Android-приложения на Kotlin: почему это 👍

Yan Zhulanow, JetBrains

04/06/2016

- Что такое Java
- Почему Kotlin
- Anko в примерах

Android 1.0 (Apple Pie)

- 2008 год
- Java 6

Android 6.0

– 2015 год

– Java 6*

* С кусочками Java 7. Просто чтобы вам было завидно.

Java 8!

Лямбды

Java 6/7 и

Java

А что, собственно, не так с Java?

Багаж неудачных решений

Класс “Person”

Person

```
+ Id
+ First name
+ Last name
```

Что хочется

Что-то вроде такого

```
public class Person {
    public final long id;
    public final String firstName;
    public final String lastName;
}
```

```
public class Person {
    public final long id;
    public final String firstName;
    public final String lastName;

    public Person(long id, String firstName, String lastName) {
        this.id = id;
        this.firstName = firstName;
        this.lastName = lastName;
    }
}
```

```java
public class Person {
    private final long id;
    private final String firstName;
    private final String lastName;

    public Person(long id, String firstName, String lastName) {
        this.id = id;
        this.firstName = firstName;
        this.lastName = lastName;
    }

    public long getId() {
        return id;
    }

    public String getFirstName() {
        return firstName;
    }

    public String getLastName() {
        return lastName;
    }
}
```

```java
public class Person {
    private final long id;
    private final String firstName;
    private final String lastName;

    public Person(long id, String firstName, String lastName) {
        this.id = id;
        this.firstName = firstName;
        this.lastName = lastName;
    }

    public long getId() {
        return id;
    }

    public String getFirstName() {
        return firstName;
    }

    public String getLastName() {
        return lastName;
    }

    @Override
    public String toString() {
        return "Person{" +
                "firstName='" + firstName + '\'' +
                ", id=" + id +
                ", lastName='" + lastName + '\'' +
                '}';
    }
}
```

```java
public class Person {
    private final long id;
    private final String firstName;
    private final String lastName;

    public Person(long id, String firstName, String lastName) {
        this.id = id;
        this.firstName = firstName;
        this.lastName = lastName;
    }

    public long getId() {
        return id;
    }

    public String getFirstName() {
        return firstName;
    }

    public String getLastName() {
        return lastName;
    }

    @Override
    public String toString() {
        return "Person{" +
                "firstName='" + firstName + '\'' +
                ", id=" + id +
                ", lastName='" + lastName + '\'' +
                '}';
    }

    @Override
    public boolean equals(Object o) {
        if (this == o) return true;
        if (o == null || getClass() != o.getClass()) return false;

        Person person = (Person) o;

        if (id != person.id) return false;

        if (firstName != null
                ? !firstName.equals(person.firstName)
                : person.firstName != null)
            return false;

        return lastName != null
                ? lastName.equals(person.lastName)
                : person.lastName == null;
    }

    @Override
    public int hashCode() {
        int result = (int) (id ^ (id >>> 32));
        result = 31 * result + (firstName != null ? firstName.hashCode() : 0);
        result = 31 * result + (lastName != null ? lastName.hashCode() : 0);
        return result;
    }
}
```

```java
public class Person {
    private final long id;
    private final String firstName;
    private final String lastName;

    public Person(long id, String firstName, String lastName) {
        this.id = id;
        this.firstName = firstName;
        this.lastName = lastName;
    }

    public long getId() {
        return id;
    }

    public String getFirstName() {
        return firstName;
    }

    public String getLastName() {
        return lastName;
    }

    @Override
    public String toS
        return

                                                ass()) return false;

                                  o;

                person.id) return false;

        if (firstName != null
                ? !firstName.equals(person.firstName)
                : person.firstName != null)
            return false;

        return lastName != null
                ? lastName.equals(person.lastName)
                : person.lastName == null;
    }

    @Override
    public int hashCode() {
        int result = (int) (id ^ (id >>> 32));
        result = 31 * result + (firstName != null ? firstName.hashCode() : 0);
        result = 31 * result + (lastName != null ? lastName.hashCode() : 0);
        return result;
    }
}
```

