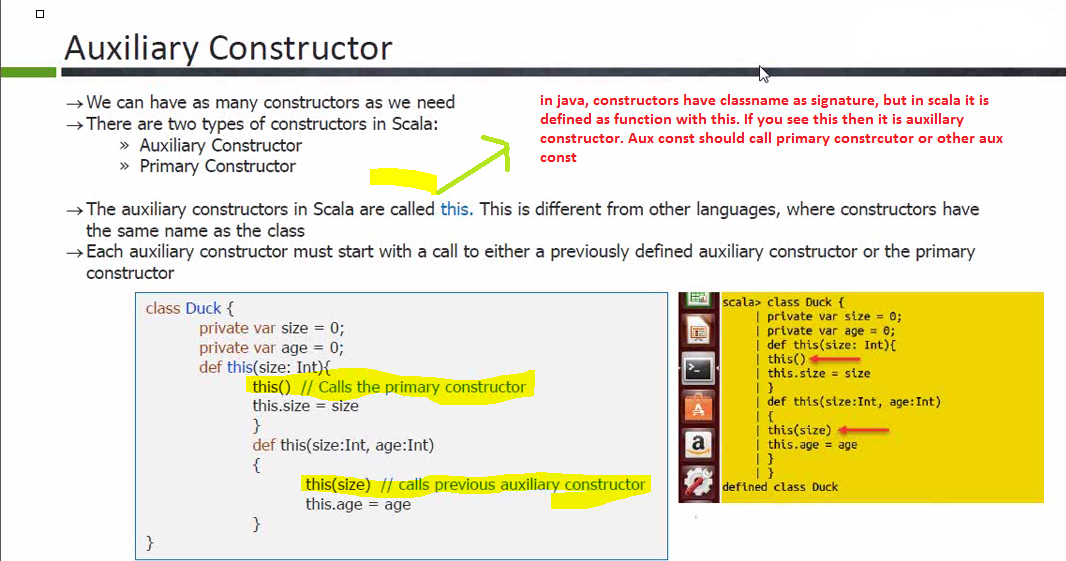
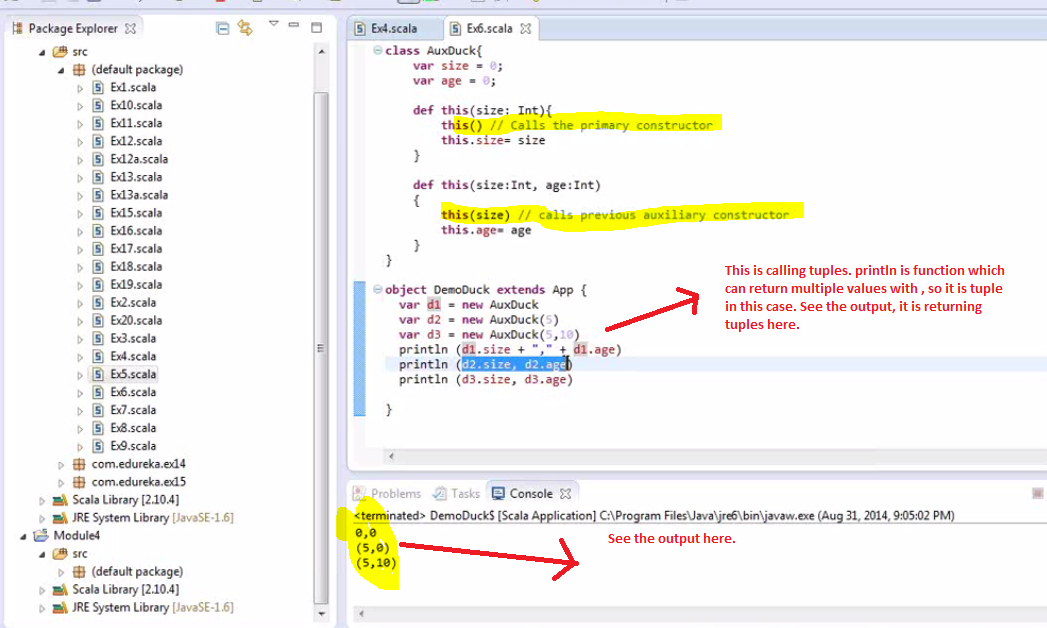
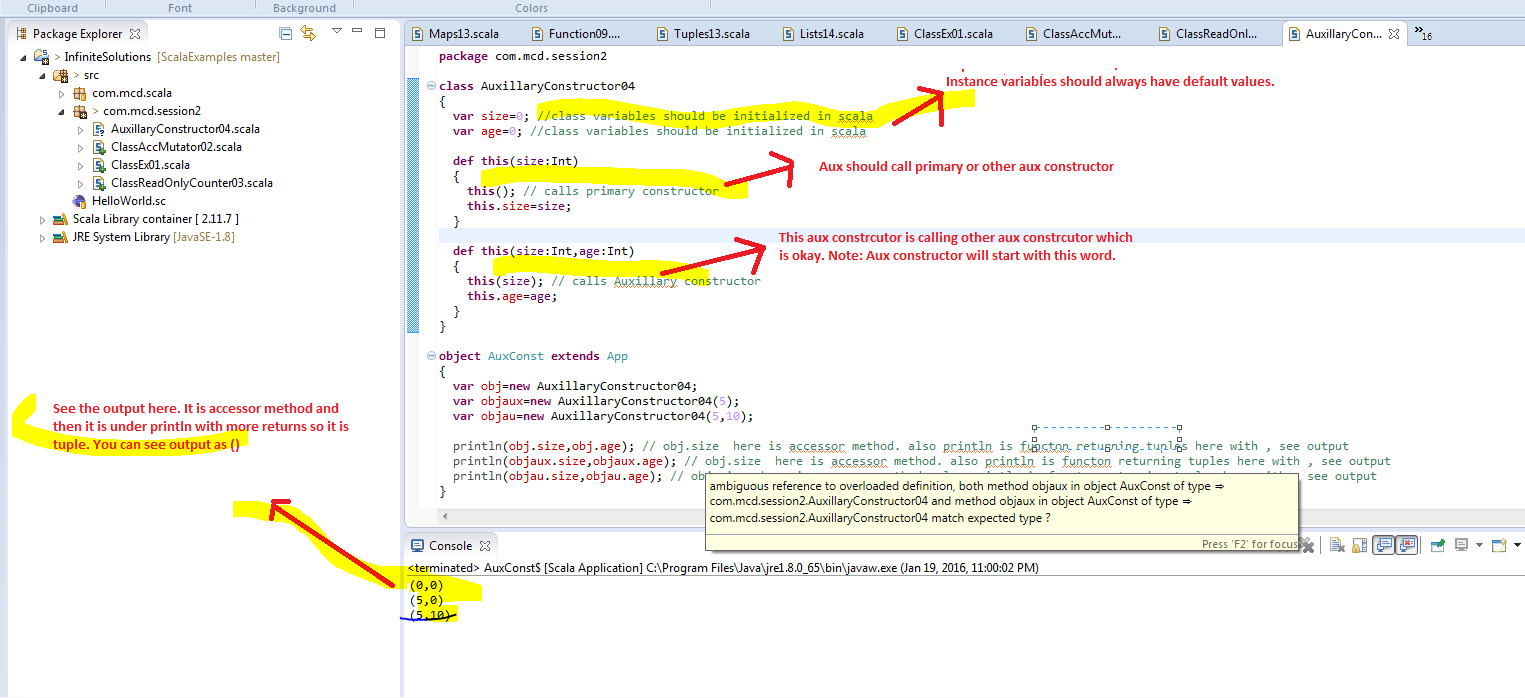
**Constructors in Scala**







