

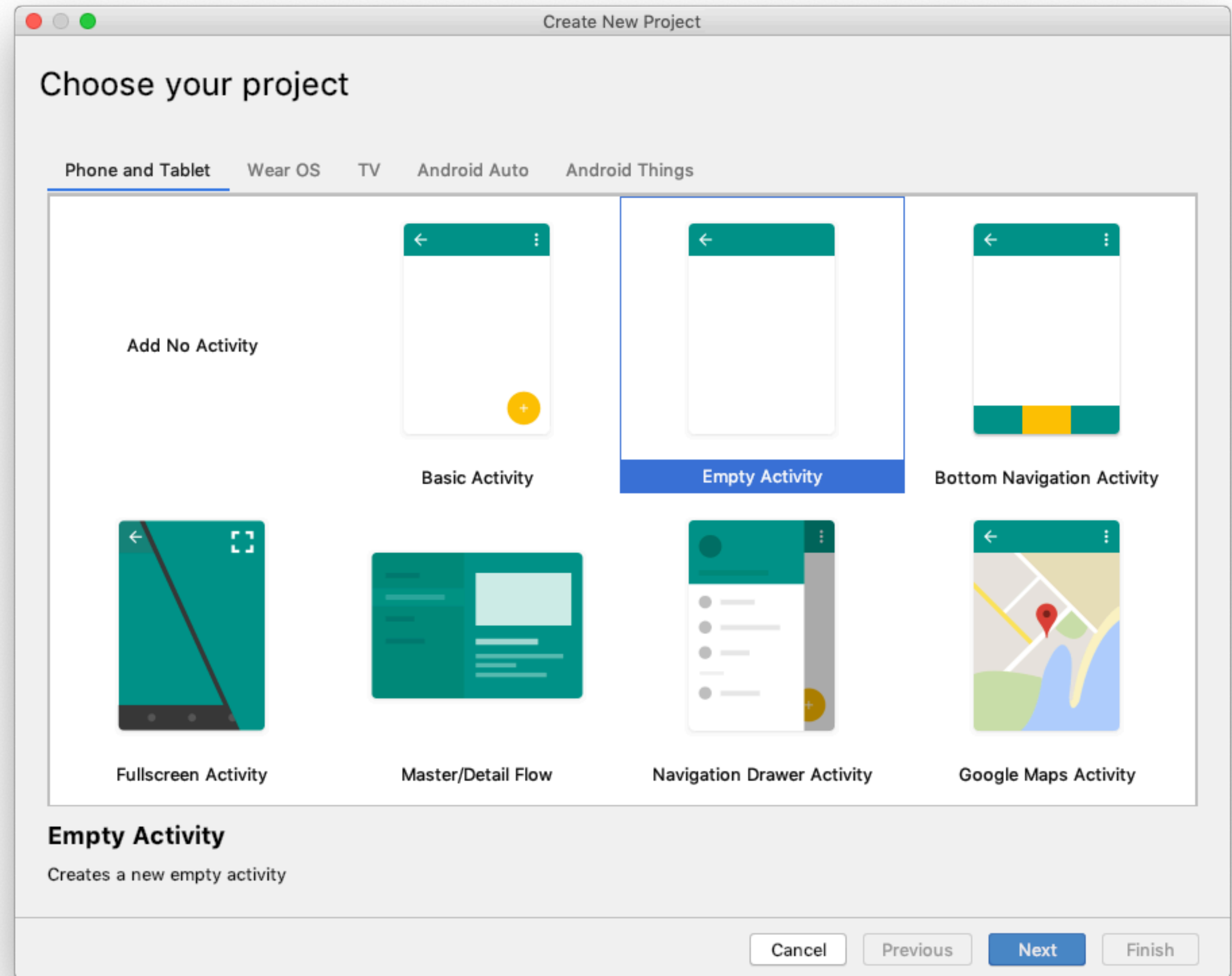


Navigation Components

Jetpack Architecture

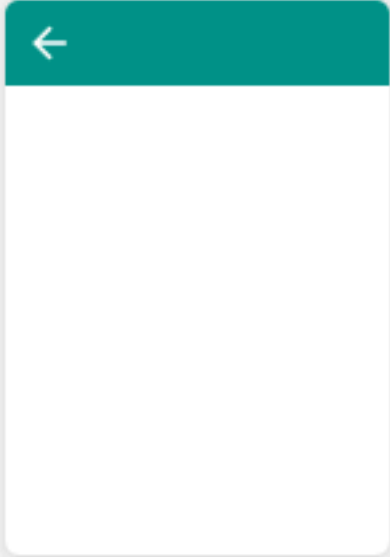
Features of Navigation Components

- Handling fragment transaction
- Handling Up and Back actions correctly
- Providing standardized resources for animations and transitions
- Including Navigation UI patterns
- Providing type safety when passing information
- Visualizing and editing navigation from Android Studio Navigation Editor



Create New Project

Configure your project



Empty Activity

Creates a new empty activity

Name
Shop

Package name
at.htl.shop

Save location
droid.kotlin/2019.jetpack.udemy/03.Navigation.Components/Shop

Language
Kotlin

Minimum API level
API 23: Android 6.0 (Marshmallow)

i Your app will run on approximately **62.6%** of devices.
[Help me choose](#)

This project will support instant apps

Use AndroidX artifacts

Cancel Previous Next Finish

Shop [~/CloudStation/htl/skripten/themen/android.kotlin/2019.jetpack.udemy/03.Navigation.Components/Shop] - .../app/src/main/java/at/htl/shop/MainActivity.kt [app]

