Desarrollo Android

Clase 06

View binding

findViewById

```
//En el archivo xml tenemos esta vista con id 'title'
<TextView
  android:id="@+id/title"
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:layout_gravity="center"/>

//Para acceder a esta vista desde la clase utilizamos el findViewById
//Este puede ser almacenado en una variable
val titleTextView = findViewById<TextView>(R.id.title)

//Luego puedo acceder a los atributos de ese tipo de vista (TextView) como por ejemplo setText
titleTextView.text = "Este es el título"
```

findViewById

Problemas:
1) Seguridad de tipo y de null pointer → al crear el findViewById le asigno el tipo y el id manualmente

2) Utilizar findViewById puede resultar muy engorroso si tengo muchas vistas en mi layout → debería crear una variable para cada vista y almacenarla con el findViewById

ViewBinding

ViewBinding es una herramienta que permite vincular una vista con el módulo del proyecto.

Esto genera una clase de vinculación para cada archivo XML que permite acceder desde las clases que lo necesiten sin tener que llamar al findViewById

ViewBinding

Para activarlo:
1) Ir al build.gradle del módulo: build.gradle (Module: **)**
2) Agregar:

```
android {
  ...
  viewBinding {
    enable = true
  }
}
```

ViewBinding

El layout:
results_profile.xml

genera una clase:
ResultsProfileBinding

Esta clase nos permite acceder a la raíz del layout (.root) o a cada vista (.title, .loginButton)

ViewBinding

Para usarlo en cada clase Activity:

1) Crear una variable

private lateinit var binding: ResultProfileBinding

2) Dentro del *onCreate()*:

binding = ResultProfileBinding.inflate(layoutInflater)

setContentView(binding.root)

Ya puedo acceder a cada vista: *binding.title*

ViewBinding

Para usarlo en cada clase Fragment:

```
private var _binding: ResultProfileBinding? = null
// This property is only valid between onCreateView and
// onDestroyView.
private val binding get() = _binding!!

override fun onCreateView(
    inflater: LayoutInflater,
    container: ViewGroup?,
    savedInstanceState: Bundle?
): View? {
    _binding = ResultProfileBinding.inflate(inflater, container, false)
    val view = binding.root
    return view
}

override fun onDestroyView() {
    super.onDestroyView()
    _binding = null
}
```

Estilos

Estilos y temas

Estilo: atributos que especifican la apariencia de una sola View (color, el tamaño de fuente, el color de fondo, etc).

