**Super keyword in java with example**

[**OOPS CONCEPT**](http://beginnersbook.com/category/oops-concept/)

In the last post we discussed about [**final keyword**](http://beginnersbook.com/2014/07/final-keyword-java-final-variable-method-class/). In this tutorial we are gonna learn about super keyword which refers to the immediate parent of a class.

**super Usage:**

1) super.<variable\_name> refers to the variable of variable of parent class.  
2) super() invokes the constructor of immediate parent class.  
3) super.<method\_name> refers to the method of parent class.

Lets discuss these three things in detail with examples.

**1) super.<variable\_name> to invoke Parent class variable**

Lets take this example to understand the need of super keyword:

//Parent class or Superclass

class Parentclass

{

int num=100;

}

//Child class or subclass

class Subclass extends Parentclass

{

/\* I am declaring the same variable

\* num in child class too.

\*/

int num=110;

void printNumber(){

System.out.println(num);

}

public static void main(String args[]){

Subclass obj= new Subclass();

obj.printNumber();

}

}

Output:  
110

In the above program we have the same variable “num” declared in both parent class and sub class. There is no way we can access the num variable of parent class without using super keyword.

**Accessing the num variable of parent class:**

//Parent class or Superclass

class Parentclass

{

int num=100;

}

//Child class or subclass

class Subclass extends Parentclass

{

int num=110;

void printNumber(){

//Super.variable\_name

System.out.println(super.num);

}

public static void main(String args[]){

Subclass obj= new Subclass();

obj.printNumber();

}

}

Output:  
100

As you can see by using super.num we accessed the num variable of parent class.

**2) super() to invoke constructor of parent class**

First we will see compiler default behavior. When we create the object of sub class, first the [**constructor**](http://beginnersbook.com/2013/03/constructors-in-java/) of parent class gets invoked and then the constructor of child class. It happens because compiler itself adds super()[it invokes parent class constructor] to the constructor of child class.

class Parentclass

{

Parentclass(){

System.out.println("Constructor of Superclass");

}

}

class Subclass extends Parentclass

{

Subclass(){

/\* Compile adds super() here at the first line

\* of this constructor implicitly

\*/

System.out.println("Constructor of Subclass");

}

Subclass(int num){

/\* Compile adds super() here at the first line

\* of this constructor implicitly

\*/

System.out.println("Constructor with arg");

}

void display(){

System.out.println("Hello");

}

public static void main(String args[]){

// Creating object using default constructor

Subclass obj= new Subclass();

//Calling sub class method

obj.display();

//Creating object 2 using arg constructor

Subclass obj2= new Subclass(10);

obj2.display();

}

}

**Output:**

Constructor of Superclass

Constructor of Subclass

Hello

Constructor of Superclass

Constructor with arg

Hello

**We can call super() explicitly too**

class Parentclass

{

Parentclass(){

System.out.println("Constructor of Superclass");

}

}

class Subclass extends Parentclass

{

Subclass(){

/\* super() must be added to the first

\* line of constructor otherwise it would

\* throw compilation error:

\* " Constructor call must be the first statement

\* in a constructor".

\*/

super();

System.out.println("Constructor of Subclass");

}

void display(){

System.out.println("Hello");

}

public static void main(String args[]){

Subclass obj= new Subclass();

obj.display();

}

}

**Output:**

Constructor of Superclass

Constructor of Subclass

Hello

**Note:**  
1) super() must be the first statement in constructor otherwise we will get the compilation error message: “Constructor call must be the first statement in a constructor”  
2) We can also invoke [**parameterized constructor**](http://beginnersbook.com/2014/01/parameterized-constructor-in-java-example/) of parent class by providing arguments while calling super. For e.g. super(10) would invoke the parametrized constructor [having one integer argument] of parent class. Similarly super(“hi”) would invoke constructor having String argument.

**3) super.<method\_name> to invoke parent class method**

super.method\_name calls Overridden method. For e.g.  
Please refer the comments in the below program to understand better:

class Parentclass

{

//Overridden method

void display(){

System.out.println("Parent class method");

}

}

class Subclass extends Parentclass

{

//Overriding method

void display(){

System.out.println("Child class method");

}

void printMsg(){

//This would call Overriding method

display();

//This would call Overridden method

super.display();

}

public static void main(String args[]){

Subclass obj= new Subclass();

obj.printMsg();

}

}

**Output:**

Child class method

Parent class method

**If there is no**[**method overriding**](http://beginnersbook.com/2014/01/method-overriding-in-java-with-example/)**then we do not need to use super to call the parent class method.**

Consider this example. Here we are able to call parent class method without using super keyword.

class Parentclass

{

void display(){

System.out.println("Parent class method");

}

}

class Subclass extends Parentclass

{

void printMsg(){

/\* This would call parent class method

\* no need to use super keyword

\*/

display();

}

public static void main(String args[]){

Subclass obj= new Subclass();

obj.printMsg();

}

}

**Output:**

Parent class method