02-September-2017

NON-STATIC NESTED CLASS

- In Non-Static nested class if 'Inner' class is not inheriting 'x' (Non-Static D.M.) of 'Outer' class, then how is it using 'x' in its 'show()' f^n without creating object of outer class.

So, To see this process of using Non-Static Data Member without inheriting we use a tool of java known as 'javap'.

'javap' tool → 50%
↳ Gives prototype of all functions, Constructors, data members of .class files.

To use javap Tool.

```
D:\f1 > javac Outer.java
D:\f1 > javap Outer
```

Now to redirect all this information to a text file

```
D:\f1 > javap Outer > abc.txt
```

ON USING javap

In jdk 1.7.

On using on Outer class

```
class Outer extends java.lang.Object
{
```

shows every class has 'Object' class.

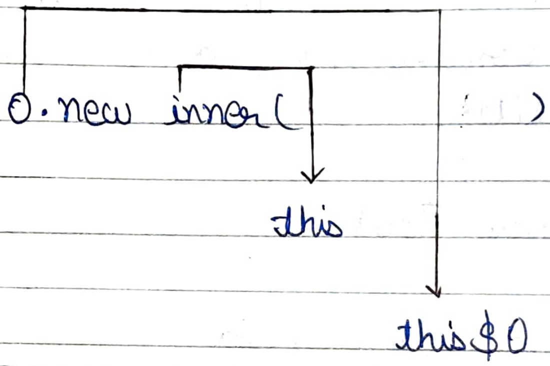
Using On Inner Class

```
class Outer$Inner extends java.lang.Object
 {
   final Outer this$0;
    Outer$Inner(Outer);
     void show();
 }
```

Being used in 'show()' fn

used when x is used by 'this$0' is placed before 'x' & a '.' in b/w them.

★ This is how inner class access 'x'.



Using Non-Static Nested Class by extending in Outer Class.

```
class Temp extends Outer.Inner
 {

       missing

 }
```

• This program will give error. why

LOCAL NESTED CLASS

- To use a local Data Member in Local Nested Class we have had to make Data Member 'final' till jdk 1.7

~~Using javap~~

~~class~~
~~{~~

But Now its not required to make Local Data members 'final' from jdk 1.8

Example:-

```
class LocalInner1
{
int x=10;
static int y=100;

void display()
{
int p=30; //it had to be
              final till jdk 1.7

class Inner
{
public void show()
{
Sop(p);
Sop(x);
Sop(y);
}
}
Inner m= new Inner();
m.show();
}

psvm()
{
LocalInner1 O
     = new LocalInner1();

O.display();
}
}
```

Example:- Running show() from Outer Class i.e. LocalInner

```
class LocalInner
{
int x=10;
static int y=100;

My display()
{
int p=30;

class Inner implements My
{
public void show()
 {
 Sop(p);
 Sop(x);
Sop(y);
}
}
My m=new Inner();
return m;
 }
```

```
psvm()
{
LocalInner O=new LocalIn-
                    -ner();

My z= O.display();
z.show()
}
}
```

```
interface My
 {
 void show();
 }
```