3.10. Боковое меню приложения

Сайт: Samsung Innovation Campus
Курс: Мобильная разработка на Kotlin
Книга: 3.10. Боковое меню приложения

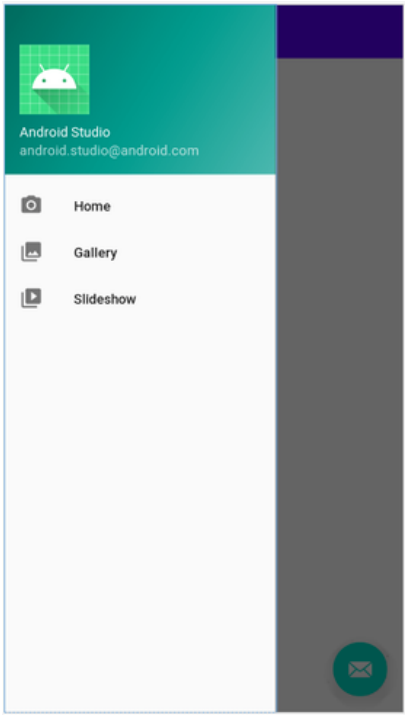
Напечатано:: Murad Rezvan
Дата: понедельник, 3 июня 2024, 17:51

Оглавление

3.10.1. Дизайн панели

На презентации Google I/O 2015, компания Google представила новую версию библиотеки поддержки, которая реализует несколько компонентов, сильно связанных со спецификациями Material Design. Среди этих компонентов есть новые типы ViewGroup такие как AppbarLayout, CollapsingToolbarLayout и CoordinatorLayout, DrawerLayout и новые элементы в т.ч. Drawer или боковое меню приложения. По сути это дополнительное меню приложения на боковой шторке, которое выдвигается либо по вытягиванию шторки слева (иногда справа) - т.е. свайп от левой границы экрана вправо, либо по нажатию специальной кнопки с drawer icon ≡ (иногда его называют гамбургер).

Выглядит drawer следующим образом:



Рассматриваемые далее примеры входят в практическую работу, исходный код которой можно скачать по ссылке

На макете экрана за боковую шторки отвечает виджет NavigationView, который обязательно должен располагаться в специальном лайоуте - DrawerLayout. DrawerLayout действует как контейнер верхнего уровня для содержимого окна, что позволяет извлекать интерактивные панели из одного или обоих вертикальных краев окна. Самый просто способ создать такую панель - создать готовый макет из набора предлагаемых Андроид студией активностей. Для этого нужно нажать *New->Activity->NavigationDrawerAcivity*. При этом будет созданы DrawerLayout и элемент шторки (NavigationView) в нем.

```
<androidx.drawerlayout.widget.DrawerLayout 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/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:openDrawer="start">

    <include
        layout="@layout/app_bar_main"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <com.google.android.material.navigation.NavigationView
        android:id="@+id/nav_view"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="start"
        android:fitsSystemWindows="true"
        app:headerLayout="@layout/nav_header_main"
        app:menu="@menu/activity_main_drawer" />

</androidx.drawerlayout.widget.DrawerLayout>
```

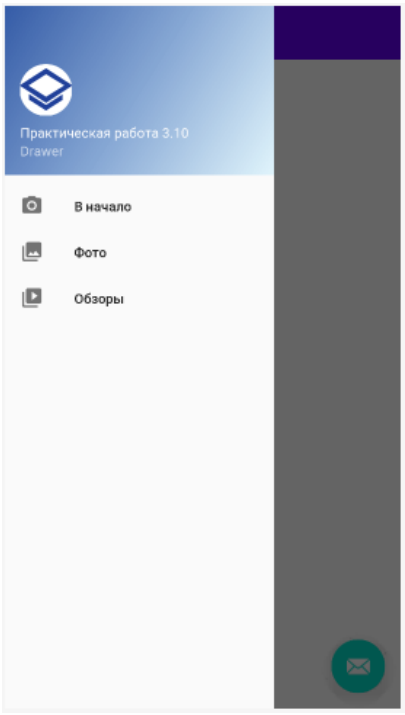
При этом в элементе NavigationView есть атрибут headerLayout, в котором задается файл с макетом верхней части шторки. В элементе шторки menu задается файл с меню действий на шторке. Рассмотрим макет res/layout/nav_header_main.xml . В нем присутствует один ImageView и два TextView.

