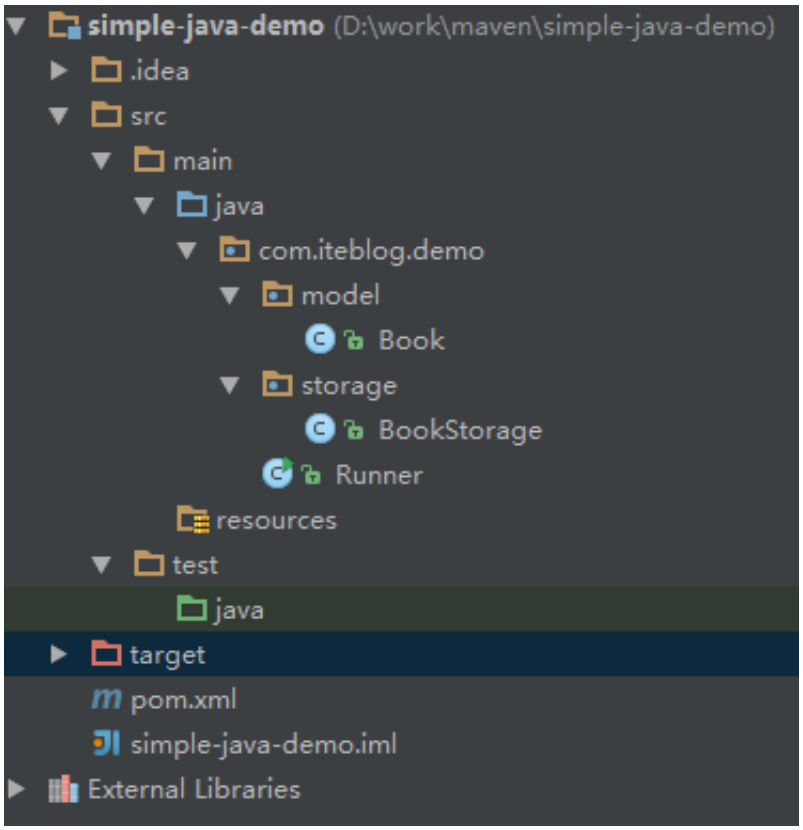
如何在Java Maven工程中编写Scala代码

今天我将介绍如何在Java工程使用Scala代码。对于那些想在真实场景中尝试使用Scala的开发人员来说，会非常有意思。这和你项目中有什么类型的东西毫无关系：不管是Spring还是Spark还是别的。我们废话少说，开始吧。

抽象Java Maven项工程

这里我们使用Maven来管理我们的Java项目，项目的结果如下所示：



如果想及时了
解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

正如你所看到的，工程的结构非常简单。它有标准的布局和仅仅三个Java类，如下所示：

```
package com.iteblog.demo.model;

/**
 * User: 过往记忆
 * Date: 2016-12-30
 * Time: 下午23:16
```

```
 * bolg: https://www.iteblog.com
 * 本文地址：https://www.iteblog.com/archives/1947.html
 * 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 * 过往记忆博客微信公共帐号：iteblog_hadoop
 */
public class Book {

    private String name = null;
    private String author = null;

    public Book(String name, String author) {
        this.name = name;
        this.author = author;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getAuthor() {
        return author;
    }

    public void setAuthor(String author) {
        this.author = author;
    }

    @Override
    public String toString() {
        return "Book {" +
                "name='" + name + '₩'' +
                ", author='" + author + '₩'' +
                '}';
    }
}
```

下面是所谓的数据存储

```
package com.iteblog.demo.storage;
```

```
import com.iteblog.demo.model.Book;

import java.util.ArrayList;

/**
 * User: 过往记忆
 * Date: 2016-12-30
 * Time: 下午23:16
 * bolg: https://www.iteblog.com
 * 本文地址：https://www.iteblog.com/archives/1947.html
 * 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 * 过往记忆博客微信公共帐号：iteblog_hadoop
 */
public class BookStorage {

    private ArrayList<Book> books = new ArrayList<>();

    public BookStorage() {
        books.add(new Book("White Fang", "Jack London"));
        books.add(new Book("The Sea-Wolf", "Jack London"));
        books.add(new Book("The Road", "Jack London"));
        books.add(new Book("The Adventures of Tom Sawyer", "Mark Twain"));
        books.add(new Book("Around the World in 80 Days", "Jules Verne"));
        books.add(new Book("Twenty Thousand Leagues Under the Sea", "Jules Verne"));
        books.add(new Book("The Mysterious Island", "Jules Verne"));
        books.add(new Book("The Four Million", "O. Henry"));
        books.add(new Book("The Last Leaf", "O. Henry"));
    }

    public ArrayList<Book> getBooks() {
        return books;
    }
}
```