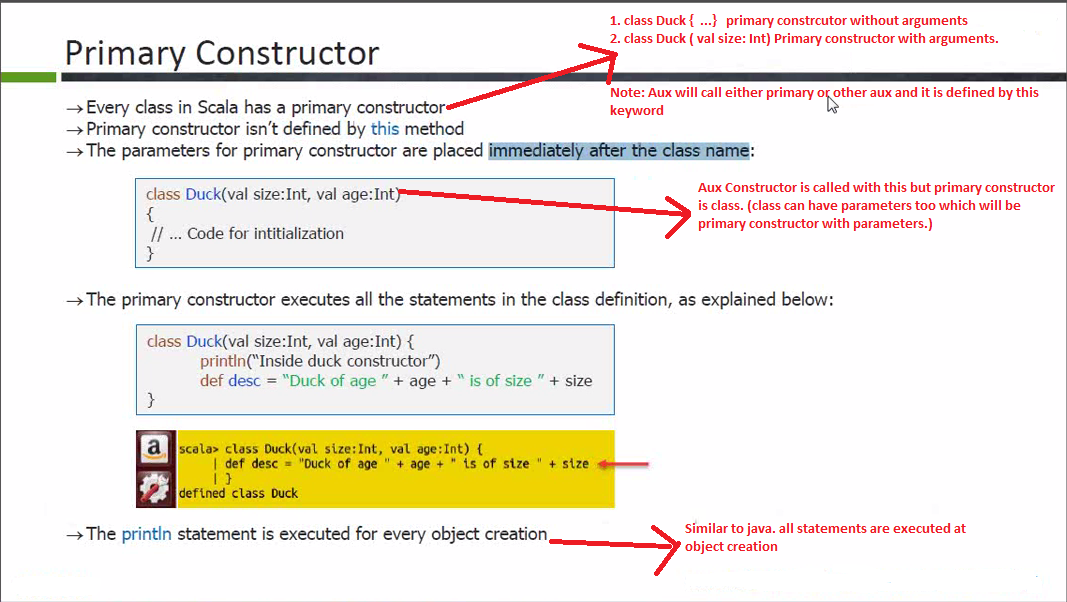
**Primary Constructors in Scala:**

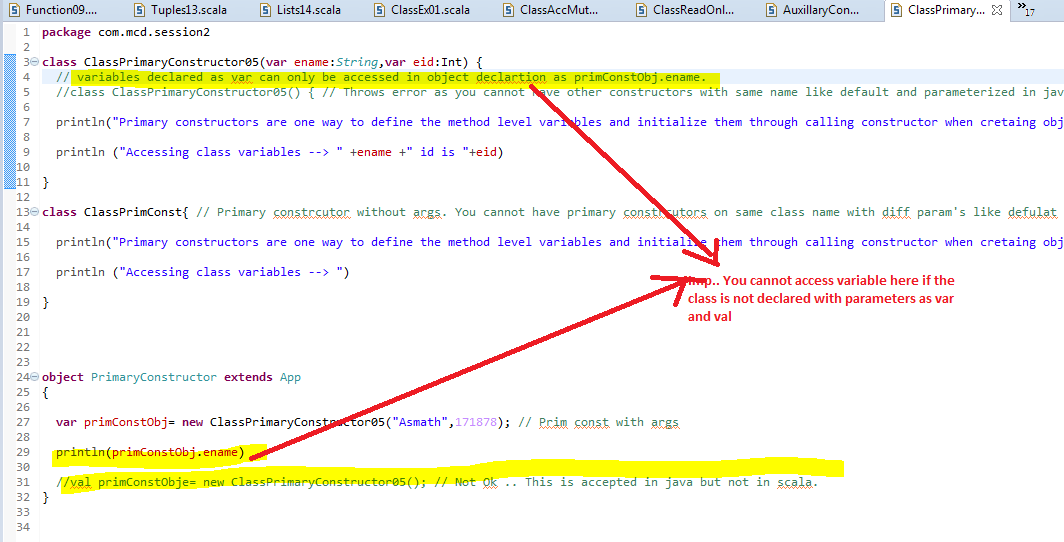
**Aux Constructor is called with this but primary constructor is class. (class can have parameters too which will be primary constructor with parameters.**

**1. class Duck { ...} primary constrcutor without arguments**

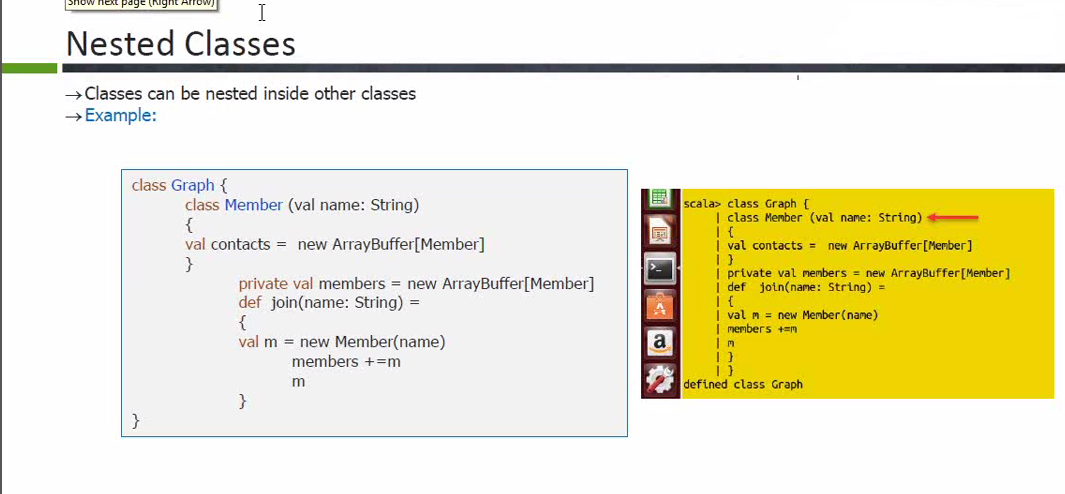
**2. class Duck ( val size: Int) Primary constructor with arguments.**

