

Mobile Application Development

Produced
by

David Drohan (ddrohan@wit.ie)

Department of Computing & Mathematics
Waterford Institute of Technology
<http://www.wit.ie>

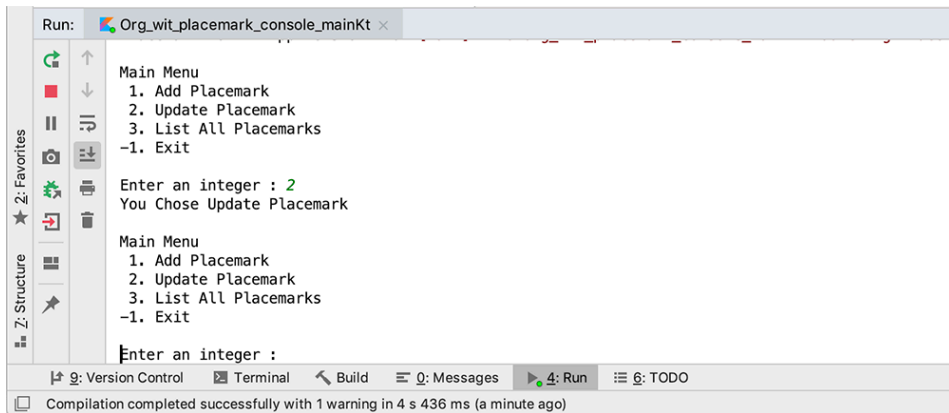


Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE



Placemark-Console

Version 1.0



The screenshot shows an IDE console window with the following content:

```
Run: Org_wit_placemark_console_mainKt x
Main Menu
1. Add Placemark
2. Update Placemark
3. List All Placements
-1. Exit
Enter an integer : 2
You Chose Update Placemark
Main Menu
1. Add Placemark
2. Update Placemark
3. List All Placements
-1. Exit
Enter an integer :
```

The IDE interface includes a sidebar with 'Favorites' and 'Structure' tabs, and a bottom status bar indicating 'Compilation completed successfully with 1 warning in 4 s 436 ms (a minute ago)'.



Features Covered (from Part 1)

- ☐ Basic Types
- ☐ Local Variables (`val` & `var`)
- ☐ Functions
- ☐ Control Flow (`if`, `when`, `for`, `while`)
- ☐ Strings & String Templates
- ☐ Ranges (and the *`in`* operator)
- ☐ Type Checks & Casts
- ☐ Null Safety
- ☐ Comments



Features Covered (from Part 2)

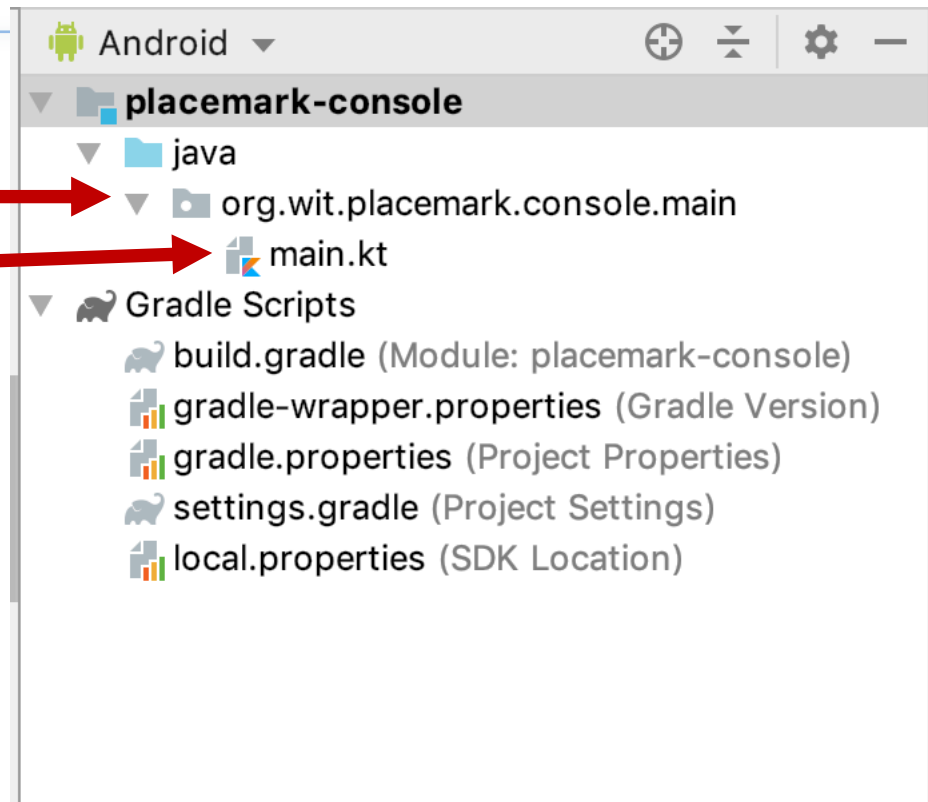
- ❑ Writing Classes (properties and fields)
- ❑ Data Classes (just for data)
- ❑ Collections: Arrays and Collections
- ❑ Collections: *in* operator and lambdas
- ❑ Arguments (default and named)



