

LAPORAN PRAKTIKUM

PEMROGRAMAN BERBASIS MOBILE

PERTEMUAN KE-7



Disusun Oleh :

NAMA : Raden Isnawan Argi Aryasatya
NIM : 195410257
JURUSAN : Informatika
JENJANG : S1
KELAS : 5

Laboratorium Terpadu
Sekolah Tinggi Manajemen Informatika Komputer
AKAKOM
YOGYAKARTA

2021

PERTEMUAN KE-7 **(MENDEFINISIKAN JALUR NAVIGASI)**

TUJUAN

Mahasiswa mampu mendefinisikan dan menggunakan navigasi dalam aplikasi

DASAR TEORI

1. Selamat datang

Menyusun pengalaman pengguna dalam menavigasi aplikasi selalu menjadi topik yang menarik bagi pengembang. Untuk aplikasi Android, Komponen Arsitektur Navigasi mempermudah penerapan navigasi. Tujuan adalah tempat mana pun di dalam aplikasi yang dapat dinavigasi oleh pengguna. Grafik navigasi untuk aplikasi terdiri dari sekumpulan tujuan dalam aplikasi. Grafik navigasi memungkinkan Anda menentukan dan menyesuaikan secara visual cara pengguna bernavigasi di antara tujuan dalam aplikasi Anda.

Apa yang akan Anda pelajari

- Cara menggunakan grafik navigasi
- Cara menentukan jalur navigasi (navigation path) di aplikasi Anda
- Apa itu tombol Naik, dan cara menambahkannya
- Cara membuat menu opsi
- Cara membuat panel samping navigasi

Apa yang akan Anda lakukan

- Buat grafik navigasi untuk Fragment Anda menggunakan pustaka navigasi dan Editor Navigasi.
- Buat jalur navigasi di aplikasi Anda.
- Tambahkan navigasi menggunakan menu opsi.
- Menerapkan tombol Naik sehingga pengguna dapat menavigasi kembali ke layar judul dari mana saja di aplikasi.
- Tambahkan menu panel navigasi.

2. Ikhtisar aplikasi

Aplikasi AndroidTrivia, yang Anda mulai kerjakan di modul praktikum sebelumnya, adalah game tempat pengguna menjawab pertanyaan tentang pengembangan Android. Jika pengguna menjawab tiga pertanyaan dengan benar, mereka memenangkan permainan. Jika Anda menyelesaikan modul praktikum sebelumnya, gunakan kode tersebut sebagai kode awal untuk modul praktikum ini. Jika tidak, unduh aplikasi **AndroidTriviaFragment** dari GitHub untuk mendapatkan kode awal.

Dalam modul praktikum ini, Anda memperbarui aplikasi AndroidTrivia dengan cara berikut:

- Anda membuat grafik navigasi untuk aplikasi.
 - Anda menambahkan navigasi untuk layar judul dan layar permainan.
 - Anda menghubungkan layar dengan tindakan, dan memberi pengguna cara untuk menavigasi ke layar game dengan mengetuk Play.
 - Anda menambahkan tombol Naik, yang ditampilkan sebagai panah kiri di bagian atas beberapa layar.
-

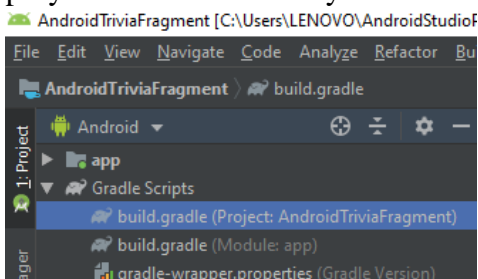
PRAKTIK

3. Tugas: Menambahkan komponen navigasi ke proyek

Langkah 1: Tambahkan dependensi navigasi

Komponen Navigation adalah library yang bisa mengelola navigasi kompleks, animasi transisi, deep linking, dan argumen centang waktu kompilasi yang lewat di antara layar dalam aplikasi Anda. Untuk menggunakan pustaka navigasi, Anda perlu menambahkan dependensi navigasi ke file Gradle Anda.

1. Untuk memulai, unduh aplikasi AndroidTriviaFragment starter atau gunakan salinan aplikasi AndroidTrivia Anda sendiri dari codelab sebelumnya. Buka aplikasi AndroidTrivia di Android Studio. (Proyek ini disertakan di ELA)
2. Di panel Project: Android, buka folder Gradle Scripts. Klik dua kali file build.gradle level proyek untuk membukanya.



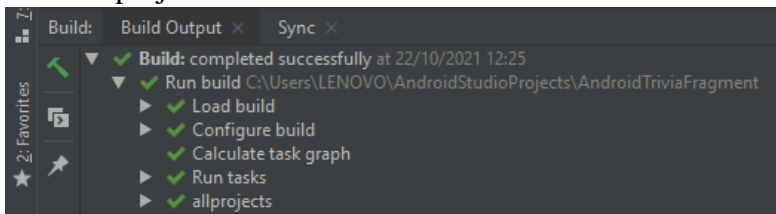
3. Di bagian atas file build.gradle level project, bersama dengan variabel ext lainnya, tambahkan variabel untuk **navigationVersion**. Untuk menemukan nomor versi navigasi terbaru.

```
ext {  
    kotlin_version = '1.4.10'  
    archLifecycleVersion = '1.1.1'  
    gradleVersion = '4.1.0'  
    supportlibVersion = '1.2.0'  
    dataBindingCompilerVersion = gradleVersion // Always need to be the same.  
    navigationVersion = '2.3.1'  
}
```

4. Di folder Gradle Scripts, buka file build.gradle level modul. Tambahkan dependensi untuk **navigation-fragment-ktx** dan **navigation-ui-ktx**, seperti yang ditunjukkan di bawah ini:

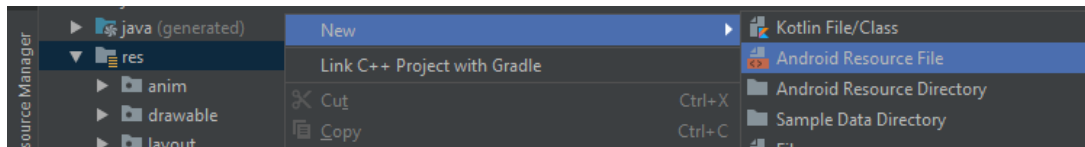
```
implementation "androidx.navigation:navigation-fragment-ktx:$navigationVersion"  
implementation "androidx.navigation:navigation-ui-ktx:$navigationVersion"
```

5. Rebuild project



Langkah 2: Tambahkan grafik navigasi ke proyek

1. Di panel Project: Android, klik kanan folder res dan pilih **New > Android Resource File**.



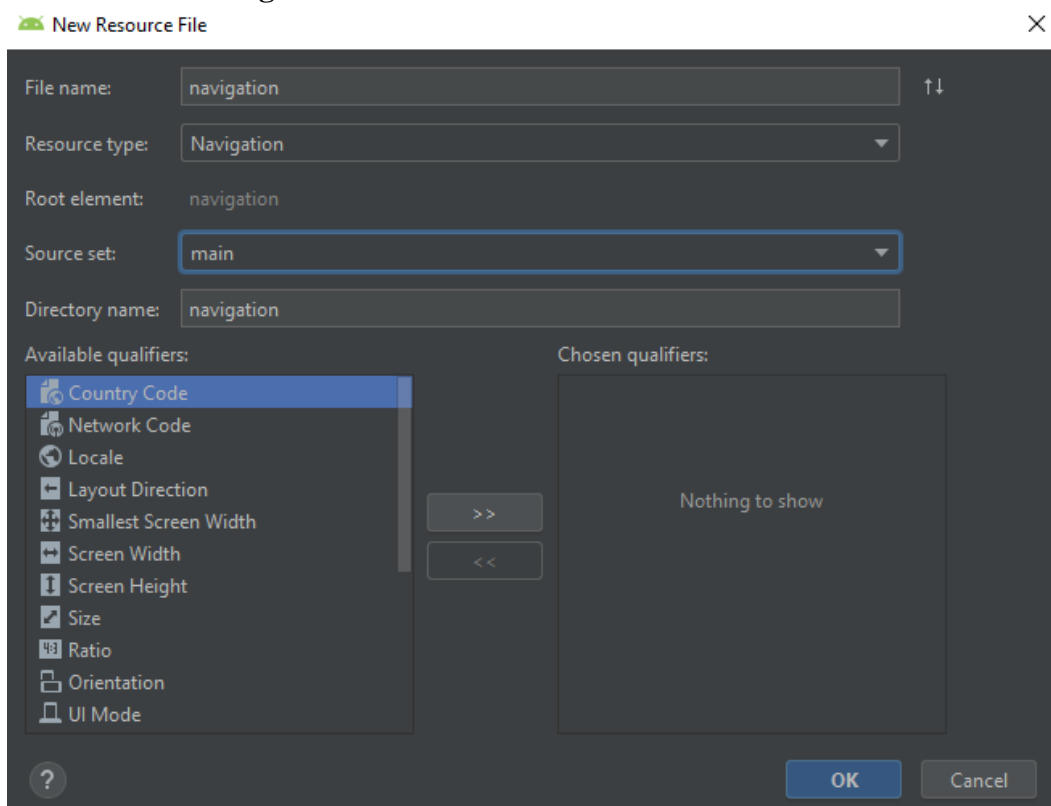
2. Dalam dialog **New Resource File**, pilih **Navigation** sebagai **Resource type**.



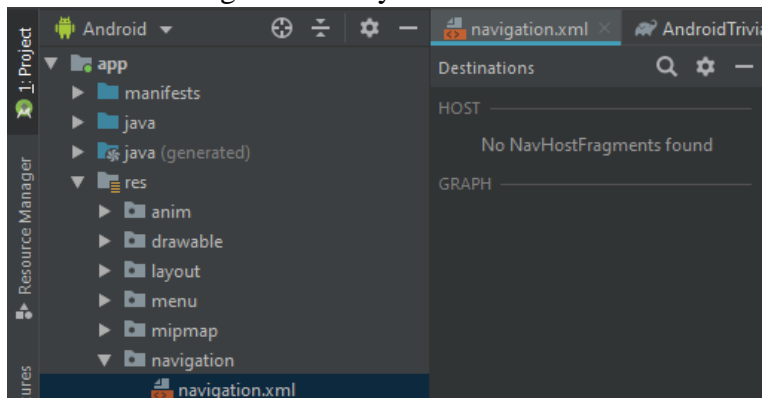
3. Di field **File name**, beri nama file **navigation**.



4. Pastikan kotak Chosen qualifiers kosong, dan klik OK. File baru, **navigation.xml**, muncul di folder **res > navigation**.