**Note: Aux will call either primary or other aux and it is defined by this keyword**



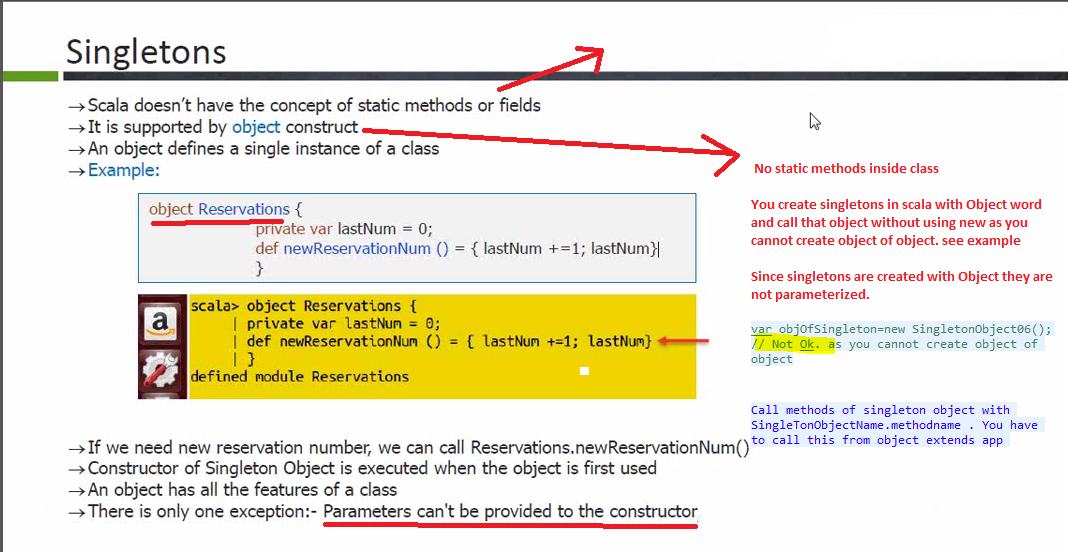


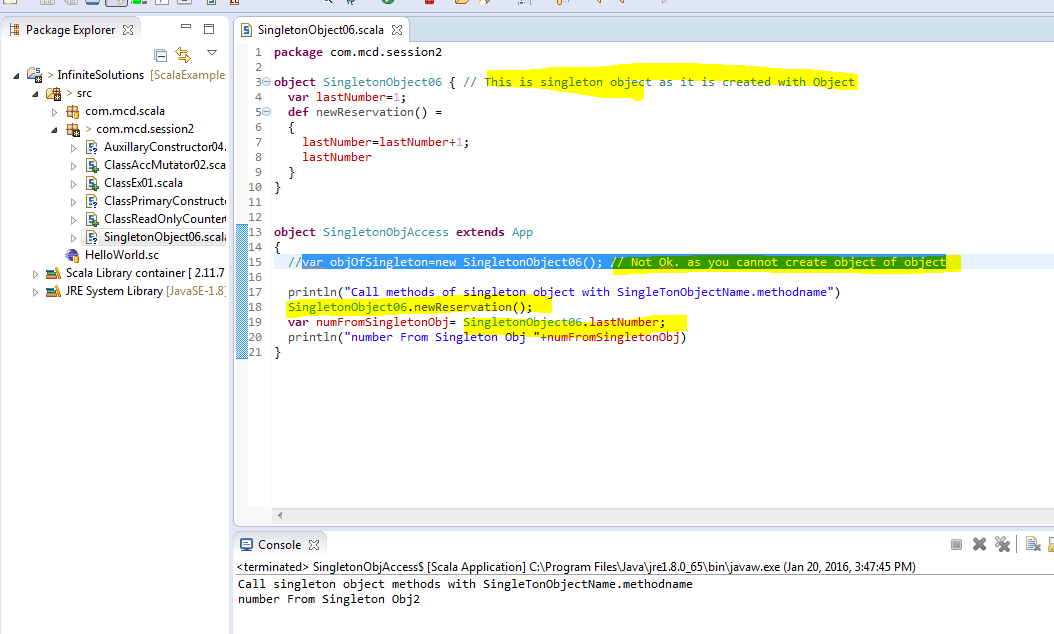
Nested Classes in Scala ---🡪



Static class cannot have instance variables in java. Anything passed to constructor becomes instance variable. Static class will not take parameters for the constructor.

**Scala Singleton and Companion Objects**





Scala classes cannot have static variables or methods. Instead a Scala class can have what is called a singleton object, or sometime a companion object.

A singleton object is declared using the object keyword. Here is an example:

object Main {

def sayHi() {

println("Hi!");

