Eureka Xpert By Eureka Edutech talenthub

"Navigation Component dan Error Handling"

Ikrima Amanda

Navigation
Component

Deskripsi

- Bagian dari Android Jetpack yang menyediakan kerangka kerja untuk mengelola navigasi dalam aplikasi Android.
- Mempermudah developer untuk mengatur dan mengelola tindakan navigasi seperti perpindahan antar-fragment atau antar-aktivitas, melalui konfigurasi yang didefinisikan secara grafis atau melalui kode.
- Komponen Utama:
    - NavGraph (graf navigasi) => mendefinisikan graf navigasi yang terdiri dari tujuan-tujuan (destinations) dan tindakan-tindakan (actions) antara tujuan-tujuan tersebut
    - NavController (pengontrol navigasi) => mengatur navigasi antar tujuan-tujuan berdasarkan NavGraph
    - NavHost (tuan rumah navigasi) => sebagai wadah tempat tujuan-tujuan tersebut ditampilkan

Langkah menggunakan Navigation Component

- Menambahkan dependensi
- Menambahkan folder Navigation
  - klik kanan pada folder res > New > Android Resource Directory > pilih resource type => navigation > OK
- Membuat NavGraph
  - klik kanan pada folder navigation > New > Navigation Resource File > beri nama dgn style snake_case > OK
  - Jika belum menambahkan dependensi akan muncul pop up warning
- Menyiapkan NavHost
  - Buat fragment yang dibutuhkan (Misal membuat HomeFragment dan ProfileFragment), tampilan sesuai dengan kebutuhan
  - Buka file XML dari activtiy
  - Tambahkan FragmentContainerView atau pilih design dan cari NavHostFragment dan masukkan ke tampilan activity
  - Buka file NavGraph yang sudah dbuat, tambahkan destinasi => HomeFragment dan ProfileFragment
  - atur action yang dibutuhkan untuk tiap fragment

Langkah menggunakan Navigation Component (2)

- Menginisialisasi NavController
  - Buka file Kotlin milik activity
  - inisialisasi navController dan navHostFragment
- Menavigasi antara tujuan-tujuan
  - Buka file HomeFragment dan ProfileFragment
  - Atur button menuju tujuan berdasarkan action yang telah disiapkan sebelumnya
- Tambahkan pengaturan lain jika diperlukan seperti popUpTo atau popUpToInclusive

Eureka Xpert talenthub

Exceptions and Exception Handling

Exceptions

- kondisi abnormal yang terjadi saat program sedang berjalan. Ketika suatu kesalahan atau kondisi yang tidak diharapkan terjadi, exception akan dilempar (thrown) oleh kode yang sedang dieksekusi. Jika exception tidak ditangkap dan diatasi, program akan menghentikan eksekusi dan menampilkan informasi tentang exception tersebut.
- Jenis Exceptions
    - Checked Exceptions => exception yang diketahui ketika compile time
    - Unchecked Exceptions => exception yang terjadi ketika run time
- Compile time => proses ketika kode program diterjemahkan dari bahasa program tingkat tinggi ke bahasa mesin yang dapat dimengerti oleh komputer. Contoh : kesalahan penulisan syntax, penggunaan variabel yang tidak dideklarasikan, salah dalam memanggil method.
- Run time => proses dimana kode program dijalankan pada mesin komputer setelah berhasil melalui compile time (tahap penerjemahan). Kesalahan runtime terjadi saat program dijalankan dan berinteraksi dengan data aktual, input pengguna, atau lingkungan eksekusi. Kesalahan runtime dapat berupa exception, seperti NullPointerException atau ArrayIndexOutOfBoundsException, yang terjadi ketika kesalahan logika program atau kondisi yang tidak diharapkan muncul saat program berjalan.

Exceptions pada Kotlin

- Dalam Kotlin, tidak ada konsep checked exceptions seperti yang ada dalam bahasa Java.
- Semua exception termasuk dalam kategori unchecked exceptions.
- Contoh umum dari unchecked exceptions:
    - NullPointerException: Terjadi ketika Anda mencoba mengakses objek yang memiliki referensi null.
    - IndexOutOfBoundsException: Terjadi ketika Anda mengakses indeks yang berada di luar rentang yang valid dalam suatu array atau koleksi.
    - ArithmeticException: Terjadi ketika terjadi kesalahan dalam operasi aritmatika, seperti pembagian dengan nol.

