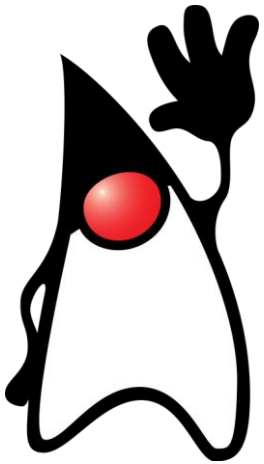
Java-ról Kotlinra



Ekler Péter
peter.ekler@aut.bme.hu
AutSoft
BME AUT

Tartalom

- Java és Kotlin kapcsolata
- Hogyan próbálhatjuk ki?
- Kotlin kultúra kialakítása cégen belül

Milyen a Kotlin a Java-hoz képest?

Történet és tulajdonságok

- 2011-ben jelent meg először
- JetBrains gondozásában
- Nyílt forráskódú nyelv
- 2017-es Google I/O: hivatalos támogatás Androidra
- Statikusan típusos
- Objektum orientáltság mellett a funkcionális programozást is támogatja

A Kotlin főbb jellemzői

- JVM byte kódra (vagy akár JavaScriptre is) fordul
- Meglévő Java API-k, keretrendszerek és könyvtárak használhatók
- Automatikus konverzió Java-ról Kotlinra
- Null-safety
  - Vége a NullPointerException korszaknak
- Kód review továbbra is egyszerű
  - A nyelv alapos ismerete nélkül is olvasható a kód

Java vs. Kotlin

```java
public class Ship {
    private String name;
    private int age;

    public Ship(String name, int age) {
        this.name = name;
        this.age = age;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
    …
}
```

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

Változók `null` értéke

- Alapból a változók értéke nem lehet `null`

```
var a: Int = null
error: null can not be a value of a non-null type Int
```

- A ‘?’ operátorral engedélyezhetjük a null értéket

```
var a: Int? = null
```

  - Lista, melyben lehetnek `null` elemek

```
var x: List<String?> =
  listOf(null, null, null)
```

  - Lista, mely lehet `null`

```
var x: List<String>? = null
```

  - Lista, mely lehet `null` és az elemei is lehetnek `null`-ok

```
var x: List<String?>? = null
x = listOf(null, null, null)
```

“Double bang” operator

- Kivételt dob, ha a változó értéke `null`

```
var x: Int? = null
x!!.toString()
kotlin.KotlinNullPointerException
```

`Null` tesztelés és az Elvis operátor

```
var nullTest : Int? = null
nullTest?.inc()
```

- `inc()` nem hívódik meg, ha `nullTest` `null`

```
var x: Int = 4
var y = x?.toString() ?: ""
```

- ha x null, akkor y “” értéket kap

Függvény Kotlin-ban és Java-ba

```java
public String whatShouldIDoToday(String mood, String weather, Integer temperature){
    if (temperature == null) temperature = 24;
    if ("sad".equals(mood) && "rainy".equals(weather))
        return "Watch a good film";
    if ("sunny".equals(weather) && temperature > 20)
        return "Go and play in the garden!";
    if (temperature < 5)
        return "Drink tea!";
    return "I don't know";
}
```

```kotlin
fun whatShouldIDoToday(mood: String, weather: String, temperature: Int = 24): String {
    return when {
        mood == "sad" && weather == "rainy" -> "Watch a good film"
        weather == "sunny" && temperature > 20 -> "Go and play in the garden!"
        temperature < 5 -> "Drink tea!"
        else -> "I don't know..."
    }
}
```

“Kotlinizáljunk”! - 1. lépés

```
fun whatShouldIDoToday(mood: String, weather: String = "sunny", temperature: Int = 24): String {
    val isSadAndRainy = mood == "sad" && weather == "rainy"
    val isSunnyAndWarm = weather == "sunny" && temperature > 20
    val isCold = temperature < 5
    return when {
        isSadAndRainy -> "Watch a good film"
        isSunnyAndWarm -> "Go and play in the garden!"
        isCold -> "Drink tea!"
        else -> "I don't know..."
    }
}
```

```
fun whatShouldIDoToday(mood: String, weather: String = "sunny", temperature: Int = 24): String {
    return when {
        isSadAndRainy(mood, weather) -> "Watch a good film"
        isSunnyAndWarm(weather, temperature) -> "Go and play in the garden!"
        isCold(temperature) -> "Drink tea!"
        else -> "I don't know..."
    }
}
private fun isSadAndRainy(mood: String, weather: String): Boolean {
    return mood == "sad" && weather == "rainy"
}
private fun isSunnyAndWarm(weather: String, temperature: Int): Boolean{
    return weather == "sunny" && temperature > 20
}
private fun isCold(temperature: Int): Boolean {
    return temperature < 5
}
```

