1 Inner class in java

Inner class means one class which is a member of another class. There are basically four types of inner classes in java.

1. Nested Inner class
2. Method Local inner classes
3. Anonymous inner classes
4. Static nested classes

1.1 Nornal Inner class

main method declarion in outer class

Regular Inner class or Nested Inner class can access any private instance variable of outer class. Like any other instance variable, we can have access modifier private, protected, public and default modifier. Like class, interface can also be nested and can have access specifiers.

```
1  class Outer1
2  {
3      class Inner
4      {
5          public void m1()
6          {
7              System.out.println("Inner class m1 method");
8          }
9      }
10     public static void main(String[] args)
11     {   //1way
12         Outer1 out=new Outer1();
13         Outer1.Inner in1=out.new Inner();
14         in1.m1();
15         //way2
16         //Outer1.Inner in1=new Outer1().new Inner();
17         //in1.m1();
18     }
19 }
```

Outer1.java

```
//Normal Inner class
D:\phani>javac Outer1.java

D:\phani>java Outer1
Inner class m1 method

.....................................
while compiling
it cereates two dot class files

1)Outer1.class
2)Outer1$Inner.class
```

main method declaring inside inner class is a wrong approach

```
class Outer2
{
void m1()
{
Sytem.out.println("outer class method");
}
class Inner
{
public static void main(String[] args)
{
Sytem.out.println("inner class main method");
}
}
}
```

```
D:\phani>java Outer2.java
Error: Could not find or load main class Outer2.java

D:\phani>javac Outer2.java
Outer2.java:5: error: package Sytem does not exist
                Sytem.out.println("outer class method")
                     ^
Outer2.java:11: error: package Sytem does not exist
                        Sytem.out.println("inner class
                             ^
Outer2.java:9: error: Illegal static declaration in inn
                public static void main(String[] args)
                                   ^
  modifier 'static' is only allowed in constant variabl
3 errors
```

outer class instance method calling inner class instance method

```
1  class Outer3
```

```
//Inner class
D:\phani>javac Outer3.java
```

```
{
    class Inner
    {
        public void m1()//instance method of inner class
        {
            System.out.println("inner class m1 method");
        }
    }
    public void m2()//instance method of outer class
    {
        System.out.println("outer class m2 method");
        Inner objinner=new Inner();
        objinner.m1();
    }
    public static void main(String[] args)
    {
        Outer3 obj=new Outer3();
        obj.m2();
    }
}
```

Outer3.java

```
D:\phani>java Outer3
outer class m2 method
inner class m1 method
```

accessing inner class instance method from outer class main method

```
class Outer4
{
    class Inner
    {
        public void m1()//instance method of inner class
        {
            System.out.println("inner class m1 method");
        }
    }
    public static void main(String[] args)
    {
        Outer4 obj = new Outer4();
        Outer4.Inner in = obj.new Inner();
        in.m1();
    }
}
```

Outer4.java

```
//Inner class
D:\phani>javac Outer4.java

D:\phani>java Outer4
inner class m1 method
```

from regula inner classes we can access instance or static members directly

```
class Outer5
{
    int x=10; //instance variable of outer class
    static int y=20; //static variable of outer class
    class Inner
    {
        public void m1()//instance method of inner class
        {
            System.out.println(x);
            System.out.println(y);
        }
    }
    public static void main(String[] args)
```

```
//Outer class variable accessing
 in Inner class instance methods

D:\phani>javac Outer5.java

D:\phani>java Outer5
10
20
```

```
14      {
15          new Outer5().new Inner().m1();
16      }
17  }
```

Outer5.java

accessing variable from different places

```
1  class Outer6
2  {
3      int x=10;  //instance variable of outer class
4      class Inner
5      {
6          int x=100; //instance variable of outer class
7          public void m1()
8          {
9              int x=1000; //local variable
10             System.out.println(x); //1000
11             System.out.println(this.x);//100
12             System.out.println(Outer6.this.x);//10
13         }
14     }
15     public static void main(String[] args)
16     {
17         new Outer6().new Inner().m1();
18     }
19 }
```

