# MAHARAJA AGRASEN INSTITUTE OF TECHNOLOGY

**(Guru Gobind Singh Indraprastha University)**

**SUMMER INTERNSHIP REPORT**

Undertaken at



**“Shreyanshi Consultancy Pvt. Ltd.”**

**IN**

**“ADVANCED JAVA(J2EE)”**

**Submitted by:**

**ARJUN KUMAR**

**11014802712**

**CSE(C-3)**

C:\Users\Saurav Singh\Desktop\certi.jpg

**NOIDA,SEC-63 DELHI MOB:08755067204**

***To Whom It May Concern***

This is to certify that the Project entitled “**CABLE ACCESS SERVICE (CAS)**” is being submitted by **Mr. ARJUN KUMAR** in 6 weeks training SHREYANSHI CONSULTANCY PVT. LTD., MTA, Noida-63, Delhi. It is an organized work carried out by him under our guidance and supervision.

**(Mr. Abdullah)**

**Project Guide**

Teacher’s signature

Place: New Delhi

Date: 20th aUGUST 2014

**PROJECT**

**CABLE ACCESS SERVICE (CAS)**

SUBMITTED IN PARTIAL FULFILLMENT OF THE DEGREE

OF

BACHELOR OF TECHNOLOGY

IN

“COMPUTER SCIENCE ENGINEERING”

**MAHARAJA AGRASEN INSTITUTE OF TECHNOLOGY, GGSIPU**

Summer Training after second Year

Year-2014

Under the guidance of:

Mr. Abdullah

Microsoft Technology Associate Pvt. Ltd.

Arjun kumar(11014802712)

**ACKNOWLEDGEMENT**

I hereby take this opportunity to thank all those people whose knowledge and experience helped me bring this report in its present form. It would have been a tough task for me to complete report without their help.

I express my sincere thanks and gratitude to My tutor **Mr.Abdullah, Microsoft Technology Associate Pvt. Ltd.** for providing me the opportunity to pursue my training at the institute.

Finally I would like to express my deep appreciation to my family and friends who have been a constant source of inspiration. I am internally grateful to them for always encouraging me wherever and whenever I needed them.

ARJUN KUMAR

11014802712

**CONTENTS**

1. Acknowledgement

2. List of abbreviations

3. Architecture

4. Institute Profile

5. Abstract

6. List of Tables

7. Introduction(J2EE)

8. Related Theory

8.1)JDBC

8.2) Servlet

8.3) Struts

8.4) JSP

8.5)Hibernate

1. Conclusion
2. References And Bibliography

**LIST OF ABBREVIATIONS**

1. **MTA(Microsoft Technology Associate)**
2. **J2EE(Java 2 Enterprise Edition)**
3. **JDBC(Java Database connectivity)**
4. **JSP(Java Server Pages)**
5. **CAS(Cable Access Services)**
6. **DAO(Data Access Object)**
7. **JVM(Java Virtual Machine)**
8. **EJB(Enterprise Java Beans)**
9. **API’s(Application Program Interface)**
10. **PHP(PHP: Hypertext Preprocessor)**
11. **JNDI(Java Name And Directory Interface)**
12. **JIT(Just In Time Compiler)**
13. **HTTP(Hypertext Transfer Protocol)**
14. **URL(Uniform Resource Location)**
15. **MVC(Model-View-Control)**
16. **JSTL(Java Server Tag Libraries)**
17. **OOPs(Object Oriented Programming)**
18. **POJO(Plain Old Java Object)**
19. **ODBC(Open Database Connectivity)**
20. **XML(Extensible Markup Languages)**

**ARCHITECTURE**

1. **Tools and Technology Used**
   1. **FRONT-END TECHNOLOGY**

**JAVA**

**Java** is a computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java applications are typically compiled to bytecode (class file) that can run on any Java virtual machine (JVM) regardless of computer architecture. Java is, as of 2014, one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers.Java was originally developed by James Gosling at Sun Microsystems (which has since merged into Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

**JAVA PLATFORM**

One characteristic of Java is portability, which means that computer programs written in the Java language must run similarly on any hardware/operating-system platform. This is achieved by compiling the Java language code to an intermediate representation called Java Bytecode, instead of directly to platform-specific [machine code](http://en.wikipedia.org/wiki/Machine_code). Java bytecode instructions are analogous to machine code, but they are intended to be [interpreted](http://en.wikipedia.org/wiki/Interpreter_(computing)) by a [virtual machine](http://en.wikipedia.org/wiki/Virtual_machine) (VM) written specifically for the host hardware. [End-users](http://en.wikipedia.org/wiki/End-user) commonly use a [Java Runtime Environment](http://en.wikipedia.org/wiki/Java_virtual_machine) (JRE) installed on their own machine for standalone Java applications, or in a web browser for Java [applets](http://en.wikipedia.org/wiki/Applet).

Standardized libraries provide a generic way to access host-specific features such as graphics, [threading](http://en.wikipedia.org/wiki/Thread_(computer_science)), and [networking](http://en.wikipedia.org/wiki/Computer_network).

A major benefit of using byte code is porting. However, the overhead of interpretation means that interpreted programs almost always run more slowly than programs compiled to native executable would. Just-in-Time (JIT) compilers were introduced from an early stage that compile byte codes to machine code during runtime.

**IMPLEMENTATION**

[Oracle Corporation](http://en.wikipedia.org/wiki/Oracle_Corporation) is the current owner of the official implementation of the Java SE platform, following their acquisition of [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems) on January 27, 2010. This implementation is based on the original implementation of Java by Sun. The Oracle implementation is available for [Windows](http://en.wikipedia.org/wiki/Windows), [Mac OS X](http://en.wikipedia.org/wiki/Mac_OS_X), [Linux](http://en.wikipedia.org/wiki/Linux) and [Solaris](http://en.wikipedia.org/wiki/Solaris_(operating_system)). Because Java lacks any formal standardization recognized by Ecma International, ISO/IEC, ANSI, or other third-party standards organization, the Oracle implementation is the [de facto standard](http://en.wikipedia.org/wiki/De_facto_standard).

