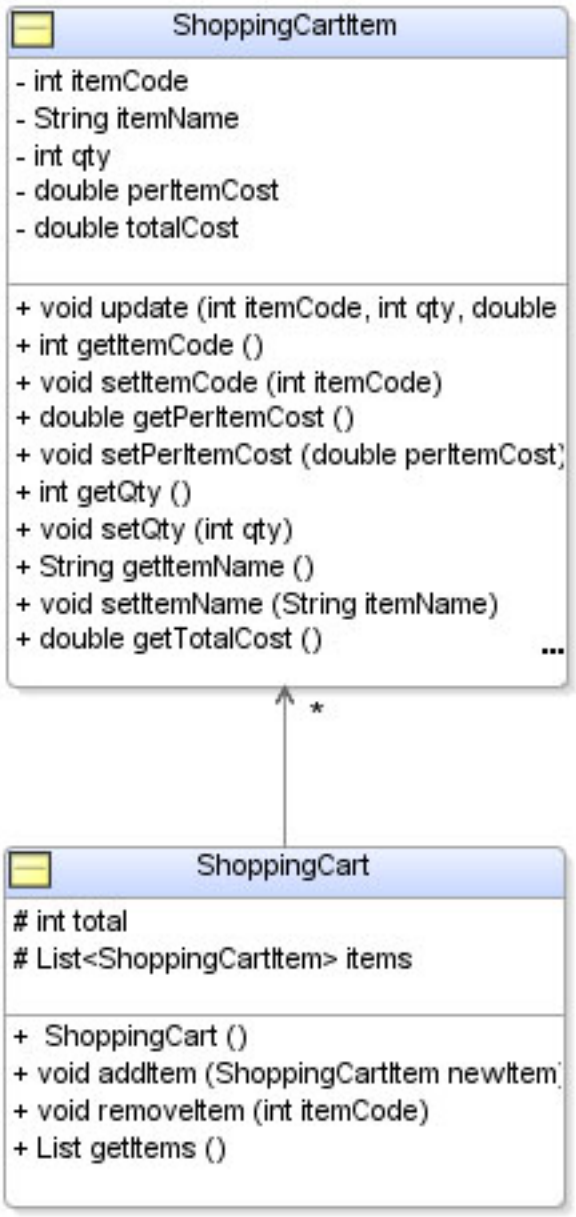

Config
- Integer id
- String storename
- String storeurl
- String ordernoticeemail
- String cursymbol
- String curcode
- String aboutustext
- String shipdeltext
+ Config (Integer id, String storename, Strii
+ Config ()
+ String getStorename ()
+ void setStorename (String storename)
+ String getStoreurl ()
+ void setStoreurl (String storeurl)
+ String getOrdernoticeemail ()
+ void setOrdernoticeemail (String ordernot
+ String getCursymbol ()
+ void setCursymbol (String cursymbol)





