Mobile Programming

Kotlin Basics – Part I

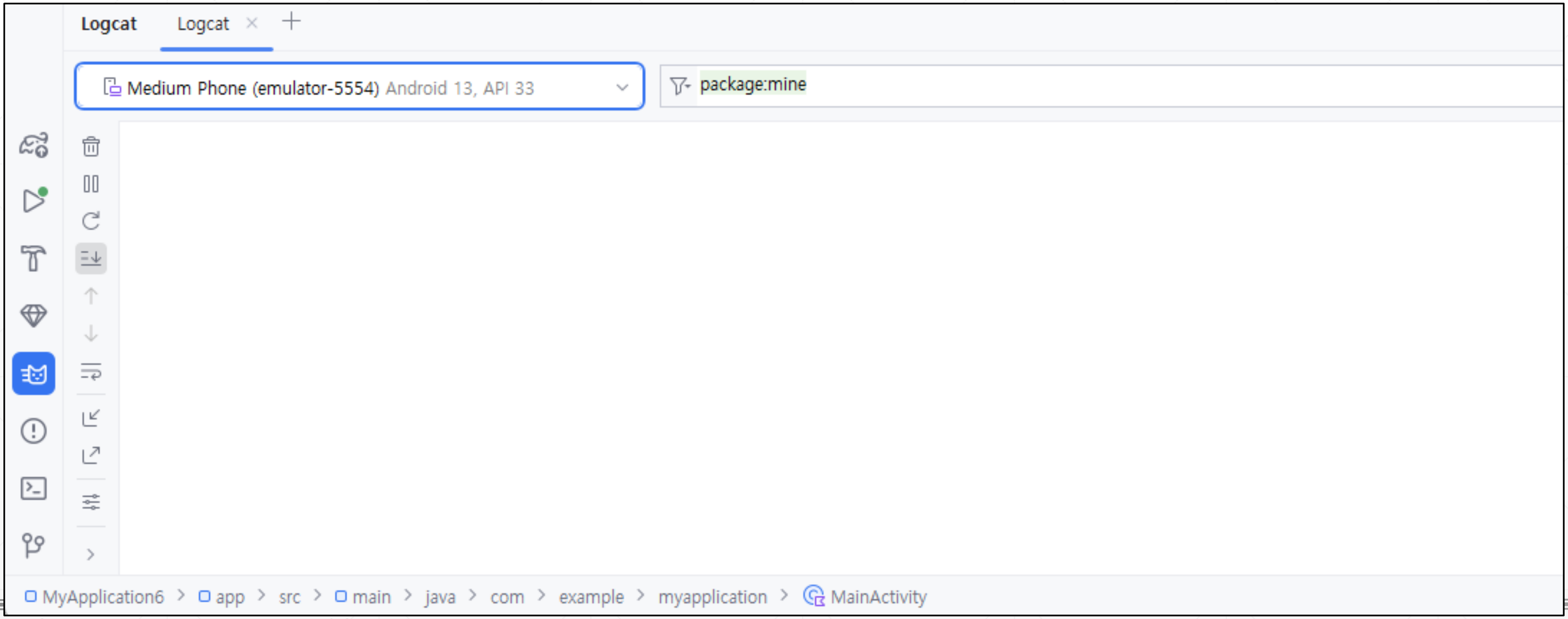
Let’s go! Kotlin!

- Kotlin
  - Default Programming Language for Android Development
  - API document: https://kotlinlang.org/docs/

Environment	Language	Toolkit
Application	Java/Kotlin	SDK (Software Development Kit)
System application	C, C++	NDK (Native Development Kit)
Hardware control / kernel	C, C++	PDK (Platform Development Kit)

Logcat (1/6)

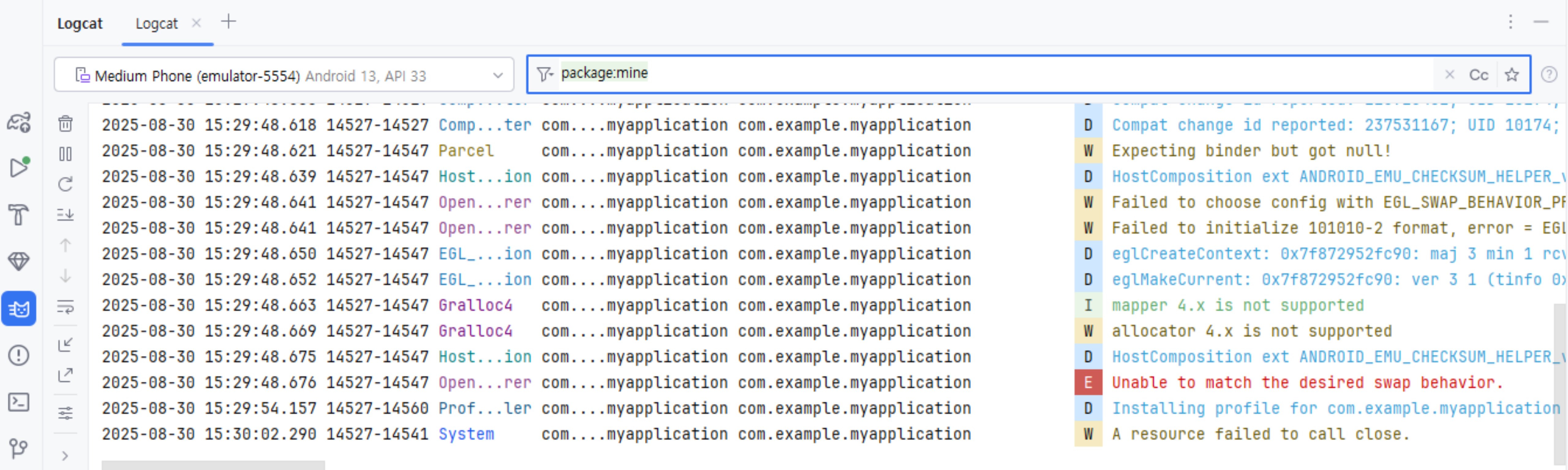
- Log monitor
  - System logs
  - Application logs
- We will check the result of Kotlin codes using Logcat!



Logcat
Logcat ×
Medium Phone (emulator-5554) Android 13, API 33
package:mine
MyApplication6 › app › src › main › java › com › example › myapplication › MainActivity

Logcat (2/6)

- Start your emulator or real device
- Then, see Logcat! What happens?



Medium Phone (emulator-5554) Android 13, API 33

package:mine

```
2025-08-30 15:29:48.618 14527-14527 Comp...ter com....myapplication com.example.myapplication  D  Compat change id reported: 237531167; UID 10174;
2025-08-30 15:29:48.621 14527-14547 Parcel     com....myapplication com.example.myapplication  W  Expecting binder but got null!
2025-08-30 15:29:48.639 14527-14547 Host...ion com....myapplication com.example.myapplication  D  HostComposition ext ANDROID_EMU_CHECKSUM_HELPER_
2025-08-30 15:29:48.641 14527-14547 Open...rer com....myapplication com.example.myapplication  W  Failed to choose config with EGL_SWAP_BEHAVIOR_P
2025-08-30 15:29:48.641 14527-14547 Open...rer com....myapplication com.example.myapplication  W  Failed to initialize 101010-2 format, error = EGL
2025-08-30 15:29:48.650 14527-14547 EGL_...ion com....myapplication com.example.myapplication  D  eglCreateContext: 0x7f872952fc90: maj 3 min 1 rc
2025-08-30 15:29:48.652 14527-14547 EGL_...ion com....myapplication com.example.myapplication  D  eglMakeCurrent: 0x7f872952fc90: ver 3 1 (tinfo 0x
2025-08-30 15:29:48.663 14527-14547 Gralloc4   com....myapplication com.example.myapplication  I  mapper 4.x is not supported
2025-08-30 15:29:48.669 14527-14547 Gralloc4   com....myapplication com.example.myapplication  W  allocator 4.x is not supported
2025-08-30 15:29:48.675 14527-14547 Host...ion com....myapplication com.example.myapplication  D  HostComposition ext ANDROID_EMU_CHECKSUM_HELPER_
2025-08-30 15:29:48.676 14527-14547 Open...rer com....myapplication com.example.myapplication  E  Unable to match the desired swap behavior.
2025-08-30 15:29:54.157 14527-14560 Prof...ler com....myapplication com.example.myapplication  D  Installing profile for com.example.myapplication
2025-08-30 15:30:02.290 14527-14541 System     com....myapplication com.example.myapplication  W  A resource failed to call close.
```