5. Buka file **res > navigation > navigation.xml** dan klik tab **Design** untuk membuka **Navigation Editor**. Perhatikan pesan **No NavHostFragments found** di editor layout. Anda memperbaiki masalah ini di tugas berikutnya.



4. Tugas: Membuat NavHostFragment

Navigation host fragment bertindak sebagai host untuk fragmen dalam grafik navigasi. Fragmen host navigasi biasanya dinamai **NavHostFragment**. Saat pengguna berpindah di antara tujuan yang ditentukan dalam grafik navigasi, Fragmen host navigasi menukar fragmen masuk dan keluar seperlunya. Fragmen juga membuat dan mengelola tumpukan belakang Fragmen yang sesuai.

Di tugas ini Anda mengubah kode untuk mengganti **TitleFragment** dengan **NavHostFragment**.

1. Buka **res > layout > activity_main.xml** dan buka tab **Code**.
2. Di file `activity_main.xml`, ubah nama Fragmen judul yang ada menjadi **`androidx.navigation.fragment.NavHostFragment`**.
3. Fragmen host navigasi perlu mengetahui resource grafik navigasi mana yang akan digunakan. Tambahkan atribut **`app:navGraph`** dan setel ke sumber daya grafik navigasi, yaitu **`@navigation/navigation`**.
4. Tambahkan atribut **`app:defaultNavHost`** dan setel ke **`"true"`**. Sekarang host navigasi ini adalah host default dan akan mencegat tombol Kembali sistem.

Di dalam file layout `activity_main.xml`, fragmen Anda sekarang terlihat seperti berikut:

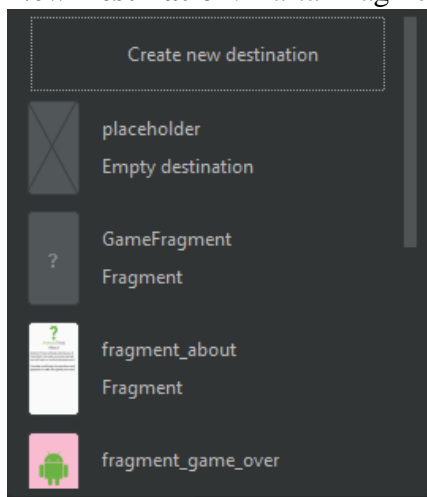
```
<fragment
    android:id="@+id/myNavHostFragment"
    android:name="androidx.navigation.fragment.NavHostFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:navGraph="@navigation/navigation"
    app:defaultNavHost="true" />
```

5. Tugas: Menambahkan fragmen ke grafik navigasi

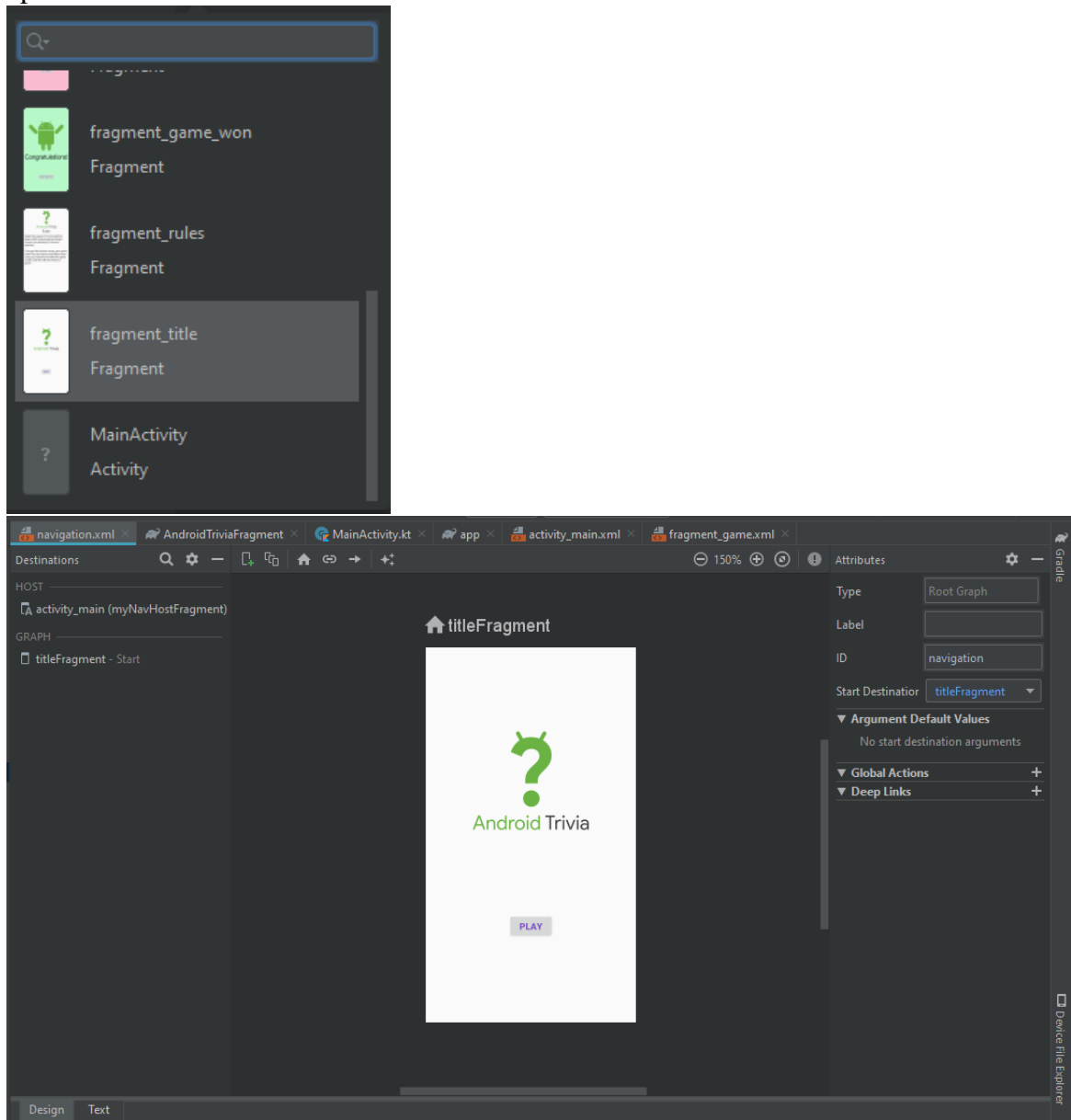
Dalam tugas ini, Anda menambahkan Fragmen judul dan Fragmen game ke grafik navigasi aplikasi. Anda menghubungkan fragmen satu sama lain. Kemudian Anda menambahkan handler klik ke tombol Play sehingga pengguna dapat menavigasi dari layar judul ke layar game.

Langkah 1: Tambahkan dua fragmen ke grafik navigasi dan hubungkan dengan tindakan

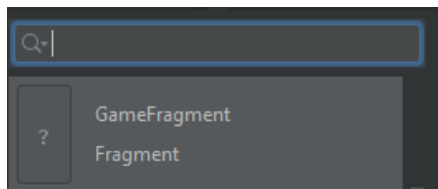
1. Buka **`navigation.xml`** dari folder sumber daya `navigation`. Di **Editor Navigasi**, klik tombol **New Destination**. Daftar fragmen dan aktivitas muncul.



2. Pilih **fragment_title**. Anda menambahkan **fragment_title** terlebih dahulu karena fragment **TitleFragment** adalah tempat pengguna aplikasi memulai saat user pertama kali membuka aplikasi.



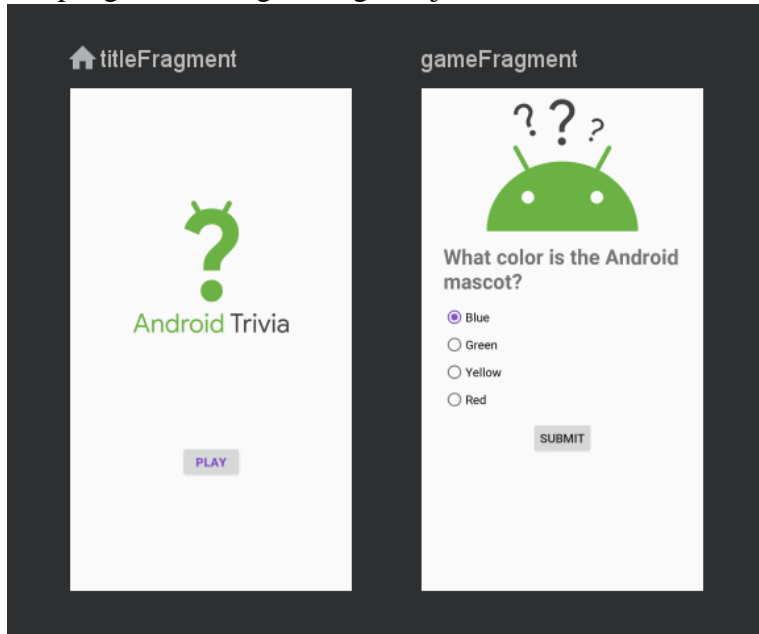
3. Gunakan tombol **New Destination** untuk menambahkan **GameFragment**



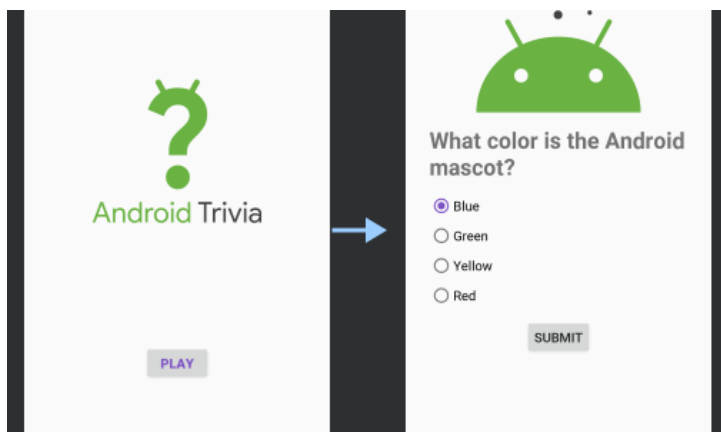
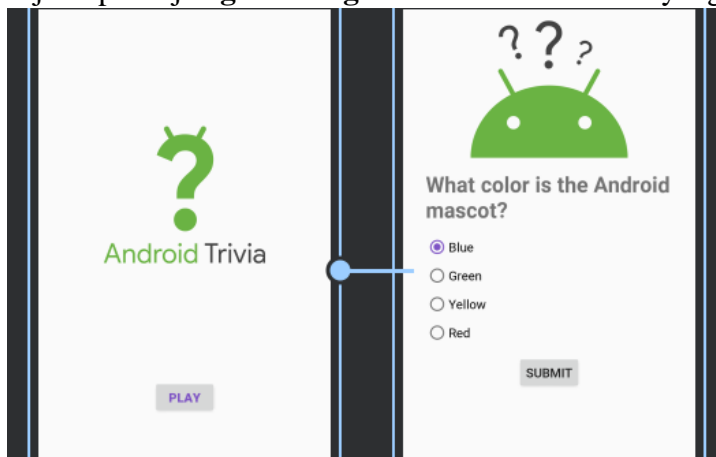
Jika pratinjau (preview) menampilkan pesan "Preview Unavailable", klik tab Code untuk membuka XML navigasi. Pastikan elemen fragment untuk gameFragment menyertakan **tools:layout="@layout/fragment_game"**, seperti yang ditunjukkan di bawah ini.

```
<fragment
    android:id="@+id/gameFragment"
    android:name="com.example.android.navigation.GameFragment"
    android:label="GameFragment"
    tools:layout="@layout/fragment_game"/>
```