The Oracle implementation is packaged into two different distributions: The Java Runtime Environment (JRE) which contains the parts of the Java SE platform required to run Java programs and is intended for end-users, and the [Java Development Kit](http://en.wikipedia.org/wiki/Java_Development_Kit) (JDK), which is intended for software developers and includes development tools such as the [Java compiler](http://en.wikipedia.org/wiki/Java_compiler), [Javadoc](http://en.wikipedia.org/wiki/Javadoc), [Jar](http://en.wikipedia.org/wiki/JAR_(file_format)), and a [debugger](http://en.wikipedia.org/wiki/Debugger).

[OpenJDK](http://en.wikipedia.org/wiki/OpenJDK) is another notable Java SE implementation that is licensed under the [GPL](http://en.wikipedia.org/wiki/GPL). The implementation started when Sun began releasing the Java source code under the [GPL](http://en.wikipedia.org/wiki/GPL). As of Java SE 7, [OpenJDK](http://en.wikipedia.org/wiki/OpenJDK) is the official Java reference implementation.

The goal of Java is to make all implementations of Java compatible. Historically, Sun's trademark license for usage of the Java brand insists that all implementations be "compatible". This resulted in a legal dispute with [Microsoft](http://en.wikipedia.org/wiki/Microsoft) after Sun claimed that the Microsoft implementation did not support [RMI](http://en.wikipedia.org/wiki/Java_remote_method_invocation) or [JNI](http://en.wikipedia.org/wiki/Java_Native_Interface) and had added platform-specific features of their own. Sun sued in 1997, and in 2001 won a settlement of US$20 million, as well as a court order enforcing the terms of the license from Sun. As a result, Microsoft no longer ships [Windows](http://en.wikipedia.org/wiki/Microsoft_Windows) with Java.

Platform-independent Java is essential to [Java EE](http://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition), and an even more rigorous validation is required to certify an implementation. This environment enables portable server-side applications.

**PERFORMANCE**

Programs written in Java have a reputation for being slower and requiring more memory than those written in [C++](http://en.wikipedia.org/wiki/C%2B%2B_(programming_language)). However, Java programs' execution speed improved significantly with the introduction of [Just-in-time compilation](http://en.wikipedia.org/wiki/Just-in-time_compilation) in 1997/1998 for [Java 1.1](http://en.wikipedia.org/wiki/Java_version_history), the addition of language features supporting better code analysis (such as inner classes, the StringBuilder class, optional assertions, etc.), and optimizations in the Java virtual machine itself, such as [HotSpot](http://en.wikipedia.org/wiki/HotSpot) becoming the default for Sun's JVM in 2000.

Some platforms offer direct hardware support for Java; there are microcontrollers that can run Java in hardware instead of a software Java virtual machine, and [ARM](http://en.wikipedia.org/wiki/ARM_architecture) based processors can have hardware support for executing Java bytecode through their [Jazelle](http://en.wikipedia.org/wiki/Jazelle) option.

**NETBEANS**

**NETBEANS PLATFORM**

[Framework](http://en.wikipedia.org/wiki/Software_framework) for simplifying the development of [Java Swing](http://en.wikipedia.org/wiki/Java_Swing) desktop applications. The NetBeans IDE bundle for Java SE contains what is needed to start developing NetBeans plugins and NetBeans Platform based applications; no additional SDK is required.

Applications can install modules dynamically. Any application can include the Update Center module to allow users of the application to download [digitally signed](http://en.wikipedia.org/wiki/Digital_signature) upgrades and new features directly into the running application. Reinstalling an upgrade or a new release does not force users to download the entire application again.

The platform offers reusable services common to desktop applications, allowing developers to focus on the logic specific to their application. Among the features of the platform are:

* User interface management (e.g. menus and toolbars)
* User settings management
* Storage management (saving and loading any kind of data)
* Window management
* Wizard framework (supports step-by-step dialogs)
* NetBeans Visual Library
* Integrated development tools

NetBeans IDE is a free, open-source, cross-platform IDE with built-in-support for Java Programming Language.

**NETBEANS IDE**

**NetBeans IDE** is an [open-source](http://en.wikipedia.org/wiki/Open_source) integrated development environment. NetBeans IDE supports development of all Java application types ([Java SE](http://en.wikipedia.org/wiki/Java_Platform,_Standard_Edition) (including [JavaFX](http://en.wikipedia.org/wiki/JavaFX)), [Java ME](http://en.wikipedia.org/wiki/Java_Platform,_Micro_Edition), [web](http://en.wikipedia.org/wiki/Web_application), [EJB](http://en.wikipedia.org/wiki/EJB) and [mobile](http://en.wikipedia.org/wiki/MIDlet) applications) out of the box. Among other features are an [Ant](http://en.wikipedia.org/wiki/Apache_Ant)-based project system, [Maven](http://en.wikipedia.org/wiki/Apache_Maven) support, [refactorings](http://en.wikipedia.org/wiki/Refactoring), [version control](http://en.wikipedia.org/wiki/Version_control_system) (supporting [CVS](http://en.wikipedia.org/wiki/Concurrent_Versions_System), [Subversion](http://en.wikipedia.org/wiki/Subversion_(software)), [Git](http://en.wikipedia.org/wiki/Git_(software)), [Mercurial](http://en.wikipedia.org/wiki/Mercurial_(software)) and [Clearcase](http://en.wikipedia.org/wiki/Clearcase)).

**Modularity**: All the functions of the IDE are provided by modules. Each module provides a well defined function, such as support for the[Java language](http://en.wikipedia.org/wiki/Java_(programming_language)), editing, or support for the [CVS](http://en.wikipedia.org/wiki/Concurrent_Versions_System) versioning system, and SVN. NetBeans contains all the modules needed for Java development in a single download, allowing the user to start working immediately. Modules also allow NetBeans to be extended. New features, such as support for other programming languages, can be added by installing additional modules. For instance, [Sun Studio](http://en.wikipedia.org/wiki/Sun_Studio_Compiler_Suite), Sun Java Studio Enterprise, and [Sun Java Studio Creator](http://en.wikipedia.org/wiki/Sun_Java_Studio_Creator) from [Sun Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems) are all based on the NetBeans IDE.

