

```

package innerClasses;

public class Demo1 {

    public static void main(String[] args) {

        EnclosingClass encClass1 = new EnclosingClass();
        encClass1.demoInnerClass();
        System.out.println(" ");

        EnclosingClass encClass2 = new EnclosingClass();
        encClass2.demoLocalInnerClass();
        System.out.println(" ");

        EnclosingClass encClass3 = new EnclosingClass();
        encClass3.demoLocalAnonymousInnerClass();
        System.out.println(" ");

        EnclosingClass.StaticClass staticClass = new EnclosingClass.StaticClass();
        System.out.println(staticClass.getName());

    }

}

/* ----- output -----

    --- start demoInnerClass ---
Enclosing Class Default Name
Enclosing Class Name Set By InnerClass
3.14
    --- end demoInnerClass ---

    --- start demoLocalInnerClass ---
Enclosing Class Default Name
Enclosing Class Name Set By LocalInnerClass
    --- end demoLocalInnerClass ---

    --- start demoLocalAnonymousInnerClass ---
BaseClass Default Name
LocalAnonymousInnerClass doSomething()
    --- end demoLocalAnonymousInnerClass ---

StaticClass Default Name

    ----- endoutput -----
*/

```

```

package innerClasses;

public class Demo2 {

    public static void main(String[] args) {
        // static nested class demo
        EnclosingClass.StaticClass staticClassRef1 = new EnclosingClass.StaticClass();
        System.out.println("staticClassRef1.getName(): " + staticClassRef1.getName() );

        // inner class demo (needs an EnclosingClass object)

        // The following line gives a compile time error.
        //EnclosingClass.InnerClass innerClassRef = new EnclosingClass.InnerClass();

        //To create an InnerClass object, first an EnclosingClass object needs
        // to be instantiated
        EnclosingClass enclosingClassRef1 = new EnclosingClass();

        //The following syntax is used to create an InnerClass object through an
        //EnclosingClass Object
        EnclosingClass.InnerClass innerClassRef1 =
            enclosingClassRef1.new InnerClass();

        System.out.println("innerClassRef1.getString(): "+innerClassRef1.getString());

        enclosingClassRef1.setString("string changed through EnclosingClassRef");
        System.out.println("innerClassRef1.getString(): "+innerClassRef1.getString());

        innerClassRef1.setString("String Changed through InnerClassRef");
        System.out.println("innerClassRef1.getString(): "+innerClassRef1.getString());

        EnclosingClass.InnerClass innerClassRef2 = enclosingClassRef1.new InnerClass();

        System.out.println("innerClassRef2.getString(): "+innerClassRef2.getString());
        innerClassRef2.setString("String Changed through InnerClassRef2");

        System.out.println("innerClassRef2.getString(): "+innerClassRef2.getString());
        System.out.println("innerClassRef1.getString(): "+innerClassRef1.getString());

    }

}

/* ----- output -----

staticClassRef1.getName(): StaticClass Default Name
innerClassRef1.getString(): Enclosing Class Default Name
innerClassRef1.getString(): string changed through EnclosingClassRef
innerClassRef1.getString(): String Changed through InnerClassRef
innerClassRef2.getString(): String Changed through InnerClassRef
innerClassRef2.getString(): String Changed through InnerClassRef2
innerClassRef1.getString(): String Changed through InnerClassRef2

----- endoutput -----
*/

```

```

package innerClasses;

class EnclosingClass {

    private String itsString = "Enclosing Class Default Name";
    private InnerClass innerClass = new InnerClass();

    public void setString(String itsString){
        this.itsString = itsString;
    }

    public void demoInnerClass() {
        System.out.println(" --- start demoInnerClass --- ");
        System.out.println(innerClass.getString());
        innerClass.setString("Enclosing Class Name Set By InnerClass");
        System.out.println(itsString);
        System.out.println(innerClass.getDouble());
        System.out.println(" --- end demoInnerClass --- ");
    }

    public void demoLocalInnerClass() {
        class LocalInnerClass {
            public void setString(String aString) {
                itsString = aString;
            }

            public String getString() {
                return itsString;
            }
        }
        System.out.println(" --- start demoLocalInnerClass --- ");
        LocalInnerClass localInnerClass = new LocalInnerClass();
        System.out.println(localInnerClass.getString());
        localInnerClass.setString("Enclosing Class Name Set By LocalInnerClass");
        System.out.println(localInnerClass.getString());
        System.out.println(" --- end demoLocalInnerClass --- ");
    }

    public void demoLocalAnonymousInnerClass() {
        BaseClass baseClass = new BaseClass() {

            public void doSomething() {
                System.out.println("LocalAnonymousInnerClass doSomething()");
            }

        } ;

        System.out.println(" --- start demoLocalAnonymousInnerClass --- ");
        System.out.println(baseClass.getString());
        baseClass.doSomething();
        System.out.println(" --- end demoLocalAnonymousInnerClass --- ");
    }

    // inner class "InnerClass" is an instance memembr of "EnclosingClass"
    public class InnerClass{

        private double itsDouble = 3.14;

        public String getString(){
            // same as: return EnclosingClass.this.itsString;
            return itsString;
        }
    }
}

```

```
}
public void setString(String aString){
    // same as: EnclosingClass.this.itsString = aString;
    itsString = aString;
}

public double getDouble(){
    // itsDouble is short for this.itsDouble
    return itsDouble;
}

} // end InnerClass

// inner class "StaticClass" is a class member (static member) of EnclosingClass
public static class StaticClass {

    private String itsName = "StaticClass Default Name";

    public String getName(){
        return itsName;
    }

} // end StaticClass

}
```

```
package innerClasses;

public class BaseClass {

    private String itsString = "BaseClass Default Name";

    public String getString() {
        return itsString;
    }

    public void doSomething() {
        System.out.println("Base Class doSomething");
    }

}
```