

ЛЕКЦИЯ I © 2020 Нет Ит

Android: Navigation

Теодор Костадинов



SOFTWARE
ACADEMY

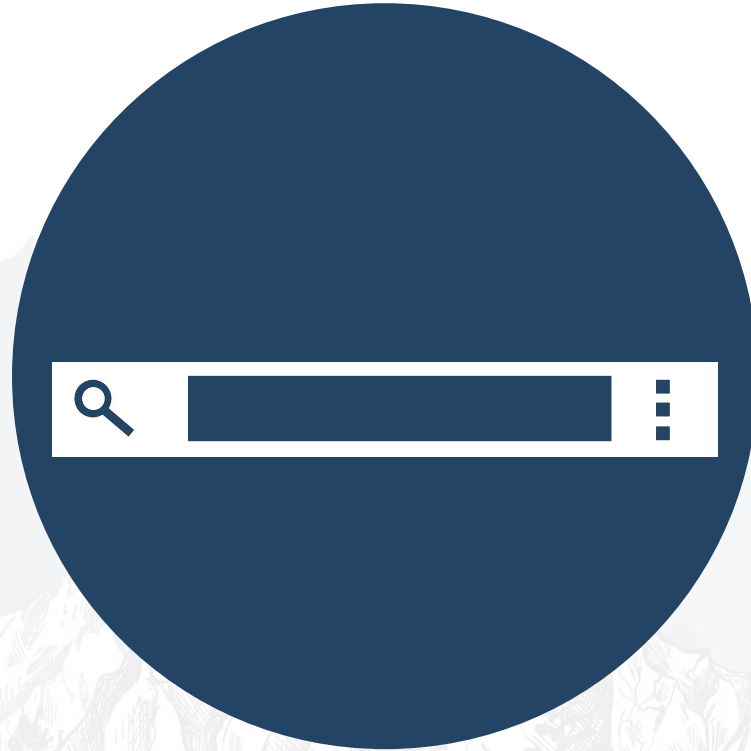


Съдържание

1. Преговор
2. Toolbar
3. Drawer Layout
4. Bottom Navigation
5. Tabs
6. Back vs Up
7. Nav Controller



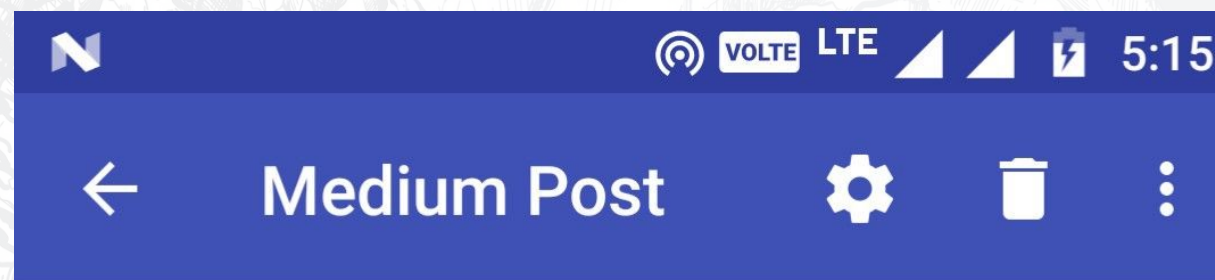
Преговор



Toolbar

Toolbar

- Потребителите имат нужда от контекст - от начин да разберат на кой екран се намират. На помощ идва AppBar-a / ActionBar-a / ToolBar-a
- По подразбиране, темата, която апп-овете имат зададена включва дефолтен ActionBar.
- Той не се препоръчва, защото има различно поведение на различните андроид версии.
- По-добре да се използва ToolBar, който идва от библиотека и ще е един и същ навсякъде.