Shop > app > src > main > java > at > htl > shop > MainActivity.kt

Project: 1: Project
Resource Manager
Structure
Favorites
Build Variants
Layout Captures

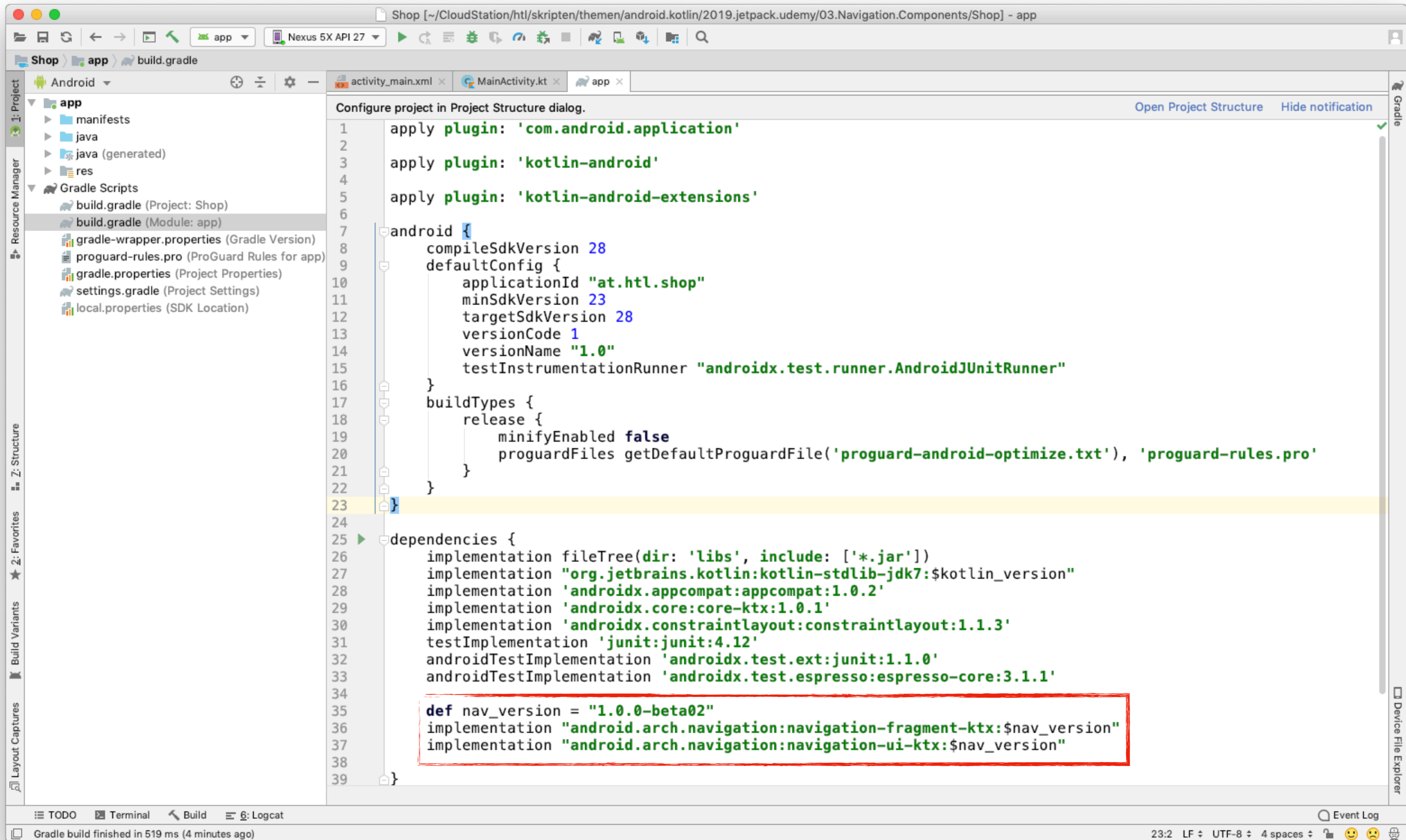
```
1 package at.htl.shop
2
3 import ...
4
5
6 class MainActivity : AppCompatActivity() {
7
8     override fun onCreate(savedInstanceState: Bundle?) {
9         super.onCreate(savedInstanceState)
10        setContentView(R.layout.activity_main)
11    }
12 }
13
```

Build: Build Output x Sync x

- Build: completed successfully at 2019-02-18 12:17 (682 ms)
- Run build /Users/stuetz/CloudStation/htl/skripten/themen/android.kotlin/2019.jetpack.udemy/03.Navigation.Components/Shop (270 ms)
 - Load build (2 ms)
 - Configure build (118 ms)
 - Calculate task graph (31 ms)
 - Run tasks (117 ms)

Event Log
Gradle build finished in 698 ms (today 12:17)

13:1 LF UTF-8 4 spaces



<https://developer.android.com/jetpack/androidx/releases/navigation>

values/strings.xml

```
<resources>
  <string name="app_name">Shop</string>
  <string name="home">Home</string>
  <string name="cart">Cart</string>
  <string name="hello_blank_fragment">Hello blank fragment</string>
  <string name="detail">We empower small and medium-sized businesses 1</string>
  <string name="about_us">About Us</string>
  <string name="shop">Shop</string>
</resources>
```

We empower small and medium-sized businesses to reach millions of customers with a number of programmes that help boost their revenue, reach and productivity.