最后是主类：

```
package com.iteblog.demo;

import com.iteblog.demo.storage.BookStorage;

/**
 * User: 过往记忆
```

```
 * Date: 2016-12-30
 * Time: 下午23:16
 * bolg: https://www.iteblog.com
 * 本文地址：https://www.iteblog.com/archives/1947.html
 * 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 * 过往记忆博客微信公共帐号：iteblog_hadoop
 */
public class Runner {

    public static void main(String[] args) {
        BookStorage storage = new BookStorage();
        storage.getBooks().stream().forEach(System.out::println);
    }

}
```

最后别忘记我们还有个pom.xml文件，它可能包含一些依赖，插件和构建目标；不过这些并不重要。

将Scala混合到Java Maven项目中

为了能够在Java Maven工程中使用Scala，我们需要使用一个Maven插件：scala-maven-plugin，把下面的代码加入到pom.xml文件中

```
<plugin>
    <groupId>net.alchim31.maven</groupId>
    <artifactId>scala-maven-plugin</artifactId>
    <executions>
        <execution>
            <id>scala-compile-first</id>
            <phase>process-resources</phase>
            <goals>
                <goal>add-source</goal>
                <goal>compile</goal>
            </goals>
        </execution>
        <execution>
            <id>scala-test-compile</id>
            <phase>process-test-resources</phase>
            <goals>
                <goal>testCompile</goal>
            </goals>
```

```
      </execution>
    </executions>
</plugin>
```

因为我们需要使用Scala代码，所以我们还需要加入scala-library依赖，如下：

```
<dependency>
    <groupId>org.scala-lang</groupId>
    <artifactId>scala-library</artifactId>
    <version>2.11.7</version>
</dependency>
```

最后完整的额pom.xml文件内容如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/
maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.iteblog.demo</groupId>
    <artifactId>iteblog</artifactId>
    <version>1.0-SNAPSHOT</version>

    <dependencies>
        <dependency>
            <groupId>org.scala-lang</groupId>
            <artifactId>scala-library</artifactId>
            <version>2.11.7</version>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <!-- This plugin compiles Scala files -->
            <plugin>
                <groupId>net.alchim31.maven</groupId>
                <artifactId>scala-maven-plugin</artifactId>
                <executions>
```

```
        <execution>
          <id>scala-compile-first</id>
          <phase>process-resources</phase>
          <goals>
            <goal>add-source</goal>
            <goal>compile</goal>
          </goals>
        </execution>
        <execution>
          <id>scala-test-compile</id>
          <phase>process-test-resources</phase>
          <goals>
            <goal>testCompile</goal>
          </goals>
        </execution>
      </executions>
    </plugin>
    <!-- This plugin compiles Java files -->
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <configuration>
        <source>1.8</source>
        <target>1.8</target>
      </configuration>
      <executions>
        <execution>
          <phase>compile</phase>
          <goals>
            <goal>compile</goal>
          </goals>
        </execution>
      </executions>
    </plugin>
    <!-- This plugin adds all dependencies to JAR file during 'package' command.
    Pay EXTRA attention to the 'mainClass' tag.
    You have to set name of class with entry point to program ('main' method) -->
    <plugin>
      <artifactId>maven-assembly-plugin</artifactId>
      <version>2.5.3</version>
      <configuration>
        <descriptorRefs>
          <descriptorRef>jar-with-dependencies</descriptorRef>
        </descriptorRefs>
        <archive>
          <manifest>
```

```
                <mainClass>ScalaRunner</mainClass>
              </manifest>
            </archive>
          </configuration>
          <executions>
            <execution>
              <phase>package</phase>
              <goals>
                <goal>single</goal>
              </goals>
            </execution>
          </executions>
        </plugin>

    </plugins>
  </build>

</project>
```

更新完上面的内容之后，你需要等待Maven下载完所有的依赖。

现在我们可以在Java工程中使用Scala代码了。为此，您需要创建新的文件夹src/main/scala；Scala Maven插件将会识别这些目录，并且编译其中的Scala文件：
现在我们在工程里面加入Scala代码：

