84. Introducción a Android

1.Fundamentos de una Aplicación en Android 2.Componentes de una aplicación. 3.Android Manifest 4.Recursos de una aplicación 5.La Clase R 6.Estructura de un proyecto 6.1.Instalación de Android Studio 6.2.Iniciando el entorno de desarrollo. 6.3.Elementos de un proyecto. 6.4.Configurar el dispositivo donde probar las aplicaciones. •7.Los permisos en Android.

84.1 Componentes de una aplicación

- **Activities**
- Pantallas individuales con interfaces de usuario.
- Cada pantalla es una `Activity`.
- **Services**
- Operan en segundo plano sin interfaz de usuario.
- Tipos: `Started Services` (Iniciados) y `Bound Services` (Vinculados).
- **Broadcast Receivers**
- Reciben y responden a eventos del sistema o aplicaciones.
- Funcionan sin interfaz de usuario.
- **Content Providers**
- Gestionan el acceso a datos estructurados, compartiéndolos entre aplicaciones.
- **Intents**
- Mensajes para la comunicación entre componentes de la aplicación.
- **Fragments**
- Componentes reutilizables con ciclo de vida propio que pueden ser parte de una `Activity`.
- **Views y ViewGroups**
- `Views` son elementos de UI como botones o campos de texto.
- `ViewGroups` son contenedores que organizan `Views` en layouts.
- **Widgets**
- Componentes interactivos para la pantalla de inicio, proporcionando acceso rápido a la app.
- **Notifications**

- Informan al usuario con alertas, sonidos o luces fuera de la UI normal.
- **Resources**
- Elementos como strings, imágenes y archivos XML, usados para diseño y soporte de localización.
- **Manifest File**
- Archivo XML que declara los componentes y requerimientos de la aplicación.

84.2 Android Manifest

Para usar una Activity se debe declarar en el fichero de manifiesto.

```
    <manifest ... >
      <application ... >
          <activity android:name=".ExampleActivity" />
          ...
      </application ... >
      ...
    </manifest >
```

El único atributo obligatorio es `android:name`

En el manifiesto se declara: * Activities * Filtros de Intents

```
    <activity android:name=".ExampleActivity"
android:icon="@drawable/app_icon">
        <intent-filter>
            <action android:name="android.intent.action.SEND" />
            <category android:name="android.intent.category.DEFAULT" />
            <data android:mimeType="text/plain" />
        </intent-filter>
    </activity>
```

* Permisos de usuario de la aplicación

```
    <manifest>
    <activity android:name="...."
       android:permission="com.google.socialapp.permission.SHARE_POST"

    />
```

84.3 Recursos de una aplicación

- **Layouts** (`res/layout/`)
- Definen la UI de las actividades, fragmentos, etc.

- **Values** (`res/values/`)
- **Strings** (`res/values/strings.xml`)
  - Textos utilizados en la app, facilitan la localización.
- **Colors** (`res/values/colors.xml`)
  - Define colores para usar en diseño y código.
- **Styles** (`res/values/styles.xml`)
  - Personaliza la apariencia de las vistas.
- **Dimens** (`res/values/dimens.xml`)
  - Especifica dimensiones y tamaños de texto comunes.
- **Themes** (`res/values/themes.xml`)
  - Estilos aplicados a la app o actividades individuales.
- **Drawables** (`res/drawable/`)
- Imágenes y definiciones de gráficos para la UI.
- **Mipmap** (`res/mipmap/`)
- Iconos de la aplicación en diferentes densidades.
- **Animaciones** (`res/anim/`)
- Animaciones para vistas.
- **Menus** (`res/menu/`)
- Menús para barra de acción y menús contextuales.
- **Raw** (`res/raw/`)
- Archivos binarios como audio o video.
- **XML** (`res/xml/`)
- Archivos XML para configuraciones y proveedores.
- **Assets** (`assets/`)
- Almacena archivos como fuentes y archivos de texto, accesibles vía `AssetManager`.

84.4 La Clase R

En Android, la clase R es una clase autogenerada que actúa como un índice de referencia para todos los recursos disponibles en la carpeta res/ del proyecto.
Cuando se compila una aplicación Android, el compilador genera esta clase, que contiene subclases estáticas para cada tipo de recurso, como drawable, layout, string, colors, y más

- **R.drawable**

- Identificadores para recursos gráficos en `res/drawable`.
- **R.layout**
- Identificadores para archivos de diseño de UI en `res/layout`.
- **R.string**
- Identificadores para cadenas de texto en `res/values/strings.xml`.
- **R.color**
- Identificadores para definiciones de color en `res/values/colors.xml`.
- **R.dimens**
- Identificadores para dimensiones comunes en `res/values/dimens.xml`.
- **R.id**
- Identificadores para elementos de UI con `android:id` en archivos XML.
- **R.menu**
- Identificadores para menús en `res/menu`.
- **R.raw**
- Identificadores para archivos binarios en `res/raw`.
- **R.anim**
- Identificadores para animaciones en `res/anim`.
- **R.attr**
- Identificadores para atributos personalizados en recursos XML.
- **R.style**
- Identificadores para estilos en `res

Por ejemplo para usar `R.string`, primero debes definir el recurso en el archivo strings.xml, que se encuentra dentro del directorio res/values/

```
<resources>
    <string name="hello_world">Hello World!</string>
</resources>
```

En el código ser usa :

```
@Composable
fun Greeting() {
    val context = LocalContext.current

    val helloWorldString = context.getString(R.string.hello_world)

    Text(text = helloWorldString)
}
```

84.5 Estructura de un proyecto

84.6 Instalación de Android Studio

84.7 Iniciando el entorno de desarrollo.

85. Estructura de Proyecto Kotlin en Android Studio

- **Directorio** `src` Contiene todo el código fuente y recursos del proyecto.
- **main**: Código fuente principal del proyecto.
- **java** Contiene los archivos de código Kotlin.
- **res** Almacena los recursos como layouts, strings, imágenes, etc.
- **AndroidManifest.xml** Archivo de configuración principal para actividades, permisos, etc.
- **Directorio** `gradle` Contiene scripts y configuraciones para el sistema de construcción Gradle.
- **Archivos de Configuración de Gradle**
- `build.gradle (Project)` : Configuración de Gradle a nivel de proyecto.
- `build.gradle (Module)` : Configuración de Gradle a nivel de módulo.
- **Directorio** `test` Para pruebas unitarias.

Ç **Directorio** `androidTest` Para pruebas de integración y UI específicas de Android.

- `README.md` Archivo Markdown opcional para documentar el proyecto.
- `.gitignore` Especifica los archivos y directorios que deben ignorarse en los commits (si se usa Git).

85.1 Elementos de un proyecto.

Cuando se crea una aplicación, el asistente de Android Studio crea un árbol de directorios y una serie de ficheros:

85.2 Configurar el dispositivo para probar las aplicaciones.

La aplicación se puede ejecutar (y depurar) en un dispositivo real o simulado.

85.3 Apéndice

Enlaces:

- Documentación sobre Activity

Ver 0.5 (30-10-23)