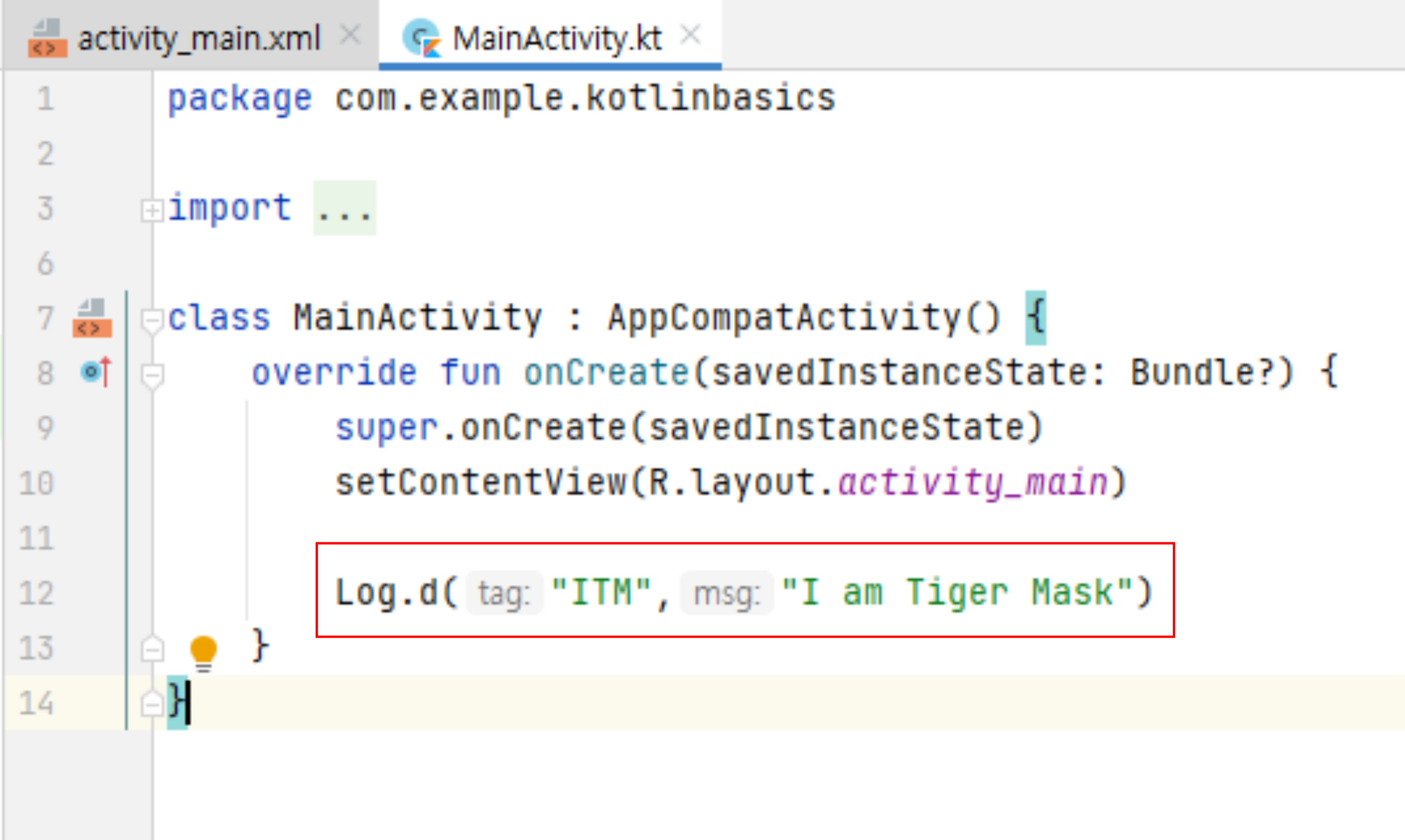
Logcat (3/6)

- Log : Information that is output outside the app to understand the flow of code
- Logcat : Log monitor which can be used to actually monitor logs with a variety of filter functions

Method	Meaning	Subject
Log.v( )	verbose	Output verbose data 상세 데이터 출력
Log.d( )	debug	Output logs for debug (for developers)
Log.i( )	information	Output logs for information
Log.w( )	warning	Output logs for warning
Log.e( )	error	Output logs for error!

Logcat (4/6)

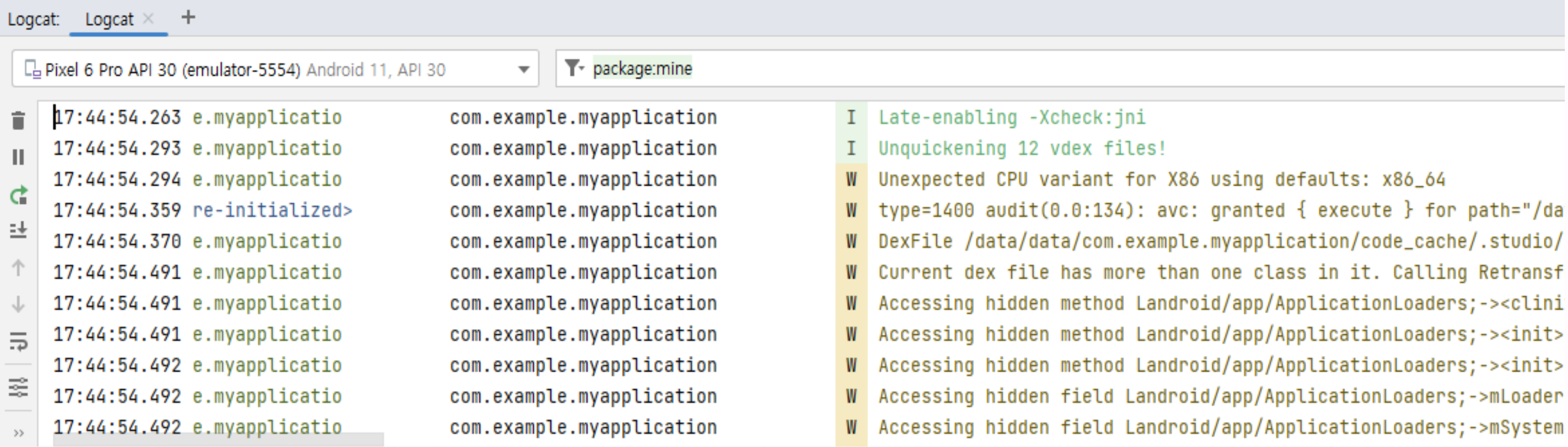
- The first application log
  - Tag + message



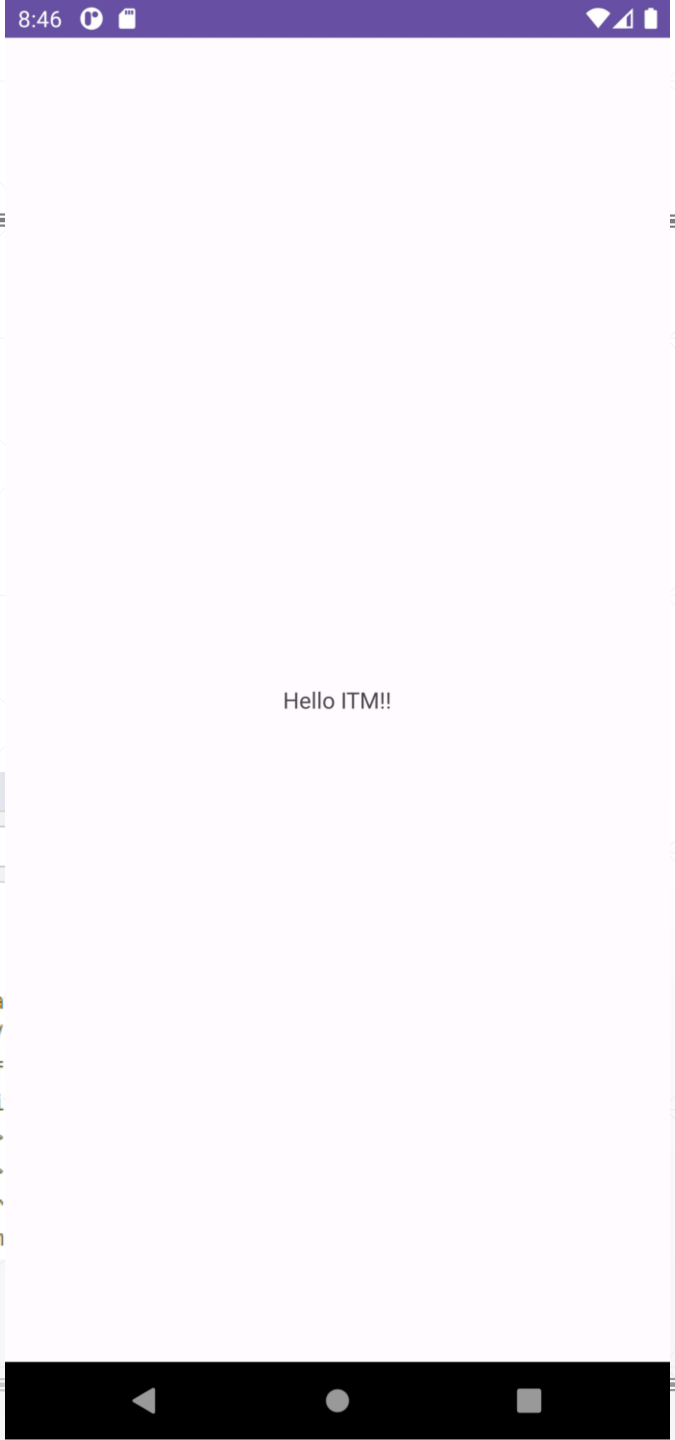
```
package com.example.kotlinbasics

import ...

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

        Log.d( tag: "ITM", msg: "I am Tiger Mask")
    }
}
```