Outer6.java

```
//inner class/Outer class variable accessing
 D:\phani>javac Outer6.java

D:\phani>java Outer6
1000
100
10
```

levels of inner class

```
1  class Outer7
2  {
3      class Inner
4      {
5          class Innermost
6          {
7              public void m1()
8              {
9                  System.out.println("inner most class");
10             }
11         }
12     }
13     public static void main(String[] args)
14     {
15         Outer7 obj1 = new Outer7();
16         Outer7.Inner obj2 = obj1.new Inner();
17         Outer7.Inner.Innermost obj3 =obj2. new Innermost();
18         obj3.m1();
19     }
20 }
```

Outer7.java

```
//outer-inner-innermost

D:\phani>javac Outer7.java

D:\phani>java Outer7
inner most class
```

2 Method Local inner classes

Inner class can be declared within a method of an outer class. Using method-local classes can increase the readability of your code by keeping related parts together.

```
1  class Outer8
2  {
3      public void m1()//instance method
4      {
5          class Inner
6          {
7              public void m2()
8              {
9                  System.out.println("inner class m2 method");
10             }
11         }
12         System.out.println("outer class m1 method");
13         Inner objinner= new Inner();
14         objinner.m2();
15     }
16
17     public static void main(String[] args)
18     {
19         Outer8 obj1 = new Outer8();
20         obj1.m1();
21     }
22 }
```

Outer8.java

```
//outer class -instance method- inner class

D:\phani>javac Outer8.java

D:\phani>java Outer8
outer class m1 method
inner class m2 method
```

```
1  class Outer9
2  {
3      public static void m1()//static method
4      {
5          class Inner
6          {
7              public void m2()
8              {
9                  System.out.println("inner class m2 method");
10             }
11         }
12         System.out.println("outer class m1 method");
13         Inner objinner= new Inner();
14         objinner.m2();
15     }
16
17     public static void main(String[] args)
18     {
19         Outer9 obj1 = new Outer9();
20         obj1.m1();
21     }
22 }
```

Outer9.java

```
outer class -static method
 inner class-instance method m2

D:\phani>javac Outer9.java

D:\phani>java Outer9
outer class m1 method
inner class m2 method
```

non-static variables can be accessed by non-static method inner classes
non-static variables van be cannot accessed by inner classe static method

Local inner class cannot access non-final local variable till JDK 1.7. Since JDK 1.8, it is possible to access the non-final local variable in method local inner class.

```
class Outer10
{
    public void m1()//static method
    {
        final int x=100;
        class Inner
        {
            public void m2()
            {
                System.out.println(x);
            }
        }
        Inner objinner= new Inner();
        objinner.m2();
    }

    public static void main(String[] args)
    {
        Outer10 obj1 = new Outer10();
        obj1.m1();
    }
}
```

Outer10.java

```
outer class -instance method- inner class

D:\phani>javac Outer10.java

D:\phani>java Outer10
100
```

inner class static method declarion is not possible

```
class Outer11
{
    public void m1()//static method
    {
        int x=100;
        class Inner
        {
            public static void m2()
            {
                System.out.println(x);
            }
        }
        Inner objinner= new Inner();
        objinner.m2();
    }

    public static void main(String[] args)
    {
        Outer11 obj1 = new Outer11();
        obj1.m1();
    }
}
```

Outer11.java

```
outer class -instance method m1
inner class- static method m2

D:\phani>javac Outer11.java
Outer11.java:8: error:
Illegal static declaration in inner class Inner
                        public static void m2()
                                           ^
  modifier 'static' is only allowed in
constant variable declarations
1 error
```

values accessing from outer ,inner classes

```
class Outer12
{
    int i=10;
    static int j=20;
```

```
outer class -instance ,static variable
instance method m1- instance ,final variable-
accessing from - inner class- instance method m2
D:\phani>javac Outer12.java
```

```
    public void m1()//static method
    {
        int k=30;
        final int l=40;
        class Inner
        {
            public void m2()
            {
                System.out.println(i);
                System.out.println(j);
                System.out.println(k);
                System.out.println(l);
            }
        }
        Inner objinner= new Inner();
        objinner.m2();
    }

    public static void main(String[] args)
    {
        Outer12 obj1 = new Outer12();
        obj1.m1();
    }
}
```

