Welcome to My Presentation

My Presentation Topic is

Class Object Interface

PRESENTED BY:

NAME : TUSHAR SARKAR

STUDENT ID: 18CSE035

SECOND YEAR FIRST SEMESTER

DEPARTMENT OF CSE,BSMRSTU,

GOPALGANJ-8100.

What is an Classes and Objects?

- Classes and objects are the two main aspects of object-oriented programming.

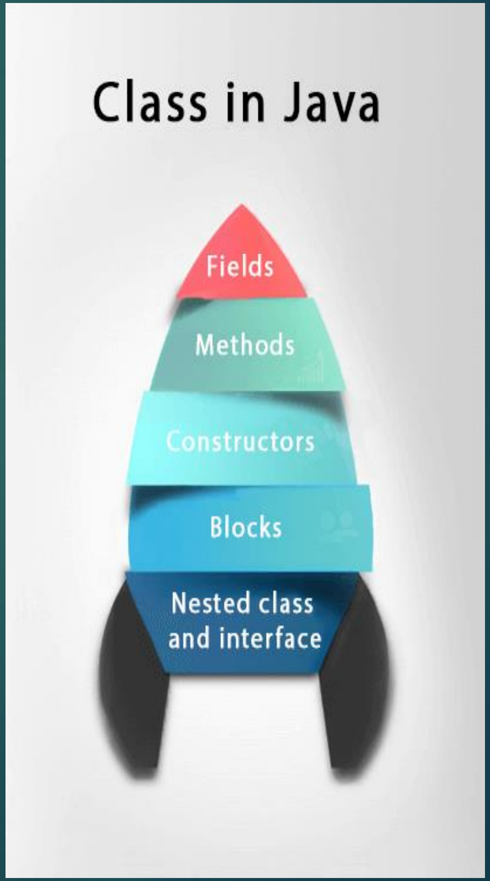
Class	Objects
Fruit	Orange Apple Mango

What is a class in java

- A class is a group of objects which have common properties.
- It is a template or blueprint from which objects are created.
- It is a logical entity. It can't be physical.

A class in Java can contain:

- Fields
- Methods
- Constructors
- Blocks
- Nested class and interface



Syntax to declare a class:

```
class <class_name>{
          field;
          method;
 }
```

- Method in Java is like a function which is used to expose the behavior of an object.
- Advantage of Method
  - Code Reusability
  - Code Optimization

