Final keyword in Java

Java – Final

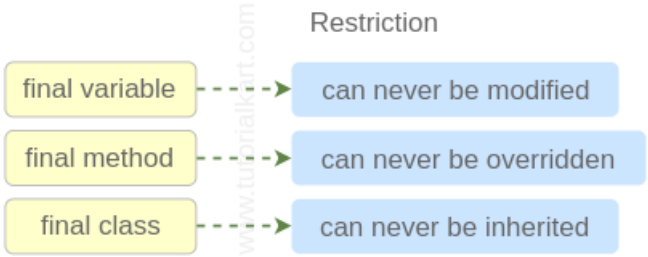
Final keyword in Java can be applied to a Variable, Method or Class. Java keeps restrictions on those, that are declared final.

In this tutorial, we shall learn about the final keyword, and its usage, with the help of illustrative example programs.

Final keyword in Java

Final keyword in Java can be applied in the following scenarios:

- **Final Variable** – Once a variable is declared as final, it can be initialized during declaration or in the constructor. And can never be changed during the course of the program. Hence static final variables are also called constants.
- **Final Method** – Once a method is declared as final, it can never be overridden by any sub class that is inheriting the method's class.
- **Final Class** – Once a class is declared as final, it can never be inherited.



Final Variable in Java

Final variable once initialized can never be modified.

Following Example Java program, Audi.java, demonstrates following two scenarios :

- BRAND is a final variable and initialized to "AUDI" during declaration itself and can never be changed. They remain constant for all the objects of class type Audi.
- EngineNumber is only declared as final but not initialized. These kind of variables could be initialized in Constructor. They remain constant only for the object of Audi, i.e., each Audi object can have different

EngineNumber.

Audi.java

```
public class Audi {

    public final String BRAND = "AUDI";
    public final String EngineNumber;

    public Audi(String EngineNumber){
        this.EngineNumber = EngineNumber;
    }

    public static void main(String[] args) {
        Audi audi = new Audi("ABCD1234CDEF4567");
        System.out.println("Engine Number : "+audi.EngineNumber);
        System.out.println("Car Brand : "+audi.BRAND);
    }
}
```

Output

```
Engine Number : ABCD1234CDEF4567
Car Brand : AUDI
```

Now we shall try to modify the final variable and understand what happens.

Audi.java

```
public class Audi {

    public final String EngineNumber;

    public Audi(String EngineNumber){
        this.EngineNumber = EngineNumber;
    }

    public static void main(String[] args) {
        Audi audi = new Audi("ABCD1234CDEF4567");
        audi.EngineNumber = "ERTY1234CDEF4568";
        System.out.println("Engine Number : "+audi.EngineNumber);
    }
}
```

Java Compiler throws an Error.

```
The final field Audi.EngineNumber cannot be assigned.
```

Final Method in Java

Final method cannot be overridden. This property could also be used to keep restrictions in Polymorphism –

Method Overriding.

In the following example, we have two classes: Car.java and Audi.java. Audi class inherits Car class, and `accelerate()` method of car is declared final.

Car.java

```
public class Car {

    public void brake(){
        System.out.println("break in Car");
    }

    public final void accelerate(){
        System.out.println("accelerate in Car");
    }
}
```

Audi.java

```
public class Audi extends Car {

    public static void main(String[] args) {
        Audi audi = new Audi();
        audi.accelerate();
        audi.brake();
    }
}
```

Output

```
accelerate in Car
break in Car
```

If we try to override `accelerate()` method in `Audi.java`, we will get Error.

Audi.java

```
public class Audi extends Car {

    public static void main(String[] args) {
        Audi audi = new Audi();
        audi.accelerate();
        audi.brake();
    }

    public void accelerate(){
        System.out.println("accelerate in Audi");
    }
}
```

Java Compilation Error occurs

```
Cannot override the final method from Car
```

Final Class in Java

Final Class can never be inherited by other classes.

AudiR8.java

```
public final class AudiR8 {

    public final String BRAND = "AUDI";
    public final String EngineNumber;

    public AudiR8(String EngineNumber){
        this.EngineNumber = EngineNumber;
    }
}
```

We shall now try to extend this AudiR8.java from another class and understand what happens

AnotherCar.java

```
public class AnotherCar extends AudiR8 { }
```

Java Compiler throws the following error.

```
Exception in thread "main" java.lang.Error: Unresolved compilation problem:

    at AnotherCar.main(AnotherCar.java:4)
```

Conclusion

In this Java Tutorial, we have learned about **final** keyword in Java that can be applied to a Variable, Method or Class and also the restrictions Java keeps on those, that are declared final.

In our next tutorial, we shall learn about another Object Oriented Concept – Abstraction in Java.

Java Tutorial

✦ Java Tutorial