Programming with Java

Java-At-A-Glance

- ✓ Widely used, high-level programming language
- ✓ Developed by Sun Microsystems in 1995 (which was acquired by Oracle Corporation in 2010)
- ✓ An object-oriented programming language (OOP)
- ✓ The OOP approach divides programming tasks into modules called **classes**. Each class is a collection of data and related instructions for performing actions on the data
- ✓ Programs written in Java are portable (programs must run similarly on any hardware/operating-system platform)

Compilers & Interpreters

The difference between the way Java and other programming languages worked was revolutionary. Code in other languages is first translated by a ***compiler*** *into instructions for a specific type of computer. The Java compiler instead turns code into something called* ***Bytecode****, which is then interpreted by software called the* ***Java Runtime Environment (JRE),*** *or the* ***Java virtual machine (VM)****. The JRE acts as a virtual computer that interprets Bytecode and translates it for the host computer. Because of this, Java code can be written the same way for many platforms ("write once, run anywhere"), which helped lead to its popularity for use on the Internet, where many different types of computers may retrieve the same Web page.* (http://www.britannica.com)

A Java Application

```
public class HelloWorld {
      public static void main (String args[]){
            System.out.println ("Hello World");
      }
}
```

Some Essential Java Vocabulary

A Java Application is a _____ with at least one _____

that contains a _____ method.

A **package** is _____

A **class** is _____

A **method** is _____

An **object** is _____

Example 1

```
//    Example1.Java
//    Displays a welcome message
public class Example1 {
    public static void main(String[] args) {
        System.out.println ("         Welcome To");
        System.out.println ("Dundas Valley Secondary School");
        System.out.println ("==============================");

        System.out.print ("\tPhone \t\t\tFax\n");
        System.out.print ("\t905.628.2203\t905.627.2904\n\n");
    }
}
```

Output →

```
         Welcome To
Dundas Valley Secondary School
==============================
    Phone           Fax
    905.628.2203    905.627.2904
```

Displaying Output

An ***output stream*** is a data channel to the operating system, which redirects the bytes of data to a specific hardware device. Java is preconfigured with several standard streams:

- _____ (_____)
- _____ (_____)
- _____ (_____)

Standard output is usually handled by these Java statements

- _____
- _____
- _____

Example 2

```
public class Example2 {
  public static void main(String[] args) {
      String  partDescription;
      int     partNumber;

      System.out.println("\n\t\tAcme Warehouse Inventory");
      System.out.print  ("\t\t==== ========= =========\n\n");

      partDescription = "Inkjet Printer";
      partNumber = 1052;

      System.out.print ("\tPart\n");
      System.out.print ("\tNumber\t\t\tDescription\n");
      System.out.print ("\t------\t\t\t-----------\n");

      System.out.println ("\t" + partNumber + "\t\t\t" + partDescription );

      partDescription = "DVD Burner";
      partNumber = 2199;
      System.out.println ("\t" + partNumber + "\t\t\t" + partDescription );
   }
}
```

Output →

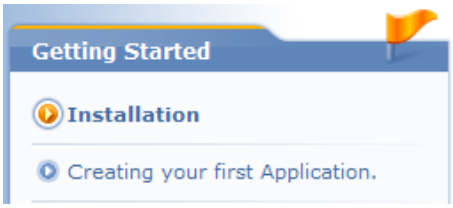
```
                Acme Warehouse Inventory
                ==== ========= =========

        Part
        Number                  Description
        ------                  -----------
        1052                    Inkjet Printer
        2199                    DVD Burner
```

Java Learning Activities

*Name:*_____

Activity # 1



- Start **JCreator**
- Click **Creating your first Application**
- Find the meaning of these terms:

a) What is a **project**?

_____

b) What is a **project workspace**?

_____

c) Scroll down to ***Creating a New Project*** and summarize the steps here:

1. _____
2. _____
3. _____
4. _____
5. _____
6. _____

d) In JCreator follow these steps to create a new project called **Example3**.

e) Type this program into the `Example3.java` file:

```
public class Example3 {
    public static void main(String[] args) {
        System.out.println("My Favorite Movie");
        System.out.println ("\n\n\t\tCasablanca\n\n");
    }
}
```

f) Replace **`Casablanca`** with your favorite movie.

g) Compile your program: **Build → Compile Project** NOTE: your file is automatically saved

h) Run your program: **Build → Execute Project**
The result is displayed in the Build Output window.

i) Show your first Java program to the teacher:

Activity # 2

1. Close the Workspace and all document windows from Activity #1 (File → Close Workspace)
2. Create a new project and file called **Example2**.
3. Type this program

```
//  Program Example #4
//  Description: The "Hello World" program expanded
//   by <your name>
public class Example4 {
  public static void main (String[] args) {
      // press the spacebar about 20 times to center the title
    System.out.println ("\n\n                    Hello World!");
    System.out.println ("");
    System.out.print ("\t\t\t\t programmed by <your name>");
    System.out.print ("\n\n");
  } // end main() method
} // end Example2 class
```

4. Compile your project with the F7 key

5. What does the **//** do at the beginning of a line?

   _____

6. How does the **\n** change the output?

   _____

7. How does the **\t** change the output?

   _____

8. Show your second Java program to the teacher:

Activity # 3—Reading Review

Read Chapter 3: Introducing Java of the textbook, A Guide to Programming in Java, pages 59 to 63.