Toolbar

dependency > implementation 'androidx.appcompat:appcompat:1.2.0'

view > androidx.appcompat.widget.Toolbar

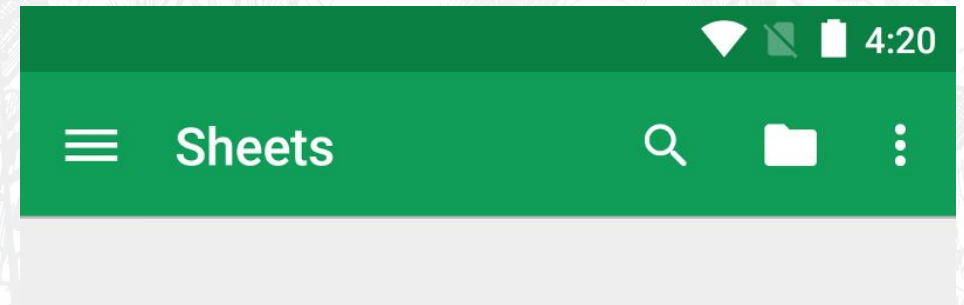
Methods:

`.inflateMenu(R.menu.main_manu)`

`.setSupportActionBar(toolbar)`

Menu

- Към всеки AppBar могат да се добавят бутони и менюта.
- Менюто се дефинира с xml.
- Ако сетваме тулбара като екшънбар:
 - Менюто трябва да се добави към AppBar-а в onCreateOptionsMenu(Menu menu)
 - Може да се слуша за клик евенти в onOptionsItemSelected(MenuItem item)
- Иначе може да се използва inflateMenu метода върху тулбара



menu/main_menu.xml

```
<menu >
```

```
  <item
```

```
    android:id="@+id/menu_main_setting"
```

```
    android:icon="@drawable/ic_settings"
```

```
    android:orderInCategory="100"
```

```
    app:showAsAction="always/ifRoom/never/withText/collapseActionView"
```

```
    android:title="Setting" />
```

```
</menu>
```




Drawer Layout

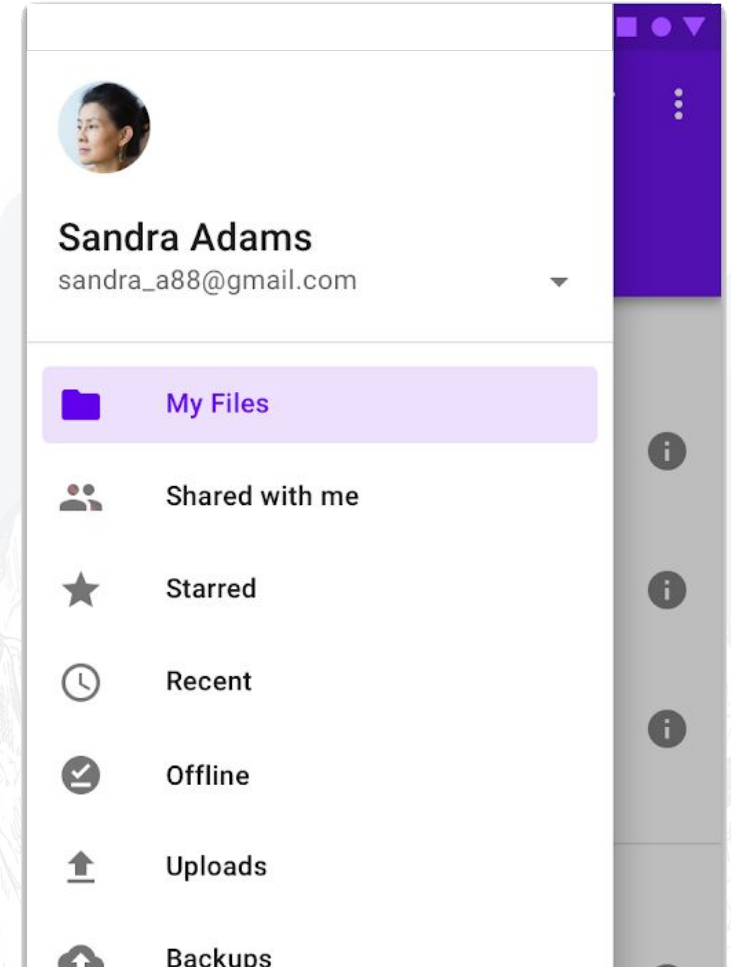
Drawer Layout

DrawerLayout-а е меню, което се появява отстрани на екрана. В други ОС се нарича hamburger menu.

Обикновено съдържа основната навигация в един апп.

