



Introdução ao desenvolvimento Android com Kotlin

Ricardo Robaina

Tchelinux Bagé

5 de Outubro de 2019



Sobre o palestrante



Ricardo Robaina

- 📍 Bagé
- 🎓 Engenheiro de Computação (Unipampa)
- 🎓 Mestrando em Computação Aplicada (PPGCAP)
- 💡 Voluntário do Tchelinux

Slides & Arquivos

🔗 <https://github.com/robainaricardo/talks>



Sumário

1. Por que aprender isso?
2. Uma linguagem de programação legal chamada Kotlin
3. Desenvolvimento Android
4. Android + Kotlin, como começar
5. Próximos Passos



População mundial

Earth / Population

7.53 billion (2017)



Explore more

Sources include: World Bank

Feedback



"População Android"



Android ✅
@Android

▼

10 years and now over 2.5 billion active devices.
Thanks for joining us on this journey. #io19



2:58 PM · May 7, 2019 · Sprinklr



Android



O Android é o sistema operacional para dispositivos móveis mais conhecido do mundo. Ele está presente em bilhões de dispositivos, desde smartphones até relógios, tablets, TVs e muito mais.



Formas de desenvolver uma aplicação móvel

- Aplicação web
- Desenvolvimento híbrido
 - Flutter
 - Ionic
 - Xamarin
- Desenvolvimento nativo
 - **Android nativo**
 - IOS nativo



Benefícios em desenvolver aplicativos nativos

- Desempenho
- Tamanho do app (download, armazenamento)
- Acesso aos componentes de hardware (bateria, sensores, etc..)
- Experiência do usuário (UX)



Kotlin

- JetBrains (2010)
- Open Source
- Inspirada em linguagens como: Java, C, JavaScript
- Multiparadigma:
 - OO
 - Funcional
- Multiplataforma



Android & Kotlin

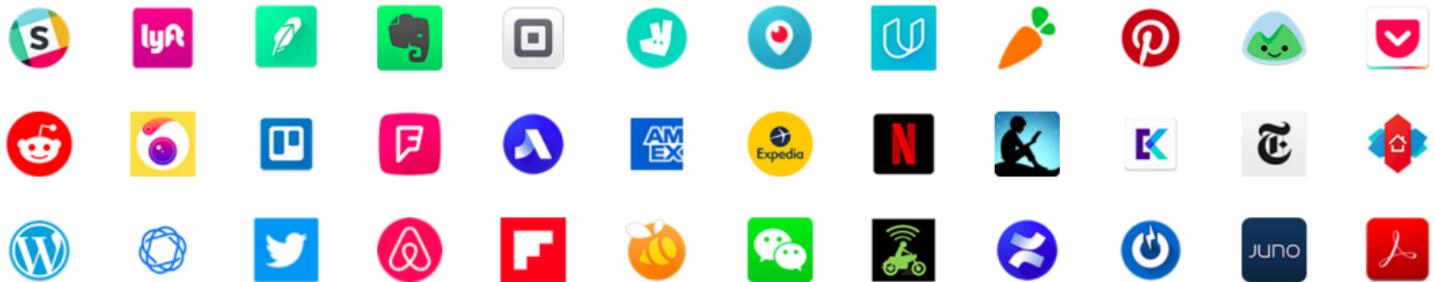
- Comunidade
- Google I/O 2017 - linguagem com suporte oficial para desenvolvimento Android



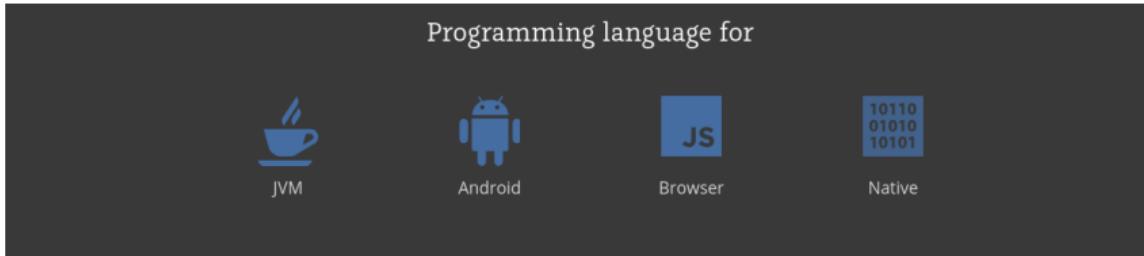
Kotlin é uma realidade

Apps criados com o Kotlin

Muitos apps já são criados com o Kotlin, tanto em startups de destaque quanto em empresas da Fortune 500.