}

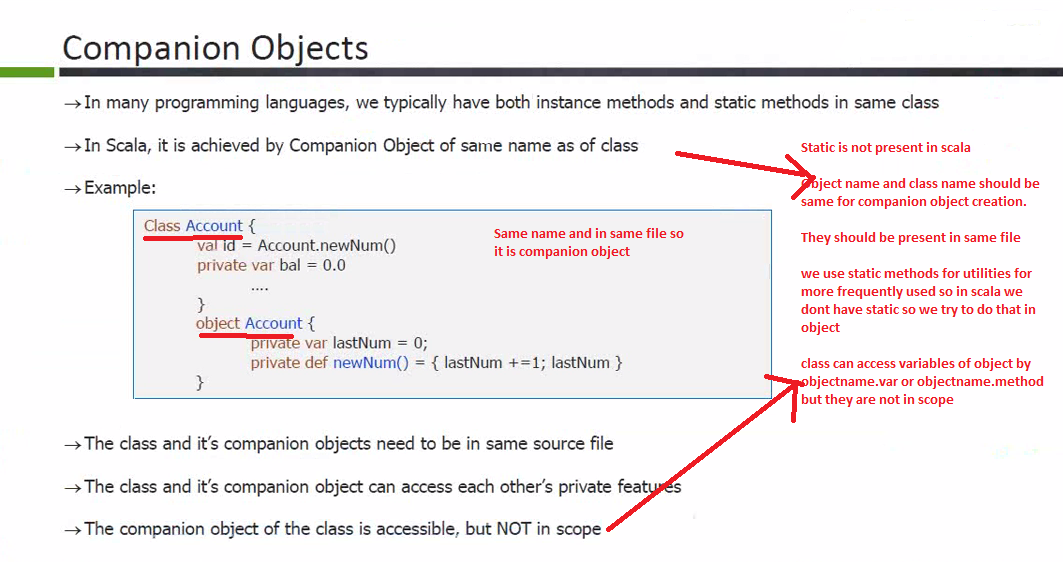
}

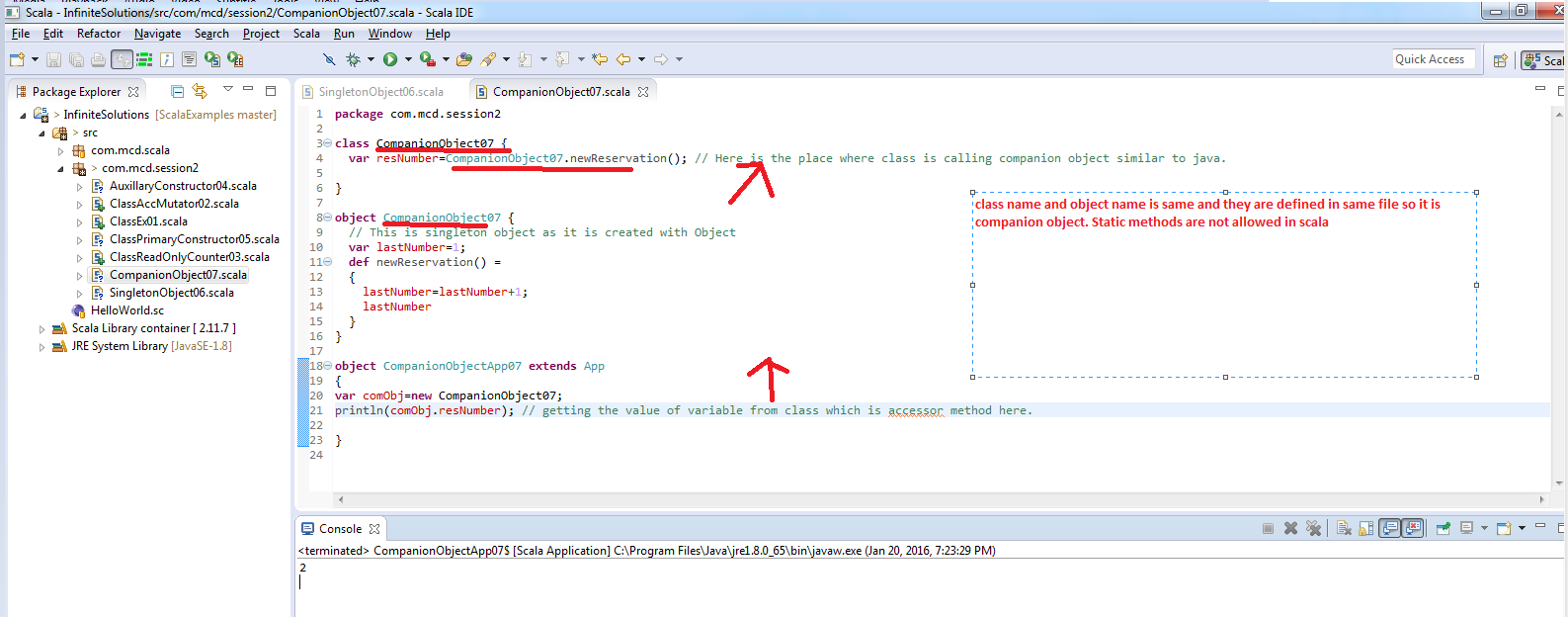
This example defines a singleton object called Main. You can call the method sayHi() like this:

Main.sayHi(); // Similar to Java by Asmath

Notice how you write the full name of the object before the method name. No object is instantiated. It is like calling a static method in Java, except you are calling the method on a singleton object instead.

**Companion Objects**

****



When a singleton object is named the same as a class, it is called a companion object. A companion object must be defined inside the same source file as the class. Here is an example:

class Main {

def sayHelloWorld() {

println("Hello World");

}

}

object Main {

def sayHi() {

println("Hi!");

}

}

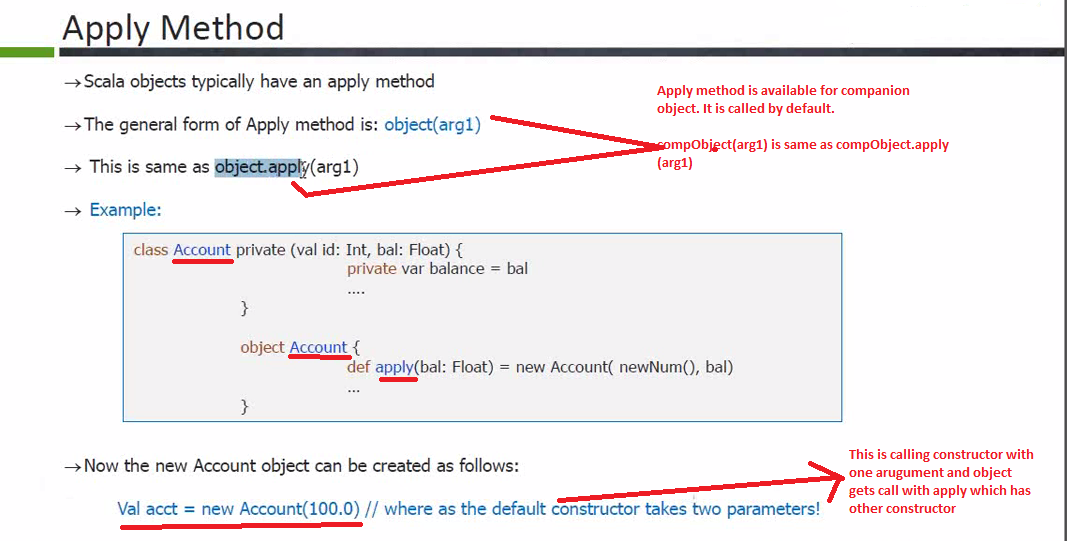
In this class you can both instantiate Main and call sayHelloWorld() or call the sayHi() method on the companion object directly, like this:

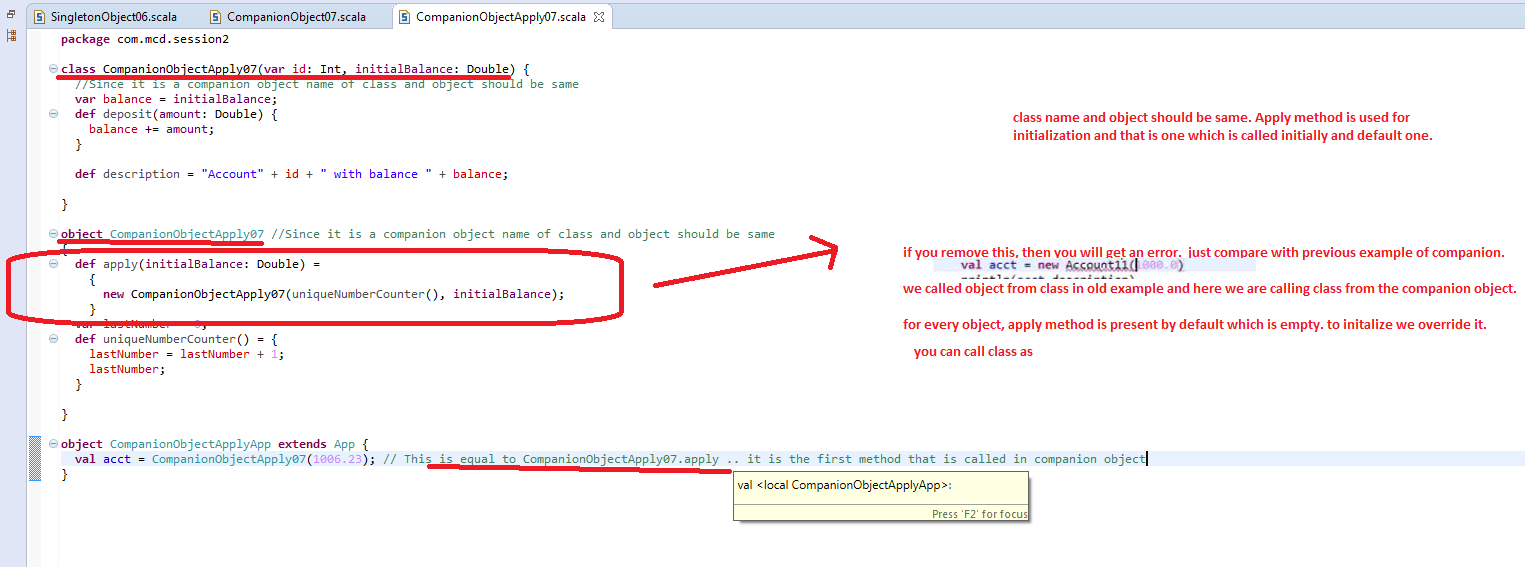
var aMain : Main = new Main();

aMain.sayHelloWorld();

