
concrete
PART OF ACCENTURE

Por que eu devo usar Android Navigation 🤔?

Diego Nascimento

Android Developer at concrete
@diego_figue

sli.do

/navigation

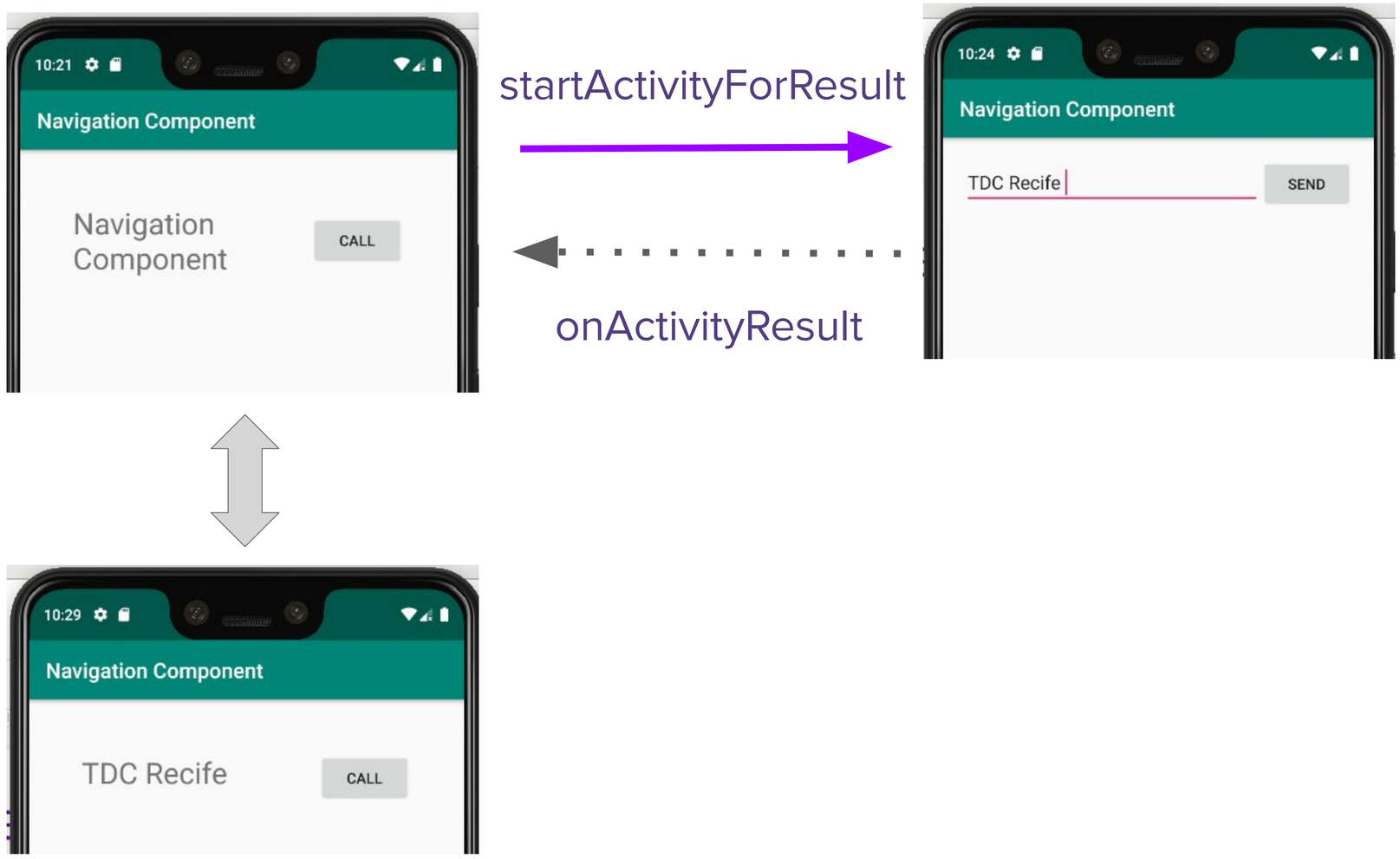
Quando queremos navegar para uma nova tela do nosso app