**License**: From July 2006 through 2007, NetBeans IDE was licensed under Sun's [Common Development and Distribution License](http://en.wikipedia.org/wiki/Common_Development_and_Distribution_License) (CDDL), a license based on the [Mozilla Public License](http://en.wikipedia.org/wiki/Mozilla_Public_License) (MPL). In October 2007, Sun announced that NetBeans would henceforth be offered under a [dual license](http://en.wikipedia.org/wiki/Dual_license) of the CDDL and the [GPL](http://en.wikipedia.org/wiki/GPL) version 2 licenses, with the [GPL linking exception](http://en.wikipedia.org/wiki/GPL_linking_exception) for [GNU Classpath](http://en.wikipedia.org/wiki/GNU_Classpath)

**1.2 BACK END TECHNOLOGY**

**SQL Server 2005 Database Objects**

**Server**

The server scope encompasses all the objects that exist on the instance of SQL Server, regardless of theirrespective database or namespace. The database object resides within the server scope.One of the more confusing terms when working with SQL Server 2005 is the term *server*.

**Database**

An instance of SQL Server 2005 cancontain many databases. A typical database application is constrained within one database that containsall the data objects required to provide the functionality the application requires. This is not always thecase, but it is the most common.

**Schema**

Each database can contain one or more schemas. A schema is a namespace for database objects. All dataobjects in a SQL Server 2005 database reside in a specific schema.SQL Server 2005 implements the ANSI schema object. A database schema is a defined namespace inwhich database objects exist.

**Object Names**

Every object in a SQL Server 2005 database is identified by a four-part, fully qualified name. This fullyqualified name takes the form of server.Database.Schema.Object.

**SQL Server 2005 Databases**

There are two types of databases in SQL Server: system databases and user databases. The *systemdatabases* are used to store system-wide data and metadata. *User databases* are created by users who havethe appropriate level of permissions to store application data.

**System Databases**

The system databases are comprised of Master, Model, MSDB, TempDB, and the hidden Resourcedatabase. If the server is configured to be a replication distributor, there will also be at least one systemdistribution database that is named during the replication configuration process.

* The Master Database

The Master database is used to record all server-level objects in SQL Server 2005. This includes ServerLogon accounts, Linked Server definitions, and EndPoints. The Master database also records informationabout all the other databases on the server.

* The Model Database

The Model database is a template database. Whenever a new database is created (including the systemdatabase TempDB), a copy of the Model database is created and renamed with the name of the databasebeing created.

* The MSDB Database

I mostly think of the MSDB database as the SQL Server Agent’s database. That’s because the SQL ServerAgent uses the MSDB database extensively for the storage of automated job definitions, job schedules,operator definitions, and alert definitions.

* The TempDB Database

The TempDB database is used by SQL Server to store data—yes, you guessed it, temporarily.

**SQL SERVER 2005**

SQL Server 2005 (codenamed Yukon), released in October 2005, is the successor to SQL Server 2000. It included native support for managing XML data, in addition to relational data. For this purpose, it defined an xml data type that could be used either as a data type in database columns or as literals in queries. XML columns can be associated with XSD schemas; XML data being stored is verified against the schema. XML is converted to an internal binary data type before being stored in the database. Specialized indexing methods were made available for XML data. XML data is queried using XQuery; Common Language Runtime (CLR) integration was a main features with this edition, enabling one to write SQL code as Managed Code by the CLR. SQL Server 2005 added some extensions to the T-SQL language to allow embedding XQuery queries in T-SQL. In addition, it also defines a new extension to XQuery, called XML DML that allows query-based modifications to XML data. SQL Server 2005 also allows a database server to be exposed over web services using TDS packets encapsulated within SOAP (protocol) requests. When the data is accessed over web services, results are returned as XML.

For relational data, T-SQL has been augmented with error handling features (try/catch) and support for recursive queries with CTEs (Common Table Expressions). SQL Server 2005 has also been enhanced with new indexing algorithms, syntax and better error recovery systems. Data pages are checksummed for better error resiliency, and optimistic concurrency support has been added for better performance. Permissions and access control have been made more granular and the query processor handles concurrent execution of queries in a more efficient way. Partitions on tables and indexes are supported natively, so scaling out a database onto a cluster is easier. SQL CLR was introduced with SQL Server 2005 to let it integrate with the .NET Framework.

SQL Server 2005 introduced “MAMS” (Multiple Active Results Sets), a method of allowing usage of database connections for multiple purposes.

SQL Server 2005 introduced DMVs (Dynamic Management Views), which are specialized views and functions that return server state information that can be used to monitor the health of a server instance, diagnose problems, and tune performance.

SQL Server 2005 introduced Database Mirroring, but it was not fully supported until the first Service Pack release (SP1). In the initial release (RTM) of SQL Server 2005, database mirroring was available, but unsupported. In order to implement database mirroring in the RTM version, you had to apply trace flag 1400 at startup. Database mirroring is a high availability option that provides redundancy and failover capabilities at the database level. Failover can be performed manually or can be configured for automatic failover. Automatic failover requires a witness partner and an operating mode of synchronous (also known as high-safety or full safety)

SQL Server includes better compression features, which also helps in improving scalability. It enhanced the indexing algorithms and introduced the notion of filtered indexes. It also includes Resource Governor that allows reserving resources for certain users or workflows. It also includes capabilities for transparent encryption of data (TDE) as well as compression of backups.SQL Server 2008 supports the ADO.NET Entity Framework and the reporting tools, replication, and data definition will be built around the Entity Data Model.SQL Server Reporting Services will gain charting capabilities from the integration of the data visualization products from Dundas Data Visualization, Inc., which was acquired by Microsoft.[ On the management side, SQL Server 2008 includes the Declarative Management Framework which allows configuring policies and constraints, on the entire database or certain tables, declaratively. The version of SQL Server Management Studio included with SQL Server 2008 supports IntelliSense for SQL queries against a SQL Server 2008 Database Engine. SQL Server 2008 also makes the databases available via Windows Power Shell providers and management functionality available as Camlets, so that the server and all the running instances can be managed from Windows Power Shell.

**INSTITUTE PROFILE**

**ABOUT MICROSOFT TECHNOLOGY ASSOCIATE(MTA)**



**Microsoft Training Associate** a programming language based Company, providing the best technique and solution to the problem. It offers professional courses like C, C++, CORE JAVA, ADVANCED JAVA(J2EE), LINUX, PHP etc. which increases knowledge of the aspirants, helps them to know how to work in a company and gives them the confidence to apply their logics.