58 LoC

У нас есть изолента

– Lombok

– AutoValue

– Всякая другая всячина

У нас есть изолента

– Lombok

– AutoValue

– Всякая другая всячина

Немного о Kotlin

- Разработка началась в **2010** году
- Первый стабильный релиз (1.0) – февраль **2016**
- JVM, Android, JS (experimental)

CRASH COURSE

Очень Простая Функция

```
fun max(a: Int, b: Int): Int {
    if (a > b) {
        return a
    } else {
        return b
    }
}
```

“Expression Body”

`if` — это выражение

```
fun max(a: Int, b: Int) = if (a > b) a else b
```

Тип функции выводится
из типа выражения

Классы

Конструктор

Свойства

```
class Person(
    val id: Int,
    val firstName: String,
    val familyName: String)
```

Дата-классы

toString(), equals(),
hashCode(), copy(), ...

```
data class Person(
        val id: Int,
        val firstName: String,
        val familyName: String)
```

Свойства

Отображаем Toast

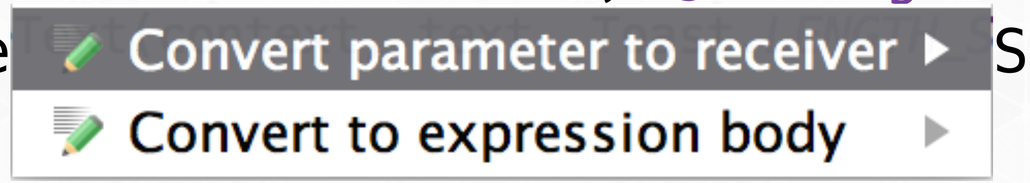
```
fun showToast(context: Context, @StringRes text: Int) {
    Toast.makeText(context, text, LENGTH_SHORT).show()
}
```

. . .

```
class MyActivity : Activity() {
    fun showAbout() {
        showToast(this, R.string.about)
    }
}
```

Отображаем Toast

```
fun showToast(context: Context, @StringRes text: Int) {
    Toast.make                                    SHORT).show()
}
```



. . .

```
class MyActivity : Activity() {
    fun showAbout() {
        showToast(this, R.string.about)
    }
}
```

Extension-функции

```
fun Context.showToast(@StringRes text: Int) {
    Toast.makeText(this, text, LENGTH_SHORT).show()
}
```

. . .

```
class MyActivity : Activity() {
    fun showAbout() {
        // Или this.showToast(R.string.about)
        showToast(R.string.about)
    }
}
```

Лямбды

Дорогая штука

```
debug("MyActivity", debugInfo())
```

. . .

```
fun debug(tag: String, message: String) {
    if (Log.isLoggable(tag, Log.DEBUG)) {
        Log.d(tag, message)
    }
}
```

Лямбды

Вычислим когда-нибудь потом

```
debug("MyActivity", { debugInfo() })
```

. . .

Функциональный тип

```
fun debug(tag: String, message: () -> String) {
    if (Log.isLoggable(tag, Log.DEBUG)) {
        Log.d(tag, message())
    }
}
```

Вычисляем лямбду

Лямбды

```
debug("MyActivity") { debugInfo() }
```

. . .

```
fun debug(tag: String, message: () -> String) {
    if (Log.isLoggable(tag, Log.DEBUG)) {
        Log.d(tag, message())
    }
}
```

Лямбды

```
fun repeat(count: Int, f: (Int) -> Unit) {
    for (i in 1..count) f(i)
}
```

. . .

```
repeat(5) {
    println("Iteration $it")
}
```

I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,
I WILL NOT REPEAT MYSELF AGAIN, I WILL NOT REPEAT MYSELF AGAIN,

Простой пример

ViewGroup

```
int childCount = parentView.getChildCount();
for (int i = 0; i < childCount; ++i) {
    View child = parentView.getChildAt(i);
    // your code
}
```

Простой пример

```
int childCount = parentView.getChildCount();
for (int i = 0; i < childCount; ++i) {
    View child = parentView.getChildAt(i);
    // your code
}
```

Declaration site:

```
public interface Action1<T> {
    void invoke(T t);
}

void loopViews(ViewGroup parentView, Action1<View> action) {
    int childCount = parentView.getChildCount();
    for (int i = 0; i < childCount; ++i) {
        action.invoke(parentView.getChildAt(i));
    }
}
```

Use site:

```
ViewUtils.loopViews(parentView,
        v -> /* your code */);
```

⚠ Создание нового объекта!

