SSE3052: Embedded Systems Practice

Jinkyu jeong
jinkyu@skku.edu
Computer Systems Laboratory
Sungkyunkwan University
http://csl.skku.edu

Kotlin

- An open-source, statically-typed programming language that supports both object-oriented and functional programming
- Designed to interoperate fully with Java
- Mainly targets the JVM, but also compiles to Javascript or native code (via LLVM)
- Officially supported by Google for mobile development on Android

Why Kotlin?

- Concise
  - Drastically reduce the amount of boilerplate code
- Safe
  - Avoid entire classes of errors such as null pointer exceptions
- Interoperable
  - Leverage existing libraries for the JVM, Android, and the browser
- Tool-friendly
  - Choose any Java IDE or build from the command line

Basic Syntax – defining functions

- Function having two *int* parameters with *Int* return type

```
fun sum(a: Int, b: Int): Int {
    return a+b
}
```

- Function with an expression body and inferred return type

```
fun sum(a: Int, b:Int) = a + b
```

- Function returning no meaningful value:

```
fun printSum(a: Int, b: Int): Unit {
    println("sum of $a and $b is ${a+b}")
}
```

* Unit return type can be omitted

Basic Syntax – defining variables

- Read-only local variables are defined using the val keyword
  - They can be assigned a value only once

```
val a: Int = 1      // immediate assignment
val b = 2           // 'int' type is inferred
val c: Int          // Type required when no initializer is provided
c = 3               // deffered assignment
```

- Variables that can be assigned use the var keyword

```
var x = 5 // 'Int' type is inferred
x += 1
```

- Top-level variables:

```
val PI = 3.14
var x = 0

fun increment() {
        x += 1
}
```

Basic Syntax – string templates

- Using string templates

```
var a = 1
// simple name in template
val s1 = “a is $a”

a = 2
val s2 = “${s1.replace(“is”,“was”)}, but now is $a”
println(s2)
------
a was 1, but now is 2
```

Basic Syntax – if/else

```
val a = 20
if (a > 20) {
    println(“a is greater than 20”)
} else if (a < 20) {
    println(“a is smaller than 20”)
} else {
    println(“a is 20”)
}


var b = if (condition) trueval else falseval
```

Basic Syntax - when

```
fun describe(obj: Any): String =
    when (obj) {
        1              -> println("One")
        "Hello"        -> println("Greeting")
        is Long        -> println("Long")
        !is String     -> println("Not a String")
        else           -> println("Unknown")
    }
```

Basic Syntax – for loop

```
val items = listOf(“apple”,“banana”,“kiwifruit”)
for (item in items) {
    println(item)
}
```

OR

```
val items = listOf(“apple”,“banana”,“kiwifruit”)
for (index in items.indices) {
    println(“item at $index is ${items[index]}”)
}
```

Basic Syntax – for loop

Other features

```
for (i in 1..10) {}                  // 1 2 3 … 10
for (i in 1 until 10) {}             // 1 2 3 … 9
                                        (except 10)
for (i in 1..10 step 2) {}           // 1 3 5 7 9
for (i in 10 downTo 1) {}            // 10 9 8 … 1
for (i in 10 downTo 1 step 2) {}     // 10 8 6 4 2
```

Basic Syntax – Functions

Declaration (with fun keyword)

```
fun welcome(name: String, Job: string): String {
    return “Welcome $job $name to Kotlin!”
}
```

Calling

```
val new_member = welcome(“Semi”, 21)
```

Basic Syntax – Functions

Returning

- Default return type: Unit

Overloading

```
fun square(number: Int) = number * number
fun square(number: Double) = number * number
square(4)      // result is 16 (Int)
square(3.14)   // result is 9.8596 (Double)
```

Concise Features

- Create a POJO (Plain Old Java Object)
  - With getters, setter, equals(), hashCode(), toString() and copy() in a single line:

```
data class Customer(val name: String, val email: String, val company: String)
```

- Filter a list using a lambda expression

```
val positiveNumbers = list.filter { it > 0 }
```

- If you need a singleton, create an object

```
object ThisIsASingleton {
  val companyName: String = "JetBrains"
}
```

Safe Features

- Get rid of those pesky NullPointerExceptions

```
var output: String
output = null            //Compilation error
```

