9. 중첩 클래스(Nested Class)와 중첩 인터페이스(NestedInterface)

중첩 클래스(Nested class)

- 클래스 내부에 선언한 클래스로서 내부 클래스(Inner class)라고도 함

  - 외부에 노출이 불필요한 클래스를 감춤으로써 코드의 복잡성을 줄이는 목적

- 중첩 클래스는 클래스의 멤버로서 선언되는 **멤버 클래스**와 생성자 또는 메소드 내부에 선언되는 **로컬 클래스**로 구분

  - static으로 선언한 정적 멤버 클래스는 주(主) 클래스의 인스턴스 생성 없이도 접근 가능

  - 인스턴스 멤버 클래스는 주(主) 클래스 인스턴스를 생성한 후 접근 가능

  - 로컬 클래스는 해당 메소드나 생성자 블록 안에서만 유효

    - 로컬 클래스는 접근 제한자를 붙일 수 없음

```
class A {
  class B { // 인스턴스 멤버 클래스
    int a;
    // static int b;  // 에러
    void method( ) { ..... }
    // static void method( ) { ..... } // 에러
  }
  void method( ) {
    B b = new B();  // 메소드 내부 사용
  }
  static void method2( ) {
    // B b = new B();  // 에러
  }
}
// 클래스 외부에서 사용
A a = new A( ); // A class 객체부터 생성
A.B b = a.new B( );
b.a = 3;
b.method( );
```

```
class A {
  static class C {// 정적 멤버 클래스
    int a;
    static int b;
    void method( ) {
      C c = new C( );
    }
    static void method2( ) {
      C c = new C( );
    }
}

// 클래스 외부에서 사용
A.C c = new A.C( );
c.a = 3;
c.method( );
A.C.b = 10;
A.C.method2( );
```

```
class A {
  void method( ) {
    class D {  // 로컬 클래스
     int a;
      // static int b;  // 에러
      void method( ) { ..... }
      // static void method( ) {..... } // 에러
    }
    D d = new D( ); // 메소드 내부 사용
    d.a = 5;
    d.method( );
  }
}
```

실습(1)

[Ex1-1] A.java

```java
package chap09.nested;

class A {
    int a;

    A() { System.out.println("A 객체가 생성됨"); }

    public class B { // 인스턴스 멤버 클래스
        B() { System.out.println("B 객체가 생성됨"); }
        int field1;
        //static int field2;  // 정적필드 사용 안됨
        void method1() {System.out.println(field1); }
        //static void method2() { } // 정적메소드 사용 안됨
    }

    static class C { // 정적 멤버 클래스
        C() { System.out.println("C 객체가 생성됨"); }
        int field1;
        static int field2;
        void method1() { }
        static void method2() { System.out.println(field2);}

    }

    void method() {
        class D { // 로컬 클래스
            D() { System.out.println("D 객체가 생성됨"); }
            int field1;
            //static int field2; // 정적필드 사용 안됨
            void method1() { }
            //static void method2() { } // 정적메소드 사용 안됨
        }
        D d = new D();
        d.field1 = 3;
        d.method1();
    }
}
```

[Ex1-2] Main.java

```java
package chap09.nested;

public class Main {
    public static void main(String[] args) {
        A a = new A();

        //인스턴스 멤버 클래스 객체 생성
        A.B b = a.new B();
        b.field1 = 3;
        b.method1();

        //정적 멤버 클래스 객체 생성
        A.C c = new A.C();
        c.field1 = 3;
        c.method1();
        A.C.field2 = 3;
        A.C.method2();
        c.field2 = 10;
        c.method2();

        //로컬 클래스 객체 생성을 위한 메소드 호출
        a.method();
    }
}
```

실습(2)

[Ex2] A2.java

```java
package chap09.nested;

public class A2 {
    //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(); // B 클래스는 인스턴스 생성이 필요하므로 static 부적절
    static C field4 = new C();

    //정적 메소드
    static void method2() {
        //B var1 = new B(); // static 메소드 안에서 인스턴스 클래스나 변수 참조 못함
        C var2 = new C();
    }

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

실습(3)

[Ex3] A3.java

```
package chap09.nested;

public class A3 {
    int field1;
    void method1() { }

    static int field2;
    static void method2() { }

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

            field2 = 10;
            method2();
        }
    }

    static class C {
        void method() {
            //field1 = 10; // static 클래스에서 인스턴스 변수 사용 못함
            //method1();  // static 클래스에서 인스턴스 메소드 사용 못함