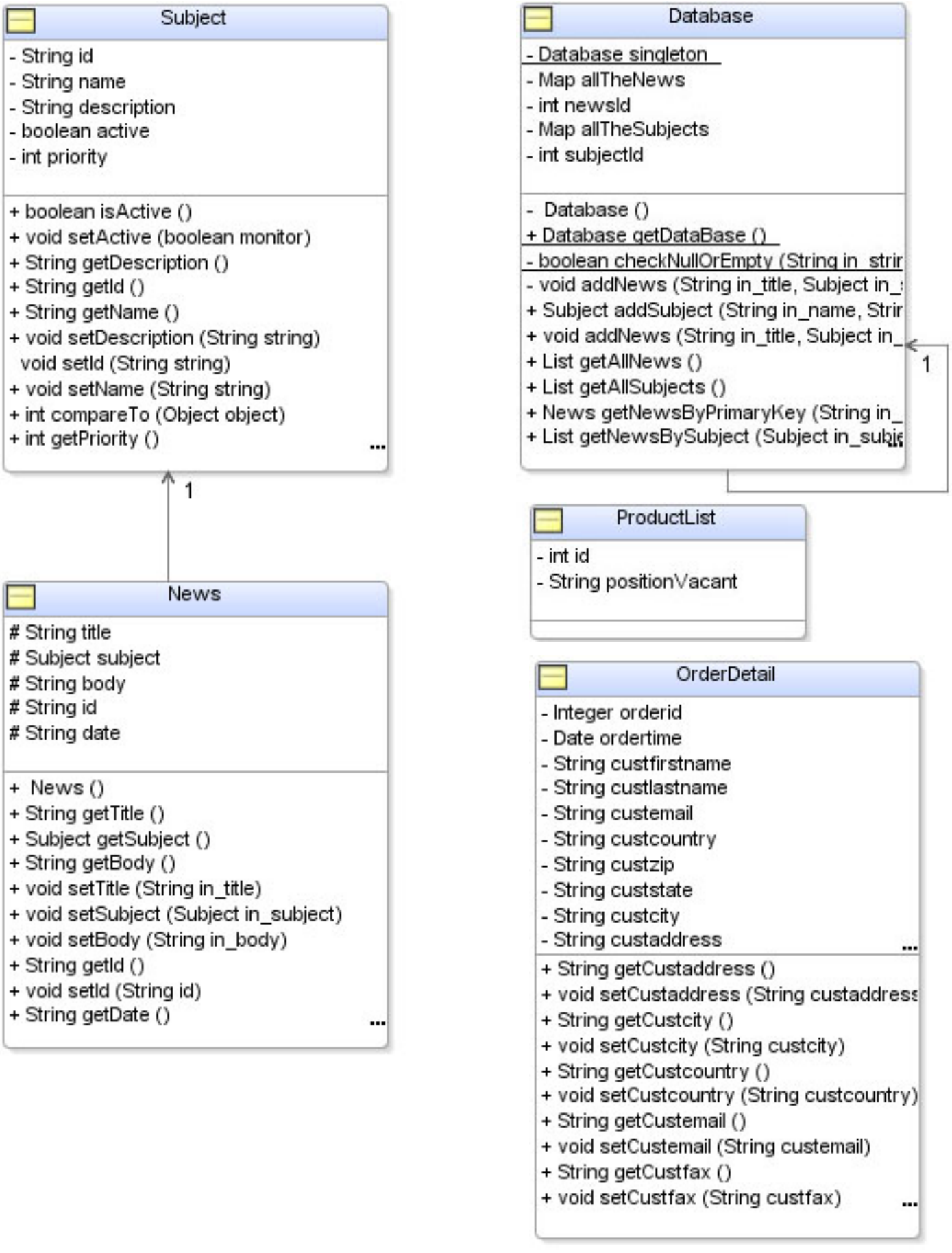
3. The **package roseindia.services** contains a single class ServiceFinder needed to retrieve various integrated services from Spring Application context.

4. The package **roseindia.web.cart** is the heart of the whole application. It contains 2 main classes which enable to add, update and remove items into a shopping cart.

