Abstraction, Abstract Methods & Abstract Class

Abstraction

Hide internal implementation and just highlight the set of services, is called abstraction.
By using *abstract classes* and *interfaces* we can implement abstraction.

Example:

By using ATM GUI screen bank people are highlighting the set of services what they are offering without highlighting internal implementation.

The main advantages of Abstraction are:

1) We can achieve security as we are not highlighting our internal implementation.
2) Enhancement will become very easy because without effecting end user we can able to perform any type of changes in our internal system.
3) It provides more flexibility to the end user to use system very easily.
4) It improves maintainability of the application.

Abstract Modifier

Abstract is the modifier applicable only for methods and classes but not for variables.

Abstract Methods:
Even though we don't have implementation still we can declare a method with abstract modifier. That is abstract methods have only declaration but not implementation. Hence abstract method declaration should compulsorily end with semicolon.

Example:

public abstract void m1( );----------> valid

public abstract void m1( ){}----------->invalid

***Child classes are responsible to provide implementation for parent class abstract methods.**

Example:

```
abstract class vehicle
{
   public abstract int getNoOfWheels();
}

class Bus extends vehicle
{
    public int getNoOfWheels()
        {
            return 7;
        }
}

class Bike extends vehicle
{
    public int getNoOfWheels()
        {
            return 2;
        }
}
```

The main advantage of abstract methods is, by declaring abstract method in parent class we can provide guide lines to the child class such that which methods they should compulsorily implement.

Abstraction, Abstract Methods & Abstract Class

Abstract method never talks about implementation whereas if any modifier talks about implementation, it is always illegal combination.
The abstract method m can only set a visibility modifier, one of public or protected

Abstract class

For any java class if we are not allowed to create an object such type of class, we have to declare with abstract modifier that is for abstract class instantiation is not possible.

Example:

```
abstract class Test
{
      public static void main(String args[ ] )
        {
          Test t = new Test ( );
        }
}
```

Output:

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem: Cannot instantiate the type Test
      at Test4.main(Test.java:5)
```

What is the difference between abstract class and abstract method?

If a class contain at least on abstract method then compulsory the corresponding class should be declare with abstract modifier. Because implementation is not complete and hence we can't create object of that class.

Even though class doesn't contain any abstract methods still we can declare the class as abstract that is an abstract class can contain zero no of abstract methods also.

Example: 1:

```
class Parent
{
    public void methodOne();
}
```

Output:
Compile time error. (missing method body, or declare abstract)

Example 2:

```
class Parent
{
    public abstract void methodOne(){}
}
```

Output:
Compile time error. (abstract methods cannot have a body)

Example 3:

```
class Parent
{
    public abstract void methodOne();
}
```

Abstraction, Abstract Methods & Abstract Class

Output:
Compile time error. (Parent is not abstract and does not override abstract method methodOne() in Parent class)

*****If a class extends any abstract class, then compulsory, we should provide implementation for every abstract method of the parent class otherwise we have to declare child class as abstract.**

Example:

```
abstract class Parent
{
        public abstract void methodOne();
        public abstract void methodTwo();
}
class child extends Parent
{
        public voidmethodOne(){}
}
```

Output:
Compile time error. (child is not abstract and does not override abstract method methodTwo() in Parent class)

*****If we declare class child as abstract then the code compiles fine but child of child is responsible to provide implementation for methodTwo().**

Accesses Constructor of Abstract Classes

An abstract class can have constructors like the regular class. And, we can access the constructor of an abstract class from the subclass using the super keyword.

Key Points to Remember

- We use the abstract keyword to create abstract classes and methods.
- An abstract method doesn't have any implementation (method body).
- A class containing abstract methods should also be abstract.
- We cannot create objects of an abstract class.
- To implement features of an abstract class, we inherit subclasses from it and create objects of the subclass.
- A subclass must override all abstract methods of an abstract class. However, if the subclass is declared abstract, it's not mandatory to override abstract methods.
- We can access the static attributes and methods of an abstract class using the reference of the abstract class.
- We cannot use the abstract keyword with the final.
- We cannot declare abstract methods as private.
- We cannot declare abstract methods as static.