Session 15: SCALA - SESSION IV

Assignment 15.1

Student Name:	Abarajithan SA
Course:	Big Data Hadoop & Spark Training
Start Date:	2017-09-09
End Date:	2017-11-26

Assignment 15.1–

Write a simple program in Scala to show Simple Inheritance and Multiple Inheritance.

Contents

Introduction

In this assignment, we are going to write a simple SCALA code to show Single Inheritance and Multiple inheritance in Scala.

Problem Statement

1. Write a simple program to show inheritance in scala.
2. Write a simple program to show multiple inheritance in scala.

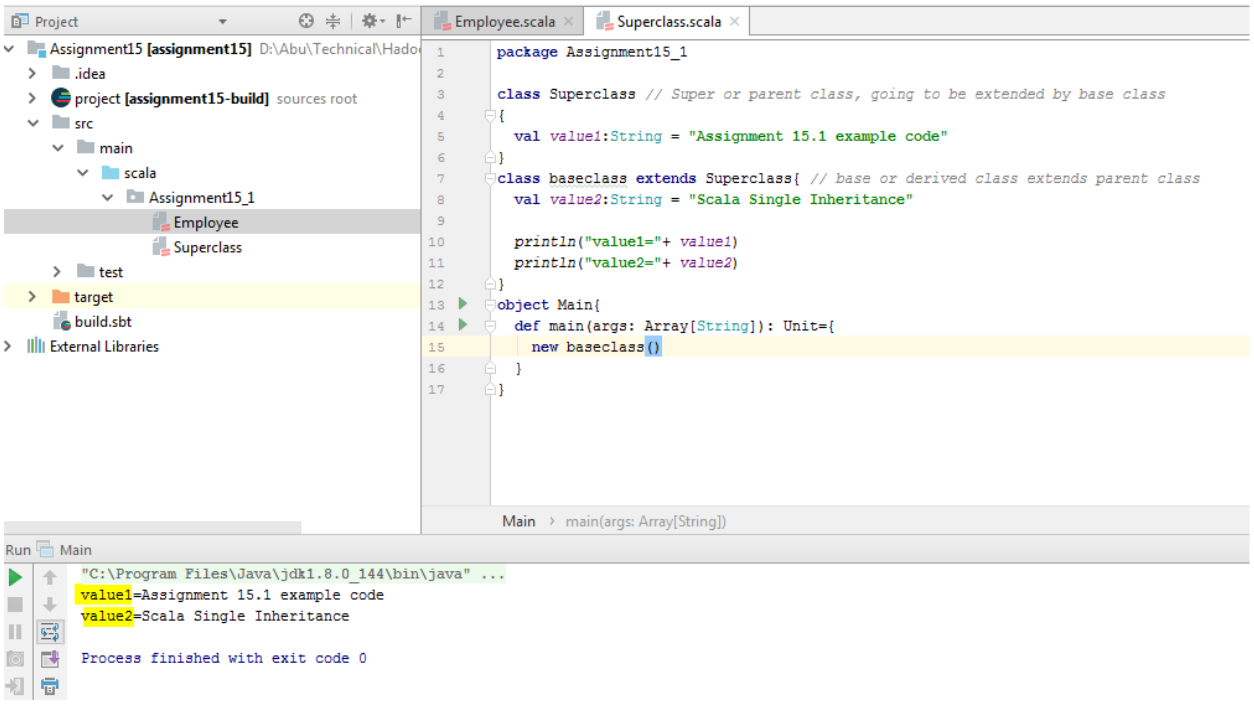
Task 1 – Write a simple program to show inheritance in scala.

Inheritance is an object oriented concept which is used to reusability of code. You can achieve inheritance by using **extends** keyword. To achieve inheritance a class must extend to other class. A class which is extended called **super** or **parent** class. A class which extends class is called **derived** or **base** class.

Scala Code

```
package Assignment15_1

class Superclass // Super or parent class, going to be extended by base class
{
  val value1:String = "Assignment 15.1 example code"
}
class baseclass extends Superclass{ // base or derived class extends parent class
  val value2:String = "Scala Single Inheritance"

  println("value1="+ value1)
  println("value2="+ value2)
}
object Main{
  def main(args: Array[String]): Unit={
    new baseclass()
  }
}
```

Output



```
package Assignment15_1

class Superclass // Super or parent class, going to be extended by base class
{
  val value1:String = "Assignment 15.1 example code"
}
class baseclass extends Superclass{ // base or derived class extends parent class
  val value2:String = "Scala Single Inheritance"

  println("value1="+ value1)
  println("value2="+ value2)
}
object Main{
  def main(args: Array[String]): Unit={
    new baseclass()
  }
}
```

```
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
value1=Assignment 15.1 example code
value2=Scala Single Inheritance

Process finished with exit code 0
```

Task 2 – Write a simple program to show multiple inheritance in scala.