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="@dimen/nav_header_height"
    android:background="@drawable/side_nav_bar"
    android:gravity="bottom"
    android:orientation="vertical"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:theme="@style/ThemeOverlay.AppCompat.Dark">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:contentDescription="@string/nav_header_desc"
        android:paddingTop="@dimen/nav_header_vertical_spacing"
        app:srcCompat="@mipmap/ic_launcher_round" />

    <TextView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:paddingTop="@dimen/nav_header_vertical_spacing"
        android:text="@string/nav_header_title"
        android:textAppearance="@style/TextAppearance.AppCompat.Body1" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/nav_header_subtitle" />

</LinearLayout>
```

Сами виджеты вполне простые, интереснее атрибуты заключающего их лайоута. Например атрибут background указывает на файл с фоном в котором задается градиент фона шторки:

```
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <gradient
        android:angle="135"
        android:centerColor="#009688"
        android:endColor="#00695C"
        android:startColor="#4DB6AC"
        android:type="linear" />
</shape>
```

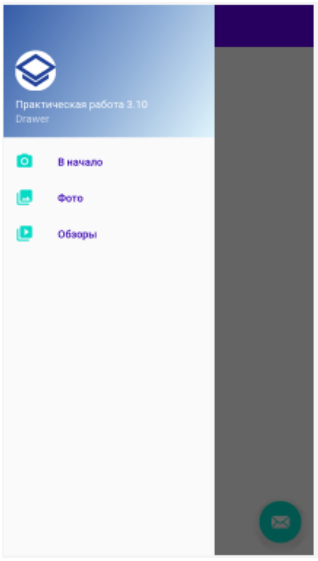
Изменяя эти файлы, а также файл ресурсов strings.xml можно легко создать нужный разработчику дизайн. Например такой:



Кроме того на внешний вид шторки влияет и непосредственно атрибуты самого виджета NavigationView в макете окна (см. начало). Например если добавить атрибуты:

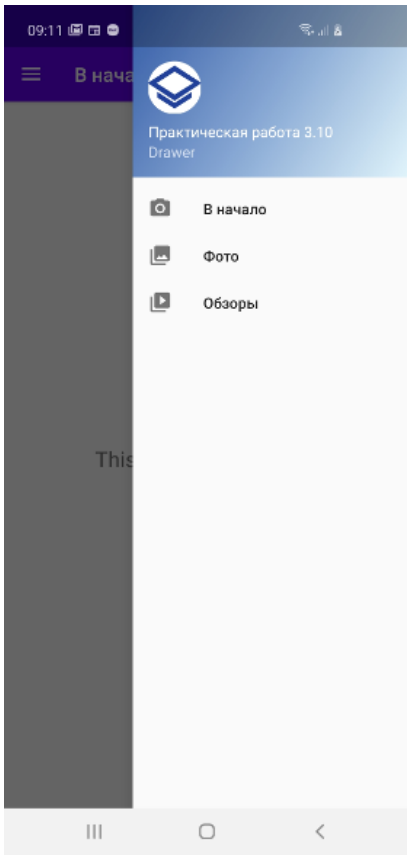
```
<com.google.android.material.navigation.NavigationView
   ....
   app:itemIconTint="@color/colorAccent"
   app:itemTextColor="@color/colorPrimaryDark"
   ....
   />
```

то на шторке изменится цвет элементов меню:



Можно даже добавить вторую шторку справа, хотя с точки зрения дизайна это и плохое решение. Для этого можно добавить в макет второй NavigationView с другим id и layout_gravity установленным в end.

```
<com.google.android.material.navigation.NavigationView
        android:id="@+id/nav_view_other"
        android:layout_width="wrap_content"
        android:layout_height="match_parent"
        android:layout_gravity="end"
        android:fitsSystemWindows="true"
        app:headerLayout="@layout/nav_header_main"
        app:menu="@menu/activity_main_drawer" />
```



Обратите внимание, что если использовать одну шторку с расположением справа (так иногда делают для языков с написанием справа на лево), то она будет работать корректно, за одним исключением: нажатие на drawer icon ≡ будет вызывать ошибку.

Не рассмотренным остался элемент макета **include**, который ссылается на файл ресурса **res/layout/app_bar_main.xml**. Он соотвествует шаблону Blank Activity, из Android Support Design. Только там он находился в файле **activity_main.xml**, а здесь он в файле **app_bar_main.xml**. Всё остальное осталось без изменений.

Теперь рассмотрим код активности для работы со шторкой.

3.10.2. Навигация

Переход по клику на пункты меню могут обрабатываться разными способами. В старом стиле (раньше 2018г, до появления Navigation API), в созданном мастером классе активности нужно было реализовать интерфейс **OnNavigationItemSelectedListener** с его методом **onNavigationItemSelected()**:

```
class MainActivity:AppCompatActivity(), NavigationView.OnNavigationItemSelectedListener {
  fun onNavigationItemSelected(item:MenuItem):Boolean {
    val id = item.getItemId()
    if (id == R.id.nav_home)
    {
      // Здесь обработка клика на первом пункте меню
    }
    else if (id == R.id.nav_gallery)
    {
      // Здесь обработка клика на втором пункте меню
    }
    else if (id == R.id.nav_slideshow)
    {
      // Здесь обработка клика на третьем пункте меню
    }
    val drawer = findViewById(R.id.drawer_layout) as DrawerLayout
    drawer.closeDrawer(GravityCompat.START)
    return true
  }
}
```

Open in Playground → Target: JVM Running on v.2.0.0

В современных версиях, реализован подход с использованием Navigation API. При этом в основном макете **activity_main.xml**, через **include** подключен макет **res/layout/app_bar_main**, в котором в свою очередь через второй **inclide** подключен **/res/layout/content_main**. Последний файл как раз и содержит контейнер навигации - **NavHostFragment**:

