JAC444 - Lecture 2

Inner and Anonymous Classes

Segment 3

Special Classes

In this segment you will be learning about:

- Enum Type
- Nested Class
- Inner Class
- Anonymous Class

Enum Types

An enum type is a special data type that enables for a variable to be a set of predefined constants.

```
public enum Cardinals {
  EAST,
  WEST,
  NORTH,
  SOUTH
}
```

The names of an enum type's fields are in uppercase letters, since they are constants.

Enum - Set of Constants

```
public class Test {

    public static void main(String[] args) {

        Cardinals direction = Cardinals.EAST;

        switch (direction) {
            case SOUTH:
                System.out.println("You should go to South");
                break;
            case NORTH:
                System.out.println("You should go to North");
                break;
            default:
                throw new AssertionError("Unknown directions");
        }
    }
}
```

Nested - Inner Class

- A class that is defined inside another class is called *nested class*

```
class Outer {
    ...
    class Nested {
        ...
    }
}
```

- As fields/methods, a nested class could be static, private, public.

- Non-static nested classes are called inner classes

Inner Class

```
class OuterClass extends X implements I1, I2 {
    // field(s), constructor(s)
    // method declarations

    class InnerClass extends Y implements J1 {
        // field(s), constructor(s)
        // method declarations
    }
}
```

- A declaration of a type in an inner class shadows any other declarations in the enclosing scope that have the same name

Inner Class Fields/Methods

```
class X {
        private int i = 10;
        public void render() {
            System.out.println("in outer: " + i + " " + this.new Y().i);
        }

        public class Y {
            private int i = 20;
            public void render() {
                System.out.println("in inner: " + X.this.i + " " + i);
            }
        }
}
```

Use Inner Class Example

```
public class TestInner {
        public static void main(String[] arg) {

            X outer = new X();
            outer.render();

            X.Y inner = outer.new Y();
            outer.render();

            new X().new Y().render();
        }
}
```

Anonymous Class

- A class without a name is called *anonymous* class

One can declare and instantiate a class at the same time

The anonymous class expression:

- The new operator
- The name of an interface or a class to extend
- Parentheses that contain the arguments to a constructor /empty pair of parentheses for interface
- A class declaration body

Example Anonymous Class

```
interface Sayable {
        public void say();
}
class TestAnonymousClass {
        public static void main(String[] args) {
            //anonymous class
            Sayable s = new Sayable() {
                @Override
                public void say() {
                    System.out.println("From an anonymous class");
                }
            };
            s.say();
        }
}
```