Ízelítő a Kotlin nyelvű Android fejlesztésből

Braun Márton Szabolcs

zsmb13@sch.bme.hu

github.com/zsmb13

Háttér

- JetBrains
  - 2010: fejlesztés kezdete
  - 2016 február: Kotlin 1.0
- JVM nyelv
  - Java 6-os bytecode-ra fordul
  - 100%-os együttműködés Java kóddal

Használat

- A megszokott környezetben használható
- Android Studio plugin
- 4 sor a Gradle build fájlba
  - Kis méretű runtime és stdlib (<1 MB)
- Fájlonként adoptálható
  - Új kód Kotlinban
  - Meglévő Java kód refaktorálása

Kotlin gyorstalpaló

Változók

```kotlin
// Kotlin

var x: Int = 1

val x = 1
```

```java
// Java

int x = 1;

final int x = 1;
```

Függvények

```
fun add(a: Int, b: Int): Int {
    return a + b
}
```

Osztály deklaráció

```
class Person(val name: String, var age: Int) {

}
```

Osztály deklaráció

```
class Person(val name: String, var age: Int)
```

Az osztály Java megfelelője

```java
public class Person {

    private final String name;
    private int age;

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

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

}
```

Data class

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

A data class Java megfelelője

```java
public class Person {

    private final String name;
    private int age;

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

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public String toString() {
        return "Person{" +
                "name='" + name + '\'' +
                ", age=" + age +
                '}';
    }

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

        Person person = (Person) o;

        if (age != person.age) return false;
        return name != null ? name.equals(person.name)
                            : person.name == null;
    }

    public int hashCode() {
        int result = name != null ? name.hashCode() : 0;
        result = 31 * result + age;
        return result;
    }

}
```

Data class - összefoglaló

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

```java
public class Person {

    private final String name;
    private int age;

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

    public String getName() {
        return name;
    }

    public int getAge() {
        return age;
    }

    public void setAge(int age) {
        this.age = age;
    }

    public String toString() {
        return "Person{" +
                "name='" + name + '\'' +
                ", age=" + age +
                '}';
    }

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

        Person person = (Person) o;

        if (age != person.age) return false;
        return name != null ? name.equals(person.name) : person.name == null;
    }

    public int hashCode() {
        int result = name != null ? name.hashCode() : 0;
        result = 31 * result + age;
        return result;
    }

}
```

Példák

Property hozzáférések

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


// Kotlin

val person = Person("Sam", 41)

println(person.name)

person.age = 42
```

Property hozzáférések

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

```
// Java

Person person = new Person("Sam", 41);

System.out.println(person.getName());

person.setAge(42);
```

Property hozzáférések

```
// Java
recyclerView.setAdapter(adapter);
recyclerView.setDrawingCacheEnabled(true);
recyclerView.setDrawingCacheQuality(QUALITY_HIGH);

// Kotlin
recyclerView.adapter = adapter
recyclerView.isDrawingCacheEnabled = true
recyclerView.drawingCacheQuality = QUALITY_HIGH
```

Extension function

```
button.setVisibility(View.GONE)

button.visibility = View.GONE

button.hide()

fun View.hide() {
    this.visibility = View.GONE
}
```

Extension function

```kotlin
// Android.kt
package util.extensions

import android.view.View

fun View.show() {
    this.visibility = View.VISIBLE
}

fun View.hide() {
    this.visibility = View.GONE
}
```

Kotlin Android Extensions

```
// activity_login.xml
<Button android:id="@+id/btnLogin" />
<WebView android:id="@+id/webview" />

// LoginActivity.kt
import kotlinx.android.synthetic.main.activity_login.*
import util.extensions.*

fun showWebView() {
    webview.show()
    btnLogin.hide()
}
```

Lambda kifejezések