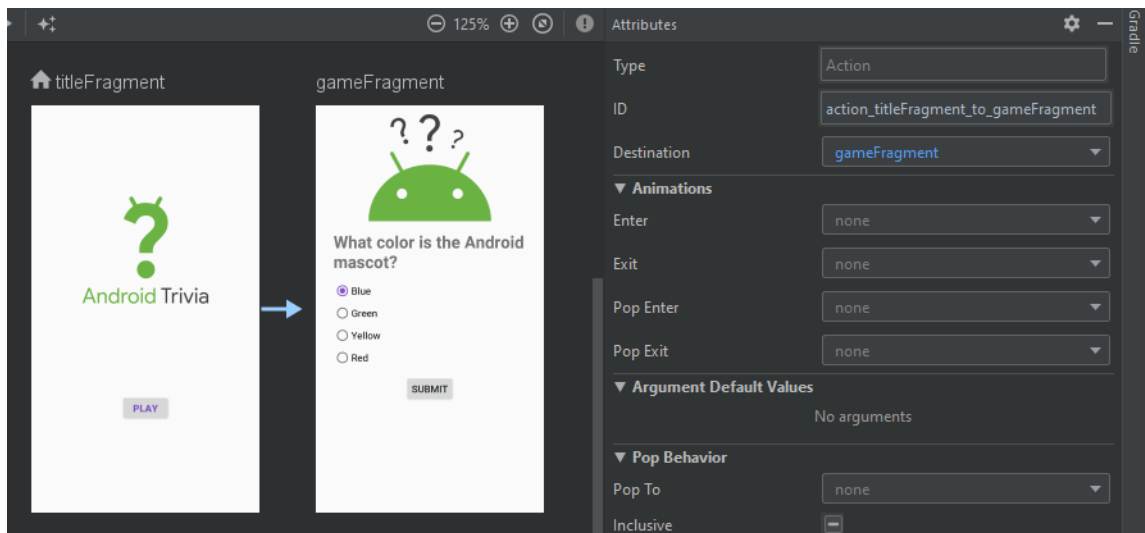
4. Di editor layout (menggunakan tampilan **Design**), seret Fragmen game ke kanan agar tidak tumpang tindih dengan Fragmen judul.



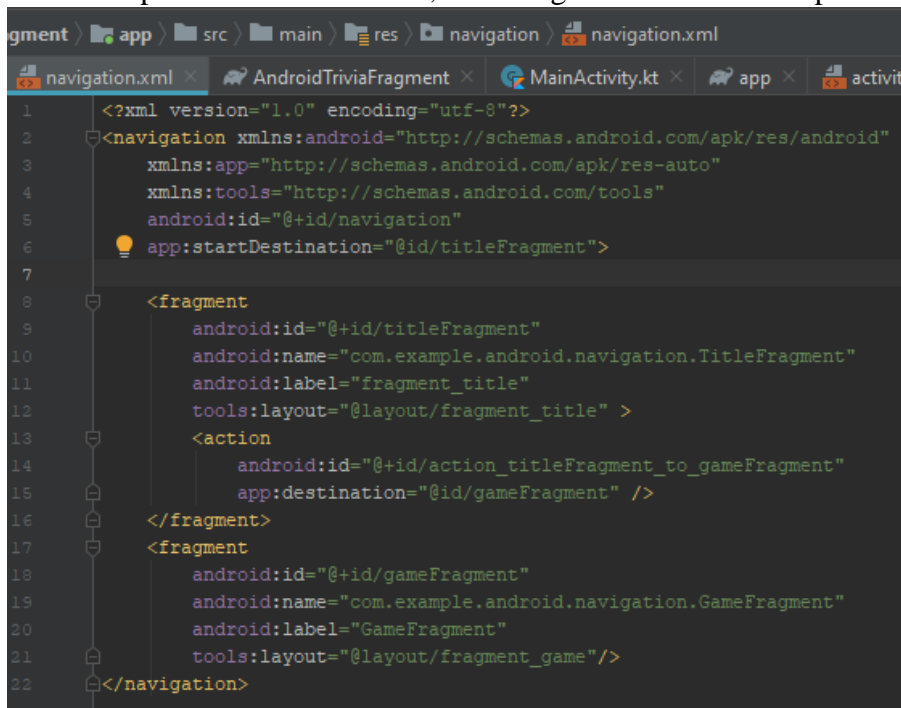
5. Di pratinjau, tahan penunjuk di atas Fragmen judul. Titik koneksi melingkar muncul di sisi kanan tampilan Fragmen. Klik titik koneksi dan seret ke **gameFragment** atau seret ke mana saja di pratinjau **gameFragment**. Tindakan dibuat yang menghubungkan dua fragmen.



6. Untuk melihat atribut Action, klik panah yang menghubungkan dua fragmen. Di panel Attributes, periksa apakah ID tindakan disetel ke **action_titleFragment_to_gameFragment**.



Jika ditampilkan dalam tab Code, file navigation.xml adalah seperti berikut:



Langkah 2: Tambahkan handler klik ke tombol Putar

Fragmen judul terhubung ke game Fragmen dengan tindakan. Sekarang Anda ingin tombol Play di layar judul untuk mengarahkan pengguna ke layar game.

1. Di Android Studio, buka file **TitleFragment.kt**. Di dalam metode **onCreateView()**, tambahkan kode berikut sebelum pernyataan return:

```
binding.playButton.setOnClickListener{
```
2. Di dalam **setOnClickListener**, tambahkan kode untuk mengakses tombol **Play** melalui class binding dan buka fragmen game:

```

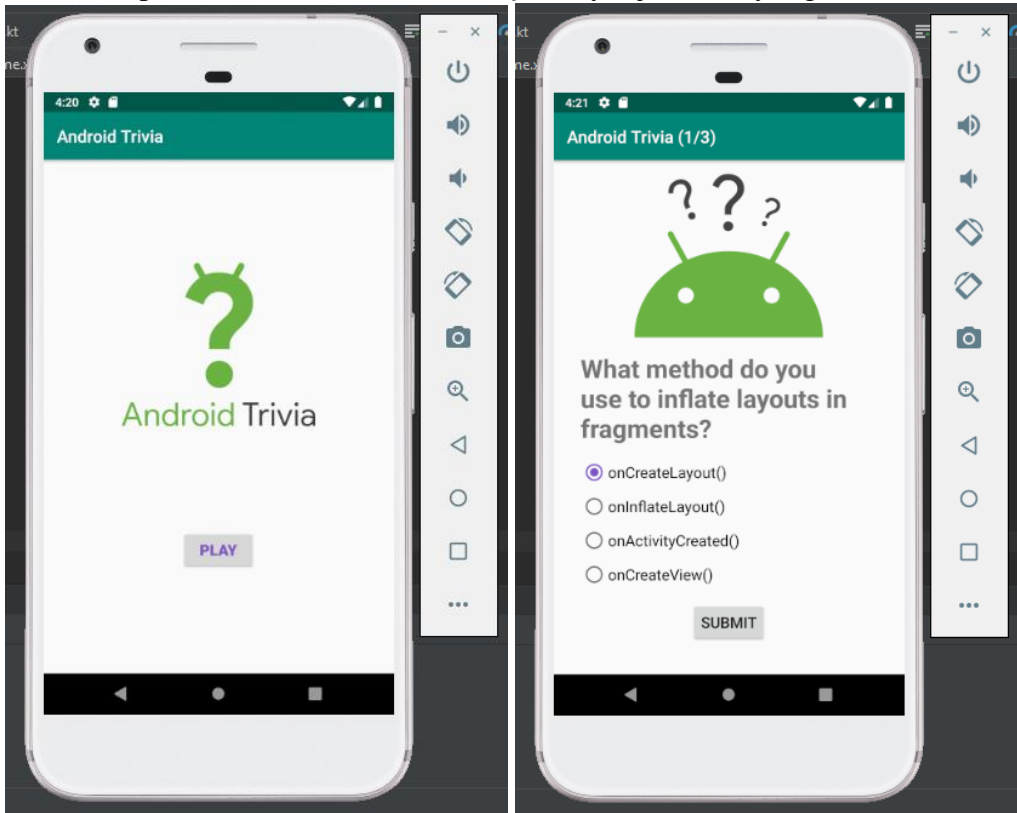
binding.playButton.setOnClickListener{view : View ->
    view.findNavController().navigate(R.id.action_titleFragment_to_gameFragment)
}

```


3. Pastikan aplikasi tersebut memiliki semua impor yang dibutuhkannya. Misalnya, Anda mungkin perlu menambahkan baris berikut ke file **TitleFragment.kt**:

```
import androidx.navigation.findNavController
```

4. Jalankan aplikasi dan ketuk tombol **Play** di layar judul. Layar game akan terbuka.



6. Tugas: Menambahkan navigasi bersyarat

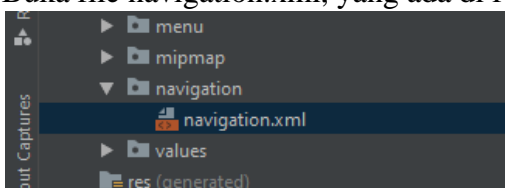
Pada langkah ini, Anda menambahkan navigasi bersyarat, yaitu navigasi yang hanya tersedia untuk pengguna dalam konteks tertentu. Kasus penggunaan umum untuk navigasi bersyarat adalah saat aplikasi memiliki aliran yang berbeda, bergantung pada apakah pengguna login atau tidak. Kasus aplikasi Anda berbeda: Aplikasi Anda akan mengarah ke Fragmaen yang berbeda, berdasarkan apakah pengguna menjawab semua pertanyaan dengan benar.

