**Final Keyword In Java**

# Heading 1

The **final keyword** in java is used to restrict the user. The java final keyword can be used in many context. Final can be:

1. variable
2. method
3. class

The final keyword can be applied with the variables; a final variable that have no value it is called blank final variable or uninitialized final variable. It can be initialized in the constructor only. The blank final variable can be static also which will be initialized in the static block only.



### **Example of final variable**

There is a final variable speedlimit, we are going to change the value of this variable, but It can't be changed because final variable once assigned a value can never be changed.

1. **class** Bike9{
2. **final** **int** speedlimit=90;//final variable
3. **void** run(){
4. speedlimit=400;
5. }
6. **public** **static** **void** main(String args[]){
7. Bike9 obj=**new**  Bike9();
8. obj.run();
9. }
10. }//end of class

[**Test it Now**](http://www.javatpoint.com/opr/test.jsp?filename=Bike9)

Output:Compile Time Error

### **Example of final method**

1. **class** Bike{
2. **final** **void** run(){System.out.println("running");}
3. }
5. **class** Honda **extends** Bike{
6. **void** run(){System.out.println("running safely with 100kmph");}
8. **public** **static** **void** main(String args[]){
9. Honda honda= **new** Honda();
10. honda.run();
11. }
12. }

[**Test it Now**](http://www.javatpoint.com/opr/test.jsp?filename=Honda)

Output:Compile Time Error

### **Example of final class**

1. **final** **class** Bike{}
3. **class** Honda1 **extends** Bike{
4. **void** run(){System.out.println("running safely with 100kmph");}
6. **public** **static** **void** main(String args[]){
7. Honda1 honda= **new** Honda1();
8. honda.run();
9. }
10. }

[**Test it Now**](http://www.javatpoint.com/opr/test.jsp?filename=Honda1)

Output: Compile Time Error

### **Q) Is final method inherited?**

Ans) Yes, final method is inherited but you cannot override it. For Example:

1. **class** Bike{
2. **final** **void** run(){System.out.println("running...");}
3. }
4. **class** Honda2 **extends** Bike{
5. **public** **static** **void** main(String args[]){
6. **new** Honda2().run();
7. }
8. }

[**Test it Now**](http://www.javatpoint.com/opr/test.jsp?filename=Honda2)

Output: running...

### **Que) Can we initialize blank final variable?**

Yes, but only in constructor. For example:

1. **class** Bike10{
2. **final** **int** speedlimit;//blank final variable
4. Bike10(){
5. speedlimit=70;
6. System.out.println(speedlimit);
7. }
9. **public** **static** **void** main(String args[]){
10. **new** Bike10();
11. }
12. }

[**Test it Now**](http://www.javatpoint.com/opr/test.jsp?filename=Bike10)

Output: 70

### **static blank final variable**

A static final variable that is not initialized at the time of declaration is known as static blank final variable. It can be initialized only in static block.

### **Example of static blank final variable**

1. **class** A{
2. **static** **final** **int** data;//static blank final variable
3. **static**{ data=50;}
4. **public** **static** **void** main(String args[]){
5. System.out.println(A.data);
6. }
7. }

### **Q) What is final parameter?**

If you declare any parameter as final, you cannot change the value of it.

1. **class** Bike11{
2. **int** cube(**final** **int** n){
3. n=n+2;//can't be changed as n is final
4. n\*n\*n;
5. }
6. **public** **static** **void** main(String args[]){
7. Bike11 b=**new** Bike11();
8. b.cube(5);
9. }
10. }

[**Test it Now**](http://www.javatpoint.com/opr/test.jsp?filename=Bike11)

Output: Compile Time Error

### **Q) Can we declare a constructor final?**

No, because constructor is never inherited.

When you set a method as final it means: "I don't want any class override it." But according to the Java Language Specification:

[JLS 8.8](https://docs.oracle.com/javase/specs/jls/se11/html/jls-8.html#jls-8.8) - "Constructor declarations are not members. They are never inherited and therefore are not subject to hiding or overriding."

When you set a method as abstract it means: "This method doesn't have a body and it should be implemented in a child class." But the constructor is called implicitly when the new keyword is used so it can't lack a body.

When you set a method as static it means: "This method belongs to the class, not a particular object." But the constructor is implicitly called to initialize an object, so there is no purpose in having a static constructor.