Project Structure

□ Fairly basic at the moment

- 1 package
- 1 Kotlin source FILE



main.kt

❑ Fairly basic at the moment

- 1 package
- 1 Kotlin source FILE
- ◆ Limited Features

```
main.kt x
1 package org.wit.placemark.console.main
2
3 import mu.KotlinLogging
4
5 private val logger = KotlinLogging.logger {}
6
7 fun main(args: Array<String>) {
8     logger.info { "Launching Placemark Console App" }
9     println("Placemark Kotlin App Version 1.0")
10
11     var input: Int
12
13     do {...} while (input != -1)
14     logger.info { "Shutting Down Placemark Console App" }
15 }
16
17 fun menu() : Int {...}
18
19 fun addPlacemark(){...}
20
21 fun updatePlacemark() {...}
22
23 fun listPlacemarks() {...}
```

Basic Types & Variables

Placemark-Console Version 1.0

Basic Types in Placemark

```
private val logger = KotlinLogging.logger {}  
  
fun main(args: Array<String>) {  
    logger.info { "Launching Placemark Console App" }  
    println("Placemark Kotlin App Version 1.0")  
  
    var input: Int  
  
    do {...} while (input != -1)  
    logger.info { "Shutting Down Placemark Console App" }  
}
```

- ❑ **var Int** declaration called **input**
 - *can be changed*

Basic Types in Placemark

```
private val logger = KotlinLogging.logger {}  
  
fun main(args: Array<String>) {  
    logger.info { "Launching Placemark Console App" }  
    println("Placemark Kotlin App Version 1.0")  
  
    var input: Int  
  
    do {...} while (input != -1)  
    logger.info { "Shutting Down Placemark Console App" }  
}
```


- ❑ **val** declaration called **logger** with type inferred
 - cannot be changed after assignment

Functions & Control Flow

Placemark-Console Version 1.0

Functions in Placemark

```
private val logger = KotlinLogging.logger {}  
  
fun main(args: Array<String>) {  
    logger.info { "Launching Placemark Console App" }  
    println("Placemark Kotlin App Version 1.0")  
  
    var input: Int  
  
    do {...} while (input != -1)  
    logger.info { "Shutting Down Placemark Console App" }  
}
```



❑ **fun main**, the 'main' function in our application

- Takes a single parameter **args**, of type **String Array** (not actually used)

Functions in Placemark

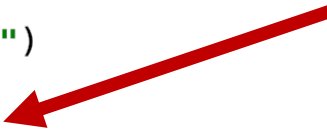
```
fun menu() : Int {  
    var option : Int  
    var input: String? = null  
  
    println("MAIN MENU")  
    println(" 1. Add Placemark")  
    println(" 2. Update Placemark")  
    println(" 3. List All Placemarks")  
    println("-1. Exit")  
    println()  
    print("Enter an integer : ")  
    input = readLine()!!  
    option = if (input.toIntOrNull() != null  
                && !input.isEmpty())  
        input.toInt()  
    else  
        -9  
    return option  
}
```

❑ **fun menu**, the 'menu' displayed to the user

- also declares 2 **var** variables to be used within the function

Functions in Placemark

```
fun addPlacemark(){  
    println("You Chose Add Placemark")  
}  
  
fun updatePlacemark() {  
    println("You Chose Update Placemark")  
}  
  
fun listPlacemarks() {  
    println("You Chose List All Placemarks")  
}
```

- 
- ❑ the functions called based on the users choice
 - very basic at the moment, we will refactor these functions in future versions of Placemark.

Control Flow in Placemark

```
var input: Int

do {
    input = menu()
    when(input) {
        1 -> addPlacemark()
        2 -> updatePlacemark()
        3 -> listPlacemarks()
        -1 -> println("Exiting App")
        else -> println("Invalid Option")
    }
    println()
} while (input != -1)
logger.info { "Shutting Down Placemark Console App" }
```

❑ **do-while** loop to call our functions based on user input

- Using **when** & lambdas to keep our code clean and concise.
- 'Exits' on -1

Strings & Null Safety

Placemark-Console Version 1.0

Strings & Null Safety in Placemark

```
fun menu() : Int {  
    var option : Int  
    var input: String? = null  
  
    println("MAIN MENU")  
    println(" 1. Add Placemark")  
    println(" 2. Update Placemark")  
    println(" 3. List All Placemarks")  
    println("-1. Exit")  
    println()  
    print("Enter an integer : ")  
    input = readLine()!!  
    option = if (input.toIntOrNull() != null  
                && !input.isEmpty())  
        input.toInt()  
    else  
        -9  
    return option  
}
```

□ **var** variable **input** is declared as a **String** and **nullable**

- We need the **?** to assign a **null** value
- Allows us to check for accidental empty values
- Some basic validation on **input**



References

Sources: <http://kotlinlang.org/docs/reference/basic-syntax.html>
<http://petersommerhoff.com/dev/kotlin/kotlin-for-java-devs/>
<https://www.programiz.com/kotlin-programming>
<https://medium.com/@napperley/kotlin-tutorial-5-basic-collections-3f114996692b>