```
fun useNumber(number: Int, operation: (Int) -> Unit) {
    operation(number)
}

useNumber(5, { x -> println(x * 2) })

useNumber(5) { x -> println(x * 2) }

useNumber(5) { println(it * 2) }
```

SAM conversion

```
public interface OnClickListener {
    void onClick(View v);
}
```

SAM conversion

```
btnLogin.setOnClickListener(
        object : View.OnClickListener {
            override fun onClick(v: View) {
                presenter.startLogin()
            }
        }
)

btnLogin.setOnClickListener {
    presenter.startLogin()
}
```

SAM conversion

```
public interface OnClickListener {
    void onClick(View v);
}
```

```
btnLogin.setOnClickListener {
    presenter.startLogin()
}
```

Lambda + extension function

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

buildString {
    this.append("Google I/O")
    append(" Extended")
}
```

Deklaratív hierarchia leírás

```
<LinearLayout
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:orientation="vertical">

    <EditText
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <Button
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
</LinearLayout>
```

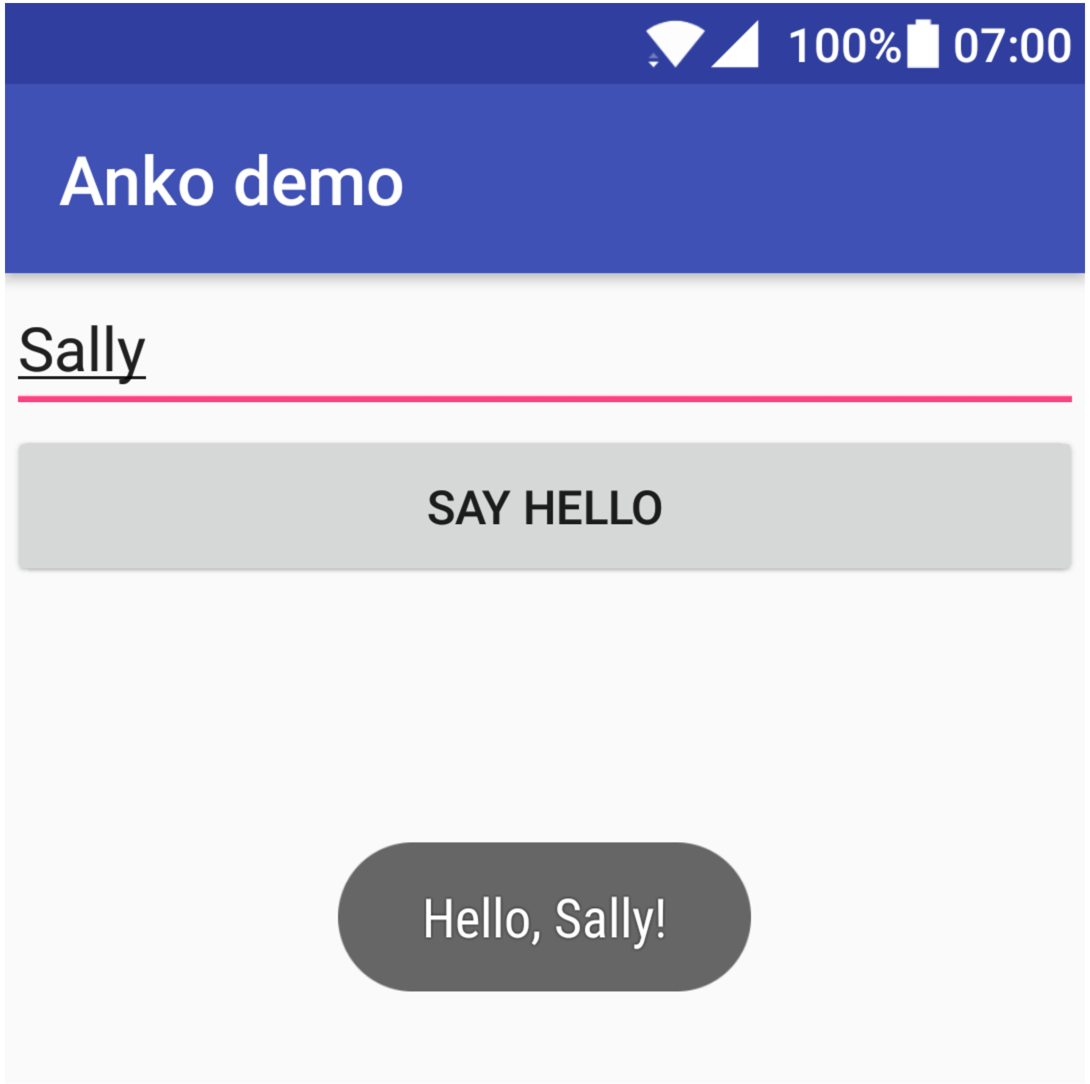
Imperatív hierarchia építés

```
LinearLayout linearLayout = new LinearLayout(this);

EditText editText = new EditText(this);
Button button = new Button(this);

button.setOnClickListener(...);

linearLayout.addView(editText);
linearLayout.addView(button);

setContentView(linearLayout);
```

