Bevezetés a Kotlin fejlesztésbe

Ekler Péter

peter.ekler@aut.bme.hu

BME AUT Department of Automation and Applied Informatics

AutSoft

Tanfolyam felépítése

1. Kotlin bevezetés
   - Nyelv alap elemei
   - Android alkalmazás fejlesztés
2. Listák kezelése Kotlin nyelven
   - RecyclerView használata
   - Fejlett nyelvi elemek
3. Szerver oldali fejlesztés Kotlin nyelven
   - Spring framework
   - REST API megvalósítás

- GIT:
  - https://github.com/AutSoft/NetAcademiaKotlinIntro

Kotlin alapok

Konstansok, változók (val vs. var)

- Egyszeri értékadás – „val”

```
val score: Int = 1  // azonnali értékadás
val idx = 2   // típus elhagyható
val age: Int  // típus szükséges ha nincs azonnali értékadás
age = 3       // későbbi értékadás
```

- Változók (megváltoztatható) – „var”

```
var score = 0 // típus elhagyható
score += 1
```

- String sablonok

```
var score = 1
val scoreText = "$score pont"

score = 2
// egyszerű kifejezeések string-ek esetében:
val newScoreText = "${scoreText.replace("pont", "volt, most ")} $score"
```

Függvények

- Függvény szintaxis

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

- Kifejezés törzs, visszatérési típus elhagyható

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

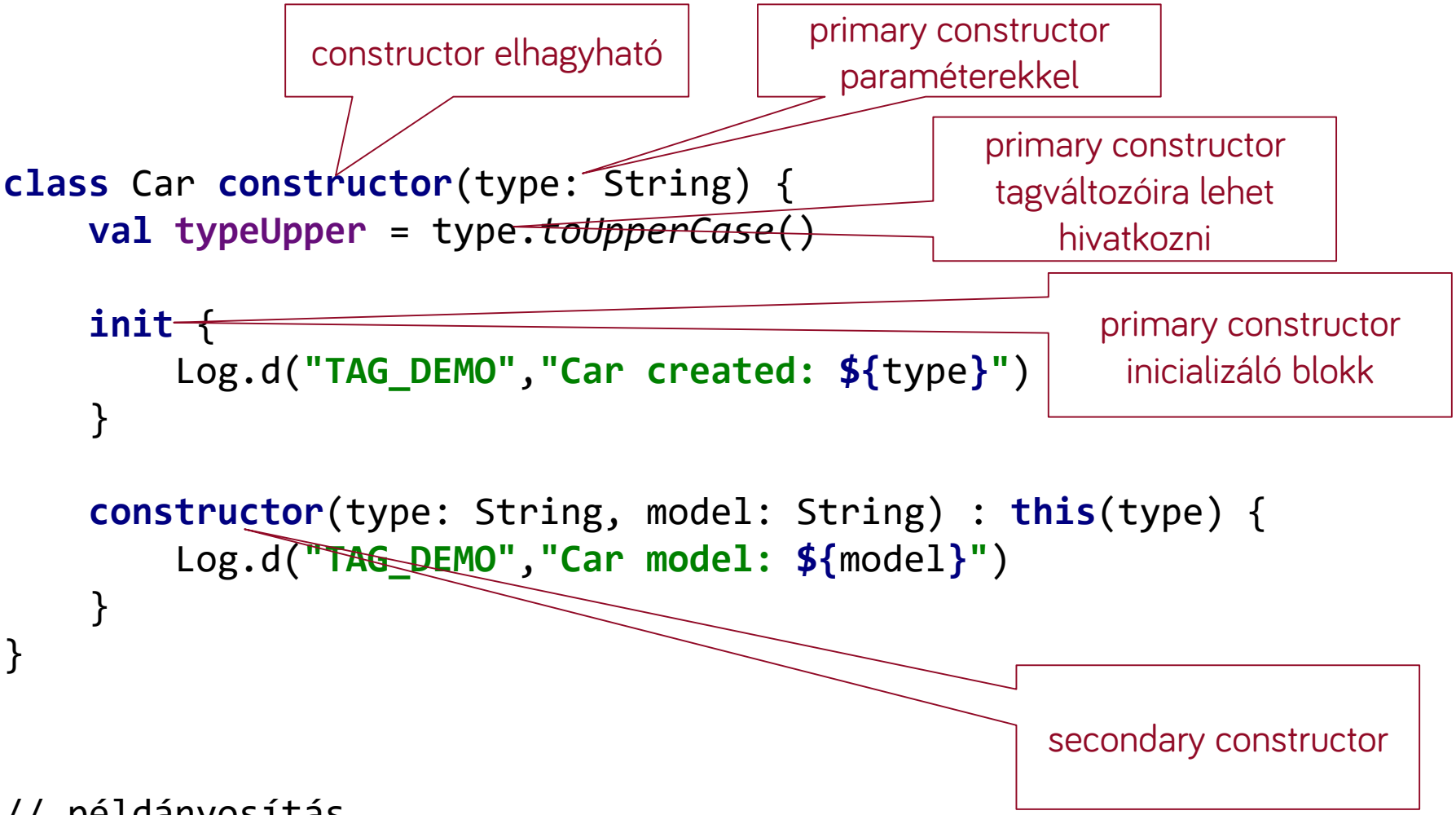
- Érték nélküli visszatérés – Unit

```
fun printAddResult(a: Int, b: Int): Unit {
    println("$a + $b értéke: ${a + b}")
}
```