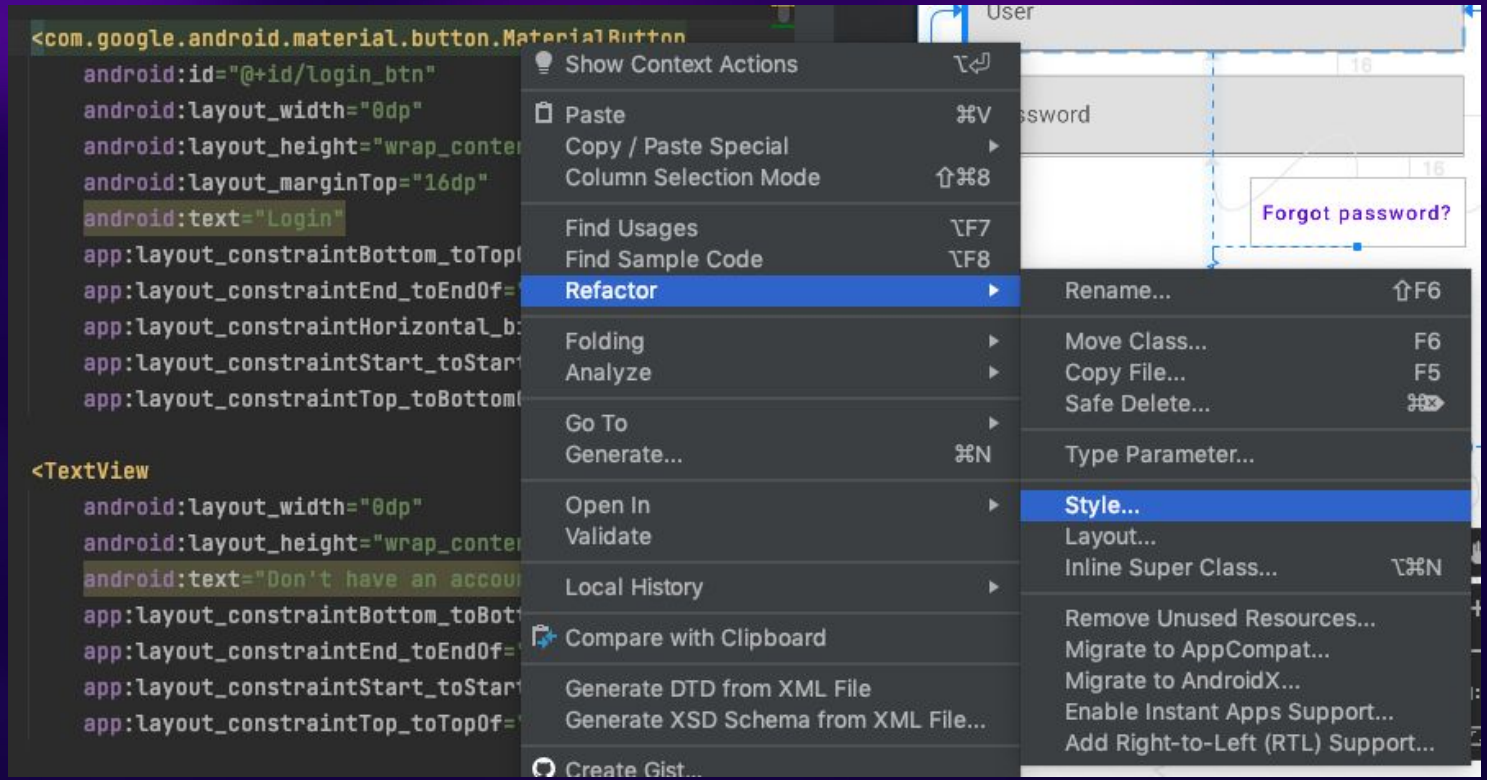
Tema: atributos que se aplican a toda una app, actividad o jerarquía de vistas, no solo a una vista individual.

