Kotlin

Szécsi László

3D Grafikus Rendszerek

1. előadás

Kotlin és WebGL

- Java helyett
  - mobilprogramozás
- JavaScriptre is fordul
  - WebGL fejlesztésre alkalmas
  - erősen típusos
  - nem runtime hibákat kell bogarászni

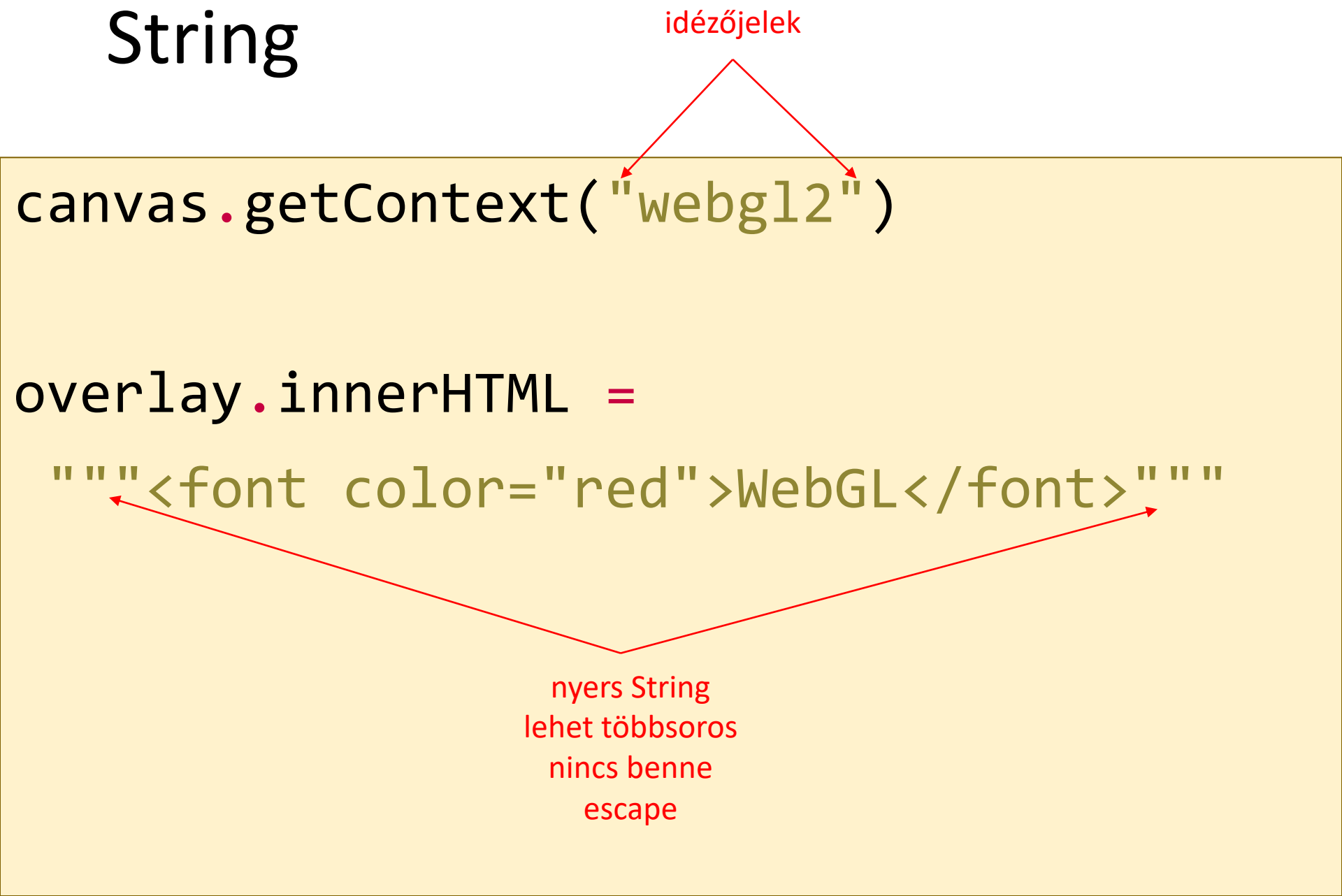
Alaptípusok

- Byte, Short, Int, Long, **Float**, Double, Char, Boolean
  - mind osztályként viselkedik
  - nem elérhető a belső primitív