Ресурс:

<https://developer.android.com/guide/navigation/navigation-ui>



Drawer Layout

implementation 'androidx.drawerlayout:drawerlayout:1.1.1'

implementation 'com.google.android.material:material:1.2.1'

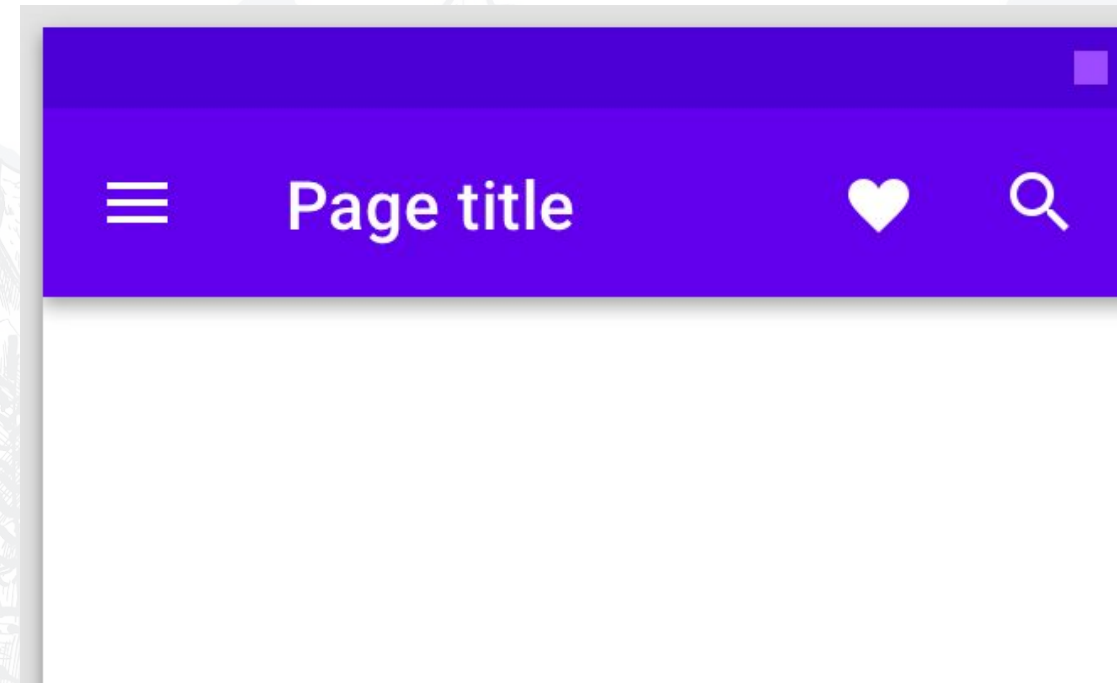
androidx.drawerlayout.widget.DrawerLayout

com.google.android.material.navigation.NavigationView

Drawer Layout

DrawerLayout-а трябва да съдържа NavigationView, което ще съдържа фрагментите.

DrawerLayout-а може без NavigationView.



Navigation View

```
app:menu="@menu/navigation"
```

```
NavigationView.setNavigationItemSelectedListener(Navigation  
View.OnNavigationItemSelectedListener)
```

```
boolean onNavigationItemSelectedListener(MenuItem item)
```

menu/navigation.xml

```
<menu >
```

```
  <item
```

```
    android:checkable="true" // or false
```

```
    android:id="@+id/itemIdentifier"
```

```
    android:title="Item Title" />
```

```
</menu>
```




Bottom Navigation

Bottom Navigation

Най-често използваният компонент за навигация.

Предпочита се пред NavDrawer-а, защото прави основните страници на апп-а видими за потребителя и по-кликвани.

Представява табове, които са в долната част на екрана.