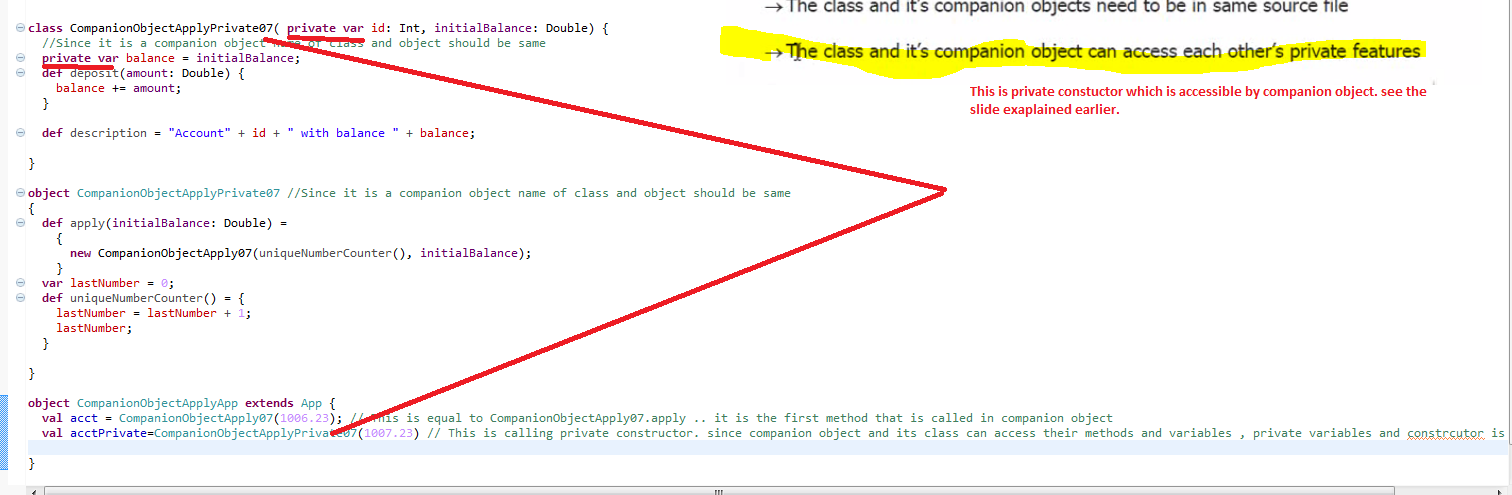
Main.sayHi();