Through technological innovations, quality and timely deliveries, It work to ensure increased values and optimized throughput for the customers.

Its solutions ensure customers sustain their competitive edge and added value and volumes to their business.

It aspire to serve the industry through training, development and consulting.

A career at Microsoft Training Associate comes with opportunities for continuous learning coupled with professional challenges that highlight individual performance. We consistently believed in and practiced openness, transparency and equal opportunity.

**ACTIVITIES**

* MTA offers advanced java course which mainly includes collection, wrapper class, autoboxing, set, list, hash, JDBC, DAO activity, servlet, cookies, JSP, struts and hibernate. This helps in creating a database, writing SQL queries etc.

The advanced java process involves:

* + Making the code platform independent
  + The code is converted into byte code when goes to JVM
  + Based on exception handling and techniques of threading
  + Making online websites or online projects
* J2EE helps customers and developers save valuable time as it could make a website which has all the requirements of the customer with minimum possibility of error.  
  It provide expert design support and development services.

Our Services include:

* + Website Designing
  + Game Designs
  + Web based application
  + J2EE API’s and container
  + JDBC
  + Servlets and JSP
  + Enterprise java beans(EJB)
  + Struts and Hibernate
  + Design patterns(DAO and DTO)

#### Design Realization

#### MTA offers product realization services from idea-to-solution, concept-to-deployment covering various aspects of conceptualization, architecture, website design, planning implementation, development, website development, coding and enhancement.

**ABSTRACT**

**ABOUT**

**Objective-**

This course is designed to ensure that students of Engineering College with academic capabilities will have the skill set needed to deal with the challenges involved in real-world website designing technologies to meet the needs of industries both today and in the future.  
The course considers programming techniques which can help to ensure that the web applications are reliable.

In my project, we have made a website which gets all the customer details and tell the subscription details of customers. The website also provide the facility to choose the package or channels of their own choice which they love to watch and also fits their budget. We have different kinds and range of packages of various prices. Also we have special package offers for limited period and exciting offers for new customers who joins us.

The course is taught mainly using the JAVA programming language, with MYSQL, also a bit of JAVASCRIPT and CSS in addition as they are used in making the websites. It also made me comfortable with NETBEANS, which is a very good software to code upon. It is very easy to deal with.

**LIST OF TABLES**

|  |  |
| --- | --- |
| **DATA FIELD** | **DATA TYPE** |
| FIRST NAME | STRING |
| LAST NAME | STRING |
| MOBILE NUMBER | NUMBERS |
| ADDRESS | VARCHAR |
| CITY | STRING |
| STATE | STRING |
| E-MAIL ID | VARCHAR |

**INTRODUCTION**

Cable access service(CAS) is a kind of service in which we make a web application like DISH TV and TATA SKY. This is a very customer friendly website because in this customers does not have to go outside their room or stand in queue for long time. What they have to do is just open the website, click on the recharge button and its done!

In this project, we have created a database which will collect the information of the customers and use them at the time of requirement. It retains the complete detail of every customer i.e. when the customer did their last recharge, what packages they selected, information of new users like when they joined the website.

By this we have made the world class TV viewing experience with superb picture quality and sound. We named our service as D-TV. We are the front runners in entertainment and Asia’s largest DTH company.

In this project, we also provided our contacts to the customers so that they can reach us anytime and anywhere. We have shown our achievements in the past years and made our website very easy to understand so that the customer does not face any kind of problem. We have provided different kinds of packages according to the needs of the customers and at very genuine rates.

Key benefits of Cable Access Service

* Everything is done online
* Very less human efforts
* Better services and informations provided to the customers
* Available for 24x7
* Easy to use and quicker to access
* Less efforts in doing calculations and recording details of customers

**The course is split into Six modules:**

* **Basic Concepts**
* **List, shell and Hash**
* **JDBC**
* **Java Server Pages(JSP)**
* **Struts and Hibernate**
* **Springs**

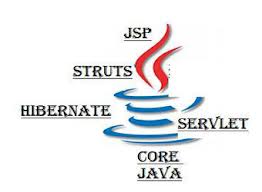
**The Duration of the Training is:**

* 60 Sessions of 2 hrs each (About 1.5 years).

(In 6 weeks training, I have covered the top five modules of the training.)

**RELATED THEORY**

**ADVANCED JAVA (J2EE):-**



Java is a high-level programming language originally developed by Sun Microsystems and released in 1995. Java runs on a variety of platforms, such as Windows, Mac OS, and the various versions of UNIX. This tutorial gives a complete understanding of Java.

Java is Programming language as well as the platform for developing and deploying the applications of various types. Java is Programming Language as it provides the API, Compiler and the testing tools for developing the applications. It is a platform as it provides the Java Virtual Machine for running the applications on different operating system.

**Java as a Programming Language**

Java is a Programming Language which provides many features and used to compile the Java program and test the application using the JDK. The main feature of Java Programming Language is its platform independence, means once it is developed on one platform it can run on any other platform (Windows. Linux, Unix, Mac etc..). So, this feature of Java Programming is also known as write one and run anywhere.

Here are the features of the Java Programming Language:

* **Simple:** Java is simple and its free from pointers which makes it ideal programming language for writing the bug free code. If we use the pointers in any programming language the there are much more changes of introducing the bugs in the program. Java was designed with the objective of easier and bug free code development.
* **Object oriented:** Java is fully Object Oriented Programming Language
* **Distributed:** Java programming language is very powerful can be used to develop the network based applications. It supports the development of Centralized and Distributed applications.
* **Multithreaded:** Java supports multithreading and it is extensively used for the development of multithreaded applications.
* **Dynamic:** Java programming language is dynamic as it supports the dynamic memory and also it enables the developers to create the dynamic application.
* **Architecture neutral:** Java Programming Language is architecture neutral as it can run on processors of any architecture.
* **Portable:** Java Programming Language is highly portable as it can run on all the operating system having the JVM of that system.
* **High performance:** Java Programming Language provides high performance due to Just in time compiler (JIT), automatic garbage collection, no pointers and support of multithreading.
* **Robust:** Java is good programming language in terms of handling the compile time and runtime errors.
* **Secure:** Java programming language is deal for developing the secured applications and it supports Internal and External security for the applications.