```
<fragment
        android:id="@+id/nav_host_fragment"
        android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:defaultNavHost="true"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:navGraph="@navigation/mobile_navigation" />
```

Как было рассказано в лекции 3.8 - **NavHostFragment** это новое API, которое позволяет существенно упростить и унифицировать процесс навигации. Собственно при его использовании никак специально не нужно прописывать никаких Listener-ов, как в примере выше. Теперь код значительно упростился:

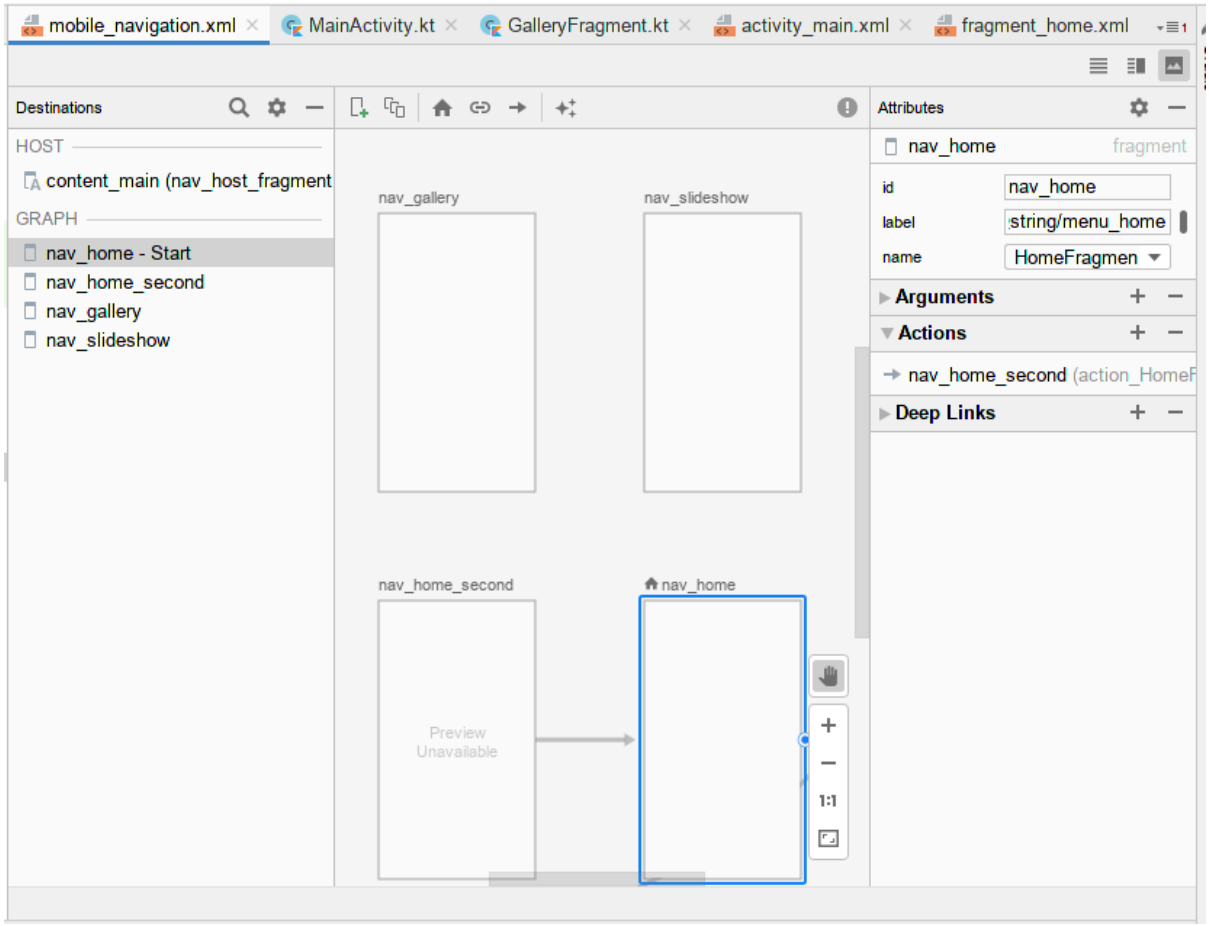
```
class MainActivity : AppCompatActivity() {
    private lateinit var appBarConfiguration: AppBarConfiguration
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
        // ......
        val navView: NavigationView = findViewById(R.id.nav_view) // шторка
        val navController = findNavController(R.id.nav_host_fragment)  // находим NavController
        // ......
        navView.setupWithNavController(navController) // устанавливается NavController для шторки
    }
```

Open in Playground → Target: JVM Running on v.2.0.0

Собственно теперь навигацией управляет NavController, осуществляя навигационные переходы как они описаны в меню шторки, в файле **res/menu/activity_main_drawe.xml**:

```
<item
          android:id="@+id/nav_home"
          android:icon="@drawable/ic_menu_camera"
          android:title="@string/menu_home" />
```

где атрибут **id** совпадает с ID фрагментов добавленных в граф навигации **res/navigation/mobile_navigatioon.xml**, как это было описано в лекции 3.8.



Начать тур для пользователя на этой странице