values/styles.xml

```
<resources>
```

```
    <!-- Base application theme. -->
```

```
    <style name="AppTheme" parent="Theme.AppCompat.Light.NoActionBar">
```

```
        <!-- Customize your theme here. -->
```

```
        <item name="colorPrimary">@color/colorPrimary</item>
```

```
        <item name="colorPrimaryDark">@color/colorPrimaryDark</item>
```

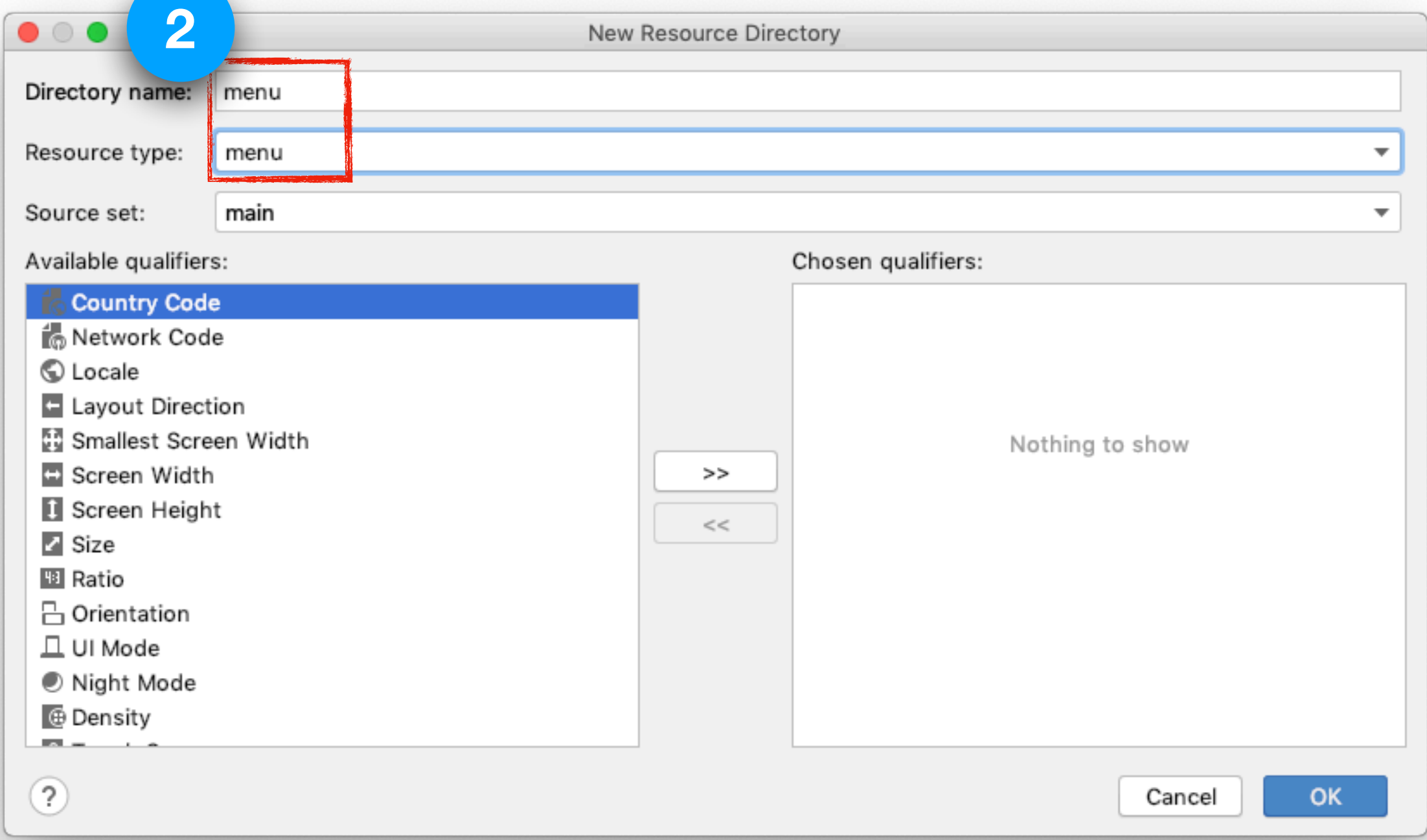
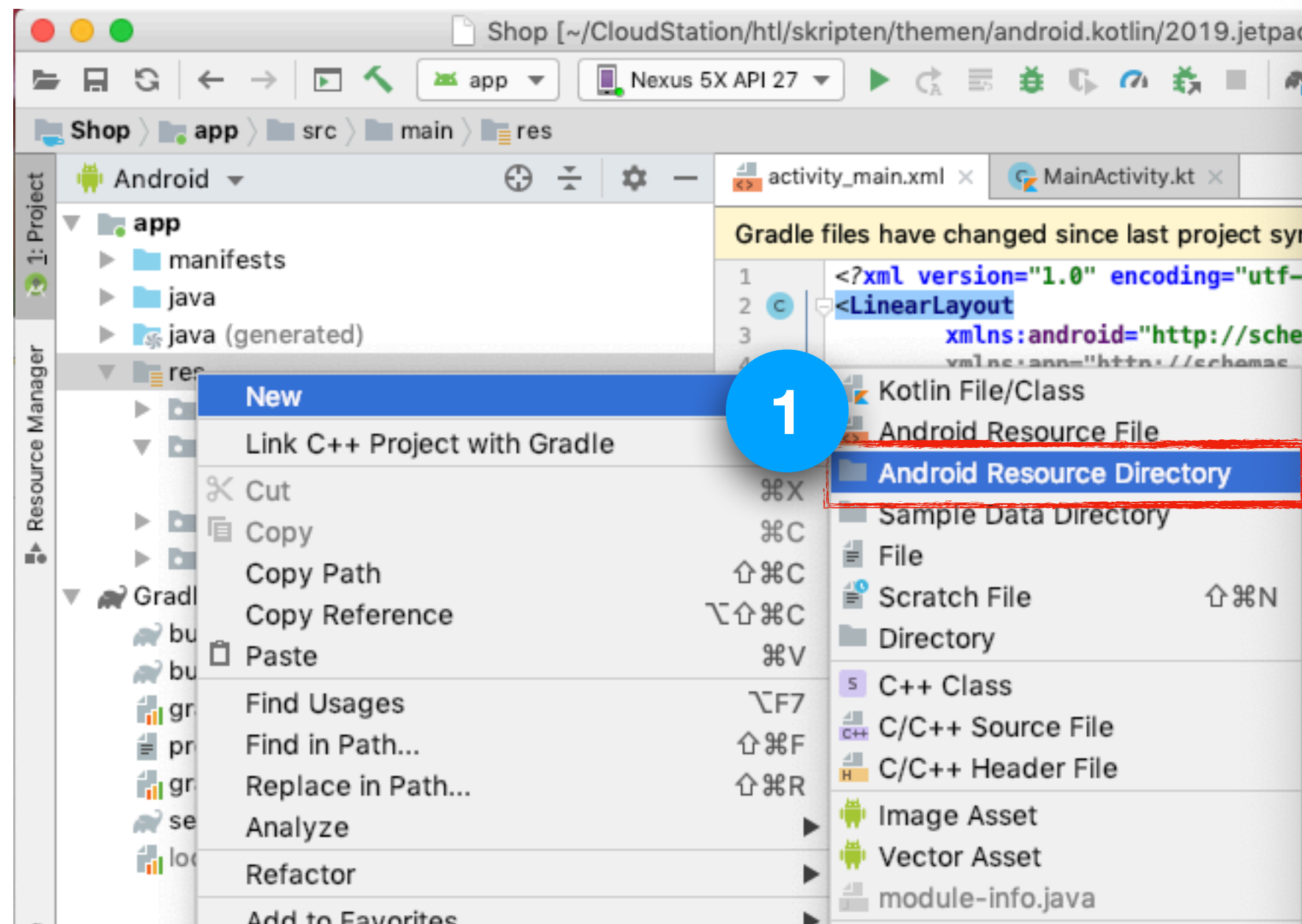
```
        <item name="colorAccent">@color/colorAccent</item>
```

```
    </style>
```

