Super keyword in java

The **super** keyword in java is a reference variable that is used to refer immediate parent class object.

Whenever you create the instance of subclass, an instance of parent class is created implicitly i.e. referred by super reference variable.

Usage of java super Keyword

1. super is used to refer immediate parent class instance variable.
2. super() is used to invoke immediate parent class constructor.
3. super is used to invoke immediate parent class method.

1) super is used to refer immediate parent class instance variable.

Problem without super keyword

```
class Vehicle{
  int speed=50;
}
class Bike3 extends Vehicle{
  int speed=100;
  void display(){
   System.out.println(speed);//will print speed of Bike
  }
  public static void main(String args[]){
   Bike3 b=new Bike3();
   b.display();
}
}
```

In the above example Vehicle and Bike both class have a common property speed. Instance variable of current class is referred by instance by default, but I have to refer parent class instance variable that is why we use super keyword to distinguish between parent class instance variable and current class instance variable.

Solution by super keyword

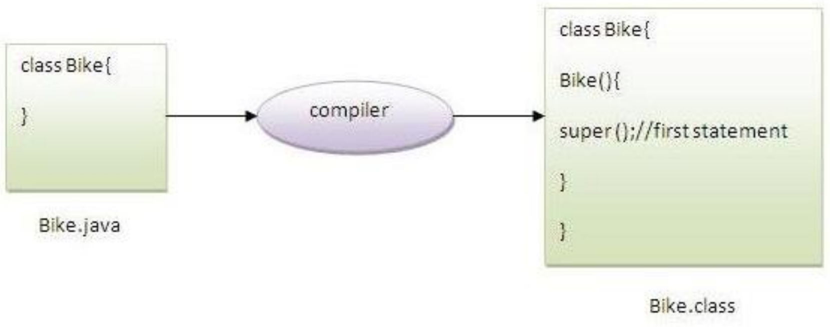
```
//example of super keyword

class Vehicle{
  int speed=50;
}

class Bike4 extends Vehicle{
  int speed=100;

  void display(){
   System.out.println(super.speed);//will print speed of Vehicle now
  }
  public static void main(String args[]){
   Bike4 b=new Bike4();
   b.display();

}
}
```

2) super is used to invoke parent class constructor.

The super keyword can also be used to invoke the parent class constructor as given below:

```
class Vehicle{
  Vehicle(){System.out.println("Vehicle is created");}
}

class Bike5 extends Vehicle{
  Bike5(){
   super();//will invoke parent class constructor
   System.out.println("Bike is created");
  }
  public static void main(String args[]){
   Bike5 b=new Bike5();

}
}
```

Note: super() is added in each class constructor automatically by compiler.



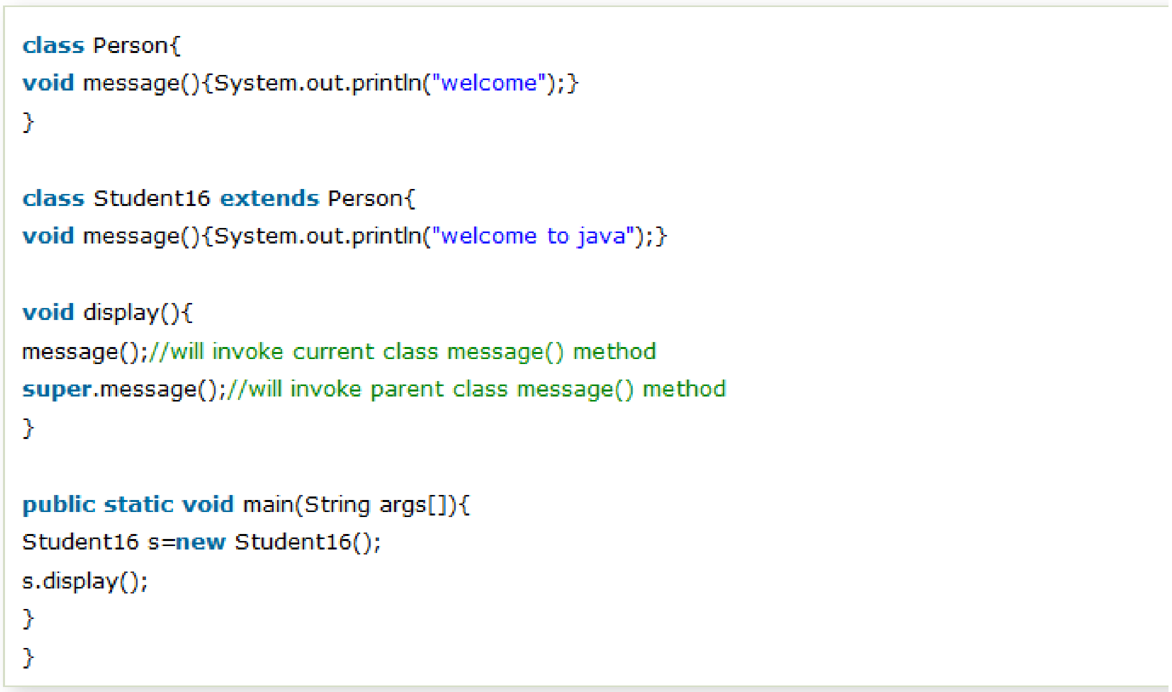
As we know well that default constructor is provided by compiler automatically but it also adds super() for the first statement. If you are creating your own constructor and you don't have either this() or super() as the first statement, compiler will provide super() as the first statement of the constructor.