**How Java Program Works?**

Java program is written in text editor and saved at simple text file with the .java extension. For example HelloWorld.java is very simple program in Java.

Advertisement

Then Java compiler is used to compile the program. If no error is found in the program Java compiler compiles the file into a bytecode file. For example if you compile the HelloWorld.java file it will produce the HelloWorld.class file.

The Java interpreter is used for running the Java program. Following diagram shows the process of compilation and execution of Java program.

Through Java interpreter (JVM) you can run the compiled program on any other OS and platform.



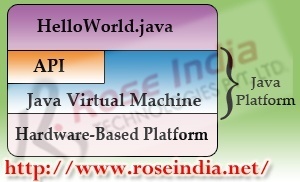
**Java as a Platform**

The term Platform is used for the Hardware and Software environment which actually runs the software. These days there are many platforms available for the users such as Windows, Linux, Solaris, Mac OS etc. Java program can be run on all these platforms.

The Java Platform (JVM) is software only platform which runs on all the OS (Windows, Linux, Solaris, Mac OS). There are following components of the Java Platform:

1. **The Java Virtual Machine:** The Java Virtual Machine contains the compiler for compiling the Bytecode (java) into machine specific instructions. The JVM compiles the bytecode into the machine specific code and then run on the platform.
2. **The Java Application Programming Interface (API):** In Java there are large number of ready-made components which can be used to develop the application. All these components are commonly know as the libraries in Java.

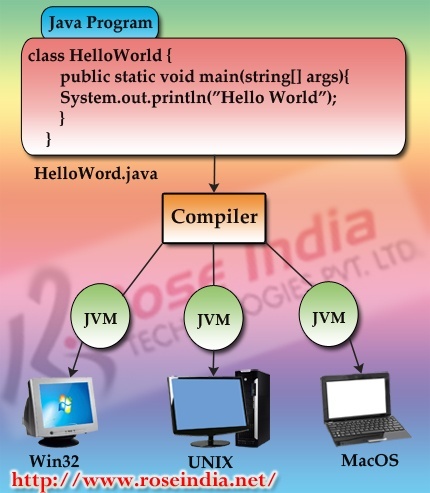
Following diagram shows the stack of the Java Platform:



Since Java is platform-independent environment and it runs in JVM, it is a bit slower then the native programs. But the advancement in the compiler and JVM technologies makes it comparable to the native code. More over you will get the portability and platform independence with the Java Programming Language.

**Distribution to the Java Program**

Since Java is platform independent programming language it can run on any platform. Once Java file is compiled into .class file it can be distributed on any platform. Following diagram shows how a program can run on different platform:



**What Java Programming Technology Can do?**

Java is general-purpose, high-level, object oriented programming language and a software platform for deploying the applications. It gives you following features for your applications:

* **Development Tool:** Java provides the development tool form compiling (javac), for running the application (java) and documentation (javadoc) tools for the easy development and testing of the applications.
* **API:** It provides the Application Programming Interface (API) which is ready made packages for accomplishing various tasks in the programming. Java provides large set the API to the developers.
* **Deployment technologies:** Java programming language provides many deployment technologies such as Web Start, Applets etc. for easy deployment of the applications.
* **User Interfaces Toolkits:** It provides the JavaFX, Swing, and Java 2D toolkits for easy development of GUI for the applications.
* **Various Integration Libraries:** There are many integration libraries such as Java IDL API, JDBC API, Java Naming and Directory Interface (JNDI) API, Java RMI, and Java Remote Method Invocation over Internet Inter-ORB Protocol Technology (Java RMI-IIOP Technology) which can be used for the development of Java based applications.

**Why Java is important as top Programming Language?**

Java Programming Language is one of the top object oriented programming language used in the Software industry. Java Programming Language can be leaned with less effort and it enables the developers to better code.

Here are thing in which Java Programming Language will help you much:

* **Quick Start:** Java is easy to learn programming language and C++ programmer can learn it very easily.
* **Write less code:** Java programming language is much easier and less no of line is need for any program. It is estimated that it if you write and program in C++ and write the same program in Java, in the Java program will have have to write 4 times less code.
* **Better coding:** Java is devoid of pointer and it makes Java programming much easier. In Java you can user Generics which type checks your code in compile time. Java Programming Language is all about writing the better code with less efforts. This is possible only due to availability of various API's available with the Java Programming Language.
* **Platform independence and write one and run anywhere:** Java Programming Language is platform independent and you can write your code once, compile it, package it and run it on any platform.

**JAVA DATABASE CONNECTIVITY (JDBC):-**

**JDBC** is a Java-based data access technology (Java Standard Edition platform) from Oracle Corporation. This technology is an API for the Java programming language that defines how a client may access a database. It provides methods for querying and updating data in a database. JDBC is oriented towards relational databases. A JDBC-to-ODBC0 bridge enables connections to any ODBC-accessible data source in the JVM host environment.

JDBC allows multiple implementations to exist and be used by the same application. The API provides a mechanism for dynamically loading the correct Java packages and registering them with the JDBC Driver Manager. The Driver Manager is used as a connection factory for creating JDBC connections.

JDBC connections support creating and executing statements. These may be update statements such as SQL's CREATE, INSERT, UPDATE and DELETE, or they may be query statements such as SELECT. Additionally, stored procedures may be invoked through a JDBC connection. JDBC represents statements using one of the following classes:

* Statement – the statement is sent to the database server each and every time.
* PreparedStatement – the statement is cached and then the execution path is pre-determined on the database server allowing it to be executed multiple times in an efficient manner.
* CallableStatement – used for executing stored procedures on the database.

Update statements such as INSERT, UPDATE and DELETE return an update count that indicates how many rows were affected in the database. These statements do not return any other information.

Query statements return a JDBC row result set. The row result set is used to walk over the result set. Individual columns in a row are retrieved either by name or by column number. There may be any number of rows in the result set. The row result set has metadata that describes the names of the columns and their types.

There is an extension to the basic JDBC API in the javax.sql.