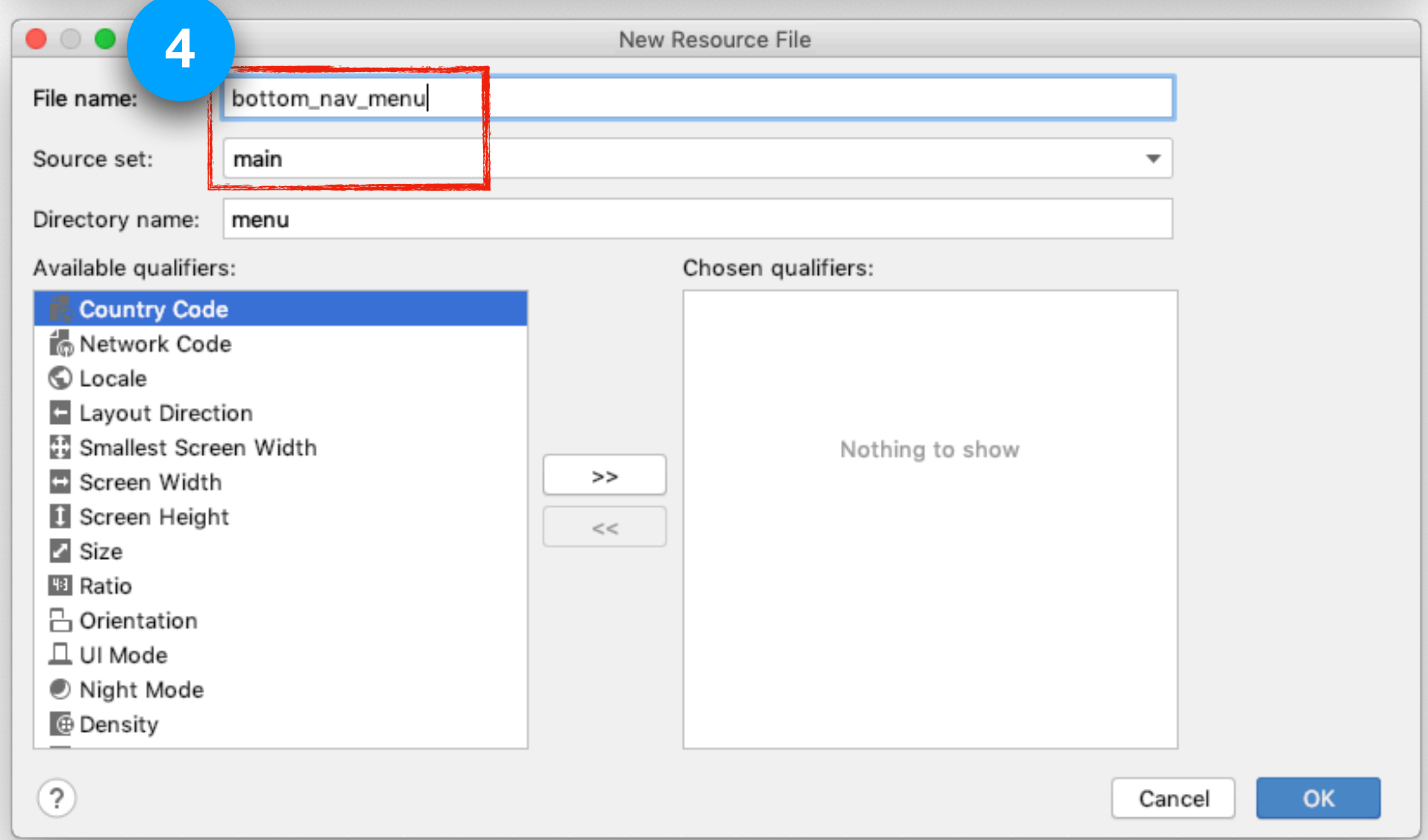
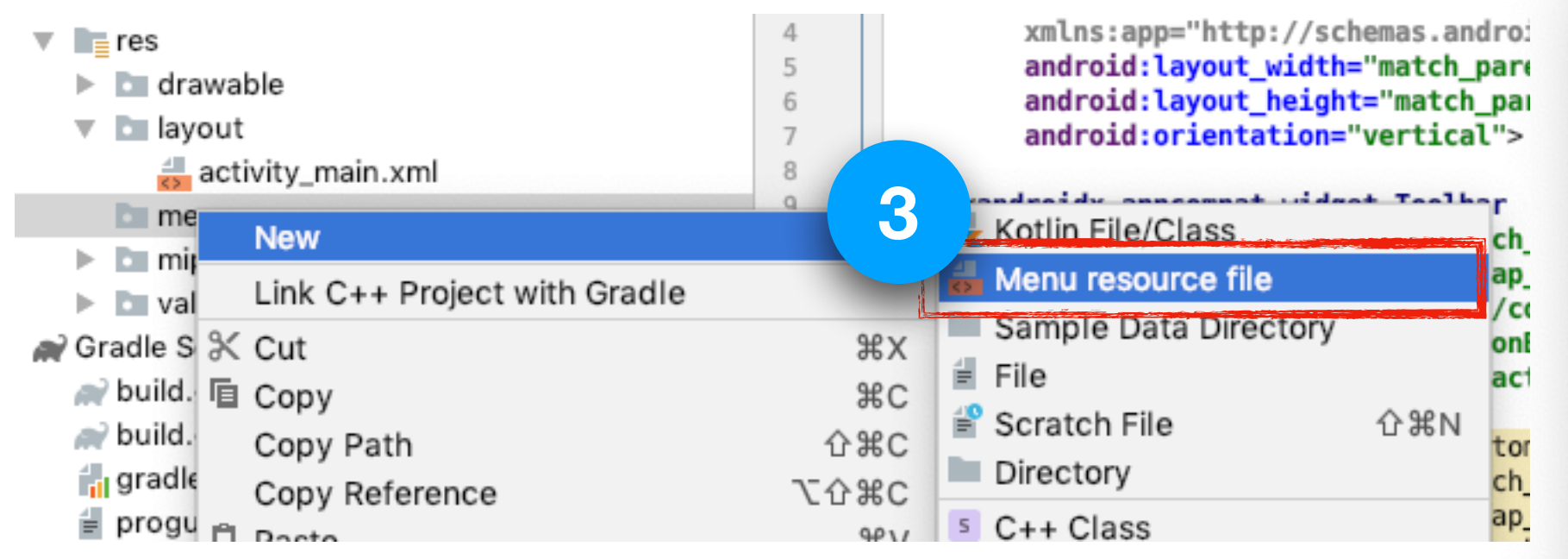