Változók

```
var yaw : Float
val myPi = 3.14f
```

String


idézőjelek
canvas.getContext("webgl2")
overlay.innerHTML =
"""WebGL"""
nyers String
lehet többsoros
nincs benne
escape

String template

```
var fontSize = 1

overlay.innerHTML =

"""<div style=

"position:absolute;left:${mouseX}px;bottom:-${mouseY}px">

<font color="red" size="${fontSize}px">
${keysPressed.toString()}

</font></div>""";
```


kiértékelt kifejezés

Matek

```
import kotlin.math.sqrt


fun length() : Float {
   return sqrt(lengthSquared());
}
```

Matek

```
import kotlin.math.sqrt


fun length() : Float {
   return sqrt(lengthSquared());
}
```

Random

```
import kotlin.random.Random


fun makeRandom(minVal: Vec4 = Vec4.zeros, maxVal: Vec4 = Vec4.ones) :
Vec4 {
  return Vec4(
    Random.nextFloat() *
        (maxVal.storage[0] - minVal.storage[0]) + minVal.storage[0],
    Random.nextFloat() * (maxVal.storage[1] - minVal.storage[1]) + minVal.storage[1],
    Random.nextFloat() * (maxVal.storage[2] - minVal.storage[2]) + minVal.storage[2],
    Random.nextFloat() * (maxVal.storage[3] - minVal.storage[3]) + minVal.storage[3]
  )
}
```

Függvények

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

Osztályok

```
class Hello {
   val greeting : String = "Hello world!"

   fun greet() {
      print(greeting)
   }
}
```

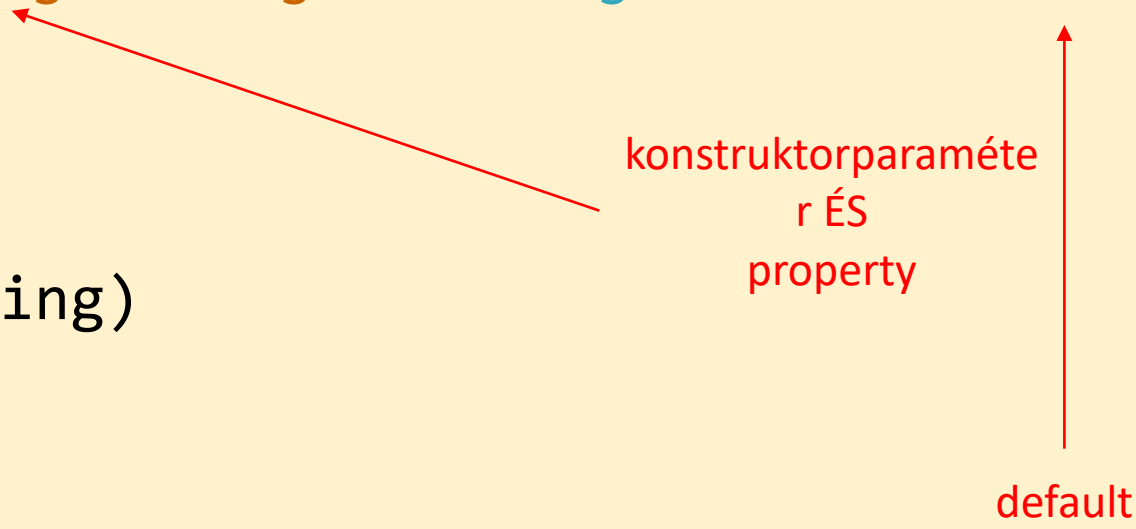
property

metódus

Elsődleges konstruktor és tulajdonságok

```
class Hello(val greeting : String = "Hello world!") {

  fun greet() {
    print(greeting)
  }
}
```



Ősosztály

elsődleges konstruktor paramétere

```
class Material(program : Program)
  : UniformProvider("material") {

  init {
  addComponentsAndGatherUniforms(program);
  }
}
```

ősosztály

ős konstruktorparamétere

inicializáló blokk konstruáláshoz (nem is kell konstruktortörzs)