JDBC connections are often managed via a connection pool rather than obtained directly from the driver. Examples of connection pools include BoneCP, C3P0 and DBCP

## Examples

The method Class.forName(String) is used to load the JDBC driver class. The line below causes the JDBC driver from *some jdbc vendor* to be loaded into the application. (Some JVMs also require the class to be instantiated with .newInstance().)

**Class**.forName( "com.somejdbcvendor.TheirJdbcDriver" );

In JDBC 4.0, it is no longer necessary to explicitly load JDBC drivers using Class.forName(). See JDBC 4.0 Enhancements in Java SE 6.

When a Driver class is loaded, it creates an instance of itself and registers it with the DriverManager. This can be done by including the needed code in the driver class's static block. E.g., DriverManager.registerDriver(Driver driver)

Now when a connection is needed, one of the DriverManager.getConnection() methods is used to create a JDBC connection.

**Connection** conn = **DriverManager**.getConnection(

"jdbc:somejdbcvendor:other data needed by some jdbc vendor",

"myLogin",

"myPassword" );

**try** {

*/\* you use the connection here \*/*

} **finally** {

*//It's important to close the connection when you are done with it*

**try** { conn.close(); } **catch** (**Throwable** ignore) { */\* Propagate the original exception*

*instead of this one that you may want just logged \*/* }

}

Using Java's try-with-resources statement will make the above code cleaner:

**try** (**Connection** conn = **DriverManager**.getConnection(

"jdbc:somejdbcvendor:other data needed by some jdbc vendor",

"myLogin",

"myPassword" ) ) {

*/\* you use the connection here \*/*

} *// the VM will take care of closing the connection*

The URL used is dependent upon the particular JDBC driver. It will always begin with the "jdbc:" protocol, but the rest is up to the particular vendor. Once a connection is established, a statement can be created.

**try** (**Statement** stmt = conn.createStatement()) {

stmt.executeUpdate( "INSERT INTO MyTable( name ) VALUES ( 'my name' ) " );

}

Note that Connections, Statements, and ResultSets often tie up operating system resources such as sockets or file descriptors. In the case of Connections to remote database servers, further resources are tied up on the server, e.g., cursors for currently open ResultSets. It is vital to close() any JDBC object as soon as it has played its part; garbage collection should not be relied upon. Forgetting to close() things properly results in spurious errors and misbehaviour. The above try-with-resources construct is a recommended code pattern to use with JDBC objects.

Data is retrieved from the database using a database query mechanism. The example below shows creating a statement and executing a query.

**try** (**Statement** stmt = conn.createStatement();

**ResultSet** rs = stmt.executeQuery( "SELECT \* FROM MyTable" )

) {

**while** ( rs.next() ) {

**int** numColumns = rs.getMetaData().getColumnCount();

**for** ( **int** i = 1 ; i <= numColumns ; i++ ) {

*// Column numbers start at 1.*

*// Also there are many methods on the result set to return*

*// the column as a particular type. Refer to the Sun documentation*

*// for the list of valid conversions.*

**System**.out.println( "COLUMN " + i + " = " + rs.getObject(i) );

}

}

}

Typically, however, it would be rare for a seasoned Java programmer to code in such a fashion. The usual practice would be to abstract the database logic into an entirely different class and to pass preprocessed strings (perhaps derived themselves from a further abstracted class) containing SQL statements and the connection to the required methods. Abstracting the data model from the application code makes it more likely that changes to the application and data model can be made independently.

An example of a PreparedStatement query, using conn and class from first example.

**try** (**PreparedStatement** ps =

conn.prepareStatement( "SELECT i.\*, j.\* FROM Omega i, Zappa j WHERE i.name = ? AND j.num = ?" )

){

*// In the SQL statement being prepared, each question mark is a placeholder*

*// that must be replaced with a value you provide through a "set" method invocation.*

*// The following two method calls replace the two placeholders; the first is*

*// replaced by a string value, and the second by an integer value.*

ps.setString(1, "Poor Yorick");

ps.setInt(2, 8008);

*// The ResultSet, rs, conveys the result of executing the SQL statement.*

*// Each time you call rs.next(), an internal row pointer, or cursor,*

*// is advanced to the next row of the result. The cursor initially is*

*// positioned before the first row.*

**try** (**ResultSet** rs = ps.executeQuery()) {

**while** ( rs.next() ) {

**int** numColumns = rs.getMetaData().getColumnCount();

**for** ( **int** i = 1 ; i <= numColumns ; i++ ) {

*// Column numbers start at 1.*

*// Also there are many methods on the result set to return*

*// the column as a particular type. Refer to the Sun documentation*

*// for the list of valid conversions.*

**System**.out.println( "COLUMN " + i + " = " + rs.getObject(i) );

} *// for*

} *// while*

} *// try*

} *// try*

If a database operation fails, JDBC raises an SQLException. There is typically very little one can do to recover from such an error, apart from logging it with as much detail as possible. It is recommended that the SQLException be translated into an application domain exception (an unchecked one) that eventually results in a transaction rollback and a notification to the user.

An example of a database transaction:

**boolean** autoCommitDefault = conn.getAutoCommit();

**try** {

conn.setAutoCommit(**false**);

*/\* You execute statements against conn here transactionally \*/*

conn.commit();

} **catch** (**Throwable** e) {

**try** { conn.rollback(); } **catch** (**Throwable** ignore) {}

**throw** e;

} **finally** {

**try** { conn.setAutoCommit(autoCommitDefault); } **catch** (**Throwable** ignore) {}

}

**SERVLET:-**

The **servlet** is a Java programming language class used to extend the capabilities of a server. Although servlets can respond to any types of requests, they are commonly used to extend the applications hosted by web servers, so they can be thought of as Java applets that run on servers instead of in web browsers.These kinds of servlets are the Java counterpart to other dynamic Web content technologies such as PHP and ASP.NET.