Bottom Navigation

implementation

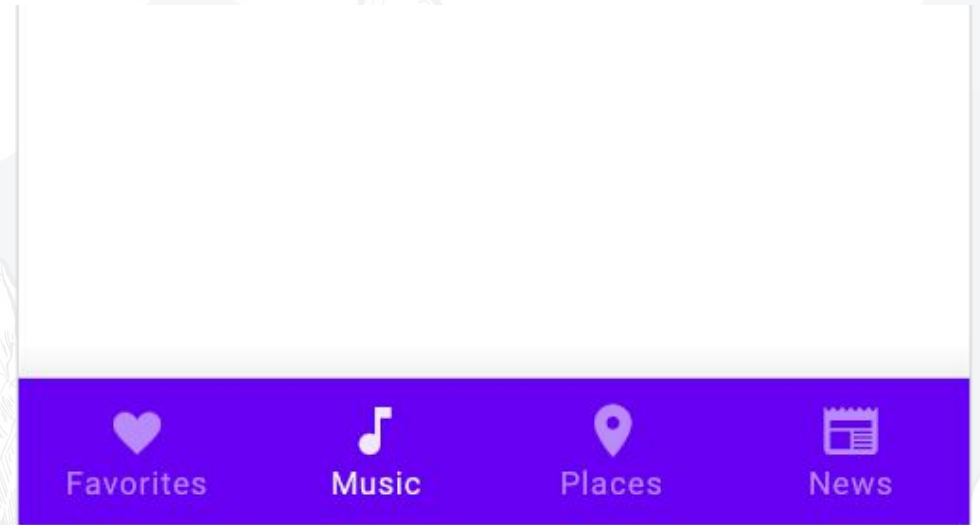
'com.google.android.material:material:1.2.1'

com.google.android.material.bottomnavigation.BottomNavigationView

Bottom Navigation

```
app:menu="@menu/my_navigation_
items"
```

```
.setOnNavigationItemSelectedListener(  
    OnNavigationItemSelectedListener)
```



ПОЧИВКА
до 20:25





Tabs

Tabs

- Табовете вече не са толкова предпочитани, колкото долната навигация, заради неудобното им разположение в горната част на екрана.
- Въпреки това все още се използват.
- Могат да се видят екрани и с табове и с долна навигация едновременно, но е лоша практика.

implementation 'com.google.android.material:material:1.2.1'

implementation "androidx.viewpager2:viewpager2:1.0.0"

com.google.android.material.tabs.TabLayout
androidx.viewpager2.widget.ViewPager2

Pagers

- Съществува специално view за табове: ViewPager.
- То може бързо да сменя между няколко екрана, показвайки хоризонтална анимация.
- То може да отчита и суайпване.
- За да работи, има нужда от адаптер, подобно на RecyclerView-то.

Pager Adapter

```
public class TabAdapter extends FragmentStateAdapter {  
  
    @Override public Fragment createFragment(int position)  
  
    @Override public int getItemCount()  
  
}
```

Pager Adapter

За сетъпване на PageView-то се извикват следните методи:

```
viewPager.setAdapter(tabAdapter)  
viewPager.setUserInputEnabled(false)  
viewPager.registerOnPageChangeCallback
```


Tab Mediator

Използва се за сменяне на таб-а на клик автоматично.

```
new TabLayoutMediator(TabLayout, ViewPager2,  
TabConfigurationStrategy)
```

```
tabMediator.attach()
```



Back vs Up

Back vs Up

Софтуерния/хардуерния андроид бутон “назад” (Back) връща на предното активити. Ако няма такава - затваря приложението.

Стрелката назад в туулбара (Up) връща назад в йерархията от екрани.

Така поне е обяснено в документацията на Андроид. В действителност често двата бутона правят едно и също.

Up

Показваме бутона, в активитито, за назад в тулбара така:

```
getSupportActionBar().setDisplayHomeAsUpEnabled(true)  
getSupportActionBar().setDisplayHomeAsUpEnabled(true)
```


Back

Можем да засечем клик на хардуерния/софтуерния бутон назад като override-нем `onBackPressed()` в активитито.

Ако решим може да не връщаме назад, ако не извикаме `onBackPressed()` на бащиния клас в нашият `onBackPressed()`.



