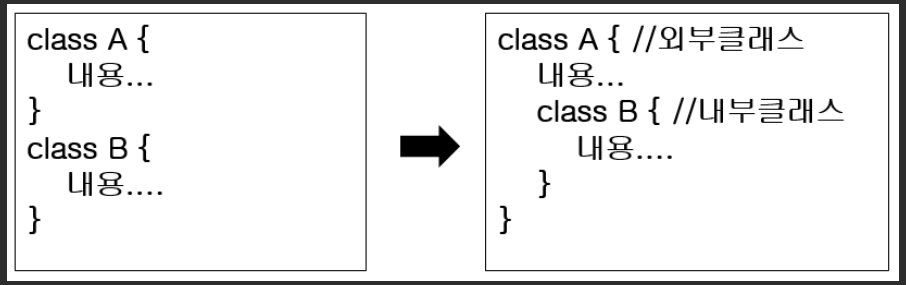
Java 기초

내부 클래스

내부 클래스

- 내부 클래스(inner class)의 정의
  - 클래스 안에 선언된 클래스
  - 특정 클래스 내에서만 주로 사용되는 클래스를 내부 클래스로 선언한다.
  - GUI어플리케이션(AWT, Swing)의 이벤트처리에 주로 사용된다.



- 내부 클래스의 장점
  - 내부 클래스에서 외부 클래스의 멤버들을 쉽게 접근할 수 있다.
  - 코드의 복잡성을 줄일 수 있다.(캡슐화)

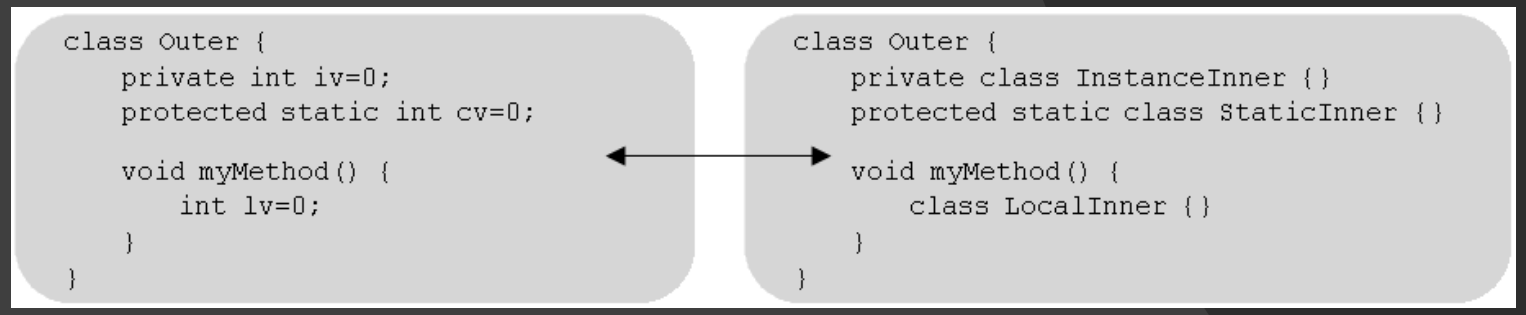
내부 클래스

- 내부 클래스(inner class)의 종류

인스턴스클래스 (instance class)	외부클래스의 멤버변수 선언위치에 선언하며, 외부클래스의 인스턴스멤버처럼 다루어진다. 주로 외부클래스의 인스턴스멤버들과 관련된 작업에 사용될 목적으로 선언된다.
스태틱클래스 (static class)	외부클래스의 멤버변수 선언위치에 선언하며, 외부클래스의 static멤버처럼 다루어진다. 주로 외부클래스의 static멤버, 특히 static메서드에서 사용될 목적으로 선언된다.
지역클래스 (local class)	외부클래스의 메서드나 초기화블럭 안에 선언하며, 선언된 영역 내부에서만 사용될 수 있다.
익명클래스 (anonymous class)	클래스의 선언과 객체의 생성을 동시에 하는 이름없는 클래스(일회용)

내부 클래스

- 내부 클래스의 제어자와 접근성(1/7)
  - 내부 클래스의 접근제어자는 변수에 사용할 수 있는 접근제어자와 동일하다.