Exception Handling

- mekanisme dalam Kotlin (dan bahasa pemrograman lainnya) yang digunakan untuk menangani exception yang terjadi saat program dijalankan. Exception handling membantu untuk mengambil tindakan yang tepat ketika kesalahan atau situasi yang tidak diharapkan terjadi dalam kode program.
- Exceptions handling dilakukan menggunakan blok try-catch yang memiliki format sebagai berikut:

```
try {
    // Kode yang berpotensi melemparkan pengecualian
} catch (exceptionType: Exception) {
    // Penanganan pengecualian yang sesuai dengan jenis pengecualian
} finally {
    // Blok finally opsional yang akan dijalankan terlepas dari apakah
    // pengecualian terjadi atau tidak
}
```

try-catch

- **Blok try** => menempatkan kode yang berpotensi melemparkan exception. Ketika exception terjadi dalam blok try, eksekusi program akan segera beralih ke blok catch yang sesuai.
- **Blok catch** => menentukan logika penanganan yang sesuai, seperti menampilkan pesan kesalahan atau melakukan tindakan lain yang diperlukan.
- **Blok finally** => Blok finally adalah blok opsional yang dapat digunakan untuk menentukan kode yang harus dijalankan terlepas dari apakah pengecualian terjadi atau tidak. Blok finally biasanya digunakan untuk membersihkan sumber daya atau menjalankan tindakan terakhir yang perlu dilakukan sebelum keluar dari blok try-catch.

Tab Layout and View Pager

Tab Layout dan View Pager

- Dua komponen yang sering digunakan bersama-sama dalam pengembangan aplikasi Android dengan Kotlin untuk menyediakan navigasi antara beberapa tampilan (fragmen) yang terkait.
- Tab Layout
  - Komponen UI yang menyediakan antarmuka pengguna untuk menampilkan dan mengelola tab-tab navigasi.
  - Berisi judul atau ikon yang mewakili tampilan atau fragmen yang terkait.
  - Tab dapat diklik untuk beralih antara tampilan yang berbeda.
  - TabLayout biasanya ditempatkan di atas ViewPager.
- View Pager
  - Komponen yang digunakan untuk mengelola tampilan fragmen yang dapat digulir secara horizontal.
  - User dapat menggeser layar ke kiri atau ke kanan untuk beralih antara fragmen-fragmen yang terkait.
  - Fragmen yang ditampilkan dalam ViewPager dapat berubah sesuai dengan tab yang dipilih di TabLayout.
  - ViewPager bekerja dengan adaptor yang menyediakan fragmen-fragmen yang akan ditampilkan.

Langkah membuat Tab Layout dan View Pager

- Menambahkan dependensi

```kotlin
dependencies {
    implementation 'com.google.android.material:material:1.4.0'
}
```

- Membuat XML Layout
  - Menyiapkan activity yang memiliki tab layout dan view pager
  - Menyiapkan fragment atau beberapa tampilan yang dibutuhkan
- Membuat Pager Adapter
- Inisialisasi Tab Layout dan View Pager

Referensi

- https://developer.android.com/guide/navigation/get-started?hl=id
- https://developer.android.com/guide/health-and-fitness/health-connect/common-workflows/exceptions?hl=id
- https://developer.android.com/guide/navigation/navigation-swipe-view?hl=id

Tugas

Tampilan Tab Sederhana

- Terdapat splash screen aplikasi
- Terdapat 2 Tab (Peserta A dan Peserta B) => data peserta A dan B harus berbeda
- Dibuat menggunakan Tab Layout dan View Pager
- Tiap tab menampilkan list pengguna yang ditampilkan menggunakan recyclerview
- Data pengguna mencakup => Photo Profile, Name, Email, Jurusan, Semester
- Item pengguna dapat diklik dan berpindah ke halaman baru dengan detail data dari pengguna yang diklik/dipilih sebelumnya.
- Total ada 3 tampilan => splash screen, 2 tab dari list pengguna, dan detail pengguna.
- Dikumpulkan dalam bentuk **Link Google Drive (Berisi link repo GitHub dan APK)**
- Deadline pengumpulan tugas => **9 Juli 2023 12.00 WIB**
- **Link pengumpulan tugas** => https://forms.gle/5FC5VmZSgHB82zRM9

Stuck?

talk with ur best friend...
chatgpt, google, stack over flow, medium

The Most Important Thing…

https://www.instagram.com/reel/CtoupCfADAv/?igshid=Y2IzZGU1MTFhOQ%3D%3D

“Practice doesn’t make perfect. Perfect practice makes perfect”

- Vince Lombardi

Terima kasih!

telegram : @ikrimaamandaa

LinkedIn : https://www.linkedin.com/in/ikrimaamanda/