Project Report on Library Record System in Java

Submitted to Panjab University, Chandigarh
In the Partial Fulfilment of
Bachelor of Computer Applications
Session 2012-13
Department of Computer Science and Applications

KAMLA LOHTIA SD COLLEGE
LUDHIANA

SUBMITTED TO:
MISS. DIVYA SANDAL
DEPARTMENT OF
COMPUTER SCIENCE

SUBMITTED BY:
VICTOR B. WILSON
BCA III
ROLL NO. 2909
PUPIN NO. 15109000227

Preface

If you put yourself in a position where you have to stretch outside your comfort zone, then you are forced to expand your consciousness.
- Les Brown

In every work there are competitions and challenges. Everybody wants to be successful in his/her life and reach the top of the world. Today the world has changed and every day new innovations take place. Application of Computers is the focus in every field. The Use of computers reduce the paper work and saves time by automating various tedious tasks.

Every day new user friendly software's are launched. We are trying to do the same here. The software that has been developed is related to Library Management**.** The objective of this software is to help myself become proficient in **Java** as well as step outside my comfort zone and developing something that is close to industry standards. It helped me in learning and expanding my knowledge of Java in a way that made me cross my self-defined limit. All features, commands & functions of Java are explained with the help of simple but practical examples. The Source Code for the application had been provide and an inbuilt HELP system has also been provided about all the options available in this software. You can make use of it whenever you get stuck.

Acknowledgement

I would like to take this opportunity to express my gratitude towards all the people who have in various ways, helped in the successful completion of my project.

I must convey my gratitude to **Miss. Divya Sandal (Teacher In charge)** for giving me the constant source of inspiration and help in preparing the project, personally correcting my work and providing encouragement throughout the project.

I also thank all my faculty members for steering me through the tough as well as easy phases of the project in a result oriented manner with concern attention.

Thanking You,

Victor B. Wilson

Introduction to Java

Java is a general purpose, object- oriented programming language developed by Sun Microsystems of USA in 1991. Originally called Oak by James Gosling, one of the inventors of the language, Java was designed for the development of software for consumer electronic device like TVs, VCRs, toasters and such other electronic machines.

This goal had a strong impact on the development team to make the language simple, portable and highly reliable .The Java team which included Patrick Naughton discovered that the existing languages like C and C++ had limitations in terms of both reliability and portability. However, they modeled there new Language Java on C and C++ but removed a number of feature of C and C++ that were considered as sources of problems and thus made Java a really simple, reliable, portable and powerful language.

The most striking feature of the language is that it is a platform-neutral language. Java is the first programming language that is not tied to any particular hardware or operating system. Programs developed in Java can be executed anywhere on any system. We can call Java as a revolutionary technology because it has brought in a fundamental shift in how we develop and use programs. Nothing like this has happened to the software industry before.

Features of Java

The Inventors of Java wanted to design a language, which could offer solutions to some of the problems encountered in modern programming. They wanted the language to be not only reliable, portable and distributed but also simple, compact and interactive. Sun Microsystems officially describe Java with following attributes:

- ✓ Compiled and interpreted
- ✓ Platform Independent and Portable
- ✓ Object-Oriented
- ✓ Robust and Secure
- ✓ Distributed
- ✓ Familiar, Simple and Small
- ✓ Multithreaded and Interactive
- ✓ High Performance
- ✓ Dynamic and Extensible

Although the above appears to be a list of buzzwords, they aptly describe the full potential of the language. These features have made Java the first application language of the World Wide Web. Java will also become the premier language for general-purpose stand –alone applications.

- **Compiled and Interpreted:** Usually a computer language is either compiled or interpreted. Java combines both these approaches thus making Java a two-stage system. First, Java compiler translates surface code into what is known as **byte code** instructions. Byte codes are not machine instructions and there for, in the second stage, Java interpreter generates machine code that can be directly executed by the machine that is running the Java program. We can thus say that Java is both a compiled and an interpreted language.

