# Independent Study Report

# Java Programming
# Project: Student Registration System

**Hari Krishna Vardhan Yerramsetty**
**Email: hyerram1@binghamton.edu**

This report summarizes an independent study on Java web programming by Hari Krishna Vardhan Yerramsetty , advised by Prof. Leslie Lander.

# What I have learned from this Independent Study.

**Introduction:**

**Java** is a general-purpose 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 compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2015, Java is 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 been acquired by 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.

The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (byte code compiler), GNU Class path(standard libraries), and IcedTea-Web (browser plugin for applets).

The latest version is Java 8, which is the only version currently supported for free by Oracle, although earlier versions are supported both by Oracle and other companies on a commercial basis.

**Syntax:**

The syntax of Java is largely influenced by C++. Unlike C++, which combines the syntax for structured, generic, and object-oriented programming, Java was built almost exclusively as an object-oriented language. All code is written inside classes, and every data item is an object, with the exception of the primitive data types, *i.e.* integers, floating-point numbers, Boolean values, and characters, which are not objects for performance reasons. Java reuses some popular aspects of C++ (such as printf() method).

Unlike C++, Java does not support operator overloading or multiple inheritance for *classes*, though multiple inheritance is supported for interfaces. This simplifies the language and aids in preventing potential errors and anti-pattern design.

Java uses comments similar to those of C++. There are three different styles of comments: a single line style marked with two slashes (`//`), a multiple line style opened with `/*` and closed with `*/`, and the Javadoc commenting style opened with `/**` and closed with `*/`. The Javadoc style of commenting allows the user to run the Javadoc executable to create documentation for the program.

**SQL** is a special-purpose programming language designed for managing data held in a relational database management system (RDBMS), or for stream processing in a relational data stream management system (RDSMS).

Originally based upon relational algebra and tuple relational calculus, SQL consists of a data definition language, data manipulation language, and a data control language. The scope of SQL includes data insert, query, update and delete, schema creation and modification, and data access control. Although SQL is often described as, and to a great extent is, a declarative language (4GL), it also includes procedural elements.

SQL was one of the first commercial languages for Edgar F. Codd's relational model, as described in his influential 1970 paper, "A Relational Model of Data for Large Shared Data Banks. Despite not entirely adhering to the relational model as described by Codd, it became the most widely used database language.

SQL became a standard of the American National Standards Institute (ANSI) in 1986, and of the International Organization for Standardization (ISO) in 1987. Since then, the standard has been revised to include a larger set of features. Despite the existence of such standards, though, most SQL code is not completely portable among different database systems without adjustments.

The most common operation in SQL, the query, makes use of the declarative `SELECT` statement. `SELECT` retrieves data from one or more tables, or expressions. Standard `SELECT` statements have no persistent effects on the database. Some non-standard implementations of `SELECT` can have persistent effects, such as the `SELECT INTO` syntax provided in some databases.

Queries allow the user to describe desired data, leaving the database management system (DBMS) to carry out planning, optimizing, and performing the physical operations necessary to produce that result as it chooses.

A query includes a list of columns to include in the final result, normally immediately following the `SELECT` keyword. An asterisk ("`*`") can be used to specify that the query

should return all columns of the queried tables. SELECT is the most complex statement in SQL, with optional keywords and clauses that include:

- The FROM clause, which indicates the table(s) to retrieve data from. The FROM clause can include optional JOIN subclauses to specify the rules for joining tables.
- The WHERE clause includes a comparison predicate, which restricts the rows returned by the query. The WHERE clause eliminates all rows from the result set where the comparison predicate does not evaluate to True.
- The GROUP BY clause projects rows having common values into a smaller set of rows. GROUP BY is often used in conjunction with SQL aggregation functions or to eliminate duplicate rows from a result set. The WHERE clause is applied before the GROUP BY clause.
- The HAVING clause includes a predicate used to filter rows resulting from the GROUP BY clause. Because it acts on the results of the GROUP BY clause, aggregation functions can be used in the HAVING clause predicate.
- The ORDER BY clause identifies which column[s] to use to sort the resulting data, and in which direction to sort them (ascending or descending). Without an ORDER BY clause, the order of rows returned by an SQL query is undefined.
- The DISTINCT keyword eliminates duplicate data.

