


The diagram shows six colored hexagons arranged in a circle around the text "OOPS". The hexagons are labeled: Encapsulation (blue), Abstraction (green), Polymorphism (red), Inheritance (magenta), Class (dark blue), and Object (yellow). To the right of the hexagons is the Java logo, which consists of a blue coffee cup with a red flame and the word "Java" in red.

ABESIT
COLLEGE OF ENGINEERING
CRC-Training

Instance, Static Variable and Method

OOPS USING JAVA 1



Instance Variable

- ❖ Instance variable are declared in a class, but outside a method, constructor or any block
- ❖ It belongs to the instance of a class or you can say it belong to the object.
- ❖ Instance variable have separate value of each and every instance of the class.

OOPS USING JAVA 2

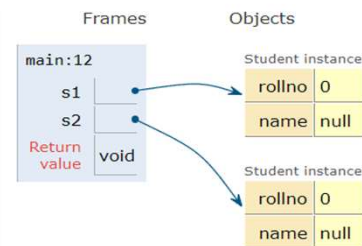
Instance Variable

Java 8
[known limitations](#)

```

1  class Student{
2      int rollno;
3      String name;
4  }
5
6
7
8  public class demo {
9      public static void main(String[] args) {
10         Student s1 = new Student();
11         Student s2 = new Student();
12     }
13 }
  
```

Instance Variable



OOPS USING JAVA

3

Class Variable / Static Variable

- ❖ Static variables are declared in a class, but outside a method, constructor or any block.
- ❖ Static Variable belongs to class not an individual object or instance.
- ❖ A static variable is created by adding the “static” keyword before the variable.
- ❖ The static variables can be accessed in all types of methods: static or non-static.

OOPS USING JAVA

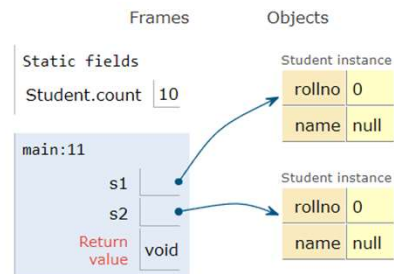
4

Class Variable / Static Variable

Java 8
[known limitations](#)

```

1 class Student{
2     static int count=10;
3     int rollno;
4     String name;
5 }
6
7 public class demo {
8     public static void main(String[] args) {
9         Student s1 = new Student();
10        Student s2 = new Student();
11    }
12 }
```



OOPS USING JAVA

5

How to access Static Variable

❖ We can use class name or object for accessing static variables.

❖ e.g.

Student.count;

or

s1.count;

OOPS USING JAVA

6

Local Variable

- ❖ A local variable is a variable that's declared within the body of a method, constructor or any block.
- ❖ they are not visible to other method.

Method

- ❖ A java method is a collection of Statements that are grouped together to perform an operation.
- ❖ A class can contain any number of methods.
- ❖ methods can be with parameter and without parameter.

Method

- ❖ Instance Method
- ❖ Static Method / Class Method

Instance Method

- ❖ Instance methods can access instance variables and instance methods directly.
- ❖ Instance methods can access a static variables and static methods directly.
- ❖ we can't define static variables inside the instance method.

Instance Method without Parameter

Syntax –

```
Modifier return_type method_name (){  
  
    // Method Body  
}
```

Modifier – It defines the access type of the method and it is optional.

Instance Method with Parameter

Syntax –

```
Modifier return_type method_name (parameter1, parameter2){  
  
    // Method Body  
}
```

```
Modifier return_type method_name (type ... par){  
  
    // Method Body  
}
```

Calling Instance Method

❖ Calling Instance Method through object - reference_variable.method_name()

```
class Student{
    int roll;
    String Name;
    void display1(){
        System.out.println("I am in display-1");
    }
}
```

```
public class demo {
    public static void main(String[] args) {
        Student s1 = new Student();
        s1.display1();
    }
}
```

OOPS USING JAVA

13

Calling Instance Method

❖ Calling Instance Method in Instance Method

- ❖ Instance methods can call an instance method directly.
- ❖ There is no need to use objects.

```
class Student{
    int roll;
    String Name;
    void display1(){
        System.out.println("I am in display-1");
    }
    void display2(){
        display1();
    }
}
```

```
public class demo {
    public static void main(String[] args) {
        Student s1 = new Student();
        s1.display2();
    }
}
```

OOPS USING JAVA

14

Calling Instance Method

- ❖ **Calling Instance Method in Static Method -**
 - ❖ We can not call an instance method in a static method directly.
- ❖ **Example will be discussed in later slide**

Static Method / Class Method

- ❖ A static method belongs to the class rather than the object of a class.
- ❖ A static method can be invoked without the object of the class.
- ❖ The static method can access static data members and can change their value of it.
- ❖ The static method can not use non-static data members.

Syntax to declare the static method

```
Access_modifier static void methodName()
{

}
```

Syntax to call static method

```
className.methodName();
```

OOPS USING JAVA

17

Example - static method

```
1 class Student{
2     static void display(){
3         System.out.println("I am in static");
4     }
5 }
6
7 public class YourClassNameHere {
8     public static void main(String[] args) {
9         Student s1 = new Student();
10        s1.display();
11        Student.display();
12    }
13 }
```

Print output (drag lower right corner to resize)

```
I am in static
I am in static
```

OOPS USING JAVA

18

Static Block

- ❖ The static block is a block of statements inside a Java class that will be executed when a class is first loaded into the JVM.
- ❖ A static block helps to initialize the static data members, just like constructors help to initialize instance members.

```
class Test{  
    static {  
        //Code goes here  
    }  
}
```