Find the answers to these questions:

a) What is an **OOPL**?

_____

b) What are the 3 features of every object-oriented language?

_____
_____
_____

c) Why are Java programs platform-independent applications? (HINT: see the end of chapter glossary)

_____
_____

d) OOPL development involves objects. What does an object consist of?

_____
_____
_____

e) The design for an object is called a _____.

f) What does a class define?

_____

g) What is a package (also called a _____)?

_____
_____

h) Packages are importable, which means …

_____
_____

Coding Conventions

Code conventions are a set of guidelines for that govern the style of your program code. They cover such areas as

- standard practices for writing comments.
- customary methods for naming variables and other identifiers
- statement formatting
- and more

Sun Microsystems has published a document recommending certain standard ways to code Java programs. Here is an excerpt that explains why they think this is important.

1.1 Why Have Code Conventions

Code conventions are important to programmers for a number of reasons:

- 80% of the lifetime cost of a piece of software goes to maintenance.
- Hardly any software is maintained for its whole life by the original author.
- Code conventions improve the readability of the software, allowing engineers to understand new code more quickly and thoroughly.
- If you ship your source code as a product, you need to make sure it is as well packaged and clean as any other product you create.

For the conventions to work, every person writing software must conform to the code conventions. Everyone.

For the full document, visit → http://java.sun.com/docs/codeconv/

The whole document is twenty pages long. Here are a few conventions to start with:

✓ _____

✓ _____

✓ _____

✓ _____

✓ _____

✓ _____

✓ _____

✓ _____

✓ _____

✓

Introduction to Java and JCreator Pro

You can use any simple text editor like Windows Notepad to type in a Java program and then compile it with Oracle's compiler (javac), but with this method there are no programming tools or help files to make the job easier. Programmers often use an "all-in-one" editor for ***typing***, ***editing***, ***debugging*** and ***running*** their own programs.

JCreator is an application that runs on top of the Java JDK (or Java Development Kit) from Oracle and helps the student develop Java programs. It is a program that lets you learn about Java programs and create your own programs. Here is a description of JCreator from the people who designed it:

> *JCreator is a powerful IDE for Java™ technologies that provides more power at your fingertips than all the ordinary IDEs combined.*

Features

- It is a ***IDE***: an ***Integrated Development Environment***

  This means that all the tasks of writing programs are ***integrated*** (built-in) into a single ***environment*** (application) for ***developing*** programs.

  An IDE has

  ① A **text editor** for typing your Java programs (***source code***)

  ② A **compiler** for translating the Java source code into a class file (***byte code***) which runs on any computer platform (i.e. any type of computer) with the Java ***VM*** (virtual machine) installed and

  ③ A **debugger** to help find and fix errors in your Java source code.

Download & Install Java from Oracle: (current version: *8u11*)

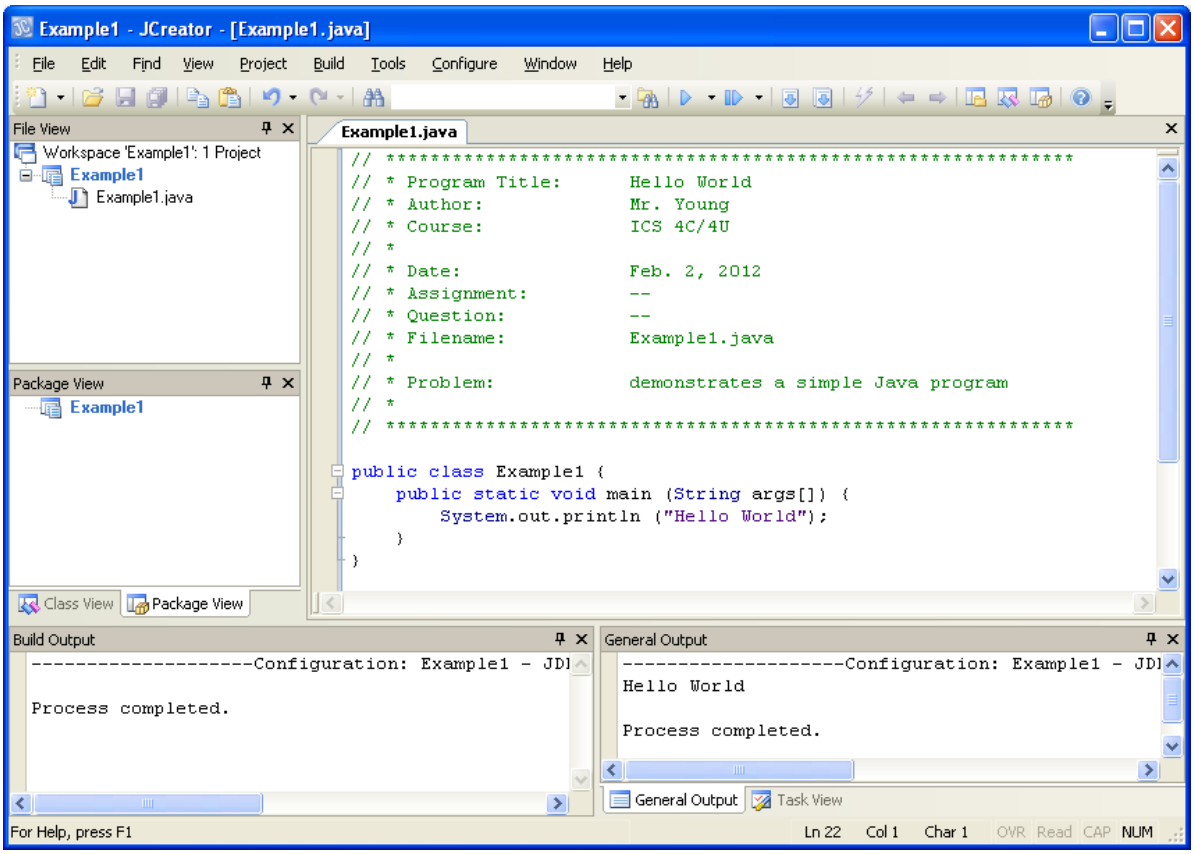
http://www.oracle.com/technetwork/java/javase/downloads/

