

5COSC023W - MOBILE APPLICATION DEVELOPMENT

Lecture 4: More on Kotlin - Android Shared Preferences

Dr Dimitris C. Dracopoulos

Classes

```
class Employee (colour: String, n: String) {  
    val eyeColour: String = colour  
    var age: Int = 25  
    val name: String = n  
  
    override fun toString(): String {  
        return "name: $name, eyeColour: $eyeColour, age: $age"  
    }  
}  
  
fun main() {  
    val e1 = Employee("green", "John")  
    println(e1)  
}
```

Creating Class Properties Automatically

- ▶ Use `var` or `val` when you declare the parameters of the constructor:

```
class Employee (val eyeColour: String,
                var age: Int,
                var name: String) {
    override fun toString(): String {
        return "name: $name, eyeColour: $eyeColour, age: $age"
    }
}

fun main() {
    val e2 = Employee("brown", 18, "Helen")
    println(e2)
}
```

Variable Number of Arguments

- ▶ Use the `vararg` keyword.
- ▶ The `vararg` parameter becomes an `Array`.
- ▶ A function definition can only specify one parameter as `vararg`.
- ▶ Try to choose the last parameter of a function to be the `vararg`.

```
fun foo(date: String, vararg names: String) {  
    println("date: $date")  
    for (n in names)  
        println(n)  
}  
  
fun main() {  
    foo("26th of February", "James", "Helen", "Joe", "Alice")  
}
```

Maps

```
fun main() {  
    var capitals = mapOf("Netherlands" to "Amsterdam",  
                        "Hungary" to "Budapest",  
                        "Finland" to "Helsinki")  
  
    println(capitals["Hungary"])  
    println(capitals.getValue("Finland"))  
  
    for ((key, value) in capitals)  
        println("$key -> $value")  
  
    for (entry in capitals)  
        println(entry.key + ":: " + entry.value)  
}
```

Sets

Cannot contain duplicate elements.

```
fun main() {  
    var cities = mutableSetOf("London", "Paris",  
                              "Berlin", "London",  
                              "Paris")  
  
    for (c in cities)  
        print(c+ " ")  
    println()  
  
    cities += "Warsaw"  
    cities -= "Paris"  
  
    print("Updated set contains: ")  
    for (c in cities)  
        print(c + " ")  
  
}
```

The usual mathematical set operations (union, intersection, difference and others) are also available.

Nullable References - An Attempt to fix Tony Hoare's "Billion Dollar Mistake"

- ▶ By default, references cannot receive the value of `null`.

```
var s: String = null // Compiler error!
```

- ▶ A question mark `?` needs to be appended to make a variable nullable:

```
var s: String? = null // OK
```

- ▶ A nullable type cannot be dereferenced:

```
var s2: String? = "abc"  
s2.length // Compiler error!
```

- ▶ Use the safe call `?.` to attempt to dereference a nullable value:

```
var s2: String? = "abc"  
s2?.length // Will give back a value of null if s2 is null
```

- ▶ Alternatively, use the non-null assertion operator `!!`

```
var s3: String? = "abc"  
s3!! // if null throws a NullPointerException
```

Comparing Variables

- ▶ Use `==` (or `equals`) for structural comparison
- ▶ Use `===` to check if 2 references point to the same object

Saving Data in an Android Application

- ▶ Use `onSaveInstanceState()` for configuration changes or system destroying and re-creating the activity.
- ▶ Saving Key-Value Sets (small amounts)
- ▶ Saving in Files
- ▶ Saving in SQL databases (large amounts of structured data)

SharedPreferences (Saving Key-Value Sets)

To create a new shared preference file or access an existing one, call one of the following methods to get a SharedPreferences object:

- ▶ `getSharedPreferences()`: if you need multiple shared preferences files (the name of the preference file is the first argument) - can be called from any Context in the app

```
sharedPref: SharedPreferences =  
    getSharedPreferences("preference_filename",  
                        Context.MODE_PRIVATE);
```

- ▶ `getPreferences()`: call from an activity to use only one shared preference file associated with the activity

```
sharedPref = getPreferences(Context.MODE_PRIVATE);
```

Usage of `MODE_WORLD_READABLE` or `MODE_WORLD_WRITEABLE` imply that any other app can access your data (if it knows the filename)

Saving Key-Value Sets (Writing to Shared Preferences)

1. Create a `SharedPreferences.Editor` by calling `edit()` on `SharedPreferences`.
2. Write the keys and values with `putInt()`, `putString()`, etc.
3. Call `apply()` or `commit()`.

```
sharedPref: SharedPreferences = getActivity().getPreferences(  
    Context.MODE_PRIVATE);  
editor: SharedPreferences.Editor = sharedPref.edit(); // step 1  
editor.putInt("key_name", newHighScore); // step 2  
editor.apply(); // step 3
```