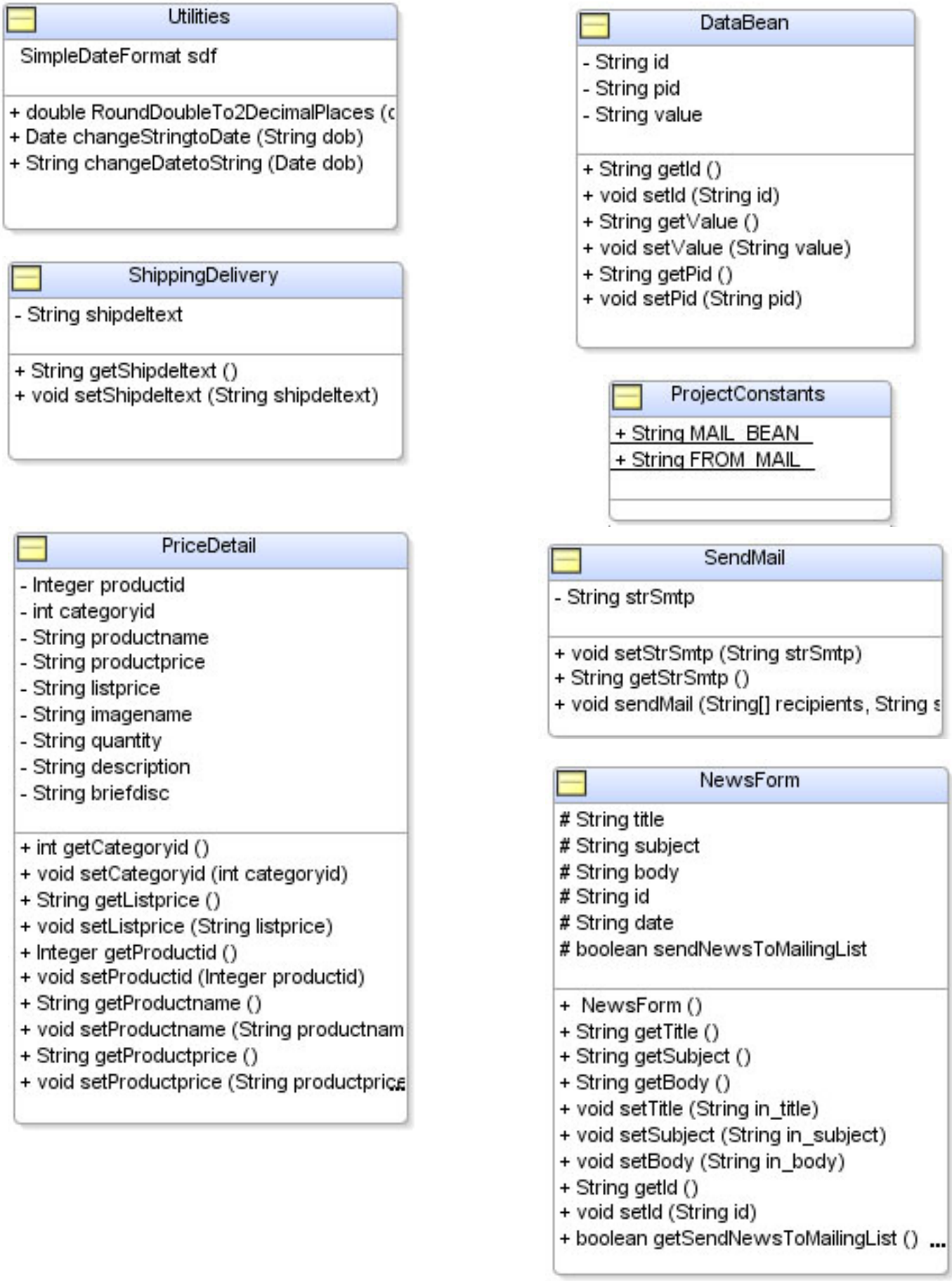


5. **The package roseindia.web.common contains various helper and utility classes needed to process the business logic**.

Shopping Cart



Shopping Cart



The Presentation Layer

After the Web Controller layer routes the request to the Business layer and retrieves the results of the processing, it then forwards the request to the Presentation layer, which is responsible for generating the HTML sent to the user's browser.

Getting Shopping Cart Code

We are maintaining the code and documentation of the shopping cart at http://www.roseindia.net/shoppingcart/. Here, you can download the latest version of the shopping cart application.

Readers Forum: Get Involved

Welcome to the Java Jazz Up community (Readers Forum). Join us as a member of Java Jazz Up community and share your views. This forum is created to collect and disseminate information about Java.

At Java Jazz Up we are in constant process to provide you all information regarding Java Technology at large. Our expert programmers and developers are here to answer your queries that you come across .

You are invited to discuss various issues relating Java. This is the right place to update your Java skills.

The Java Jazz Up team provides all the java-centric stuff in the form of Java tutorials, articles, technological-news updates and development projects.

Post your valuable opinions to make it a grand success around the globe. Send your queries on Java, our developer's team will provide the best possible solution to you, in no-time.

So get involved with Java Jazz Up and feel the excitement, be a member of Jazz-Up Reader's Forum. Add something new to your knowledge box, everyday.

RoseIndia

Please Send Your Query, Problems & Valued Suggestions at :
editor@javajazzup.com

Java Jazz up

A BETTER WAY TO LEARN PROGRAMMING

E - Commerce Application

http://www.roseindia.net/shoppingcart/