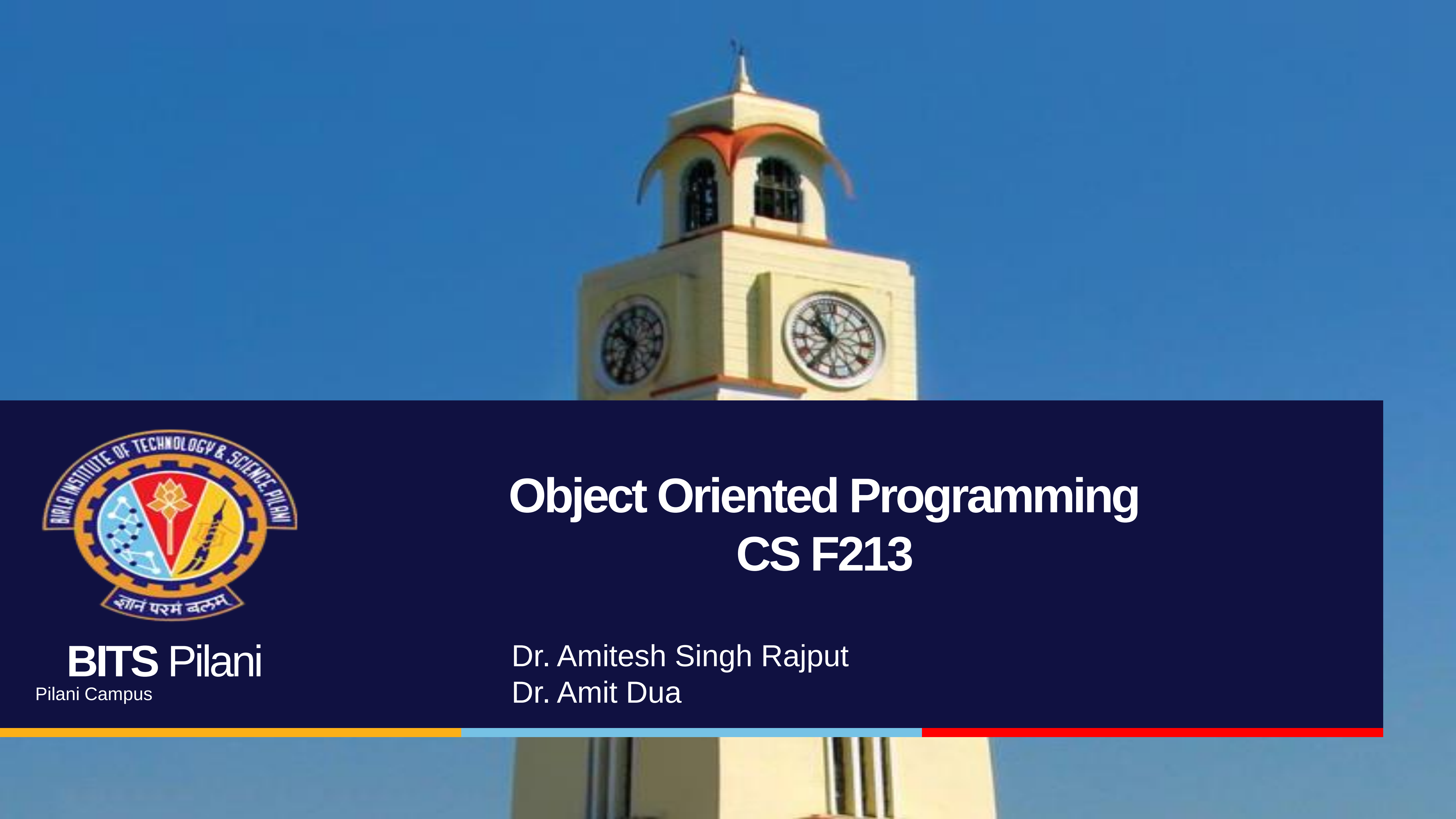


Object Oriented Programming CS F213

Dr. Amitesh Singh Rajput
Dr. Amit Dua

BITS Pilani
Pilani Campus

Variables and their types

BITS Pilani

Pilani Campus

Inheritance - Example Program

```
public class ParentOne {
  int a, b;
  void showOne() {
   System.out.println("a = "+a+" b = "+b);
  }
}

public class ChildOne extends ParentOne{
  int c;
  void showTwo() {
   System.out.println("c = "+c);
  }
  void showThree() {
   System.out.println("a = "+a+" b = "+b+" c = "+c);
  }
```

```
public static void main(String args[]){
   ChildOne o1 = new ChildOne();
   o1.a = 10;
   o1.b = 20;
   o1.c = 30;
   o1.showOne();
   o1.showTwo();
   o1.showThree();
   ParentOne p1 = new ParentOne();
   p1.showOne();
  }
}
```

