GETTING STARTED

The best way to learn Scala depends on what you know already and the way you prefer to learn things. There is a variety of resources available including books, tutorials, training courses, presentations, and of course the Scala compiler for practice. Many people find a good combination is to have one of the Scala books at hand and to start right away trying the examples with the Scala compiler. On the other hand, you may want to get started with a Scala training course or using the material available online.

As your knowledge of Scala grows, you will find there is more advanced material and a very friendly Scala community at hand to help you. They all share a passion for Scala and welcome newcomers warmly. Many have written helpful material for programmers new to Scala, will respond to emails asking for help or are sharing neat new techniques, advanced concepts or tools in one of several Scala forums or personal blogs.

Scala for Programming Beginners

If you are just starting to learn how to code, you will find that a large portion of the material about Scala assumes that you already have some programming experience. There are two valuable resources which we can recommend to programming beginners that will take you directly into the world of Scala:

- The online class Functional Programming Principles in Scala, available on Coursera. Taught by the creator of Scala, Martin Odersky, this online class takes a somewhat academic approach to teach the fundamentals of functional programming. You will learn a lot of Scala by solving the programming assignments.
- Kojo is an interactive learning environment that uses Scala programming to explore and play with math, art, music, animations and games.

Your first lines of code

The “Hello, world!” Program

As a first example, we use the standard “Hello, world!” program to demonstrate the use of the Scala tools without knowing too much about the language.

```
1. object HelloWorld {
2.   def main(args: Array[String]): Unit = {
3.     println("Hello, world!")
4.   }
5. }
```

The structure of this program should be familiar to Java programmers: it consists of the method `main` which prints out a friendly greeting to the standard output.

Environment	Variable	Value (example)
Unix	`$SCALA_HOME`	`/usr/local/share/scala`
	`$PATH`	`$PATH:$SCALA_HOME/bin`
Windows	`%SCALA_HOME%`	`c:\Progra~1\Scala`
	`%PATH%`	`%PATH%;%SCALA_HOME%\bin`

Run it interactively!

The `scala` command starts an interactive shell where Scala expressions are interpreted interactively.

```
1.  > scala
2.  This is a Scala shell.
3.  Type in expressions to have them evaluated.
4.  Type :help for more information.
5.
6.  scala> object HelloWorld {
7.       |   def main(args: Array[String]): Unit = {
8.       |     println("Hello, world!")
9.       |   }
10.      | }
11. defined module HelloWorld
12.
13. scala> HelloWorld.main(Array())
14. Hello, world!
15.
16. scala>:q
17. >
```

The shortcut `:q` stands for the internal shell command `:quit` used to exit the interpreter.

Compile it!

The `scalac` command compiles one (or more) Scala source file(s) and generates Java bytecode which can be executed on any standard JVM. The Scala compiler works similarly to `javac` , the Java compiler of the Java SDK.

```
1.  > scalac HelloWorld.scala
```

By default `scalac` generates the class files into the current working directory. You may specify a different output directory using the `-d` option.

```
1.  > scalac -d classes HelloWorld.scala
```

Execute it!

The `scala` command executes the generated bytecode with the appropriate options:

```
1.  > scala HelloWorld
```

```
> scala -cp classes HelloWorld
```

The argument of the `scala` command has to be a top-level object. If that object extends trait scala.App, then all statements contained in that object will be executed; otherwise you have to add a method `main` which will act as the entry point of your program.

Here is how the “Hello, world!” example looks like using the `App` trait:

```
object HelloWorld extends App {
  println("Hello, world!")
}
```

Script it!

We may also run our example as a shell script or batch command (see the examples in the man pages of the `scala` command).

The bash shell script `script.sh` containing the following Scala code (and shell preamble):

```
#!/usr/bin/env scala

object HelloWorld extends App {
  println("Hello, world!")
}
HelloWorld.main(args)
```

can be run directly from the command shell:

```
> ./script.sh
```

Note: We assume here that the file `script.sh` has execute permission and the search path for the `scala` command is specified in the `PATH` environment variable.