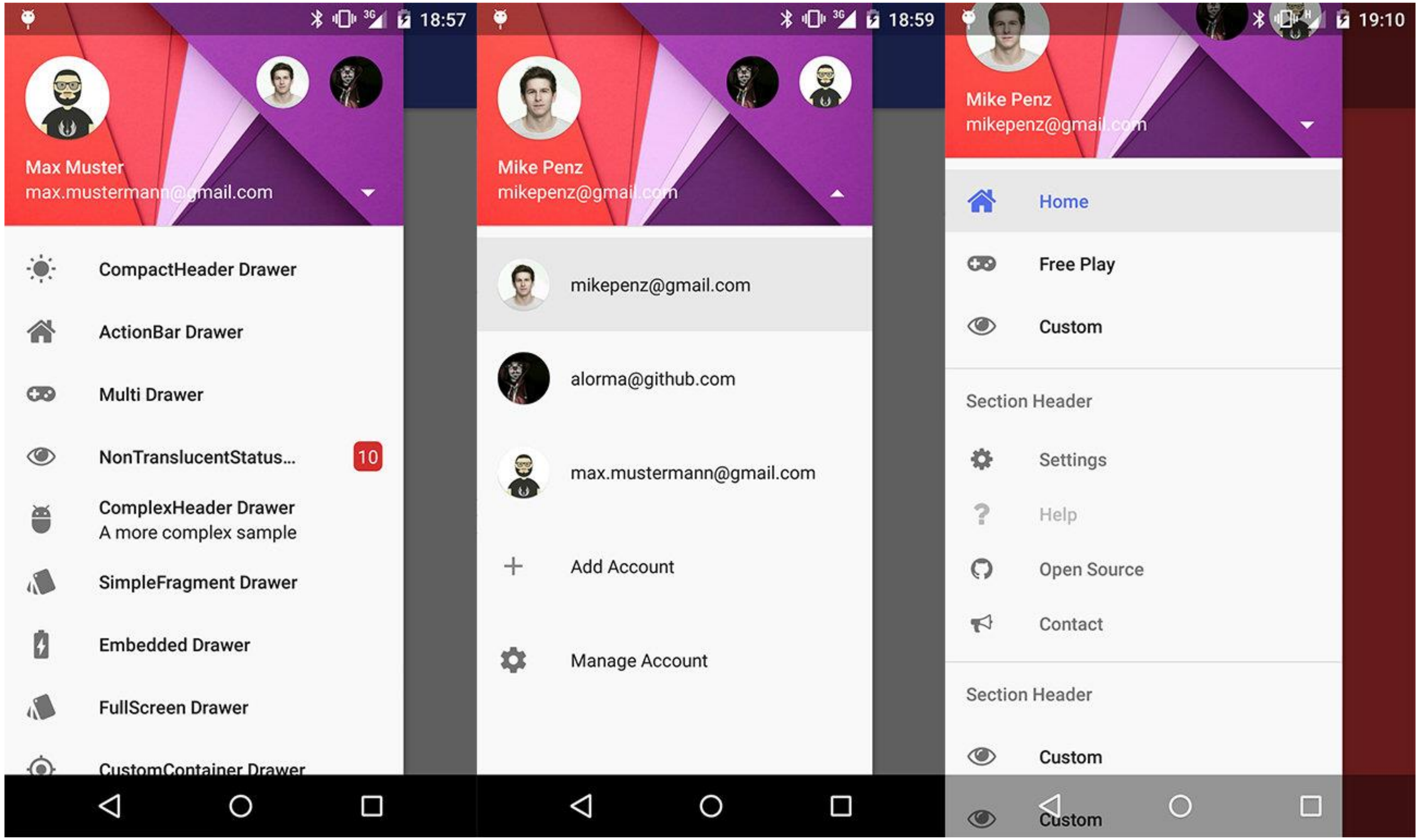
Anko View DSL

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

    verticalLayout {
        val name = editText()
        button("Say Hello") {
            onClick {
                toast("Hello, ${name.text}!")
            }
        }
    }
}
```

Anko View DSL



MaterialDrawer

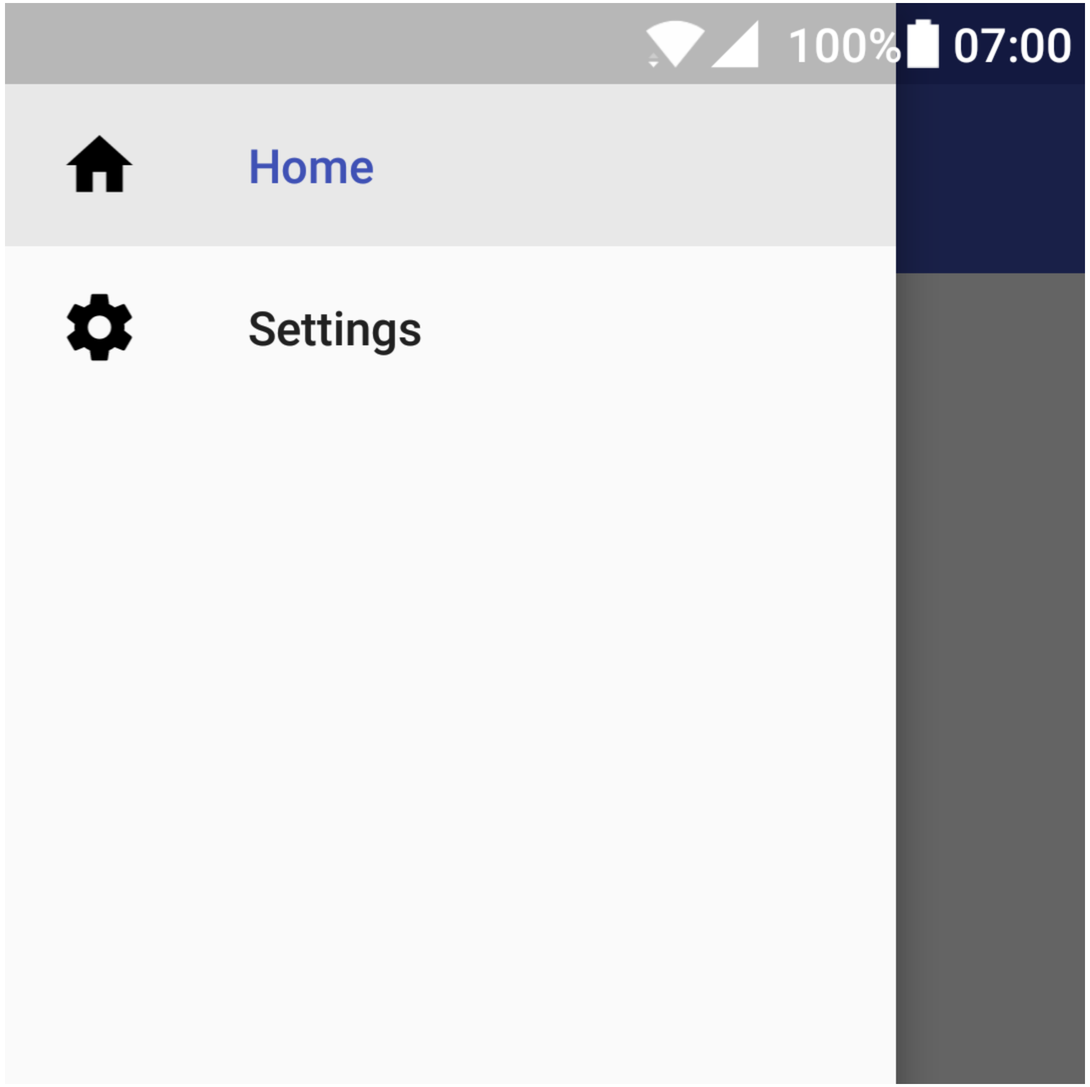


https://github.com/mikepenz/MaterialDrawer

MaterialDrawer

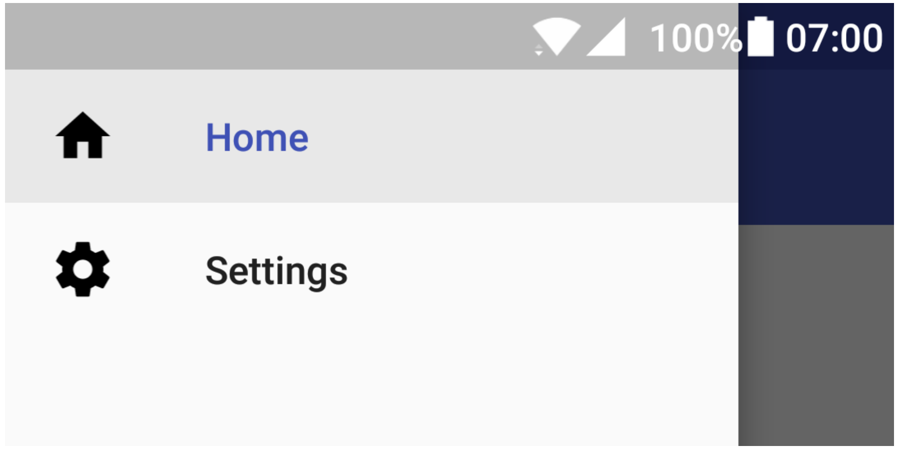
```java
new DrawerBuilder(this)
        .withCloseOnClick(true)
        .addDrawerItems(
                new PrimaryDrawerItem()
                        .withName("Home")
                        .withIcon(R.drawable.ic_home),
                new PrimaryDrawerItem()
                        .withName("Settings")
                        .withIcon(R.drawable.ic_settings)
        )
        .build();
```

MaterialDrawer



MaterialDrawerKt

```
drawer {
    closeOnClick = true
    primaryItem("Home") {
        icon = R.drawable.ic_home
    }
    primaryItem("Settings") {
        icon = R.drawable.ic_settings
    }
}
```



https://github.com/zsmb13/MaterialDrawerKt

Levezetés

Több, mint egy “jobb Java”

- JVM
  - Android
  - Desktop
  - Szerver
- Gradle Script Kotlin
- JavaScript (1.1 óta stabil)
  - Kliens oldal
  - Szerver oldal
- (Native)

Merre tovább?

- kotlinlang.org

- Android Development with Kotlin (Jake Wharton)
- 10 Kotlin Tricks in 10(ish) Minutes (Jake Wharton)
- Kotlin in Production (Christina Lee)
- Kotlin – Ready for Production (Hadi Hariri)

Köszönöm a figyelmet!

zsmb13@sch.bme.hu

github.com/zsmb13

Fotó: Alexey Sergeev