Logcat (5/6)

- Run the app!
  - Nothing happens to your app visually
- What about the log perspective?
  - Some text there!



```
17:44:54.263 e.myapplicatio        com.example.myapplication         I  Late-enabling -Xcheck:jni
17:44:54.293 e.myapplicatio        com.example.myapplication         I  Unquickening 12 vdex files!
17:44:54.294 e.myapplicatio        com.example.myapplication         W  Unexpected CPU variant for X86 using defaults: x86_64
17:44:54.359 re-initialized>       com.example.myapplication         W  type=1400 audit(0.0:134): avc: granted { execute } for path="/da
17:44:54.370 e.myapplicatio        com.example.myapplication         W  DexFile /data/data/com.example.myapplication/code_cache/.studio/
17:44:54.491 e.myapplicatio        com.example.myapplication         W  Current dex file has more than one class in it. Calling Retransf
17:44:54.491 e.myapplicatio        com.example.myapplication         W  Accessing hidden method Landroid/app/ApplicationLoaders;-><clini
17:44:54.491 e.myapplicatio        com.example.myapplication         W  Accessing hidden method Landroid/app/ApplicationLoaders;-><init>
17:44:54.492 e.myapplicatio        com.example.myapplication         W  Accessing hidden method Landroid/app/ApplicationLoaders;-><init>
17:44:54.492 e.myapplicatio        com.example.myapplication         W  Accessing hidden field Landroid/app/ApplicationLoaders;->mLoader
17:44:54.492 e.myapplicatio        com.example.myapplication         W  Accessing hidden field Landroid/app/ApplicationLoaders;->mSystem
```

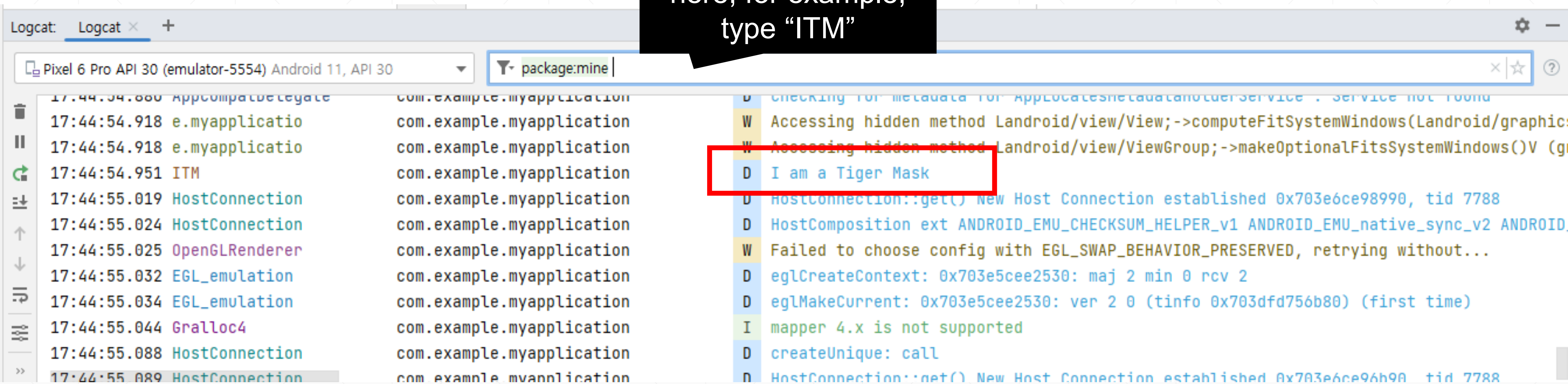


Hello ITM!!

Logcat (6/6)

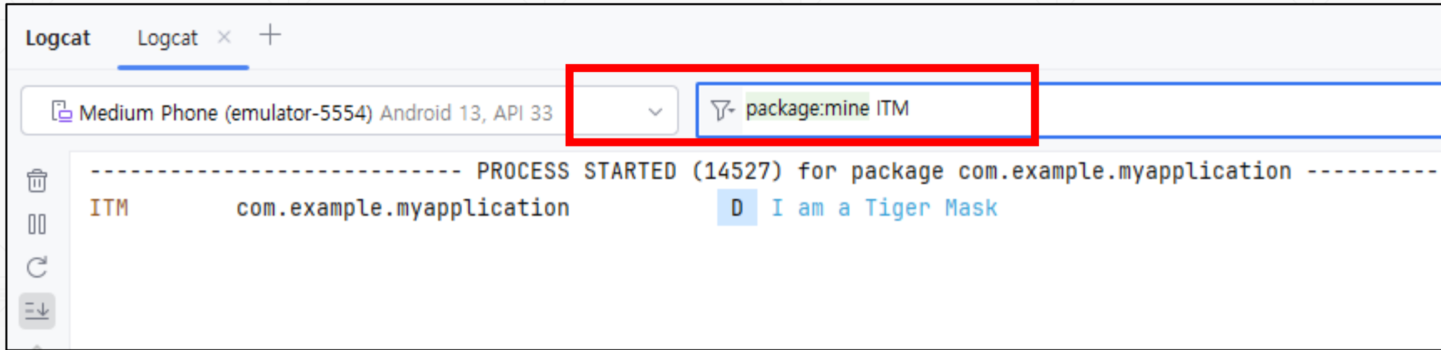
- … and our log!

You can use a filter here, for example, type “ITM”



package:mine

```
17:44:54.880 AppCompatDelegate   com.example.myapplication   D  Checking for metadata for AppLocalesMetadataHolderService : Service not found
17:44:54.918 e.myapplicatio      com.example.myapplication   W  Accessing hidden method Landroid/view/View;->computeFitSystemWindows(Landroid/graphic
17:44:54.918 e.myapplicatio      com.example.myapplication   W  Accessing hidden method Landroid/view/ViewGroup;->makeOptionalFitsSystemWindows()V (g
17:44:54.951 ITM                 com.example.myapplication   D  I am a Tiger Mask
17:44:55.019 HostConnection      com.example.myapplication   D  HostConnection::get() New Host Connection established 0x703e6ce98990, tid 7788
17:44:55.024 HostConnection      com.example.myapplication   D  HostComposition ext ANDROID_EMU_CHECKSUM_HELPER_v1 ANDROID_EMU_native_sync_v2 ANDROID
17:44:55.025 OpenGLRenderer      com.example.myapplication   W  Failed to choose config with EGL_SWAP_BEHAVIOR_PRESERVED, retrying without...
17:44:55.032 EGL_emulation       com.example.myapplication   D  eglCreateContext: 0x703e5cee2530: maj 2 min 0 rcv 2
17:44:55.034 EGL_emulation       com.example.myapplication   D  eglMakeCurrent: 0x703e5cee2530: ver 2 0 (tinfo 0x703dfd756b80) (first time)
17:44:55.044 Gralloc4            com.example.myapplication   I  mapper 4.x is not supported
17:44:55.088 HostConnection      com.example.myapplication   D  createUnique: call
17:44:55.089 HostConnection      com.example.myapplication   D  HostConnection::get() New Host Connection established 0x703e6ce96b90, tid 7788
```