Kode awal berisi dua fragmen untuk Anda gunakan dalam navigasi bersyarat:

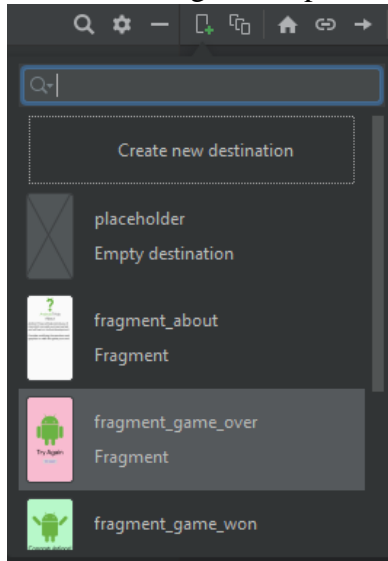
- **GameWonFragment** membawa pengguna ke layar yang menampilkan pesan " Congratulations!".
- **GameOverFragment** membawa pengguna ke layar yang menampilkan pesan " Try Again".

Langkah 1: Tambahkan GameWonFragment dan GameOverFragment ke grafik navigasi

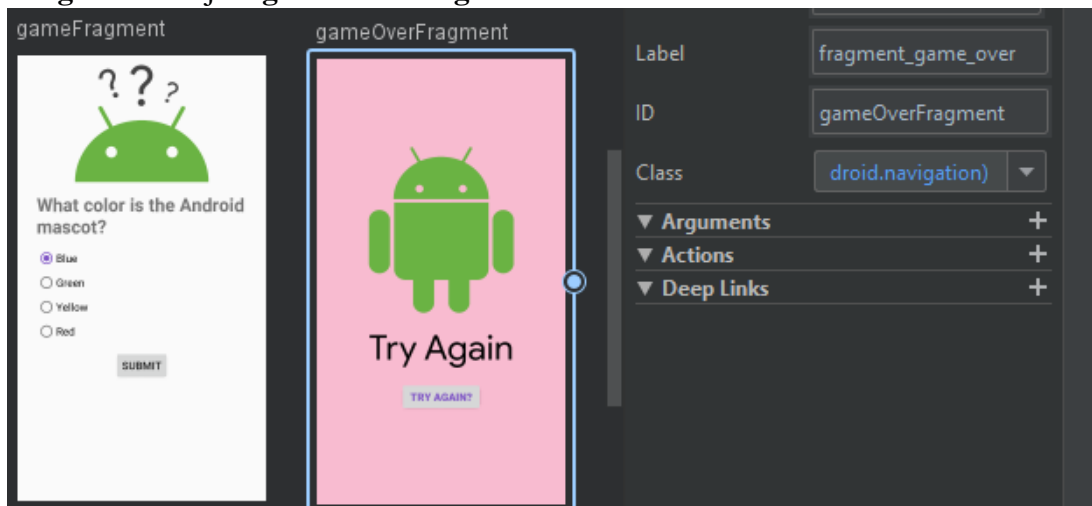
1. Buka file navigation.xml, yang ada di folder navigation.



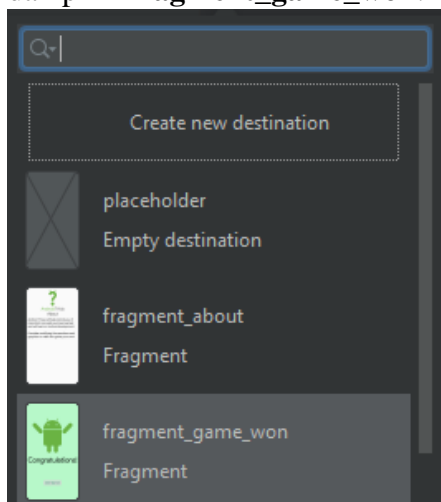
2. Untuk menambahkan Fragmen game-over ke grafik navigasi, klik tombol **New Destination** di Editor Navigasi dan pilih **fragment_game_over**.



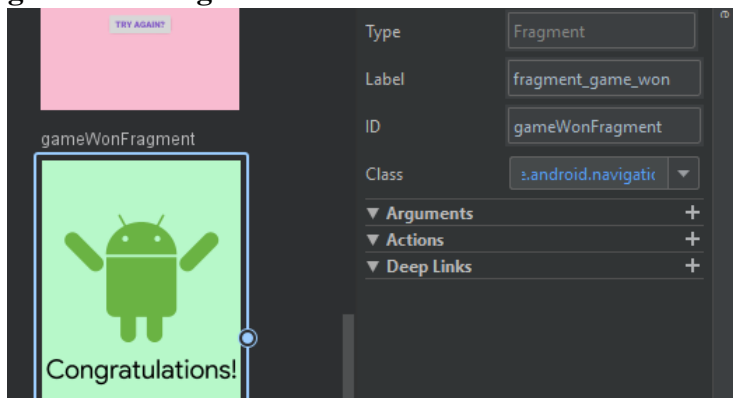
3. Di area pratinjau editor layout, seret Fragmen game-over ke kanan Fragmen game sehingga keduanya tidak tumpang tindih. Pastikan untuk mengubah atribut ID dari **game-over Fragment** menjadi **gameOverFragment**.



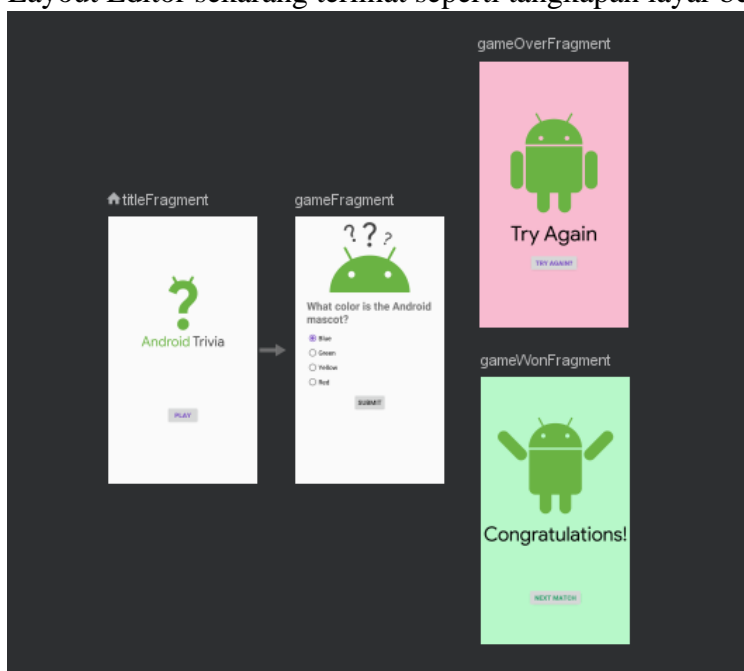
4. Untuk menambahkan Fragmen game-won ke grafik navigasi, klik tombol **New Destination** dan pilih **fragment_game_won**.



5. Seret Fragmen game-won game di bawah Fragmen game-over sehingga keduanya tidak tumpang tindih. Pastikan untuk menamai atribut ID Fragment game-won sebagai **gameWonFragment**.



6. Layout Editor sekarang terlihat seperti tangkapan layar berikut:



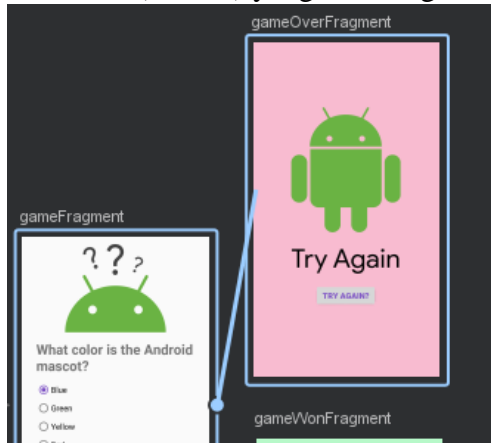
Langkah 2: Hubungkan Fragmen game ke Fragmen hasil game

Pada langkah ini, Anda menghubungkan Fragmen game ke Fragmen game-won dan Fragmen game-over.

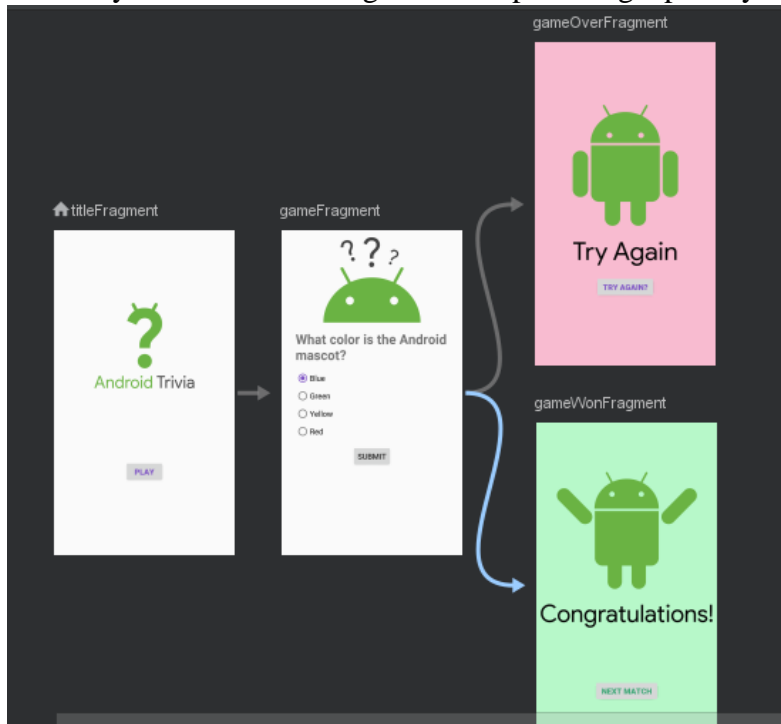
1. Di area pratinjau Editor Tata Letak, tahan penunjuk di atas Fragmen game hingga titik koneksi melingkar muncul.