“Kotlinizáljunk”! - 2. lépés

```
fun whatShouldIDoToday(mood: String, weather: String = "sunny", temperature: Int = 24): String {
    return when {
        isSadAndRainy(mood, weather) -> "Watch a good film"
        isSunnyAndWarm(weather, temperature) -> "Go and play in the garden!"
        isCold(temperature) -> "Drink tea!"
        else -> "I don't know..."
    }
}
private fun isSadAndRainy(mood: String, weather: String): Boolean {
    return mood == "sad" && weather == "rainy"
}
private fun isSunnyAndWarm(weather: String, temperature: Int): Boolean{
    return weath
}
private fun isCo
    return tempe
}
```

```
fun whatShouldIDoToday(mood: String, weather: String = "sunny", temperature: Int = 24): String {
    return when {
        isSadAndRainy(mood, weather) -> "Watch a good film"
        isSunnyAndWarm(weather, temperature) -> "Go and play in the garden!"
        isCold(temperature) -> "Drink tea!"
        else -> "I don't know..."
    }
}
private fun isSadAndRainy(mood: String, weather: String):Boolean = mood == "sad" && weather == "rainy"
private fun isSunnyAndWarm(weather: String, temperature: Int):Boolean = weather == "sunny" && temperature > 20
private fun isCold(temperature: Int): Boolean = temperature < 5
```

“Kotlinizáljunk”! - 3. lépés

```
fun whatShouldIDoToday(mood: String, weather: String = "sunny", temperature: Int = 24): String {
    return when {
        isSadAndRainy(mood, weather) -> "Watch a good film"
        isSunnyAndWarm(weather, temperature) -> "Go and play in the garden!"
        isCold(temperature) -> "Drink tea!"
        else -> "I don't know..."
    }
}
private fun isSadAndRainy(mood: String, weather: String): Boolean = mood == "sad" && weather == "rainy"
private fun isSunnyAndWarm(weather: String, temperature: Int): Boolean = weather == "sunny" && temperature > 20
private fun isCold(temperature: Int): Boolean = temperature < 5
```

```
fun whatShouldIDoToday(mood: String, weather: String = "sunny", temperature: Int = 24): String {
    return when {
        isSadAndRainy(mood, weather) -> "Watch a good film"
        isSunnyAndWarm(weather, temperature) -> "Go and play in the garden!"
        isCold(temperature) -> "Drink tea!"
        else -> "I don't know..."
    }
}
private fun isSadAndRainy(mood: String, weather: String) = mood == "sad" && weather == "rainy"
private fun isSunnyAndWarm(weather: String, temperature: Int) = weather == "sunny" &&
  temperature > 20
private fun isCold(temperature: Int) = temperature < 5
```

“Kotlinizáljunk”! - 4. lépés

```
fun whatShouldIDoToday(mood: String, weather: String = "sunny", temperature: Int = 24): String {
    return when {
        isSadAndRainy(mood, weather) -> "Watch a good film"
        isSunnyAndWarm(weather, temperature) -> "Go and play in the garden!"
        isCold(temperature) -> "Drink tea!"
        else -> "I don't know..."
    }
}
```

```
fun whatShouldIDoToday(mood: String, weather: String = "sunny", temperature: Int =
getDefaultTemperature()): String {
    return when {
        isSadAndRainy(mood, weather) -> "Watch a good film"
        isSunnyAndWarm(weather, temperature) -> "Go and play in the garden!"
        isCold(temperature) -> "Drink tea!"
        else -> "I don't know..."
    }
}

fun getDefaultTemperature() = 24
```