Estilos: extracción

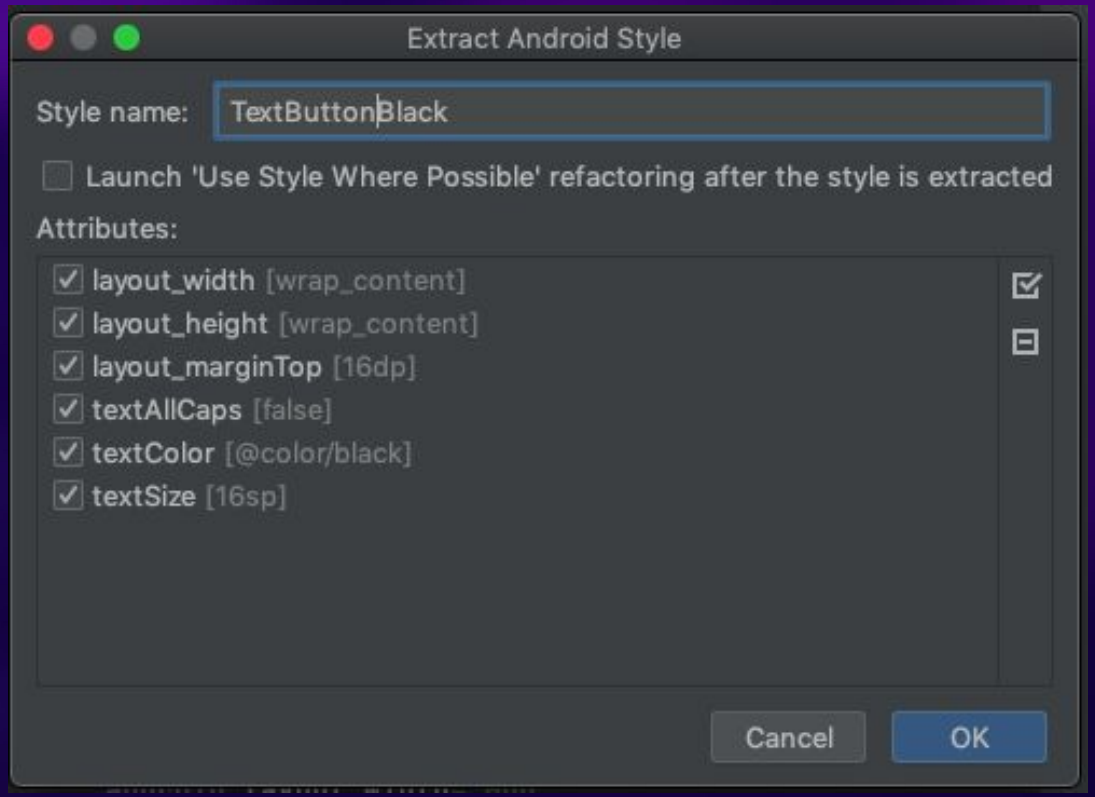
Es posible extraer un estilo de una view ya diseñada.

Para esto:
1) click derecho sobre la vista
2) Refactor → Style

Estilos: extracción



Estilos: extracción



<u>Recomendación</u>: ***layout_width*, *layout_height* y *margins* no conviene agregarlo al style**

Estilos: extracción

```
<com.google.android.material.button.MaterialButton
    android:id="@+id/forgot_pwd_btn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    style="@style/TextButtonBlack"
    android:text="Forgot password?"
    app:layout_constraintEnd_toEndOf="@id/user_input_layout"
    app:layout_constraintTop_toBottomOf="@id/pwd_input_layout" />
```

Estilos: extracción

ty_main.xml | styles.xml | build.gradle (:app) | MainActivity.kt

```xml
<?xml version="1.0" encoding="utf-8"?>
<resources>

    <style name="TextButtonBlack" parent="Widget.MaterialComponents.Button.TextButton">
        <item name="android:layout_marginTop">16dp</item>
        <item name="android:textAllCaps">false</item>
        <item name="android:textColor">@color/black</item>
        <item name="android:textSize">16sp</item>
    </style>
</resources>
```

Navegación entre Activities

Navegación

1) Crear una nueva Activity llamada LoginActivity (File → New → Activity → Empty Views Activity
2) Para navegar desde MainActivity hacia LoginActivity:
   startActivity(intent: Intent)

Navegación

```kotlin
//En MainActivity.kt
val intent = Intent(context, LoginActivity::class.java)
startActivity(intent) //esto abre la nueva activity sin finalizar MainActivity
finish() //si quiero finalizar MainActivity
```

Navegación

```kotlin
//En MainActivity.kt
val intent = Intent(context, LoginActivity::class.java)
//Si quiero pasarle parámetros a LoginActivity:
intent.putExtra("nombreParémetro", valor)
startActivity(intent) //esto abre la nueva activity sin finalizar MainActivity
finish() //si quiero finalizar MainActivity
```

Navegación

```kotlin
//En MainActivity.kt
binding.loginBtn.setOnClickListener {
    val intent = Intent(this, LoginActivity::class.java)
    intent.putExtra("param", true)
    startActivity(intent)
}

//En LoginActivity.kt
val isParam = intent.extras?.getBoolean("param") ?: false
binding.textVw.text = if (isParam) {
    "Es param"
} else {
    "No es param"
}
```