# Lab 00 Tool Setup

## Overview

Every participant needs to have computer with internet access, on which the following is installed: (In the paranthesis, we have written the version, that the setup is tested with)

* JDK 8 (1.8.0\_74)
* Scala 2.12
* Sbt
* Gradle
* Eclipse with the Scala plugin  (Oxygen.2 Release (4.7.2))
* Git client

## Install Java (JDK)

You need JDK v 8 (1.8.0\_74).

You can find the installer appropriate for your platform on Oracle's page:

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

After installation, open a shell and type:

java -version

Reply should be similar to:

ava version "1.8.0\_74"

Java(TM) SE Runtime Environment (build 1.8.0\_74-b02)

Java HotSpot(TM) 64-Bit Server VM (build 25.74-b02, mixed mode)

## Install Scala

Next, you need to install the Scala distribution. Download and follow instructions for your platform here: <https://www.scala-lang.org/download/2.12.4.html>

After installation, open a shell and type:

scala -version

Reply should be similar to:

Scala code runner version 2.12.4 -- Copyright 2002-2017, LAMP/EPFL and Lightbend, Inc.

## Recommended Environment Variables

Recommended environment variables should be:

| 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 |

## Install Sbt

Sbt is the preferred build tool for Scala.

Download and install Sbt for your platform from: <https://www.scala-sbt.org/1.x/docs/Setup.html>

After installation, open a shell and type:

sbt about

Reply should be similar to:

This is sbt 1.1.1

# Install Gradle

We will be exploring how to interact with the JVM ecosystem, and use the most popular Java build tool, Gradle.

Download and install Gradle for your platform from:

https://gradle.org/install/

After installation, open a shell and type:

gradle --version

Reply should be similar to:

------------------------------------------------------------

Gradle 4.4.1

------------------------------------------------------------

## Install a development environment Eclipse + Scala plugin

### First, Eclipse

Next, download and install Eclipse, to have a nice development environment:

<https://www.eclipse.org/downloads/>

After installing, check that you can start the Eclipse application.

### Next, the Scala plugin

Now install the Scala plugin for Eclipse:

Start Eclipse > Goto Help Menu > Eclipse Marketplace > Type ”scala” in search bar> Select ”Scala IDE” > Press Install button > Accept.

After installation, restart Eclipse.

Test that everything is installed correctly:

In Eclipse > Goto File Menu > Select ”New” > Other … > Type ”Scala worksheet” in search bar > Select Scala worksheet (Next) > Give it a name >

Check that the worksheet has a green comment //Welcome to ...

## Install Git

Download a Git client appropriate for your platform:

<https://git-scm.com/downloads>

After installation, open a shell and type:

git –version

Reply should be similar to:

git version 2.6.4

# Lab 01 Hello World – Scala on the JVM

We will setup, compile, build and test a simple project in Eclipse.

Purpose: Get acquainted with running Scala code on JVM

For all lab exercises, use the prepared project scala\_labs.

You can see the solutions in the project scala\_solutions.

You will be working in the package dk.lundogbendsen.scala.labs.helloworld\_01

### Exercises

1. In the file HelloWorld.scala, write a simple hello world, that takes your name from the command line, and prints a greeting. Run.

2. Use the Jodatime class from Java,

3. Compile from command line

4. Run using Scala

5. Run using Java

6. Inspect generated classes

7. Add a unit test

8. Build project using the sb build script

9. Build project using the gradle build script

# Lab 02 Language Basics

We will be working with the REPL, and also we will be working with a worksheet in Eclipse.

See the package:

language\_basics\_02

# 03 First steps with functional programming

We will look at basic concepts in functional programming, and try to convince your mind, to move from an imperative to functional style.

See the package:

language\_basics\_03

# 04 Higher order functions

We will explore higher order functions, multiparameter lists and currying in this lab. Learn to write exressive and concise code.

See the package:

higher\_order\_functions\_04

# 05 Objects and data

The staple of OOP is classes and objects. We work with objects, classes, and design of classes – constructors, parameters, members, visibility.

See the package:

objects\_and\_data\_05

# 06 Class Hierarchies

More advanced topics, adding functionality using traits, Scala's class hierarchy.

See the package:

class\_hierarchies\_06

# 07 Polymorphism

We will work with OO and FP style polymorphism.

See the package:

polymorphism\_07

# 08 Types and pattern matching

See the package:

# 09 Scala Collections

Let's explore the Scala collections, and write elegant, functional style data processing pipelines.

See the package:

collections\_09

# 10 Functional patterns

Time to tie the pieces together, and do a larger project!

See the package:

big\_picture\_10

# 11 Building and testing

See the package:

building\_and\_testing\_11