Kotlin Workshop

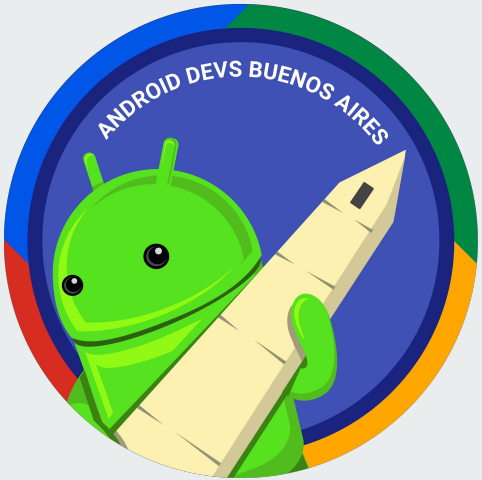
Hagamos una app Android!

NERDEAR.LA

Android Devs Buenos Aires

twitter.com/AndroidDevsBsAs

meetup.com/es-ES/Android-Devs-Buenos-Aires



Agenda

- ¿Qué es Kotlin?
- ¿Por qué usar Kotlin?
- Sintaxis básica
- Variables / Inferencia de tipos / Nullability
- Funciones
- Clases
- Expresiones
- Kotlin Android Extensions
- Extension Functions
- Hagamos la app

¿Qué es Kotlin?

- Kotlin es un lenguaje de programación estáticamente tipado que corre sobre la JVM
- Fue desarrollado por JetBrains
- Su nombre viene de Kotlin Island, al igual que Java, que fue nombrado así por island of Java
- Viene siendo desarrollado desde el 2010
- En febrero del 2016 salió su primer versión estable (1.0)
- En el Google I/O del 2017 lo anuncian como lenguaje soportado oficialmente para el desarrollo de aplicaciones Android
- Se puede utilizar para el desarrollo Android, backend, y frontend ya que también compila a JavaScript

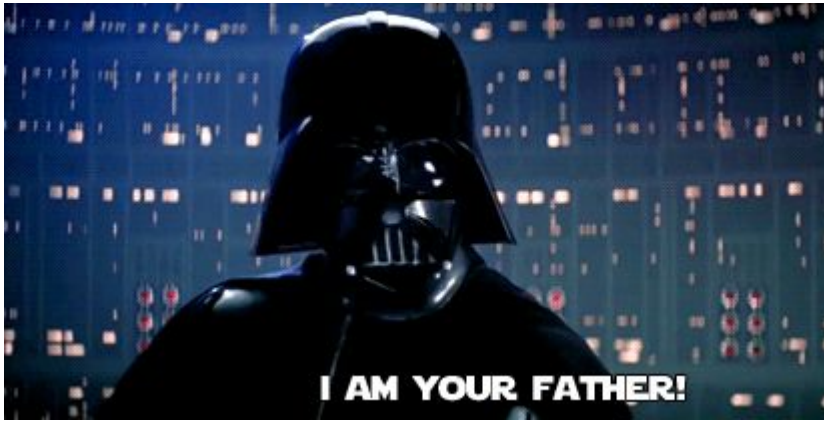
¿Por qué usar Kotlin?

- Total interoperabilidad con Java
- Reducción de código repetitivo
- Seguridad a la hora de programar (Nullability)
- Curva de aprendizaje baja
- Respaldado por JetBrains + Google
- Características de programación funcional
- Corre en cualquier lugar donde corra Java
- Sintaxis moderna, menos verboso
- Google está trabajando mucho para que sea el lenguaje principal de Android
- Magia

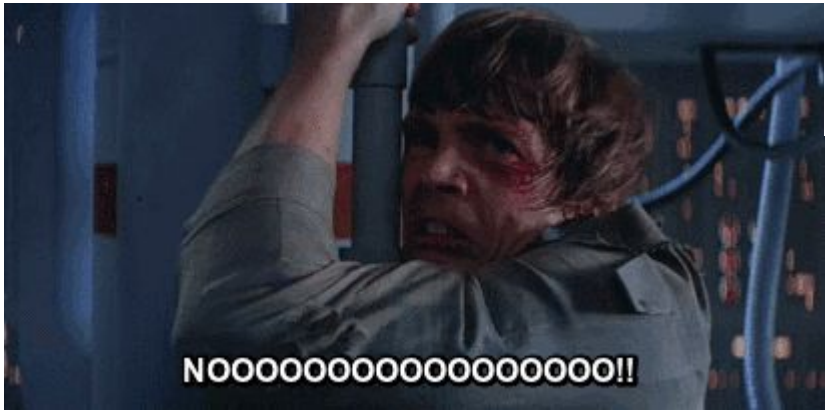
¿Y Java?