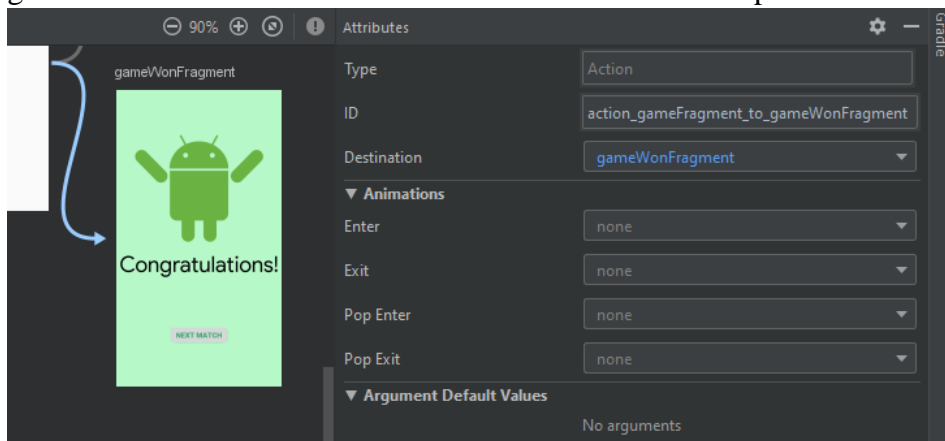
2. Klik titik koneksi dan seret ke Fragmen game-over. Panah koneksi biru muncul, mewakili Tindakan (Action) yang sekarang menghubungkan Fragmen game ke Fragmen game-over.



3. Dengan cara yang sama, buat aksi yang menghubungkan Fragmen game ke Fragmen game-won. Layout Editor sekarang terlihat seperti tangkapan layar berikut:



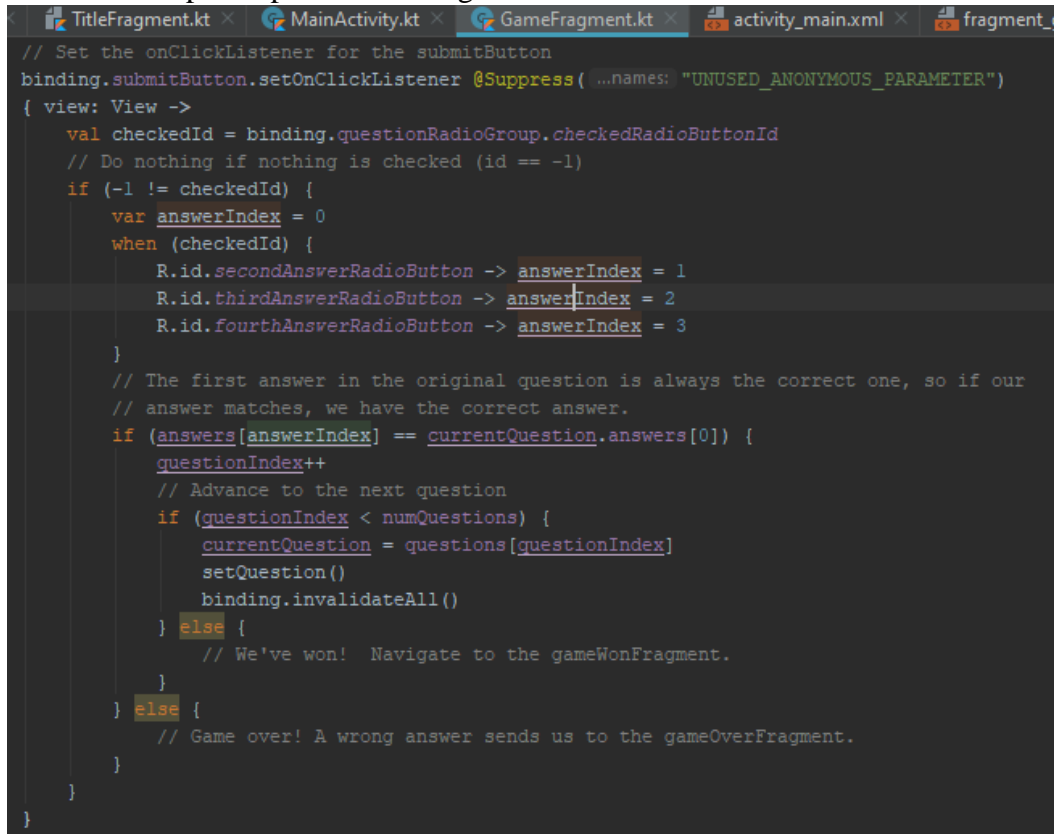
4. Di pratinjau, tahan penunjuk di atas garis yang menghubungkan Fragmen game ke Fragmen game-won. Perhatikan bahwa ID untuk Action telah ditetapkan secara otomatis.



Langkah 3: Tambahkan kode untuk menavigasi dari satu Fragmen ke Fragmen berikutnya

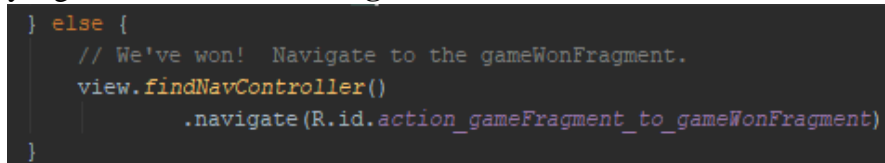
GameFragment adalah kelas Fragmen yang berisi pertanyaan dan jawaban untuk game tersebut. Kelas juga menyertakan logika yang menentukan apakah pengguna menang atau kalah dalam permainan. Anda perlu menambahkan navigasi bersyarat di kelas GameFragment, bergantung pada apakah pemain menang atau kalah.

1. Buka file **GameFragment.kt**. Metode **onCreateView ()** mendefinisikan kondisi if/else yang menentukan apakah pemain menang atau kalah:



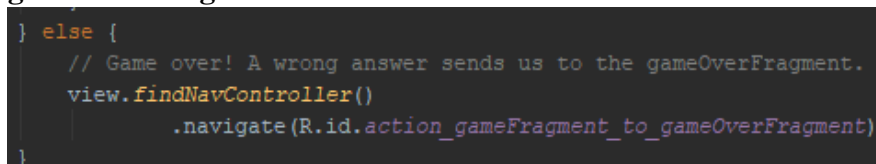
```
// Set the onClickListener for the submitButton
binding.submitButton.setOnClickListener @Suppress("UNUSED_ANONYMOUS_PARAMETER")
{ view: View ->
    val checkedId = binding.questionRadioGroup.checkedRadioButtonId
    // Do nothing if nothing is checked (id == -1)
    if (-1 != checkedId) {
        var answerIndex = 0
        when (checkedId) {
            R.id.secondAnswerRadioButton -> answerIndex = 1
            R.id.thirdAnswerRadioButton -> answerIndex = 2
            R.id.fourthAnswerRadioButton -> answerIndex = 3
        }
        // The first answer in the original question is always the correct one, so if our
        // answer matches, we have the correct answer.
        if (answers[answerIndex] == currentQuestion.answers[0]) {
            questionIndex++
            // Advance to the next question
            if (questionIndex < numQuestions) {
                currentQuestion = questions[questionIndex]
                setQuestion()
                binding.invalidateAll()
            } else {
                // We've won! Navigate to the gameWonFragment.
            }
        } else {
            // Game over! A wrong answer sends us to the gameOverFragment.
        }
    }
}
```

2. Di dalam kondisi else untuk memenangkan permainan, tambahkan kode berikut, yang menavigasi ke **gameWonFragment**. Pastikan bahwa nama Action (**action_gameFragment_to_gameWonFragment** dalam contoh ini) sama persis dengan yang disetel dalam file **navigation.xml**.



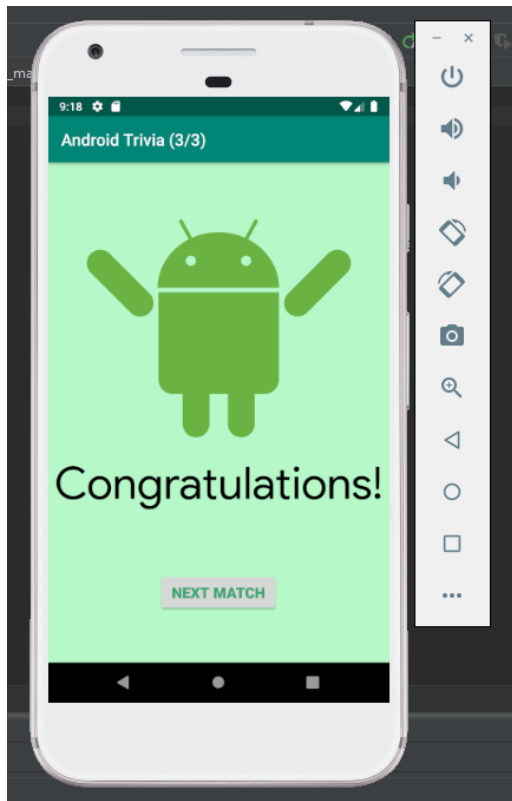
```
} else {
    // We've won! Navigate to the gameWonFragment.
    view.findNavController()
        .navigate(R.id.action_gameFragment_to_gameWonFragment)
}
```

3. Di dalam kondisi else untuk kekalahan game, tambahkan kode berikut, yang mengarah ke **gameOverFragment**:

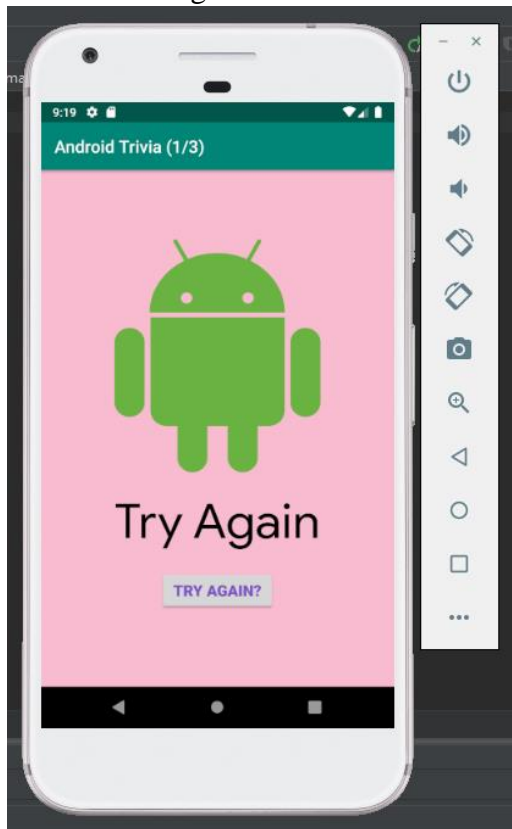


```
} else {
    // Game over! A wrong answer sends us to the gameOverFragment.
    view.findNavController()
        .navigate(R.id.action_gameFragment_to_gameOverFragment)
}
```

4. Jalankan aplikasi dan mainkan game dengan menjawab pertanyaan. Jika Anda menjawab ketiga pertanyaan dengan benar, aplikasi menavigasi ke **GameWonFragment**.



Jika Anda salah menjawab pertanyaan, aplikasi akan segera menavigasi ke GameOverFragment.



