Kotlin & Android

Sergejs Luhmirins, 2016

Who am I?

- Android Developer for 4+ years
- Worked at Accenture, Fabula, Philips
- Now at FullContact
- Experimenting with Kotlin since first beta

Agenda

1. View binding
2. Power of extensions
3. Better listeners
4. Faster layouts
5. Fluent testing
6. Costs & Benefits
7. How to start

Assume we have a view

```xml
<RelativeLayout
  android:id="@+id/form_root">

  <ImageView android:id="@+id/form_photo" />
  <TextView android:id="@+id/form_first_name" />
  <TextView android:id="@+id/form_last_name" />
  <Button android:id="@+id/form_edit" />

</RelativeLayout>
```

View binding (classic way)

```java
RelativeLayout root;
// ...
Button edit;

@Override void onCreate(Bundle savedInstanceState) {
  // super + setContentView
  root = (RelativeLayout) findViewById(R.id.form_root);
  photo = (ImageView) findViewById(R.id.form_photo);
  firstName = (TextView) findViewById(R.id.form_first_name);
  lastName = (TextView) findViewById(R.id.form_last_name);
  edit = (Button) findViewById(R.id.form_edit);
}
```

Official docs

View binding (proper way)

```java
@BindView(R.id.form_root) RelativeLayout root;
@BindView(R.id.form_photo) ImageView photo;
@BindView(R.id.form_firstName) TextView firstName;
@BindView(R.id.form_lastName) TextView lastName;
@BindView(R.id.form_edit) Button edit;

@Override void onCreate(Bundle savedInstanceState) {
   // super + setContentView
   ButterKnife.bind(this);
}
```

View binding (Kotlin way)

```
import kotlinx.android.synthetic.main.form_layout.*

@Override
fun onCreate(savedInstanceState: Bundle) {
  // super + setContentView

  form_first_name.text = "John"
  view.form_last_name.text = "Doe"
}
```

Neat extension functions

```kotlin
fun ViewGroup.inflate(
  @LayoutRes layoutRes: Int,
  attachToRoot: Boolean = false
): View {
 return LayoutInflater.from(context)
           .inflate(layoutRes, this, attachToRoot)
}
```

```kotlin
parent.inflate(R.layout.my_layout)
parent.inflate(R.layout.my_layout, true)
```

Neat extension functions

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

```
imageView.loadUrl("http://....")
```

Neat extension functions

```kotlin
fun View.setVisible(visible: Boolean) {
    this.visibility = if (visible) View.VISIBLE else View.GONE
}
```

```kotlin
imageView.setVisible(true)
imageView.setVisible(false)
```

Listeners & callbacks (old way)

View action listeners:

```
button.setOnClickListener(new OnClickListener() {
  @Override
  void onClick(View view) {
    // do stuff when button clicked
  }
});
```

Listeners & callbacks (newer way)

View action listeners:

```
button.setOnClickListener(view -> {
    // do stuff when button clicked
});
```

But requires Jack&Jill or Retrolambda

Listeners & callbacks (Kotlin way)

View action listeners:

```
button.setOnClickListener { view ->
  // do stuff when button clicked
}
```

Listeners & callbacks (old way)

Basic callback:

```
interface ActionCallback {
  void onActionDone();
}

public void doAction(int param, @Nullable ActionCallback cb) {
  // do stuff
  if (cb != null ) {
    cb.onActionDone()
  }
}
```

Listeners & callbacks (old way)

Basic callback:

```
public void fooBar() {
  doAction(5, new ActionCallback(){
    @Override
    void onActionDone() {
      // do stuff when action is done
    }
  });
}
```

Listeners & callbacks (newer way)

Basic callback:

```
public void fooBar() {
  doAction(5, () -> {
      // do stuff when action is done
  });
}
```

But requires Jack&Jill or Retrolambda

Listeners & callbacks (Kotlin way)

Basic callback:

```kotlin
fun doActions(param: Int, callback: (() -> Unit)?)  {
  // do stuff
  callback?.invoke()
}

fun fooBar() {
  doAction(5) {
      // do stuff when action is done
  }
  doAction(7, null)
}
```

Anko

- Lots of extensions
- DSL for layouts
  - Completly code based
  - ~~Allegedly~~ 4x faster than xml

Anko (extensions)

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

```
alert("Hi, I'm Roy", "Have you tried turning it off and on again?"
    yesButton { toast("Oh…") }
    noButton {}
}.show()
```

```
doAsync {
    // Long background task
    uiThread { result.text = "Done" }
}
```

Anko (dsl)

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

    verticalLayout {
        padding = dip(30)
        editText { hint = "Name" }
        editText { hint = "Password" }
        button("Edit") {
            textSize = 26f
        }
    }
}
```

Testing

General frameworks:

- Spek
- KotlinTest
- expekt

Libraries for Android:

- mockito-kotlin
- hamkrest
- Kluent

Testing (Kluent)

```
val subject : String = getInput()

subject shouldEqual "hello"

subject shouldNotEqual "world"
```

```
val mock = mock(Database::class)

mock.getPerson()


Verify on mock that mock.getPerson() was called
```

Costs

- Added file size <1MB
- Slight mess while in transition
- Gradle slower by 0-15%
  - 15% for clean cold build
  - 0% for incremental build on deamon

Benefits

- Up to 30% less code
- Powerful standart library
  - No need for Retrolabda, Guava, ButterKnife etc.
  - Less need for RxJava
- Coding is pure joy

Keepsafe article

How to start? Easy way

- Install plugin
- Tools > Kotlin > Configure Kotlin in Project
- Code > Convert Java file to Kotlin file

How to start? Proper way

Project `build.gradle`:

```
buildscript {
  ext.kotlin = "X.Y.Z"
  dependencies {
    classpath
      "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin"
    classpath
      "org.jetbrains.kotlin:kotlin-android-extensions:$kotlin"
  }
}
```

How to start? Proper way

Module `build.gradle`:

```
apply plugin: 'kotlin-android'

apply plugin: 'kotlin-android-extensions'


sourceSets {

    main.java.srcDirs += 'src/main/kotlin'

}


dependancies {

  compile "org.jetbrains.kotlin:kotlin-stdlib:$kotlin"

}
```

Things to read

- Kotlin in Action
- Kotlin for Anroid developers
- Learn Kotlin by developing (medium.com)