- Kotlin protects you from mistakenly operating on nullable types

```
val name: String? = null    //   Nullable type
println(name.length())      // Compilation error
println(name?.length())     // if (name != null)   println(name.length())
```

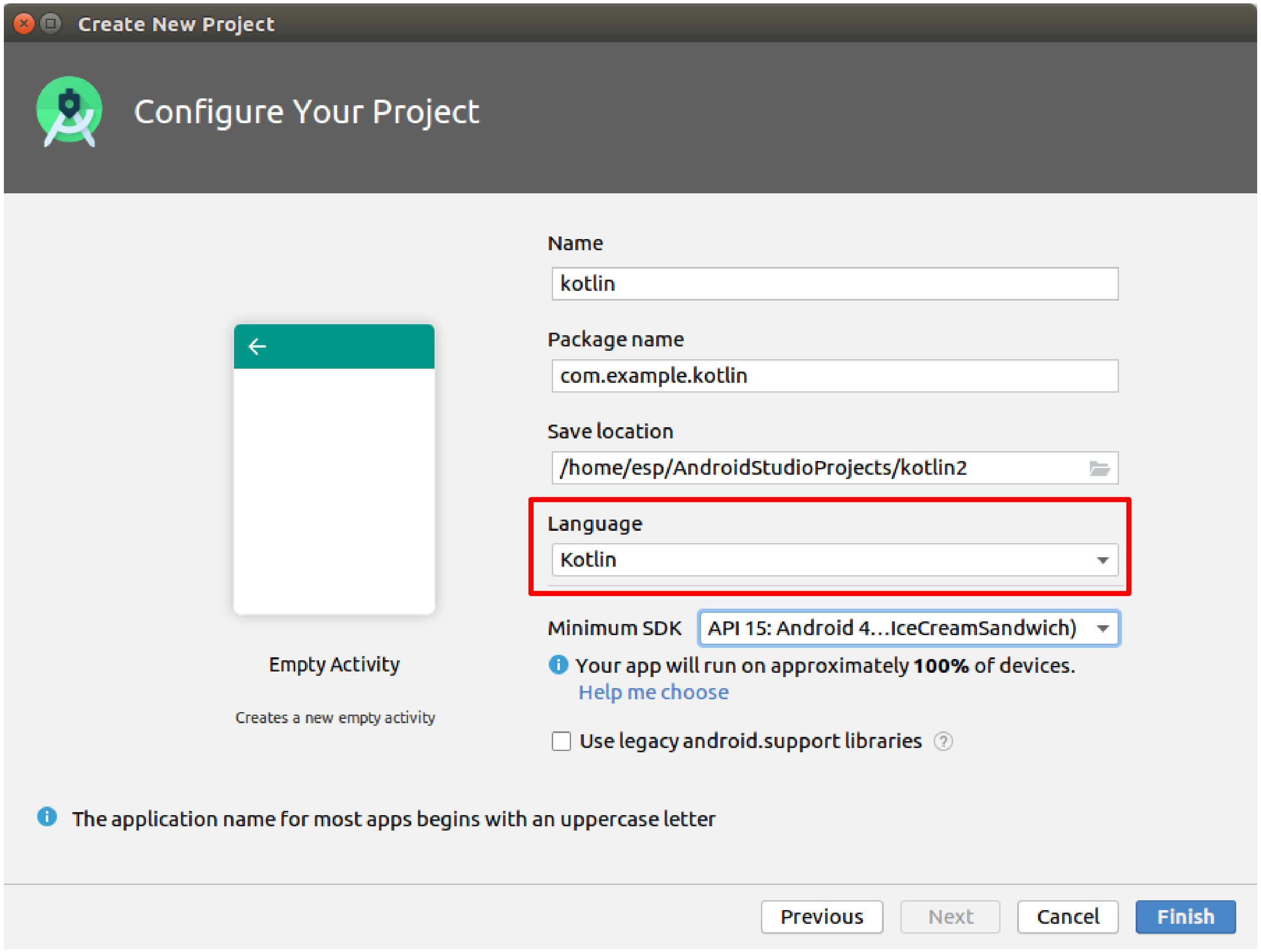
- Handling nullability with !!not-null assertion

```
val account = Account("name", "type")
val accountName = account.name!!.trim()
// if name is null, it throws NullPointerException
```