Tombol Kembali sistem Android ditampilkan sebagai 1 pada gambar di atas. Jika pengguna menekan tombol Kembali di fragmen game-won atau Fragmen game-over, aplikasi akan membuka layar pertanyaan. Idealnya, tombol Kembali harus mengarah kembali ke layar judul aplikasi. Anda mengubah tujuan untuk tombol Kembali di tugas berikutnya.

7. Tugas: Mengubah tujuan tombol Kembali

Sistem Android melacak ke mana pengguna bernavigasi di perangkat yang diberdayakan Android. Setiap kali pengguna pergi ke tujuan baru di perangkat, Android menambahkan tujuan itu ke back-stack.

Saat pengguna menekan tombol Kembali, aplikasi pergi ke tujuan yang ada di bagian atas tumpukan belakang. Secara default, bagian atas back-stack adalah layar yang terakhir dilihat pengguna. Tombol Kembali biasanya merupakan tombol paling kiri di bagian bawah layar, seperti yang ditunjukkan di bawah ini. (Tampilan persis tombol Kembali berbeda di perangkat yang berbeda.)



Hingga saat ini, Anda telah membiarkan pengontrol navigasi menangani back-stack untuk Anda. Saat pengguna menavigasi ke tujuan di aplikasi Anda, Android menambahkan tujuan ini ke back-stack.

Di aplikasi AndroidTrivia, saat pengguna menekan tombol Kembali dari layar GameOverFragment atau GameWonFragment, mereka akan kembali ke GameFragment. Tetapi Anda tidak ingin mengirim pengguna ke GameFragment, karena permainan telah berakhir. Pengguna dapat memulai ulang game, tetapi pengalaman yang lebih baik adalah menemukan diri mereka kembali di layar judul.

Tindakan navigasi dapat mengubah back-stack. Dalam tugas ini, Anda mengubah tindakan yang menavigasi dari fragmen game sehingga tindakan tersebut menghapus GameFragment dari back-stack. Saat pengguna menang atau kalah, jika mereka menekan tombol Kembali, aplikasi melewati GameFragment dan kembali ke TitleFragment.

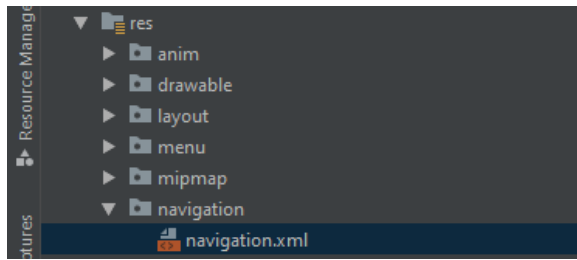
Langkah 1: Atur perilaku pop untuk tindakan navigasi

Pada langkah ini, Anda mengelola back-stack sehingga saat pengguna berada di layar GameWon atau GameOver, menekan tombol Kembali akan mengembalikan mereka ke layar judul. Anda mengelola back-stack dengan menyetel perilaku "pop" untuk tindakan yang menghubungkan fragmen:

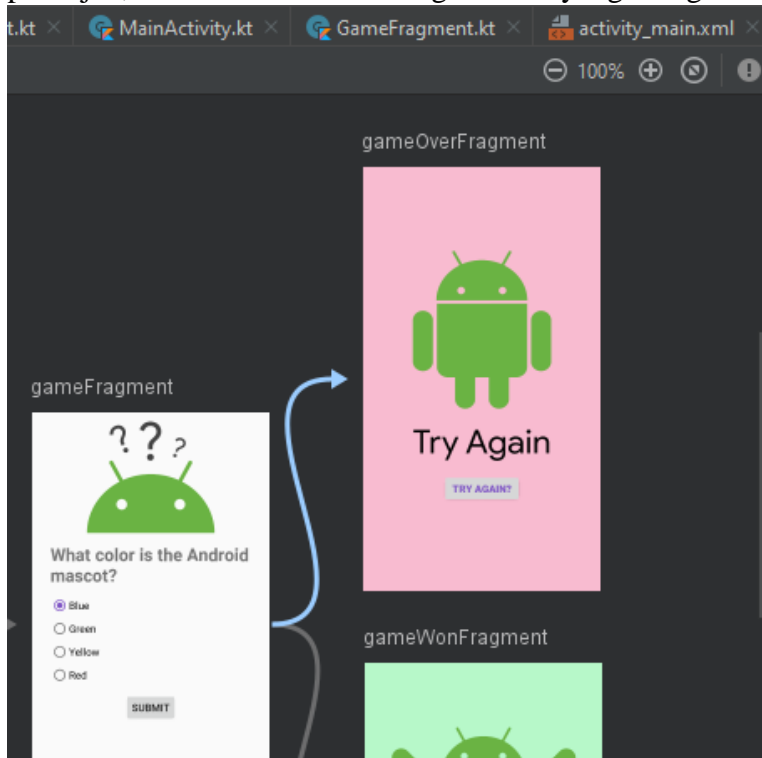
- Atribut **popUpTo** dari suatu tindakan "memunculkan" back-stack ke tujuan tertentu sebelum menavigasi. (Tujuan dihapus dari tumpukan belakang.)
- Jika atribut **popUpToInclusive** adalah false atau tidak disetel, **popUpTo** akan menghapus tujuan hingga tujuan yang ditentukan, tetapi meninggalkan tujuan yang ditentukan di back-stack.
- Jika **popUpToInclusive** disetel ke true, atribut **popUpTo** menghapus semua tujuan hingga dan termasuk tujuan yang diberikan dari back-stack.
- Jika **popUpToInclusive** benar dan **popUpTo** disetel ke lokasi awal aplikasi, tindakan akan menghapus semua tujuan aplikasi dari back-stack. Tombol Kembali membawa pengguna keluar dari aplikasi.

Pada langkah ini, Anda menyetel atribut popUpTo untuk dua tindakan yang Anda buat di tugas sebelumnya. Anda melakukan ini menggunakan field Pop To di panel Attributes dari Layout Editor.

1. Buka **navigation.xml** di **res > folder navigation**. Jika grafik navigasi tidak muncul di editor tata letak, klik tab **Desain**.
(screenshot di halaman selanjutnya)



2. Pilih tindakan untuk menavigasi dari **gameFragment** ke **gameOverFragment**. (Di area pratinjau, tindakan diwakili oleh garis biru yang menghubungkan dua fragmen.)



3. Di panel Attributes, setel **popUpTo** ke **gameFragment**. Pilih kotak centang **popUpToInclusive**.

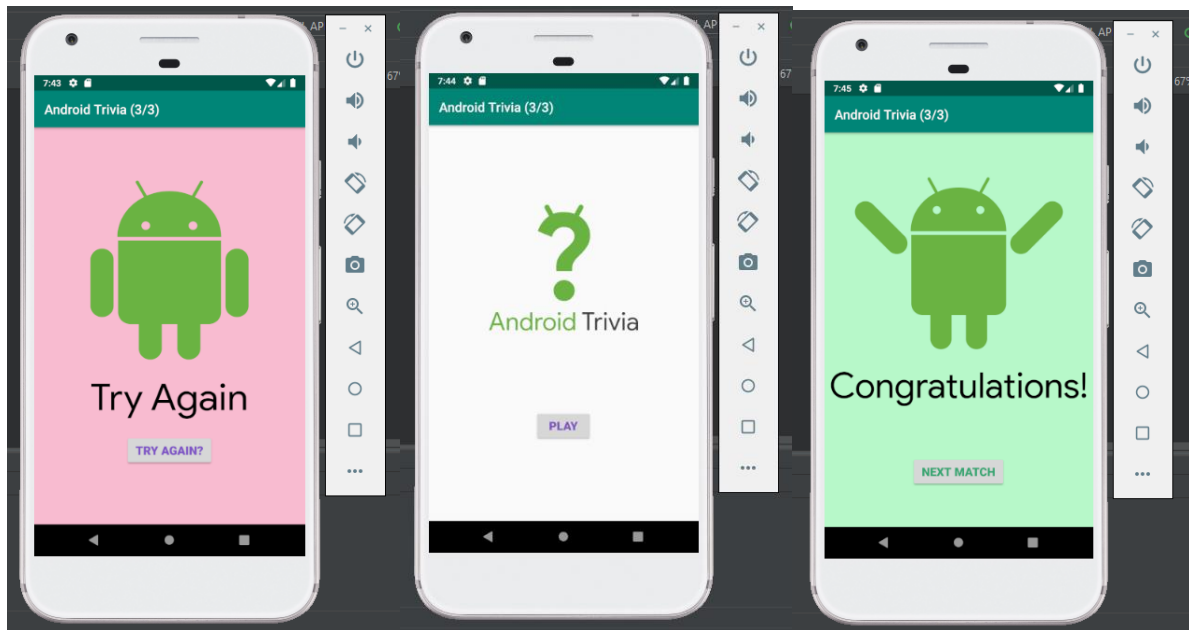


Ini menyetel atribut **popUpTo** dan **popUpToInclusive** di XML. Atribut memberi tahu komponen navigasi untuk menghapus fragmen dari back-stack hingga dan termasuk **GameFragment**. (Ini memiliki efek yang sama seperti menyetel bidang **popUpTo** ke **titleFragment** dan mengosongkan kotak centang **popUpToInclusive**.)

4. Pilih tindakan untuk menavigasi dari **gameFragment** ke **gameWonFragment**. Sekali lagi, atur **popUpTo** ke **gameFragment** di panel Attributes dan pilih kotak centang **popUpToInclusive**.



5. Jalankan aplikasi dan mainkan gamenya, lalu tekan tombol Kembali. Apakah Anda menang atau kalah, tombol Kembali membawa Anda kembali ke TitleFragment.



Langkah 2: Tambahkan lebih banyak tindakan navigasi dan tambahkan penanganan onClick

Aplikasi Anda saat ini memiliki alur pengguna berikut:

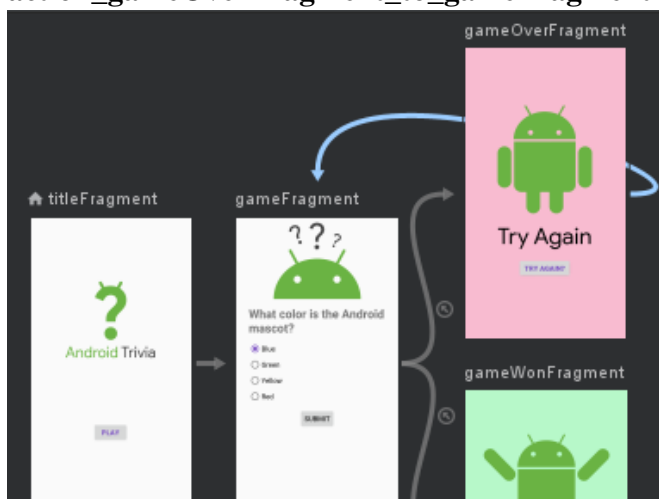
- Pengguna memainkan permainan dan menang atau kalah, dan aplikasi menavigasi ke layar **GameWon** atau **GameOver**.
- Jika pengguna menekan tombol Kembali sistem pada saat ini, aplikasi akan menavigasi ke **TitleFragment**. (Anda menerapkan perilaku ini pada Langkah 1 dari tugas ini, di atas.)