- StartActivity

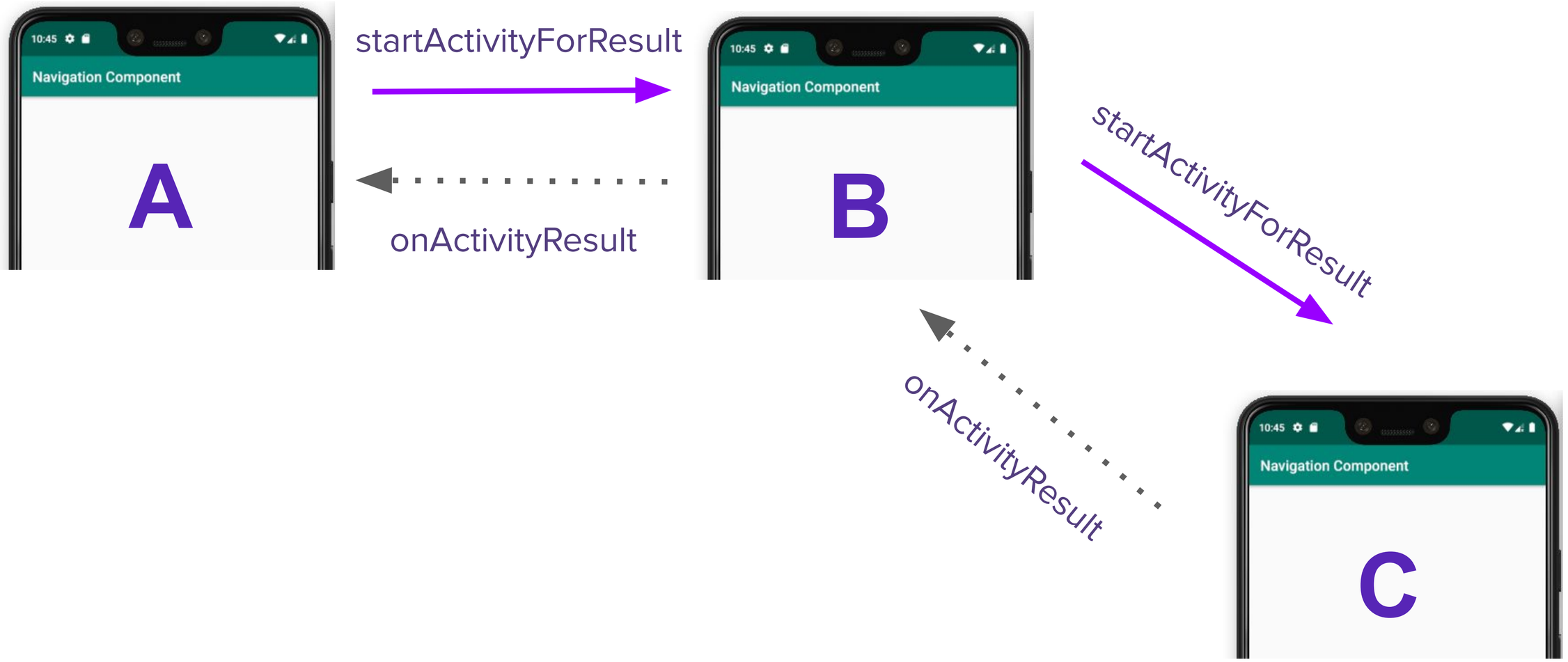
startActivity

```
const val EXTRA_MESSAGE = "br.com.dfn.navigationcomponent.MESSAGE"

class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }

    fun sendMessage(view: View) {
        val editText = findViewById<EditText>(R.id.editText)
        val message = editText.text.toString()
        val intent = Intent(this, DisplayMessageActivity::class.java).apply {
            putExtra(EXTRA_MESSAGE, message)
        }
        startActivity(intent)
    }
}
```