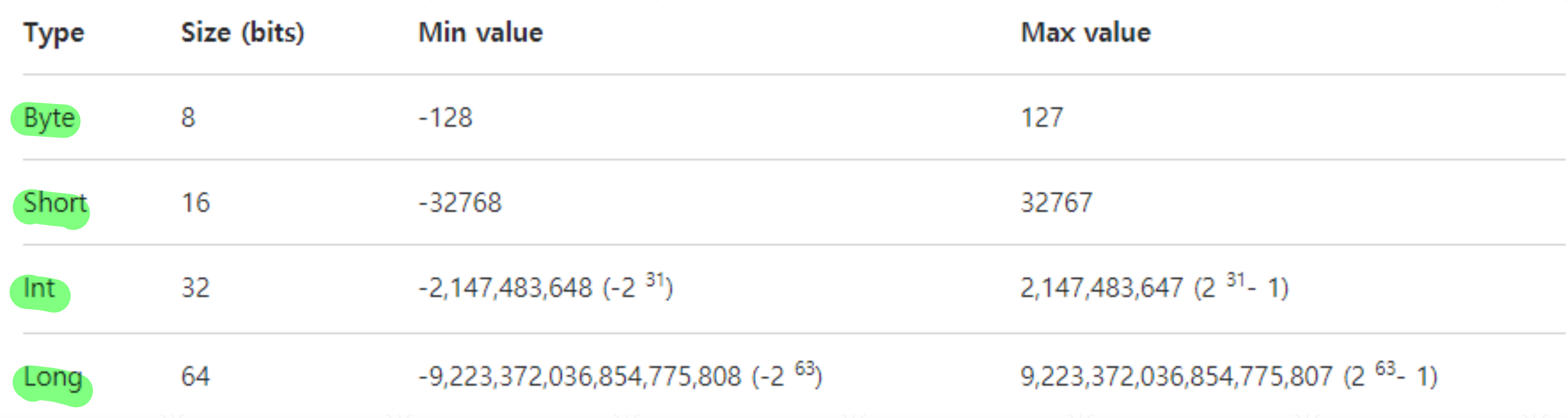


package:mine ITM

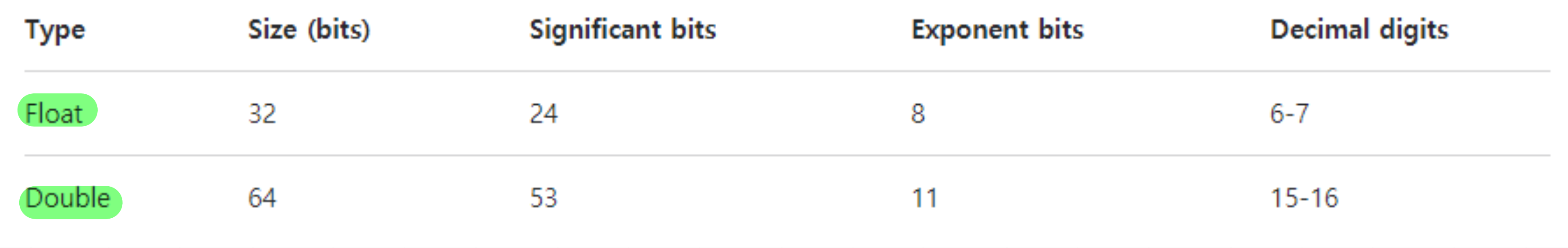
```
---------------------------- PROCESS STARTED (14527) for package com.example.myapplication ----------
ITM        com.example.myapplication              D  I am a Tiger Mask
```

Data Type (1/3)

- Very similar to modern programming languages



Type	Size (bits)	Min value	Max value
Byte	8	-128	127
Short	16	-32768	32767
Int	32	-2,147,483,648 (-2^{31})	2,147,483,647 $(2^{31}- 1)$
Long	64	-9,223,372,036,854,775,808 (-2^{63})	9,223,372,036,854,775,807 $(2^{63}- 1)$



Type	Size (bits)	Significant bits	Exponent bits	Decimal digits
Float	32	24	8	6-7
Double	64	53	11	15-16

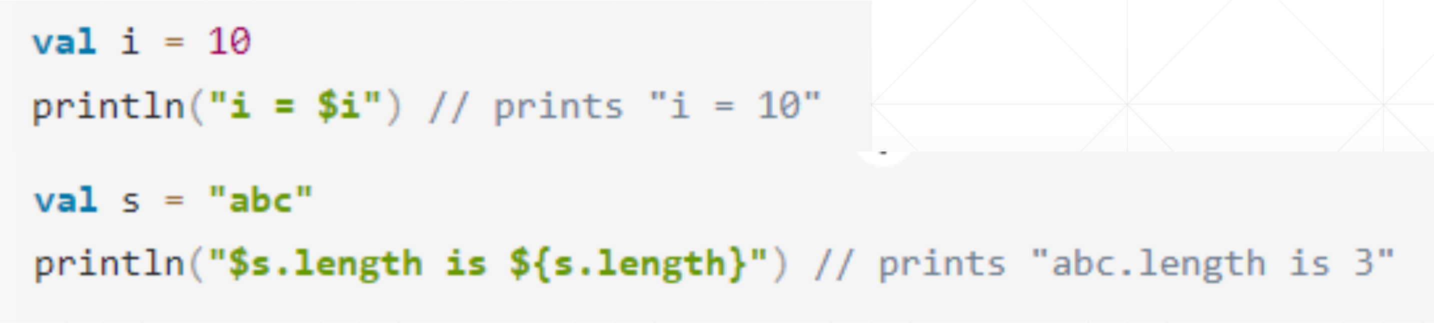
Data Type (2/3)

- Very similar to modern programming languages
  - The type Boolean represents boolean objects that can have two values: true and false
  - Characters are represented by the type Char
    - Character literals go in single quotes: '1'
    - Special characters start from an escaping backslash \. The following escape sequences are supported: \t, \b, \n, \r, \', \", \\ and \$
    - To encode any other character, use the Unicode escape sequence syntax: '\uFF00’
      - https://symbl.cc/en/unicode/blocks/
  - Strings in Kotlin are represented by the type String
    - A string value is a sequence of characters in double quotes (")

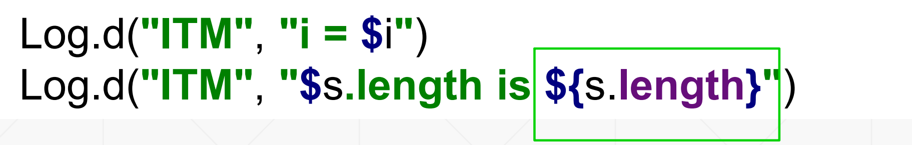
Data Type (3/3)

String template

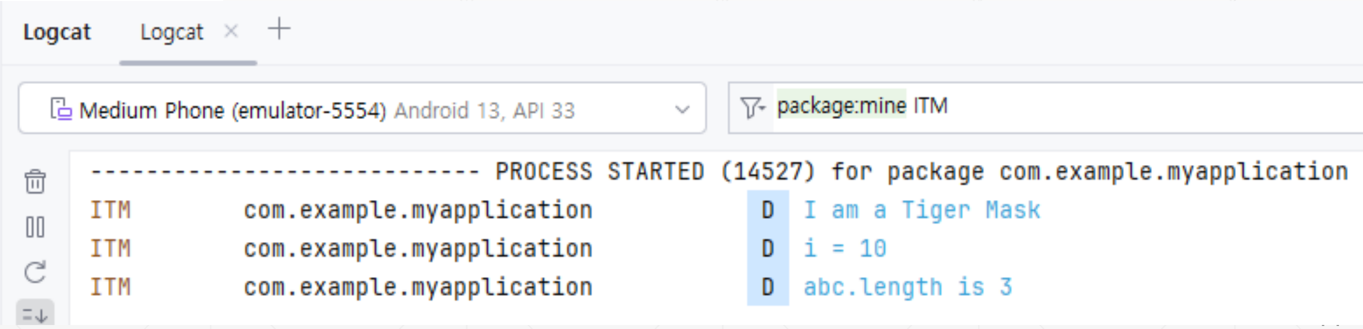
- Template expressions - pieces of code that are evaluated and whose results are concatenated into the string
- Template expression starts with a dollar sign ($) and consists of either
  - Name
  - Expression in curly braces



```
val i = 10
println("i = $i") // prints "i = 10"

val s = "abc"
println("$s.length is ${s.length}") // prints "abc.length is 3"
```



```
Log.d("ITM", "i = $i")
Log.d("ITM", "$s.length is ${s.length}")
```



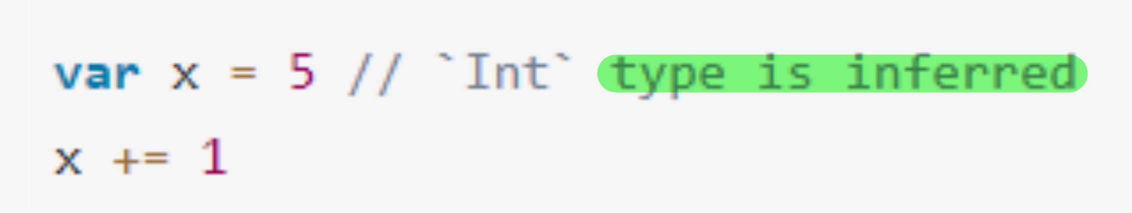
Medium Phone (emulator-5554) Android 13, API 33

package:mine ITM

```
---------------------------- PROCESS STARTED (14527) for package com.example.myapplication
ITM        com.example.myapplication          D  I am a Tiger Mask
ITM        com.example.myapplication          D  i = 10
ITM        com.example.myapplication          D  abc.length is 3
```

Variable and Constant

- Variables that can be reassigned use the *var* keyword

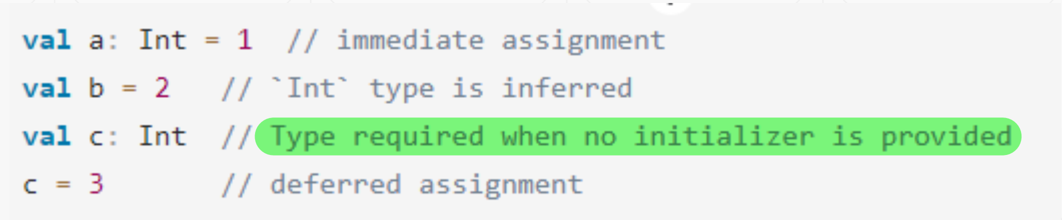


```
var x = 5 // `Int` type is inferred
x += 1
```