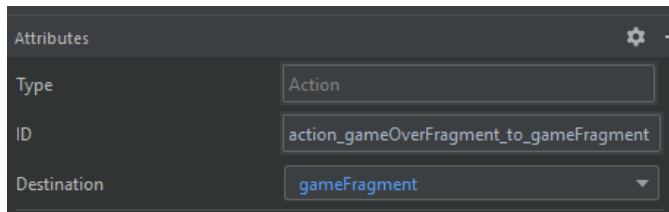
Pada langkah ini, Anda menerapkan dua langkah lagi dari alur pengguna:

- Jika pengguna mengetuk tombol **Next Match** atau **Try Again**, aplikasi menavigasi ke layar **GameFragment**.
- Jika pengguna menekan tombol Kembali sistem pada titik ini, aplikasi menavigasi ke layar **TitleFragment** (bukan kembali ke layar **GameWon** atau **GameOver**).

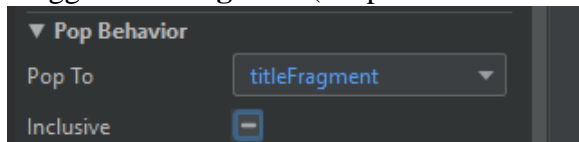
Untuk membuat alur pengguna ini, gunakan atribut `popUpTo` untuk mengelola back-stack:

1. Di dalam file **navigation.xml**, tambahkan tindakan navigasi yang menghubungkan **gameOverFragment** ke **gameFragment**. Pastikan nama Fragmen di ID tindakan cocok dengan nama Fragmen yang ada di XML. Misalnya, ID tindakan mungkin adalah **action_gameOverFragment_to_gameFragment** **. **

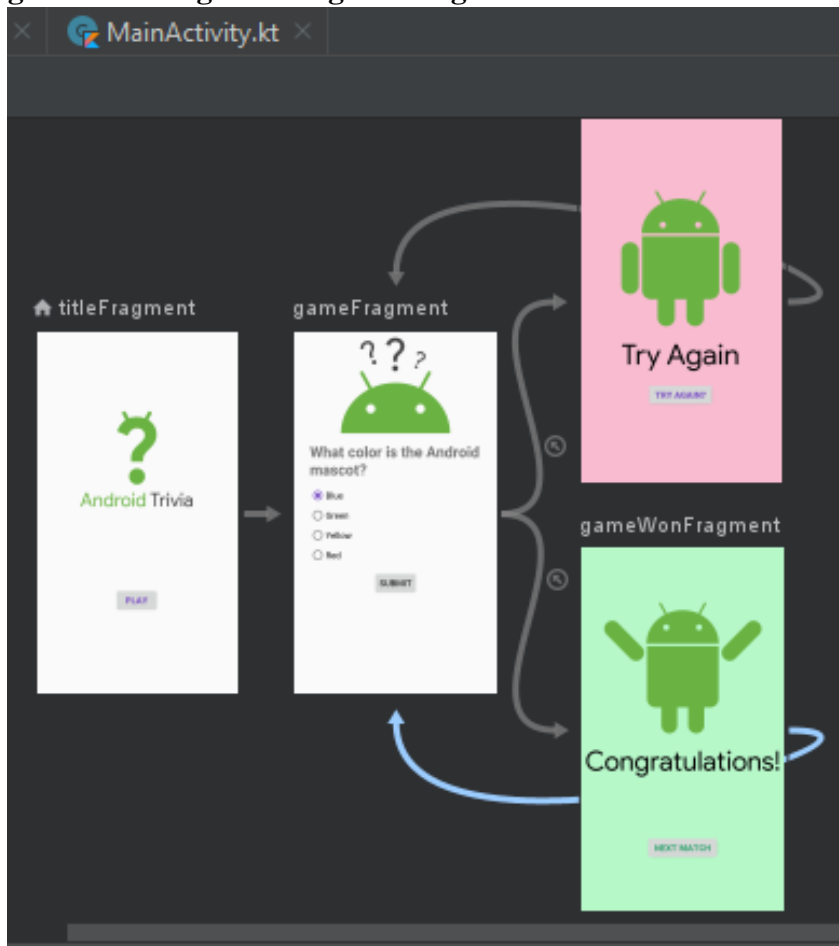




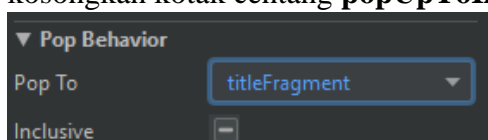
2. Di panel Attributes, setel atribut **popUpTo** tindakan ke **titleFragment**.
3. Kosongkan kotak centang **popUpToInclusive**, karena Anda tidak ingin **titleFragment** disertakan dalam tujuan yang dihapus dari back-stack. Sebaliknya, Anda ingin semuanya hingga **TitleFragment** (tetapi tidak termasuk) untuk dihapus dari back-stack.



4. Di dalam file **navigation.xml**, tambahkan tindakan navigasi yang menghubungkan **gameWonFragment** ke **gameFragment**.



5. Untuk tindakan yang baru saja Anda buat, setel atribut **popUpTo** ke **titleFragment** dan kosongkan kotak centang **popUpToInclusive**.



Sekarang tambahkan fungsionalitas ke tombol Try Again dan Next Match. Saat pengguna mengetuk salah satu tombol, Anda ingin aplikasi menavigasi ke layar GameFragment sehingga pengguna dapat mencoba game lagi.

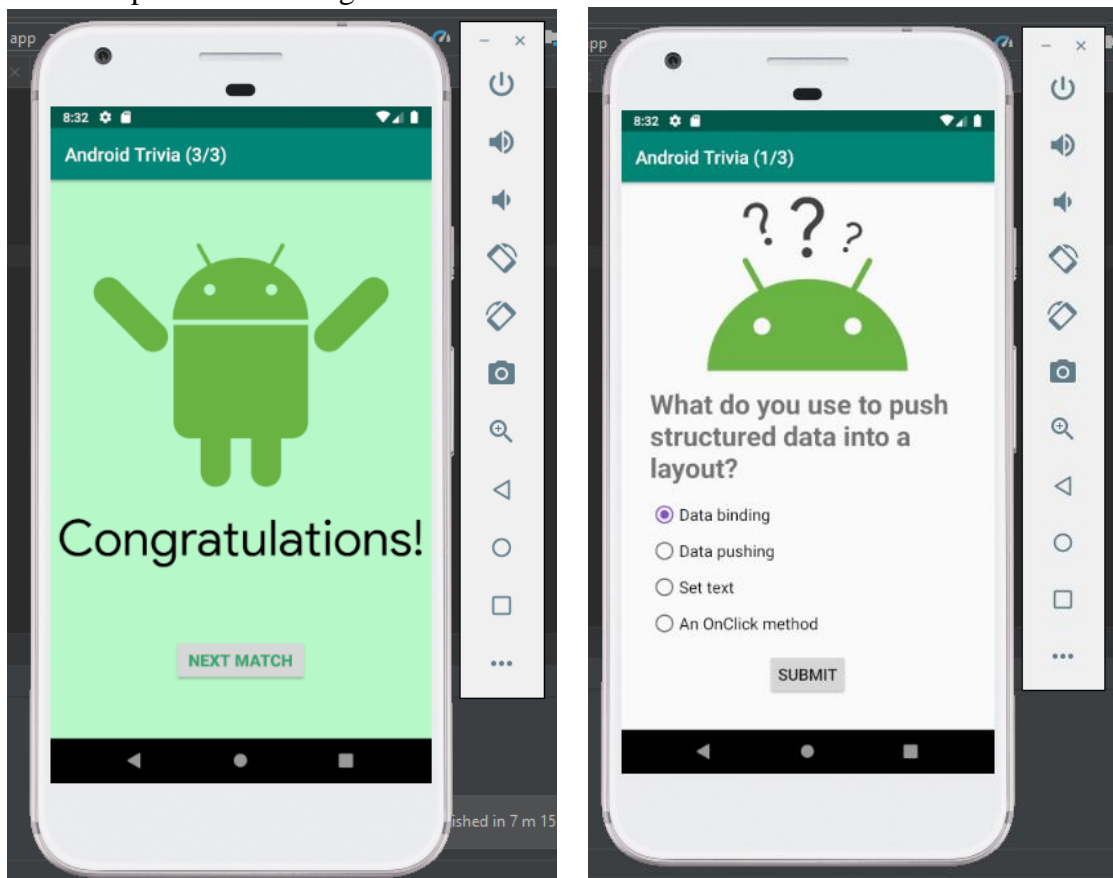
1. Buka file Kotlin **GameOverFragment.kt**. Di akhir metode **onCreateView ()**, sebelum pernyataan return, tambahkan kode berikut. Kode menambahkan pendengar klik ke tombol **Try Again**. Saat pengguna mengetuk tombol tersebut, aplikasi akan membuka Fragmen game.

```
// Add OnClick Handler for Try Again button
binding.tryAgainButton.setOnClickListener{view: View->
    view.findNavController()
        .navigate(R.id.action_gameOverFragment_to_gameFragment)}
```

