

중첩 클래스의 접근 제한

중첩 클래스의 접근 제한

❖ 바깥 필드와 메소드에서 사용 제한

```
public class A {  
    //인스턴스 멤버 클래스  
    class B {}  
  
    //정적 멤버 클래스  
    static class C {}  
}
```

```
public class A {  
    //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 초기화  
    //static B field3 = new B();    ----- (x)  
    static C field4 = new C();      ----- (o)  
  
    //정적 메소드  
    static void method2() {  
        //B var1 = new B();        ----- (x)  
        C var2 = new C();          ----- (o)  
    }
```

중첩 클래스의 접근 제한

❖ 바깥 필드와 메서드에서 사용 제한: A.java

```
public class A {  
    // 인스턴스 필드  
    B field1 = new B();  
    C field2 = new C();  
  
    // 인스턴스 메소드  
    void method1() {  
        B var1 = new B();  
        C var2 = new C();  
    }  
  
    // 정적 필드 초기화  
    // static B field3 = new B();  
    static C field4 = new C();  
  
    // 정적 메소드  
    static void method2() {  
        // B var1 = new B();  
        C var2 = new C();  
    }  
}
```

중첩 클래스의 접근 제한

❖ 바깥 필드와 메서드에서 사용 제한: A.java

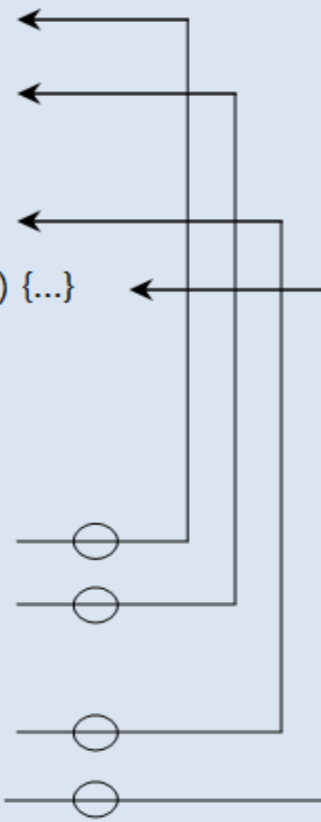
```
// 인스턴스 멤버 클래스
class B {
}

// 정적 멤버 클래스
static class C {
}
}
```

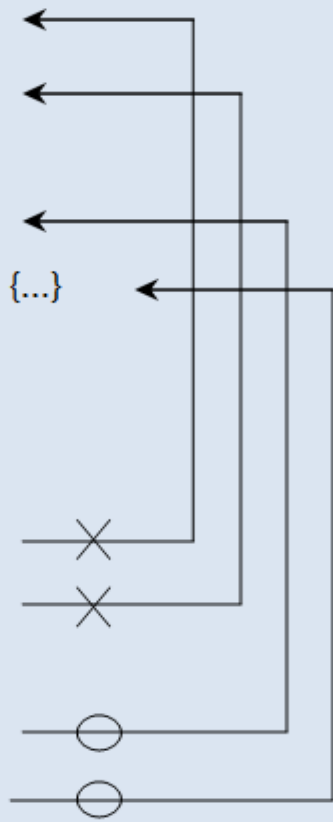
중첩 클래스의 접근 제한

❖ 멤버 클래스에서 사용 제한

```
class A {  
    int field1;  
    void method1() { ...}  
  
    static int field2;  
    static void method2() {...}  
  
    class B {  
        void method() {  
            field1 = 10;  
            method1();  
  
            field2 = 10;  
            method2();  
        }  
    }  
}
```



```
class A {  
    int field1;  
    void method1() { ...}  
  
    static int field2;  
    static void method2() {...}  
  
    static class C {  
        void method() {  
            field1 = 10;  
            methodA1();  
  
            fieldA2 = 10;  
            methodA2();  
        }  
    }  
}
```



중첩 클래스의 접근 제한

❖ 멤버 클래스에서 사용 제한: A.java

```
public class A {  
    int field1;  
  
    void method1() {  
    }  
  
    static int field2;  
  
    static void method2() {  
    }  
  
    class B {  
        void method() {           // 모든 멤버와 메서드에서 접근할 수 있다.  
            field1 = 10;  
            method1();  
  
            field2 = 10;  
            method2();  
        }  
    }  
}
```

중첩 클래스의 접근 제한

❖ 멤버 클래스에서 사용 제한: A.java

```
static class C {  
    void method() {        // 인스턴스 필드와 메서드는 접근할 수 없다.  
        // field1 = 10;  
        // method1();  
  
        field2 = 10;  
        method2();  
    }  
}  
}
```

중첩 클래스의 접근 제한

❖ 로컬 클래스에서 사용 제한

```
void outMethod(final int arg1, int arg2) {  
    final int var1 = 1;  
    int var2 = 2;  
    class LocalCalss {  
        void method() {  
            int result = arg1+arg2+var1+var2;  
        }  
    }  
}
```



```
class LocalClass {  
    int arg2 = 매개값;    필드로 복사  
    int var2 = 2;  
    void method() {  
        int arg1 = 매개값;    로컬 변수로 복사  
        int var1 = 1;  
        int result = arg1+arg2+var1+var2;  
    }  
}
```


중첩 클래스의 접근 제한

❖ 로컬 클래스에서 사용 제한

```
public class Outer {  
    //자바7 이전  
    public void method1(final int arg) {  
        final int localVariable = 1;  
        //arg = 100; (x)  
        //localVariable = 100; (x)  
        class Inner {  
            public void method() {  
                int result = arg + localVariable;  
            }  
        }  
    }  
}
```

final 매개변수와 로컬 변수는
로컬 클래스의 메소드의 로컬변수로 복사
(final 붙이지 않으면 컴파일 오류 발생)

```
//자바8 이후  
public void method2(int arg) {  
    int localVariable = 1;  
    //arg = 100; (x)  
    //localVariable = 100; (x)  
    class Inner {  
        public void method() {  
            int result = arg + localVariable;  
        }  
    }  
}
```

매개변수와 로컬 변수는 final 특성을 가지며,
로컬 클래스의 필드로

중첩 클래스의 접근 제한

❖ 중첩 클래스에서 바깥 클래스 참조 얻기

```
public class Outer {  
    String field = "Outer-field";  
    void method() {  
        System.out.println("Outer-method");  
    }  
  
    class Nested {  
        String field = "Nested-field";  
        void method() {  
            System.out.println("Nested-method");  
        }  
        void print() {  
            System.out.println(this.field);  
            this.method();  
            System.out.println(Outer.this.field);  
            Outer.this.method();  
        }  
    }  
}
```

중첩 객체 참조

바깥 객체 참조

중첩 클래스의 접근 제한

❖ 실행 클래스: OutterExample.java

```
public class OutterExample {  
    public static void main(String[] args) {  
        Outter outter = new Outter();  
        Outter.Nested nested = outter.new Nested();  
        nested.print();  
    }  
}
```