  - You can just 'declare' a variable with its type definition
  - e.g) var x: Int

constant

- Read-only local variables are defined using the keyword *val*
  - 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       // deferred assignment
```

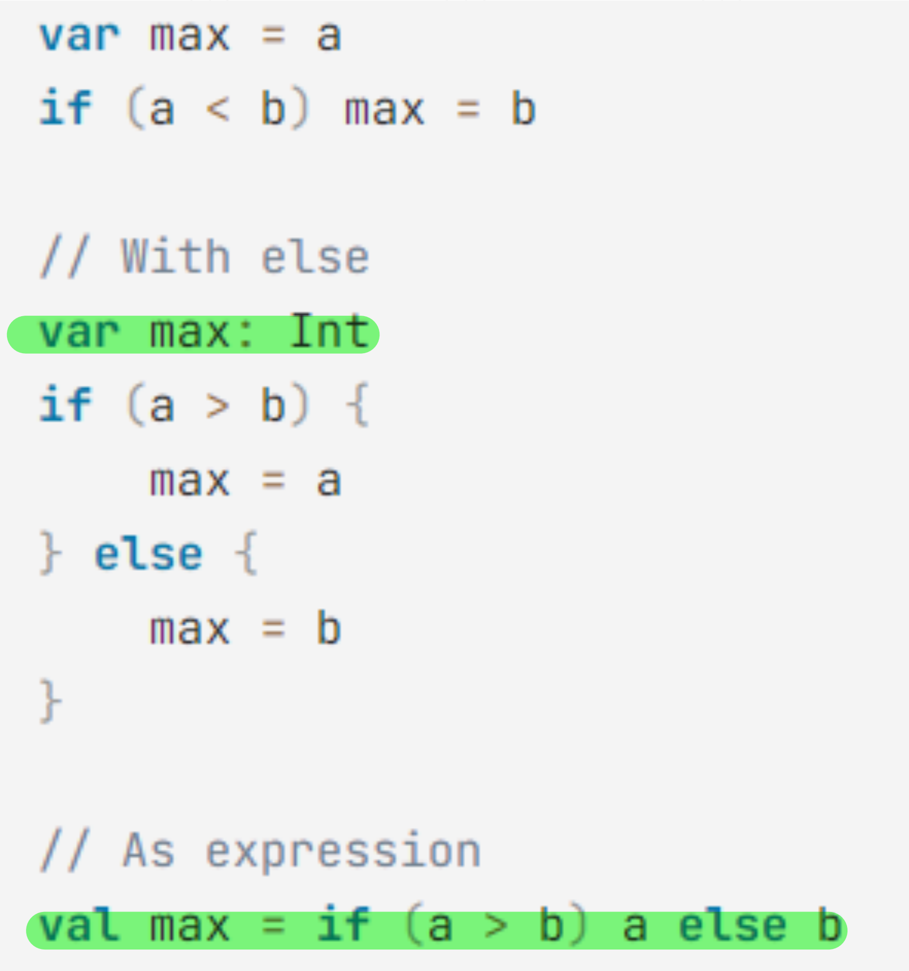
Conditional Expression (1/5)

- if statement
  - if (single condition)
  - if else (exclusive condition)
  - if else if … else (branches)

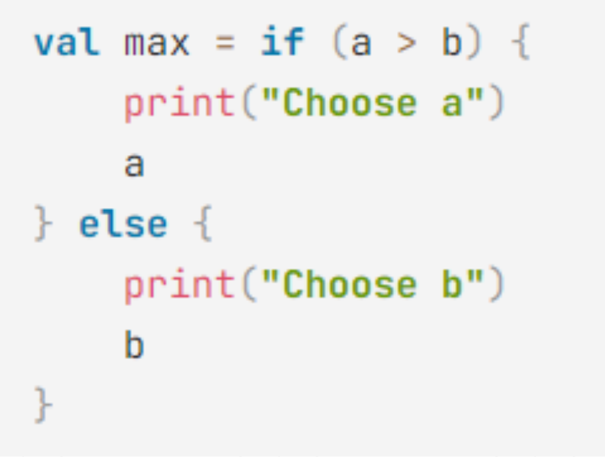
- Conditions
  - Equality checks: a == b and a != b
  - Comparison operators: a < b, a > b, a <= b, a >= b
  - ||: disjunction (logical OR)
  - &&: conjunction (logical AND)
  - !: negation (logical NOT)

Conditional Expression (2/5)

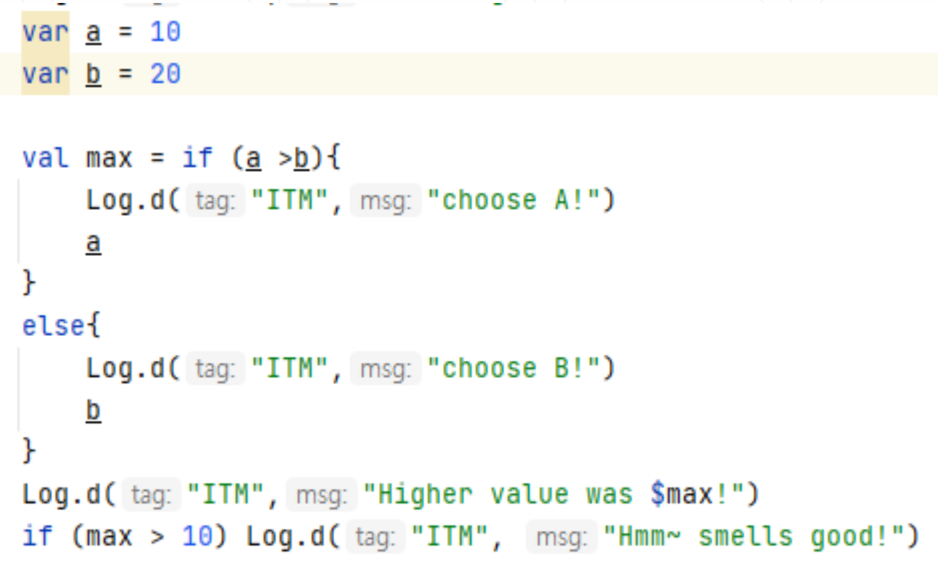
- if statement



```
var max = a
if (a < b) max = b

// With else
var max: Int
if (a > b) {
    max = a
} else {
    max = b
}

// As expression
val max = if (a > b) a else b
```



```
val max = if (a > b) {
    print("Choose a")
    a
} else {
    print("Choose b")
    b
}
```



```
var a = 10
var b = 20

val max = if (a >b){
    Log.d( tag: "ITM", msg: "choose A!")
    a
}
else{
    Log.d( tag: "ITM", msg: "choose B!")
    b
}
Log.d( tag: "ITM", msg: "Higher value was $max!")
if (max > 10) Log.d( tag: "ITM",  msg: "Hmm~ smells good!")
```

Conditional Expression (3/5)

- when statement
  - Very similar to switch-case statements in other languages
  - Basic syntax

```
when (parameter) {
    condition1 -> action1
    condition2 -> action2
    else -> { // Note the block
        action3
    }
}
```

```
val grade = "A+"
when(grade){
    "A+" -> Log.d("ITM","Oh.. great..!!")
    "B+" -> Log.d("ITM","Hey?")
    else -> {
        Log.d("ITM","I don't know who you are.")
    }
}
```

  - With comma

```
when (parameter) {
    condition1,  condition2 -> action1
    else -> { // Note the block
        action2
    }
}
```

```
val grade = "F"
when (grade) {
    "A+","B+","C+" -> Log.d("ITM", "Oh.. great..!!")
    "A","B","C" -> Log.d("ITM", "Hey?")
    else -> {
        Log.d("ITM", "I don't know who you are.")
    }
}
```

Conditional Expression (4/5)

- when statement
  - in or !in syntax for a range check

```
val score = 89
when (score) {
    in 90..100 -> Log.d("ITM", "A+")
    else -> Log.d("ITM", "resit!!!")
}
```

  - when without a parameter
    - Replacement for an if- else if chain

```
val score = 49
when {
    score > 90 -> Log.d("ITM","A+")
    score in 50..89 -> Log.d("ITM", "A0")
    else -> Log.d("ITM","resit!!!")
}
```

Conditional Expression (5/5)

- when statement
  - Type checking (is syntax)

```
val x:Any =20.5 float
when (x){
   is Int -> Log.d("ITM","It's Int type!")
   is String -> Log.d("ITM", "it' String type!")
   else -> Log.d("ITM","What is this?")
}
```

  - when as expression