NAVIGATION COMPONENT

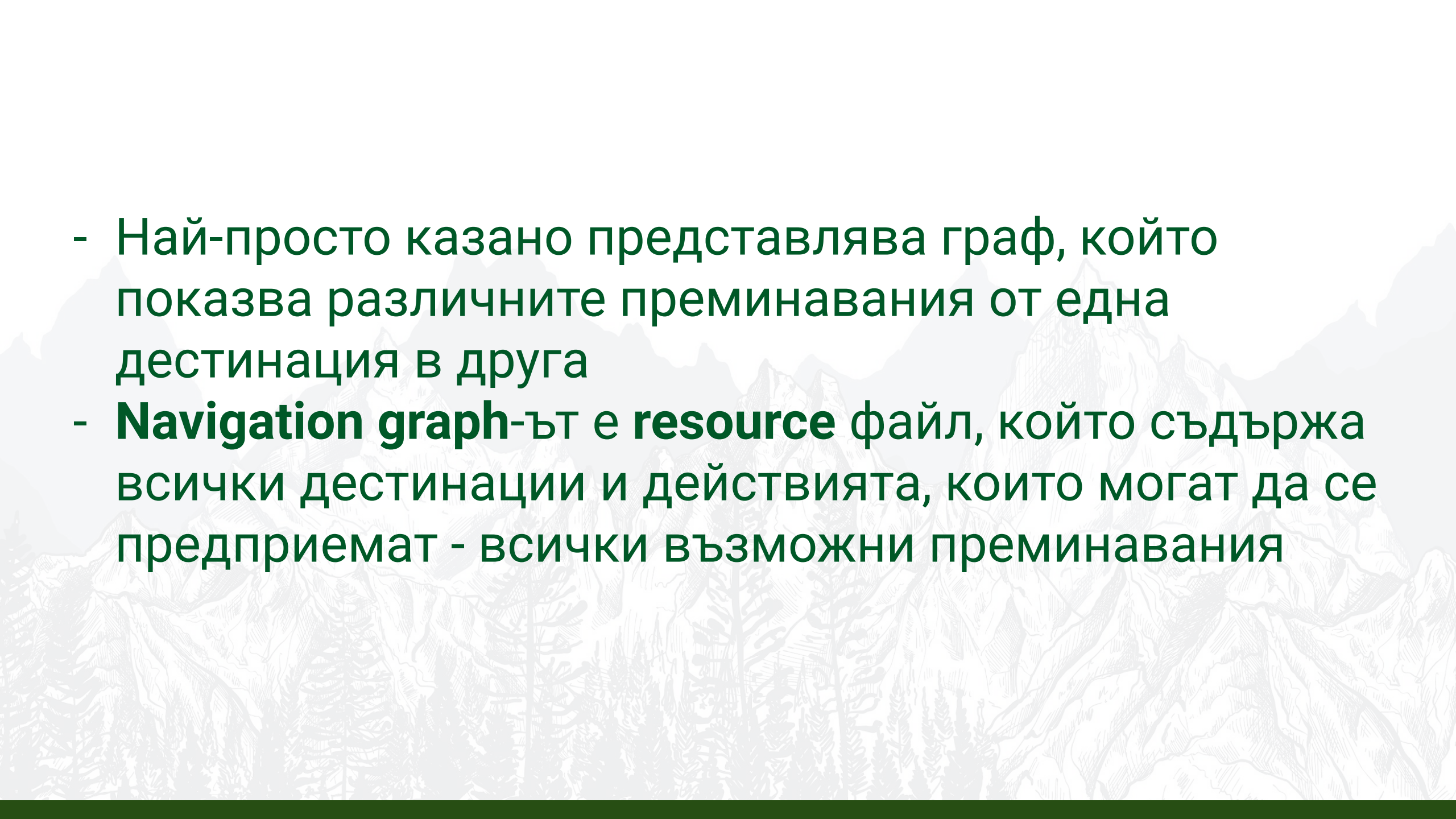
За да можем да ползваме Navigation Component:

```
android {  
  ...  
  compileOptions {  
    sourceCompatibility JavaVersion.VERSION_1_8  
    targetCompatibility JavaVersion.VERSION_1_8  
  }  
}
```

Трябва да можем
да използваме Java
8

```
dependencies {  
  def nav_version = "2.3.0"  
  def fragment_version = "1.2.5"
```

```
  implementation "androidx.fragment:fragment:$fragment_version"  
  implementation "androidx.navigation:navigation-fragment:$nav_version"  
  implementation "androidx.navigation:navigation-ui:$nav_version"  
}
```

- 
- Най-просто казано представлява граф, който показва различните преминавания от една дестинация в друга
 - **Navigation graph**-ът е **resource** файл, който съдържа всички дестинации и действията, които могат да се предприемат - всички възможни преминавания

Дестинации
и

Показва ни
възможните
преминавания от
един екран към
останалите

fragmentMain

VIEW TRANSACTIONS

SEND MONEY

VIEW BALANCE

1

fragmentViewTransact...

