Interfaces

- Multiple inheritance: interfaces
- Interfaces define only abstract methods and final fields.
- Interfaces do not specify any code to implement these methods and data fields contain only constants.
- Therefore it is the responsibility of the class that implements an interface to define the code for implementation of these methods.

Defining interfaces

- Syntax:

```
interface InterfaceName
{
   return-type method-name1(parameter list);
……….
return-type method-nameN(parameter list);
Type final-variablename1=value;
……..
Type final-variablenameN=value;
}
```

Defining interfaces

- Example:

```
interface Area
{
  float pi=3.14F;
  float compute(float x,float y);
  void show();
}
```

Implementing interfaces

- Syntax:

```
class classname implements InterfaceName
{
   //class-body
}
```

or

```
class classname extents superclass implements interface1,interface2,…
{
//class-body
}
```

Extending Interfaces

- Syntax:

interface InterfaceName2 **extends** InterfaceName1

{

//body

}

Extending Interfaces

- Example:

```
 interface ItemConstants
{
int code=1001;
String name=“Pencil”;
}
interface Item extends ItemConstants
{
void display();
}
```

Extending Interfaces

- Example:

```
 interface ItemConstants
{
int code=1001;
String name=“Pencil”;
}
interface ItemMethods
{
void display();
 }
interface Item extends ItemConstants,ItemMethods
{
……
}
```

Accessing Interface Variables

Interfaces can be used to declare a set of constants that can be used in different classes.

Difference between abstract class and Interface

Interface	Abstract
Java interface are implicitly abstract and cannot have implementations	A Java abstract class can have instance methods that implements a default behavior
Variables declared in a Java interface is by default final	An abstract class may contain non-final variables.
Members of a Java interface are public by default	A Java abstract class can have the usual flavors of class members like private, protected, etc
Java interface should be implemented using keyword "implements"	A Java abstract class should be extended using keyword "extends"
An interface can extend another Java interface only	an abstract class can extend another Java class and implement multiple Java interfaces.
Interface is absolutely abstract and cannot be instantiated	A Java abstract class also cannot be instantiated, but can be invoked if a main() exists.
java interfaces are slow as it requires extra indirection	Comparatively fast