800+ Q&As | Logout | Contact

Java-Success.com

Prepare to fast-track, choose & go places with 800+ Java & Big Data Q&As with lots of code & diagrams.

search here … Go

Home Why? ▾ 300+ Java FAQs ▾ 300+ Big Data FAQs ▾ Courses ▾

Membership ▾ Your Career ▾

Home › bigdata-success.com › Tutorials - Big Data › TUT - Starting Spark & Scala ›

02. Setting up Scala IDE & using Maven plugins for Java developers

02. Setting up Scala IDE & using Maven plugins for Java developers

Posted on July 25, 2016

This extends Setting up Java, Maven & Eclipse and Setting up Scala & practicing the concepts via REPL the Scala way for Java developers.

Step 1: Scala IDE is an eclipse plugin. Following is the Scala IDE update site URL for Eclipse 4.4 (Luna) and Eclipse 4.5 (Mars).

```

http://download.scala-ide.org/sdk/lithium/e44/scal
```

300+ Java Interview FAQs

300+ Java FAQs 🔥

16+ Java Key Areas Q&As

150+ Java Architect FAQs

80+ Java Code Quality Q&As

150+ Java Coding Q&As

300+ Big Data Interview FAQs

300+ Big Data FAQs 🔥

Tutorials - Big Data

TUT - ⏯ Starting Big Data

TUT - Starting Spark & Scala

TOP

Open eclipse., and then go to Help –>> Install New Software –> Add –>

Name: Scala IDE
Location: http://download.scala-ide.org/sdk/lithium/e44/scala211/stable/site

Select "**Scala IDE for Eclipse**", and install it and restart eclipse.

Step 3: This tutorial assumes that you have already installed Java, Scala, and maven as per the links at the top of this tutorial. So, let's create a simple Scala maven project called **"simple-scala**". "-B" is used for Maven non-interactive mode.

```

C:\Users\java-_000\projects>mvn archetype:generate

```

Step 4: Import the project "simple-scala" into eclipse.

File –> **Import** –> "**Existing Maven Projects**" –> select "C:\Users\java-_000\projects\simple-scala". It actually looks at the pom.xml in that folder.

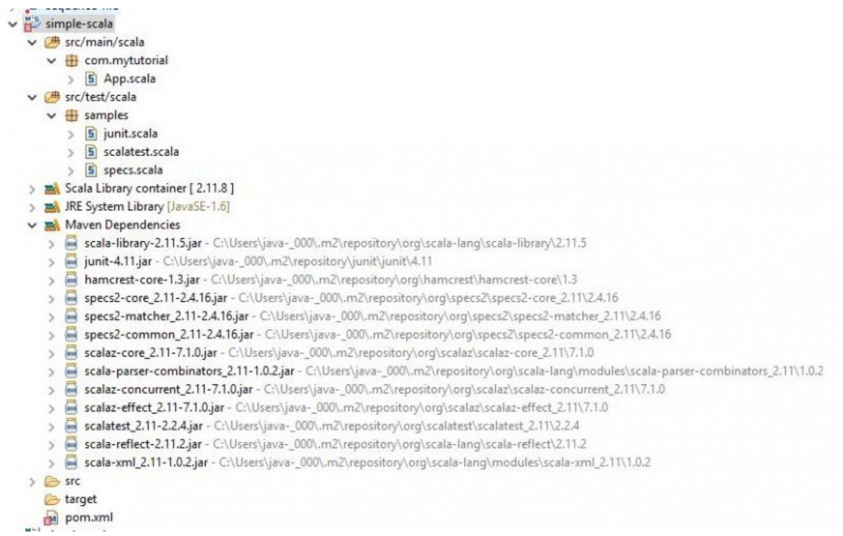
Step 5: Right mouse click on "simple-scala", and then select "**Configure** –> **Add Scala Nature**", now you will see an "M" for Maven, and "S" for Scala. If you are interested, you can check the hidden file ".project".

Open "MySpecTest.scala" and press "**ctrl+shift+o**" to organize imports and fix the compile error.

