FAQ's on Interfaces

1. What is interface in Java and what are its uses?

Interface in Java is similar to a class, which may contain method's signature only but not bodies and it is a formal set of method and constant declarations that must be defined by the class that implements it. Interfaces are useful for:

- Declaring methods that one or more classes are expected to implement.
- Capturing similarities between unrelated classes without forcing a class relationship.
- Determining an object's programming interface without revealing the actual body of the class.

2. Class C implements interface 'I' containing method m1 and m2 declarations. Class C has provided implementation for method m2. Can an object of Class C be created?

No. Class C should provide implementation for all the methods in the interface 'I'. Since Class C didn't provided implementation for m1 method, it has to be declared as abstract. Abstract classes can't be instantiated.

3. Can a method inside an interface be declared as final?

No. Doing so will result in compilation error. "public" and "abstract" are the only applicable modifiers for the method declaration in an interface.

4. Can an interface implement another interface?

Interfaces doesn't provide implementation hence an interface cannot implement another interface.

5. Can an interface extend another interface?

Yes. An Interface can inherit another Interface, for that matter an Interface can extend more than one Interface.

6. Can a class extend more than one class?

No. A class can extend only one class but can implement any number of interfaces.

7. Can an interface be final?

No. Doing so will result in compilation error.

8. Can a class be defined inside an interface?

Yes. It's possible.

9. Can an interface be defined inside a class?

Yes. It's possible.

10. What is a Marker Interface?

An interface which doesn't have any declaration inside, but still enforces a mechanism is known as Marker Interface.

11. Can we define private and protected modifiers for variables in interfaces?

No. We cannot define private and protected modifiers for variables in interfaces

12. What is Externalizable?

Externalizable is an interface that extends Serializable Interface and sends data into Streams in compressed format. It has two methods, *writeExternal(ObjectOuput out)* and *readExternal(ObjectInput in).*

13. What modifiers are allowed for methods in an interface?

Only “public” and “abstract” modifiers are allowed for the methods in the interfaces.

14. When can an object reference be cast to an interface reference?

When the object implements the referenced interface, an object reference is cast to an interface reference.

15. What is the difference between an Interface and an Abstract class?

An abstract class can have instance methods that implement a default behavior. An Interface can only declare constants and instance methods, but cannot implement default behavior and all methods are implicitly abstract.

An interface has all public members and no implementation. An abstract class is a class which may have the usual flavors of class members (private, protected, etc.), but has some abstract methods.

16. Can an anonymous class be declared as implementing an interface and extending a class?

An anonymous class may implement an interface or extend a superclass, but may not be declared to do both.

17. What is an abstract class?

An abstract class is a class designed with implementation gaps for subclasses to fill in and is deliberately incomplete.