Name: Lamar Myles

Email: lamar3571@gmail.com

Tel: 631-569-9156

Download and install Java 8 from:

http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

Download and install Eclipse from:

http://www.eclipse.org/downloads/index.php

Create Github account from:

https://github.com

Java is a high level programming language

Why Java ?

Java is somewhat different Java has a principle, "write once, run anywhere".

What does that mean?

Platform independence for compiled Java code How?

The Java Virtual Machine

Java programs are compiled into Java bytecode bytecode is executed by the Java Virtual Machine (JVM)

Java Virtual Machine

A program that runs Java programs and manages memory for Java programs.

Why?

Each platform is different (Mac/PC/Linux/etc.)

Java can be used to develop Web applications.

Java Applets

Java can also be used to develop applications for handheld devices such as cell phones
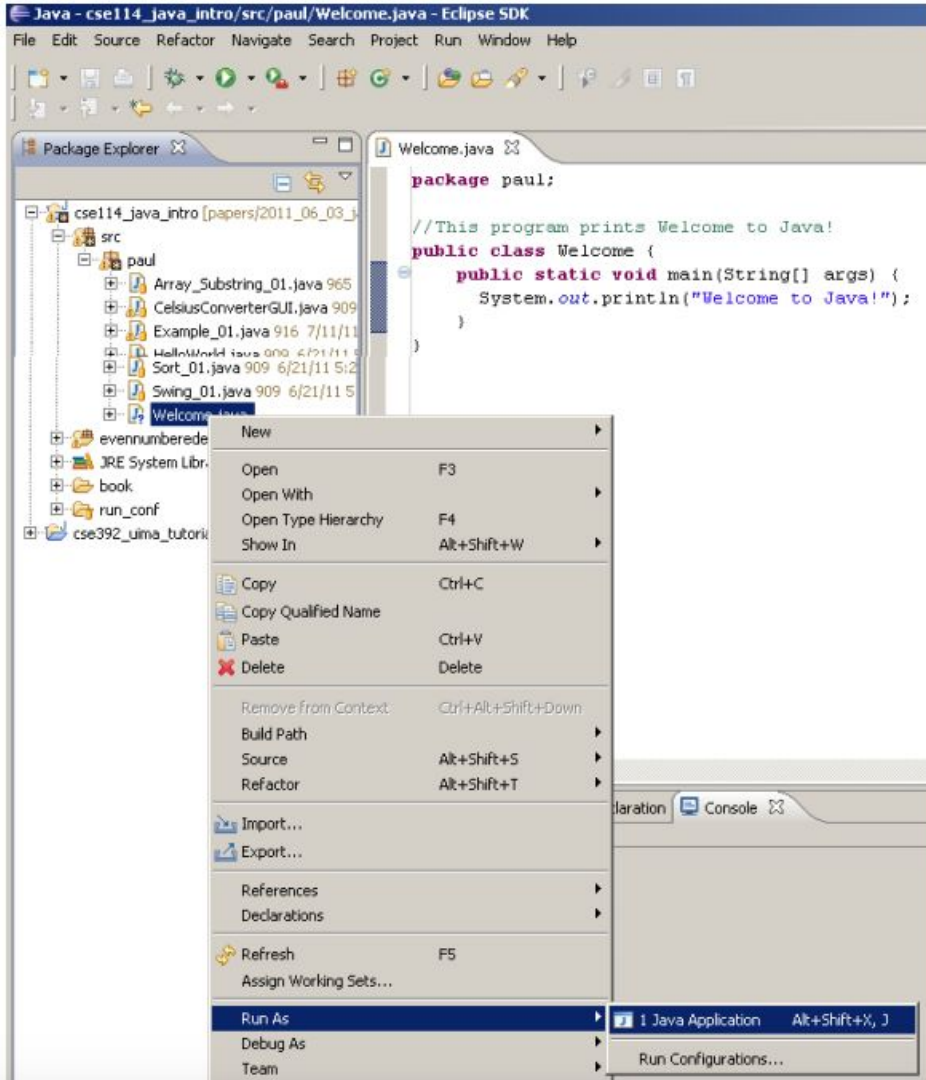
A Simple Java Program

```java
//This program prints Welcome to Java!
public class Welcome {
    public static void main(String[] args) {
        System.out.println("Welcome to Java!");
    }
}
```

Running Programs in Eclipse

Anatomy of a Java Program

Comments

Statements

Classes

The main method

Reserved words

Modifiers

Blocks

Methods

claim.Academy
(st. louis)

Comments

Line comment: A line comment is preceded by two slashes (//) in a line.

Paragraph comment: A paragraph comment is enclosed between /* and */ in one or multiple lines.

javadoc comment: javadoc comments begin with /** and end with */. They are used for documenting classes, data, and methods. They can be extracted into an HTML file using JDK's javadoc command.

Reserved Words

Reserved words or keywords are words that have a specific meaning to the compiler

Cannot be used for other purposes in the program

Example: class

the word after class is the name for the class

Java Keywords

Abstract,assert,boolean,break,byte,case,catch,char,class,const, continue ,default,do,double,else,enum,extends,false,final,finally, float,for,goto, if,implements,import,instanceof,int,interface,long, native,new,null, package,private,protected,public,return,short, static,strictfp,super, switch,synchronized,this,throw,throws, transient,true,try,void, volatile,while.

http://docs.oracle.com/javase/tutorial/java/nutsandbolts/_keywords.html

Modifiers

Java uses certain reserved words called modifiers that specify the properties of the data, methods, and classes and how they can be used

Examples: public, static, private, final, abstract, protected

A public datum, method, or class can be accessed by other programs

A private datum or method cannot be accessed by other programs

Statements

A statement represents an action or a sequence of actions

System.out.println("Welcome to Java!");

is a statement to display the greeting "Welcome to Java!"

Every statement in Java ends with a semicolon (;)

Blocks

A pair of braces in a program forms a block that groups components of a program.

```
public class Test {
    public static void main(String[] args) {
        System.out.println("Welcome to Java!");  Method block
    }
}
```

Class block

Rules for Identifiers

Should contain only letters, numbers, & '_'

'$' is allowed, but only for special use

Cannot begin with a digit!

Uppercase and lowercase letters are considered to be different characters

Examples:

Legal: myVariable, my_class, my4Var

Illegal: 4myVariable, my class, my!Var, @#$myClass

Common Java Naming Conventions

Variables & Methods start with lower case letters: x,

toString Classes start with upper case letters: Person

Variables and Class identifiers should generally be nouns

Method identifiers should be verbs

Use Camel notation: myVariable, MyClass

Although it is legal, do not begin with '_' (underscore).

Use descriptive names: LinkedList, compareTo

Programming Errors

Syntax Errors:

Detected by the compiler

Runtime Errors:

Causes the program to abort

Logic Errors:

Produces incorrect result

Syntax Error

```
public class ShowSyntaxError {

    public static void main(String[] args) {

    i = 30; // Detected by the compiler

    System.out.println(i + 4);

    }

    }
```

Runtime Error

```java
public class ShowSyntaxError {

    public static void main(String[] args) {

    int i = 1 / 0; // Division with 0

    }

    }
```

Logic Errors

```java
public class ShowLogicError {
 // Determine if a number is between 1 and 100 inclusively
public static void main(String[] args) {
Scanner input = new Scanner(System.in);
int number = input.nextInt(); // Display the result
System.out.println( "The number is between 1 and 100, inclusively: " + ((1 < number) && (number < 100)) );
// Wrong result if the entered number is 1 or 100
System.exit(0); } }
```