Another example of super keyword where super() is provided by the compiler implicitly.

```
class Vehicle{
  Vehicle(){System.out.println("Vehicle is created");}
}

class Bike6 extends Vehicle{
  int speed;
  Bike6(int speed){
    this.speed=speed;
    System.out.println(speed);
  }
  public static void main(String args[]){
   Bike6 b=new Bike6(10);
 }
}
```

3) super can be used to invoke parent class method

The super keyword can also be used to invoke parent class method. It should be used in case subclass contains the same method as parent class as in the example given below:

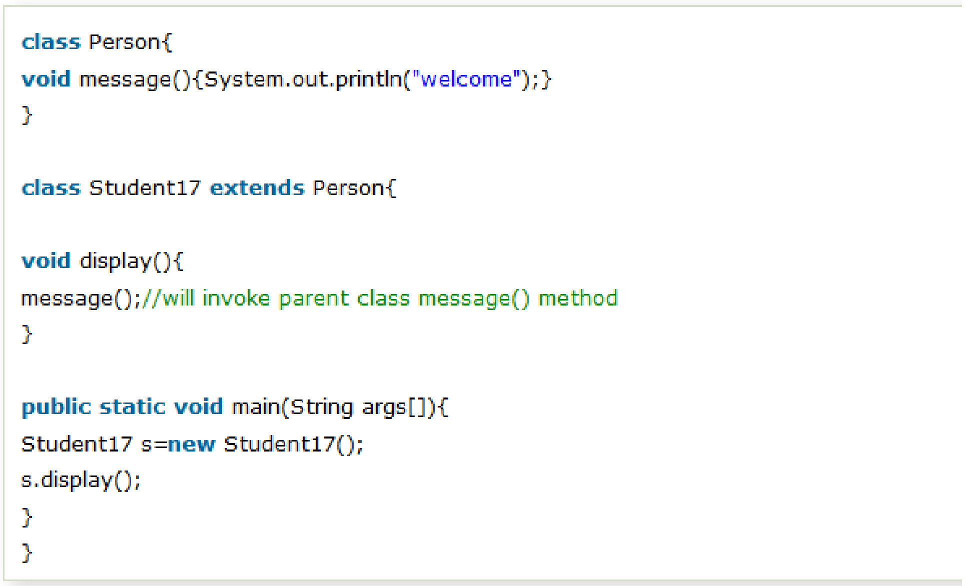


```
class Person{
void message(){System.out.println("welcome");}
}

class Student16 extends Person{
void message(){System.out.println("welcome to java");}

void display(){
message();//will invoke current class message() method
super.message();//will invoke parent class message() method
}

public static void main(String args[]){
Student16 s=new Student16();
s.display();
}
}
```

In the above example Student and Person both classes have message() method if we call message() method from Student class, it will call the message() method of Student class not of Person class because priority is given to local.

In case there is no method in subclass as parent, there is no need to use super. In the example given below message() method is invoked from Student class but Student class does not have message() method, so you can directly call message() method.

Program in case super is not required



```
class Person{
void message(){System.out.println("welcome");}
}

class Student17 extends Person{

void display(){
message();//will invoke parent class message() method
}

public static void main(String args[]){
Student17 s=new Student17();
s.display();
}
}
```