Declaration site:

```
inline fun ViewGroup.loopViews(action: (View) -> Unit) {
    for (i in 0..childCount) {
        action(getChildAt(i))
    }
}
```

Use site:

```
parentView.loopViews { /* code */ }
```

A 1E9 dollar mistake

A 1E9 dollar mistake

`NullPointerException`

Null safety

```
val name: String? = null

name.length

val len1: Int? = name?.length

val len2: Int = name!!.length
```

Null safety

Elvis-оператор

```
val elvis: Int = name?.length ?: len1 ?: 0

if (name != null) {
    name.length
}

if (context is Activity) {
    context.finish()
}
```

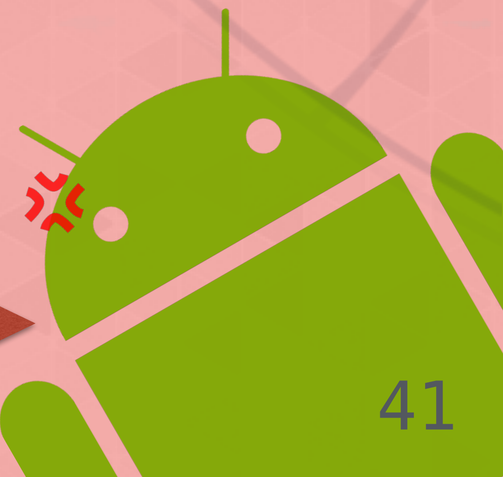
Smart cast

Без обёрток! 👍

findViewById()

`findViewById()`

FATAL EXCEPTION: main
java.lang.**ClassCastException**: android.widget.**Button** cannot be cast to com.yourapp.ui.**SuperButton**
    at com.yourapp.SomeActivity.onCreate(SomeActivity.java:82)
    ...



ButterKnife

```
class SimpleActivity : Activity() {
    @BindView(R.id.name)
    lateinit var name: TextView

    override fun onCreate(savedInstanceState: Bundle?) {
        ...

        ButterKnife.bind(this)
        name.text = "Какой-нибудь текст"
    }
}
```

Property Delegation

Делегированное свойство

Делегат

```
class ClassWithLazy {
    val lazy: String by lazy {
        "А вот " + "моё " + "значение"
    }
}
```

. . .

Вызов конструктора. new не нужен

```
val clazz = ClassWithLazy()
val lazy = lazySample.lazy
```

KotterKnife

by Jake Wharton

Property delegation

```
class KotterKnifeActivity : Activity() {
    private val name by bindView<TextView>(R.id.name)

    override fun onCreate(savedInstanceState: Bundle?) {
        ...

        name.text = "Какой-нибудь текст"
    }
}
```

getText()

Android Extensions

Импорт из специального пакета

```
import kotlinx.android.synthetic.main.activity_main.name

class MyActivity : Activity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        ...
        name.text = "Какой-нибудь текст"
    }
}
```

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:id="@+id/name"/>
```

Anko

```
class AnkoActivity : Activity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        ...
        verticalLayout {
            textView("Какой-нибудь текст”)
            button("Нажми меня") {
                textSize = 18f
                onClick { handleClick() }
            }
        }
    }

    fun handleClick() = ...
}
```

Создаём TextView напрямую, без XML

Extension-лямбды

```
buildString {
    append(“first ”) // Куда append()???
    append("second")
}
```

Extension-лямбды

```
buildString {
    append(“first ”)
    append("second")
}
```

```
inline fun buildString(f: StringBuilder.() -> Unit): String {
    val builder = StringBuilder()
    builder.f()
    return builder.toString()
}
```

Extension-лямбды

```
buildString { // this: StringBuilder
    this.append(“first ")
    this.append(“second")
}

inline fun buildString(f: StringBuilder.() -> Unit): String {
    val builder = StringBuilder()
    builder.f()
    return builder.toString()
}
```

Extension-лямбды