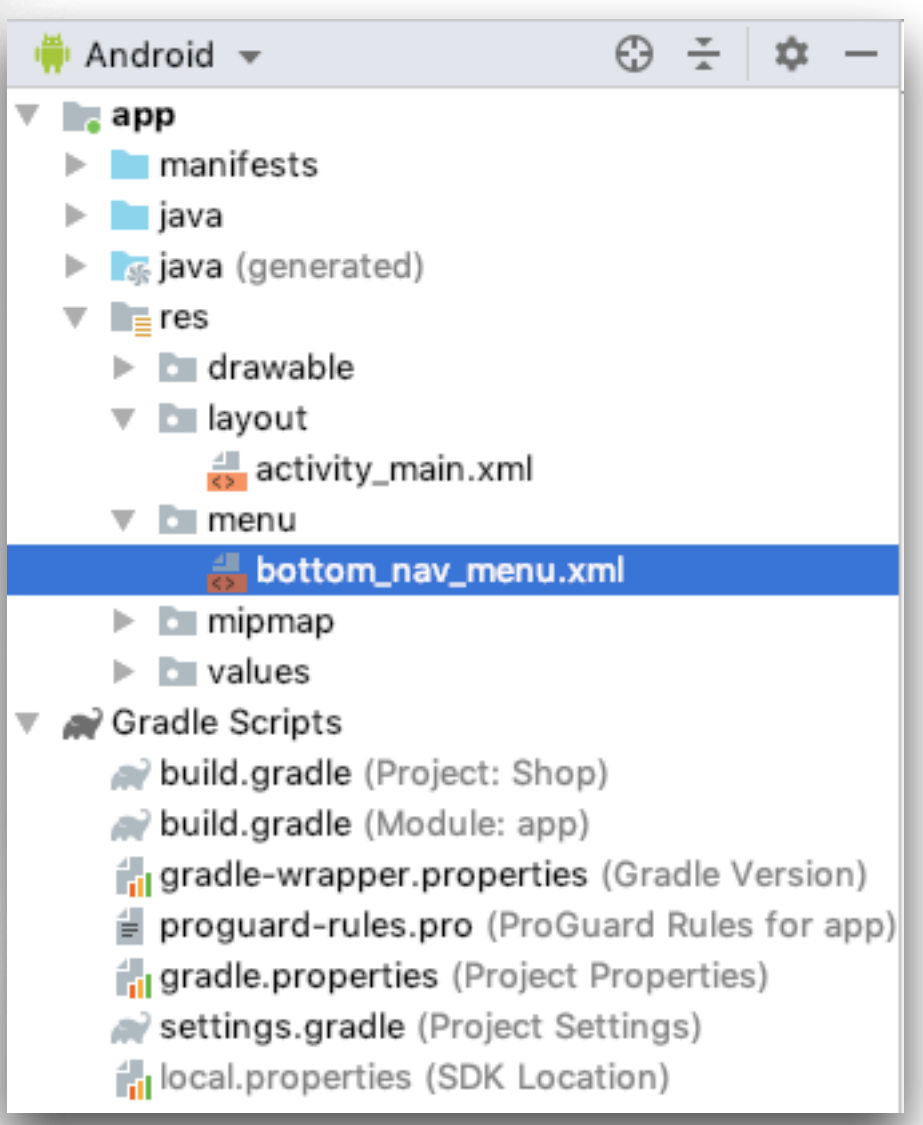
```
</resources>
```