Output

```
a = 10 b = 20
c = 30
a = 10 b = 20 c = 30
a = 0 b = 0
```

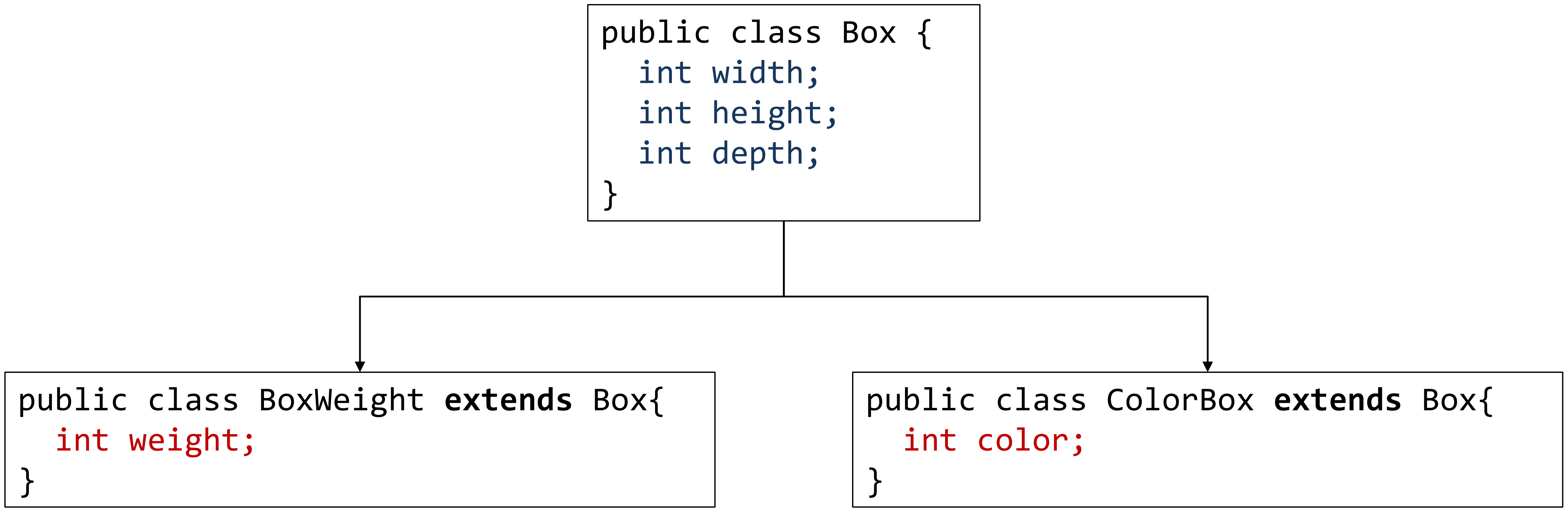
Inheritance - Example Program

```
public class ParentOne {
  private int a, b;
  void showOne() {
   System.out.println("a = "+a+" b = "+b);
  }
}

public class ChildOne extends ParentOne{
  int c;
  void showTwo() {
   System.out.println("c = "+c);
  }
  void showThree() {
   System.out.println("a = "+a+" b = "+b+" c = "+c);
  }
```

```
public static void main(String args[]){
   ChildOne o1 = new ChildOne();
   o1.a = 10;
   o1.b = 20;
   o1.c = 30;
   o1.showOne();
   o1.showTwo();
   o1.showThree();
   ParentOne p1 = new ParentOne();
   p1.showOne();
  }
}
```

```
The field ParentOne.a is not visible
The field ParentOne.b is not visible
```

