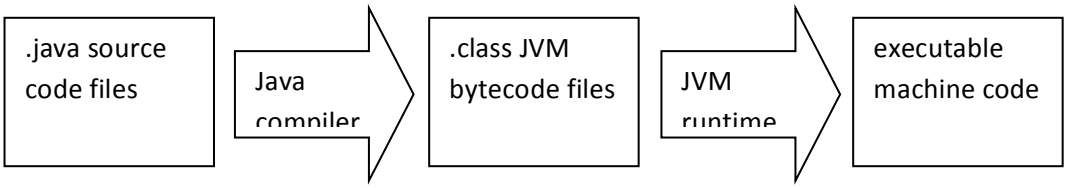
Java Programming 2 – Lecture #1 – Jeremy.Singer@glasgow.ac.uk

About the Java Programming Language

Java is an object-oriented, high-level programming language. It is a platform-neutral language, with a 'write once run anywhere' philosophy. This is supported by a virtual machine architecture called the Java Virtual Machine (JVM). Java source programs are compiled to JVM bytecode class files, which are converted to native machine code on platform-specific JVM instances.



Java is currently one of the top programming languages, according to most popularity metrics.[1] Since its introduction in the late 1990s, it has rapidly grown in importance due to its familiar programming syntax (C-like), good support for modularity, relatively safe features (e.g. garbage collection) and comprehensive library support.

Our First Java Program

It is traditional to write a 'hello world' program as a first step in a new language:

```
/**
 * a first example program to print Hello world
 */
public class Hello {
  public static void main(String [] args) {
    System.out.println("Hello world");
  }
}
```

Contrast with Python

Whereas Python programs are concise, Java programs appear verbose in comparison. Python has dynamic typing, but Java uses static typing. Python scripts are generally interpreted from source, whereas Java programs are compiled to bytecode then executed in a high-performance just-in-time native compiler.

[1] E.g. see http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html

Supporting User Input in Simple Java Programs

There are two ways to receive text-based user input in simple programs like our 'hello world' example.

1) Users can pass command line arguments to the program, which are stored in consecutive elements of the String array that is the only parameter of the main method. Here is a simple example:

```
public static void main(String [] args) {
  System.out.println("Hello " + args[0]);
}
```

2) Programs can request input from the standard input stream using the Scanner library class[2]. The Scanner class will parse the input and return it (if possible) as a value of the appropriate type. Here is a simple example:

```
public static void main(String [] args) {
  java.util.Scanner scanner =
                new java.util.Scanner(System.in);
  int i = scanner.nextInt();
  System.out.println("Come in, number " + i);
```

Questions

1) What does object-orientation mean?

2) So far we have come across at least five different Java types. How many can you spot in the source code examples above?

[2] Check out the Scanner library documentation at
http://docs.oracle.com/javase/7/docs/api/java/util/Scanner.html