Extension Methods

- Új funkció hozzáadása egy osztályhoz anélkül, hogy leszármaztatnánk belőle

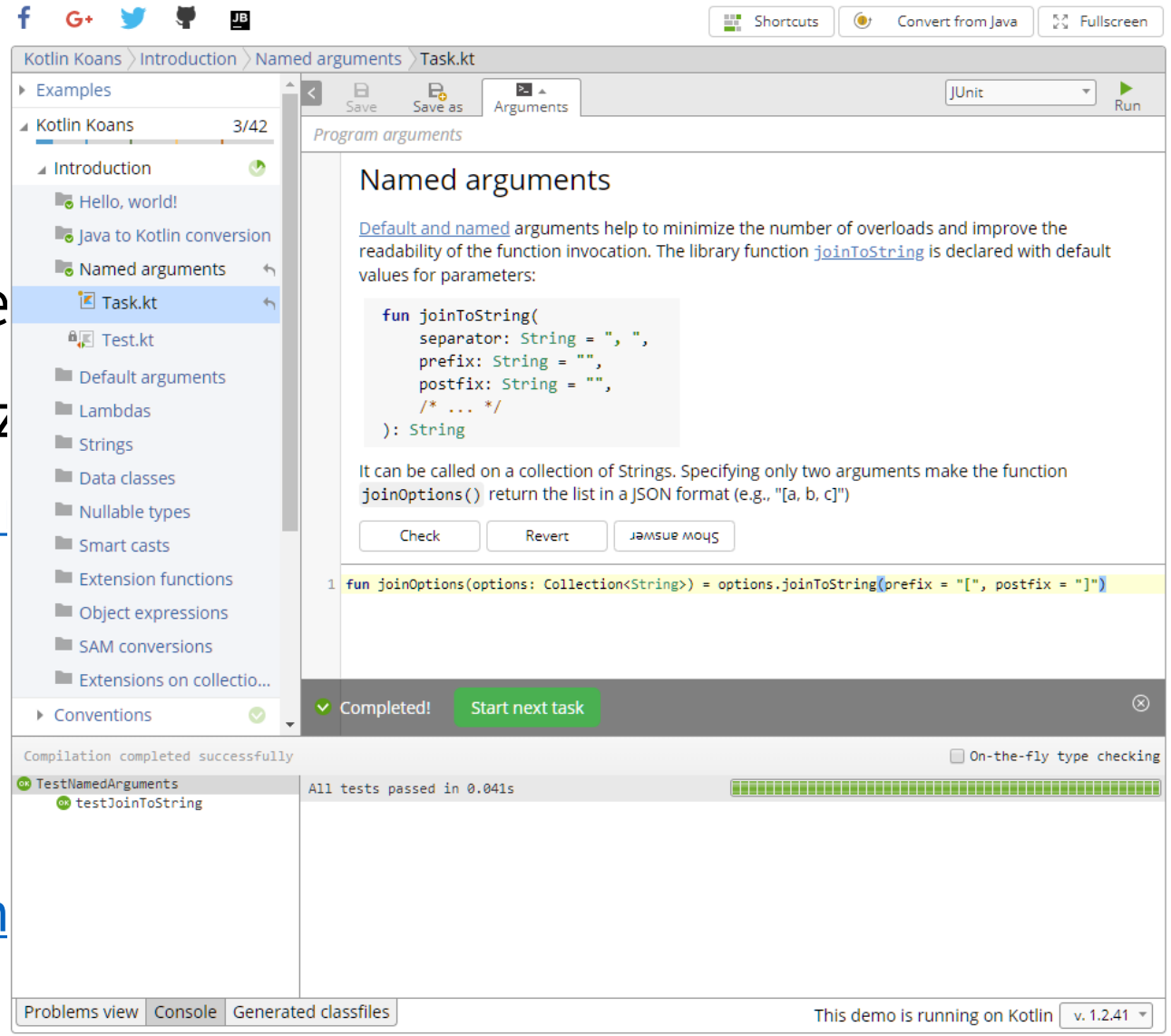
```
fun ImageView.loadUrl(url: String) {
    Picasso.with(context).load(url).into(this)
}
```