Servlets are most often used to:

* Process or store data that was submitted from an HTML form.
* Provide dynamic content such as the results of a database query
* Manage state information that does not exist in the stateless HTTP protocol, such as filling the articles into the shopping cart of the appropriate customer

Technically speaking, a "servlet" is a Java class in Java EE that conforms to the Java Servlet API, a standard for implementing Java classes which respond to requests. Servlets could in principle communicate over any client–server protocol, but they are most often used with the HTTP protocol. Thus "servlet" is often used as shorthand for "HTTP servlet". Thus, a software developer may use a servlet to add dynamic content to a web server using the Java platform. The generated content is commonly HTML, but may be other data such as XML. Servlets can maintain state in session variables across many server transactions by using HTTP cookies, or URL rewriting.

To deploy and run a servlet, a web container must be used. A web container (also known as a servlet container) is essentially the component of a web server that interacts with the servlets. The web container is responsible for managing the lifecycle of servlets, mapping a URL to a particular servlet and ensuring that the URL requester has the correct access rights.

The Servlet API, contained in the Java package hierarchy javax.servlet, defines the expected interactions of the web container and a servlet.

A Servlet is an object that receives a request and generates a response based on that request. The basic Servlet package defines Java objects to represent servlet requests and responses, as well as objects to reflect the servlet's configuration parameters and execution environment. The package javax.servlet.http defines HTTP-specific subclasses of the generic servlet elements, including session management objects that track multiple requests and responses between the web server and a client. Servlets may be packaged in a WAR file as a web application.

Servlets can be generated automatically from Java Server Pages (JSP) by the JavaServer Pages compiler. The difference between servlets and JSP is that servlets typically embed HTML inside Java code, while JSPs embed Java code in HTML. While the direct usage of servlets to generate HTML (as shown in the example below) has become rare, the higher level MVC web framework in Java EE (JSF) still explicitly uses the servlet technology for the low level request/response handling via the FacesServlet. A somewhat older usage is to use servlets in conjunction with JSPs in a pattern called "Model 2", which is a flavor of the model–view–control

The current version of Servlet is 3.1.

**STRUTS:-**

**Apache Struts** was an open-source web application framework for developing Java EE web applications. It uses and extends the Java Servlet API to encourage developers to adopt a model–view–controller (MVC) architecture. It was originally created by Craig McClanahan and donated to the Apache Foundation in May, 2000. Formerly located under the Apache Jakarta Project and known as **Jakarta Struts**, it became a top-level Apache project in 2005.

The WebWork framework spun off from Apache Struts aiming to offer enhancements and refinements while retaining the same general architecture of the original Struts framework. However, it was announced in December 2005 that Struts would re-merge with WebWork. WebWork 2.2 has been adopted as Apache Struts 2, which reached its first full release in February 2007.

In a standard Java EE web application, the client will typically call to the server via a web form. The information is then either handed over to a Java Servlet which interacts with a database and produces an HTML-formatted response, or it is given to a JavaServer Pages (JSP) document that intermingles HTML and Java code to achieve the same result. Both approaches are often considered inadequate for large projects because they mix application logic with presentation and make maintenance difficult.

The goal of Struts is to separate the *model* (application logic that interacts with a database) from the *view* (HTML pages presented to the client) and the *controller* (instance that passes information between view and model). Struts provides the controller (a servlet known as ActionServlet) and facilitates the writing of templates for the view or presentation layer (typically in JSP, but XML/XSLT and Velocity are also supported). The web application programmer is responsible for writing the model code, and for creating a central configuration file struts-config.xml that binds together model, view, and controller.

Requests from the client are sent to the controller in the form of "Actions" defined in the configuration file; if the controller receives such a request it calls the corresponding Action class that interacts with the application-specific model code. The model code returns an "ActionForward", a string telling the controller what output page to send to the client. Information is passed between model and view in the form of special JavaBeans. A powerful custom tag library allows it to read and write the content of these beans from the presentation layer without the need for any embedded Java code.

Struts is categorized as a Model 2 request-based web application framework.

Struts also supports internationalization by web forms, and includes a template mechanism called "Tiles" that (for instance) allows the presentation layer to be composed from independent header, footer, menu navigation and content components.

**JAVA SERVER PAGE (JSP):-**

Architecturally, JSP may be viewed as a high-level abstraction of Java servlets. JSPs are translated into servlets at runtime; each JSP servlet is cached and re-used until the original JSP is modified.

JSP can be used independently or as the view component of a server-side model–view–controller design, normally with JavaBeans as the model and Java servlets (or a framework such as Apache Struts) as the controller. This is a type of Model 2 architecture.

JSP allows Java code and certain pre-defined actions to be interleaved with static web markup content, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, use Java bytecode rather than a native software format. Like any other Java program, they must be executed within a Java virtual machine (JVM) that integrates with the server's host operating system to provide an abstract platform-neutral environment.

JSPs are usually used to deliver HTML and XML documents, but through the use of OutputStream, they can deliver other types of data as well.

The Web container creates JSP implicit objects like pageContext, servletContext, session, request & response.

## Syntax

|  |  |
| --- | --- |
| http://upload.wikimedia.org/wikipedia/commons/thumb/d/df/Wikibooks-logo-en-noslogan.svg/40px-Wikibooks-logo-en-noslogan.svg.png | [***J2EE Programming/JavaServer Pages***](http://en.wikibooks.org/wiki/J2EE_Programming/JavaServer_Pages) |

JSP pages use several delimiters for scripting functions. The most basic is **<% ... %>**, which encloses a JSP *scriptlet.* A scriptlet is a fragment of Java code that is run when the user requests the page. Other common delimiters include **<%= ... %>** for *expressions,* where the scriptlet and delimiters are replaced with the result of evaluating the expression, and *directives*, denoted with **<%@ ... %>**.

Java code is not required to be complete or self-contained within its scriptlet element block, but can straddle markup content providing the page as a whole is syntactically correct. For example, any Java if/for/while blocks opened in one scriptlet element must be correctly closed in a later element for the page to successfully compile. Markup which falls inside a split block of code is subject to that code, so markup inside an *if* block will only appear in the output when the *if* condition evaluates to true; likewise, markup inside a loop construct may appear multiple times in the output depending upon how many times the loop body runs.

The following would be a valid for loop in a JSP page:

<**p**>Counting to three:</**p**>

<% for (int i=1; i<4; i++) { %>

<**p**>This number is <%= i %>.</**p**>

<% } %>

<**p**>OK.</**p**>

The output displayed in the user's web browser would be:

Counting to three:

This number is 1.