values/colors.xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="colorPrimary">#3F51B5</color>
    <color name="colorPrimaryDark">#303F9F</color>
    <color name="colorAccent">#D81B60</color>
</resources>
```

Erstellen des Menü-Ressourcenordners und eines Menüs



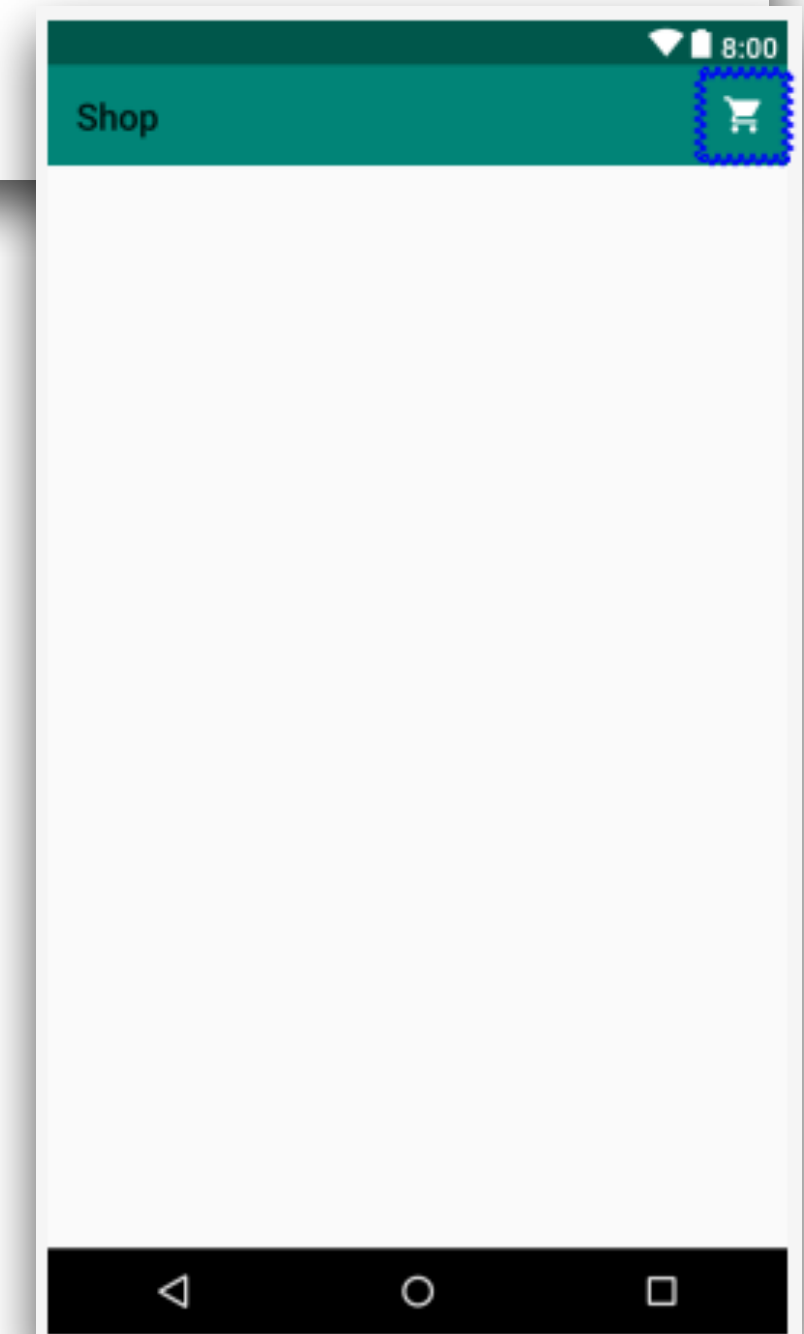
Menüs

