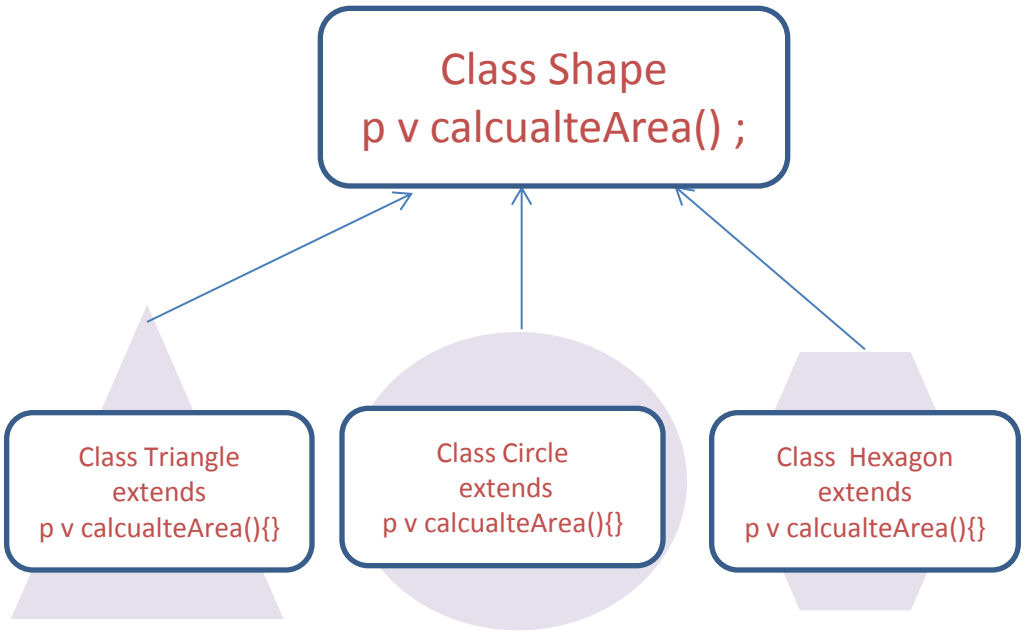
Abstract Classes & Interfaces

Authored by : Sangeeta Joshi ***Presented by: Sangeeta Joshi***

Agenda

- Abstract classes
- Interfaces

Abstract class



calArea() can be implemented in each subclass i.e. Traingle,Circle, Hexagon

- Method calArea() can not be implemented in Shape class.
- Such a method which is declared but not defined is called an ***abstract*** method
- So class Shape must be declared as abstract & Class Shape should not be instantiated
- Implementation of calArea() can be provided in subclasses i.e. Triangle , Circle, Hexagon

Abstract class

- **Java Abstract classes** are used to declare common characteristics of subclasses.
- Abstract classes are used to provide a template or design for concrete subclasses down the inheritance tree.
- Abstract classes are declared with the abstract keyword.
- One or more methods in an abstract class are declared but not defined.

Abstract class….continued

- Any class containing even a single method as abstract must be declared as abstract
- An abstract class may contain concrete methods as well.
- An abstract class ***can not be*** instantiated
- Sub class of an abstract class must implement all abstract methods of super class or it must also be declared as abstract
- Constructors & static methods can not be declared as abstract

Interfaces

```
Interface Printable
{
    public void print()
}
```

- an *interface* is a reference type, similar to a class, that can contain *only* constants & method signatures
- There are no method bodies inside an interface
- Interfaces cannot be instantiated

Interface

- Methods declared in an interface are by default abstract
- Variables declared in an interface are by default ***public ,static & final***
- Interfaces help to club non-related classes under one roof
- Interfaces provide support for multiple inheritance in Java. (They support *dynamic polymorphism* but there is no code reusability)
- Interfaces are some times known as “Programming by contract”. A class that implements the interface is bound to implement all the methods defined in Interface.
- Interfaces provide ‘Lose Coupling’

Interface

- If a class that implements an interface does not define all the methods of the interface,

  Then it must be declared abstract and the method definitions must be provided by the subclass that extends the abstract class.

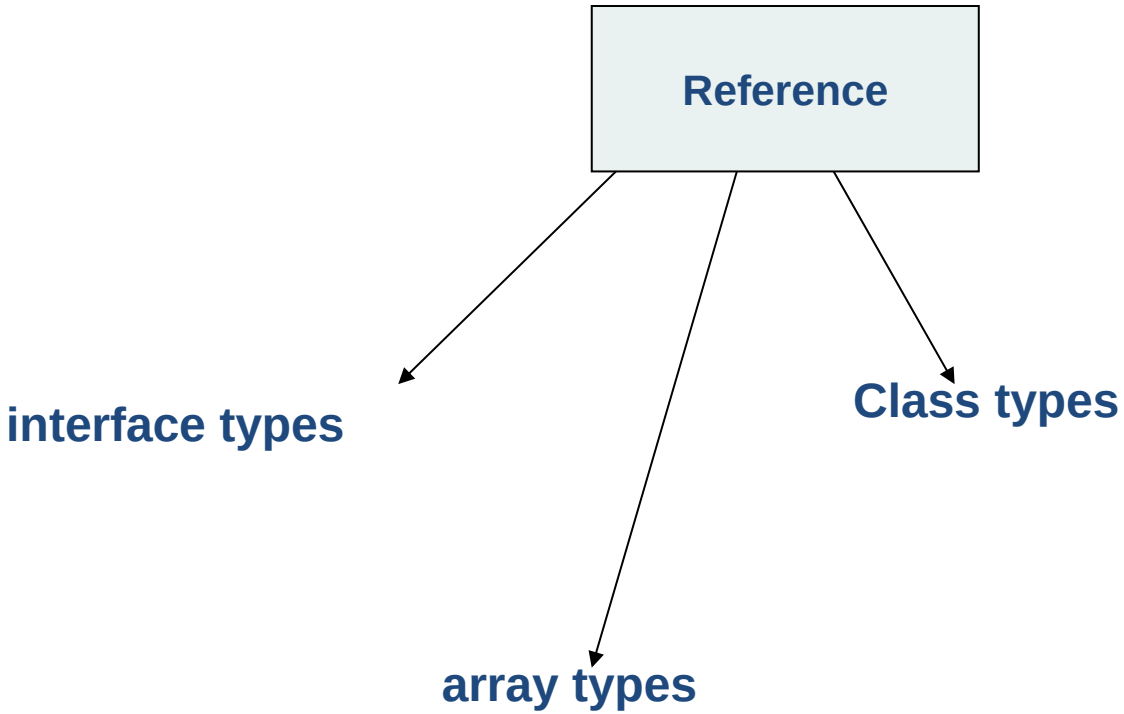
- The interface without any method declaration (empty interface) is called as Marker or Tagging interface.

  Ex: Serializable,Cloneable

Abstract class Vs. Interface

Abstract Classes	Interfaces
"Is a" type relationship exists between abstract super class & subclass extending it	Interfaces are used to club together Non related classes
Can contain implemented methods as well	Can contain only abstract methods
subclass can not extend from more than one abstract class	A class can implement no. of interfaces

Reference Types



Assignments

Any Questions?