```
val myScore = 80

var myGrade = when (myScore){
   in 90..100 -> "A+"
   else -> "F"
}
Log.d("ITM","my score is $myScore, so my grade is $myGrade")
```

Array (1/3)

- Set of data with a fixed length
  - Initialization
    - Array() constructor takes the size and the function that returns values of array elements given its index
    - arrayOf() takes items as input and create an array of them
  - get()/set() functions ([] operator)
  - size property
  - …

- Primitive-type array
  - ByteArray, ShortArray, IntArray, LongArray, CharArray, FloatArray, DoubleArray, BooleanArray

Array (2/3)

Examples

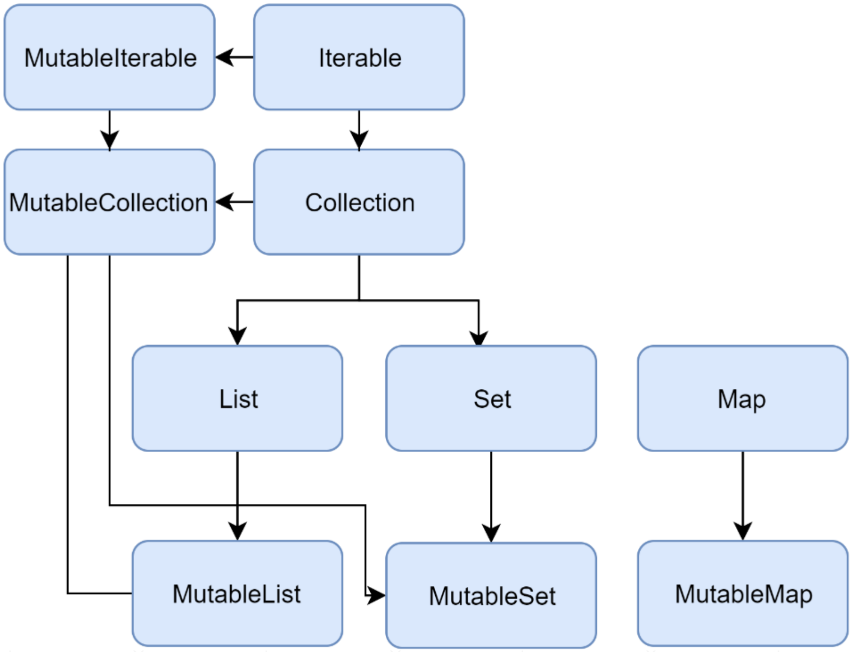
```
val x: IntArray = intArrayOf(1, 2, 3) // use these values to create intArray
val y = arrayOf("one","two","three")
val arr = IntArray(5) // create IntArray with size of 5 {0,0,0,0,0}
val arr2 = IntArray(5) { 42 } // create IntArray with size of 5, values of 42 {42, 42, 42, 42, 42}
var arr3 = IntArray(5) { it+1 } // create IntArray with size of 5, values of index+1 {1, 2, 3, 4, 5}
var arr4 = Array(5){i -> i*i} {0^2, 1^2, 2^2, 3^2, 4^2}

Log.d("ITM","x.size: ${x.size}") 3
Log.d("ITM","arr.size: ${arr.size}") 5

Log.d("ITM",Arrays.toString(x))
Log.d("ITM",Arrays.toString(y))
Log.d("ITM",Arrays.toString(arr))
Log.d("ITM",Arrays.toString(arr2))
Log.d("ITM",Arrays.toString(arr3))
Log.d("ITM",Arrays.toString(arr4))
```

Array (3/3)

- Some utilities
  - get/set ([])
  - first()/last()
  - sort(), sortedArray()
  - …

```
val arr: IntArray = IntArray(10){it+1} {1,2,3,4,5,6,7,8,9,10}
val first = arr[0]      1
val first2 = arr.first() 1
val first3 = arr.get(0)  1
val last = arr.last()  10

Log.d("ITM","${Arrays.toString(arr)}")
Log.d("ITM","first: $first $first2 $first3 last: $last")

arr[0] = 100
arr.set(1,200)

Log.d("ITM","${Arrays.toString(arr)}") {100, 200, 3, 4, 5, 6, 7, 8, 9, 10}

arr.sort() // in-place sorting {3, 4, 5, 6, 7, 8, 9, 10, 100, 200}
val arr2 = arr.sortedArrayDescending() // return a new sorted array (desc.)
                                  {200, 100, 10, 9, 8, 7, 6, 5, 4, 3}
Log.d("ITM","${Arrays.toString(arr)}")
Log.d("ITM","${Arrays.toString(arr2)}")
```

Collections (1/2)



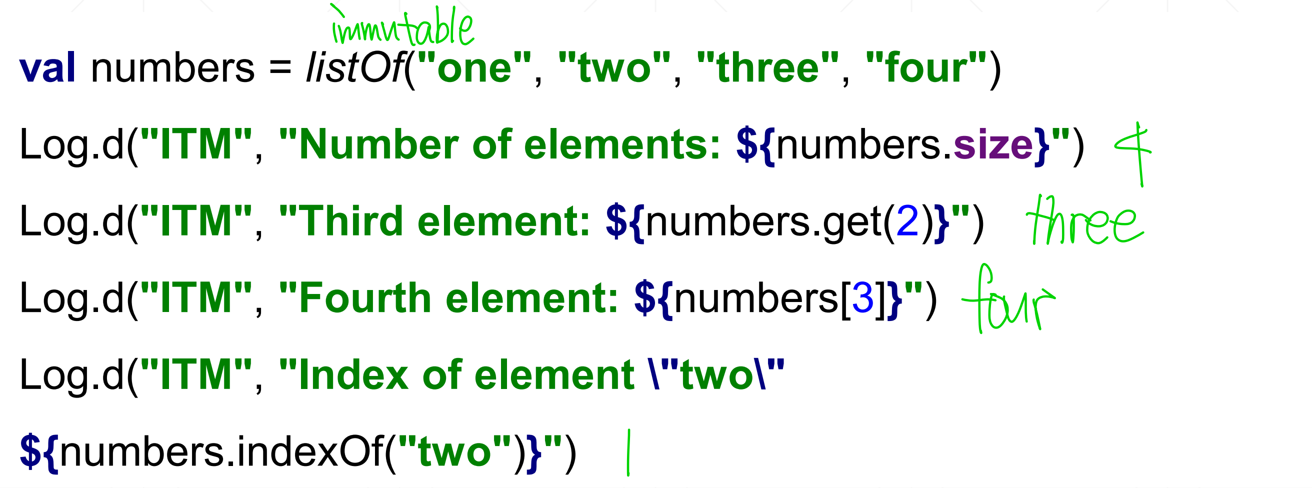
- Container to store a number of objects dynamically
  - List
    - Ordered collection with access to elements by indices
    - Elements can occur more than once in a list
  - Set
    - Collection of unique elements
    - Reflects the mathematical abstraction of set: a group of objects without repetitions
    - The order of set elements has no significance
  - Map (or dictionary)
    - A set of key-value pairs
    - Keys are unique, and each of them maps to exactly one value
    - Values can be duplicates

Collections (2/2)

- Immutable collections
  - A read-only interface that provides operations for accessing collection elements
- Mutable collections
  - Extends the corresponding read-only interface with write operations: adding, removing, and updating its elements
- Construction of collections
  - listOf(), mutableListOf()
  - setOf(), mutableSetOf()
  - mapOf(), mutableMapOf()

Collections: List (1/2)

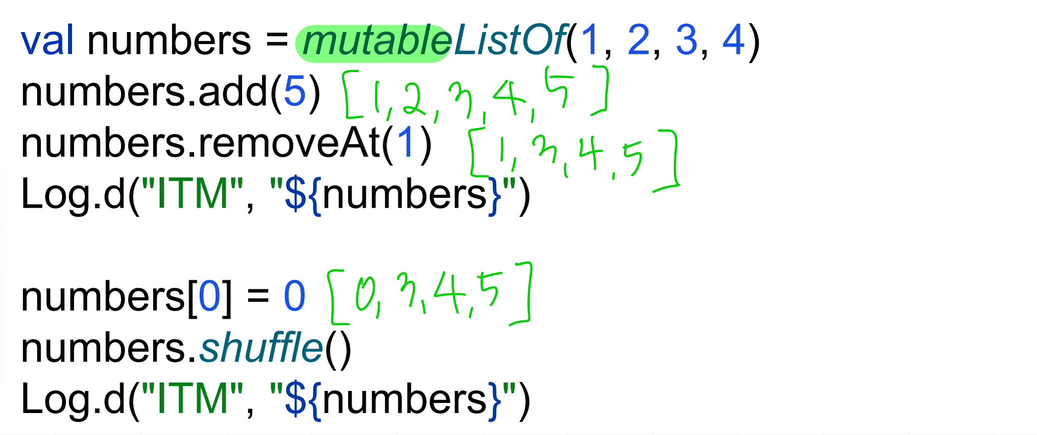
- Stores elements in a specified order and provides indexed access to them
  - Very similar to the array!
- Elements can duplicate
  - List can contain any number of equal objects or occurrences of a single object



