**Nested Classes**

**In Java, Just like methods, variables of class too can have another class as its member. The class within a another class.**

**Class Outer\_demo{**

**Class Inner\_Demo{**

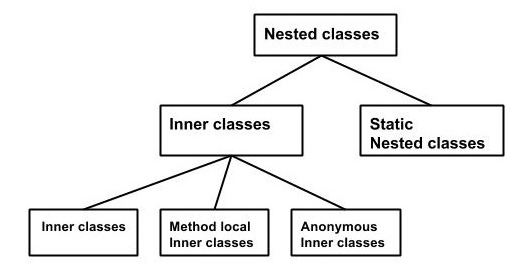
**}**

**}**

**Nested class are divided into two types –**

**Non static nested classes – The class that holds another non-static class.**

**Static nested classes – The class that holds another static class.**

****

**Inner Classes [Non static Nested class]**

**Inner Classes or non static Nested class – are divided into 3 types.**

1. **Inner Class**
2. **Method Local Inner class**
3. **Anonymous Inner Class**

**Inner class –**

**Inner class is a class, that resides another class. You can declare inner class with any of the access modifier.**

class Outer\_Demo {

int num;

// inner class

private class Inner\_Demo {

public void print() {

System.out.println("This is an inner class");

}

}

// Accessing he inner class from the method within

void display\_Inner() {

Inner\_Demo inner = new Inner\_Demo();

inner.print();

}

}

public class My\_class {

public static void main(String args[]) {

// Instantiating the outer class

Outer\_Demo outer = new Outer\_Demo();

// Accessing the display\_Inner() method.

outer.display\_Inner();

}

}

class Outer\_Demo {

// private variable of the outer class

private int num = 175;

// inner class

public class Inner\_Demo {

public int getNum() {

System.out.println("This is the getnum method of the inner class");

return num;

}

}

}

public class My\_class2 {

public static void main(String args[]) {

// Instantiating the outer class

Outer\_Demo outer = new Outer\_Demo();

// Instantiating the inner class

Outer\_Demo.Inner\_Demo inner = outer.new Inner\_Demo();

System.out.println(inner.getNum());

}

}

**2. Method – Inner Local Class**

**We can write a class within a method of another class. Like as a another member of local class inside another class.**

**A method inner local class can be instantiate only within the method where the inner class is defined.**

public class Outerclass {

// instance method of the outer class

void my\_Method() {

int num = 23;

// method-local inner class

class MethodInner\_Demo {

public void print() {

System.out.println("This is method inner class "+num);

}

} // end of inner class

// Accessing the inner class

MethodInner\_Demo inner = new MethodInner\_Demo();

inner.print();

}

public static void main(String args[]) {

Outerclass outer = new Outerclass();

outer.my\_Method();

}

}

**Anonymous Inner Class –**

**A class can declare inside the another class without class name, known as Anonymous Inner class.**

**We declare and instantiate them at the same time.**

**AnonymousClass obj-innerclass = new AnonymousClass{**

**Void diplay(){**

**}**

**}**

**Anonymous class we can declare as Abstract or Interface.**

**Anonymous inner class as Arguments**

**Obj.Method(new MyClass(){**

**Public void display(){**

**///Implement statement**

**}**

**});**

abstract class AnonymousInner {

public abstract void mymethod();

}

public class Outer\_class {

public static void main(String args[]) {

AnonymousInner inner = new AnonymousInner() {

public void mymethod() {

System.out.println("This is an example of anonymous inner class");

}

};

inner.mymethod();

}

}

**Static Nested class –**

**A class can hold another class like its static member. This can be accessed without Instantiate the outer class.**

**Class Outerclass{**

**Static class innerclass{**

**}**

**}**

public class Outer {

static class Nested\_Demo {

public void my\_method() {

System.out.println("This is my nested class");

}

}

public static void main(String args[]) {

Outer.Nested\_Demo nested = new Outer.Nested\_Demo();

nested.my\_method();

}

}