This page gives a very brief tutorial on the Scala programming language.

Classes, Traits, Objects and Packages

Classes

Classes in Scala are very similar to classes in Java. They are templates containing fields and methods. Like in Java, classes can be instantiated using the new construct, there can be many “instances” (or “objects”) of the same class.

In Scala there exists a special kind of class named case classes. You will learn about case classes during the course.

Classes in Scala cannot have static members. You can use objects (see below) to achieve similar functionality as with static members in Java.

Traits

Traits are like interfaces in Java, but they can also contain concrete members, i.e. method implementations or field definitions.

Objects

Object in Scala are like classes, but for every object definition there is only one single instance. It is not possible to create instances of objects using new, instead you can just access the members (methods or fields) of an object using its name.

Packages

Adding a statement such as package foo.bar at the top of a file makes the code in a file part of the package foo.bar. You can then do import foo.bar._ to make everything from package foo.bar available in your code. The content of a package can be scattered across many files. If you define a class MyClass in package foo.bar, you can import that specific class (and not anything else from that package) with import foo.bar.MyClass.

In Scala, everything can be imported, not only class names. So for instance if you have an object baz in package foo.bar, then import foo.bar.baz._ would import all the members of that object.

Hello, World! in Scala

Here are two ways to define a program which outputs “Hello, World!” in Scala:

```
1  object HelloWorld extends App {
2    println("Hello, World!")
3  }
```

or

```
object HelloWorld {
  def main(args: Array[String]) {
    println("Hello, World!")
  }
}
```

In Scala, the main or entry point method is defined in an object. An object can be made executable by either adding extending the type App or by adding a method def main(args: Array[String]).

Source Files, Classfiles and the JVM

Scala source code is stored in text files with the extension .scala. Typically Scala programmers create one source file for each class, or one source file for a class hierarchy: In fact, Scala allows multiple classes and objects to be defined in the same source file.

- The name of a Scala source file can be chosen freely, but it is recommended to use the name of a class which is defined in that file.
- Package hierarchies should be reflected in directory structure: a source file defining class C in package foo.bar should be stored in a subdirectory as foo/bar/C.scala. Scala does not really enforce this convention, but some tools such as the Scala IDE for eclipse might have problems otherwise.

The scala compiler compiles .scala source files to .class files, like the Java compiler. Classfiles are binary files containing machine code for the Java Virtual Machine. In order to run a Scala program, the JVM has to know the directory where classfiles are stored. This parameter is called the “classpath”.

If you are using eclipse or sbt to compile and run your Scala code, you don’t need to do any of the above manually - these tools take care of invoking the Scala compiler and the JVM with the correct arguments.

External Documentation

There is a large number of online resources available for learning Scala. The Learning Resources wiki page contains a few links that we think are the most useful for our class.

✓ Complete