STATIC AND REFERENCES

Presented by:

Dr. Shivangi K. Surati

Assistant Professor,

Department of Computer Science and Engineering,

School of Technology,

Pandit Deendayal Energy University

Outline

- Static Keyword and features
- Static Variable Example
- Static Method and Example
- Restrictions for the static method
- Static-Points to remember
- Call by Value, Call by Reference
- ‘this’ reference in Java
- Explore ‘this’ reference

Static Keyword

- used for memory management mainly
- used to refer to the common property of all objects (not unique for each object)
- single copy storage for variables or methods
- The members that are declared with the static keyword inside a class are called static members in java.
- can be accessed/called even if no instance (object) of the class exists
  - not tied to a particular instance
  - shared across all instances of the class
  - EX: main method is static, so can be called by JVM without creating an object

Features of static keyword

- Can be applied with variables, methods, inner (nested) classes, and blocks
- A class cannot be static, but an inner class can be static
- Property (attribute or method) of a class, not of an instance (object)
- Memory is allocated only once when class is loaded into memory
- the static variable is created and initialized into the common memory location only once

Static Variable Example

```
class Counter{
        int count=0;          // the instance variable
        Counter(){
                count++;   //incrementing value
                System.out.println(count);
        }
  public static void main(String args[]){
        //Creating objects
        Counter c1=new Counter();
        Counter c2=new Counter();
        Counter c3=new Counter();
        }
}
```

Output:

```
1
1
1
```

Example…

```
class Counter{
        static int count=0; // the static variable
        Counter(){
                count++;    //incrementing value
                System.out.println(count);
        }
  public static void main(String args[]){
        //Creating objects
        Counter c1=new Counter();
        Counter c2=new Counter();
        Counter c3=new Counter();
        }
}
```

Output:

```
1
2
3
```

Example…

Static_prog_ex.doc

How to change value of static variable?

Static Method

- If you apply static keyword with any method, it is known as static method.
  - belongs to the class rather than the object of a class
  - can be invoked without the need for creating an instance of a class
  - can access static data member and can change the value of it

Static method example

Static_prog_ex.doc

Restrictions for the static method

- The static method can not use non static data member or call non-static method directly.
- this and super cannot be used in static context.
- EX:

```
class A{
        int a=40;       //non static

 public static void main(String args[]){
       System.out.println(a);
}
}
```

//error, can’t access without object

Static-Points to remember

- Static member cannot call an instance member.
- Static method can call a static method.
- Instance method can call a static method.
- Instance method can call an instance method.
- Static can be called with object name, but instance can’t be called using class name.

- Math class- all static methods
- String class- all instance methods

Call by Value, Call by Reference

Reference_Obj_prog.doc

‘this’ reference in Java

1. Using ‘this’ keyword to refer current class instance variables

```
class Test
{
    int a;
    int b;
        // Parameterized constructor
    Test(int a, int b)
    {
        this.a = a;
        this.b = b;
    }
 }
```

2. Using this() to invoke current class constructor

```
class Test
{
    int a;
    int b;

    //Default constructor
    Test()
    {
        this(10, 20);
        System.out.println("Inside  default constructor \n");
    }
```

```
 //Parameterized constructor
    Test(int a, int b)
    {
        this.a = a;
        this.b = b;
        System.out.println("Inside parameterized constructor");
    }
public static void main(String[] args)
    {
        Test object = new Test();
    }
}
```

Explore ‘this’ reference

3. Using ‘this’ keyword to return the current class instance
4. Using ‘this’ keyword as method parameter
5. Using ‘this’ keyword to invoke current class method
6. Using ‘this’ keyword as an argument in the constructor call