**🡪Apply method on Companion Object----------------------🡪**

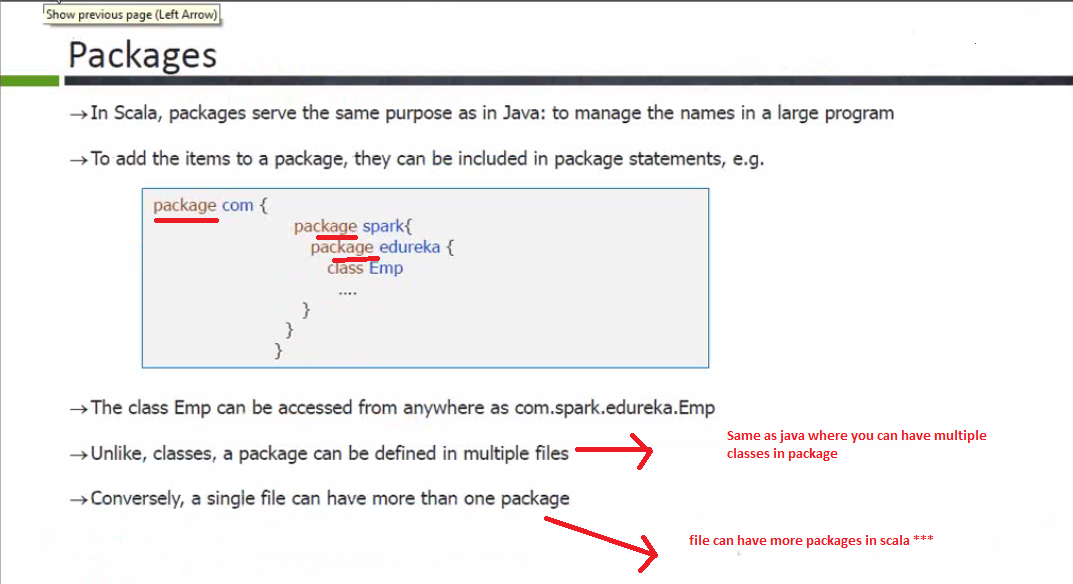


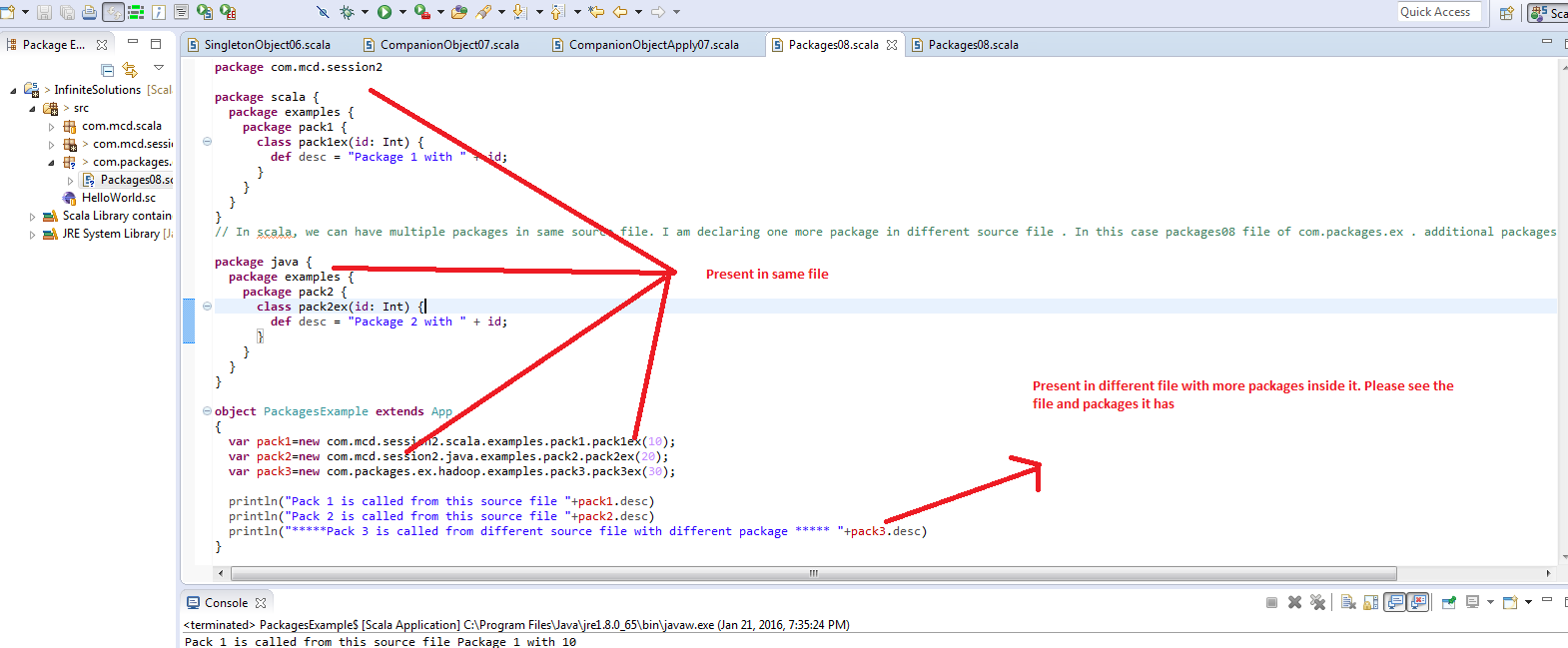


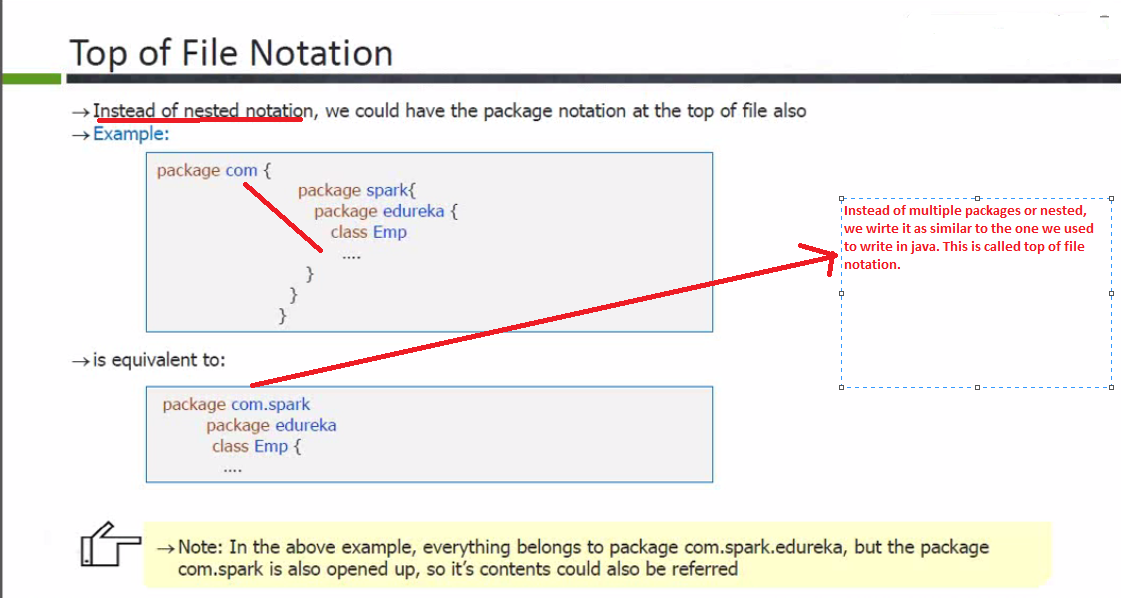
Below is example of private constructor. See the screenshot for more details.