(Dijo nadie, nunca)

Java



Kotlin



Sintaxis básica - Variables

En Kotlin tenemos dos formas de declarar una variable: Mutable o Inmutable.

Mutable *(var)*

Podemos modificar su valor.

Inmutable *(val)*

Una vez declarada, no podemos modificar su valor.

Sintaxis básica - Variables

Java

```java
public int numberMutable = 7;
public final int numberImmutable = 8;
```

Kotlin

```kotlin
var numberMutable: Int = 7
val numberImmutable: Int = 8
```

Sintaxis básica - Variables, inferencia de tipos

Declaro el tipo de valor

```
var numberMutable: Int = 7
val numberImmutable: Int = 8
```

Infiero el tipo de valor

```
var numberMutable = 7
val numberImmutable = 8
```

Sintaxis básica - Nullability

Java

```
public Integer numberMutable;
```

(Es null hasta que se le asigne un valor)

Kotlin

```
var numberMutable: Int? = null
```

(? indica que puede ser null)

Funciones

Java

```java
public int addFive(int number) {
    return number + 5;
}
```

Kotlin

```kotlin
fun addFive(number: Int): Int {
    return number + 5
}
```

fun

Funciones - Single-expression functions

Regular function

```
fun addFive(number: Int): Int {
    return number + 5
}
```

Single-expression function

```
fun addFive(number: Int) = number + 5
```

Clases

Java

```java
public class Person {

    private String name;
    private String surname;

    public Person(String name, String surname) {
        this.name = name;
        this.surname = surname;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getSurname() {
        return surname;
    }

    public void setSurname(String surname) {
        this.surname = surname;
    }
}
```

26 líneas de código

Clases

Kotlin

```kotlin
class Person(val name: String, val surname: String)
```

1 línea de código

Clases

Kotlin

```kotlin
class Person(val name: String, val surname: String)

val person = Person("Mariana", "Lopez")
person.name // Mariana
person.surname // Lopez
```

Clases

Kotlin

```kotlin
class Person(val name: String, val surname: String) {

    fun getFullName(): String {
        return "$name $surname"
    }
}
```

Clases - Singleton

object

```
object ApiClient {
    val githubApi: GitHubApi
    // ...
}

// Acceso
val githubApi = ApiClient.githubApi
```

Clases - Herencia / Implementar interface

```
:

class MainActivity :
        AppCompatActivity(),
        Callback<List<GitHubRepo>> {
    // ...
}
```

Clases - Data Class

Este tipo de clase es muy útil para trabajar con modelos de datos

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

Automáticamente nos habilita las funciones:

- **equals()**
- **hashCode()**
- **toString()**
- **copy()**

Clases - Interface

Similares a Java 8

```
interface GitHubApi {

    @GET("repositories")
    fun getPublicRepositories(): Call<List<GitHubRepo>>

    fun getBaseUrl(): String {
        return "https://api.github.com"
    }
}
```

Clases - Companion Object

*En Kotlin no existe el modificador **static***

```
class ArticlesFragment : Fragment() {
    // ...
    companion object {
        const val TAG = "ArticlesFragment"
        fun newInstance(): ArticlesFragment {
            return ArticlesFragment()
        }
    }
}
```

Clases - Companion Object

*En Kotlin no existe el modificador **static***

```
val fragment = ArticlesFragment.newInstance()
val tag = ArticlesFragment.TAG
```

Desde Java

```
public ArticlesFragment fragment =
                        ArticlesFragment.Companion.newInstance()
public String tag = ArticlesFragment.TAG
```