OR ?. Safe-call operator

```
val accountName = account.name?.trim()
// if name is null, name?.trim() returns null
```

New Project with Kotlin



Create New Project

Configure Your Project

Name: kotlin

Package name: com.example.kotlin

Save location: /home/esp/AndroidStudioProjects/kotlin2

Language: Kotlin

Minimum SDK: API 15: Android 4...IceCreamSandwich)

Your app will run on approximately **100%** of devices.
Help me choose

☐ Use legacy android.support libraries

Empty Activity

Creates a new empty activity

The application name for most apps begins with an uppercase letter

MainActivity

- MainActivity.java

```
public class MainActivity extends AppComaptActivity {
        @override
        protected void onCreate(Bundle savedInstanceState){
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
        }
}
```

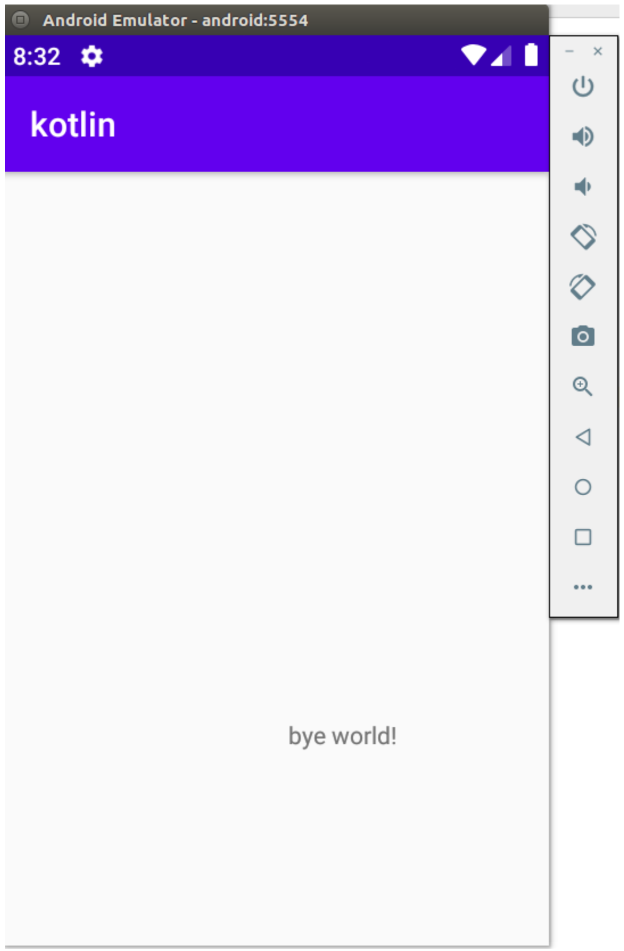
- MainActivity.kt

```
class MainActivity :AppCompatActivity() {
        override fun onCreate(savedInstanceState: Bundle?) {
                super.onCreate(savedInstanceState);
                setContentView(R.layout.activity_main);
        }
}
```