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

실습(4)

[Ex4] A4.java

```
package chap09.nested;

public class A4 {
    //자바7 이전
    public void method1(final int arg) {
        final int localVariable = 1; // 자바7 이전엔 무조건 final 속성 써야 함
        //arg = 100; (x)
        //localVariable = 100; (x)
        class Inner {
            public void method() {
                int result = arg + localVariable;
            }
        }
    }
    // 자바8 이후
    public void method2(int arg) {
        int localVariable = 1; // 자바8 이후로 자동으로 final 속성 부여
        //arg = 100; (x)
        //localVariable = 100; // 값을 부여하면 localVariable에 자동 부여된 final 속성 풀림
                    // final 속성이 풀리면 아래의 result2 = arg + localVariable 에러남

        final int localVariable2 = 10;
        class Inner {
            public void method() {
                int result = arg + localVariable2;
                int result2 = arg + localVariable; // 위에서 localVariable의 final 속성이 풀리면
                                                   // Inner 클래스 안에서 localVariable을 사용할 수 없음
            }
        }


    }
}
```

실습(5)

[Ex5] OutterExample.java

```java
package chap09.nested;

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

class Outter {
    String field = "Outter-field";
    void method() {
        System.out.println("Outter-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(Outter.this.field);
            Outter.this.method();
        }
    }
}
```

중첩 인터페이스(Nested interface)

- 인터페이스도 중첩 클래스와 같이 클래스 내부에 선언할 수 있음
  - 오직 해당 클래스와 밀접한 관계를 가진 인터페이스이기에 외부 노출 불필요
  - 중첩 클래스와 마찬가지로 인스턴스 멤버 인터페이스와 정적 멤버 인터페이스 모두 가능
  - UI 구현 시, 이벤트 처리하는 로직에서 주로 사용

```
class A {
    interface MyInterface {
        void method( );
    }

    static interface MyInterface2 {
        void method( );
    }
}

class B implements A.MyInterface {
    void method( ){
        ……..
    }
}
```

실습(6)

[Ex6-1] Button.java

```java
package chap09.nested;

public class Button {
    OnClickListener listener;

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

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

    static interface OnClickListener { // static 생략 가능
        void onClick();
    }
}
```

[Ex6-2] CallListener.java

```java
package chap09.nested;

public class CallListener implements Button.OnClickListener {
    @Override
    public void onClick() {
        System.out.println("전화를 겁니다.");
    }
}
```

[Ex6-3] MessageListener.java

```java
package chap09.nested;

public class MessageListener implements Button.OnClickListener {
    @Override
    public void onClick() {
        System.out.println("메시지를 보냅니다.");
    }
}
```

[Ex6-4] ButtonExample.java

```java
package chap09.nested;

public class ButtonExample {
    public static void main(String[] args) {
        Button btn = new Button();

        btn.setOnClickListener(new CallListener());
        btn.touch();

        btn.setOnClickListener(new MessageListener());
        btn.touch();
    }
}
```

익명 클래스(Anonymous class)

- 이름이 없는 클래스로서 클래스 선언과 객체의 생성이 동시에 이루어지는 클래스
  - 오직 하나의 객체만을 생성할 수 있는 일회용 클래스
  - 어떤 클래스를 상속하거나 인터페이스를 구현하는 경우에만 사용 가능
  - 생성자를 가질 수 없으며, 둘 이상의 인터페이스를 구현할 수 없음
  - UI 구현 시, 이벤트 처리하는 로직에서 주로 사용

```
class A {
    int a;
    void method( ) { }
    void method1( ) { }
}

class B {
    A a = new A( ){ // 익명 클래스는 A를 자동적으로 상속함
        int a1;
        void method1( ) { }  // class A의 method1을 오버라이딩
    }
}
```

실습(7)

[Ex7-1] Person.java

```java
package chap09.nested;

public class Person {
    void wake() {
        System.out.println("7시에 일어납니다.");
    }
}
```

[Ex7-3] AnonymousExample.java

```java
package chap09.nested;

public class AnonymousExample {
    public static void main(String[] args) {
        Anonymous anony = new Anonymous();
        //익명 객체 필드 사용
        anony.field.wake();
        //익명 객체 로컬변수 사용
        anony.method1();
        //익명 객체 매개값 사용
        anony.method2(
            new Person() {
                void study() {
                    System.out.println("공부합니다.");
                }
                @Override
                void wake() {
                    System.out.println("8시에 일어납니다.");
                    study();
                }
            }
        );
    }
}
```

