2014.Java7AComprehensiveTutorial

Nested classes

1. static inner (nested?) classes

2. non-static nested classes (inner classes)

2.1 member inner classes

2.2 local inner classes

2.3 anonymous inner classes

# INTRODUCING NESTED AND INNER CLASSES.

**2014\_JavaTheCompleteReference\_HerbertSchildt – Chapter 7**

* It is possible to define a class within another class: such class are known as ***nested classes.***
* The scope of a nested class is bounded by the scope of its enclosing tags.
* If class B is defined within class A, then B does not exist independently of A.
* A nested class has access to the members, including private members, on the class in which is nested.
* The enclosing class (clase externa) does not have access to the members of the nested class.
* A nested class that is declared directly within its enclosing tag class scope is a member of its enclosing class.
* It is also possible to declare a nested class that is local to a block.
* There are two types of nested classes: **static** and **non-static (inner class)**
* A **static nested class** is one that has the static modifier applied.
* Because it is static, it must access the non-static members of its enclosing class (clase externa) through an object.
* It cannot refer to non-static members of its enclosing class directly.
* Because of this restriction, static nested classes are seldom used.

**INNER CLASS**

* The most important type of nested class is the **inner class**.
* An **inner class** is a **non-static nested class**.
* It has access to all of the variables and methods of its outer class and may refer to them directly in the same way that other non-static members of the outer class do.
* The following program illustrates how to define and use an **inner class**.

|  |  |
| --- | --- |
|  |  |
|  | x = 100 |

* The class named **Outer** has one instance variable named **x**, one instance method name **test()**, and defines one inner lass called **Inner**.
* An inner class named **Inner** is defined within the scope of class **Outer.**
* Any code in class **Inner** can directly access the variable **x**.
* An instance method named **display()** is defined inside **Inner**. This display **x**.
* In main, an instance of class **Outer** is created and invokes its **test()** method.
* That method creates an instance of class **Inner** and the display method is call.
* Is important to realize that an instance of **Inner** can be created only in the context of class **Outer**.
* An inner class instance is often created by code within its enclosing scope.

|  |  |
| --- | --- |
|  | * An inner class has access to all of the members of its enclosing class, but the reverse is not true. * Members of the inner class are known only within the scope of the inner class and may not be used by the outer class. * Here, **y** is declared as a instance variable of **Inner**, it is not known outside of that class and it cannot be used by **showy()** |

**INNER CLASS IN A BLOCK**

* It is possible to define a inner class within any block scope.
* You can define a nested class within the block defined by a method
* Or even within the body of a for lop.

|  |  |
| --- | --- |
|  | x = 100  x = 100  x = 100  x = 100  x = 100  x = 100  x = 100  x = 100  x = 100  x = 100 |

* While nested classes are not applicable to all situations, they are particularly helpful when handling events.
* Inner classes can be used to simplify the code needed to handle certain type of events.
* ***Anonymous inner classes***are inner classes that don´t have a name.