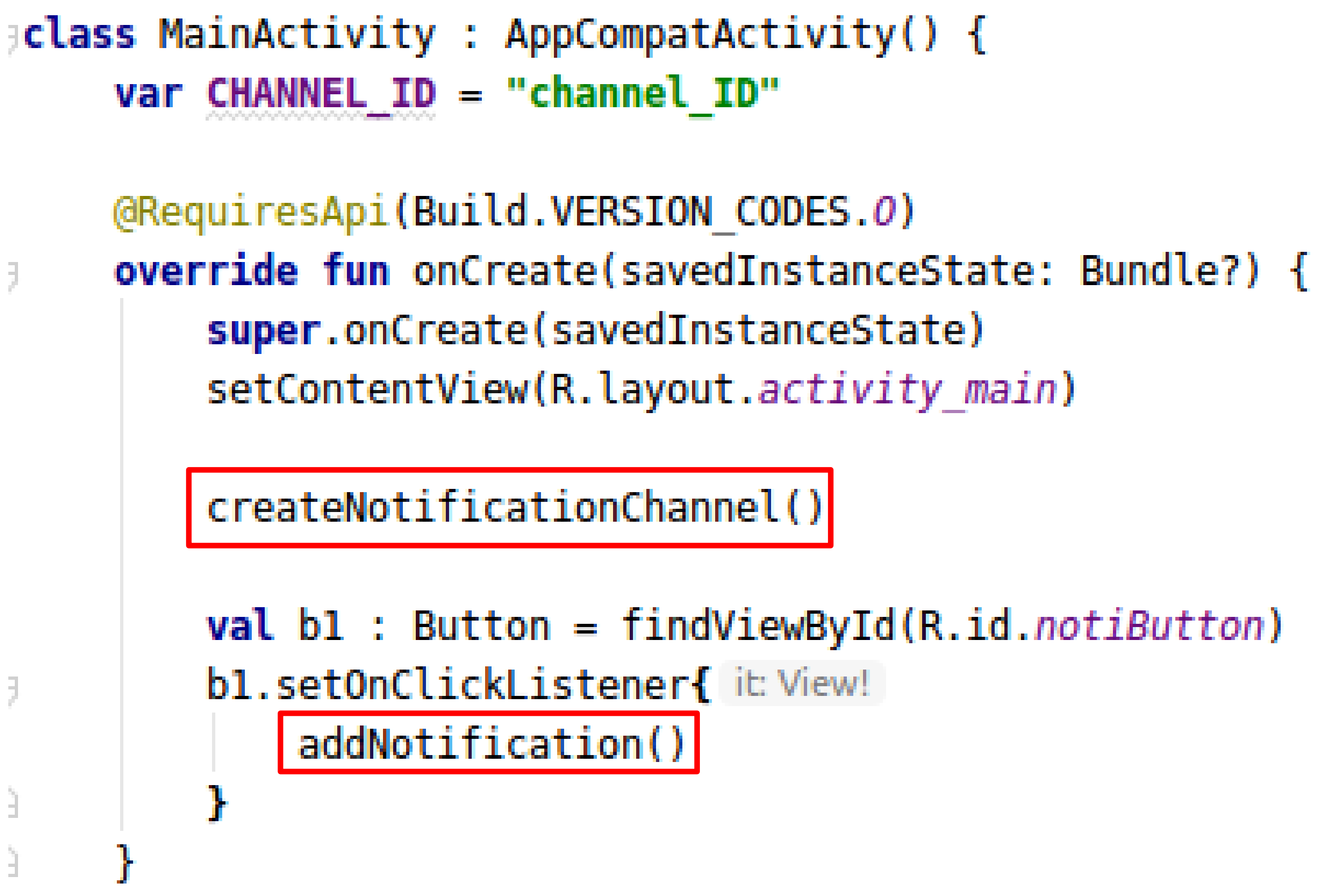
Add Views - TextView

- MainActivity.kt

```
class MainActivity : AppCompatActivity() {
    override fun onCreate (savedInstanceState: Bundle) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        val textView :TextView = findViewById(R.id.textView)
        textView.setText("Bye World!")
    }
}
```



Basic Notification w/ Kotlin (1)