`var` vs `val`

```
val pos = Vec3()

pos += Vec3(1f, 2f, 3f)
```

ez rendben van,
pos maradt, ami volt,
vagyis referencia ugyanarra az objektumra,
csak a tulajdonságai változtak

lamdba

```
class App(
  val keysPressed = HashSet<String>()
  fun registerEventHandlers() {
    document.onkeydown = {
      event : KeyboardEvent ->
      keysPressed.add( keyNames[event.keyCode] )
    }
  }
}
```

ez egy függvény

paraméterlista

az utolsó kifejezés a visszatérési érték

a tartalmazó scope simán elérhető (zárvány)

Regex

magánhangzóra illeszkedő minta

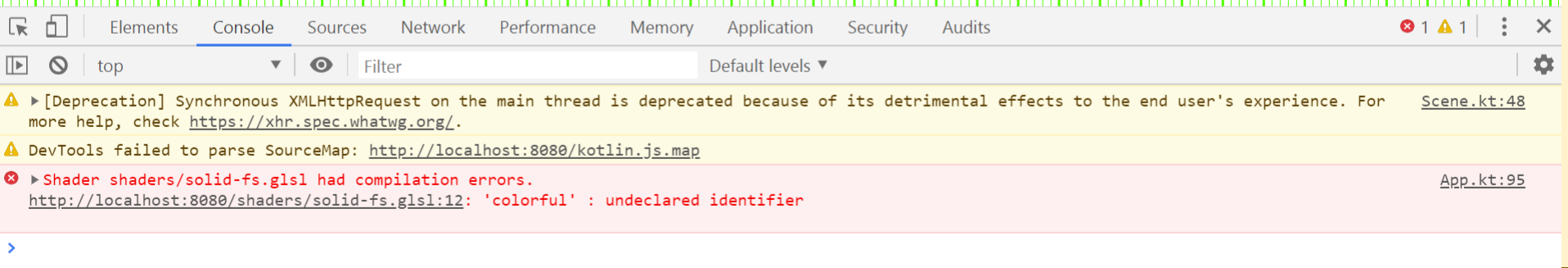
```
val text = "Te tudsz így beszélni?"
val secret =
Regex("[aáeéiíoóöőuúü]").replace(text) {
    it.value + 'v' + it.value  }
```

egyparaméteres lambda automatikus paramétere (még paraméterlista sem kell)

