**Nested Classes**

**Define a class within another class. Such a class is called as Nested class.**

* **2 Categories:** STATIC and NON-STATIC
* Nested classes that are declared static are called static nested classes.
* Non static nested classes are called inner classes

classOuterClass {

...

static class StaticNestedClass {

...

}

classInnerClass {

...

}

}

* A nested class is a member of its enclosing class**, Non-Static class** have access to all members of its enclosing class even if members are **declared as private**.
* Static Nested classes **do not have access** to other members of enclosing class.
* Nested class can be declared as **private, public, protected or package private**.

**Uses of Nested Classes:**

* **Logical grouping:** if a class is useful to only one other class, it’s logical to embed it in that class and keep both together. (**Nesting such "helper classes" makes their package more streamlined**).
* **Increase Encapsulation:**Consider 2 top level classes A and B, b needs access to members of A. Otherwise those members in A would have declared private.

By hiding class B inside A, since B has access to members in A even its declared as private.in addition class B can be hidden from outside world.

* **It can lead to more readable and maintainable code:** Nesting small classes within top-level classes places the code closer to where it is used.

**Static Nested Classes:**

* As Like static class methods, static nested class **cannot refer directly** to instance variables and methods defined in enclosing class. It can use them through only object reference.
* Static Nested classes are accessed using enclosing class name:

OuterClass.StaticNestedClass

**for example:**

OuterClass.StaticNestedClass staticNestedClass = new OuterClass.StaticNestedClass();

**Inner Classes:**

* As with instance methods and variables , an inner class is also associated with an **instance of enclosing class** and has **direct access** to that object's methods and fields.
* Also an inner class is associated with an instance, **it cannot** define any static member itself.
* An inner class can only exist within a instance of **OuterClass** and has direct access to methods and variables of its enclosing instance.
* To instantiate an inner class, you must first instantiate the outer class. Then, create the inner object within the outer object with this syntax:

OuterClass outerClass = new OuterClass();

OuterClass.InnerClass innerObject = outerObject.new InnerClass();

**Inner Class Example:**

public class DataStructure {

// Create an array

private final static int SIZE = 15;

private int[] arrayOfInts = new int[SIZE];

public DataStructure() {

// fill the array with ascending integer values

for (int i = 0; i < SIZE; i++) {

arrayOfInts[i] = i;

}

}

public void printEven() {

// Print out values of even indices of the array

DataStructureIterator iterator = this.new EvenIterator();

while (iterator.hasNext()) {

System.out.print(iterator.next() + " ");

}

System.out.println();

}

interface DataStructureIterator extends java.util.Iterator<Integer> { }

// Inner class implements the DataStructureIterator interface,

// which extends the Iterator<Integer> interface

private class EvenIterator implements DataStructureIterator {

// Start stepping through the array from the beginning

private int nextIndex = 0;

public boolean hasNext() {

// Check if the current element is the last in the array

return (nextIndex <= SIZE - 1);

}

public Integer next() {

// Record a value of an even index of the array

Integer retValue = Integer.valueOf(arrayOfInts[nextIndex]);

// Get the next even element

nextIndex += 2;

return retValue;

}

}

public static void main(String s[]) {

// Fill the array with integer values and print out only

// values of even indices

DataStructure ds = new DataStructure();

ds.printEven();

}

}

The output is:

0 2 4 6 8 10 12 14

There are two special kinds of **inner classes:**

* [local classes](https://docs.oracle.com/javase/tutorial/java/javaOO/localclasses.html)
* [Anonymous classes](https://docs.oracle.com/javase/tutorial/java/javaOO/anonymousclasses.html).