Outer12.java

```
D:\phani>java Outer12
10
20
30
40
```

if we declare m1 method as static then

```
class Outer13
{
    //int i=10;
    static int j=20;
    public static void m1()//static method
    {
        int k=30;
        final int l=40;//final
        class Inner
        {
            public  void m2()
            {
                //System.out.println(i);
                System.out.println(j);
                System.out.println(k);
                System.out.println(l);
            }
        }
        Inner objinner= new Inner();
        objinner.m2();
    }

    public static void main(String[] args)
    {
        Outer13 obj1 = new Outer13();
        obj1.m1();
    }
}
```

Outer13.java

```
outer class -static variable
static method m1- instance ,final variable-
accessing from - inner class- instance method m2

D:\phani>javac Outer13.java

D:\phani>java Outer13
20
30
40
```

declaring static methods in inner classs is not possible

```java
class Outer14
{
    int i=10;
    static int j=20;
    public  void m1()//instance method
    {
        int k=30;
        final int l=40;//final
        class Inner
        {
            public static void m2()
            {
                System.out.println(i);
                System.out.println(j);
                System.out.println(k);
                System.out.println(l);
            }
        }
        Inner objinner= new Inner();
        objinner.m2();
    }

    public static void main(String[] args)
    {
        Outer14 obj1 = new Outer14();
        obj1.m1();
    }
}
```

Outer14.java

```
outer class -instance,static variable
instancec method- instance ,final variable-
accessing from - inner class- static method

D:\phani>javac Outer14.java
Outer14.java:13: error:
non-static variable i cannot be referenced from a static
 context
                                    System.out.println(i);
                                                       ^
Outer14.java:11: error:
Illegal static declaration in inner class Inner
                          public static void m2()
                                             ^
modifier 'static' is only allowed in constant variable dec
2 errors
```

The only applicable modifier for method local inner classes sre final , abstract ,strictfp

3 Anonymous inner classes

It is an inner class without a name and for which only a single object is created. An anonymous inner class can be useful when making an instance of an object with certain "extras" such as overloading methods of a class or interface, without having to actually subclass a class.
Anonymous inner classes are useful in writing implementation classes for listener interfaces in graphics programming.
Anonymous inner class are mainly created in two ways:

1. Class (may be abstract or concrete)
2. Interface

```
// Test can be interface,abstract/concrete class
Test t = new Test()
{
   // data members and methods
   public void test_method()
   {
      ........
      ........
    }
};
```

Example

Types of anonymous inner class : Based on declaration and behavior, there are 3 types of anonymous Inner classes:

1. Anonymous Inner class that extends a class
2. Anonymous Inner class that implements a interface
3. Anonymous Inner class that defines inside method/-constructor argument

```
1  class Sweet
2  {
3      public void taste()
4      {
5          System.out.println("good");
6      }
7  }
8  class Testanonymous
9  {
10     public static void main(String[] args)
11     {
12         Sweet s1=new Sweet()
13         {
14             public void taste()
15             {
16                 System.out.println("s1 --sweet");
17             }
18         };
19         s1.taste();
20         Sweet s2=new Sweet()
21         {
22             public void taste()
23             {
24                 System.out.println("s2 --sweet");
25             }
26         };
27         s2.taste();
28     }
29 }
```

Testanonymous.java

```
Anonymous Inner
D:\phani>javac Testanonymous.java

D:\phani>java Testanonymous
s1 --sweet
s2 --sweet

it creates 3 dot class files

1)Testanonymous$1.class
2)Testanonymous$2.class
3)Testanonymous$1.class
```