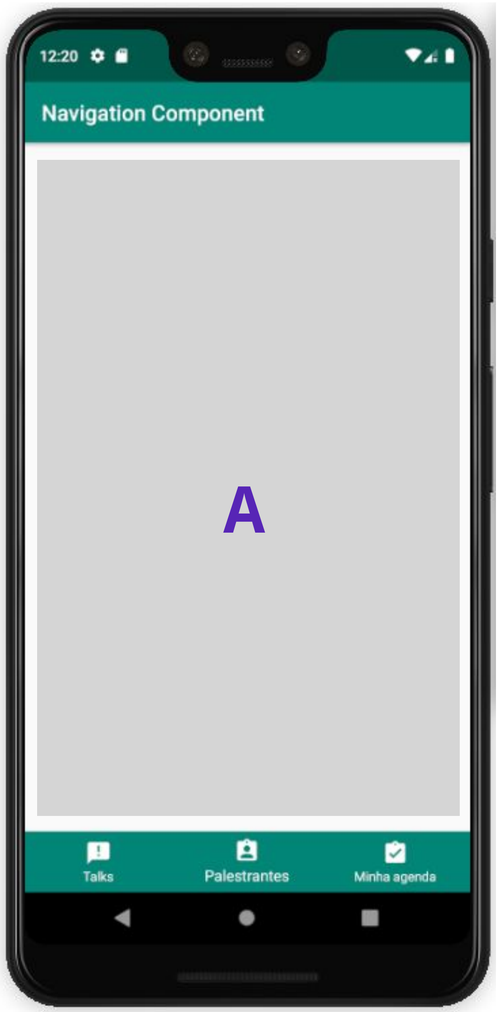
Quando precisamos pegar um resultado de outra tela



E se precisarmos navegar por mais de uma tela para obter o resultado ?



Como criamos uma navegação com viewPager e BottomNavigation ?



B

C

Fragment

fragment

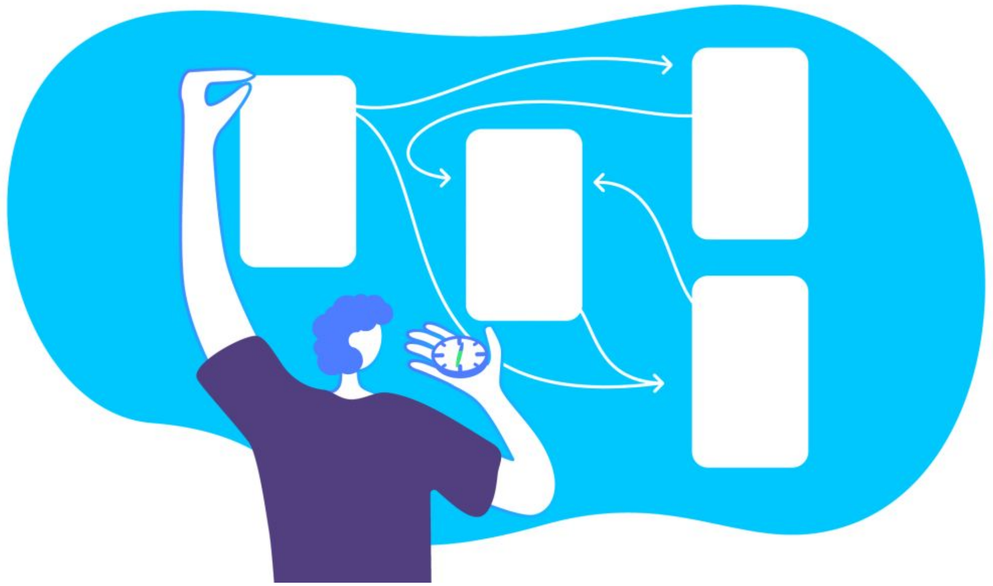
```
class SampleFragmentActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_sample_fragment)

        supportFragmentManager.beginTransaction()
            .replace(R.id.container, TalksFragment())
            .commit()

        val bottomNavigation = findViewById<BottomNavigationView>(R.id.bottom_navigation)
        bottomNavigation.setOnNavigationItemSelectedListener { menu ->
            val fragment: Fragment
            when (menu.itemId) {
                R.id.talks -> { fragment = TalksFragment() }
                R.id.speakers -> {  fragment = SpeakersFragment()  }
                else -> {  fragment = MyAgendaFragment() }
            }
            supportFragmentManager.beginTransaction().replace(R.id.container, fragment)
                .addToBackStack(fragment.toString())
                .commit()
            true
        }
    }
```

fragment

```
override fun onBackPressed() {
    if (!supportFragmentManager.popBackStackImmediate()) {
        super.onBackPressed()
    }
}
```

Então quando temos um padrão de navegação simples usamos **Activities** e quando temos um padrão mais complexo usamos a navegação por **Fragments** ?