[Ex7-2] Anonymous.java

```java
package chap09.nested;

public class Anonymous {
    //필드 초기값으로 대입
    Person field = new Person() {
        void work() {
            System.out.println("출근합니다.");
        }
        @Override
        void wake() {
            System.out.println("6시에 일어납니다.");
            work();
        }
    };

    void method1() {
        //로컬변수값으로 대입
        Person localVar = new Person() {
            void walk() {
                System.out.println("산책합니다.");
            }
            @Override
            void wake() {
                System.out.println("7시에 일어납니다.");
                walk();
            }
        };
        //로컬변수 사용
        localVar.wake();
    }

    void method2(Person person) {
        person.wake();
    }
}
```

실습(8)

[Ex8-1] RemoteControl.java

```
package chap09.anonymous;

public interface RemoteControl {
    public void turnOn();
    public void turnOff();
}
```

[Ex8-3] AnonymousExample.java

```
package chap09.anonymous;

public class AnonymousExample {
    public static void main(String[] args) {
        Anonymous anony = new Anonymous();
        //익명 객체 필드 사용
        anony.field.turnOn();
        //익명 객체 로컬변수 사용
        anony.method1();
        //익명 객체 매개값 사용
        anony.method2(
            new RemoteControl() {
                @Override
                public void turnOn() {
                    System.out.println("SmartTV를 켭니다.");
                }
                @Override
                public void turnOff() {
                    System.out.println("SmartTV를 끕니다.");
                }
            }
        );
    }
}
```

[Ex8-2] Anonymous.java

```
package chap09.anonymous;

public class Anonymous {
    //필드 초기값으로 대입
    RemoteControl field = new RemoteControl() {
        @Override
        public void turnOn() {
            System.out.println("TV를 켭니다.");
        }
        @Override
        public void turnOff() {
            System.out.println("TV를 끕니다.");
        }
    };

    void method1() {
        //로컬변수값으로 대입
        RemoteControl localVar = new RemoteControl() {
            @Override
            public void turnOn() {
                System.out.println("Audio를 켭니다.");
            }
            @Override
            public void turnOff() {
                System.out.println("Audio를 끕니다.");
            }
        };
        //로컬변수 사용
        localVar.turnOn();
    }

    void method2(RemoteControl rc) {
        rc.turnOn();
    }
}
```

실습(9)

[Ex9-1] Button.java

```
package chap09.event;

public class Button {
    OnClickListener listener;

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

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

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

[Ex9-2] Window.java

```
package chap09.event;

public class Window {
    Button button1 = new Button();
    Button button2 = new Button();

    //필드 초기값으로 대입
    Button.OnClickListener listener = new Button.OnClickListener() {
        @Override
        public void onClick() {
            System.out.println("전화를 겁니다.");
        }
    };

    Window() {
        button1.setOnClickListener(listener);

        //매개값으로 대입
        button2.setOnClickListener(new Button.OnClickListener() {
            @Override
            public void onClick() {
                System.out.println("메시지를 보냅니다.");
            }
        });
    }
}
```

[Ex9-3] Main.java

```
package chap09.event;

public class Main {
    public static void main(String[] args) {
        Window w = new Window();
        w.button1.touch();
        w.button2.touch();
    }
}
```

실습(10)

[Ex10-1] Calculatable.java

```java
package chap09.anonymous2;

public interface Calculatable {
    public int sum();
}
```

[Ex10-3] AnonymousExample.java

```java
package chap09.anonymous2;

public class AnonymousExample {
    public static void main(String[] args) {
        Anonymous anony = new Anonymous();
        anony.method(0, 0);
    }
}
```

[Ex10-2] Anonymous.java

```java
package chap09.anonymous2;

public class Anonymous {
    private int field;

    public void method(final int arg1, int arg2) {
        final int var1 = 0;
        int var2 = 0;

        field = 10;

        //arg1 = 20; (x)
        //arg2 = 20; (x)

        //var1 = 30; (x)
        //var2 = 30; (x)

        Calculatable calc = new Calculatable() {
            @Override
            public int sum() {
                int result = field + arg1 + arg2 + var1 + var2;
                return result;
            }
        };

        System.out.println(calc.sum());
    }
}
```