Inheritance – Specific details with subclasses

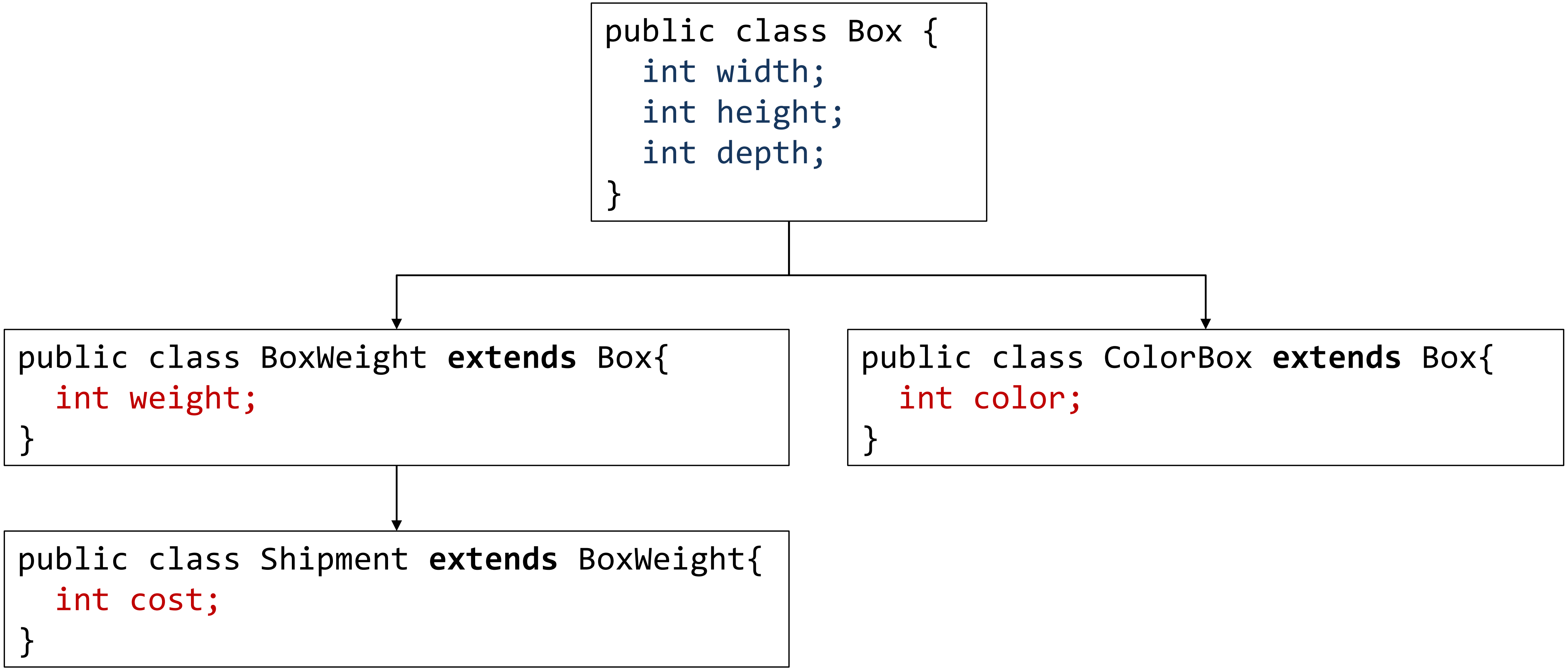


```
public class Box {
  int width;
  int height;
  int depth;
}
```

```
public class BoxWeight extends Box{
  int weight;
}
```

```
public class ColorBox extends Box{
  int color;
}
```

Multilevel Inheritance
Specific details with subclasses



```java
public class Box {
  int width;
  int height;
  int depth;
}
```

```java
public class BoxWeight extends Box{
  int weight;
}
```

```java
public class ColorBox extends Box{
  int color;
}
```

```java
public class Shipment extends BoxWeight{
  int cost;
}
```

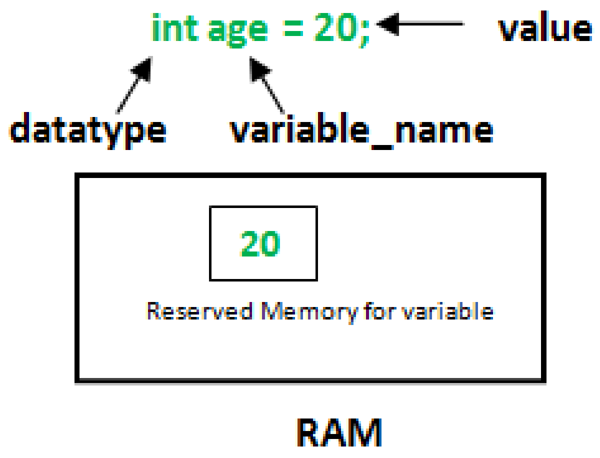
Naming Convention

- Used to name identifiers such as class, package, variable, constant, method etc.

Name	Convention
class name	should start with uppercase letter and be a noun e.g. String, Color, Button, System, Thread etc.
interface name	should start with uppercase letter and be an adjective e.g. Runnable, Remote, ActionListener etc.
method name	should start with lowercase letter and be a verb e.g. actionPerformed(), main(), print(), println() etc.
variable name	should start with lowercase letter e.g. firstName, orderNumber etc.
package name	should be in lowercase letter e.g. java, lang, sql, util etc.
constants name	should be in uppercase letter. e.g. RED, YELLOW, MAX_PRIORITY etc.