Food	\$200
Data	\$200
Rent	\$200
Candles	\$3600
Utility	\$200

fragmentChooseRecipi...

Recipient
Name

NEXT CANCEL

fragmentViewBalance

\$1

fragmentSpecifyAmount

Amount
\$0.00

SENDI CANCEL

fragmentConfirmation

\$0
was sent

Как да си създадем Navigation graph?

1. Отиваме на res папката
2. New > Android resource file
3. Въвеждаме подходящо име
4. Избираме **Navigation** от **Resource type** падащото меню

nav_graph.xml

fragment_view_balance.xml

fragment_main.xml

Destinations

HOST

activity_main (fragment)

GRAPH

fragmentMain - Start

fragmentViewBalance

navigation

Съдържа визуалната репрезентация на графа

fragmentMain

VIEW TRANSACTIONS

VIEW BALANCE

SEND MONEY

fragmentViewBala...

navigation

Nested Graph

Показва ни текущия nav host и всички възможни дестинации

1

Показва ни атрибутите на текущия обект

3

Attributes

nav_graph

navigation

id

nav_graph

label

startDestination

fragmentMain

Argument Default Values

Global Actions

Deep Links

XML-ът на Navigation Component изглежда така:

```
<?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"  
            android:id="@+id/nav_graph">  
  
</navigation>
```




NavHost

```
<androidx.fragment.app.FragmentContainerView
```

```
    android:id="@+id/nav_host_fragment"
```

```
    android:name="androidx.navigation.fragment.NavHostFragment"
```

```
    android:layout_width="0dp"
```

```
    android:layout_height="0dp"
```

```
    app:layout_constraintLeft_toLeftOf="parent"
```

```
    app:layout_constraintRight_toRightOf="parent"
```

```
    app:layout_constraintTop_toTopOf="parent"
```

```
    app:layout_constraintBottom_toBottomOf="parent"
```

```
    app:defaultNavHost="true"
```

```
    app:navGraph="@navigation/nav_graph" />
```

Слагаме го като **част от дадено активити**, за да можем да използваме възможностите за навигация

Името на NavHost имплементацията

За да можем да използваме функционалността за back бутона

Асоциацията с Navigation graph

- За да добавим нова дестинация в графа, просто трябва да натиснем бутона за **New Destination** и в диалогов прозорец ще ни излязат всичките ни налични фрагменти и activity-та
- Имаме и възможност да добавим ***placeholder***, който да ни служи като временен шаблон, който после да заместим с конкретен фрагмент/activity

XML-ът на Navigation Component вече изглежда така:

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<navigation xmlns:app="http://schemas.android.com/apk/res-auto"  
  xmlns:tools="http://schemas.android.com/tools"  
  xmlns:android="http://schemas.android.com/apk/res/android"  
  app:startDestination="@id/authenticationActivity">
```