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

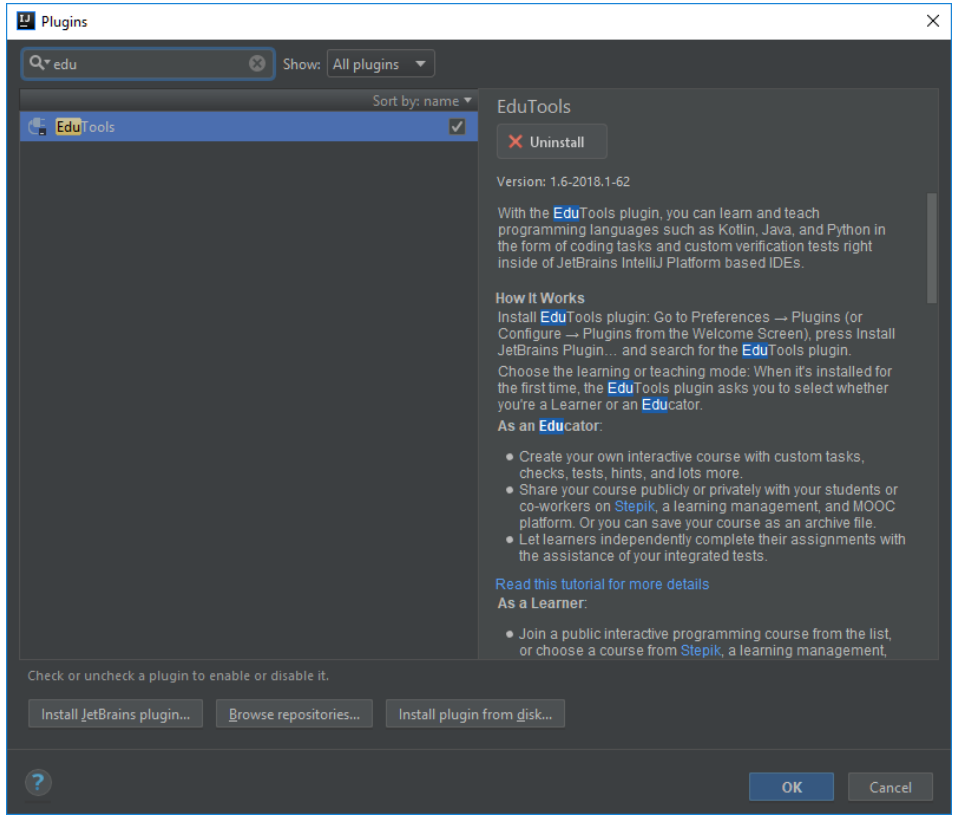
Hogyan próbálhatom ki?