Por que Kotlin?



Why Kotlin?



Concise

Drastically reduce the amount of boilerplate code.

[See example](#)



Safe

Avoid entire classes of errors such as null pointer exceptions.

[See example](#)



Interoperable

Leverage existing libraries for the JVM, Android, and the browser.

[See example](#)



Tool-friendly

Choose any Java IDE or build from the command line.

[See example](#)



Características

- Estaticamente tipada
- Expressiva
- **Null Safety**
- Lambdas, Data classes
- Interpolação de strings, parâmetros default
- **Extension functions**



Extension functions

```
fun main() {
    var x = 15.Metade()
    print(x)
}

fun Int.Metade(): Int{
    return this/2
}
```



Classe Java

```
public class AlunoJava {  
  
    private int matricula;  
    private String nome;  
    private String curso;  
  
    public AlunoJava(int matricula, String nome, String curso) {  
        this.matricula = matricula;  
        this.nome = nome;  
        this.curso = curso;  
    }  
  
    public int getMatricula() {  
        return matricula;  
    }  
    //...  
    public void setMatricula(int matricula) {  
        this.matricula = matricula;  
    }  
}
```



Data classe Kotlin

```
class Aluno(var matricula: Int, var nome: String?, var curso: String?)
```

kotlin



Interoperabilidade Java/Kotlin

```
fun main(args: Array<String>) {  
  
    var joao = AlunoJava(190999, "João da Silva", "Eng. Comp")  
    var ricardo = Aluno(nome = "Ricardo Robaina", curso = "PPGCAP", matricula = 131151690)  
  
    print("""  
        ${joao.nome}  
        ${ricardo.nome}  
    """.trimIndent())  
}
```

kotlin



Poder debaixo dos panos

```
$ javap -c Aluno.class
|
Compiled from "Aluno.kt"
public final class Aluno {
    public final int getMatricula();
        Code:
            0: aload_0
            1: getfield      #10           // Field matricula:I
            4: ireturn

    public final void setMatricula(int);
        Code:
            0: aload_0
            1: iload_1
            2: putfield      #10           // Field matricula:I
            5: return

    public final java.lang.String getNome();
        Code:
            0: aload_0
            1: getfield      #22           // Field nome:Ljava/lang/String;
            4: areturn
```



Poder debaixo dos panos

```
public final void setCurso(java.lang.String);
Code:
  0: aload_0
  1: aload_1
  2: putfield      #28          // Field curso:Ljava/lang/String;
  5: return

public Aluno(int, java.lang.String, java.lang.String);
Code:
  0: aload_0
  1: invokespecial #34          // Method java/lang/Object."<init>":()V
  4: aload_0
  5: iload_1
  6: putfield      #10          // Field matricula:I
  9: aload_0
 10: aload_2
 11: putfield      #22          // Field nome:Ljava/lang/String;
 14: aload_0
 15: aload_3
 16: putfield      #28          // Field curso:Ljava/lang/String;
 19: return
```



}

Android Studio

android studio

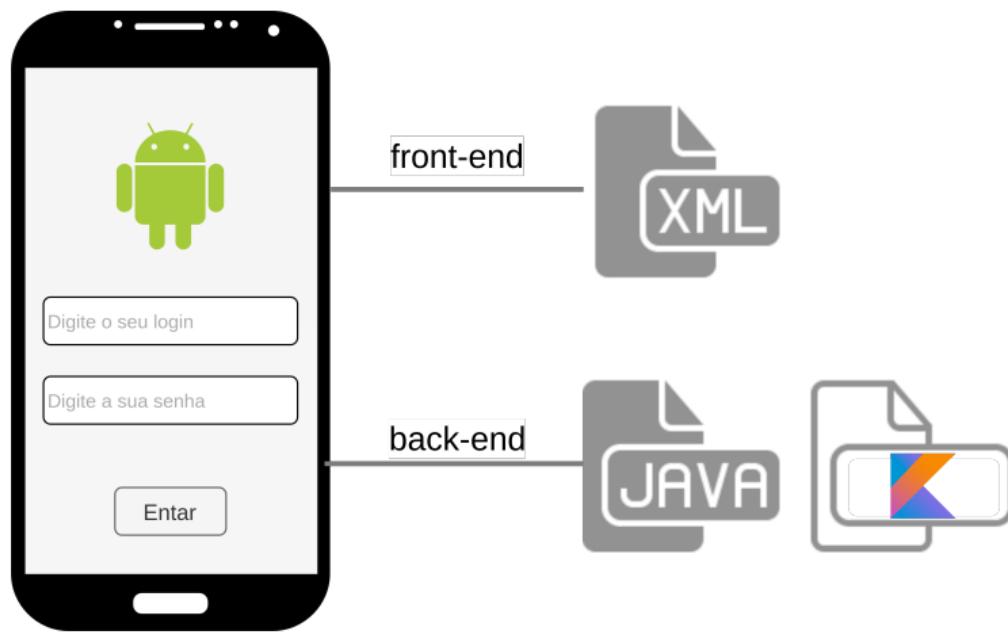
- IDE Oficial
- Debugging
- Teste: dispositivo virtual (VM) & dispositivo real
- Alto consumo de memória