Multiple inheritance is a feature of some object-oriented computer programming languages in which an object or class can inherit characteristics and features from more than one parent object or parent class. It is distinct from single inheritance, where an object or class may only inherit from one particular object or class.

Scala supports various types of inheritance including single, multilevel, **multiple**, and hybrid. You can use single, multilevel and hierarchal in your class. **Multiple** and **hybrid** can only be achieved by using **traits**.

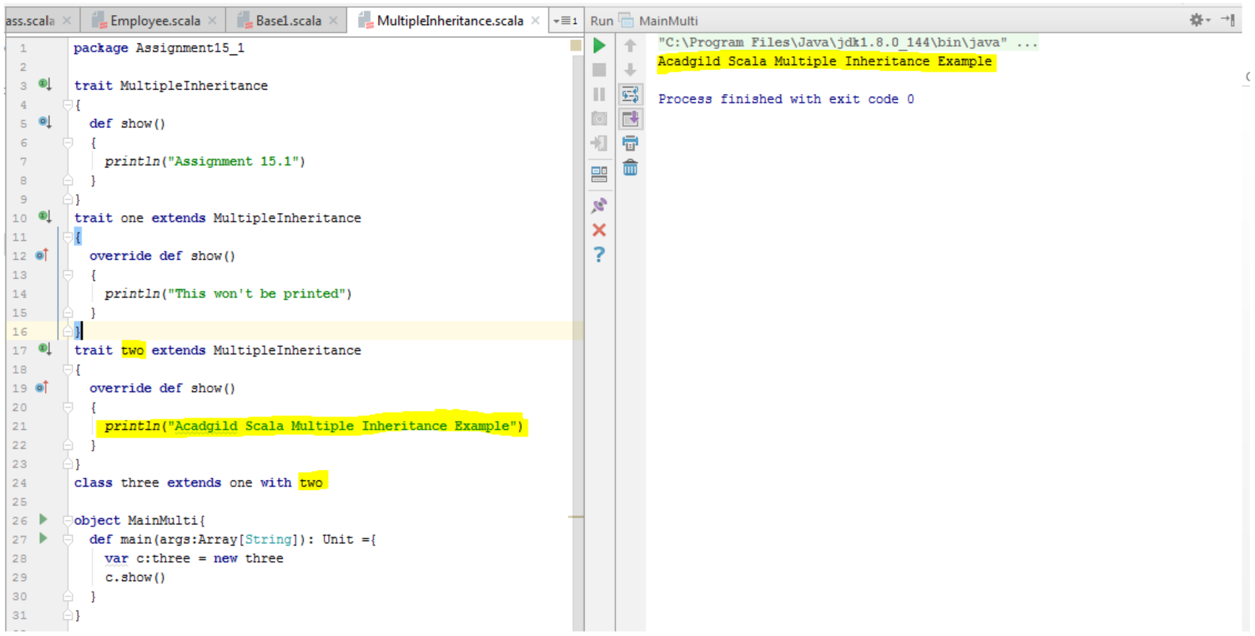
Scala **doesn't allow for multiple inheritance** per se, but allows to extend multiple **traits**.

Traits are used to share interfaces and fields between classes. They are similar to Java 8's interfaces. Classes and objects can extend traits but traits cannot be instantiated and therefore have no parameters. Traits in Scala are best described as "**interfaces that can provide concrete members**."

```
package Assignment15_1

trait MultipleInheritance //parent trait
{
  def show() // defining the function show()
  {
    println("Assignment 15.1")
  }
}
trait one extends MultipleInheritance // extending the parent trait
{
  override def show()
  {
    println("This won't be printed")
  }
}
trait two extends MultipleInheritance // extending the parent trait
{
  override def show()
  {
    println("Acadgild Scala Multiple Inheritance Example")
  }
}
class three extends one with two  //extending the base traits, calling the function
show()

object MainMulti{
  def main(args:Array[String]): Unit ={
    var c:three = new three // it will call last function which is mentioned in the
class three, changing the order will give different result
    c.show()
  }
}
```

