Inheritance in Kotlin

In Kotlin, classes are final by default. So if you want to create a subclass of a class, then you must mark the class with `open` modifier.

Also, there is no `extends` keyword in Kotlin. So to inherit a class, you must use a colon after the class name to inherit a base class.

Syntax of inheritance

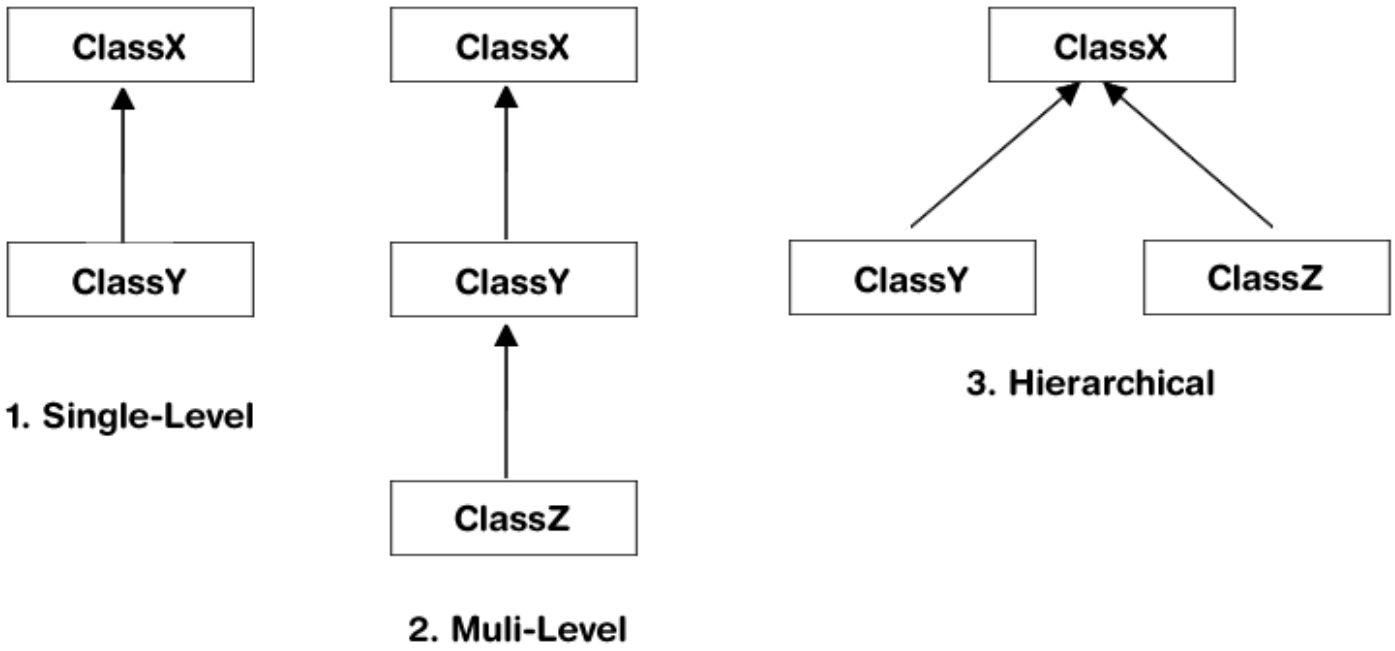
```
open class BaseClass{
   //Properties and Methods
}
class DerivedClass: BaseClass(){
   //Properties and Methods
}
```

Types of Inheritance in Kotlin

Just like Java, Kotlin supports three types of inheritance which are listed below-

1. **Single-Level Inheritance**
2. **Multi-Level Inheritance**
3. **Hierarchical Inheritance**

Note- If you want to implement multiple and hybrid inheritance, then you must use interface.



Single-Level Inheritance

```
open class Father{
   fun fatherIdentity(){
      println("I am a Father.")
   }
}
class Son: Father(){
   fun sonIdentity(){
      println("I am a Son.")
   }
}
fun main(args: Array<String>) {
   val s = Son()
   s.fatherIdentity()
   s.sonIdentity()
}
```

```
I am a Father.
I am a Son.
```

Multi-Level Inheritance

```
open class Father{
   fun fatherIdentity(){
      println("I am a Father.")
   }
}
open class Son: Father(){
   fun sonIdentity(){
      println("I am a Son.")
   }
}
class GrandSon: Son(){
   fun grandsonIdentity(){
      println("I am a GrandSon.")
   }
}
fun main(args: Array<String>) {
   val g = GrandSon()
   g.fatherIdentity()
   g.sonIdentity()
   g.grandsonIdentity()
}
```

```
I am a Father.
I am a Son.
I am a GrandSon.
```

Hierarchical Inheritance

```
open class Father{
   fun fatherIdentity(){
      println("I am a Father.")
   }
}
class Son: Father(){
   fun sonIdentity(){
      println("I am a Son.")
   }
}
class Daughter: Father(){
   fun daughterIdentity(){
      println("I am a Daughter.")
   }
}
fun main(args: Array<String>) {
   val s = Son()
   s.fatherIdentity()
   s.sonIdentity()
   val d = Daughter()
   d.fatherIdentity()
   d.daughterIdentity()
}
```

```
I am a Father.
I am a Son.
I am a Father.
I am a Daughter.
```