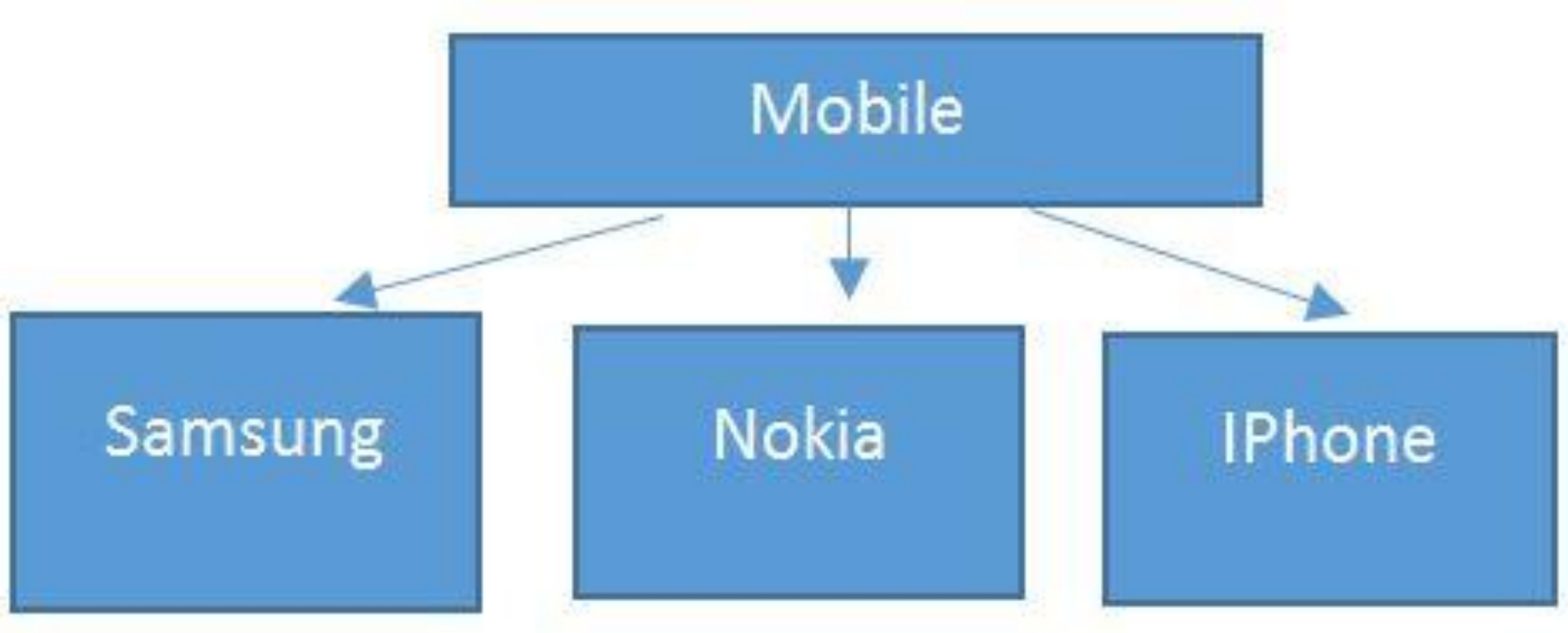
Create a Class

- Main.java

```
public class Main {
        int x = 5;
}
```

Create a class named "Main" with a variable x.

Real world examples of objects



Characteristics of Object

An object has three characteristics:

- **State**: represents the data (value) of an object.
- **Behavior**: represents the behavior (functionality) of an object such as deposit, withdraw, etc.
- **Identity**: An object identity is typically implemented via a unique ID. The value of the ID is not visible to the external user. However, it is used internally by the JVM to identify each object uniquely.

Object definitions

- An object is an instance of a class.
- A class is a template or blueprint from which objects are created.
- So, an object is the instance(result) of a class.

Create an Object

```
public class Main{
        public static void main(String[] args) {
                value f = new Value();    //create an Value object
                System.out.println(f.x);
        }
}
class Value{
        int x = 10;
}
```

Interface in Java

- An interface in Java is a blueprint of a class.
- It has static constants and abstract methods.

- There are mainly three reasons to use interface
  - It is used to achieve abstraction.
  - By interface, we can support the functionality of multiple inheritance.
  - It can be used to achieve loose coupling.

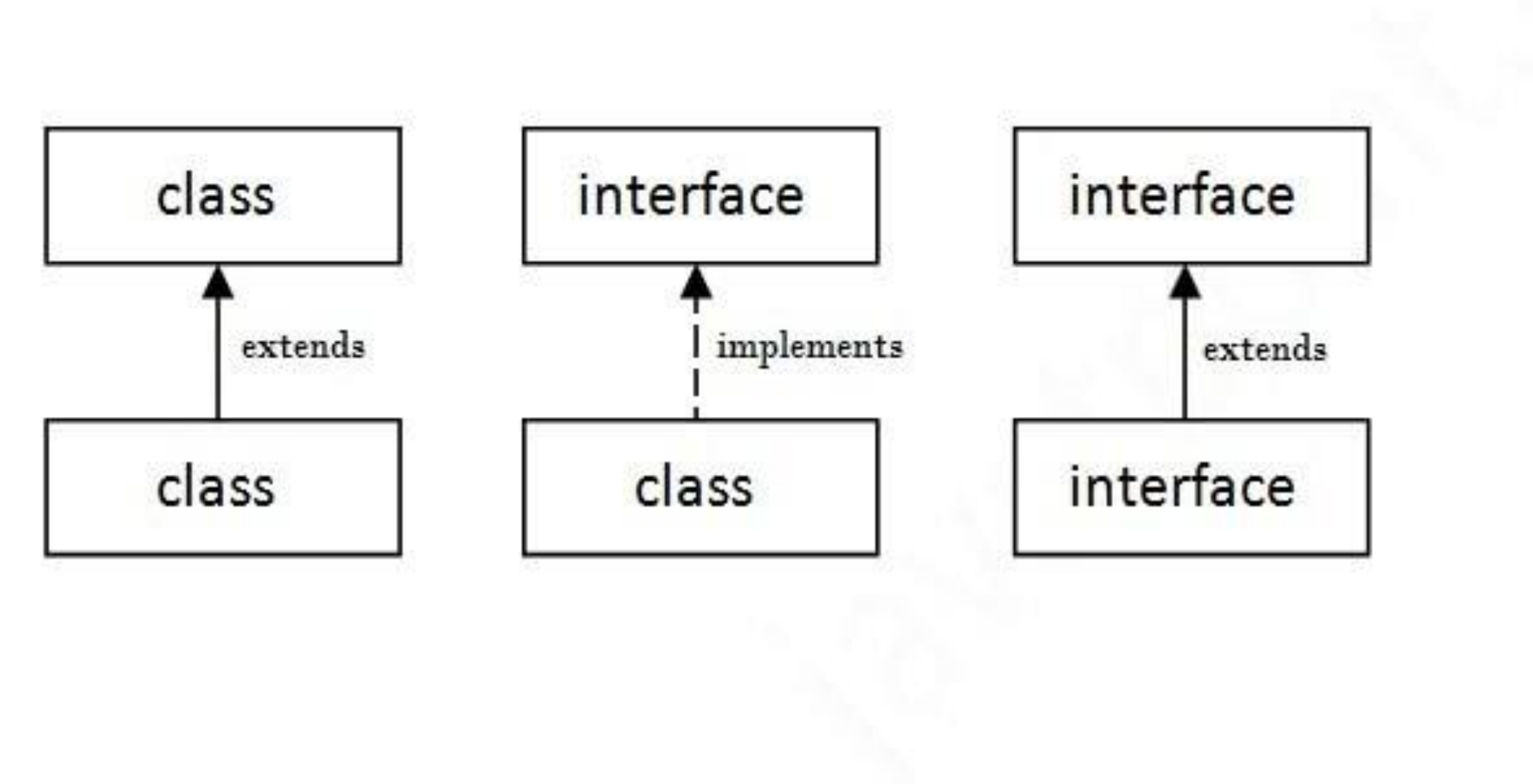
How to declare an interface?

