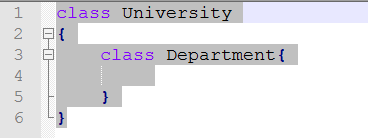
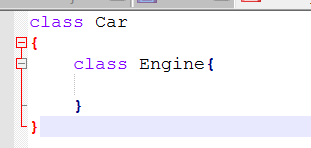
Inner class theory comes in case If there is no chance of existence of inner class object without existence of outer class object.

Outer and Inner class have Has-A relationship. Eg: 1) University has a department 2) Car has a Engine

Eg. There is no chance of existence of department object without existence of University object. But University can exist without Department class.



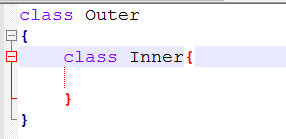


Types of inner class based on behavior and position of declaration:

1. Normal or Regular Inner class
2. Method local Inner class
3. Anonymous Inner Class
4. Static Nested class
5. Normal or Regular Inner class

Any named class directly declared inside a class without static modifier is called Normal/Regular Inner class.

Eg:



Compilation

Javac Outer.java --🡪 Outer.class , Outer$Inner.class

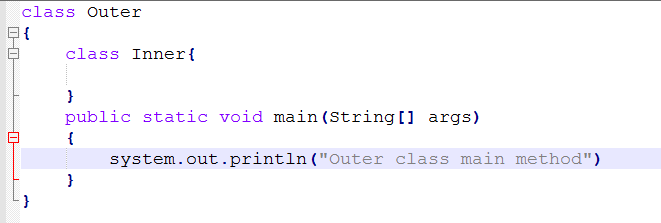
Run:

Java Outer

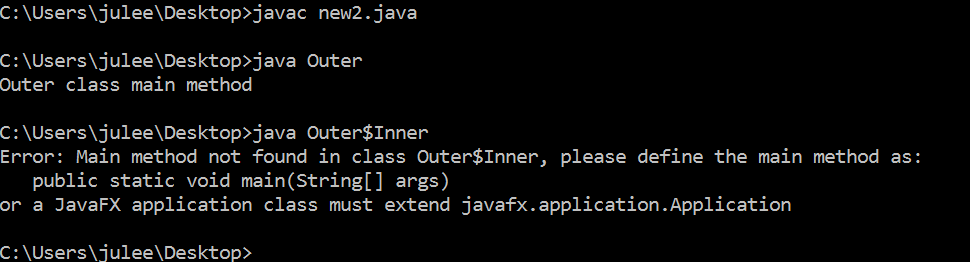
We get run time exception : NoSuchMethodError : Main

Java Outer$Inner

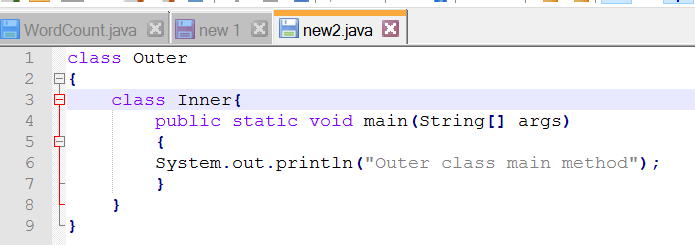
We get run time exception : NoSuchMethodError : Main

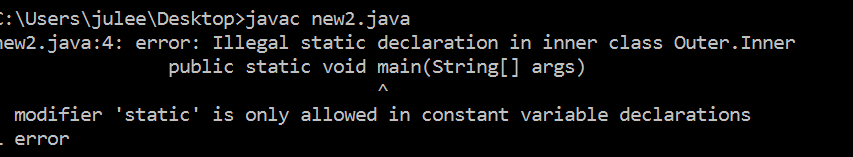


Output:



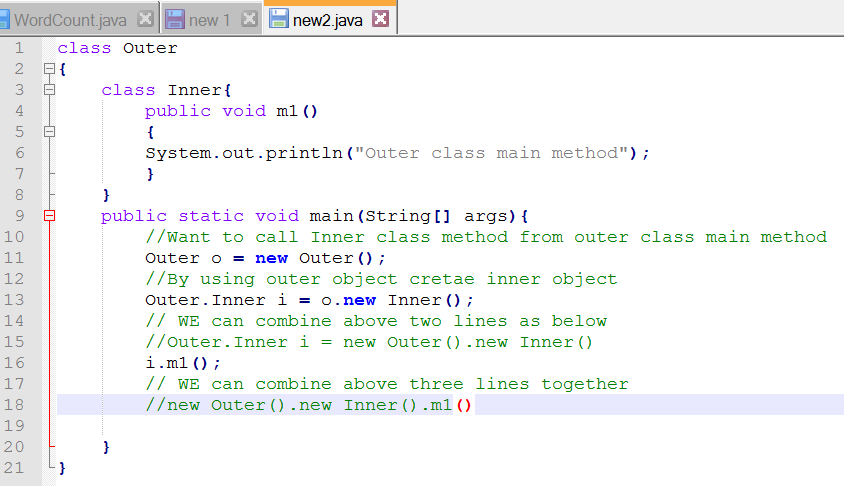
Inside non static inner class we can’t declare any static method.





