04-5. Navigation

Compose

NavHostController

내비게이션

- 컴포저블을 교체하면서 화면 전환
- 화면 전환을 하기 위해서는 navigation 라이브러리를 이용

• navigation 라이브러리 추가

```
implementation("androidx.navigation:navigation-compose:2.8.4")
```

- navigation 라이브러리의 핵심 클래스는 NavHost와 NavHostConstroller입니다.
- NavHost는 화면 단위의 컴포저블을 등록하고 화면을 스택 구조로 관리하는 역할
- NavHostController 는 실제 필요한 경우 화면 전환을 하기 위한 함수를 제공

NavHostController

- NavHostController 는 Jetpack Navigation Component 라이브러리에서 제공되는 API 인 NavController 의 하위 타입이다.
- Navigation 을 위한 대부분의 구현은 NavController 에 구현되어 있으며 NavHostController 는 아래의 4개 함수가 추가된 형태이다.

- enableOnBackPressed() : 시스템 뒤로가기 버튼에 반응할 것인지 설정
- setLifecycleOwner() : LifecycleOwner 를 지정
- setOnBackPressedDispatcher() : 시스템 뒤로가기 버튼 이벤트를 처리하기 위한 OnBackPressedDispatcher 설정
- setViewModelStore() : 탐색 대상간 ViewModel 관리

NavHostController

- 컴포즈에서는 두가지 종류의 NavHost 이용

- androidx.navigation.NavHost
- androidx.navigation.compose.NavHost

- 컴포즈에서 두 NavHost 모두 이용이 가능하지만 androidx.navigation.compose.NavHost 를 이용해야 navigation() 에 의한 설정이 가능하며 이를 이용해 컴포저블간 뷰모델을 공유시킬 수 있다.
- compose.NavHost 를 이용하려면 NavHostController 이어야 한다.
- 컴포즈에서는 NavHostController 를 이용하는 것이 좋다.

NavHostController

• navigation 라이브러리 추가

```
@Composable
fun MainScreen() {
    //NavController를 먼저 선언하고 NavController로 화면을 등록하는 NavHost를 만든다.
    val navController = rememberNavController()
    NavHost(navController = navController, startDestination = "home" ){
        composable("home"){ HomeScreen(navController) }
        composable("one"){ OneScreen(navController) }
        composable("two"){ TwoScreen(navController) }
    }
}
```

NavHostController

- NavHost에 등록된 컴포저블에서 화면 전환 명령을 내릴 때는 NavHost에 등록된 NavController의 navigate 함수를 호출하면서 매개변수로 이동하고자 하는 화면의 이름을 명시

• 화면 전환을 위한 navigate 함수 호출

```
Button(onClick = { navController.navigate("one") }) {
            Text("Go One")
}
```

NavHostController

이전 화면으로 이동

- 안드로이드의 '뒤로 가기' 버튼에 의한 이동은 별도로 처리하지 않아도 이전 화면 컴포저블로 잘 이동됩니다.
- 프로그램적으로 이전 화면으로 전 환하고 싶다면 NavController의 popBackStack 함수를 호출합니다.

• 이전 화면으로의 전환을 위한 popBackStack 함수 호출

```
Button(onClick = { navController.popBackStack() }) {
    Text("Go Back")
}
```

NavHostController

- 이전 컴포저블 화면이 아닌 특정 컴포저블 화면으 로 이동하고 싶다면 navigate() 함수를 호출하고 popUpTo() 함수로 옵션을 지정하면 됩니다.

• 특정 컴포저블 화면으로 이동하기 위한 popUpTo 함수 호출

```
Button(onClick = {
    navController.navigate("onesub"){
        popUpTo("home")
    }
}) {
    Text("Go OneSub")
}
```

NavHostController

데이터 전달

- 이름 에 전달해야 하는 데이터를 같이 명시해서 등록할 수 있습니다.

• composable 함수에 데이터 추가

```
composable("one/{userId}/{no}", arguments = listOf(navArgument("userId"){
    type = NavType.StringType
}, navArgument("no"){
    type = NavType.IntType
})){ navBackStackEntry ->
    OneScreen(
        navController,
        navBackStackEntry.arguments?.getString("userId"),
        navBackStackEntry.arguments?.getInt("no")
    )
}
```

• navigate 함수를 호출해 전달할 데이터 명시

```
Button(onClick = { navController.navigate("one/kkang/20") }) {
    Text("Go One")
}
```

NavHostController

- 이전 화면으로 되돌아갈 때 현재 화면에서의 결과 데이터를 전달하겠다면 popBackStack 함수 실행 전에 다음과 같이 set 함수를 이용해 결과 데이터를 먼저 담으면 됩니다..

• set 함수로 결과 데이터를 담아 전달

```
Button(onClick = {
    //이전 화면으로 되돌아가면서 데이터 전달
    navController.previousBackStackEntry?.savedStateHandle?.set("msg", "kim")
    navController.popBackStack()
}) {
    Text("Go Back")
}
```

NavHostController

- 받는 곳에서도 동일한 이름으로 획득하면 됩니다.

• 결과 데이터 획득

```
val msg =
    navController.currentBackStackEntry?.savedStateHandle?.get<String>("msg")
```

감사합니다

단단히 마음먹고 떠난 사람은
산꼭대기에 도착할 수 있다.
산은 올라가는 사람에게만 정복된다.

윌리엄 셰익스피어
William Shakespeare