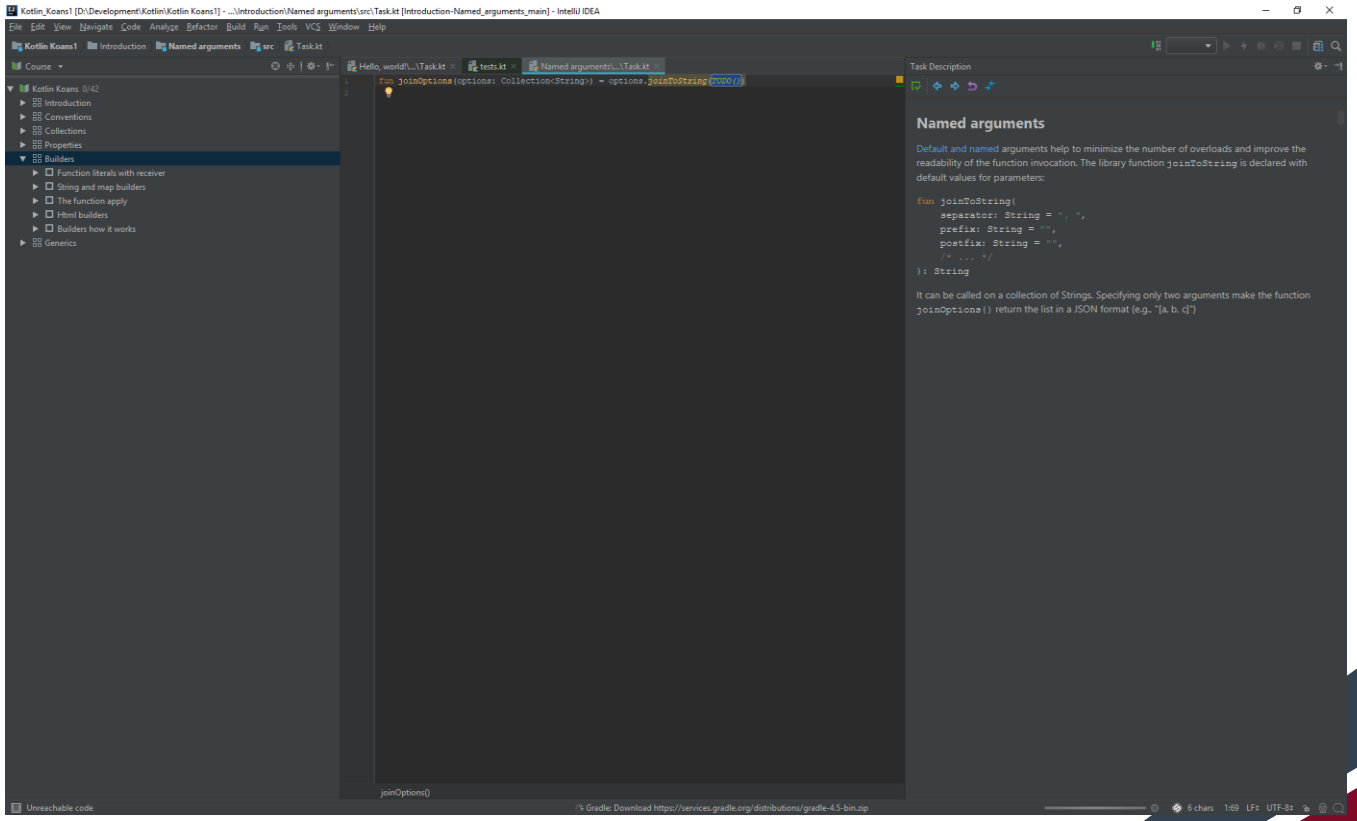
Kotlin Koans

- Gyakorló kódrészek Unit teszte
- IDEA és Android Studio környez
  - https://kotlinlang.org/docs/tuto
- Online tutorial
  - https://try.kotlinlang.org
- GitHub
  - https://github.com/Kotlin/kotlin



