




Bộ môn Công nghệ Phần mềm  
Viện CNTT & TT  
Trường Đại học Bách Khoa Hà Nội

## LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG

### Bài 4B. Nested Class

## 1. Khái niệm

- Java cho phép nhúng 1 class trong class khác → G í là **nested class**
- Ví dụ :  

```
class OuterClass {
    ...
    class NestedClass {
        ...
    }
}
```

2

## 2. Tại sao sử dụng nested class?

3

## 3. Phân loại

- Nested class chia làm 2 loại:
- Ví dụ :  

```
class OuterClass {
    ...
    static class StaticNestedClass {
        ...
    }
    class InnerClass {
        ...
    }
}
```

4

### 3.1. Static nested classes

- Có truy cập tới tên của class bao nó
- Chỉ có 1 instance của static nested class:
- Chỉ có truy cập các thành viên static của class bao nó

5

### 3.1. Static nested classes (2)

```
public class Outside {
    public static class Skinside {
        public Skinside()
        {
            System.out.println("Demo static");
        }
    }

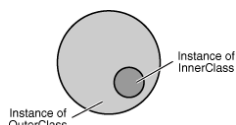
    public class Inside {
    }

    public static void main(String[] arg) {
        Outside.Skinside example = new Outside.Skinside();
    }
}
```

6

## 3.2. Inner Class

- 1 th hi n (instance) c a inner class ch t n t i c trong 1 th hi n c a outer class



7

## 3.2. Inner Class (2)

- Inner class có th truy c p t i l member b t k c a outer class
  - Inner class không c có thành ph n static
- ```
public class Outer {
    private int id;
    private class Inner
    {
        private static int defaultId; //Error
        public Inner()
        {
            id = 00001; //Truy c p c id ngoài
        }
    }
}
```

8

```
public class DataStructure {
    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() { //In ch s l trong m ng
        InnerEvenIterator iterator = this.new InnerEvenIterator();
        while (iterator.hasNext()) {
            System.out.println(iterator.getNext() + " ");
        }
    }
    private class InnerEvenIterator { //inner class implements the Iterator pattern
        //start stepping through the array from the beginning
        private int next = 0;
        public boolean hasNext() {
            return (next <= SIZE - 1); //check if current element is the last in the array
        }
        public int getNext() {
            int retValue = arrayOfInts[next];
            next += 2; //get the next even element
            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();
    }
}
```

9

## 3.2. Inner Class (3)

- Inner Class l i chia làm 2 lo i con:

10

## a. local inner class

```
class Outer {
    int outer_x = 100;

    void test() {
        for(int i=0; i<10; i++) {
            class Inner {
                void display() {
                    System.out.println("display: outer_x = " + outer_x);
                }
            }
            Inner inner = new Inner();
            inner.display();
        }
    }
}

class InnerClassDemo {
    public static void main(String args[]) {
        Outer outer = new Outer();
        outer.test();
    }
}
```

11

## b. anonymous inner classes

```
interface Counter {
    int next();
}

public class Outer {
    private int count = 0;
    Counter getCounter(final String name) {
        return new Counter() {
            {
                System.out.println("Constructor Counter()");
            }
            public int next() {
                System.out.print(name); // Access local final
                System.out.println(count);
                return count++;
            }
        };
    }

    public static void main(String[] args) {
        Outer out = new Outer();
        Counter c1 = out.getCounter("Local inner ");
        c1.next();
        c1.next();
    }
}
```

12