

***** What is Variable Shadow in Java ?**

Variable shadowing is the process of having exactly same name than method level variable or field class level variable inside the method OR constructor OR block body. This concept is known as Variable Shadow.

Case 1:

```
package com.ait.variable_shadow;
class Test {
    static int a = 100; //Static Field
    int b = 200; //Non static field
    public void printValue(int a) //a is parameter variable
    {
        int b = 400; //local variable
        System.out.println(a); //100
        System.out.println(b); //400
    }
}
```

Case 2:

In the above program, it is clear that inside the method local variable is having more priority. So we want to represent class level fields inside the method OR block OR constructor OR block body then we should use the static keyword.

If no static keyword is present then static use

For static field - We can use the static keyword

For non static field - We should use this keyword

```
package com.ait.variable_shadow;
class Test {
    static int a = 100; //Static Field
    int b = 200; //Non static field
    public void printValue(int a) //a is Parameter Variable
    {
        int a = 400; //local variable
        System.out.println(a); //100
        System.out.println(b); //200
    }
}
```

public class variable shadow

```
public static void main(String[] args)
{
    BBB b1 = new BBB();
    b1.printValue(400);
}
```

}

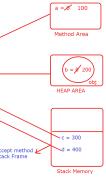
***** What is Method Local Search algorithm in Java :**

Whenever any method, block or constructor is executed first of all compiler will search the variable in the current method or constructor if found then it will use it.

* If Variable declaration is not available at method level then Compiler will search in the class (class Level).

//Program - Output

```
package com.ait.variable_shadow;
class Test {
    static int a = 100; //Static Field
    int b = 200; //Non static field
    public void printValue(int a) //a is parameter variable
    {
        int a = 400; //local variable
        System.out.println("a = "+a);
        System.out.println("b = "+b);
        System.out.println("a = "+a);
        System.out.println("b = "+b);
    }
}
```



This keyword in Java : (In the statement (Customer) = Object compiler, the LLC class should be in the tree)

* This is a very special keyword in java which is used to refer the current Object as shown in the diagram.

public class Thiskeyword

```
public static void main(String[] args)
{
    Customer scott = new Customer();
    scott.setId(1);
    scott.setName("Scott");
    scott.getCustomerData();
}
```

* By using this keyword, We can refer the non-static field & non-static method anywhere in the BLC class.

* This keyword, we cannot use from static context (From Static Method OR Static Block OR Static Nested Class).

* IN JAVA, WHENEVER WE CREATE AN OBJECT THEN AUTOMATICALLY JAVA COMPILER WILL ADD THAT THIS KEYWORD AS A PARAMETER TO ALL THE NON STATIC METHODS OF THAT CLASS.

Customer.java

```
public class Customer
{
    private int id;
    public void setId(int id)
    {
        this.id = id;
    }
    public int getId()
    {
        return id;
    }
    public void getName()
    {
        System.out.println("Name");
    }
}
package com.ait.this_keyword;
class Test
{
    private int id;
    private String name;
    public void setCustomerData(int id, String name)
    {
        this.id = id;
        this.name = name;
    }
    public void getCustomerData()
    {
        System.out.println("Customer id is "+this.id);
        System.out.println("Customer Name is "+this.name);
    }
}
public class Thiskeyword
{
    public static void main(String[] args)
    {
        Customer scott = new Customer();
        scott.setId(1);
        scott.setName("Scott");
        scott.getCustomerData();
    }
}
```