```
<activity
```

```
  android:id="@+id/authenticationActivity"
```

```
  android:name="com.example.AuthenticationActivity"
```

```
  android:label="AuthenticationActivity" />
```

```
<fragment
```

```
  android:id="@+id/blankFragment"
```

```
  android:name="com.example.BlankFragment"
```

```
  android:label="BlankFragment"
```

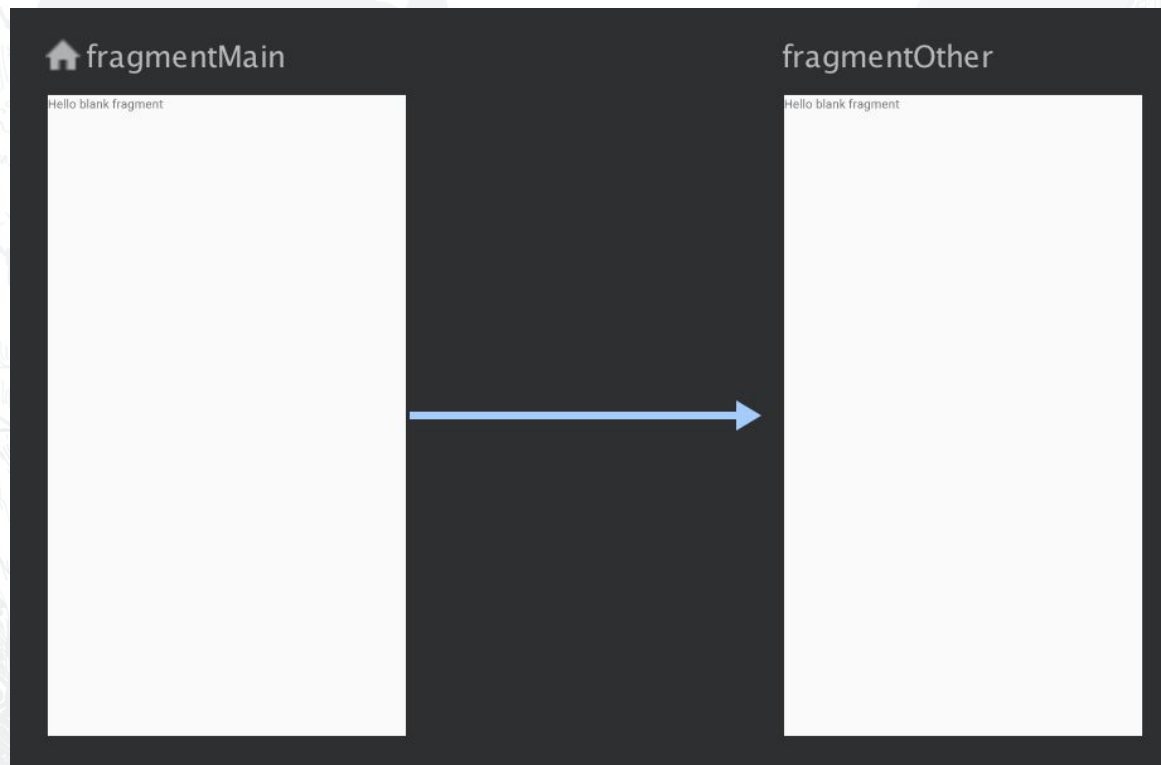
```
  tools:layout="@layout/fragment_blank" />
```

```
</navigation>
```

Тип на дестинацията


```
<fragment
    android:id="@+id/blankFragment"
    android:name="com.example.cashdog.cashdog.BlankFragment"
    android:label="fragment_blank"
    tools:layout="@layout/fragment_blank" >
    <action
        android:id="@+id/action_blankFragment_to_blankFragment2"
        app:destination="@id/blankFragment2" />
</fragment>
```

Действието, което ще
се предприеме





NavController

- За да можем да навигираме до дадена дестинация, е нужно да използваме **NavController**
- Всеки **NavHost** има отговарящ му **NavController**

