Documentación: Crea un diseño básico con Jetpack Compose

Fernando Rodriguez

1. Ruta de aprendizaje 1

Introducción

Este documento proporciona una visión general de los conceptos fundamentales de programación en Kotlin, esenciales para el desarrollo de aplicaciones Android utilizando Jetpack Compose. A lo largo de este módulo, se exploran temas como la creación y uso de variables, funciones y la estructura básica de un programa en Kotlin.

Tu primer programa en Kotlin

El primer paso en la programación con Kotlin es comprender cómo estructurar un programa básico. Un programa mínimo en Kotlin se ve de la siguiente manera:

```
fun main() {
    println(" Hola , mundo!")
}
```

Listing 1: Programa básico en Kotlin

Explicación:

- `fun main()`: Define la función principal, punto de entrada del programa.
- `println("¡Hola, mundo!")`: Imprime el mensaje en la consola.

Este programa muestra cómo Kotlin permite una sintaxis concisa y clara para tareas comunes.

Crear y usar variables en Kotlin

Las variables en Kotlin se definen utilizando las palabras clave `val` para valores inmutables y `var` para valores mutables. Ejemplo:

```
val nombre = "Juan"
var edad = 25
```

Listing 2: Definición de variables

Explicación:

- `val nombre = "Juan"`: Define una variable inmutable `nombre` con el valor `"Juan"`.
- `var edad = 25`: Define una variable mutable `edad` con el valor `25`.

El uso adecuado de `val` y `var` ayuda a mantener la integridad y claridad del código.

Crear y usar funciones en Kotlin

Las funciones en Kotlin se definen utilizando la palabra clave `fun`. Ejemplo:

```
fun saludar(nombre: String) {
    println(" Hola , $nombre!")
}
```

Listing 3: Definición de una función

Explicación:

- `fun saludar(nombre: String)`: Define una función `saludar` que toma un parámetro `nombre` de tipo `String`.
- `println("¡Hola,` $nombre!") : $Imprime un saludo personalizado utilizando interpolación de cadenas.$

Esta estructura permite modularizar el código y reutilizar lógica en diferentes partes de la aplicación.

Práctica: Fundamentos de Kotlin

Para consolidar los conocimientos adquiridos, se recomienda realizar ejercicios prácticos que impliquen:

- Definir y utilizar variables de diferentes tipos.
- Crear funciones que realicen operaciones básicas.
- Implementar estructuras de control como condicionales y bucles.

Estas prácticas ayudarán a familiarizarse con la sintaxis y las mejores prácticas de Kotlin.

Conclusión

Este módulo proporciona una base sólida en los fundamentos de Kotlin, esenciales para el desarrollo de aplicaciones Android modernas. Al comprender cómo estructurar programas, definir variables y crear funciones, los desarrolladores pueden comenzar a construir aplicaciones más complejas y funcionales utilizando Jetpack Compose.

2. Ruta de aprendizaje 2

Introducción

Este documento proporciona una guía detallada para instalar y configurar Android Studio, crear tu primer proyecto y ejecutar la aplicación en un dispositivo o emulador. Estos pasos son fundamentales para comenzar a desarrollar aplicaciones Android utilizando Jetpack Compose.

Instalación de Android Studio

Para comenzar, es necesario descargar e instalar Android Studio:

1. Visita el sitio oficial de Android Studio: https://developer.android.com/studio.
2. Selecciona la versión adecuada para tu sistema operativo (Windows, macOS o Linux).
3. Descarga el instalador y sigue las instrucciones proporcionadas para completar la instalación.

Creación de tu primer proyecto

Una vez instalado Android Studio, puedes crear tu primer proyecto:

1. Abre Android Studio.
2. En la pantalla de bienvenida, selecciona `Start a new Android Studio project`.
3. Elige la plantilla `Empty Compose Activity` para utilizar Jetpack Compose.
4. Configura el nombre del proyecto, el dominio y la ubicación del proyecto.
5. Selecciona el lenguaje `Kotlin` y la versión mínima de SDK.
6. Haz clic en `Finish` para crear el proyecto.

