Inheritance In Kotlin

Inheritance in Kotlin is a mechanism where a class (child or derived class) can inherit properties and functions from another class (parent or base class). This allows you to reuse and extend the functionality of the parent class in the child class.

Key Concepts in Inheritance:

1. **Base (Parent) Class**: The class whose properties and functions are inherited by another class.
2. **Derived (Child) Class**: The class that inherits from the base class.

The `open` Keyword:

- In Kotlin, classes and methods are **final** by default, which means they cannot be inherited or overridden.
- To make a class inheritable, we use the `open` keyword and to make method overridden we use `open`.
- Similarly, if you want a method or property in a class to be overridden by a subclass, you must mark it with `open`.

Constructor Call in Parent Class:

- When a child class is created, the parent class's constructor is called first.
- If the parent class has a **primary constructor**, it is necessary to call it in the child class using the parent class name with constructor parameters (if any).
- If the parent class has a **secondary constructor**, you can either call it or not, depending on your use case.

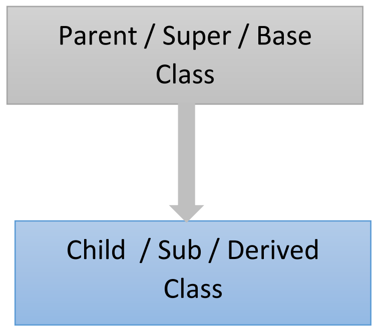
Example Code:

```
// Parent (Base) open class- make the class and all its property and method inheritable.
open class Animal(val name: String) {
    // Open function so it can be overridden
    open fun sound() {
        println("Animal makes a sound")
    }

    // Non-open function, cannot be overridden but is inherited
    fun eat() {// internal declaration- public final fun eat(){}
        println("$name is eating")
    }
}

// Child (Derived) class              //parent primary constructor called
class Dog(name: String, val breed: String) : Animal(name) {

// Overriding the open function
    override fun sound() {
        println("Dog barks")
    }
}

fun main() {
    val myDog = Dog("Buddy", "Golden Retriever")
    myDog.sound()  // Output: Dog barks
    myDog.eat()    // Output: Buddy is eating
}
```

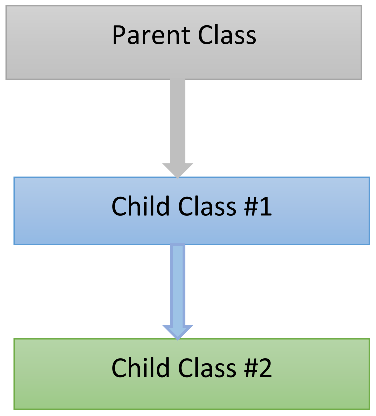
Types of Inheritance

Like java, Kotlin support following types of inheritance –

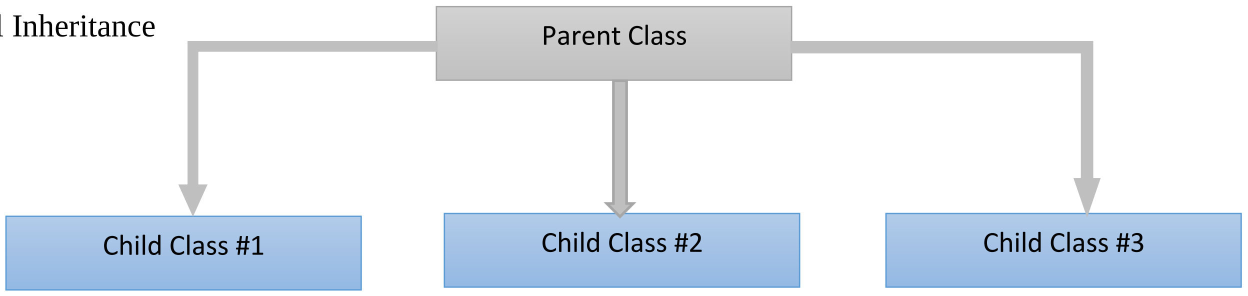
Simple Inheritance



Multilevel Inheritance



Hierarchical Inheritance



Multiple and Hybrid Inheritance is not allowed due to ambiguity happening in Child class, but it can be done by use of interface that provides only single implementation.