```
buildString { // this: StringBuilder
    this.append(“first ")
    this.append(“second")
}
```

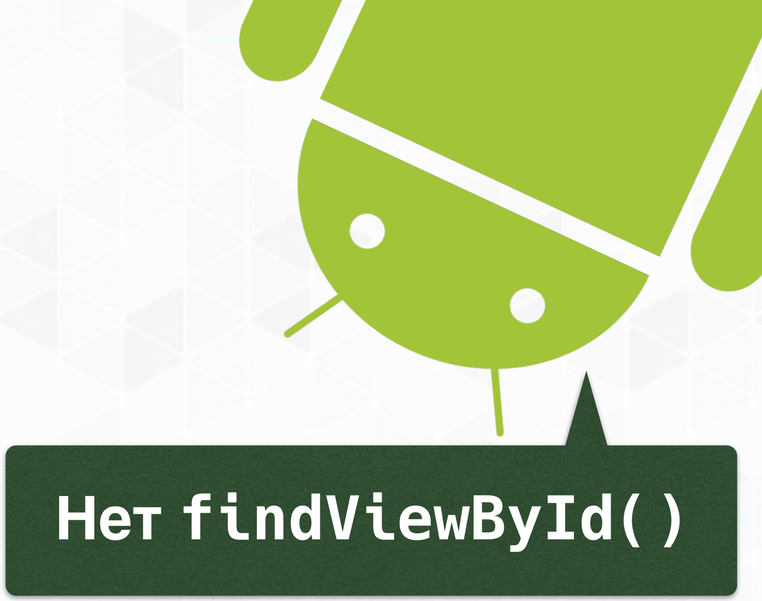
“first second”

```
inline fun buildString(f: StringBuilder.() -> Unit): String {
    val builder = StringBuilder()
    builder.f()
    return builder.toString()
}
```

```
verticalLayout {
    padding = dip(32)

    imageView(R.drawable.ic_login).lparams {
        margin = dip(16)
        gravity = Gravity.CENTER
    }

    val name = editText {
        hintResource = R.string.name
    }

    val password = editText(InputConstraints.PASSWORD) {
        hintResource = R.string.password
    }

    button("Log in") {
        onClick { tryLogin(name.text, password.text) }
    }
}
```

```
verticalLayout { // this: VerticalLayout
    padding = dip(32)

    imageView(R.drawable.ic_login).lparams {
        // this: VerticalLayout.LayoutParams
        margin = dip(16)
        gravity = Gravity.CENTER
    }

    val name = editText { // this: EditText
        hintResource = R.string.name
    }

    val password = editText(InputConstraints.PASSWORD) {
        // this: EditText
        hintResource = R.string.password
    }

    button("Log in") { // this: Button
        onClick { tryLogin(name.text, password.text) }
    }
}
```

```
verticalLayout { // this: VerticalLayout
    padding = dip(32)

    imageView(R.drawable.ic_login).lparams {
        // this: VerticalLayout.LayoutParams
        margin = dip(16)
        gravity = Gravity.CENTER
    }

    val name = editText { // this: EditText
        hintResource = R.string.name
    }

    val password = editText(InputConstraints.PASSWORD) {
        // this: EditText
        hintResource = R.string.password
    }

    button("Log in") { // this: Button
        onClick { tryLogin(name.text, password.text) }
    }
}
```



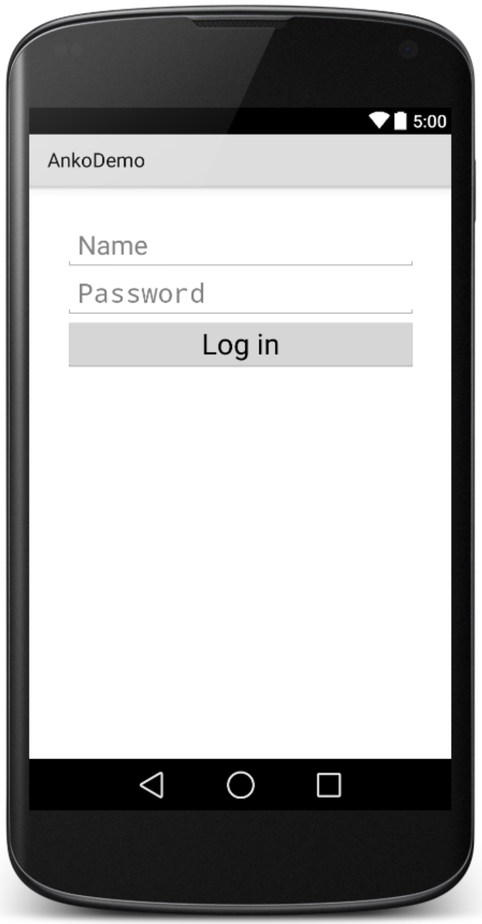
MainActivityUi.kt - anko-example - [~/jb/anko-example]