Output

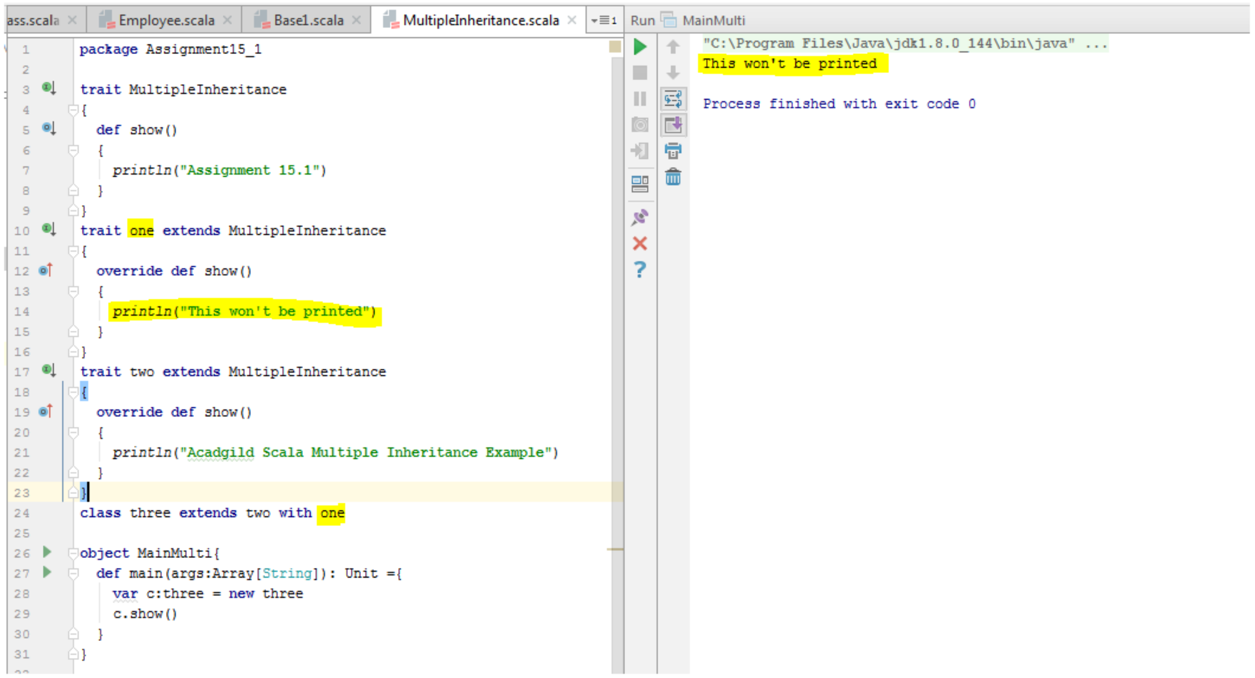
Example 1, here the class ***three*** calling the trait one with ***two***, the ***two*** in the last order and hence the function of ***two*** will be called and output is,



```
package Assignment15_1

trait MultipleInheritance
{
  def show()
  {
    println("Assignment 15.1")
  }
}
trait one extends MultipleInheritance
{
  override def show()
  {
    println("This won't be printed")
  }
}
trait two extends MultipleInheritance
{
  override def show()
  {
    println("Acadgild Scala Multiple Inheritance Example")
  }
}
class three extends one with two

object MainMulti{
  def main(args:Array[String]): Unit ={
    var c:three = new three
    c.show()
  }
}
```

```
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
Acadgild Scala Multiple Inheritance Example

Process finished with exit code 0
```

Example 2, in this example the object ***MainMulti*** called the trait ***one*** and see the result below,



```
package Assignment15_1

trait MultipleInheritance
{
  def show()
  {
    println("Assignment 15.1")
  }
}
trait one extends MultipleInheritance
{
  override def show()
  {
    println("This won't be printed")
  }
}
trait two extends MultipleInheritance
{
  override def show()
  {
    println("Acadgild Scala Multiple Inheritance Example")
  }
}
class three extends two with one

object MainMulti{
  def main(args:Array[String]): Unit ={
    var c:three = new three
    c.show()
  }
}
```

```
"C:\Program Files\Java\jdk1.8.0_144\bin\java" ...
This won't be printed

Process finished with exit code 0
```