Ejecución de la aplicación

Para ejecutar la aplicación en un dispositivo o emulador:

1. Conecta un dispositivo Android a tu computadora o configura un emulador en Android Studio.
2. Haz clic en el botón `Run` (el ícono de un triángulo verde) en la barra de herramientas.
3. Selecciona el dispositivo o emulador en el que deseas ejecutar la aplicación.
4. Android Studio compilará y ejecutará la aplicación en el dispositivo seleccionado.

Conclusión

Con estos pasos, has instalado y configurado Android Studio, creado tu primer proyecto y ejecutado la aplicación en un dispositivo o emulador. Estos son los primeros pasos esenciales para comenzar a desarrollar aplicaciones Android utilizando Jetpack Compose.

3. Ruta de aprendizaje 3

Introducción

Esta documentación cubre los puntos 3 al 6 de la ruta de aprendizaje *Crea un diseño básico* del curso *Android Basics with Compose*. Se enfoca en la construcción de interfaces de usuario simples utilizando Jetpack Compose, incluyendo el uso de composables de texto, incorporación de imágenes, prácticas de conceptos básicos y la creación de una aplicación de tarjeta de presentación.

Construye una aplicación simple con composables de texto

En esta sección, se aprende a utilizar composables de texto para mostrar mensajes en la interfaz de usuario.

Objetivos

- Comprender el uso del composable `Text`.
- Organizar múltiples elementos de texto con `Column` y `Row`.
- Aplicar modificadores para personalizar la apariencia del texto.

Ejemplo de código

```
@Composable
fun Greeting() {
    Column {
        Text(
            text = " Feliz   cumplea os!",
            fontSize = 24.sp,
            modifier = Modifier.padding(16.dp)
        )
        Text(
            text = "De parte de tu amigo",
            fontSize = 16.sp,
            modifier = Modifier.padding(start = 16.dp)
```

```
        )
    }
}
```

Listing 4: Uso básico de composables de texto

Explicación del código

El código anterior utiliza dos composables de tipo `Text`, los cuales se muestran dentro de un composable `Column`:

- `Column` es un contenedor que organiza sus hijos de manera vertical.
- El primer `Text` muestra el mensaje "¡Feliz cumpleaños!çon un tamaño de fuente de 24sp y un relleno de 16dp.
- El segundo `Text` muestra el mensaje "De parte de tu amigoçon un tamaño de fuente de 16sp y un relleno en el lado izquierdo de 16dp.

Agrega imágenes a tu aplicación de Android

Esta sección enseña cómo incorporar imágenes en la interfaz de usuario utilizando composables de imagen.

Objetivos

- Utilizar el composable `Image` para mostrar imágenes.
- Cargar imágenes desde recursos locales.
- Combinar texto e imágenes en la interfaz.

Ejemplo de código

```
@Composable
fun BirthdayCard() {
    Column {
        Image(
            painter = painterResource(R.drawable.birthday_image),
            contentDescription = "Imagen de cumplea os",
            modifier = Modifier.fillMaxWidth()
        )
        Text(
            text = " Feliz  cumplea os!",
            fontSize = 24.sp,
            modifier = Modifier.padding(16.dp)
        )
```

```
    }
}
```

Listing 5: Incorporación de una imagen en la interfaz

Explicación del código

En este código:

- Se usa el composable `Image` para cargar y mostrar una imagen desde los recursos locales de la aplicación (con `R.drawable.birthday`$_{i}mage$).El`contentDescription`$proporcionaunadescripci$
- El modificador `fillMaxWidth()` hace que la imagen ocupe todo el ancho disponible.
- Después de la imagen, se agrega un texto con el mensaje "¡Feliz cumpleaños!".

Práctica: Conceptos básicos de Compose

En esta práctica, se aplican los conceptos aprendidos para construir interfaces más complejas y reforzar el conocimiento adquirido.