- Unit elhagyható

```
fun printAddResult(a: Int, b: Int) {
    println("$a + $b értéke: ${a + b}")
}
```

Osztályok




```
class Car constructor(type: String) {
    val typeUpper = type.toUpperCase()

    init {
        Log.d("TAG_DEMO","Car created: ${type}")
    }

    constructor(type: String, model: String) : this(type) {
        Log.d("TAG_DEMO","Car model: ${model}")
    }
}


// példányosítás
val car = Car("Toyota")
```

Leszármaztatás

alapesetben minden final

```
open class Item(price: Int) {
    open fun calculatePrice() {}
    fun load() {}
}


class SpecialItem(price : Int) : Item(price) {
    final override fun calculatePrice() {}
}
```

öröklés

Később már nem lehet felülírni

Data class

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

- Automatikusan létrejön:
  - equals()/hashCode()
  - toString(): "Ship(name=Discovery, age=31)";
  - componentN() metódusok
  - copy() metódus

```
val discovery = Ship("Discovey", 31)
val (name, age) = discovery

//val name = discovery.component1()
//val age = discovery.component2()
```

- Követelmények Data osztályokkal szemben:
  - Primary constructor legalább 1 paraméterrel
  - Minden primary constructor paraméter *val* vagy *var*
  - *Data* classes nem lehet *abstract, open, sealed, vagy inner*

Delegate

```
interface Pressable {
    fun press()
}

class MyButton(val x: Int) : Pressable {
    override fun press() { Log.d("TAG_MINE","press $x") }
}

class SpecialButton(pressable: Pressable) : Pressable by pressable

fun main(args: Array<String>) {
    val btn = MyButton(10)
    SpecialButton(btn).press() // press 10
}
```

- *by* kulcsszó miatt tovább hív a btn implementációba

Higher order functions

- Olyan metódus, amely metódust kap paraméterül, vagy metódussal tér vissza

```
fun <T> lock(lock: Lock, body: () -> T): T {
    lock.lock()
    try {
        return body()
    }
    finally {
        lock.unlock()
    }
}
```

- Függvény típus: ()->T
- Használat:

```
fun toBeSynchronized() = sharedResource.operation()
val result = lock(lock, ::toBeSynchronized)
```

Extensions

- Új funkció hozzáadása egy osztályhoz anélkül, hogy leszármaztatnánk belőle

```
private fun TicTacToeView.resetGame() {
    TicTacToeModel.resetModel()
    invalidate()
}
```

- *Valójában nem ad hozzá egy új függvényt az osztályhoz, csak lehetővé teszi ezt a függvény meghívását az adott típusú objektumokon*

Néhány érdekesség

- Range-k

```
val x = 4
val y = 3
if (x in 1..y+1) {
    Log.d("TAG_DEMO", "x benne van")
}
```

- Range iteráció

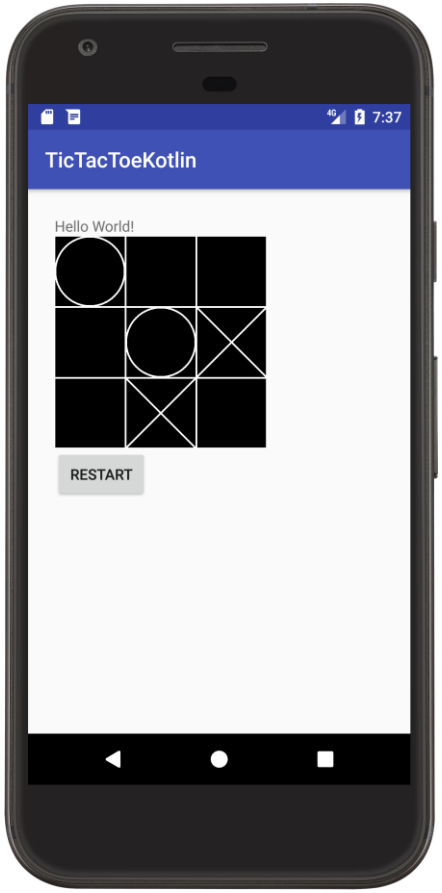
```
for (nr in 1..10 step 2) {
    Log.d("TAG_DEMO", "szam $nr")
}
```

- Lambda műveletek kollekciókon

```
val fruits = listOf("alma", "mango", "mandarin", "narancs")
fruits
    .filter { it.startsWith("m") }
    .sortedBy { it }
    .map { it.toUpperCase() }
    .forEach { Log.d("TAG_DEMO", "$it") }
```

Gyakoroljunk

- Készítsünk TicTacToe alkalmazást Kotlin nyelven



Köszönöm a figyelmet!

- Forrás:
  - https://kotlinlang.org/

peter.ekler@aut.bme.hu