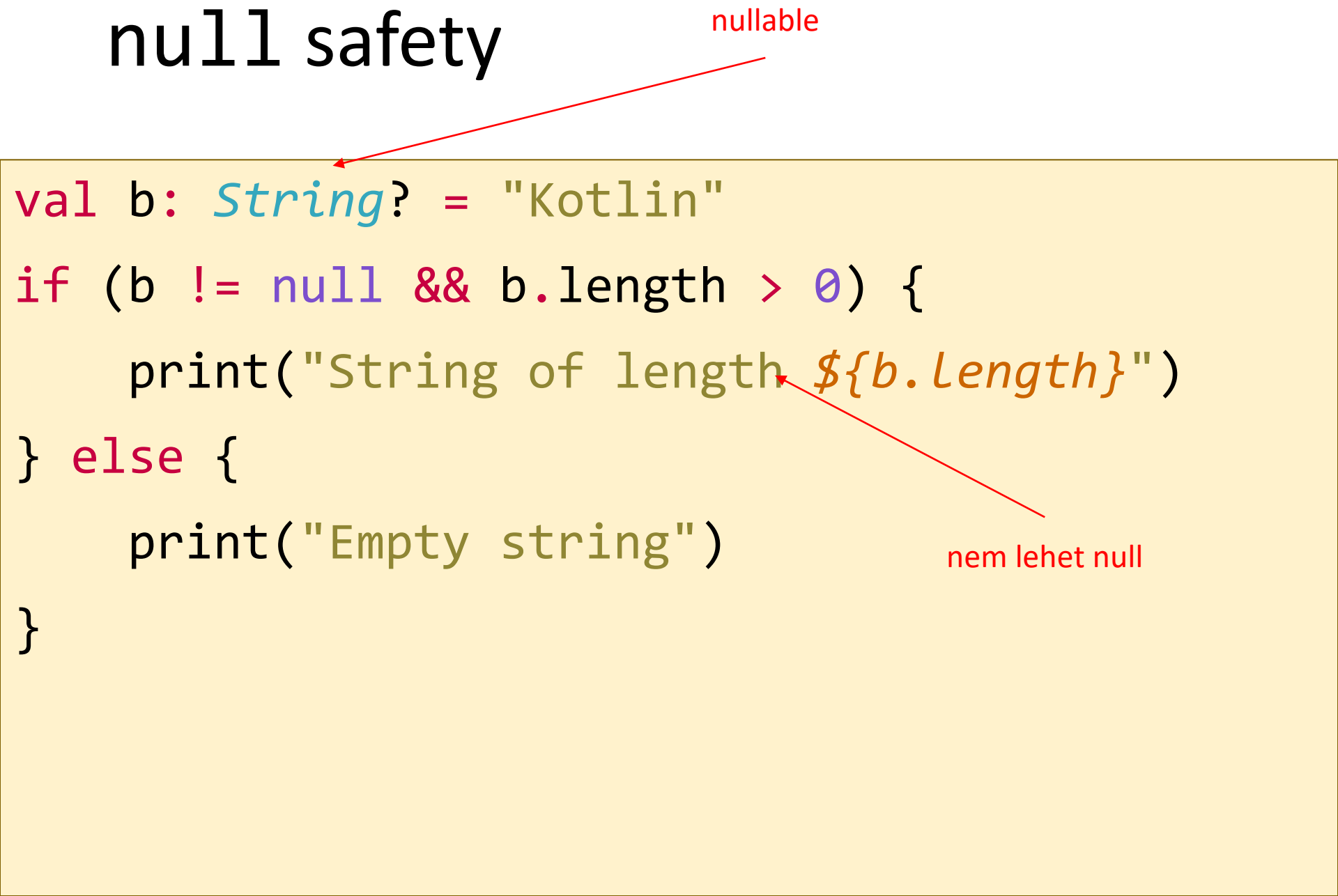
ez itt a replace utolsó paramétere, a zárójelen kívülre lehet rakni

Kivételkezelés, Error

```
throw Error("Could not link shaders [vertex shader:
${vertexShader.sourceUrl}]:[fragment shader:
${fragmentShader.sourceUrl}\n${gl.getProgramInfoLog(this.glProgram)}")
try{
  val app = App(canvas, overlay)
} catch(e : Error) {
  console.error(e.message)
}
```



null safety


nullable
val b: String? = "Kotlin"
if (b != null && b.length > 0) {
print("String of length ${b.length}")
} else {
print("Empty string")
}
nem lehet null

Safe call

Mat4?

```
avatar["modelMatrix"]
    .?scale(scale)
    .?rotate(yaw)
    .?translate(position)
```

ha null volt,
semmit nem csinál,
és az értéke null

Elvis

paramétertömb

```
fun set(vararg values : Float) {
  for(i in 0 until storage.length) {
    storage[i] = values.getOrNull(i) ?: 0.0f
  }
}

val gl = canvas.getContext("webgl2")
   ?: throw Error("Browser does not support
WebGL2")
```

ha null volt,
akkor ez legyen

Tárolók

```
val keysPressed = HashSet<String>()
val uniforms = HashMap<String, Uniform>()
val uniformDescriptors =
  HashMap<String,
  ArrayList<UniformDescriptor> >()
```

forEach, vararg, spread

```
val components = ArrayList<Drawable>()


fun draw(
        vararg uniformProviders : UniformProvider){
  components.forEach {
    it.draw(this, *uniformProviders)
  }
}
```

ez itt a forEach egyetlen paramétere, nem is kell zárójeles paraméterlista

spread operator tömbböl paraméterek

Összetett példa

```
fun drawWithOverrides(
    overrides : Map<String, UniformProvider>,
    vararg uniformProviders : UniformProvider){
  val allOverrides : MutableMap<String, UniformProvider>
          = overrideMap.toMutableMap()
  overrides.forEach {
    (key : String, value : Drawable) ->
    allOverrides[key] = value }
  this@UniformProvider.drawWithOverrides(
                              allOverrides, *uniformProviders)
}
```

bejövő asszociatív tömb

változtatható másolat

destructuring

lambdában a this-t meg kell cimkézni