- TUT - Starting with Python
- TUT - Kafka
- TUT - Pig
- TUT - Apache Storm
- TUT - Spark Scala on Zeppelin
- TUT - Cloudera
- TUT - Cloudera on Docker
- TUT - File Formats
- TUT - Spark on Docker
- TUT - Flume
- TUT - Hadoop (HDFS)
- TUT - HBase (NoSQL)
- TUT - Hive (SQL)
- TUT - Hadoop & Spark
- TUT - MapReduce
- TUT - Spark and Scala
- TUT - Spark & Java
- TUT - PySpark on Databricks
- TUT - Zookeeper

800+ Java Interview Q&As

- 300+ Core Java Q&As
- 300+ Enterprise Java Q&As
- 150+ Java Frameworks Q&As
- 120+ Companion Tech Q&As
- Tutorials - Enterprise Java



```
simple-scala
  src/main/scala
    com.mytutorial
      App.scala
  src/test/scala
    samples
      junit.scala
      scalatest.scala
      specs.scala
  Scala Library container [ 2.11.8 ]
  JRE System Library [JavaSE-1.6]
  Maven Dependencies
    scala-library-2.11.5.jar - C:\Users\java-_000\.m2\repository\org\scala-lang\scala-library\2.11.5
    junit-4.11.jar - C:\Users\java-_000\.m2\repository\junit\junit\4.11
    hamcrest-core-1.3.jar - C:\Users\java-_000\.m2\repository\org\hamcrest\hamcrest-core\1.3
    specs2-core_2.11-2.4.16.jar - C:\Users\java-_000\.m2\repository\org\specs2\specs2-core_2.11\2.4.16
    specs2-matcher_2.11-2.4.16.jar - C:\Users\java-_000\.m2\repository\org\specs2\specs2-matcher_2.11\2.4.16
    specs2-common_2.11-2.4.16.jar - C:\Users\java-_000\.m2\repository\org\specs2\specs2-common_2.11\2.4.16
    scalaz-core_2.11-7.1.0.jar - C:\Users\java-_000\.m2\repository\org\scalaz\scalaz-core_2.11\7.1.0
    scala-parser-combinators_2.11-1.0.2.jar - C:\Users\java-_000\.m2\repository\org\scala-lang\modules\scala-parser-combinators_2.11\1.0.2
    scalaz-concurrent_2.11-7.1.0.jar - C:\Users\java-_000\.m2\repository\org\scalaz\scalaz-concurrent_2.11\7.1.0
    scalaz-effect_2.11-7.1.0.jar - C:\Users\java-_000\.m2\repository\org\scalaz\scalaz-effect_2.11\7.1.0
    scalatest_2.11-2.2.4.jar - C:\Users\java-_000\.m2\repository\org\scalatest\scalatest_2.11\2.2.4
    scala-reflect-2.11.2.jar - C:\Users\java-_000\.m2\repository\org\scala-lang\scala-reflect\2.11.2
    scala-xml_2.11-1.0.2.jar - C:\Users\java-_000\.m2\repository\org\scala-lang\modules\scala-xml_2.11\1.0.2
  src
  target
  pom.xml
```