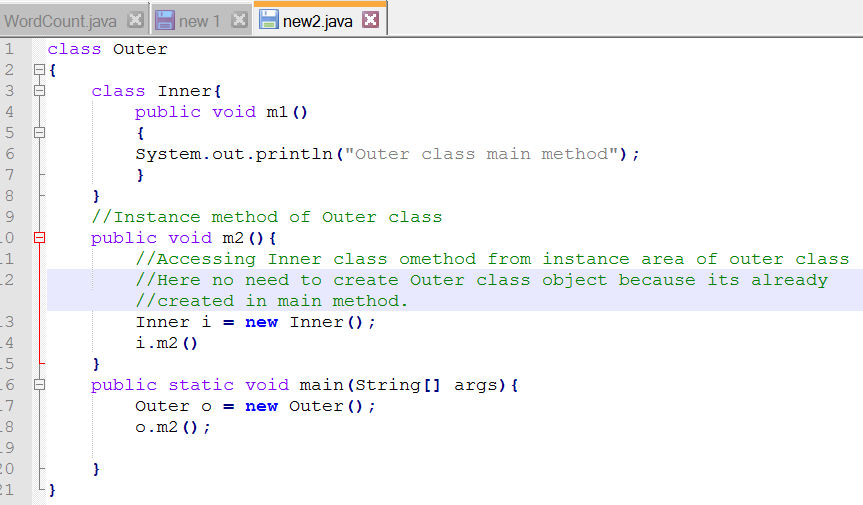
Accessing inner class code from static area of outer class:

Eg:



Case2:

Accessing Inner class code from instance area of Outer class.



Accessing Inner Class Code

Outer o = new Outer();

Outer.Inner i = o.new Inner();

i.m1();

From Instance Area of Outer class

From Static Area of Outer class

or

From Outside of Outer class

Inner i = new Inner();

i.m1();

Corejava-String manipulation-part-1

There are four class for string manipulation

1. String
2. StringBuffer
3. StringBuilder
4. StringTokenizer

String Vs StringBuffer

1. Both are final class(we unable to extand final class)
2. There are two ways of creating object for String class but only one way for StringBuffer

String s1 = “Julee”

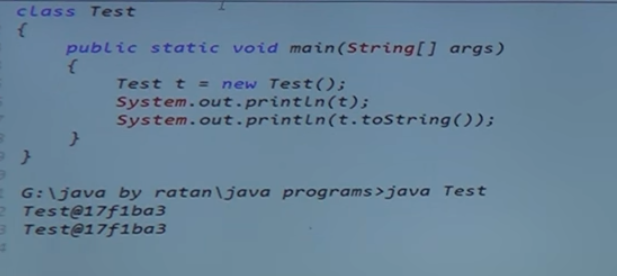
String s2 = new String(“Julee”)

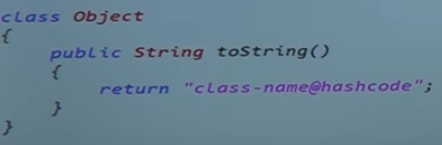
StringBuffer sb = new StringBuffer(“Julee”)

Difference between two approaches of creating objects for String classes.

|  |  |
| --- | --- |
| Without using new | With using new |
| String s1 = “Ratan”  String s2 = “Anu”  String s3 = “Ratan” | String s1 = new String(“Ratan”)  String s2 = new String(“Anu”)  String s3 = new String(“Ratan”) |
| Objects are created in scp memory(string constant pool memory) | Objects are created in Heap memory |
| In SCP area it always checks previous scp objects. If object content matches with any of previous one it would not create new object.  Eg: | There is no overriding.  Three different objects are created |
| It does not allow duplicate | Heap area allows duplicate object. |
| Modifications are not allowed for String object.(Immutable in nature)  Eg: String s1 = “Durga”  S1.concat(“soft”);  System.out.println(s1);  Output : Durga | Modifications are allowed for StringBuffer object  .(Mutable in nature)  Eg:  StringBuffer s1 = new StringBuffer (“Durga”)  S1.append(“Soft”);  System.out.println(s1);  Output : DurgaSoft |
|  |  |
|  |  |

toString:





Whenever we print ref var it calls toString() of object class internally. toString() present in Object class which is the super class of every java class.

HashCode is a hexadecimal number generated by JVM to identify the objects uniquely.