Navigation Component



Navigation graph

NavHostFragment

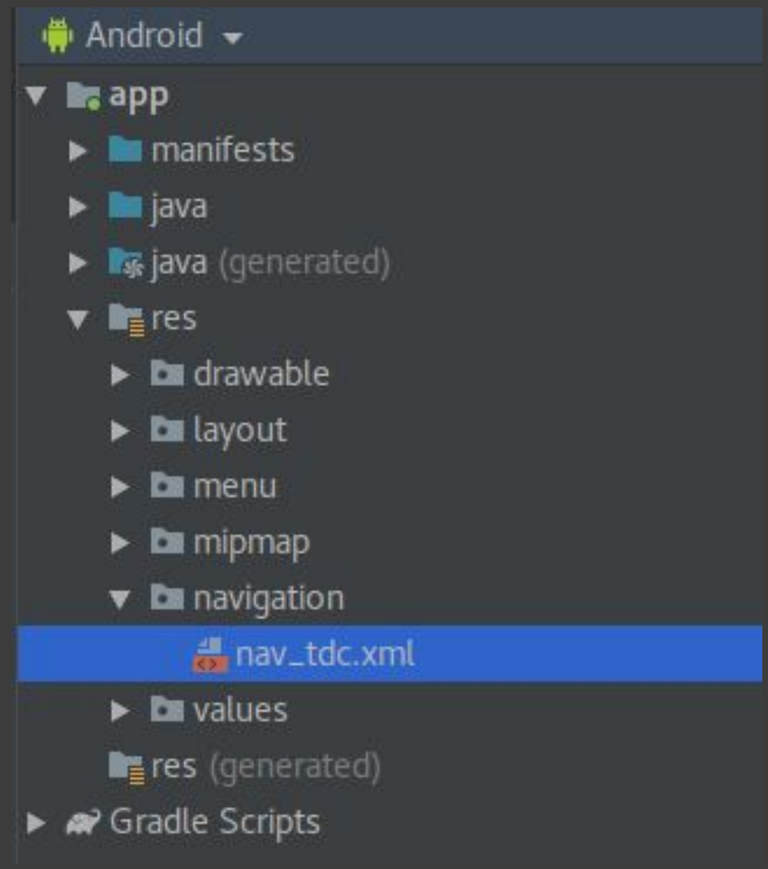
NavControler

Adicionando a dependência

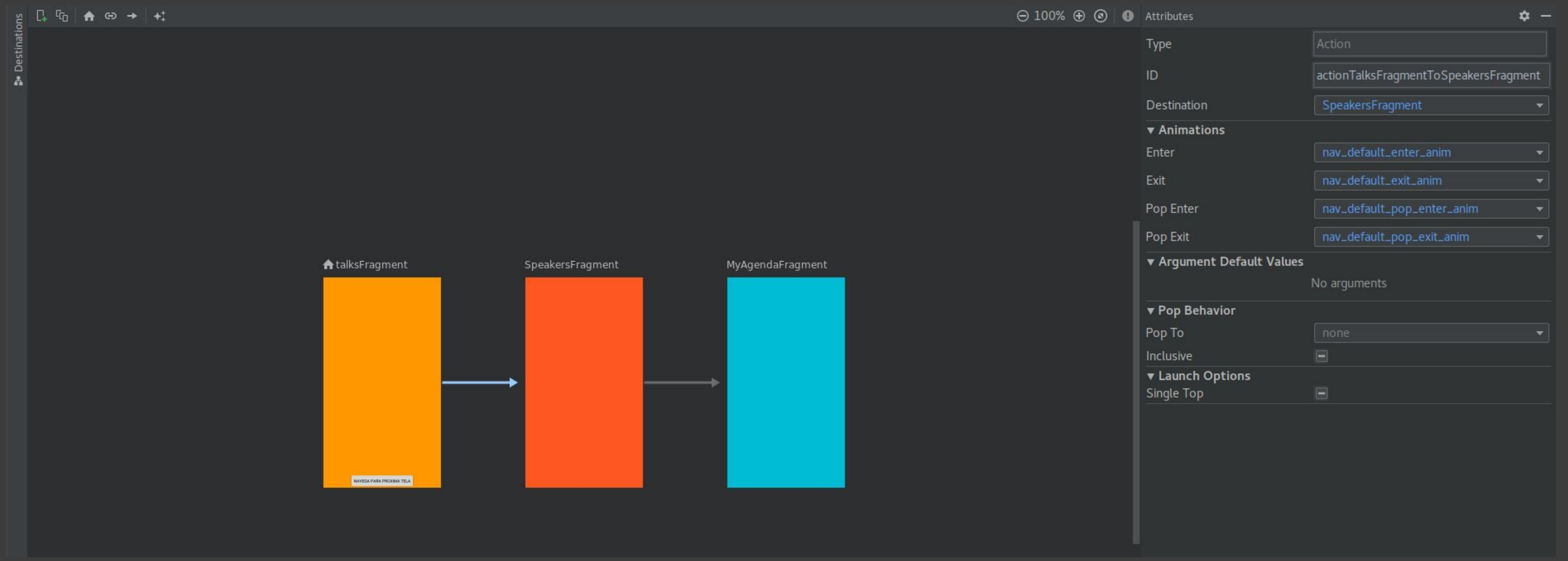
```
dependencies {
  def nav_version = "2.1.0"

  // Java
  implementation "androidx.navigation:navigation-fragment:$nav_version"
  implementation "androidx.navigation:navigation-ui:$nav_version"

  // Kotlin
  implementation "androidx.navigation:navigation-fragment-ktx:$nav_version"
  implementation "androidx.navigation:navigation-ui-ktx:$nav_version"

}
```

