92. 5.1 Navigation en Compose

https://developer.android.com/courses/pathways/android-basics-compose-unit-4-pathway-2?hl=es-419

Utilizamos clases de librería para navegar entre ventanas de una forma estandard.
Conocimientos previos:
* ViewModel * FlowState

92.1 Pasos para usar Navigation

1. Añadir dependencias.

```
implementation "androidx.navigation:navigation-compose:2.4.0-alpha10"
```

2. Crear un `NavController`

```
val navController = rememberNavController()
```

1. Definir destinos de navegación con `NavHost` Necesitamos el NavController creado en el punto anterior.

```
NavHost(navController, startDestination = "pantalla_inicio") {
    composable("pantalla_inicio") { PantallaInicio(navController) }
    composable("pantalla_detalle/{id}") { backStackEntry ->
        PantallaDetalle(navController, backStackEntry.arguments?.getString("id"))
    }
    // Agrega más destinos según sea necesario
}
```

Creamos un `composable()` para cada pantalla.

1. Navegar entre pantallas

Para ir a una pantall umo el método de NavController `navigate()`

```
Button(onClick = { navController.navigate("pantalla_detalle/$id") }) {
    Text("Ir a Detalles")
}
```

Manejo de Estado y Argumentos Pasar y Recibir Argumentos: Se pueden pasar argumentos a través de rutas, como se muestra en el ejemplo de pantalla_detalle/{id}.

Guardar Estado al Navegar: Para mantener el estado al navegar, considera usar ViewModels y SavedStateHandle.

Mejoras

Utilizar una **sealed class** para los textos de las rutas. El objetivo es mantener en un único lugar todas las rutas y usar constantes en lugar de textos en el código.

92.2 Uso de Navigation y ViewModel

Podemos usar un ViewModel para pasar argumentos entre ventanas siguiendo estos pasos:

1. Definimos sun ViewModel compartido entre las pantallas

```
class AppBarViewModel : ViewModel() {
    private val _titulo = MutableStateFlow("Inicio")
    val titulo = _titulo.asStateFlow()

    fun actualizarTitulo(nuevoTitulo: String) {
        _titulo.value = nuevoTitulo
    }
}
```

1. Usamos el viewmodel en las funciones componibles. Se actualiza el viewmodel dependiendo en la pantalla que nos encontremos.

```
@Composable
fun PantallaInicio(appBarViewModel: AppBarViewModel) {
    // Actualiza el título cuando esta pantalla se muestra
    LaunchedEffect(Unit) {
        appBarViewModel.actualizarTitulo("Inicio")
    }
    // Contenido de la pantalla
    // ...
}

@Composable
fun PantallaDetalle(appBarViewModel: AppBarViewModel, id: String) {
    // Actualiza el título para esta pantalla
    LaunchedEffect(Unit) {
        appBarViewModel.actualizarTitulo("Detalle: $id")
    }
    // Contenido de la pantalla
    // ...
}
```

1. Integramos el viewModel con la TopAppBar para actualizar el título actual

```
    @Composable
    fun MiTopAppBar(appBarViewModel: AppBarViewModel, navController: NavController) {
        val titulo = appBarViewModel.titulo.collectAsState()

        TopAppBar(
            title = { Text(text = titulo.value) },
            navigationIcon = {
                IconButton(onClick = { navController.navigateUp() }) {
                    Icon(Icons.Filled.ArrowBack, contentDescription = "Atrás")
                }
            }
        )
    }
```

1. Integramos todos los componentes con Scaffold

```
@Composable
fun MiAplicacion() {
    val navController = rememberNavController()
    val appBarViewModel = viewModel<AppBarViewModel>()

    Scaffold(
        topBar = { MiTopAppBar(appBarViewModel = appBarViewModel, navController =
navController) }
    ) {
        NavHost(navController = navController, startDestination = "inicio") {
            composable("inicio") { PantallaInicio(appBarViewModel) }
            composable("detalle/{id}") { backStackEntry ->
                PantallaDetalle(appBarViewModel,
backStackEntry.arguments?.getString("id") ?: "")
            }
            // y más rutas...
        }
    }
}
```

92.3 Apendice

Enlaces:

* Tutorial cupcake

Versión 0.5 , 12-12-23

¿Fue útil esta página?

☺ ☹