Install JCreator

http://www.jcreator.com/

JCreator is the development tool for every programmer that likes to do what he does best: programming. It is faster, more efficient and more reliable than other IDE's. Therefore it is the perfect tool for programmers of every level, from learning programmer to Java-specialist.

Here is the JCreator window:



Workspace or Main Window

- The workspace, or main window, includes a three-pane window. The top left window displays the ***File View*** and beneath is the ***Data View*** or ***Package View.***
- The top right window is the ***code editor*** window. You enter and modify Java and Html code in this window. The document window can be maximized, or toggled to full-screen. You can also navigate through the documents by using the documents tabs.
- The bottom pane of the window, the Output view, displays several tabs which allow you to view the following items:
  - **General Output** displays the general output from the Java application and general tools. You can interact with the running process by typing commands in this view.
  - **Build Output** displays the compiler errors.
  - **Task List** displays a task list in the bottom pane of the window.
  - **Debug Output** displays the output of debug operations

Customizing the Workspace

You can organize your workspace by dragging the views and placing them where you want. If you need to save space on the desktop, click the auto-hide pin in the top-right corner of each view.

Use ***File View*** to manage the contents of a project workspace by adding, moving, and deleting subprojects and reference files. The project workspace can contain several projects, and each project can consist of any number of folders and files. Organize your project by dragging and dropping folders and files within the same project.

You probably recognize and understand the tree structure of File View. File View displays the contents of a project workspace starting with its root structure and expanding to list subprojects and other items. You click the plus sign to expand an item, and you click the minus sign to deflate an item. The following image shows a partially collapsed project called Manual that contains one project.

The Editor

JCreator's multiple document interface, document selector tabs, and file view make it easy to move from one file to another. Familiar features and common commands increase productivity because of a lower learning curve. These features and commands include dragging and dropping files from Windows Explorer to the JCreator editor, Clipboard support, and Find, Undo and Redo commands.

```
    public String(char value[], int offset, int count) {
        if (offset < 0) {
            throw new StringIndexOutOfBoundsException(offset);
        }
        if (count < 0) {
            throw new StringIndexOutOfBoundsException(count);
        }
        // Note: offset or count might be near -1>>>1.
        if (offset > value.length - count) {
            throw new StringIndexOutOfBoundsException(offset + count);
        }
        char[] v = new char[count];
        System.arraycopy(value, offset, v, 0, count);
        this.offset = 0;
        this.count = count;
        this.value = v;
    }
```

In addition, the auto-completion method makes entering code faster and more accurate, by ensuring that casing and spelling are correct. The code editor uses the document profile to color-code various text blocks. You can modify these profiles and change the color settings in the Options menu. In the freeware version of JCreator, the code editor uses only three types of document profiles: Java, HTML, and plain text.

Chapter 3 Java Vocabulary

Review these terms from chapter 3 and ensure that you understand each one. Organize them into the categories in the table below.

1. Algorithm
2. Argument
3. Bytecode
4. Class
5. Code conventions
6. Comment
7. Comment block
8. Compiling
9. Controlling class
10. Encapsulation
11. Escape Sequence
12. Execute
13. Flowchart
14. Importable
15. Inheritance
16. Interpreter
17. Java application
18. Java Virtual Machine (Java VM)
19. Just-in-time compiler (JIT)
20. Library
21. Machine code
22. Method
23. Object
24. Output Stream
25. Package
26. Platform-independent application
27. Polymorphism
28. Pseudocode
29. Run
30. Source code
31. Statement
32. String
33. Syntax error

Problem Solving	OOP	Java Language	Programming Fundamentals

=