What is variable shadow ?

In Java, class level fields (static OR non static) name can be same with method level variables (Parameter OR Local) name.

If class level fields and method level variables are having exactly same name then method level variable will hide class level fields INSIDE THE method OR constructor OR block body, This concept is known as Variable Shadow.

//PROGRAM ON VARIABLE SHADOW

```java
package com.ravi.variable_shadow;

public class Student
{
   private int roll = 101;
   public static String collegeName = "NIT";

   public void accept(int roll) //roll = 201
   {
      String collegeName = "JNTU";

      IO.println(roll);
      IO.println(collegeName);    //Output is 201 & JNTU
   }
}
```

```java
package com.ravi.variable_shadow;

public class VariableShadow
{
    public static void main(String[] args)
    {
       Student s1 = new Student();
       s1.accept(201);
    }

}
```

How to represent class level fields from method OR constructor OR block body, In case of Variable Shadow concept.

Inside the method OR block OR constructor body, If we want to represent class level fields then
  a) For non static fields we should use **this** keyword
  b) For static fields we should use **class name** as shown in the program below.

```java
package com.ravi.variable_shadow;

public class Student
{
   private int roll = 101;
   public static String collegeName = "NIT";

   public void accept(int roll) //roll = 201
   {
      String collegeName = "JNTU";

      IO.println(this.roll);
      IO.println(Student.collegeName);    //Output 101 & NIT
   }

}
```

```java
package com.ravi.variable_shadow;

public class VariableShadow
{
    public static void main(String[] args)
    {
       Student s1 = new Student();
       s1.accept(201);
    }

}
```

What is local search algorithm :

- Whenever any method, block or constructor is executed then first of all compiler will search the variable declaration in the method, block or constructor (At Method Level).
- If Variable declaration is not available at method level then Compiler will search in the class (class Level)
- Only class level fields are having default value even the fields are final.

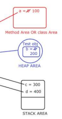
```java
class Test
{
    public static int a = 100;   //SF
    private int b = 200;   //NSF

    public void accept(int c) //c is parameter Variable
    {
        int d = 400;  //Local Varaible

        IO.println("Static Field :"+ Test.a);
        IO.println("Non Static Field :"+ this.b);
        IO.println("Parameter Variable:"+c);
        IO.println("Local Variable :"+d);
    }

}

public class LocalSearch
{
    public static void main(String[] args)
    {
        Test t1 = new Test();
        t1.accept(300);
    }
}
```



---

this keyword :

Diagram for this keyword :

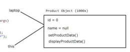
```java
package com.ravi.this_keyword;

public class ProductDemo {

    public static void main(String[] args)
    {

        Product laptop = new Product();
        laptop.setProductData(111, "HP");
        laptop.displayProductData();
    }

}
```



* this keyword is used to refer the current object that means Our current object is also refered by one keyword known as this keyword.

* this keyword is used to represent our object members anywhere in the BLC class.

* Whenever instance variable name and parameter/local variable name both are same then at the time of instance variable initialization, our runtime environment will provide more priority to parameter variable/local variable, parameter variables are hiding the instance variables (Due to variable shadow)

* To avoid the above said problem, Java software people introduced "this" keyword.

* this keyword always refers to the current object and instance variables are the part of the object so by using this keyword we can represent instance variable.

* this keyword we cannot use from any **static context** (Static Method OR Static Block OR Static nested inner class)

* **Whenever we create an object in java then automatically java comiler will add final reference variable to each and every method and constaructor as a first parameter.**

Example :

```java
public class Student
{
   private int roll;
   public void accept(int roll)
   {
      this.roll = roll;
   }
}
```

javac

```java
public class Student
{
   private int roll;

   public void accept(final Student this, int roll)
   {
      this.roll = roll;
   }
}
```

added by javac

//program :

```java
package com.ravi.this_keyword;

public class Product
{
  private int id;
  private String name;

  public void setProductData(int id, String name)
  {
       this.id = id;
       this.name = name;
  }


  public void displayProductData()
  {
      IO.println("Product Id is :"+this.id);
      IO.println("Product Name is :"+this.name);
  }

}
```

```java
package com.ravi.this_keyword;

public class ProductDemo
{
    public static void main(String[] args)
    {
        Product laptop = new Product();
        laptop.setProductData(111, "HP");
        laptop.displayProductData();

    }

}
```