03-b. data 클래스

Object Oriented Programming

데이터 클래스

데이터 클래스란?

- 데이터 여러 개를 클래스로 묶어서 이용하려는 목적으로 VO 클래스를 만듭니다.
- 코틀린에서는 이런 클래스들을 조금 더 편하게 이용하라고 데이터 클래스를 제공합니다.
- 데이터 클래스는 data라는 예약어로 선언하는 클래스입니다.

```
data class User(val name: String, val age: Int)
```

데이터 클래스

데이터 클래스란?

- 주생성자가 선언되어야 하며 주생성자의 매개변수는 최소 하나 이상이 선언되어 있어야 한다.
- 모든 주생성자의 매개변수는 var 혹은 val 로 선언되어야 한다.
- 데이터 클래스는 abstract, open, sealed, inner 등의 예약어를 추가할수 없다.

```
data class User1()//error

data class User2(name: String)//error

data abstract class User3(val name: String)//error

data class User4(val name: String, no: Int)//error
```

데이터 클래스

데이터 클래스의 함수

- 데이터 클래스가 유용한 것은 데이터 클래스에 선언한 데이터와 관련된 다양한 기능을 함수로 제공하
- 기 때문입니다.
- 데이터 클래스는 개발자가 클래스 내에 선언하지 않아도 자동으로 다음의 함수를 제공 합니다.

- equals() / hashCode()
- toString()
- componentN()
- copy()

데이터 클래스

equals()

- 객체의 데이터가 같은지에 대한 비교

```
class Product(val name: String, val price:Int)

data class User(val name: String, val age: Int)

fun main(args: Array<String>) {

    var product=Product("prod1",100)
    var product1=Product("prod1",100)
    println(product.equals(product1))//false

    var user=User("kkang",30)
    var user1=User("kkang",30)
    println(user.equals(user1))//true
}
```

데이터 클래스

equals()

- equals 함수에 의한 값의 비교는 주생성자에 선언된 프로퍼티 값만을 비교

```
data class User(val name: String, val age: Int){
    var email: String = "a@a.com"
}


fun main(args: Array<String>) {
    val user = User("kkang", 20)

    val user1 = User("kkang", 20)
    user1.email = "b@b.com"

    println(user.equals(user1))
}
```

데이터 클래스

toString()

- 데이터 클래스의 데이터를 문자열로 반환하는 함수

```
class Product(val name: String, val price:Int)

data class User(val name: String, val age: Int){
    var email: String = "a@a.com"
}

fun main(args: Array<String>) {

    var product=Product("prod1",100)
    println(product.toString())

    var user=User("kkang",30)
    println(user.toString())
}
```

실행결과

```
fourteen_four.Product@5e2de80c
User(name=kkang, age=30)
```

데이터 클래스

componentN()

- 클래스 프로퍼티 값을 획득

```
data class User(val name: String, val age: Int)

fun main(args: Array<String>) {

    var user=User("kkang",30)

    println(user.component1()) //kkang
    println(user.component2()) //30
}
```

```
data class User(val name: String, val age: Int)

fun main(args: Array<String>) {
   var user=User(age=30, name="kkang")
    val (name, age) = user

    println("name : $name, age: $age") //name : kkang, age: 30
}
```

데이터 클래스

copy()

- 객체를 복사

```
data class User(val name: String, val age: Int)

fun main(args: Array<String>) {
    var user=User(age=30, name="kkang")
    println(user.toString()) //User(name=kkang, age=30)

    var user2=user.copy(name="kim")
    println(user2.toString()) //User(name=kim, age=30)
}
```

데이터 클래스

상속

- data 클래스를 만들 때 다른 클래스를 상속받아 작성이 가능하다.
- 하지만 data 클래스를 상속받아 다른 클래스를 선언할 수 없다. data 클래스는 내부적으로 final 로 선언된다.
- 명시적으로 open 을 추가할 수 없다.

감사합니다

단단히 마음먹고 떠난 사람은
산꼭대기에 도착할 수 있다.
산은 올라가는 사람에게만 정복된다.

윌리엄 셰익스피어
William Shakespeare