### 

### Java Lectures

These lecture notes should be printed and you should add your own notes in the space provided.

They are intentionally incomplete so that you take the time to write down the missing bits in accordance with the results of [this study](http://pss.sagepub.com/content/early/2014/04/22/0956797614524581.abstract)[…](http://www.google.com/url?q=http%3A%2F%2Fpss.sagepub.com%2Fcontent%2Fearly%2F2014%2F04%2F22%2F0956797614524581.abstract&sa=D&sntz=1&usg=AFQjCNFezZtxcp66twRpSrcz1Q8A2oBwFA) (<http://pss.sagepub.com/content/early/2014/04/22/0956797614524581.abstract>)

|  |
| --- |
| Objectives: Be able to write a minimal java program and discuss various introductory topics including   1. "the write once run anywhere" philosophy of java.     1. discuss traditional compiled languages, machine language, the concept of a platform, the jvm, bytecodes    2. be able to draw a diagram explaining how the jvm solves the "platform" issue 2. the structure of a minimal java program 3. class vs object 4. references 5. the "main" entry point  Introduction  * how does a traditional compiled language work?   + Compile/Link/Run. * What is machine language and how many different machines are there?   What is the platform problem?    Write Once run Anywhere (solves the platform problem)...    What happens when you compile a java program   * Compile :   javac MyJavaProgram.java  // creates a class file containing java byte codes which are machine language for the // java virtual machine (jvm) * Produces :   MyJavaProgram.class   // note, there's no linking step * Run it :   java MyJavaProgram   Note on IDE’s\*\*   * Managing a development environment, including class paths, libraries, tools paths, etc... can be a night mare. * A Programmers Integrated Development Environment (Like netbeans, IntelliJ or eclipse) takes care of the "environment” for us, provides us with a nice gui and a set of tools like refactoring. * Although Intelli J? (eclipse) has more or less won the IDE war so to speak (yes it’s debatable, good luck with that flame war) we will use the netbeans IDE. I'll discuss this in class. * We will use IntelliJ/eclipse at a later date in this and other courses.   Writing your first java program   * Objectives:   + write a minimal java program.   + discuss the structure of a java program * java programs are based on the "class" construct   NOTE \*\* a class name begins with a Capital letter. (This is by convention only. There's nothing stopping you from using lowercase, just don't do it please)   public class <NameOfTheClass> {  }   * in order for a java class to execute it must have an entry point, the place that it goes to when you first run it. * a program's entry point is called "main" and it must have the following signature (ie it must look like this)   public static void main(String[] args) {  // your program starts here  }   * the main method is *static*, returns nothing (void) and has an array of Strings passed to it (a String array argument called "args").   + the main method may be passed an array of String arguments.   + if you run the program from the command line like this:   java -jar NameOfTheClass a b c   * + then args[]=a b c * a complete minimal program looks like this. It compiles and runs but does nothing.   package lecture1;  public class MinimalJavaProgram\_1 {  public static void main(String[] args) {  }  }   * the class must be in a file with the same name of the class so in our example, the file name is MinimalJavaProgram\_1.java.   Packages  **DEFINITION "package"** : a package contains a set of related classes.  (<http://docs.oracle.com/javase/tutorial/java/concepts/package.html>)   * a package is represented on disk as a folder   (<PROJECT\_FOLDER>\src\org\csd211\ MinimalJavaProgram\_1.java)   Objects   * + **TODO STUDENT READING** : <http://docs.oracle.com/javase/tutorial/java/concepts/object.html>   + an Object is an instance of a class.   + discuss references, classes, objects, null reference, the garbage collector.   + you can't do anything with a class, it’s just a "blueprint" for a real object.   + a class must be instantiated in order for it to be useful.     - an object is instantiated through the use of the "new" operator   String name = new String("Fred Carella");   * + a blueprint is to a house as a class is to an object.  