NavGraph

```xml
<?xml version="1.0" encoding="utf-8"?>
<navigation xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/nav_sample"
    app:startDestination="@id/talksFragment">

    <fragment
        android:id="@+id/talksFragment"
        android:name="br.com.dfn.navigationcomponent.ui.fragment.TalksFragment"
        android:label="TalksFragment"
        tools:layout="@layout/fragment_talks">
        <action
            android:id="@+id/actionTalksFragmentToSpeakersFragment"
            app:destination="@id/SpeakersFragment"" />
    </fragment>

    <fragment
        android:id="@+id/SpeakersFragment"
        android:name="br.com.dfn.navigationcomponent.ui.fragment.SpeakersFragment"
        android:label="SpeakersFragment"
        tools:layout="@layout/fragment_speakers" />
</navigation>
```



NavGraph


Destinations
100%
talksFragment
SpeakersFragment
MyAgendaFragment
Attributes
Type
Action
ID
actionTalksFragmentToSpeakersFragment
Destination
SpeakersFragment
Animations
Enter
nav_default_enter_anim
Exit
nav_default_exit_anim
Pop Enter
nav_default_pop_enter_anim
Pop Exit
nav_default_pop_exit_anim
Argument Default Values
No arguments
Pop Behavior
Pop To
none
Inclusive
Launch Options
Single Top

NavHostFragment

```
<fragment
   android:id="@+id/navHost"
   android:name="androidx.navigation.fragment.NavHostFragment"
   android:layout_width="match_parent"
   android:layout_height="0dp"
   android:layout_weight="1"
   app:defaultNavHost="true"
   app:navGraph="@navigation/nav_tdc" />
```

NavController

```
// Activity
private val navController by lazy { findNavController(R.id.navHost) }

// Fragment
private val navController by lazy { findNavController() }


// Navigation
navController.navigate(R.id.actionTalksFragmentToSpeakersFragment)
```

Testando a navegação

```
@RunWith(AndroidJUnit4::class)
class TitleScreenTest {

@Test
override fun testNavigationToInGameScreen() {
    val mockNavController = mockk<NavController>()
    val titleScenario = launchFragmentInContainer<TitleScreen>()

    titleScenario.OnFragment{ fragment ->
      Navigation.setViewNavController(fragment.requireView(), mockNavController)
    }

   onView(ViewMatchers.withId(R.id.play_btn)).perform(ViewActions.click())
   verify(mockNavController).navigate(R.id.action_title_screen_to_in_game)
}
```

Quais são as vantagens de usar o Navigation Component ?

