**Interfaces**

Definition

1. Any service requirement specification is considered as interface.
2. The contract between client and service provider is known as interface.
3. It is completely pure abstract.
4. Example – JDBC API, Servlet API

Interface declaration and implementation

1. If we are implementing the interface every method present in the interface should be implemented in the class. Otherwise make the class as abstract to avoid the compiler error.
2. Whenever we are implementing the interface methods compulsory those methods declare as public. Otherwise, we will get the compiler error.

Extend Vs Implements

1. A class can extend only one class at a time and can implement multiple interfaces at same time.
2. A class can extend a class and implement an interface simultaneously.
3. When we writing both implements and extends, we have to use extends first and implements next one.

Interface Methods

1. Every method present in the methods are always public and abstract when we are declaring or not.
2. To make method available to every implementation class, every method inside the interface is public.
3. The implement class is responsible to provide the implementation for interface methods, So those methods are abstract.
4. We can’t declare interface, with the following modifiers
   1. Private
   2. Protected
   3. strictfp
   4. Synchronized
   5. Native

Interface variables

1. To define requirement level constants, interface can contain variables
2. Every variables inside the interface always public static final
   1. Public = To make available to implementation class
   2. Static = Without existing object of implementation class can access the variables
   3. Final = The interface variables should not reassign inside the implementation class. If one implementation change the values other implementation may affect.
3. Private, protected, transient, and volatile are invalid modifiers for interface variables.
4. Inside the interface we can use the variable but we cannot modify the value. Also, we can have the variable on the implementation class with the same name.

Interface naming conflicts

1. If two interface contains, method with same name and same signature and same return type then the implementation class, we have to provide implementation for only one method.
2. If two interface contains, method with same name with different argument type then the implementation class, we have to provide implementation for both the methods and these methods will act as overloaded methods.
3. If two interface contains, method with same name and different return types then it is impossible to implement both the interfaces simultaneously.

Interface variable naming conflicts

1. Two interfaces can contain variables with the same name and there maybe chance of variable naming conflicts. We can solve this problem using interface name(A.name, b.Marks)

Marker Interface

1. If the interface doesn’t contain any methods and implementing the that interface our object will get some ability, such type of interfaces are called marker interface.
   1. Serializable
   2. Cloneable
   3. RandomAccess
   4. SingleThreadModel
2. Without having any method how the object will get the ability in the interface.
   1. Internally, JVM responsible to provide the required ability. To reduce the complexity of the programming and to make java as simple.
3. Is it possible to create our own marker interface?
   1. Yes, but customized JVM is required.

Adapter classes

1. Adapter class is a simple java class that implements an interface only with empty implementation.
2. If we implement an interface for each method of interface we have to provide implementation.
3. The problem in this approach, it increases the length of the code and reduces the readability.
4. We can solve this problem using adapted class.
5. Instead of implementing the interface, we have to provide implementation for required methods and we are not responsible to provide the implementation for every method.

Note – Marker interface, adapter class simplifies the complexity of the program.

|  |  |
| --- | --- |
| **Interface** | **Abstract class** |
| If we don’t know about the implementation and we just having the requirement specification, then we should go for interface | If we are talking about implementation but not completely then we can go for abstract class |
| Inside interface every method is public and abstract even if we are not declared | Every method present inside the abstract class need not be the public and abstract. We can have concrete methods also |
| We can’t declare with private, protected, final, static synchronized and strictfp modifiers | No restriction on abstract class method modifiers |
| Every interface variables are public, static, final implicitly | Every abstract class variable need not be public, static, final. |
| Interface variables cannot use private, protected, transient, volatile | No restriction for abstract class variables |
| Compulsory initialization at that time of declaration for every variable. | No restriction about declaration of the variables |
| Inside interface we can’t declare static and instance blocks | We can declare static and instance blocks |
| We cannot declare constructors | Can declare constructors |

Important conclusions

1. Anyway, we can’t create object for abstract class but abstract class can contain constructors, Those constructor will execute whenever creating object for child classes.
2. Either directly or indirectly we can’t create object for abstract classes.
3. The purpose of the constructor is to initialize the instance variables but in the interface there won’t be any instance variables, all the variables inside the interface are public, static, final. So the purpose of constructors is no need for interface. So interfaces don’t have constructors.
4. Whenever the child class objects created, the parent class object won’t be created, just parent class constructor will be executed for child object purpose.
5. Is it possible to replace the interface with abstract?
   1. Yes, But it is not good programming practice, If every methods are abstract then it is highly recommended to go with interface.