- **Platform-Independent & Portable:** The most significant contribution of Java over other languages is its portability. Java programs can be easily moved from one computer to another, anywhere and anytime changes and upgrades in operating system, processors and system resources will not force any changes in Java program. This is the reason why Java has become a popular language for programming on internet which interconnects different kinds of systems Worldwide. We can download Java applet from a remote computer on to our local system via Internet and execute it locally. Java ensures portability in two ways. First, Java compiler generates the byte code instructions that can be implemented on any machine. Secondly, the sizes of the primitive data types are machine independent.

- **Object-Oriented:** Java is true object-oriented language. Almost everything in Java is an object. All program code and data reside within objects and classes. Java comes with an extensive set of classes, arranged in packages that we can use in our programs by inheritance. The object model in Java is simple and easy to extend.

- **Robust and Secure:** Java is a robust language. It provides many time safeguards to ensure reliable code. It has strict compile time and run checking for data types. It is designed as a garbage-collected language relieving the programmers virtually all memory management problems. Java also incorporates the concept of exception handling, which captures series errors and eliminates any risk of crashing the system.

  Security becomes an important issue for a language that is used for programming on Internet. Threat of viruses and abuse of resources is everywhere. Java system not only verifies all memory access but also ensure that no viruses are communicated with an applet. The absence of pointers in Java ensures that programs cannot gain access to memory locations without proper authorization.

- **Distributed:** Java is designed as a distributed language for creating applications on networks. It has the ability to share both data and programs. Java applications can open and access remote object on Internet as easily as they can do in a local system. This enables multiple programmers at multiple remote locations to collaborate and work together on a single project.

- **Simple, Small and Familiar:** Java is a small and simple language. Many features of C and C++ that are either redundant or sources of unreliable code are not part of Java. For example, Java does not use pointers, preprocessor header files, go to statement and many others.

Familiarity is another striking feature of Java. To make the language look familiar to the exiting programmers, it was modelled on C and C++ languages. Java uses many constructs of C and C++ and therefore, Java code "looks like a C++" code. In fact, Java is a simplified version of C++.

- **Multithreaded and Interactive:** Multithreaded means handling multiple tasks simultaneously. Java supports multithreaded programs. This means that we need not wait for the application to finish one task before beginning another. For example, we can listen to an audio clip while scrolling the page and at the same time download an applet from a distant computer. This feature greatly improves the interactive performance of graphical applications. The Java runtime comes with tools that support multiprocessing synchronization and construct smoothly running interactive system.

- **High Performance:** Java performance is impressive for an interpreted language, mainly due to the use of intermediate byte code. According to sun, Java speed is comparable to the native C/C++. Java architecture is also designed to reduce overheads during runtime. Further, the incorporation of multithreading enhances the overall execution speed of Java programs.

- **Dynamic and Extensible:** Java is a dynamic language. Java is a capable of dynamically linking in new class libraries, methods, and object. Java can also determine the type of class through a query, making it possible to either dynamically link or abort the program, depending on response.

Java programs support functions written in other languages such as C and C++. These functions are known as native methods. This facility enables the programmers to use the efficient functions available in these languages. Native methods are linked dynamically at runtime.

Object Oriented Paradigm

The major objective of the object oriented programming is to eliminate some of the flaws encountered in the procedural approach. OOP ties data more closely to the functions that operate on it and protect it from the unintentional modification by other functions. OOP allows us to decompose a problem into a number of entities called objects and then build data and functions around these entities. The combination of data and methods make up an object.

The data of an object can be accessed only by the methods associated with that object. However, methods of one object can access the methods of other objects. Some of the features of object-oriented paradigm are:

- ✓ Emphasis is on data rather than procedure.
- ✓ Programs are divided into what are known as Objects.
- ✓ Data structures are designed such that they characterize the objects.
- ✓ Method that operates on data of an object is tied together in the data structure.
- ✓ Data is hidden and cannot be accessed by external functions.
- ✓ Objects may communicate with each other through methods.
- ✓ New data and methods can be easily added whenever necessary.

An object is considered to be a partitioned area of computer memory that can store data and set of operations that can access that data.

Basic Concepts of OOP

We shall now discuss the general concepts of OOP which form the heart of Java language.