- Syntax:

```
interface <interface_name>{

        // declare constant fields
        // declare methods that abstract
        // by default.
}
```

The relationship between classes and interfaces



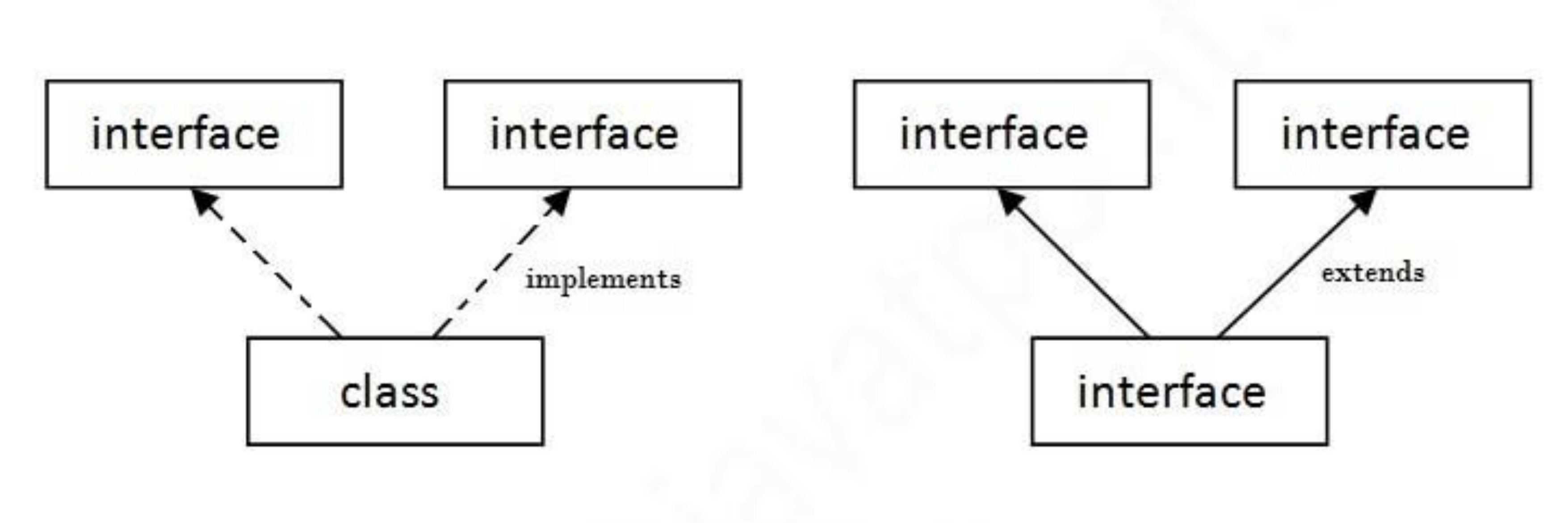
Simple code for interface

```
public class A {
        public static void main(String[] args) {
                demo f = new demo();
                f.print();
        }
}

interface printable {
        void print();
}

class demo implements printable {
        public void print() {
                System.out.println("Hello");
        }
}
```

Multiple inheritance in Java by interface



Multiple Inheritance in Java

Multiple inheritance Code

```
public class A{
        public static void main(String[] args) {
                demo f = new demo();
                f.print();
                f.show();
        }
}
interface printable{ void print();} //1st interface
interface showable{ void show();} // 2nd interface

class demo implements printable, showable {
        public void print() {
               System.out.println("Hello");
        }
        public void show(){
               System.out.println("Welcome");
        }
}
```

Interface inheritance

- A class implements an interface, but one interface extends another interface.

General code:	Interface extend another interface

interface printable{ void print();}
interface showable{ void show();}
 	
class demo implements showable,printable {
public void print() { System.out.println("Hello"); }
public void show(){ System.out.println("Welcome"); }
}
	interface printable{ void print();}
	interface showable extends printable{ void show();}
	
	class demo implements showable {
	public void print() { System.out.println("Hello"); }
	public void show(){ System.out.println("Welcome"); }
	}

THANK YOU