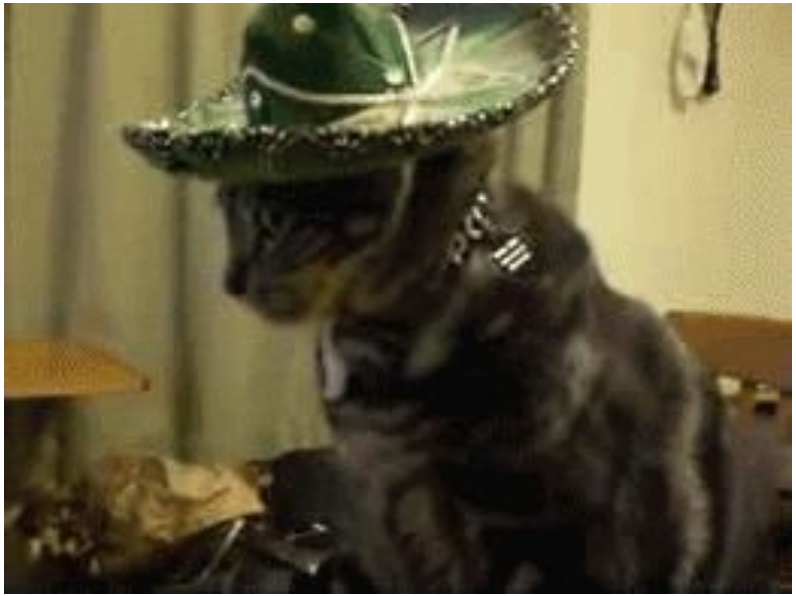
Expresiones: if

En Kotlin if es una expresión, y devuelve un valor

```
fun isOne(): String {
    return if (1 == 1) "Es uno" else "No es uno"
}
```

Expresiones: when

When? KE



Expresiones: when

When. Porque en kotlin, no existe el switch

```
when (x) {
    1 -> print("x == 1")
    2 -> print("x == 2")
    else -> print("x is neither 1 nor 2")
}
```

Control flow: for

```
for (item in collection) print(item)


for (item: Int in ints) {
    // ...
}
```

Kotlin Android Extensions

```
// Using R.layout.activity_main from the 'main' source set
import kotlinx.android.synthetic.main.activity_main.*

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

        // Instead of findViewById<TextView>(R.id.textView)
        textView.setText("Hello, world!")
    }
}
```

Extension Functions

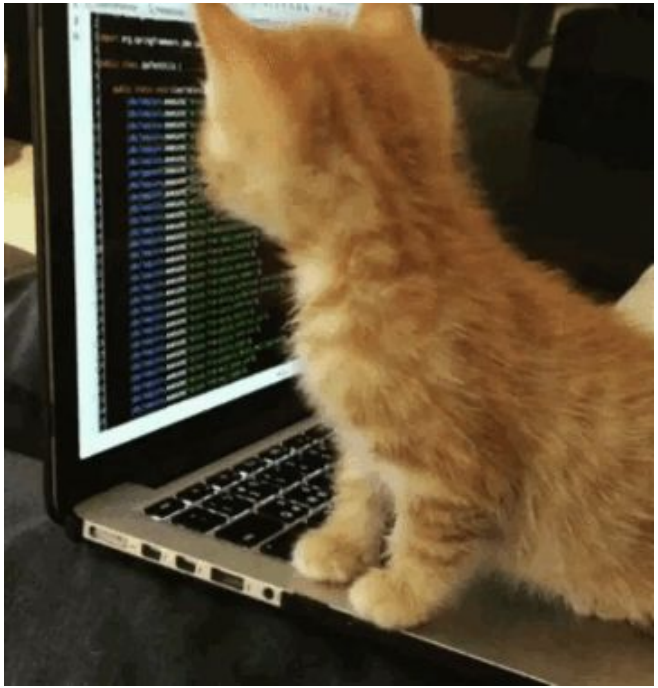
```
fun String.bePolite(): String {
    return "$this, por favor"
}

val order = "Dame un chocolate"
order.bePolite()
// Dame un chocolate, por favor
```

Extension Functions

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

Hagamos la app!



Recursos

GitHubRepos (ésta es la app que vamos a hacer)
https://github.com/ezanetta/githubrepos

Noticiapp https://github.com/ezanetta/noticiapp
Kotlin Docs http://kotlinlang.org
Ejemplos de Kotlin
https://github.com/ezanetta/KotlinExamples/tree/master/src/main/kotlin
Facu https://twitter.com/facundomr
Eze https://twitter.com/zanettapp

Slides

bit.ly/nerdearlaKotlin