```
viewTransactionsButton.setOnClickListener(new View.OnClickListener() {  
    @Override  
    public void onClick(View view) {  
        Navigation.findNavController(view).navigate(R.id.viewTransactionsAction);  
    }  
});
```

ВЪПРОСИ?



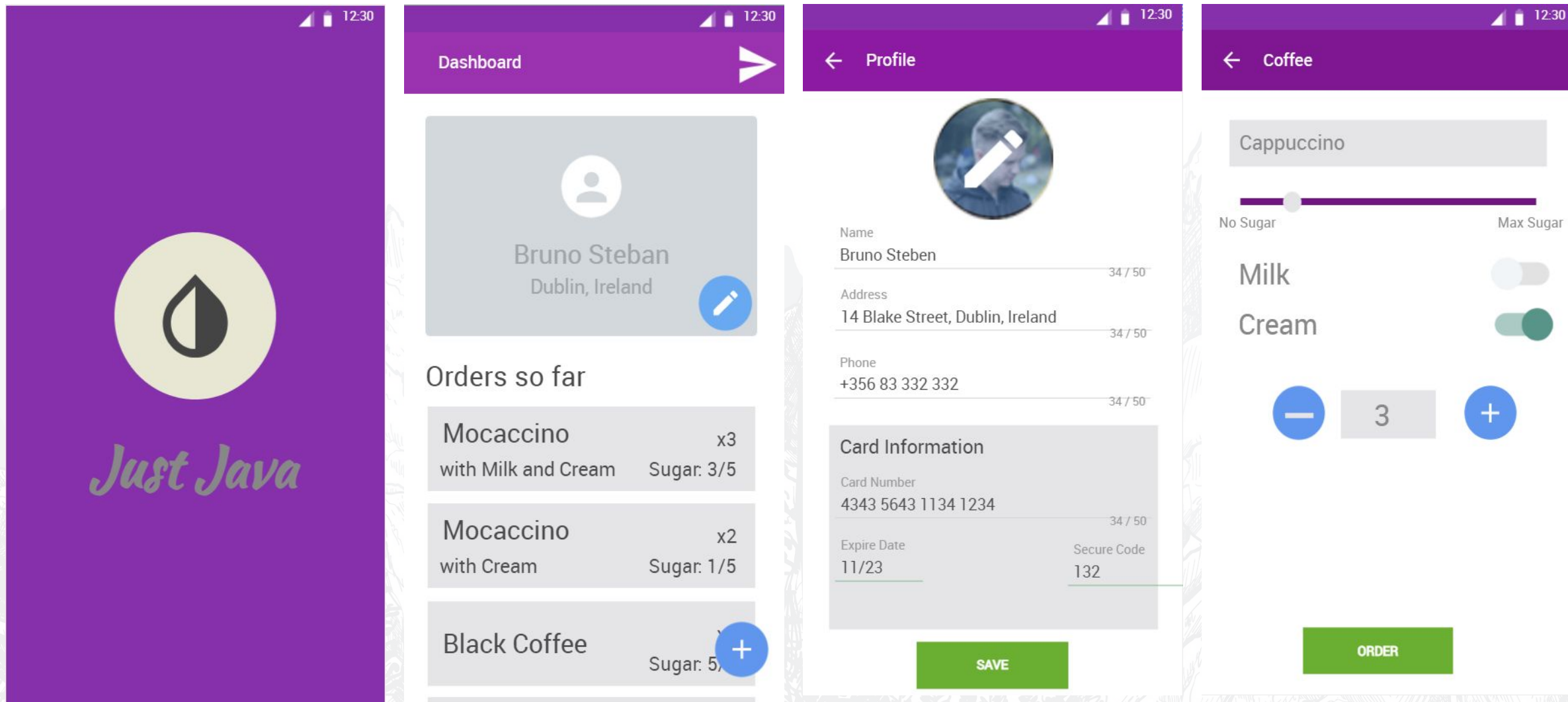


HOMEWORK

Създайте приложение за поръчка на кафе. Нека то се състои от:

- Сплаш екран, който се появява за 3 секунди и има логото и името на апа.
- Екран с поръчката до момента. Показана е информацията за потребителя и избраните кафета до този момент. Информацията не може да се редактира на този екран. Кликване на информацията за потребителя води до нов екран. Кликване на плюс бутона води до екран за добавяне на кафе. Екранът показва обща сума на поръчката до момента. Има бутон за изпращане, който отваря нов екран.
- Екран за въвеждане на информация за потребителя: Имена, телефон, адрес за доставка, информация за дебитна карта
- Екран за създаване на ново кафе: вид кафе, количество захар, мляко, сметана, бройки. Екранът автоматично изчислява цената.
- Бутонът за изпращане на основния екран отваря екран с резюме на поръчката и потвърждение. При потвърждение се отваря имейл клиента с текст: данните от поръчката.

Интерактивен дизайн: https://www.fluidui.com/editor/live/project/p_dPDllqrw79PU4bwehOH2bHTlIXaCiowr



© 2020 Нет Ит

БЛАГОДАРЯ ЗА ВНИМАНИЕТО!



SOFTWARE
ACADEMY