Simple Scala project in eclipse

Step 6: The **pom.xml** file need some changes, and should look like

```

<project xmlns="http://maven.apache.org/POM/4.0.(
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mytutorial</groupId>
  <artifactId>simple-scala</artifactId>
  <version>1.0-SNAPSHOT</version>
  <name>${project.artifactId}</name>
  <description>My wonderfull scala app</descript
  <inceptionYear>2015</inceptionYear>
  <licenses>
    <license>
      <name>My License</name>
      <url>http://....</url>
      <distribution>repo</distribution>
    </license>
  </licenses>

  <properties>
    <maven.compiler.source>1.6</maven.compiler.s
    <maven.compiler.target>1.6</maven.compiler.t
    <encoding>UTF-8</encoding>
    <scala.version>2.11.5</scala.version>
    <scala.compat.version>2.11</scala.compat.ver
  </properties>

  <dependencies>
    <dependency>
      <groupId>org.scala-lang</groupId>
      <artifactId>scala-library</artifactId>
      <version>${scala.version}</version>
    </dependency>

    <!-- Test -->
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>
      <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>org.specs2</groupId>
      <artifactId>specs2-core_${scala.compat.ver
      <version>2.4.16</version>
      <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.specs2</groupId>
        <artifactId>specs2-junit_${scala.compat.
        <version>2.4.16</version>
        <scope>test</scope>
    </dependency>
    <dependency>
      <groupId>org.scalatest</groupId>
      <artifactId>scalatest_${scala.compat.versi
      <version>2.2.4</version>
      <scope>test</scope>
    </dependency>
  </dependencies>

  <build>
    <sourceDirectory>src/main/scala</sourceDirec
    <testSourceDirectory>src/test/scala</testSou
    <plugins>
      <plugin>
        <!-- see http://davidb.github.com/scala-
        <groupId>net.alchim31.maven</groupId>
        <artifactId>scala-maven-plugin</artifact
        <version>3.2.0</version>
        <executions>
          <execution>
            <goals>
              <goal>compile</goal>
              <goal>testCompile</goal>
            </goals>
            <configuration>
              <args>
             <!--    <arg>-make:transitive</arg>
                <arg>-dependencyfile</arg>
                <arg>${project.build.directory}/
              </args>
            </configuration>
          </execution>
        </executions>
      </plugin>
      <plugin>
        <groupId>org.apache.maven.plugins</group
```

```
          <artifactId>maven-surefire-plugin</artif
          <version>2.18.1</version>
          <configuration>
            <useFile>false</useFile>
            <disableXmlReport>true</disableXmlRepo
            <!-- If you have classpath issue like
            <!-- useManifestOnlyJar>false</useMani
            <includes>
              <include>**/*Test.*</include>
              <include>**/*Suite.*</include>
            </includes>
          </configuration>
        </plugin>
      </plugins>
    </build>
</project>
```

The changes are:

1. The commenting of "<arg>-make:transitive</arg>" inside the plugin "scala-maven-plugin".

```

<!--      <arg>-make:transitive</arg> -->

```

2. Add the following dependency:

```

   <dependency>
        <groupId>org.specs2</groupId>
        <artifactId>specs2-junit_${scala.compat.ve
        <version>2.4.16</version>
        <scope>test</scope>
     </dependency>

```

Step 7: Install the "**m2e-scala**" plugin from the following update site. The steps are very similar to "installing the Scala IDE" shown in Step 1.

```

http://alchim31.free.fr/m2e-scala/update-site/

```

Step 8: Now, right click on the package "**com.mytutorial**" within the project "simple-scala", and then select

New –> **Other** –> **Scala Wizards** –> "Scala Object" and name the Object "**com.mytutorial.Hello**".

```

package com.mytutorial

object Hello extends App {
  println("Hello from Scala")
}

```

You can run as

Select "**Hello.scala**" and right mouse button click and then "**Run As**" –> "**Scala Application**".

Output:

```

Hello from Scala
```

Packaging it using Maven

Go to a DOS command prompt:

```

C:\Users\java-_000\projects\simple-scala>mvn clean

```

This is one of many ways to getting started with Scala. There is a build tool for scala known as “**sbt**“.

‹ 01. Setting up Scala & practicing the concepts via REPL the Scala way for Java developers

03: Akka tutorial “ask” request – response pattern Scala way ›

Disclaimer

The contents in this Java-Success are copyrighted and from EmpoweringTech pty ltd. The EmpoweringTech pty ltd has the right to correct or enhance the current content without any prior notice. These are general advice only, and one needs to take his/her own circumstances into consideration. The EmpoweringTech pty ltd will not be held liable for any damages caused or alleged to be caused either directly or indirectly by these materials and resources. Any trademarked names or labels used in this blog remain the property of their respective trademark owners. Links to external sites do not imply endorsement of the linked-to sites. Privacy Policy

Top