FINAL METHODS AND CLASSES

The final keyword in java is used to restrict the user. The java final keyword can be applied to:

- variable
- method
- class

Java final variable	-	To prevent constant variables
Java final method	-	To prevent method overriding
Java final class	-	To prevent inheritance

Sample Code:

A final variable speed limit is defined within a class Vehicle. When we try to change the value of this variable, we get an error. This is due to the fact that the value of final variable cannot be changed, once a value is assigned to it.

```
public class Vehicle
{
 final int speedlimit=60;//final variable
 void run()
 {
  speedlimit=400;
 }
 public static void main(String args[])
 {
 Vehicle obj=new Vehicle();
 obj.run();
 }
  }
```

Output:

```
/Vehicle.java:6: error: cannot assign a value to final variable speedlimit
  speedlimit=400;
  ^
1 error
```

Blank final variable

A final variable that is not initialized at the time of declaration is known as blank final variable. We must initialize the blank final variable in constructor of the class otherwise it will throw a compilation error.

Sample Code:

```
public class Vehicle
{
 final int speedlimit; //blank final variable
 void run()
 {
 }
 public static void main(String args[])
 {
 Vehicle obj=new Vehicle();
 obj.run();
 }
  }
```

Output:

```
/Vehicle.java:3: error: variable speedlimit not initialized in the default constructor
 final int speedlimit; //blank final variable
         ^
1 error
```

Java Final Method

A Java method with the final keyword is called a final method and it cannot be overridden in the subclass.

In general, final methods are faster than non-final methods because they are not requiredto be resolved during run-time and they are bonded at compile time.

Sample Code:

```
class XYZ
{
  final void demo()
  {
    System.out.println("XYZ Class Method");
  }
}
```

```
public class ABC extends XYZ
{
  void demo()
  {
    System.out.println("ABC Class Method");
  }
  public static void main(String args[])
  {
    ABC obj= new ABC();
    obj.demo();
  }
}
```

Output:

```
/ABC.java:11: error: demo() in ABC cannot override demo() in XYZ
  void demo()
       ^
 overridden method is final
1 error
```

The following code will run fine as the final method demo() is not overridden. This showsthat final methods are inherited but they cannot be overridden.

Sample Code:

```
class XYZ
{
  final void demo()
  {
    System.out.println("XYZ Class Method");
  }
}
public class ABC extends XYZ
{
  public static void main(String args[])
  {
    ABC obj= new ABC();
    obj.demo();
  }
```

}

Output:

XYZ Class Method

Points to be remembered while using final methods:

- Private methods of the superclass are automatically considered to be final.
- Since the compiler knows that final methods cannot be overridden by a subclass, so these methods can sometimes provide performance enhancement by removing calls to final methods and replacing them with the expanded code of their declarations at each method call location.
- Methods made inline should be small and contain only few lines of code. If it growsin size, the execution time benefits become a very costly affair.
- A final's method declaration can never change, so all subclasses use the same method implementation and call to one can be resolved at compile time. This is knownas *static binding*.

Java Final Class

- Final class is a class that cannot be extended i.e. it cannot be inherited.
- A final class can be a subclass but not a superclass.
- Declaring a class as final implicitly declares all of its methods as final.
- It is illegal to declare a class as both abstract and final since an abstract class is incompleteby itself and relies upon its subclasses to provide complete implementations.
- Several classes in Java are final e.g. String, Integer, and other wrapper classes.
- The final keyword can be placed either before or after the access specifier.

Syntax:

```
final public class A
{
   //code
}
```

```
public final class A
{
   //code
}
```

Sample Code:

```
final class XYZ
{
}

public class ABC extends XYZ
{
```

```
    void demo()
    {
      System.out.println("My Method");
    }
    public static void main(String args[])
    {
      ABC obj= new ABC();
      obj.demo();
    }
}
```

Output:

```
/ABC.java:5: error: cannot inherit from final XYZ
public class ABC extends XYZ
                 ^
1 error
```

Important points on final in Java

- Final keyword can be applied to a member variable, local variable, method or classin Java.
- Final member variable must be initialized at the time of declaration or inside the constructor, failure to do so will result in compilation error.
- We cannot reassign value to a final variable in Java.
- The local final variable must be initialized during declaration.
- A final method cannot be overridden in Java.
- A final class cannot be inheritable in Java.

- Final is a different than finally keyword which is used to Exception handling in Java.
- Final should not be confused with finalize() method which is declared in Object class and called before an object is a garbage collected by JVM.
- All variable declared inside Java interface are implicitly final.
- Final and abstract are two opposite keyword and a final class cannot be abstract in Java.
- Final methods are bonded during compile time also called static binding.
- Final variables which are not initialized during declaration are called blank final variable and must be initialized in all constructor either explicitly or by calling this(). Failure to do so compiler will complain as "final variable (name) might not be initialized".

- Making a class, method or variable final in Java helps to improve performance because JVM gets an opportunity to make assumption and optimization.