```
package com.iteblog.service

import java.util

import com.iteblog.demo.model.Book
import scala.collection.JavaConversions._

/**
 * User: 过往记忆
 * Date: 2016-12-30
 * Time: 下午23:16
 * bolg: https://www.iteblog.com
 * 本文地址：https://www.iteblog.com/archives/1947.html
 * 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 * 过往记忆博客微信公共帐号：iteblog_hadoop
 */
object BooksProcessor {
```

```
  def filterByAuthor(author: String)(implicit books: util.ArrayList[Book]) = {
    books.filter(book => book.getAuthor == author)
  }

}
```

现在我们可以在Scala代码中遍历相关的图书了：

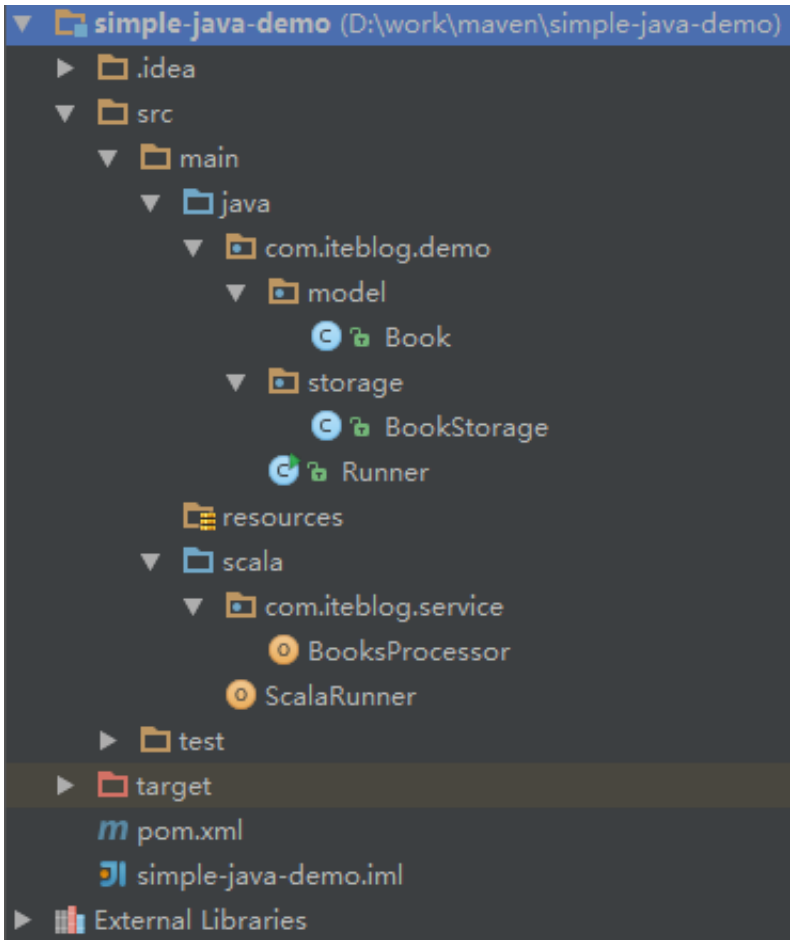
```
import com.iteblog.demo.storage.BookStorage
import com.iteblog.service.BooksProcessor

/**
 * User: 过往记忆
 * Date: 2016-12-30
 * Time: 下午23:16
 * bolg: https://www.iteblog.com
 * 本文地址：https://www.iteblog.com/archives/1947
 * 过往记忆博客，专注于hadoop、hive、spark、shark、flume的技术博客，大量的干货
 * 过往记忆博客微信公共帐号：iteblog_hadoop
 */
object ScalaRunner extends App {

  implicit val books = new BookStorage().getBooks
  BooksProcessor.filterByAuthor("Jack London").foreach(b => println(b))

}
```

最后加入Scala代码的工程看起来如下：



如果想及时了
解Spark、Hadoop或者Hbase相关的文章，欢迎关注微信公共帐号：iteblog_hadoop

运行ScalaRunner代码得到下面的结果：

```
Book {name='White Fang', author='Jack London'}
Book {name='The Sea-Wolf', author='Jack London'}
Book {name='The Road', author='Jack London'}
```