TODO TEACHER :  go over MinimalJavaProgramFullyDocumented\_1.java and MinimalJavaProgramFullyDocumented\_2.javaTODO TEACHER : go over ...  * how to create a class from scratch in Netbeans. * compile it * run it * show some errors. * print HelloWorld.    TODO STUDENT IN CLASS EXERCISE : write a minimal java program  * using a pencil, write a minimal java program.  TODO STUDENT IN CLASS EXERCISE :  1. Using the Netbeans IDE, write your own minimal java program based on MinimalJavaProgramFullyDocumented\_1.java. 2. Start by creating an empty file called **<FirstnameLastname>Minimal.java**, in a package called **org.csd211.lecture1**, of course, use your own first and last name. 3. Make your program output "Hello, my name is Firstname Lastname!" 4. Run it. 5. What is the output (should be your name.)  TODO STUDENT IN CLASS EXERCISE :   Answer the following questions. Some or all of these will be on the quiz/test.     1. What is meant when we say that java programs are "Write Once, Run Anywhere"? 2. What is the "Platform" problem? 3. Draw a diagram that compares how programs developed using traditional compiled languages differ from programs developed in java. 4. Show the commands to compile and run a java program called Hello.java.      1. If I compile Hello.java, what kind of file is produced and what kind of information does it contain.      1. A blueprint is to a house as a \_\_\_\_\_\_\_class/object\_\_\_\_\_\_\_\_\_\_\_\_\_ is to a \_\_\_\_\_\_\_\_ class/object \_\_\_\_\_\_\_\_\_.      1. What is a reference? 2. What is the relationship between a reference and an object? Draw a picture.      1. What is a null reference? Draw a picture. 2. What must a java class have in order to be executable? 3. What is the name of the method which is the entry point to a program? 4. Write the signature of the method below. 5. What is meant by : "the signature of a method" 6. What is meant by "an object is an instance of a class". 7. Show an example of a single line java comment. 8. Show an example of a multi-line java comment. 9. All java statements end in a/an \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. 10. Complete the sentence: A package is a collection of related \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_. 11. How is a package represented in a file system? 12. What folder does the file Prog1.java reside in?   package org.csd211  public class Prog1{  }   1. Show the java statement that will output the string "Hello World". 2. Is java case sensitive? 3. There can be 1 or more instances of a class. T/F 4. An instance of a class is called a/an \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ . 5. You access an object through its \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.              Files MinimalJavaProgram\_1.java  /\*  \* A Minimal Java Program, generated in Netbeans 7.1  \* Comments are removed for brevity.  \*/  package lecture1;  public class MinimalJavaProgram\_1 {  public static void main(String[] args) {  }  }    MinimalJavaProgramFullyDocumented\_2.java  /\*  \* A Minimal Java Program, generated in Netbeans 7.1  \* Always place comments at the top decribing what the  \* application does.  \*  \* Author : Fred Carella  \* Description : Lecture 1 on a minimal java program.  \* This program is  \* used as a basis to describe and understand the  \* structure of a java program.  \*  \*/  /\*  \* The first line is a package statement which tells the system what package this  \* class belongs to. A package contains related classes and is represented in  \* the filesystem as a folder on disk.  \*  \*/  package lecture1;  /\*\*  \* The @author directive is a javadoc directive indicating the author of the system.  \* javadoc is a utility for producing self documenting code. Running the javadoc utility  \* will create html documentation for your program.  \*  \* javadoc documentation : {@link http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html}  \*  \* @author fcarella  \*/  /\*  \* the class keyword:  \*  \* Every java program is a class. Therefore we always begin by defining the  \* class for our program.  \*  \* A class must have the same name as the file its defined in.  \*  \*  \*  \*/  public class MinimalJavaProgramFullyDocumented\_2 {  /\*\*  \* @param args the command line arguments  \*/  public static void main(String[] args) {  // TODO code application logic here  }  } |