Inner Class

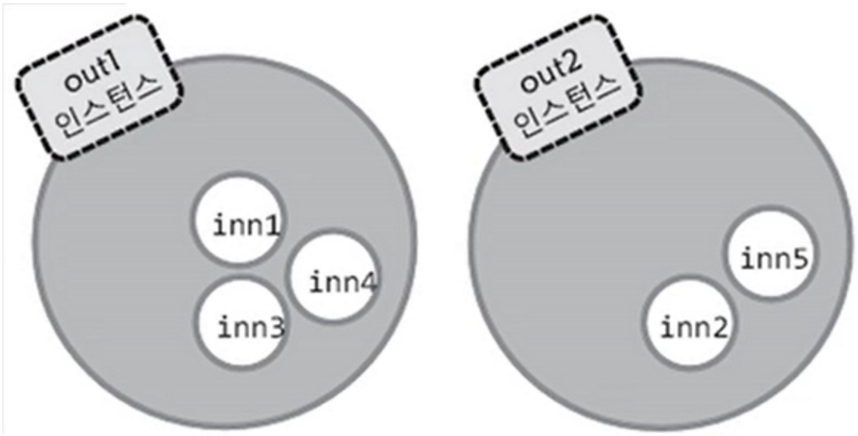
2016년 9월 29일 목요일 오전 1:00

Inner Class

```
// OuterClass
class OuterClass {

	// InnerClass
	class InnerClass {

	}

}

-----------------------------------------------------------------------------------------------------

class OuterClass {
	private String myName;
	private int num;

	OuterClass(String name){
		myName = name;
		num = 0;
	}

	public void whoAreYou(){
		num++;
		System.out.println(myName+" OuterClas "+num);
	}

	class InnerClass {
		InnerClass(){
			whoAreYou();
			// InnerClass는 OuterClass의 Member에 접근가능하다.
			// 중요한건, Method의 호출은 OuterClass가 Instance 생성이 되고 난 후에야 가능
		}
	}
}

public class InnerClassTest {
	public static void main(String[] args) {
		OuterClass out1 = new OuterClass("First");
		OuterClass out2 = new OuterClass("Second");
		out1.whoAreYou();
		out2.whoAreYou();

		OuterClass.InnerClass inn1 = out1.new InnerClass();
		                             // out1 인스턴스에 종속적인 InnerClass 인스턴스를 생성하겠다.

		OuterClass.InnerClass inn2 = out2.new InnerClass();
		OuterClass.InnerClass inn3 = out1.new InnerClass();
		OuterClass.InnerClass inn4 = out1.new InnerClass();
		OuterClass.InnerClass inn5 = out2.new InnerClass();
	}
```

```
}
```

i. InnerClass의 인스턴스는 OuterClass의 인스턴스에 의존적이다, 인스턴스 안의 인스턴스
ii. OuterClass의 인스턴스 생성 후에야 InnerClass의 인스턴스 생성이 가능하다.
iii. OuterClass의 멤버에 접근이 가능하다!




```
---------------------------------------------------------------------------------------------------
Command Prompt
First OuterClas 1
Second OuterClas 1
First OuterClas 2
Second OuterClas 2
First OuterClas 3
First OuterClas 4
Second OuterClas 3
---------------------------------------------------------------------------------------------------
```

Static Inner Class (Nested Class)

```
// OuterClass
class OuterClass {

     // Static Inner Class 또는 Nested Class
     static class InnerClass {

     }

}
---------------------------------------------------------------------------------------------------

class OuterClassOne {
     OuterClassOne(){
          NestedClass nst = new NestedClass();
          // OuterClass 내부에서 static인 Nested Class의 Instance 생성 가능
          nst.simpleMethod();
     }

     static class NestedClass {
          public void simpleMethod(){
               System.out.println("Nested Instance Method One");
          }
     }
}

public class NestedClassTest {
```

```
    public static void main(String[] args) {
        OuterClassOne one = new OuterClassOne();

        OuterClassOne.NestedClass nst1 = new OuterClassOne.NestedClass();
        //  Nested Class는 static 이므로 OuterClass 외부에서의 접근은 Outerclass.NestedClass가 된다.
        nst1.simpleMethod();
    }
}
```

- OuterClass와 NestedClass는 각각의 instance가 생성이 되면 남남이 되버린다.
  (NestedClass는 Static, OuterClass 내부에 자리만 빌려서 들어왔으므로)