Scala Method Overriding

When a subclass has the same name method as defined in the parent class, it is known as method overriding. When subclass wants to provide a specific implementation for the method defined in the parent class, it overrides method from parent class.

In scala, you must use either override keyword or override annotation to override methods from parent class.

Scala Method Overriding Example 1

```
class Vehicle{
   def run(){
      println("vehicle is running")
   }
}
class Bike extends Vehicle{
    override def run(){
      println("Bike is running")
   }
}

object MainObject{
   def main(args:Array[String]){
      var b = new Bike()
      b.run()
   }
}
```

Output:

```
Bike is running
```

Real example of method overriding

Flowchart

Scala Inheritance 3

Scala Method Overriding Example 2

This example shows how subclasses override the method of parent class.

```
class Bank{
    def getRateOfInterest()={
       0
    }
  }

  class SBI extends Bank{
    override def getRateOfInterest()={
     8
    }
  }

  class ICICI extends Bank{
    override def getRateOfInterest()={
       7
    }
  }

  class AXIS extends Bank{
    override def getRateOfInterest()={
        9
      }
    }

    object MainObject{
      def main(args:Array[String]){
        var s=new SBI();
        var i=new ICICI();
        var a=new AXIS();
        println("SBI Rate of Interest: "+s.getRateOfInterest());
        println("ICICI Rate of Interest: "+i.getRateOfInterest());
        println("AXIS Rate of Interest: "+a.getRateOfInterest());
      }
    }
```

Output:

```
SBI Rate of Interest: 8
ICICI Rate of Interest: 7
AXIS Rate of Interest: 9
```

Scala Field Overriding

In scala, you can override fields also but it has some rules that need to be followed. Below are some examples that illustrate how to override fields.

Scala Field Overriding Example1

```
class Vehicle{
  var speed:Int = 60
}
```

```
class Bike extends Vehicle{
   var speed:Int = 100
    def show(){
        println(speed)
    }
}
object MainObject{
    def main(args:Array[String]){
        var b = new Bike()
        b.show()
    }
}
```

Output:

```
Error - variable speed needs 'override' modifier
```

In scala, you must use either override keyword or override annotation when you are overriding methods or fields of super class. If you don't do this, compiler reports an error and stops execution of program.

Scala Field Overriding Example2

```
class Vehicle{
     val speed:Int = 60


}
class Bike extends Vehicle{
   override val speed:Int = 100     // Override keyword
    def show(){
        println(speed)
    }
}
object MainObject{
    def main(args:Array[String]){
        var b = new Bike()
        b.show()
    }
}
```

Output:

```
100
```

In scala, you can override only those variables which are declared by using val keyword in both classes. Below are some interesting examples which demonstrate the whole process.

Scala Field Overriding Example3

```
class Vehicle{
     var speed:Int = 60
}
class Bike extends Vehicle{
   override var speed:Int = 100
    def show(){
        println(speed)
    }
}
object MainObject{
    def main(args:Array[String]){
        var b = new Bike()
        b.show()
    }
}
```

Output:

```
variable speed cannot override a mutable variable
```

Scala Field Overriding Example4

```
class Vehicle{
     val speed:Int = 60


}


class Bike extends Vehicle{
   override var speed:Int = 100
    def show(){
        println(speed)
    }
}


object MainObject{
    def main(args:Array[String]){
        var b = new Bike()
        b.show()
    }
}
```

Output:

```
Error - variable speed needs to be a stable, immutable value
```