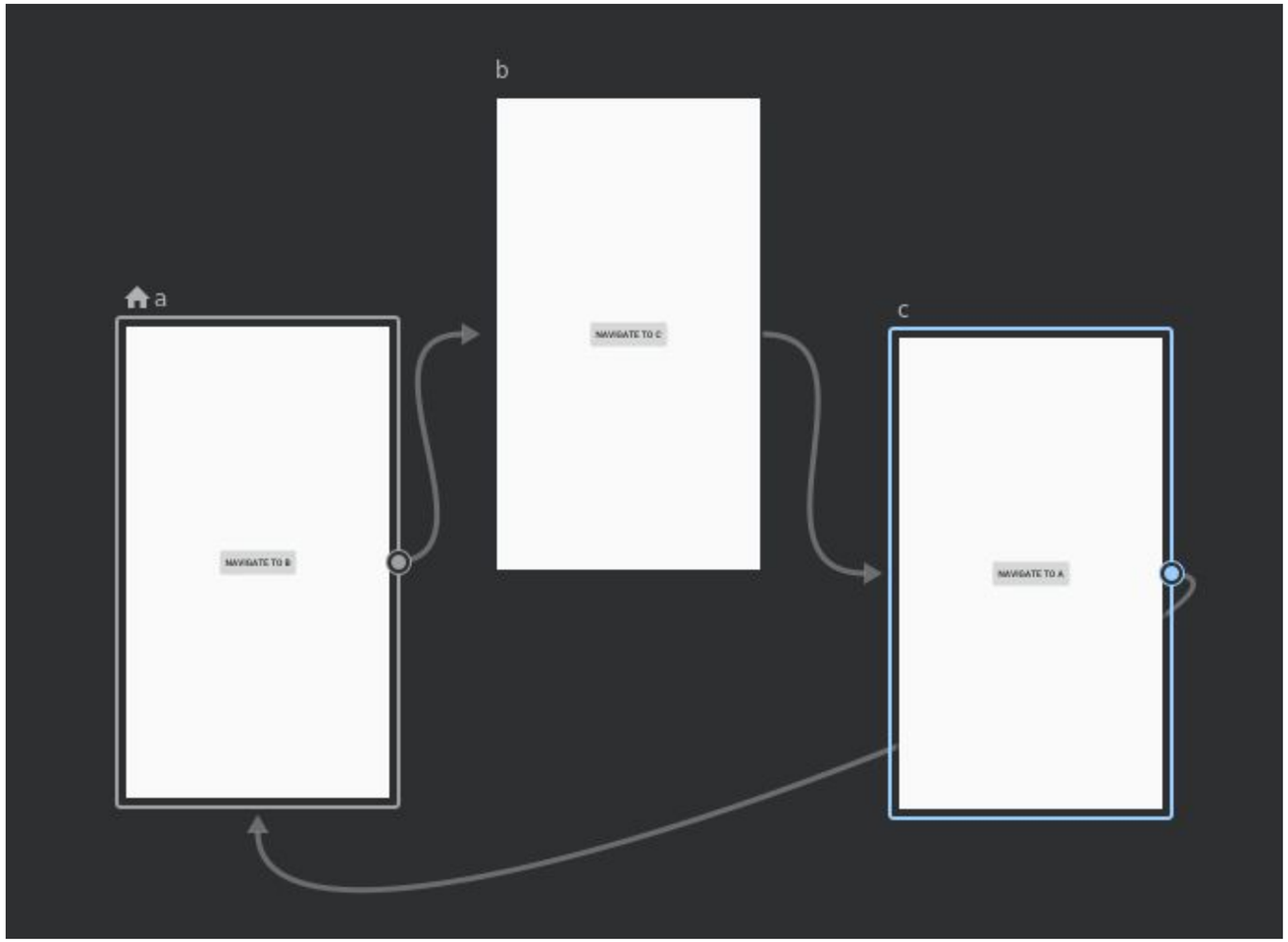
Simplifica a configuração de padrões comuns de navegação.

1. Simples Navigation
2. Options Menus
3. Bottom Navigation
4. Navigation View
5. Navigation Drawer
6. Toolbar
7. ActionBar
8. Collapsing Toolbar

Lida facilmente com o backstack

- popUpTo

  popUpToInclusive



Safe Args

Safe args - plugin

```
// Top-level build file where you can add configuration options common to all sub-projects/modules.

  dependencies {
   classpath "com.android.tools.build:gradle:${versions.android_plugin}"
   classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$versions.kotlin"
   classpath "androidx.navigation:navigation-safe-args-gradle-plugin:$versions.androidx.navigation"
   classpath "com.google.gms:google-services:${versions.firebase.version}"
  }


// Module build.gradle

  apply plugin: 'com.android.application'
  apply plugin: 'kotlin-android'
  apply plugin: 'kotlin-kapt'
  apply plugin: 'androidx.navigation.safeargs'
  apply plugin: 'kotlin-android-extensions'

  android {

    ...
  }
```