Object and Classes: Objects are the basic run time entities in an object-oriented system. They may represent a person, a place, a bank account, a table of data or any item that the program may handle. They may also represent user-defined data types such as vectors and lists. Any programming problem is analyzed in terms of objects and the nature of communication between them. Program object should be chosen such that they match closely the real world objects.

When a program is executed, the objects interact by sending message to one another. For example customer and account are two objects in a banking program, then the customer object may send a message to the account object requesting for the balance. Each object contains data and code to manipulate the data. Objects can interact without having to know the details of each other's data or code.

We just mentioned that objects contain data and code to manipulate that data. The entire set of data of an object can be made a user defined data type using the concepts of a class. A class may be thought of as a 'data type' and object as a 'variable' of that data type. Once a class has been defined, we can create any number of objects belonging to that class. Each object is associated with data of type class with which they are created. A class is thus a collection of object of similar types. For example, mango, apple and orange are members of the class fruit.

Classes are user-defined data types and behave like the built in types of a programming language. For example, the syntax used to create an object is not different than the syntax used to create an integer object in C. If fruit has been defined as a class, then the statement

```
Fruit mango;
```

will create an object **mango** belonging to the class **fruit**.

Data Abstraction and Encapsulation: The wrapping up of data and methods into a single unit is known as encapsulation. Data encapsulation is most striking feature of a class. The data is not accessible to the outside world and only those methods, which are wrapped in the class, can access it. These methods provide the interface between the object's data and the program. This insulation of data from direct access by the program is called data hiding. Encapsulation makes it possible for objects to be treated like 'black boxes', each performing a specific task without any concern for internal implementation.

Abstraction refers to the act of representing essential features without including the background details or explanations.

Dynamic Binding: Binding refers to the linking of a procedure call to the code to be executed in response to the call. Dynamic binding means that the code associated with a given procedure call is not known until the time of the call at runtime. It is associated with a polymorphic reference depends on the dynamic type of that reference.

Consider the procedure “draw” in fig. By inheritance, every object will have this procedure. Its algorithm is, however, unique to each object and so the draw procedure will be redefined in each class that defines the object. At run time, the code matching the object under current reference will be called.

Polymorphism: Polymorphism is another important OOP. Polymorphism means the ability to take more than one form. For example, an operation may exhibit different behavior in different instances. Figure below illustrate that a single function name can be used to handle different number and different type of arguments. This is something similar to a particular word having several different meanings depend on the context.

Polymorphism plays an important role in allowing objects having different internal structures to share the same external interface. This means that a general class of operation may be accessed in the same manner even through specific actions associated with each operation may differ. Polymorphism is extensively used in implementing inheritance.

Message Communication: All object-oriented program consists of a set of objects that communicate with each other. The process of programming in an object-oriented language, therefore, involves the basic steps:

1. Creating classes that defines objects and their behavior.
2. Creating objects from class definitions.
3. Establishing communication among objects.

Objects communicate with one another by sending and receiving information much the same way as people pass message to one another as shown in fig below. The concepts of message passing make it easier to talk about building systems that directly model or simulates their real-world counterpart.

A message for an object is a request for execution of a procedure, and therefore will invoke a method in the receiving objects that generates the desired results.

For example consider the statement

```
Empolyee.salary(name);
```

Here, Employee is the object, salary is the message and name is the parameter that contains the information.

Inheritance: Inheritance is the process by which the object of one class acquires the properties of objects of another class. Inheritance supports the concept of hierarchical classification. For example bird robin is a part of the class flaying bird, which is again a part of class bird. As illustrated in the fig below, the principal behind this sort of division is that each derived class shares common characteristics with the class from which it is derived.

In OOP, the concept of inheritance provides the idea of reusability. This means that we can add additional features to an existing class without modifying it. This is possible by deriving a new class from the existing one. This new class will have the combined features of both the classes. In Java, the derived class is known as ‘subclasses’.

Thus overall Java is an object oriented language. It enables us not only to organize our program code into logical units but also to take advantage of encapsulation, inheritance, and polymorphism.