**Package**

// package for a group of related classes

// Class name is Scope1

// But because it is in a package called scope, the full name of the class is scope.Scope1

// package names always have lower case first letter

//class names always have Upper case first letter

//See java’s packages and classes that you can use in your applications

//https://docs.oracle.com/javase/8/docs/api/index.html

**Compile vs run**

A java file is written in plain text and saved as a .java file

It is then compiled. You could use a command, javac, to compile the java file. The result would be a .class file.

Then, to execute (run) the .class file, you could use a command java but don’t type the .class part

Summary

>javac Scope.java // result is Scope.class

>java Scope

However, when writing applications, there could be several packages which each contain several classes. Therefore, you would use an integrated development environment (IDE) such as Netbeans or Eclipse to write the .java files and to compile (aka build) the java application into a corresponding .class files.

**Using {}**

They are used to group/surround things. To make the program easier to read, you would indent each opened bracket and then match the closing bracket with the same indentation.

{ some code

{ some code

{ some code

}

}

**One class – one method**

package scope;

// Scope1 is a small application. It has only one class.

// The class has only one method which is the main method

public class Scope1 {

// Main Method - There is always a main method in an application but

// There is not always a main method in a class (we will come to this later)

public static void main(String[] args) {

int a=1,b=2,c=3;

int result = a+b+c;

System.out.println("result in main() == "+result);

}

}

**One class – two methods – no return**

package scope;

// Scope2 is a small application. It has only one class.

// It has two methods, the main method and an add method

public class Scope2 {

// Main Method - There is always a main method in an application

// There is not always a main method in a class (we will come to this later)

public static void main(String[] args) {

int a=1,b=2,c=3;

//calls (go to the) add method

add(a,b,c);

// prints a message after the add method is finished

System.out.println("Back in main()");

}

// The add method has no return

// The add method takes in 3 variables (a,b,c) which are declared as int

// The add method then adds the 3 variables and prints the answer within the method

public static void add(int a, int b, int c){

// prints text only. There are no variables

System.out.println("in method add()");

int sum = a + b + c;

//prints text and the variable sum

System.out.println("sum in add() == "+sum);

}

}

**One class – two methods - return**

package scope;

// Scope3 is a small application. It has only one class.

// It has two methods, the main method and an add method

// The add method is different to the one in Scope2

public class Scope3 {

// Main Method - There is always a main method in an application

// There is not always a main method in a class (we will come to this later)

public static void main(String[] args) {

int a=1,b=2,c=3;

//calls (go to the) add method

int result = add2(a,b,c);

// prints a message after the add method is finished

// but this time it also prints a variable which is result

System.out.println("result in main() == "+result);

}

// The add method has a return this time

// The return is an int

// The add method takes in 3 variables (a,b,c) which are declared as int

// The add method then adds the 3 variables and assigns the answer to the variable sum

//Then, the answer sum is returned (to the main method where it was sent from)

public static int add2(int x, int y, int z){

System.out.println("in method add2()");

// add x, y and z and assign the answer to sum (which is an int)

int sum = x + y + z;

// return sum

return sum;

}

}