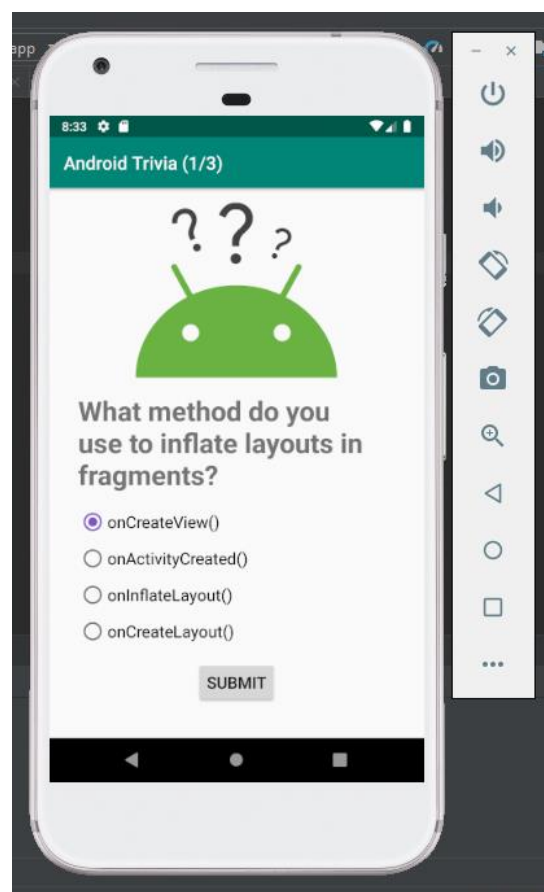
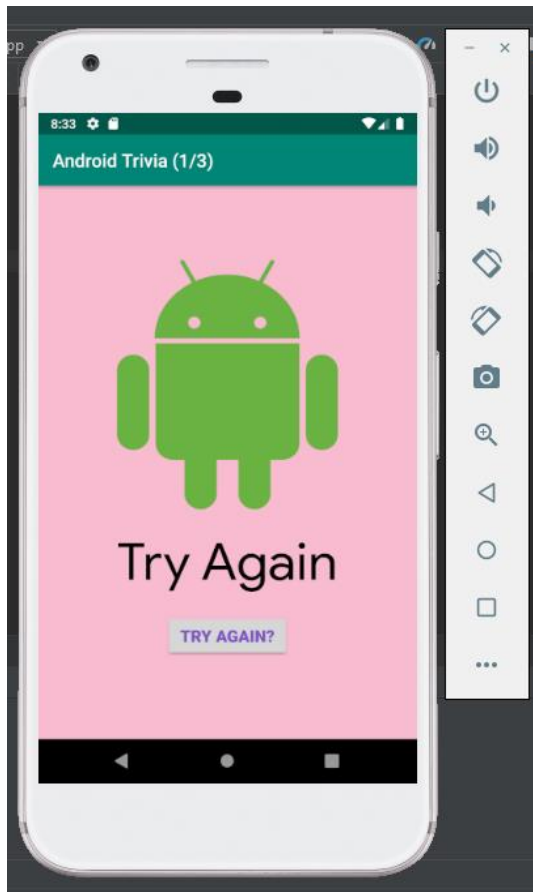
2. Buka file Kotlin **GameWonFragment.kt**. Di akhir metode **onCreateView ()**, sebelum pernyataan return, tambahkan kode berikut:

```
// Add OnClick Handler for Try Again button
binding.nextMatchButton.setOnClickListener{view: View->
    view.findNavController()
        .navigate(R.id.action_gameWonFragment_to_gameFragment)}
```

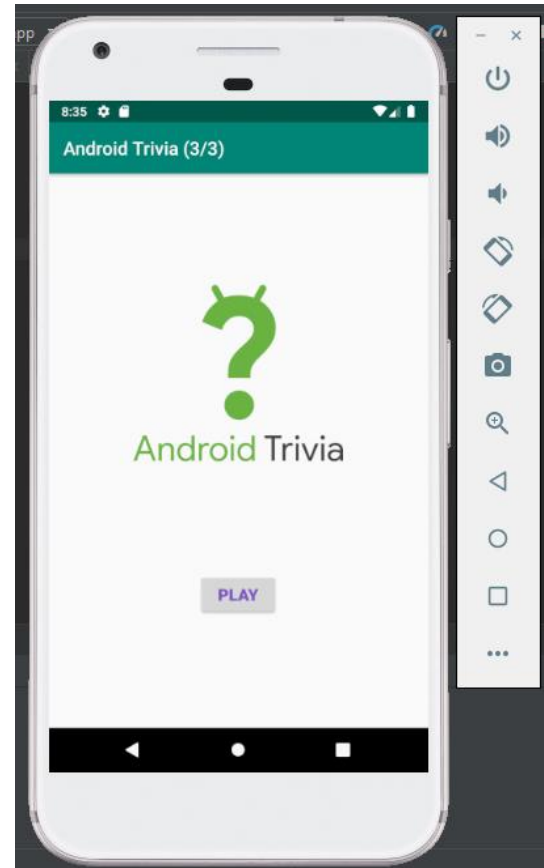
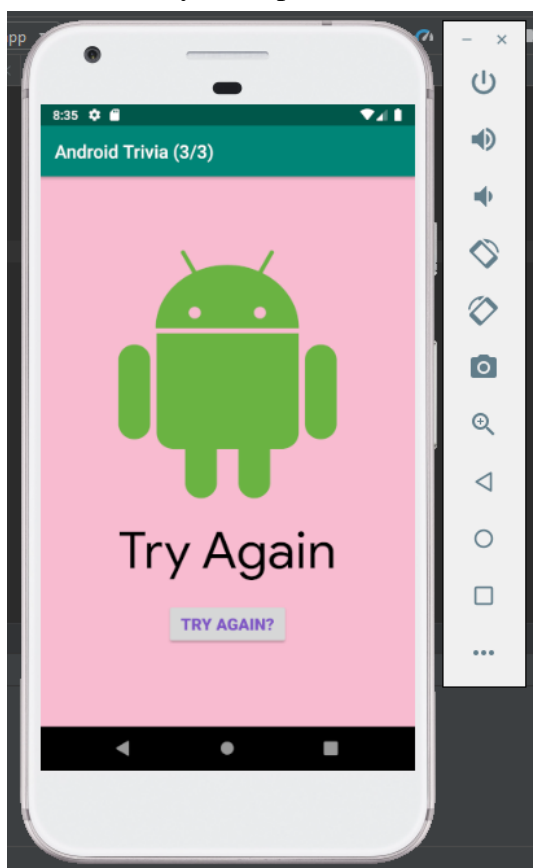
3. Jalankan aplikasi Anda, mainkan game, dan uji tombol Next Match dan Try Again. Kedua tombol tersebut akan membawa Anda kembali ke layar permainan sehingga Anda dapat mencoba permainan itu lagi.

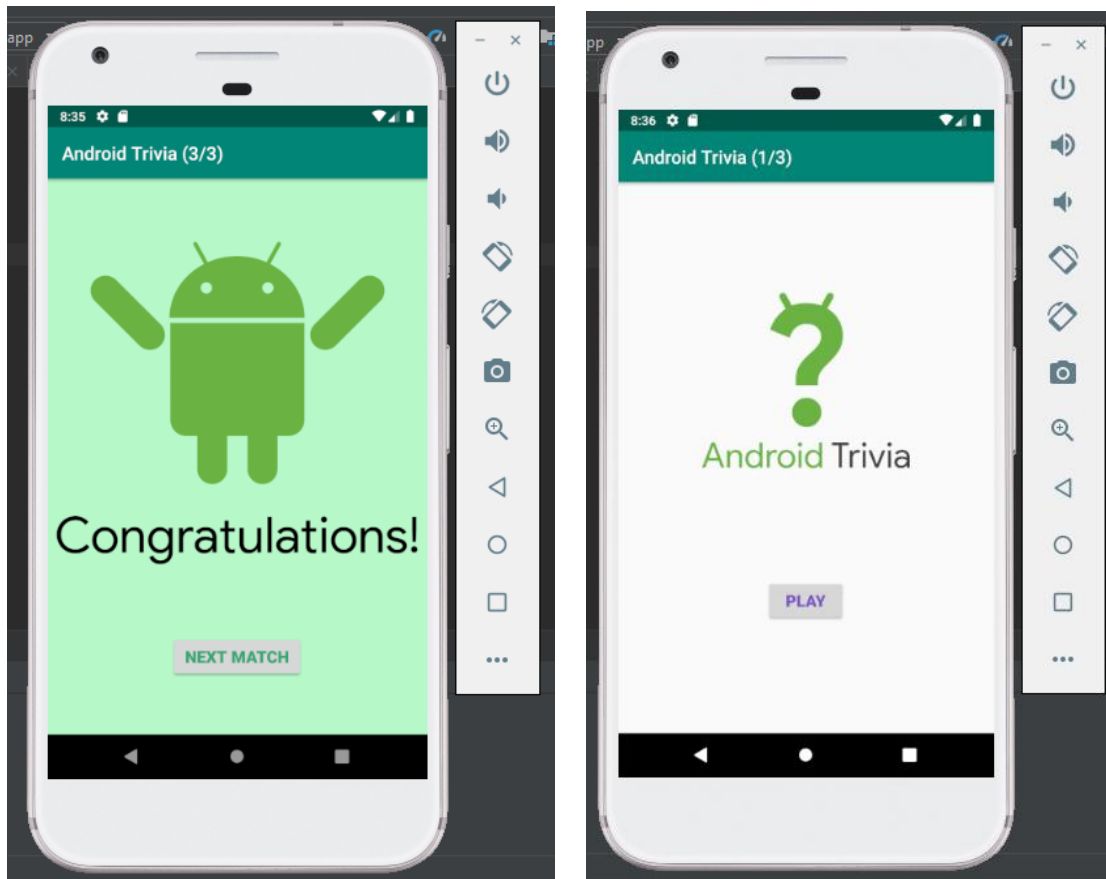


(Screenshot tombol Try Again di halaman selanjutnya)



4. Setelah Anda menang atau kalah dalam permainan, ketuk Next Match atau Try Again. Sekarang tekan tombol Kembali sistem. Aplikasi harus menavigasi ke layar judul, alih-alih kembali ke layar tempat Anda baru datang.





KESIMPULAN

Pada laporan praktikum pertemuan 7 ini, saya berhasil memenuhi tujuan dibuatnya modul 7 ini yaitu mampu mendefinisikan dan menggunakan navigasi dalam aplikasi. Navigasi yang kita pelajari di modul ini ada hubungannya dengan fragmen yang telah kita pelajari pada pertemuan 6 lalu. Pola pembuatannya kurang lebih adalah dengan membuat fragmen terlebih dahulu, lalu setelah fragmen telah dibuat, kita bisa memberikan navigasi sebagai cara bagi user untuk berpindah-pindah tujuan di dalam suatu aplikasi. Di dalam laporan yang saya kerjakan ini, saya dapat memahami cara-cara menggunakan navigasi pada Android Studio, diantaranya adalah cara menggunakan grafik navigasi, cara menentukan jalur navigasi, cara membuat menu option, cara membuat panel, dan sebagainya. Dalam proses memahami hal-hal yang berkaitan dengan navigasi tersebut, dalam laporan ini saya mempraktekkan beberapa hal seperti membuat grafik navigasi untuk fragmen yang telah dibuat, membuat jalur navigasi, menambahkan menu panel, dan lain-lain. Di pertemuan 7 ini saya mendapatkan banyak pembelajaran penting tentang navigasi pada pemrograman mobile/android.

Terima Kasih