Variable

- Name given to a memory location.
- Basic unit of storage in a program.
  - The value can be changed during program execution.
  - All the operations done on the variable effects that memory locatio
  - Variables must be declared before they can be used.



- Types of variables
  - Static or class variables
  - Instance variables
  - Local variables

Static Variables

- A **static or class variable** is any field declared with the static modifier.

- Tells the compiler that there is exactly one copy of this variable in existence, regardless of how many times the class has been instantiated.

↓

- There will be only one copy of each static variable per class, regardless of how many objects are created.

Static Variables
Example using class access

```
public class StaticVarEx {
  static String myClsVar="class or static variable";

  public static void main(String args[]){
    System.out.println(StaticVarEx.myClsVar);
    System.out.println(StaticVarEx.myClsVar);

    StaticVarEx.myClsVar = "Changed Text";

    System.out.println(StaticVarEx.myClsVar);
    System.out.println(StaticVarEx.myClsVar);
  }
}
```

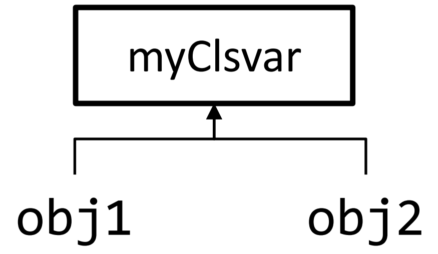
Output:

```
class or static variable
class or static variable
Changed Text
Changed Text
```

Static Variables
Example using object access

```
public class StaticVarEx {
 static String myClsVar="class or static variable";

 public static void main(String args[]){

   StaticVarEx obj1 = new StaticVarEx();
   StaticVarEx obj2 = new StaticVarEx();

   System.out.println(obj1.myClsVar);
   System.out.println(obj2.myClsVar);

   obj2.myClsVar = "Changed Text";


   System.out.println(obj1.myClsVar);
   System.out.println(obj2.myClsVar);
 }
}
```

Shared between obj1 and obj2



Output:

```
class or static variable
class or static variable
Changed Text
Changed Text
```

Static variable across classes-Example

```
class Zero{
static String classvar = "Static Variable of another class";
}

class First{
    public static void main(String args[]){
     System.out.println(Zero.classvar);
    }
}
```

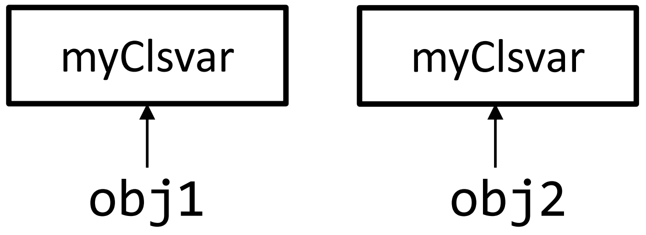
Instance (Non-static) Variables

- Objects store their individual states in "non-static fields‟.
- Declared without the static keyword.
- Known as instance variables because their values are unique to each instance of a class.

Instance Variables - Example

```
public class InstanceVarExample {
   String myInstanceVar=“instance variable";

   public static void main(String args[]){

      InstanceVarExample obj1 = new InstanceVarExample();
      InstanceVarExample obj2 = new InstanceVarExample();

      System.out.println(obj1.myInstanceVar);
      System.out.println(obj2.myInstanceVar);


      obj2.myInstanceVar = "Changed Text";

      System.out.println(obj1.myInstanceVar);
      System.out.println(obj2.myInstanceVar);
   }
}
```

Individual variable of each object



Output:

```
instance variable
instance variable
instance variable
Changed Text
```

Local Variable

- Defined within a block or method or constructor.

- These variable are created when the block is entered or the function is called and destroyed after exiting from the block or when the call returns from the function.

- The scope of these variables exists only within the block in which the variable is declared. i.e. we can access these variable only within that block.

Local Variable - Example

```
public class VariableExample {

 public String myVar="instance variable";     // instance variable

 public void myMethod(){
  String myVar = "Inside Method";              // local variable
  System.out.println(myVar);
 }

 public static void main(String args[]){

  VariableExample obj = new VariableExample();
  obj.myMethod();
  System.out.println(obj.myVar);
 }
}
```

Output:

```
Inside Method
Instance variable
```

innovate achieve lead

BITS Pilani
Pilani Campus

Thank You!