Java this keyword

In Java, ‘this’ is a keyword which is used to refer current object of a class. we can use it to refer any member of the class. It means we can access any instance variable and method by using this keyword.

The main purpose of using this keyword is to solve the confusion when we have same variable name for instance and local variables.

We can use **‘this’** keyword for the following purpose.

1. this keyword is used to refer to current object.

2. this is always a reference to the object on which method was invoked.

3. this can be used to invoke current class constructor.

4. this can be passed as an argument to another method.

Let’s first understand the most general use of this keyword. As we said, it can be used to differentiate local and instance variables in the class.

In this example, we have three instance variables and a constructor that have three parameters with same name as instance variables. Now, we will use this to assign values of parameters to instance variables.

```
class Demo
{
        Double width, height, depth;
        Demo (double w, double h, double d)
        {
                Double width=0d, height=0d, depth=0d;
                this.width = w;
                this.height= h;
                this.depth = d;
                System.out.println("Constwidth = "+width);
                System.out.println("Constheight = "+height);
```

```
        System.out.println("Constdepth = "+depth);
    }
    public static void main(String[] args) {
        Demo d = new Demo(10,20,30);
        System.out.println("width = "+d.width);
        System.out.println("height = "+d.height);
        System.out.println("depth = "+d.depth);
    }
}
```

Here, 'this' is used to initialize member of current object. Such as, this.width refers to the variable of the current object and width only refers to the parameter received in the constructor i.e the argument passed while calling the constructor.

Calling Constructor using this keyword

We can call a constructor from inside another constructor by using this keyword

Example:

In this example, we are calling a parameterized constructor from the non-parameterized constructor using this keyword along with argument.

```
class Demo
{
    Demo ()
    {
        // Calling constructor
        this("Class Constructor");
    }
    Demo(String str)
    {
```

```
        System.out.println(str);
    }
    public static void main(String[] args)
    {
        Demo d = new Demo();
    }
}
```

Accessing Method using this keyword

This is another use of this keyword that allows to access method. We can access method using object reference too but if we want to use implicit object provided by Java then use this keyword.

Example:

In this example, we are accessing getName() method using this and it works fine as works with object reference. See the below example,

```
class Demo
{
    void getName()
    {
        System.out.println("Class Method");
    }
    void display()
    {
        this.getName();
    }
    public static void main(String[] args) {
        Demo d = new Demo();
        d.display();
    }
```

```
}
```

Return Current Object from a Method

In such scenario, where we want to return current object from a method then we can use this to solve this problem.

Example:

In this example, we created a method display that returns the object of Demo class. To return the object, we used this keyword and stored the returned object into Demo type reference variable. We used that returned object to call getName() method and it works fine.

```
class Demo{
	void getName()
	{
		System.out.println("Object Returned");
	}
	Demo display()
	{
		// return current object
		return this;
	}
	public static void main(String[] args)
	{
		Demo d = new Demo();
		Demo d1 = d.display();
		d1.getName();
	}
}
```