```kotlin
package org.example.ankodemo

import ...

class MainActivityUi : AnkoComponent<MainActivity> {
    private val customStyle = { v: Any ->
        when (v) {
            is Button -> v.textSize = 26f
            is EditText -> v.textSize = 24f
        }
    }

    override fun createView(ui: AnkoContext<MainActivity>) = with(ui) {
        verticalLayout {
            padding = dip(32)

            val name = editText {
                hintResource = R.string.name
            }
            val password = editText(InputConstraints.PASSWORD) {
                hintResource = R.string.password
            }

            button("Log in") {
                onClick {
                    ui.owner.tryLogin(ui, name.text, password.text)
                }
            }
        }.applyRecursively(customStyle)
    }
}
```

Anko DSL Preview

Nexus 4 | Light

org.example.ankodemo.**MainActivityUi**



Intents

```
// Создаём новый Intent
val intent = Intent(this,
    SomeOtherActivity::class.java)

// Кладём туда что-нибудь ценное
intent.putExtra("id", 5)

// И, наконец, запустим новую Activity
startActivity(intent)
```

Intents

```
// Создаём новый Intent
val intent = Intent(this,
```

И вся эта ерунда,
только чтобы показать новое окно

```
// И, наконец, запустим новую Activity
startActivity(intent)
```

Intents

Extension-функция к `Context`

```
startActivity<MyActivity>("id" to 5)
```

Немного приятностей

```
val successful = makeCall(“+79212345678”)

browse(“http://my-cool-site.com/“)

sendSms(“+79212345678”,
    “Have you tried turning it off and on again?”)
```

Сервисы

```
val notificationManager =
    getSystemService(Context.NOTIFICATION_SERVICE)
    as NotificationManager

notificationManager.notify(id, notification)
```

Сервисы

```
notificationManager.notify(id, notification)
```

Сервисы

Extension-свойство для `Context`

```
notificationManager.notify(id, notification)
```

Async

```
doAsync {
    // Load data somehow
    val users = getUsers()
    // Store it locally
    userRepository.save(users)
}
```

Async

```
doAsync {
    // Load data somehow
    val users = getUsers()
    // Store it locally
    userRepository.save(users)

    // And update list in UI thread
    uiThread { act ->
        act.updateList(users)
    }
}
```

Anko

https://github.com/Kotlin/anko

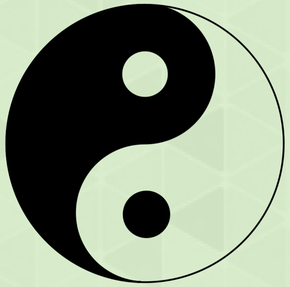
– Но мой проект уже написан на Java

Совместимость с Java: 100%

– **Kotlin** может вызвать любой метод из **Java**

– **Java** может вызвать любую функцию из **Kotlin**

– Добавить **Kotlin** в существующий Java-проект очень просто



JVM Languages

Scala

Groovy

Kotlin

Не больше 64k методов!

Kotlin Stdlib (1.0.2)

≈ Один мегабайт

≈ 5700 Java-методов

Kotlin Stdlib (1.0.2)

полезно для Android

Lint

This fragment should provide a default constructor

```
public class MyFragment extends Fragment {
    public MyFragment(String name) {
    }
}
```

Lint. Теперь и в Kotlin

This fragment should provide a default constructor

```
class MyFragment(name: String) : Fragment()
```

UAST (Universal Abstract Syntax Tree)

- http://kotlinlang.org
- Twitter: @kotlin
- https://kotlinlang.slack.com

Zhulanow Yan,

yan.zhulanow@jetbrains.com,

Twitter: @yanex_ru

QA