Safe args

```
<navigation xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/profile_nav_graph"
   app:startDestination="@id/profileFragment">
 <fragment
     android:id="@+id/profileFragment"
     android:name="br.com.dfn.presentation.features.profile.home.ProfileFragment"
     android:label="ProfilFragment"
     tools:layout="@layout/fragment_profile">
   <action
       android:id="@+id/goToEditNameFragment"
      app:destination="@+id/editNameFragment"
      app:enterAnim="@anim/toolkit_slide_in"
      app:exitAnim="@anim/toolkit_slide_out"
      app:popEnterAnim="@anim/toolkit_slide_pop_out"
      app:popExitAnim="@anim/toolkit_slide_pop_in"/>
 </fragment>

  ...
</navigation>
```

Safe args

```
// Origin
val directions = ProfileFragmentDirections.goToEditNameFragment("nascimento.diegof@gmail.com")
navController.navigate(directions)


// Destination
private val args by navArgs<EditNameFragmentArgs>()
args.email
```

Simplificando o Deeplink

Criando um deep link de forma explicita

```
val deeplink = navController.createDeepLink()
 .setDestination(R.id.editPhoneFragment)
 .setArguments(args)
 .createPendingIntent()

val notificationManager = context!!.getSystemService(Context.NOTIFICATION_SERVICE) as NotificationManager

if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.O) {
 notificationManager.createNotificationChannel(
   NotificationChannel(channelId, "Deep Links", NotificationManager.IMPORTANCE_HIGH)
 )
}

val builder = NotificationCompat.Builder(context!!, channelId)
 .setContentTitle("Navigation")
 .setContentText("Deep link to Android")
 .setSmallIcon(R.drawable.ic_star_icon)
 .setContentIntent(deeplink)
 .setAutoCancel(true)
notificationManager.notify(0, builder.build())
```

Criando um deep link de forma implicita

```
<fragment
   android:id="@+id/trilhaFragment"
   android:name="br.com.dfn.tdc.features.trilha.TrilhaFragment"
   android:label="TrilhaFragment"
   tools:layout="@layout/fragment_trilha">
 <deepLink
     android:id="@+id/deepLinkTrilha"
     app:uri="http://www.thedevelopersconference.com.br/tdc/2019/recife/{trilhaId}" />
</fragment>


<activity
   android:name="br.com.dfn.tdc.features.MainActivity"
   android:label="@string/app_name"
   android:theme="@style/AppTheme.NoActionBar">
 <nav-graph android:value="@navigation/app_nav"/>
</activity>
```