**Packages in Scala --🡪**

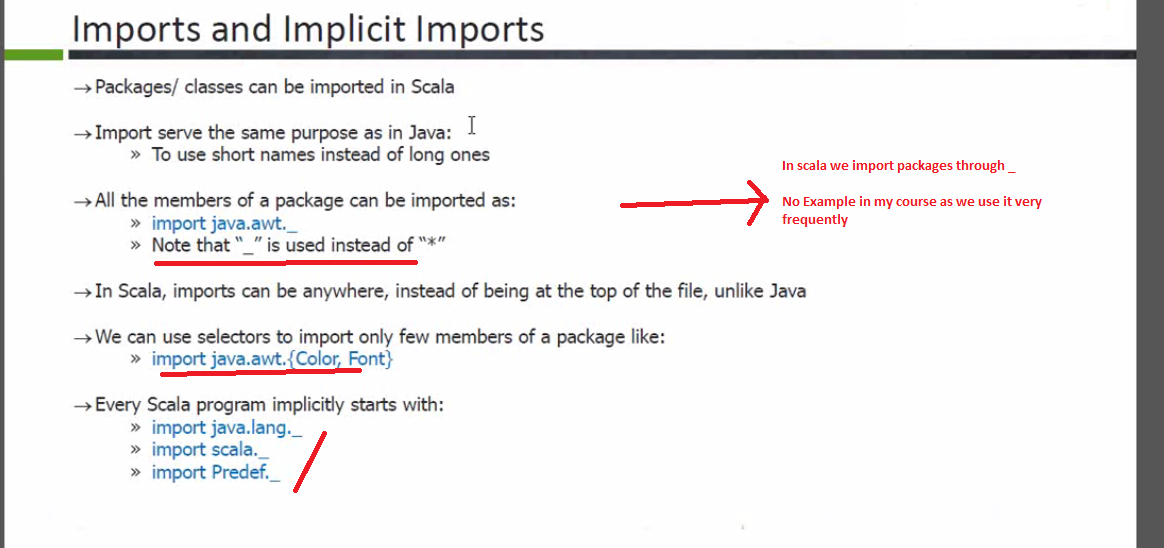




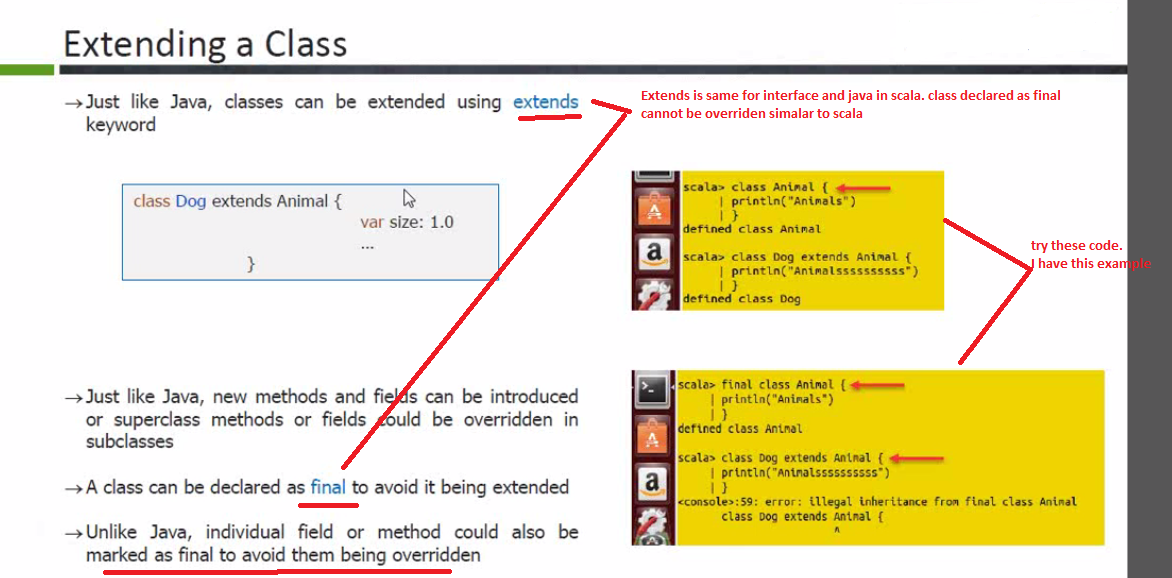


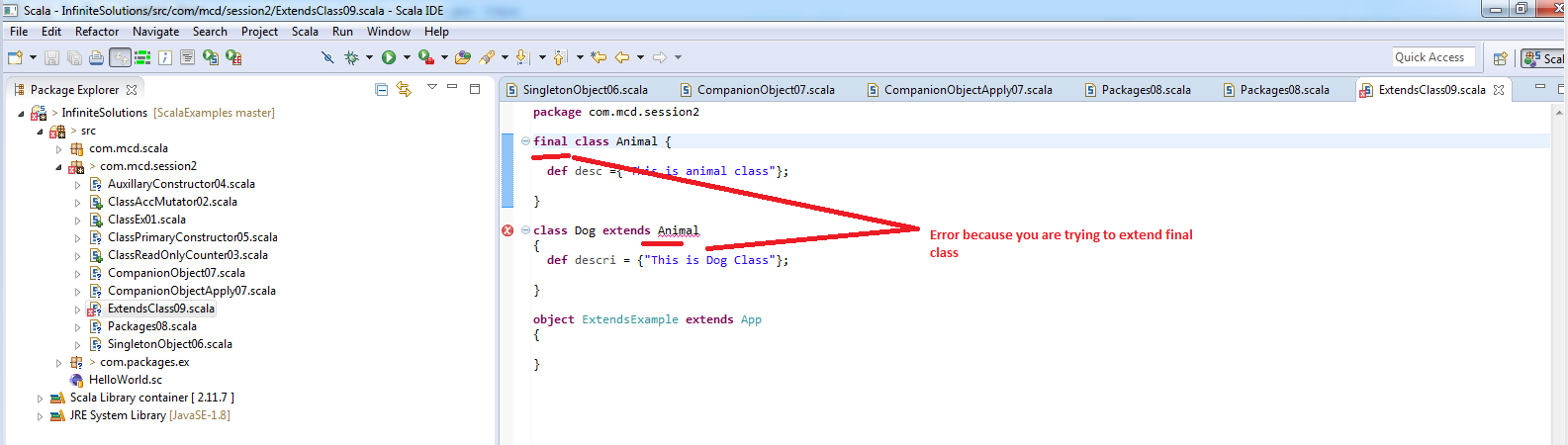
Example we wrote is called as top of file notation example.

