SCALA ACCESS MODIFIERS

Members of packages, classes, or objects can be labeled with the access modifiers **private** and **protected**, and if we are not using either of these two keywords, then access will be assumed as **public**. These modifiers restrict accesses to the members to certain regions of code. To use an access modifier, you include its keyword in the definition of members of package, class or object as we will see in the following section.

Private members:

A **private** member is visible only inside the class or object that contains the member definition. Following is the example:

```
class Outer {
   class Inner {
      private def f() { println("f") }
      class InnerMost {
         f() // OK
      }
   }
   (new Inner).f() // Error: f is not accessible
}
```

In Scala, the access *newInner*.f is illegal because f is declared private in Inner and the access is not from within class Inner. By contrast, the first access to f in class InnerMost is OK, because that access is contained in the body of class Inner. Java would permit both accesses because it lets an outer class access private members of its inner classes.

Protected members:

A **protected** member is only accessible from subclasses of the class in which the member is defined. Following is the example:

```
package p {
   class Super {
      protected def f() { println("f") }
   }
   class Sub extends Super {
      f()
   }
   class Other {
     (new Super).f() // Error: f is not accessible
   }
}
```

The access to f in class Sub is OK because f is declared protected in Super and Sub is a subclass of Super. By contrast the access to f in Other is not permitted, because Other does not inherit from Super. In Java, the latter access would be still permitted because Other is in the same package as Sub.

Public members:

Every member not labeled private or protected is public. There is no explicit modifier for public members. Such members can be accessed from anywhere. Following is the example:

```
class Outer {
   class Inner {
      def f() { println("f") }
      class InnerMost {
         f() // OK
      }
   }
```

```
   (new Inner).f() // OK because now f() is public
}
```

Scope of protection:

Access modifiers in Scala can be augmented with qualifiers. A modifier of the form private[X] or protected[X] means that access is private or protected "up to" X, where X designates some enclosing package, class or singleton object. Consider the following example:

```
package society {
   package professional {
      class Executive {
         private[professional] var workDetails = null
         private[society] var friends = null
         private[this] var secrets = null

         def help(another : Executive) {
            println(another.workDetails)
            println(another.secrets) //ERROR
         }
      }
   }
}
```

Note the following points from the above example:

- Variable *workDetails* will be accessible to any class within the enclosing package *professional*.
- Variable *friends* will be accessible to any class within the enclosing package *society*.
- Variable *secrets* will be accessible only on the implicit object within instance methods *this*.