```
                        immutable
val numbers = listOf("one", "two", "three", "four")
Log.d("ITM", "Number of elements: ${numbers.size}")  4
Log.d("ITM", "Third element: ${numbers.get(2)}")  three
Log.d("ITM", "Fourth element: ${numbers[3]}")  four
Log.d("ITM", "Index of element \"two\"
${numbers.indexOf("two")}")   1
```

Collections: List (2/2)

- List-specific write operations
  - add()
  - removeAt()
  - shuffle()
  - Indexing ([])



```
val numbers = mutableListOf(1, 2, 3, 4)
numbers.add(5)            [1,2,3,4,5]
numbers.removeAt(1)         [1,3,4,5]
Log.d("ITM", "${numbers}")

numbers[0] = 0   [0,3,4,5]
numbers.shuffle()
Log.d("ITM", "${numbers}")
```

Collections: Set (1/2)

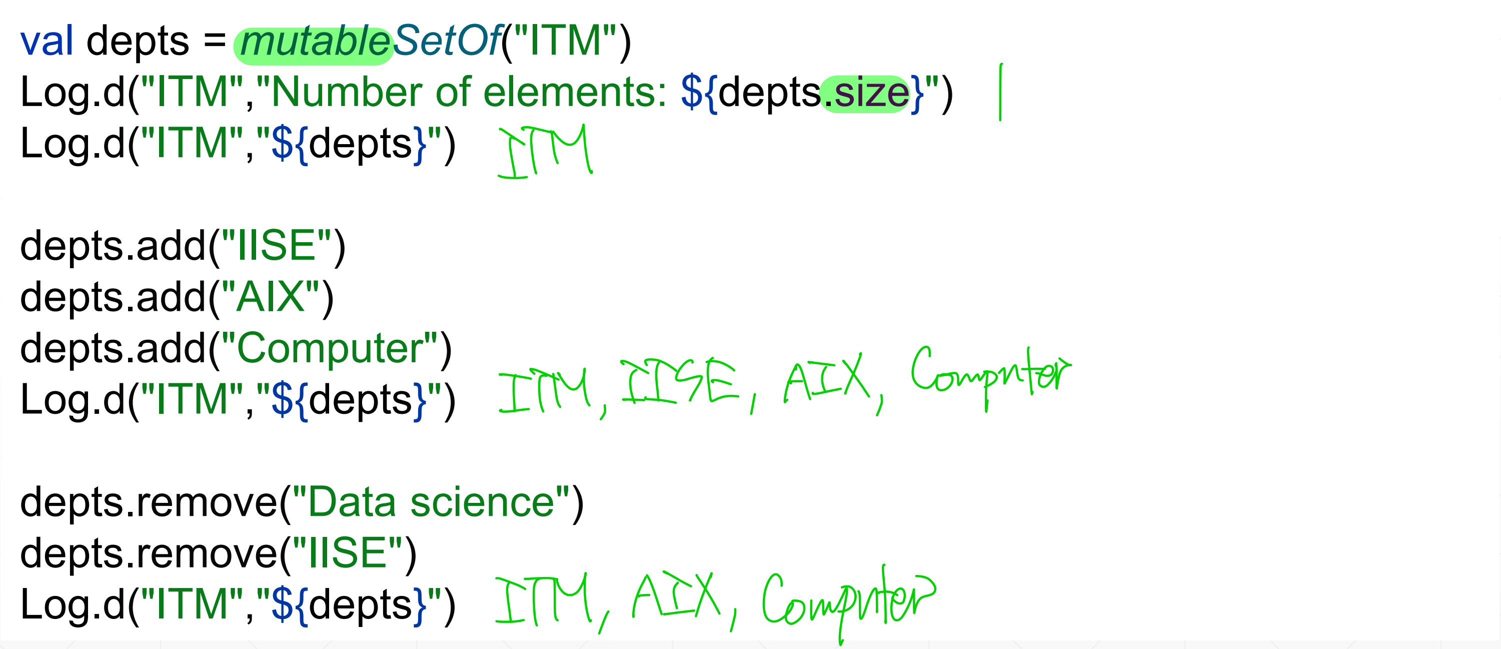
- Stores unique elements
  - Their order is generally undefined!

```
val numbers = setOf(1, 2, 3, 4)
Log.d("ITM","Number of elements: ${numbers.size}") 4
if (numbers.contains(1)) Log.d("ITM","1 is in the set") True


val numbersBackwards = setOf(4, 3, 2, 1)
Log.d("ITM","The sets are equal: ${numbers == numbersBackwards}")
                                                     True
```

  - No support of [] , get() operations
    ⇒ no order!

Collections: Set (2/2)

- Set-specific write operations
  - add()
  - remove()



```
val depts = mutableSetOf("ITM")
Log.d("ITM","Number of elements: ${depts.size}")   1
Log.d("ITM","${depts}")   ITM

depts.add("IISE")
depts.add("AIX")
depts.add("Computer")
Log.d("ITM","${depts}")   ITM, IISE, AIX, Computer

depts.remove("Data science")
depts.remove("IISE")
Log.d("ITM","${depts}")   ITM, AIX, Computer
```

Collections: Map (1/2)

- Stores key-value pairs (or entries)
  - keys are unique, but different keys can be paired with equal values

- Creation
  - mapOf<KeyType, ValueType>()
  - mutableMapOf<KeyType, ValueType>()
  - Pair() type or "Key" to "value" data can be used for initialization

- Map-specific write functions
  - put()
  - remove()

Collections: Map (2/2)

- Example

```
val studentGrade = mutableMapOf<String, Int>()
studentGrade.put("Jeong",100)
studentGrade.put("Kim",90)
studentGrade.put("Hong",80)
studentGrade.put("Park",70)

Log.d("ITM","${studentGrade}")
```

J 100 K 90
H 80 P 70

```
Log.d("ITM","${studentGrade.get("Jeong")}")
```

100

```
Log.d("ITM","${studentGrade.get("Wow")}")
```

null

updated!

```
studentGrade.put("Jeong",0)
Log.d("ITM","${studentGrade}")
```

J 0
K 90
H 80
P 70

Equivalent form!

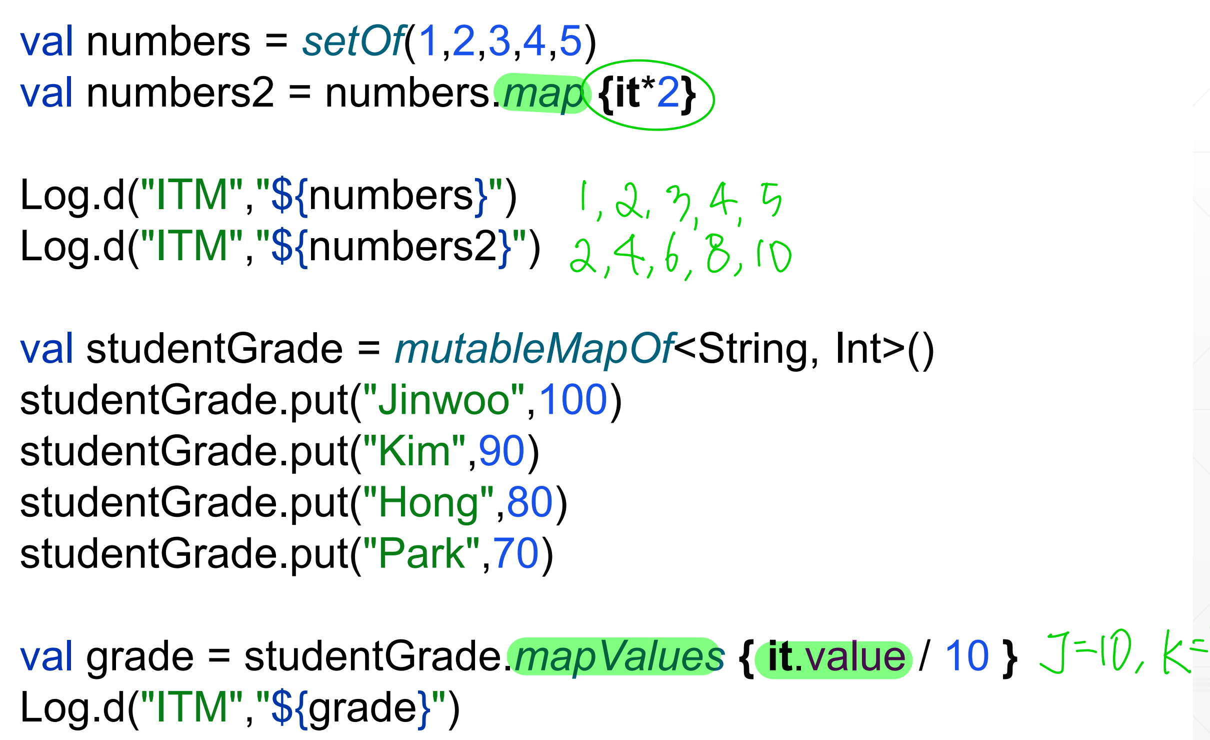
```
val studentGrade = mutableMapOf<String, Int>
("Jeong" to 100, "Kim" to 90, "Hong" to 80, "Park" to 70)
```