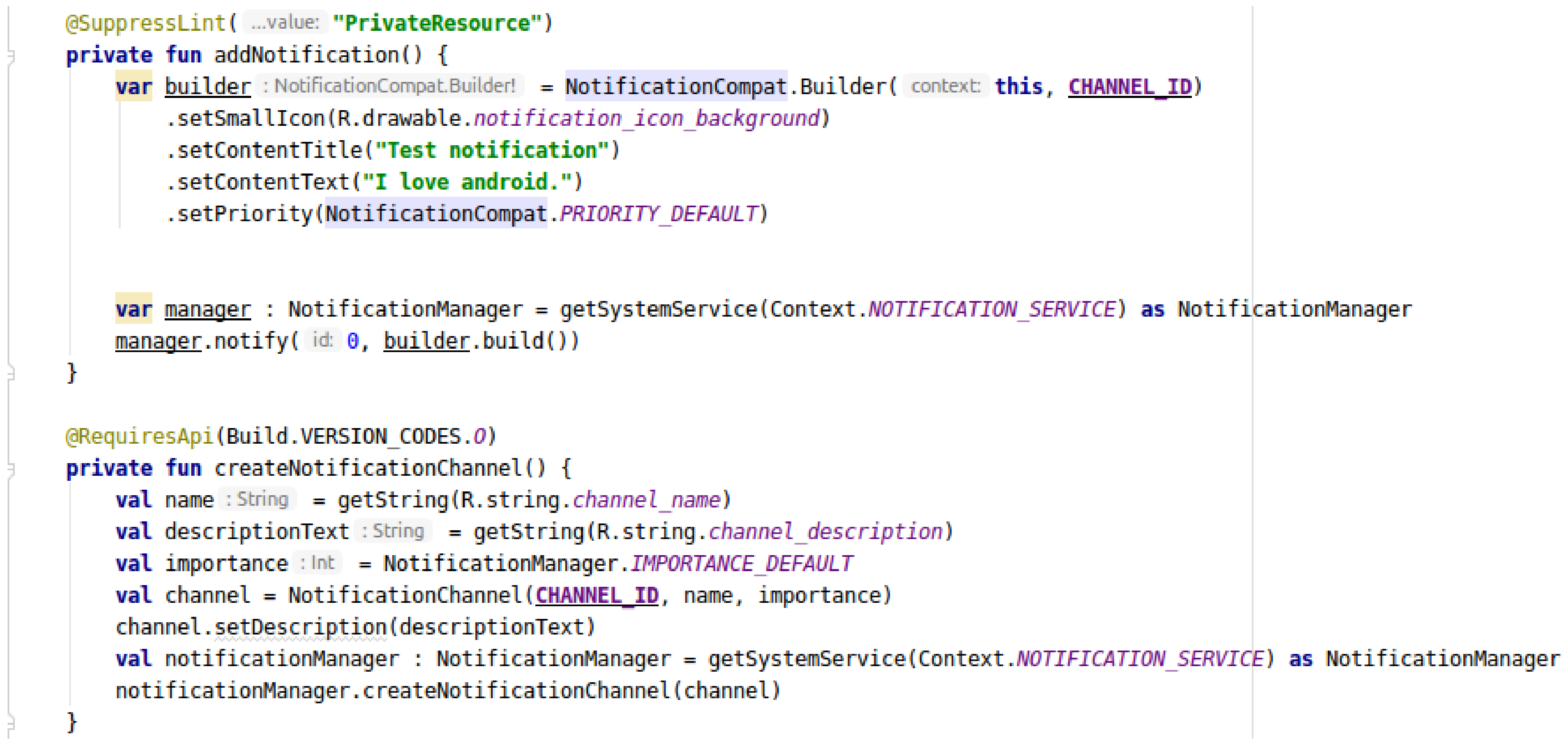
```kotlin
class MainActivity : AppCompatActivity() {
    var CHANNEL_ID = "channel_ID"

    @RequiresApi(Build.VERSION_CODES.O)
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        createNotificationChannel()

        val b1 : Button = findViewById(R.id.notiButton)
        b1.setOnClickListener{
            addNotification()
        }
    }
```

Basic Notification w/ Kotlin (2)



```
@SuppressLint( ...value: "PrivateResource")
private fun addNotification() {
    var builder : NotificationCompat.Builder!  = NotificationCompat.Builder( context: this, CHANNEL_ID)
        .setSmallIcon(R.drawable.notification_icon_background)
        .setContentTitle("Test notification")
        .setContentText("I love android.")
        .setPriority(NotificationCompat.PRIORITY_DEFAULT)


    var manager : NotificationManager = getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager
    manager.notify( id: 0, builder.build())
}

@RequiresApi(Build.VERSION_CODES.O)
private fun createNotificationChannel() {
    val name : String  = getString(R.string.channel_name)
    val descriptionText : String  = getString(R.string.channel_description)
    val importance : Int  = NotificationManager.IMPORTANCE_DEFAULT
    val channel = NotificationChannel(CHANNEL_ID, name, importance)
    channel.setDescription(descriptionText)
    val notificationManager : NotificationManager = getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager
    notificationManager.createNotificationChannel(channel)
}
```

References

- To learn Kotlin
  - https://kotlinlang.org/docs/reference/
- For Android Kotlin samples
  - https://developer.android.com/samples/index?language=kotlin
- Notification in Kotlin
  - https://developer.android.com/training/notify-user/build-notification#java

Exercise

- http://www.vogella.com/tutorials/Android/article.html#tutorialtemperature
  - Make temperature convertor in *Kotlin*
  - New notification for converting
    - contentTitle: @string/app_name
    - contentText: *xx* celcius is converted to *yy* fahrenheit