Transitions Animations

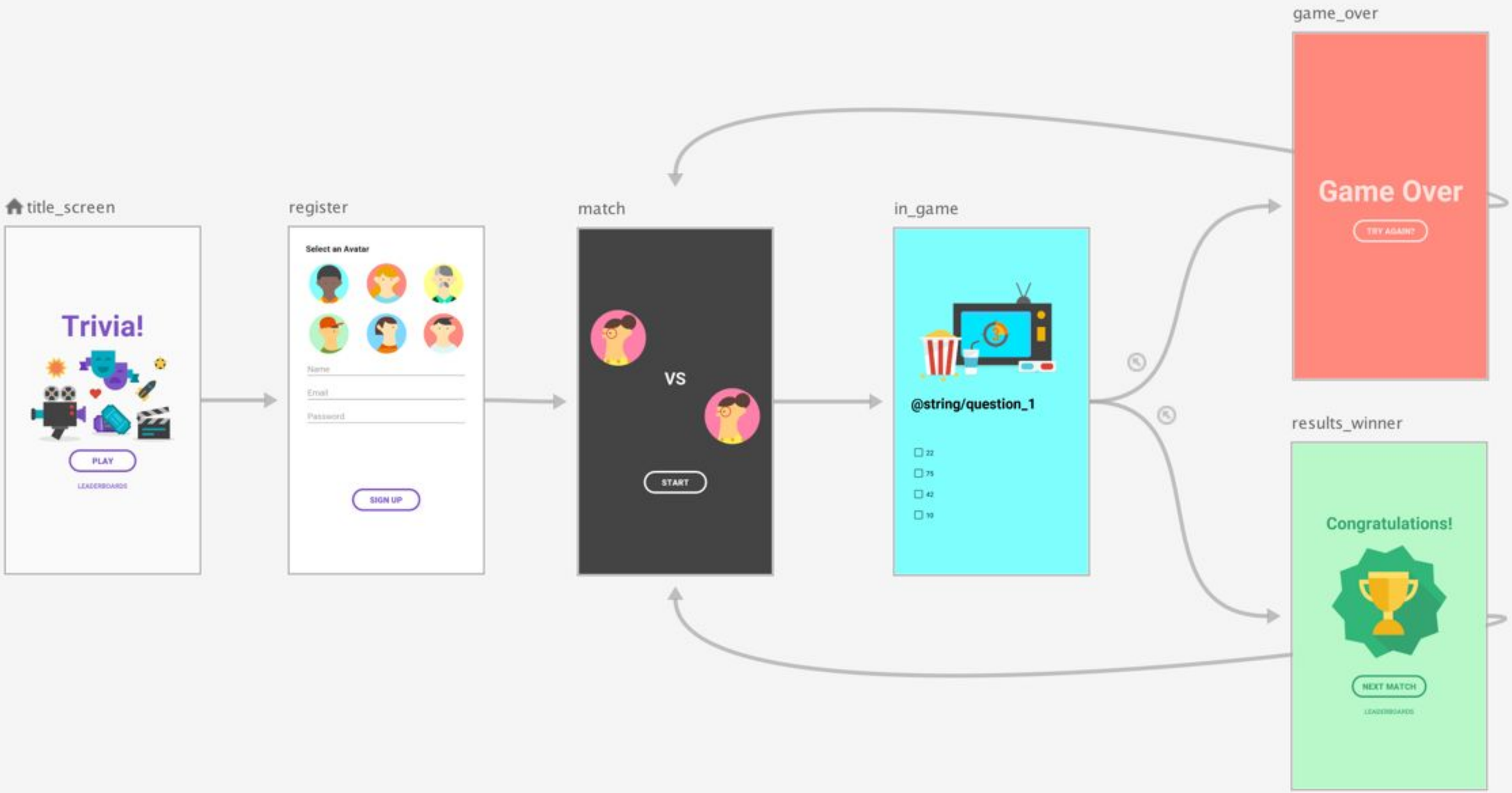
Transitions Animations

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android">
  <translate
      android:duration="@integer/toolkit_animation_duration_default"
      android:fromXDelta="100%"
      android:interpolator="@android:interpolator/accelerate_decelerate"
      android:toXDelta="0%" />
</set>
```

Transitions Animations

```
<navigation xmlns:android="http://schemas.android.com/apk/res/android"
   xmlns:app="http://schemas.android.com/apk/res-auto"
   xmlns:tools="http://schemas.android.com/tools"
   android:id="@+id/profile_nav_graph"
   app:startDestination="@id/profileFragment">
 <fragment
     android:id="@+id/profileFragment"
     android:name="br.com.dfn.presentation.features.profile.ProfileFragment"
     android:label="ProfilFragment"
     tools:layout="@layout/fragment_profile">
   <action
       android:id="@+id/goToEditNameFragment"
       app:destination="@+id/editNameFragment"
        app:enterAnim="@anim/toolkit_slide_in"
        app:exitAnim="@anim/toolkit_slide_out"
        app:popEnterAnim="@anim/toolkit_slide_pop_out"
        app:popExitAnim="@anim/toolkit_slide_pop_in"/>
   </fragment>
…

 </navigation>
```

Visualização completa da navegação


title_screen
Trivia!
PLAY
LEADERBOARDS
register
Select an Avatar
Name
Email
Password
SIGN UP
match
VS
START
in_game
@string/question_1
22
75
42
10
game_over
Game Over
TRY AGAIN?
results_winner
Congratulations!
NEXT MATCH
LEADERBOARDS

Referências

1. https://developer.android.com/guide/navigation
2. https://developer.android.com/guide/navigation/navigation-migrate

concrete
PART OF ACCENTURE

NÓS MOVEMOS O MUNDO.

RIO

Centro
Av. Presidente Wilson, 231
29º andar
(21) 2240-2030

SÃO PAULO

Cidade Monções
Av. Nações Unidas, 11.541
3º andar
(11) 4119-0449

BH

Savassi
Av. Getúlio Vargas, 671
Sala 800 - 8º andar
(31) 3360-8900

RECIFE

Ilha do Leite
Rua Sen. José Henrique, 199
2º andar
(81) 3018-6299

WWW.CONCRETE.COM.BR