```
val studentGrade = mutableMapOf<String, Int>
(Pair("Jeong",100), Pair("Kim",90), Pair("Hong",80), Pair("Park",70),)
```

Try to run these initialization codes!

Collections: Transformation (1/3)

- map()
  - Creates a collection from the results of a function on the elements of another collection
  - Applies the given lambda function to each subsequent element and returns the list of the lambda results
  - Order of results is the same as the original order of elements



```
val numbers = setOf(1,2,3,4,5)
val numbers2 = numbers.map {it*2}

Log.d("ITM","${numbers}")     1, 2, 3, 4, 5
Log.d("ITM","${numbers2}")    2,4,6,8,10

val studentGrade = mutableMapOf<String, Int>()
studentGrade.put("Jinwoo",100)
studentGrade.put("Kim",90)
studentGrade.put("Hong",80)
studentGrade.put("Park",70)

val grade = studentGrade.mapValues { it.value / 10 }  J=10, K=9, H=8, P=7
Log.d("ITM","${grade}")
```

Collections: Transformation (2/3)

zip()

- Builds pairs from elements with the same positions in both collections
- Returns the List of Pair objects
- If the collections have different sizes, the result of the zip() is the smaller size
  - The last elements of the larger collection are not included in the result

```
val colors = listOf("red", "brown", "grey")
val animals = listOf("fox", "bear", "wolf")
Log.d("ITM","${colors zip animals}")
```

(red, fox), (brown, bear), (grey, wolf)

```
val twoAnimals = listOf("fox", "bear")
Log.d("ITM","${colors.zip(twoAnimals)}")
```

(red, fox), (brown, bear)

```
Log.d("ITM", "${colors.zip(animals) { color, animal -> "The ${color.replaceFirstChar { it.uppercase() }} is $animal"}}")
```

Red is fox
Brown is bear
Grey is wolf

Collections: Transformation (3/3)

- filter()
  - When called with a predicate, filter() returns the collection elements that match it
  - For both List and Set, the resulting collection is a List, for Map it's a Map

```
val numbers = listOf("one", "two", "three", "four")
val longerThan3 = numbers.filter { it.length > 3 }
Log.d("ITM","$longerThan3")
```

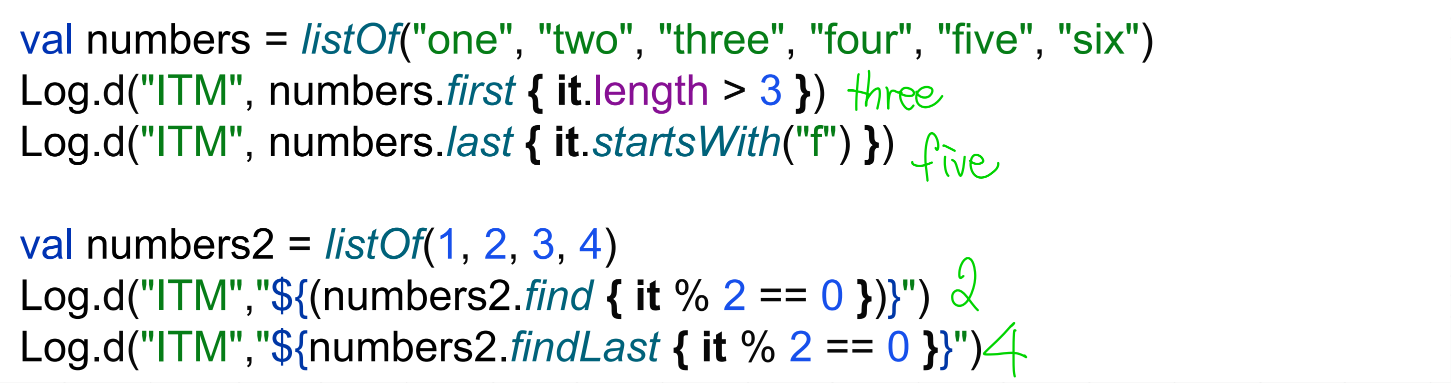
[three, four] ⇒ list

```
val numbersMap = mapOf("key1" to 1, "key2" to 2, "key3" to 3, "key11" to 11)
val filteredMap = numbersMap.filter { (key, value) -> key.endsWith("1") && value > 10}
Log.d("ITM","$filteredMap")
```

{key11=11} ⇒ map

Collections: Retrieval

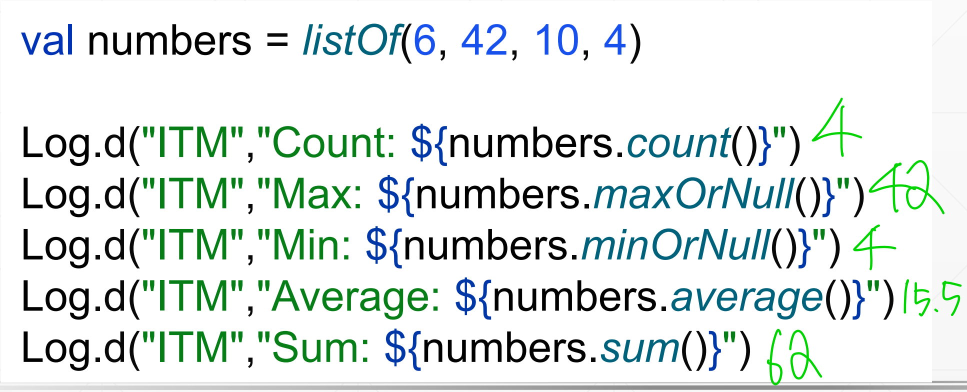
- first()/last()
  - Search a collection for elements matching a given predicate
  - first()/find(): you will receive the first element on which the predicate yields true!
  - last()/findLast(): returns the last element matching a given predicate!
  - first() & last() can throw exceptions if no element is found ⇒ app crash!



```
val numbers = listOf("one", "two", "three", "four", "five", "six")
Log.d("ITM", numbers.first { it.length > 3 }) three
Log.d("ITM", numbers.last { it.startsWith("f") }) five

val numbers2 = listOf(1, 2, 3, 4)
Log.d("ITM","${(numbers2.find { it % 2 == 0 })}") 2
Log.d("ITM","${numbers2.findLast { it % 2 == 0 }}")4
```

Collections: Aggregate

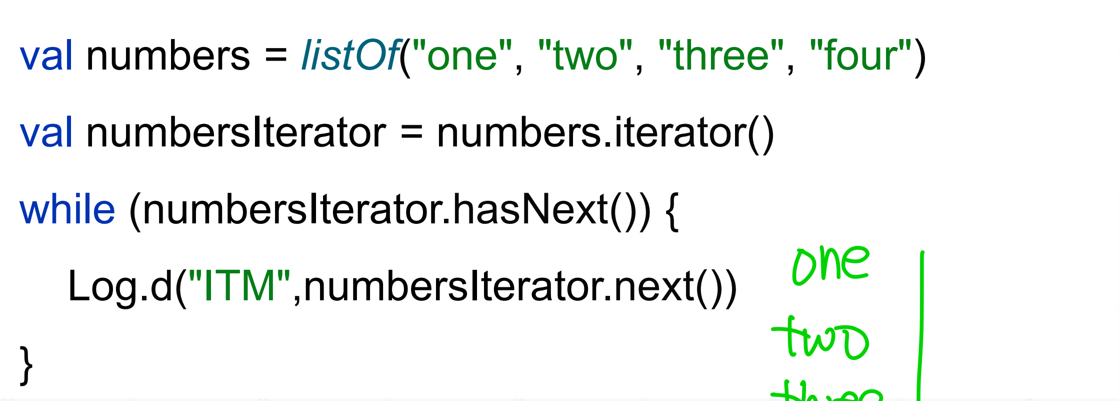
- minOrNull()/maxOrNull()
  - Return the smallest and the largest element respectively
  - On empty collections, they return null
- average()
  - Returns the average value of elements in the collection of numbers
- sum()
  - Returns the sum of elements in the collection of numbers
- count()
  - Returns the number of elements in a collection



```
val numbers = listOf(6, 42, 10, 4)

Log.d("ITM","Count: ${numbers.count()}") 4
Log.d("ITM","Max: ${numbers.maxOrNull()}") 42
Log.d("ITM","Min: ${numbers.minOrNull()}") 4
Log.d("ITM","Average: ${numbers.average()}") 15.5
Log.d("ITM","Sum: ${numbers.sum()}") 62
```

Collections: Iterator

- Object that provides access to the elements sequentially without exposing the underlying structure of the collection
- Usage
  - Iterator(): to obtain an iterator instance from the collection
  - Next(): returns the element and moves the iterator position to the following element if it exists
  - hasNext(): checks if the following element exists



```
val numbers = listOf("one", "two", "three", "four")
val numbersIterator = numbers.iterator()
while (numbersIterator.hasNext()) {
    Log.d("ITM",numbersIterator.next())
}
```

one
two
three
four

Q&A

- Next week (eClass video)
  - Kotlin Basics (Part II & III)