Objetivos

- Aplicar composables de texto e imagen en conjunto.
- Utilizar disposiciones como `Column` y `Row` para organizar la interfaz.
- Personalizar la apariencia utilizando modificadores.

Ejemplo de código

```
@Composable
fun PracticeCard() {
    Row(modifier = Modifier.padding(16.dp)) {
        Image(
            painter = painterResource(R.drawable.profile_picture),
            contentDescription = "Foto de perfil",
            modifier = Modifier.size(64.dp)
        )
        Column(modifier = Modifier.padding(start = 16.dp)) {
            Text(text = "Nombre del usuario", fontSize = 20.sp)
            Text(text = "Descripci n breve", fontSize = 14.sp)
        }
    }
}
```

Listing 6: Combinación de texto e imagen en una interfaz

Explicación del código

En este ejemplo:

- Se utiliza un `Row` para colocar horizontalmente los elementos.
- Dentro del `Row`, se incluye una imagen (`Image`) con una foto de perfil, con un tamaño de 64dp.
- A continuación, se coloca una `Column` que contiene dos elementos de texto: el nombre del usuario y una descripción breve.
- Los modificadores de padding controlan el espaciado entre los elementos y el borde de la pantalla.

Proyecto: Crea una aplicación de tarjeta de presentación

En este proyecto, se integran todos los conocimientos adquiridos para crear una aplicación completa que muestre una tarjeta de presentación.

Objetivos

- Diseñar una interfaz que muestre información de contacto.
- Utilizar composables para estructurar la información de manera clara.
- Aplicar estilos y temas para mejorar la apariencia de la aplicación.

Ejemplo de código

```
@Composable
fun BusinessCard() {
    Column(
        modifier = Modifier
            .fillMaxSize()
            .background(Color(0xFFE0F7FA))
            .padding(16.dp),
        horizontalAlignment = Alignment.CenterHorizontally
    ) {
        Image(
            painter = painterResource(R.drawable.profile_picture),
            contentDescription = "Foto de perfil",
            modifier = Modifier.size(100.dp)
        )
        Text(text = "Nombre del Profesional", fontSize = 24.sp,
           fontWeight = FontWeight.Bold)
        Text(text = "T tulo o Profesi n", fontSize = 18.sp, color
           = Color.Gray)
```

```
17          Divider(modifier = Modifier.padding(vertical = 8.dp))
18          ContactInfo(icon = Icons.Default.Phone, info = "+52 123 456
                7890")
19          ContactInfo(icon = Icons.Default.Email, info = "
                correo@ejemplo.com")
20      }
21  }
22
23  @Composable
24  fun ContactInfo(icon: ImageVector, info: String) {
25      Row(modifier = Modifier.padding(4.dp)) {
26          Icon(imageVector = icon, contentDescription = null)
27          Spacer(modifier = Modifier.width(8.dp))
28          Text(text = info)
29      }
30  }
```

Listing 7: Estructura de una tarjeta de presentación

Explicación del código

Este código crea una tarjeta de presentación con la siguiente estructura:

- Se utiliza un `Column` que ocupa todo el tamaño de la pantalla (`fillMaxSize()`) y tiene un fondo de color azul claro.
- Dentro de la columna, se muestra una imagen de perfil con un tamaño de 100dp.
- Después de la imagen, se muestra el nombre del profesional y su título o profesión con un tamaño de fuente adecuado. El nombre tiene un peso de fuente negrita (`FontWeight.Bold`).
- Se agrega un `Divider` (línea divisoria) para separar visualmente las secciones de la tarjeta.
- Luego se utiliza el composable `ContactInfo` para mostrar la información de contacto: número de teléfono y correo electrónico, acompañados de iconos representativos.
- El composable `ContactInfo` muestra una fila horizontal con un icono y el texto correspondiente.

Conclusión

A través de estos ejercicios, se ha aprendido a construir interfaces de usuario utilizando Jetpack Compose, incorporando texto e imágenes, y aplicando estilos y disposiciones para crear aplicaciones atractivas y funcionales.