**Imports in Scala -----🡪**

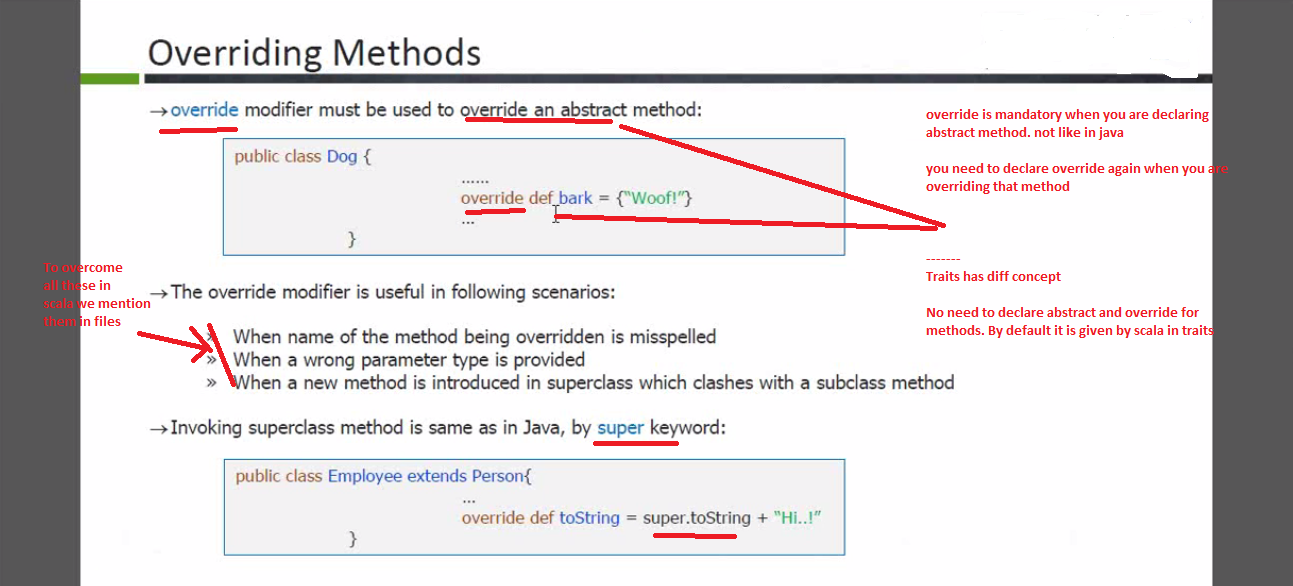


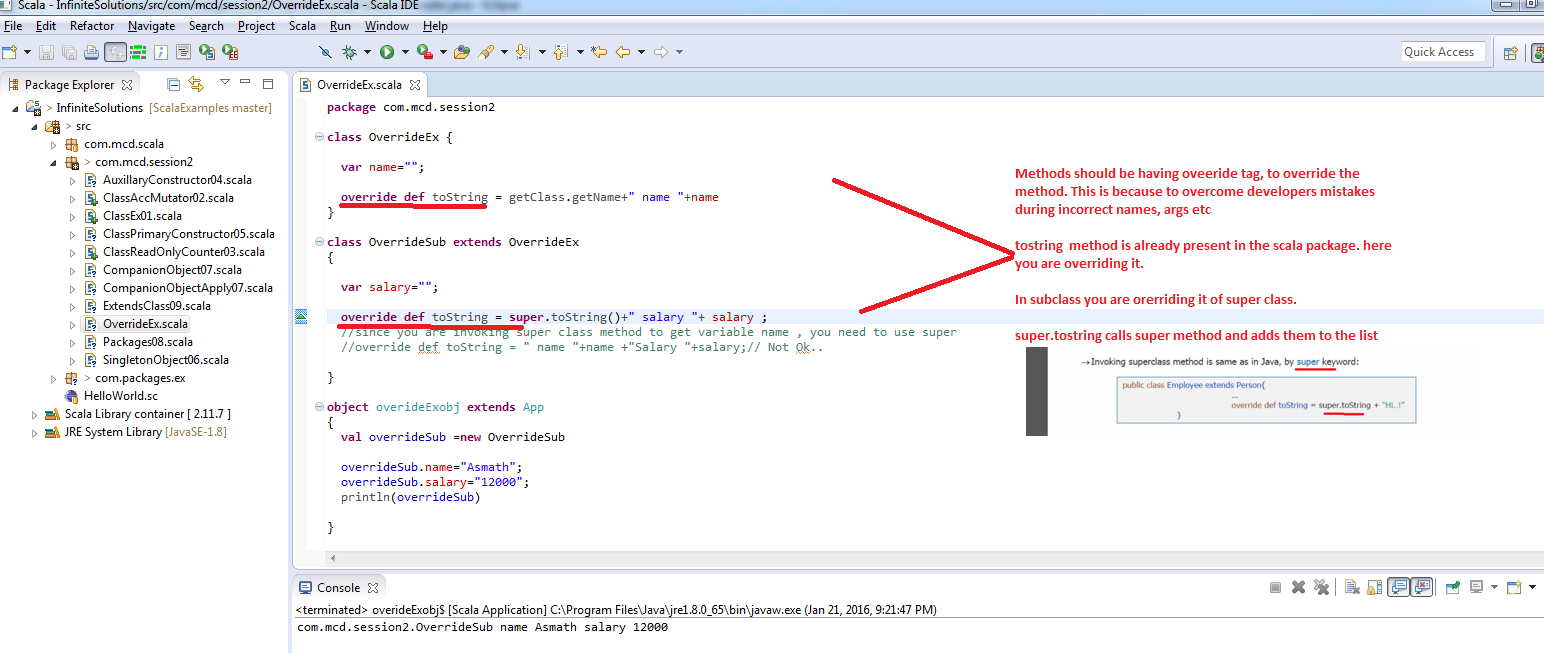
No Example for this in my course as it is not needed.





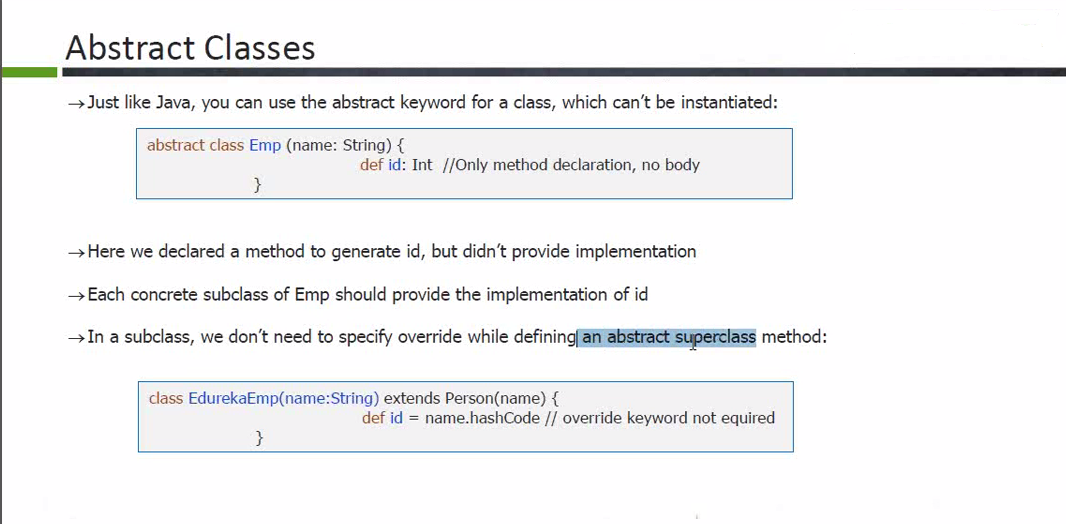
**Overriding methods in Scala ---------🡪**

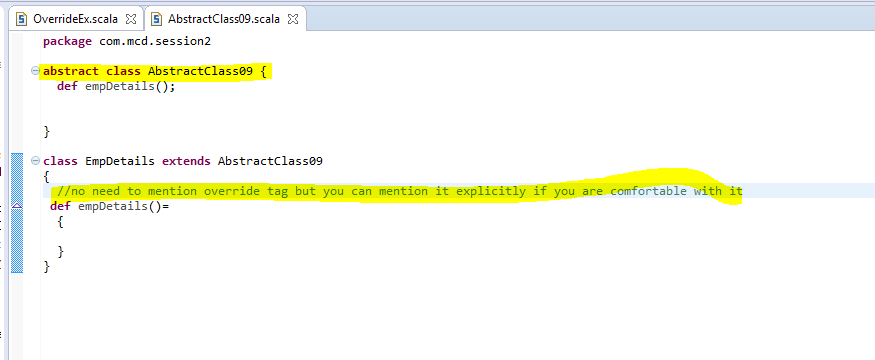


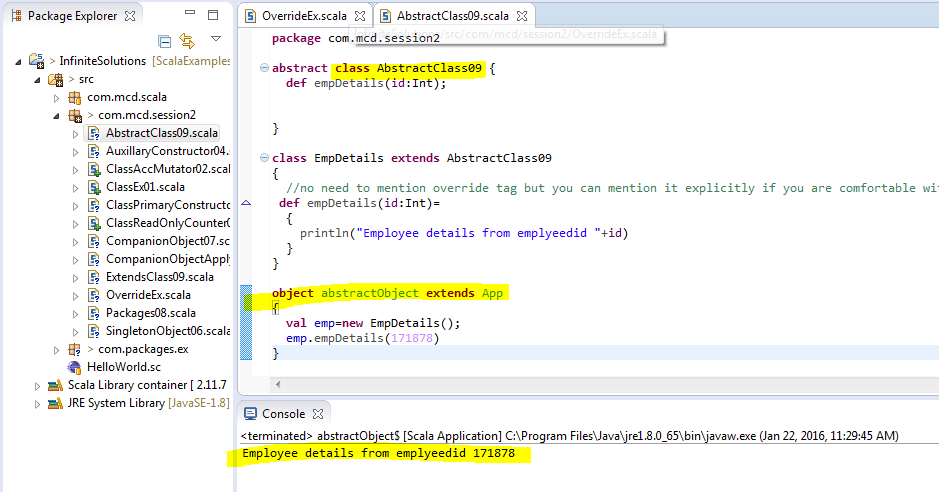


**Abstract Classes in Scala --🡪**

* Abstract classes should be declared as abstract. Can have both concrete and abstract methods.
* Classes extending abstract class need not write the override tag explicitly.







\*\*\* VV IMP

**See how you can extend class which has parameterized constructor.**

**Extend class which has parameterized constructor.**