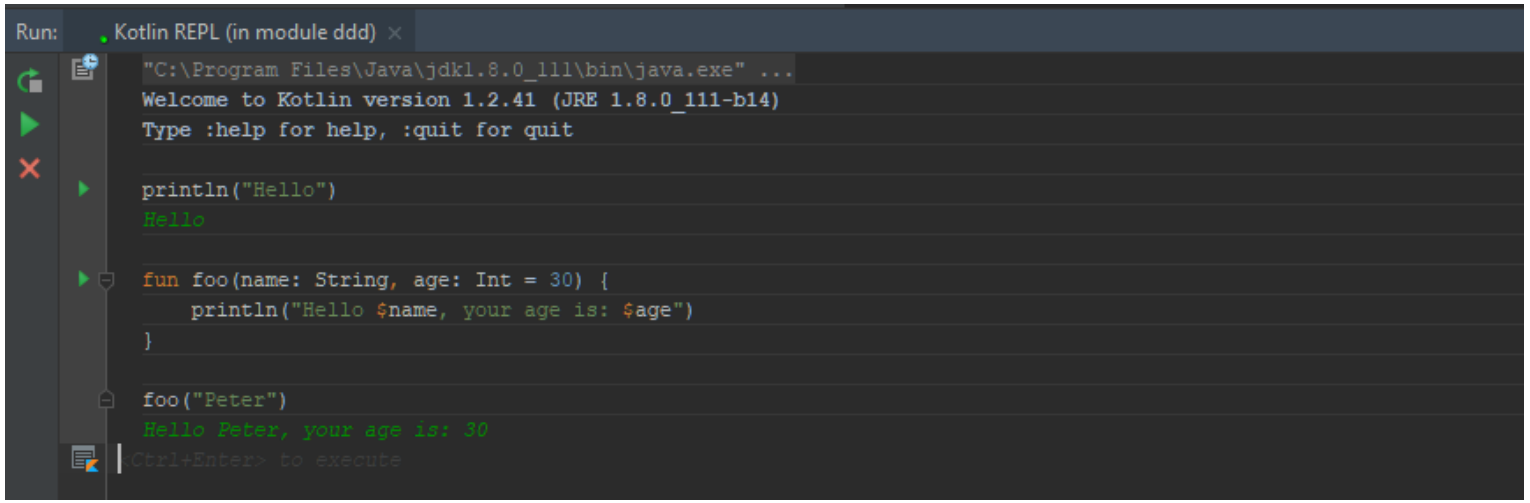
EduTools Plugin





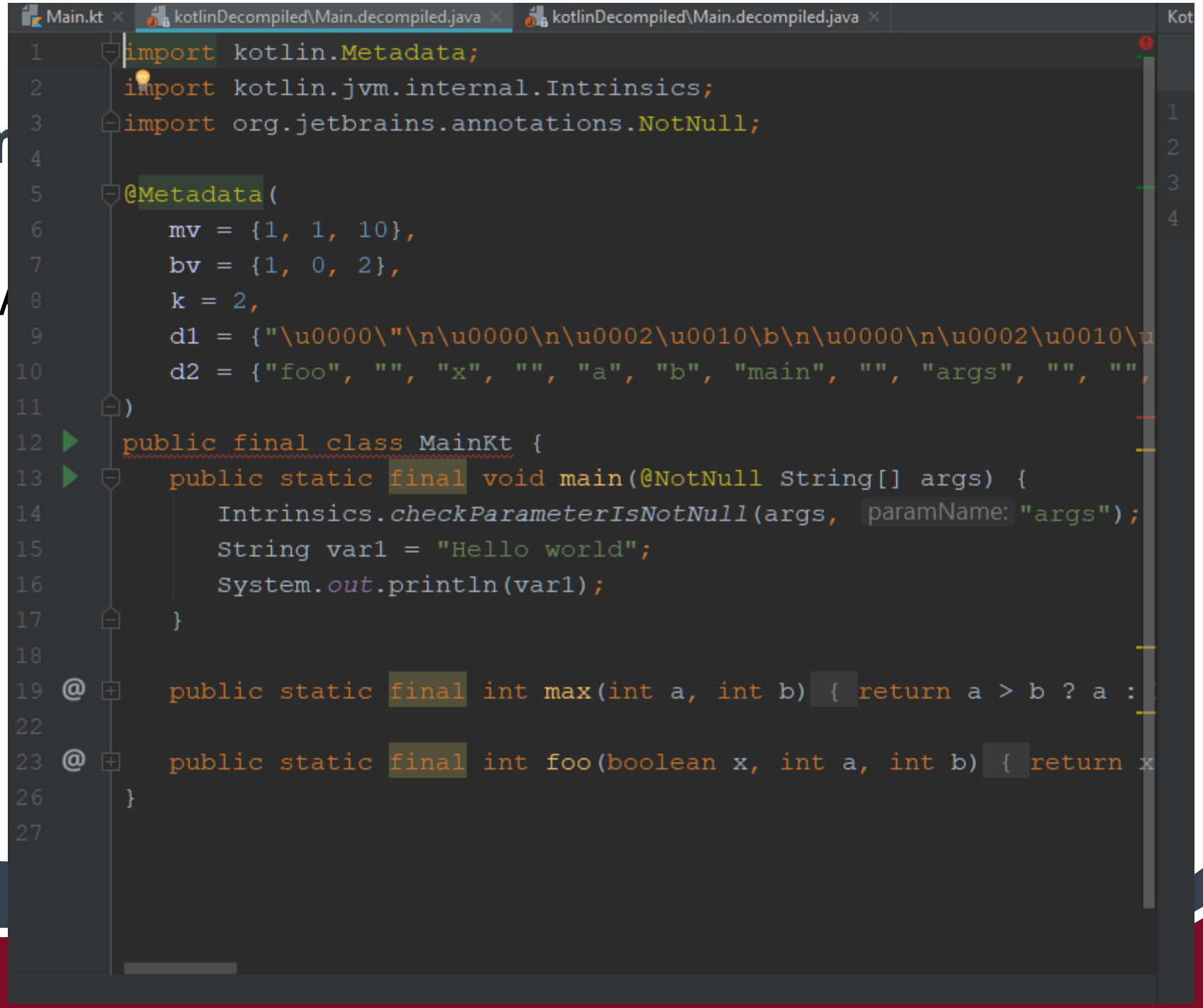
Idea környezetben

- Egyszerűen létrehozható új projekt
- IDEA REPL tool
  - Parancssorból is futtatható



Kotlin

- IDE

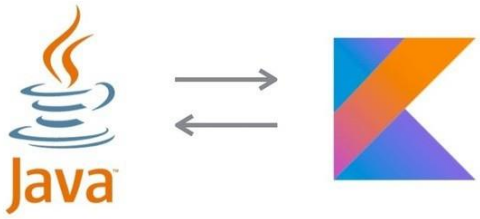

Main.kt
kotlinDecompiled\Main.decompiled.java
kotlinDecompiled\Main.decompiled.java
import kotlin.Metadata;
import kotlin.jvm.internal.Intrinsics;
import org.jetbrains.annotations.NotNull;
@Metadata(
mv = {1, 1, 10},
bv = {1, 0, 2},
k = 2,
d1 = {"\u0000\"\n\u0000\n\u0002\u0010\b\n\u0000\n\u0002\u0010\u
d2 = {"foo", "", "x", "", "a", "b", "main", "", "args", "", "",
)
public final class MainKt {
public static final void main(@NotNull String[] args) {
Intrinsics.checkParameterIsNotNull(args, paramName: "args");
String var1 = "Hello world";
System.out.println(var1);
}
public static final int max(int a, int b) { return a > b ? a :
public static final int foo(boolean x, int a, int b) { return x
}

További hasznos helyek

- Könyvek:
  - Kotlin in Action: By Dmitry Jemerov and Svetlana Isakova, Kotlin developers at JetBrains.
  - Kotlin for Android Developers: By Antonio Leiva. One of the first books about Kotlin.
  - Android Development with Kotlin: By Marcin Moskala and Igor Wojda.

