# **Super Keyword**

The super keyword refers to the objects of immediate parent class.

Super keyword is used for:

1) To access the data members of parent class when both parent and child class have member with same name  
2) To explicitly call the no-arg and parameterized constructor of parent class  
3) To access the method of parent class when child class has overridden that method.

Super Keyword to Access the Parent Class Variables:

**Accessing Parent class variables using Super keyword:**

ParentClass.java:

**package** FPPackage;

**public** **class** ParentClass {

**int** age = 10;

String name = "Subbu";

}

ChildClass.java

**package** FPPackage;

**public** **class** ChildClass **extends** ParentClass{

**int** age = 15;

String name = "Venkat";

**public** **void** displayParentDetails() {

System.***out***.println(**super**.age);

System.***out***.println(**super**.name);

}

**public** **static** **void** main(String[] args) {

ChildClass cl = **new** ChildClass();

cl.displayParentDetails();

System.***out***.println(cl.age);

System.***out***.println(cl.name);

}

}

**Accessing the parent class no-arg constructor:**

When you create an object for child class it will by default accesses the parent’s class no-arg constructor.

Example:

ParentClass:

**package** FPPackage;

**public** **class** ParentClass {

ParentClass(){

System.***out***.println("Parent class no-arg constructor");

}

}

ChildClass:

**package** FPPackage;

**public** **class** ChildClass **extends** ParentClass{

ChildClass(){

System.***out***.println("I am child class constructor");

}

**public** **static** **void** main(String[] args) {

ChildClass cl = **new** ChildClass();

}

}

**Accessing the Parent Class Argument Constructor:**

Parent Class:

**package** FPPackage;

**public** **class** ParentClass {

ParentClass(){

System.***out***.println("Parent class no-arg constructor");

}

ParentClass(**int** value){

System.***out***.println("Parent class constructor with argument "+value);

}

}

Child Class:

**package** FPPackage;

**public** **class** ChildClass **extends** ParentClass{

ChildClass(){

**super**(100);

System.***out***.println("I am child class constructor");

}

**public** **static** **void** main(String[] args) {

ChildClass cl = **new** ChildClass();

}

}

**Super Keyword When Method Overriding:**

Example:

ParentClass.java:

**package** FPPackage;

**public** **class** ParentClass {

**public** **void** methodoverriding() {

System.***out***.println("This method will be overriden");

}

}

ChildClass.java:

**package** FPPackage;

**public** **class** ChildClass **extends** ParentClass{

**public** **void** methodoverriding() {

System.***out***.println("This method is overridden");

}

**public** **void** parentmethod() {

**super**.methodoverriding();

}

**public** **static** **void** main(String[] args) {

ChildClass cl = **new** ChildClass();

cl.methodoverriding();

cl.parentmethod();

}

}

**How to access the parent method that is not overridden?**

When a parent method is not overridden you can just access it with the child class object.

Example:

ParentClass.java:

**package** FPPackage;

**public** **class** ParentClass {

**public** **void** methodoverriding() {

System.***out***.println("This method will be overriden");

}

**public** **void** methodnotoverriding() {

System.***out***.println("I am not overridden");

}

}

ChildClass.java:

**package** FPPackage;

**public** **class** ChildClass **extends** ParentClass{

**public** **void** methodoverriding() {

System.***out***.println("This method is overridden");

}

**public** **void** parentmethod() {

**super**.methodoverriding();

}

**public** **static** **void** main(String[] args) {

ChildClass cl = **new** ChildClass();

cl.methodoverriding();

cl.parentmethod();

cl.methodnotoverriding();

}

}