내부 클래스

- 내부 클래스의 제어자와 접근성(2/7)
  - static클래스만 static멤버를 정의할 수 있다.

```java
public class InnerClass {
    class B{ //인스턴스 클래스
        B(){}  // 생성자 가능
        int field1; //필드 가능
        //static int field2; // 정적 필드 불가능
        void method1(){} //인스턴스 메소드 가능
        //static void method2(){} //정적 메소드 불가능
    }
    static class C{ //인스턴스 클래스
        C(){}  // 생성자 가능
        int field1; // 필드 가능
        static int field2; // 정적 필드 가능
        void method1(){} //인스턴스 메소드 가능
        static void method2(){} //정적 메소드 가능
    }
}
void method1(){
    class D{
        D(){} //생성자 가능
        int field1; //인스턴스 필드 가능
        //static int field2 //정적 필드 불가능
        void method1(){} //인스턴스 메소드 가능
        //static void method2(){} //정적 메소드 불가능
    }
    D d = new D();
    d.field1 = 3;
    d.method1();
}
```

내부 클래스

- 내부 클래스의 제어자와 접근성(3/7)
  - 인스턴스 클래스의 인스턴스를 생성하려면 반드시 외부 클래스를 먼저 선언해야 한다..
  - 정적 클래스는 외부 클래스의 인스턴스 선언 없이 선언이 가능하다.

```
public static void main(String[] args) {

    InnerClass ic = new InnerClass();
    InnerClass.B b = ic.new B();
    b.field1 = 1;
    b.method1();

    InnerClass.C c = new InnerClass.C();
    c.field1 = 1;
    c.field2 = 2;
    c.method1();
    c.method2();
}
```

내부 클래스

- 내부 클래스의 제어자와 접근성(4/7)
  - 내부 클래스도 외부 클래스의 멤버로 간주되며, 동일한 접근성을 갖는다.

```
public class InnerClass {

    //인스턴스 필드
    B field1 = new B(); //(o)
    C field2 = new C(); //(o)
    //인스턴스 메소드
    void method1(){
        B var1 = new B(); //(o)
        C var2 = new C(); //(o)
    }
    //정적 필드
    //static B field3 = new B(); //(x)
    static C field4 = new C(); //(o)
    //정적 인스턴스 메소드
    static void method2(){
        //B var1 = new B(); //(x)
        C var2 = new C(); //(o)
    }

    class B{ //인스턴스 클래스
    static class C{ //정적 클래스
```

내부 클래스

- 내부 클래스의 제어자와 접근성(5/7)
  - 정적 내부 클래스는 외부 클래스의 필드와 메서드 접근 시 제약이 따른다.

```java
public class InnerClass2 {
    int field1;
    void method1(){};

    static int field2;
    static void method2(){};

    class B{
        void method(){
            field1 = 10;
            method1();

            field2 = 10;
            method2();
        }
    }
```

```java
public class InnerClass2 {
    int field1;
    void method1(){};

    static int field2;
    static void method2(){};

    static class B{
        void method(){
            field1 = 10; //에러
            method1();   //에러

            field2 = 10;
            method2();
        }
    }
```

내부 클래스

- 내부 클래스의 제어자와 접근성(6/7)
  - 외부 클래스의 지역변수는 final이 붙은 변수(상수)만 접근가능하다. 지역 클래스의 인스턴스가 소멸된 지역변수를 참조할 수 있기 때문이다.

```
public void method1(final int arg1, int arg2){
    final int var1 = 1;
    int var2 = 2;
    //var2 = 3; //자동 final 변환
    //arg2 = 4; //자동 final 변환
    class LocalClass{
        void method(){
            int result = var1+var2+arg1+arg2;
        }
    }
}
```

내부 클래스

- 내부 클래스의 제어자와 접근성(7/7)
  - 인터페이스를 이용한 내부 인터페이스 선언이 가능하다.

```
public class Button {
    OnClickListener listener;

    void setOnClickListener(OnClickListener listener){
        this.listener = listener;
    }

    void touch(){
        listener.onClick();
    }

    interface OnClickListener{
        void onClick();
    }
}
```

익명 클래스(Anonymous Class)

- 익명 클래스(Anonymous Class)
  - 이름이 없는 일회용 클래스. 단 하나의 객체만을 생성할 수 있다.

```
new 조상클래스이름() {
   // 멤버 선언
}

   또는

new 구현인터페이스이름() {
   // 멤버 선언
}
```

익명 클래스(Anonymous Class)

- 익명 클래스(Anonymous Class) 예제

```
public class InnerClassTest4 {
    public InnerClassTest4(InnerAbstract ia) {
        ia.method1();
    }
    public static void main(String[] args) {
        InnerClassTest4 ict4 =
                new InnerClassTest4(new InnerAbstract() {
                    public void method1() {
                        System.out.println("이것은 익명클래스입니다");
                    }
                });
    }
}
```

```
이것은 익명클래스입니다
```