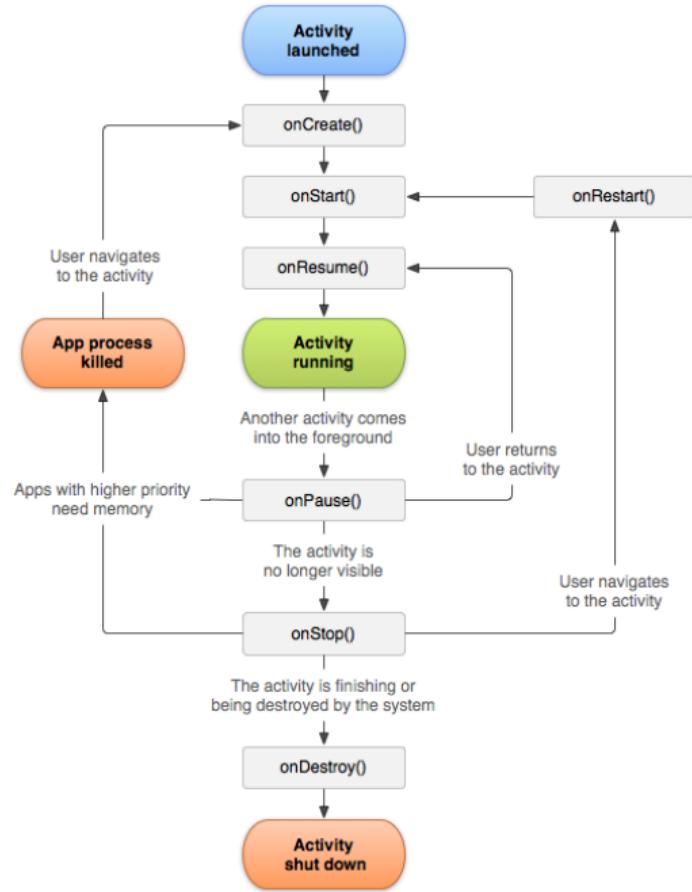
Android Studio



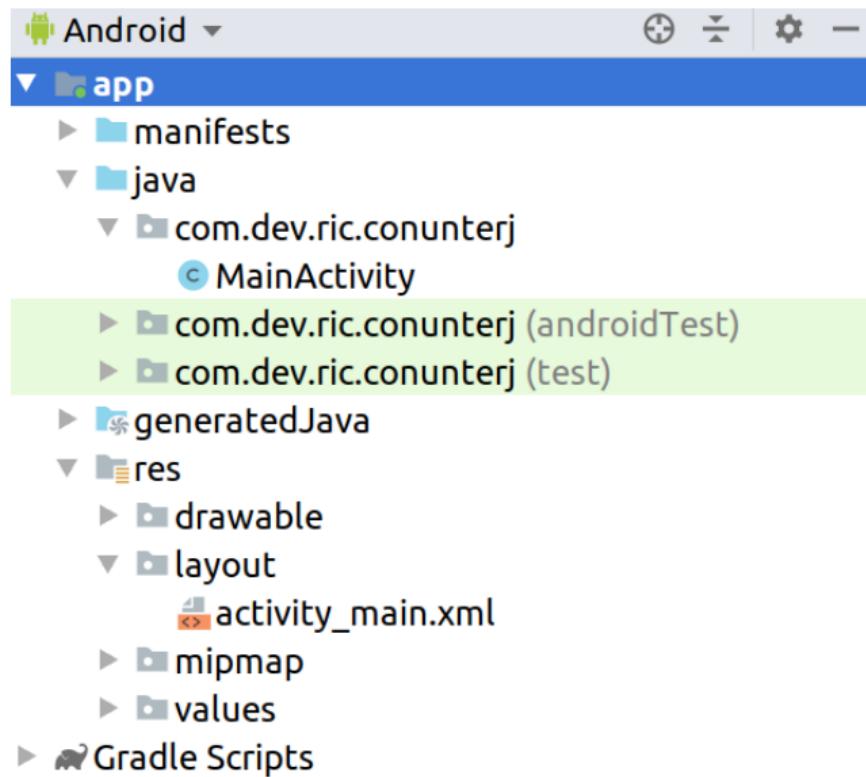
Activity



Ciclo de vida da Activity

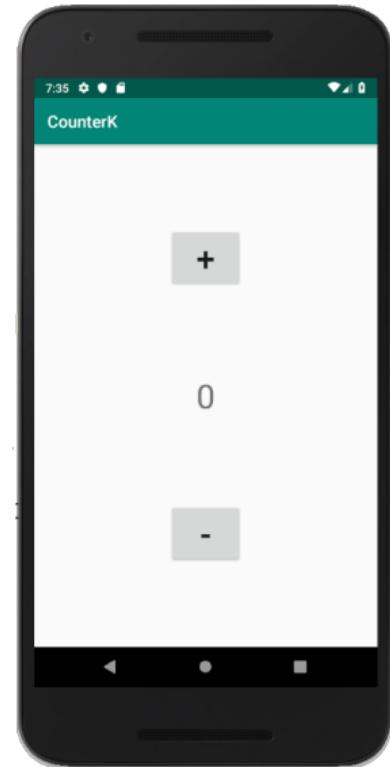


Ambiente de desenvolvimento

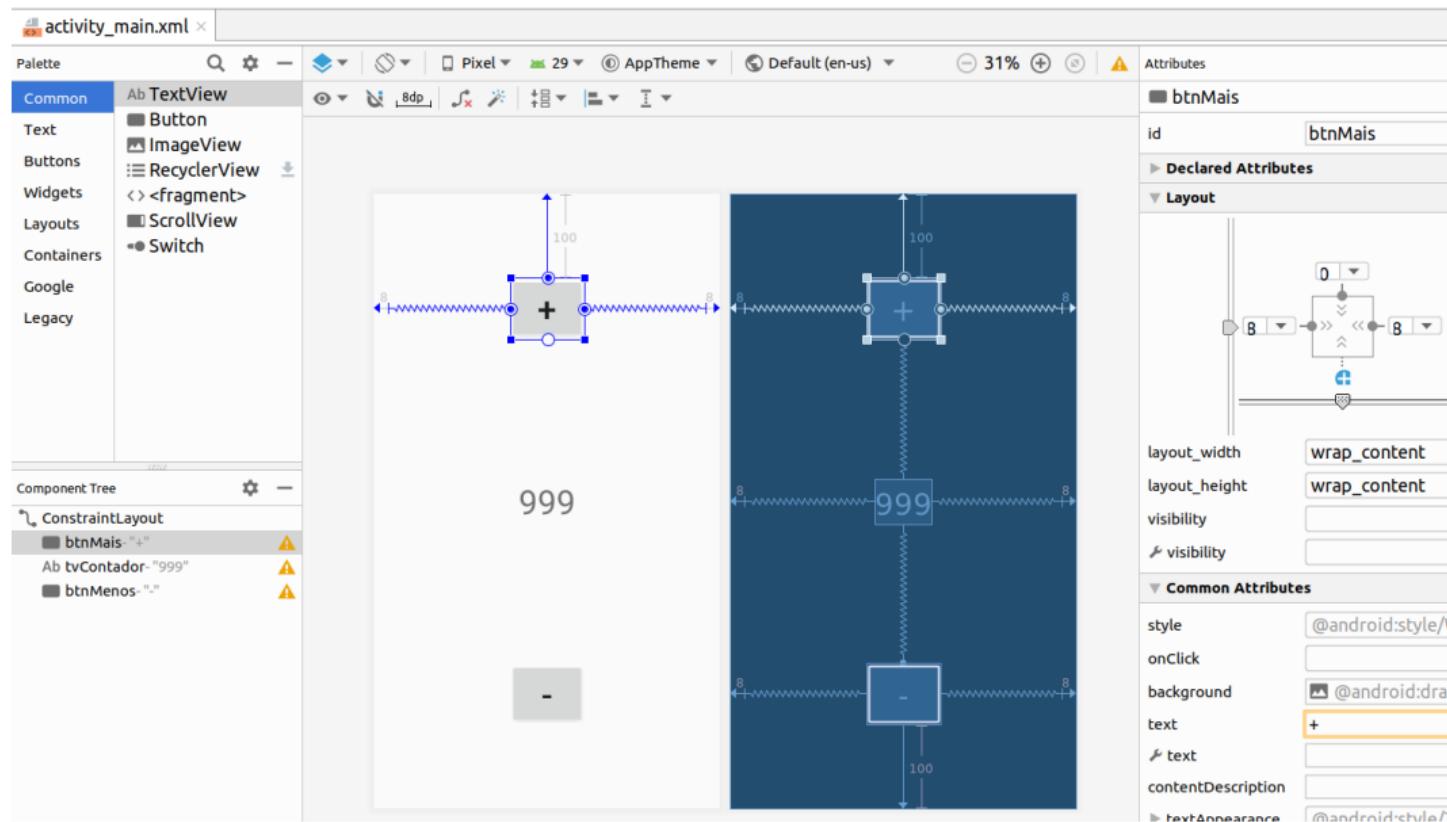


Contador

- Exemplo Simples
- Acesso a elementos de interface
- Comparação com Java



Criando a interface



XML

```
<?xml version="1.0" encoding="utf-8"?>
<android.support.constraint.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
        xmlns:app="http://schemas.android.com/apk/res-auto"
        xmlns:tools="http://schemas.android.com/tools"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        tools:context=".MainActivity">

    <Button
        android:id="@+id	btnMais"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginLeft="8dp"
        android:layout_marginTop="100dp"
        android:layout_marginEnd="8dp"
        android:layout_marginRight="8dp"
        android:text="+"
        android:textSize="40sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    /* ... */

</android.support.constraint.ConstraintLayout>
```



Código Java

```
package com.dev.ric.conunterj;

import /*Dependências*/

public class MainActivity extends AppCompatActivity {