This number is 2.

This number is 3.

OK.

### Expression Language

Version 2.0 of the JSP specification added support for the Expression Language (EL), used to access data and functions in Java objects. In JSP 2.1, it was folded into the Unified Expression Language, which is also used in JavaServer Faces.

An example of EL syntax:

The value of "variable" in the object "javabean" is ${javabean.variable}.

### Additional tags

The JSP syntax adds additional tags, called JSP actions, to invoke built-in functionality. Additionally, the technology allows for the creation of custom JSP *tag libraries* that act as extensions to the standard JSP syntax. One such library is the JSTL, with support for common tasks such as iteration and conditionals (the equivalent of "for" and "if" statements in Java.)

## Compiler

A **JavaServer Pages compiler** is a program that parses JSPs, and transforms them into executable Java Servlets. A program of this type is usually embedded into theapplication server and run automatically the first time a JSP is accessed, but pages may also be precompiled for better performance, or compiled as a part of the build process to test for errors.

Some JSP containers support configuring how often the container checks JSP file timestamps to see whether the page has changed. Typically, this timestamp would be set to a short interval (perhaps seconds) during software development, and a longer interval (perhaps minutes, or even never) for a deployed Web application.

## Criticism

In 2000, Jason Hunter, author of "Java Servlet Programming", criticized JSP for either tempting or requiring the programmer to mix Java code and HTML markup, although he acknowledged it would "wean people off" of Microsoft's Active Server Pages. Later, he added a note to his site saying that JSP had improved since 2000, but also cited its competitors, Apache Velocity and Tea.

**HIBERNATE:-**

**Hibernate ORM** (Hibernate in short) is an object-relational mapping library for the Java language, providing a framework for mapping an object-oriented domain model to a traditional relational database. Hibernate solves object-relational impedance mismatch problems by replacing direct persistence-related database accesses with high-level object handling functions.

Hibernate is a free software that is distributed under the GNU Lesser General Public License.

Hibernate's primary feature is mapping from Java classes to database tables (and from Java data types to SQL data types). Hibernate also provides data query and retrieval facilities. It generates SQL calls and relieves the developer from manual result set handling and object conversion. Applications using Hibernate are portable to supported SQL databases with little performance overhead

## Mapping

Mapping Java classes to database tables is accomplished through the configuration of an XML file or by using Java Annotations. When using an XML file, Hibernate can generate skeleton source code for the persistence classes. This is unnecessary when annotations are used. Hibernate can use the XML file or the annotations to maintain the database schema.

Facilities to arrange one-to-many and many-to-many relationships between classes are provided. In addition to managing associations between objects, Hibernate can also manage reflexive associations where an object has a one-to-many relationship with other instances of its own type.

Hibernate supports the mapping of custom value types. This makes the following scenarios possible:

* Overriding the default SQL type that Hibernate chooses when mapping a column to a property.
* Mapping Java Enum to columns as if they were regular properties.
* Mapping a single property to multiple columns.

**Definition:** Objects in a front-end application follow OOP principles, while objects in the back-end follow database normalization principles, resulting in different representation requirements. This problem is called "object-relational impedance mismatch". Mapping is a way of resolving the impedance mismatch problem.

Mapping tells the ORM tool which java class object an application is needed to be store in which table of database.

## Hibernate Query Language (HQL)

Hibernate provides an SQL inspired language called Hibernate Query Language (HQL) which allows SQL-like queries to be written against Hibernate's data objects. *Criteria Queries* are provided as an object-oriented alternative to HQL. Criteria Query is used to modify the objects and provide the restriction for the objects.

## Persistence

Hibernate provides transparent persistence for Plain Old Java Objects (POJOs). The only strict requirement for a persistent class is a no-argument constructor, not necessarily *public*. Proper behavior in some applications also requires special attention to the *equals()* and *hashCode()* methods.

Collections of data objects are typically stored in Java collection objects such as Set and List. Java generics, introduced in Java 5, are supported. Hibernate can be configured to lazy load associated collections. Lazy loading is the default as of Hibernate 3.

Related objects can be configured to *cascade* operations from one to the other. For example, a parent Album object can be configured to cascade its save and/or delete operation to its child Track objects. This can reduce development time and ensure referential integrity. A *dirty checking* feature avoids unnecessary database write actions by performing SQL updates only on the modified fields of persistent objects.

## Integration

Hibernate can be used both in standalone Java applications and in Java EE applications using servlets, EJB session beans, and JBI service components. It can also be included as a feature in other programming languages. For example, Adobe integrated Hibernate into version 9 of ColdFusion (which runs on J2EE app servers) with an abstraction layer of new functions and syntax added into CFML.

## Entities and components

In Hibernate jargon, an *entity* is a stand-alone object in Hibernate's persistent mechanism which can be manipulated independently of other objects. In contrast, a *component* is subordinate to an entity and can be manipulated only with respect to that entity. For example, an Album object may represent an entity but the Tracks object associated with the Album objects would represent a *component* of the Album entity if it is assumed that Tracks can only be saved or retrieved from the database through the Album object. Unlike J2EE, it can switch databases.

**CONCLUSION**

The **“CABLE ACCESS SERVICE”** process made to reduce human efforts and to increase the efficiency. The main focus of this project is to lessen human efforts. The maintenance of the records is made efficient, as all the records are stored in the ACCESS database, through which data can be retrieved easily. The navigation control is provided in all the forms to navigate through the large amount of records. If the numbers of records are very large then user has to just type in the search string and user gets the results immediately. The editing is also made simpler. The user has to just type in the required field and press the update button to update the desired field.

The customers are given a particular unique id no.  So that they can be accessed correctly and without errors. Our main aim of the project is to get the correct information about a particular customer available in the database so that they can make online payment and select their desired channels on their own.

The problems, which existed in the earlier system, have been removed to a large extent. And it is expected that this project will go a long way in satisfying customers requirements. The computerization of the Cable Access Service will not only improves the efficiency but will also reduce human stress thereby indirectly improving human recourses.

**REFERENCES**

 www.free-css.com

 www.thenewboston.org

 www.roseindia.net

www.google.co.in

 Basics of JAVA by Herbert Schildt

 Wikipedia