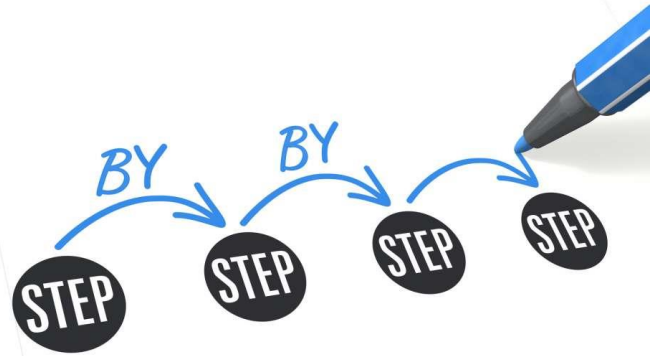
Kotlin kultúra cégen belül

Kotlin bevezetése



- Előnyök bemutatása
  - Produktivitás növelése (ki szerint számít a gépelés sebessége?)
  - Kevesebb hibalehetőség
  - Motivált fejlesztők
  - Vonzó pozíciók
- Tisztázni az alacsony kockázatot
  - Java tudás teljes mértékben felhasználható
  - Egyszerű és könnyen tanulható nyelv (~)

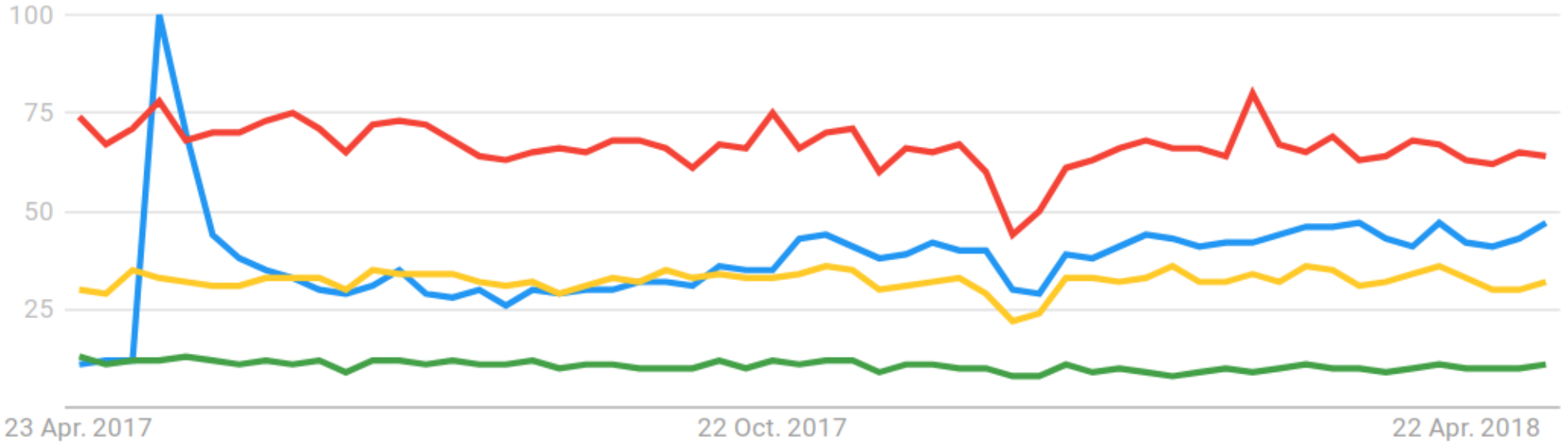
Kotlin bevezetése

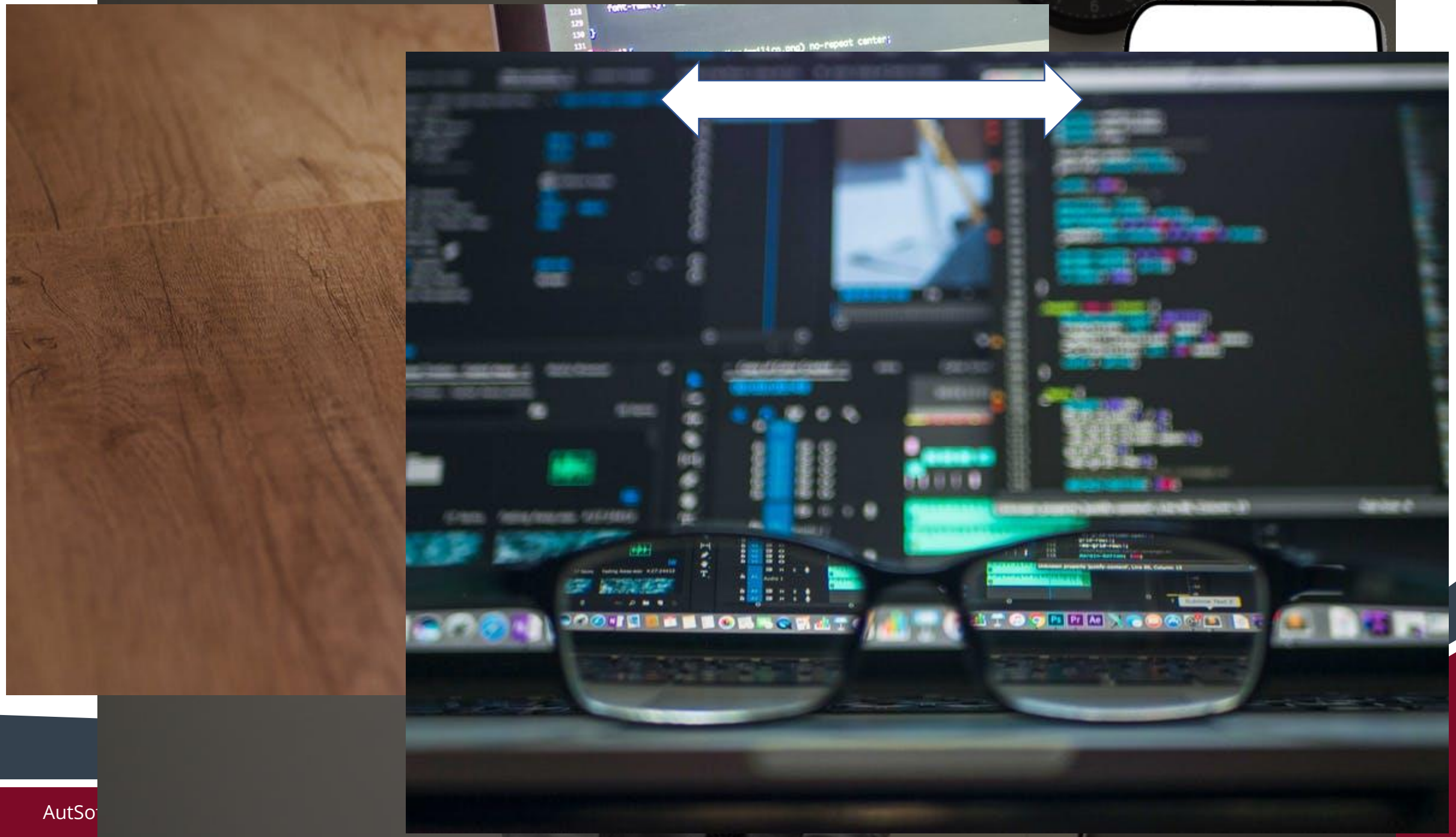


- Esetleges hátrányok
  - JetBrains-től függ a továbbfejlesztés
  - Alacsony támogatás Eclipse és NetBeans környezetben
- Bevezetés lehetősége kis lépésekben
  - Kezdjünk kis kódbázison, alacsony üzleti érdekeltségű modulban
  - Prototípusok, belső eszközök
  - Kis kompromisszumokra egyszerűbb jóváhagyást szerezni☺
  - Referencia után már könnyebb a használat kiterjesztése

Kotlin terjedése

- Forrás: Google Trends





Köszönöm a figyelmet!

Ekler Péter

peter.ekler@aut.bme.hu

AutSoft

BME AUT

Források:
https://kotlinlang.org/
https://www.xenonstack.com/blog/updates/overview-kotlin-comparison-kotlin-java/
https://blog.philipphauer.de/convincing-management-introduce-kotlin/