    //Declaração dos elementos de UI
    private Button btnMais, btnMenos;
    private TextView tvContador;
    private int valorContador;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        //Recuperando referencia para os elementos de UI
        btnMais = (Button) findViewById(R.id.btnMais);
        btnMenos = (Button) findViewById(R.id.btnMenos);
        tvContador = (TextView) findViewById(R.id.tvContador);
```



Código Java

```
valorContador = 0;

//Inicialização do contador
atualizarContador();

//Evento de click no botão mais
btnMais.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        incrementarContador();
    }
});
/*...Evento do botão menos...*/
}
/*...Métodos...*/
}
```

java



Código Kotlin

```
package com.dev.ric.counterk

import /*Dependências*/

class MainActivity : AppCompatActivity() {

    //Variável contador
    var contador = 0;

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

        //Inicialização do contador
        atualizarContador()

        //Eventos de click
        btnMais.setOnClickListener { incrementarContador() } //evento botão mais

        btnMenos.setOnClickListener { decrementarContador() } //evento botão menos
    }
    /*...Métodos...*/
}
```



Métodos que vamos precisar

```
/*...Métodos*/\n\n    //Método que atualiza o valor do contador\n    fun atualizarContador(){\n        tvContador.setText("$contador")\n    }\n\n    //Método que incrementa e atualiza o valor do contador\n    fun incrementarContador(){\n        contador++\n        atualizarContador()\n    }\n\n    //Método que decrementa e atualiza o valor do contador\n    fun decrementarContador(){\n        if(contador > 0){\n            contador--\n            atualizarContador()\n        }else{\n            Toast.makeText(this, "O contador já está zerado", Toast.LENGTH_LONG).show()\n        }\n    }\n}
```



Próximos Passos

- Instale o Android Studio
- Google Developers
- Kotlin
- Estude o projeto
- Crie o seu próprio projeto
- Material Design
- Firebase



Próximos Passos

- Instale o Android Studio
- Google Developers
- Kotlin
- Estude o projeto
- Crie o seu próprio projeto
- Material Design
- Firebase
- Junte-se ao **Tchelinux**
- **Compartilhe o seu conhecimento!**



Onde me encontrar

Obrigado pela sua atenção!



Ricardo Robaina

- ricardorobaina11@gmail.com
- <https://github.com/robainaricardo>
- <https://www.linkedin.com/robainaricardo>
- @robainaricardo
- @robainaricardo