**PL/SQL** (**Procedural Language/Structured Query Language**) is Oracle Corporation's procedural extension for SQL and the Oracle relational database. PL/SQL is available in Oracle Database (since version 7), Times Ten in-memory database (since version 11.2.1), and IBM DB2 (since version 9.7). Oracle Corporation usually extends PL/SQL functionality with each successive release of the Oracle Database.

PL/SQL includes procedural language elements such as conditions and loops. It allows declaration of constants and variables, procedures and functions, types and variables of those types, and triggers. It can handle exceptions (runtime errors). Arrays are supported involving the use of PL/SQL collections. Implementations from version 8 of Oracle Database onwards have included features associated with object-orientation. One can create PL/SQL units such as procedures, functions, packages, types, and triggers, which are stored in the database for reuse by applications that use any of the Oracle Database programmatic interfaces.

A PL/SQL program unit is one of the following: PL/SQL anonymous block, procedure, function, package specification, package body, trigger, type

specification, type body, library. Program units are the PL/SQL source code that is compiled, developed and ultimately executed on the database.

## PL/SQL anonymous block

The basic unit of a PL/SQL source program is the block, which groups together related declarations and statements. A PL/SQL block is defined by the keywords DECLARE, BEGIN, EXCEPTION, and END. These keywords divide the block into a declarative part, an executable part, and an exception-handling part. The declaration section is optional and may be used to define and initialize constants and variables. If a variable is not initialized then it defaults to NULL value. The optional exception-handling part is used to handle run time errors. Only the executable part is required. A block can have a label.

### Procedure

Procedures are similar to functions, in that they are named program units that can be invoked repeatedly. The primary difference is that **functions can be used in a SQL statement whereas procedures cannot**. Another difference is that the procedure can return multiple values whereas a function should only return a single value.

The procedure begins with a mandatory heading part to hold the procedure name and optionally the procedure parameter list. Next are the declarative, executable and exception-handling parts, as in the PL/SQL Anonymous Block.

### Package

Packages are groups of conceptually linked functions, procedures, variables, PL/SQL table and record TYPE statements, constants, cursors etc. The use of packages promotes re-use of code. Packages are composed of the package specification and an optional package body. The specification is the interface to the application; it declares the types, variables, constants, exceptions, cursors, and subprograms available. The body fully defines cursors and subprograms, and so implements the specification. Two advantages of packages are:

1. Modular approach, encapsulation/hiding of business logic, security, performance improvement, re-usability. They support object-oriented programmingfeatures like function overloading and encapsulation.
2. Using package variables one can declare session level (scoped) variables, since variables declared in the package specification have a session scope.

**Trigger**

A database trigger is like a stored procedure that Oracle Database invokes automatically whenever a specified event occurs. It is a named PL/SQL unit that is stored in the database and can be invoked repeatedly. Unlike a stored procedure, you can enable and disable a trigger, but you cannot explicitly invoke it. While a trigger is enabled, the database automatically invokes it—that is, the trigger fires—whenever its triggering event occurs. While a trigger is disabled, it does not fire.

You create a trigger with the CREATE TRIGGER statement. You specify the triggering event in terms of triggering statements, and the item they act on. The trigger is said to be created on or defined on the item—which is either a table, a view, a schema, or the database. You also specify the timing point, which determines whether the trigger fires before or after the triggering statement runs and whether it fires for each row that the triggering statement affects.

If the trigger is created on a table or view, then the triggering event is composed of DML statements, and the trigger is called a DML trigger. If the trigger is created on a schema or the database, then the triggering event is composed of either DDL or database operation statements, and the trigger is called a system trigger.

An INSTEAD OF trigger is either: A DML trigger created on a view or a system trigger defined on a CREATE statement. The database fires the INSTEAD OF trigger instead of running the triggering statement.

**Purpose of triggers**

Triggers can be written for the following purposes:

- Generating some derived column values automatically
- Enforcing referential integrity
- Event logging and storing information on table access
- Auditing
- Synchronous replication of tables
- Imposing security authorizations
- Preventing invalid transactions