```
bottom_nav_menu.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <menu xmlns:android="http://schemas.android.com/apk/res/android">
3
4      <item
5          android:id="@+id/home_destination"
6          android:title="Home"
7          android:icon="@drawable/ic_home"
8      />
9
10     <item
11         android:id="@+id/shop_destination"
12         android:title="Shop"
13         android:icon="@drawable/ic_shop"
14     />
15 </menu>
```



Erstellen Sie ein weiteres Menü: toolbar_menu.xml

```
toolbar_menu.xml
1  <?xml version="1.0" encoding="utf-8"?>
2  <menu xmlns:android="http://schemas.android.com/apk/res/android"
3        xmlns:app="http://schemas.android.com/apk/res-auto">
4
5      <item
6          android:id="@+id/cart_destination"
7          android:icon="@drawable/ic_shopping_cart"
8          android:title="Cart"
9          app:showAsAction="ifRoom"
10     />
11 </menu>
```



activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<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="match_parent"
    android:orientation="vertical">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/colorPrimary"
        android:theme="@style/ThemeOverlay.AppCompat.Dark"/>

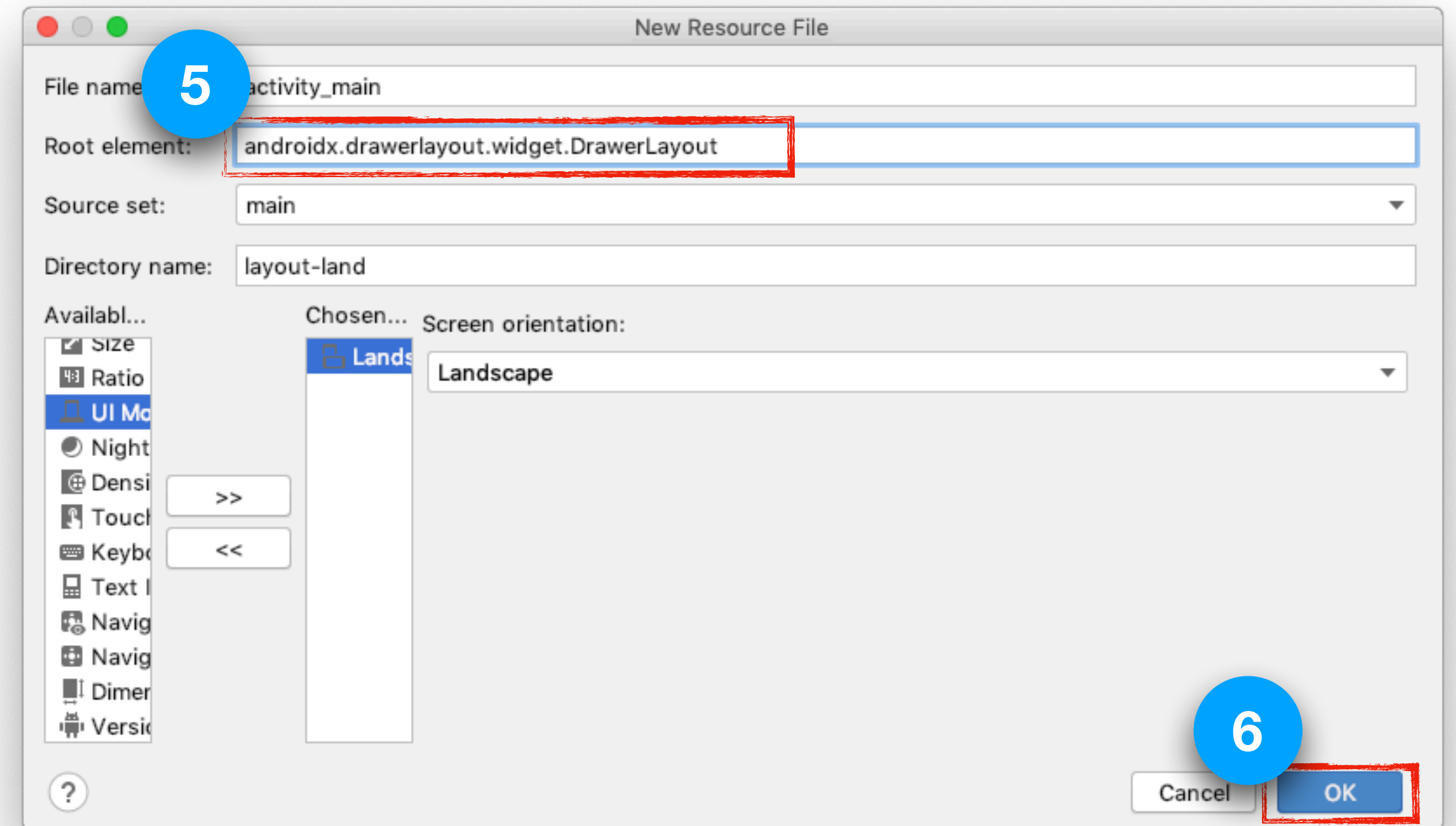
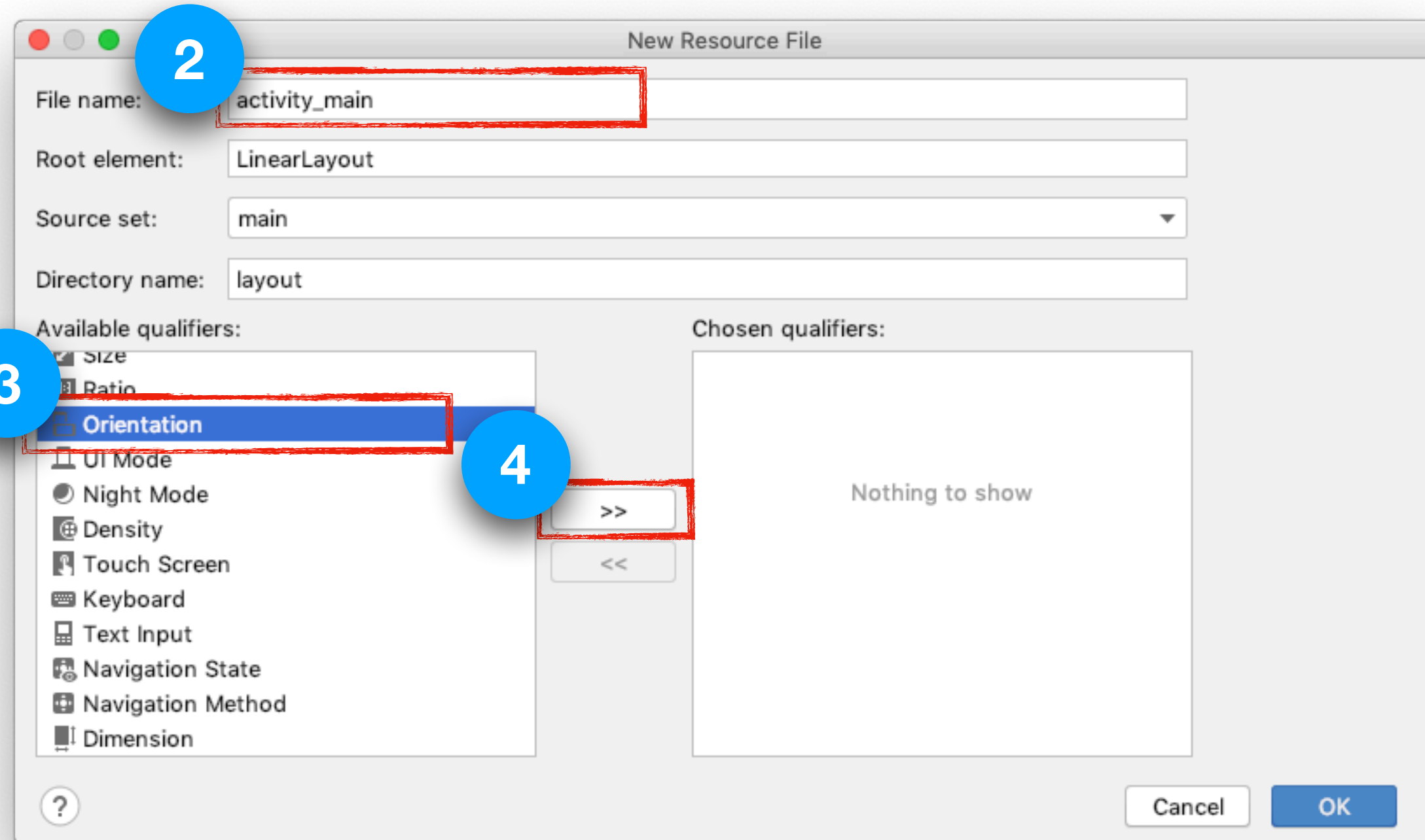
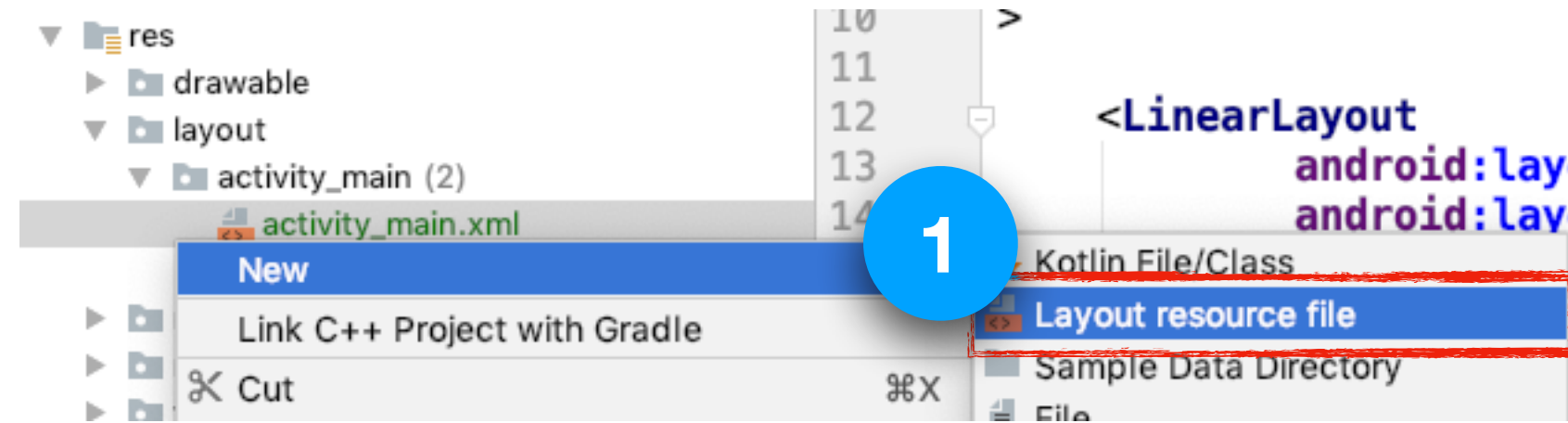
    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/bottom_nav"
        app:menu="@menu/bottom_nav_menu"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>

</LinearLayout>
```

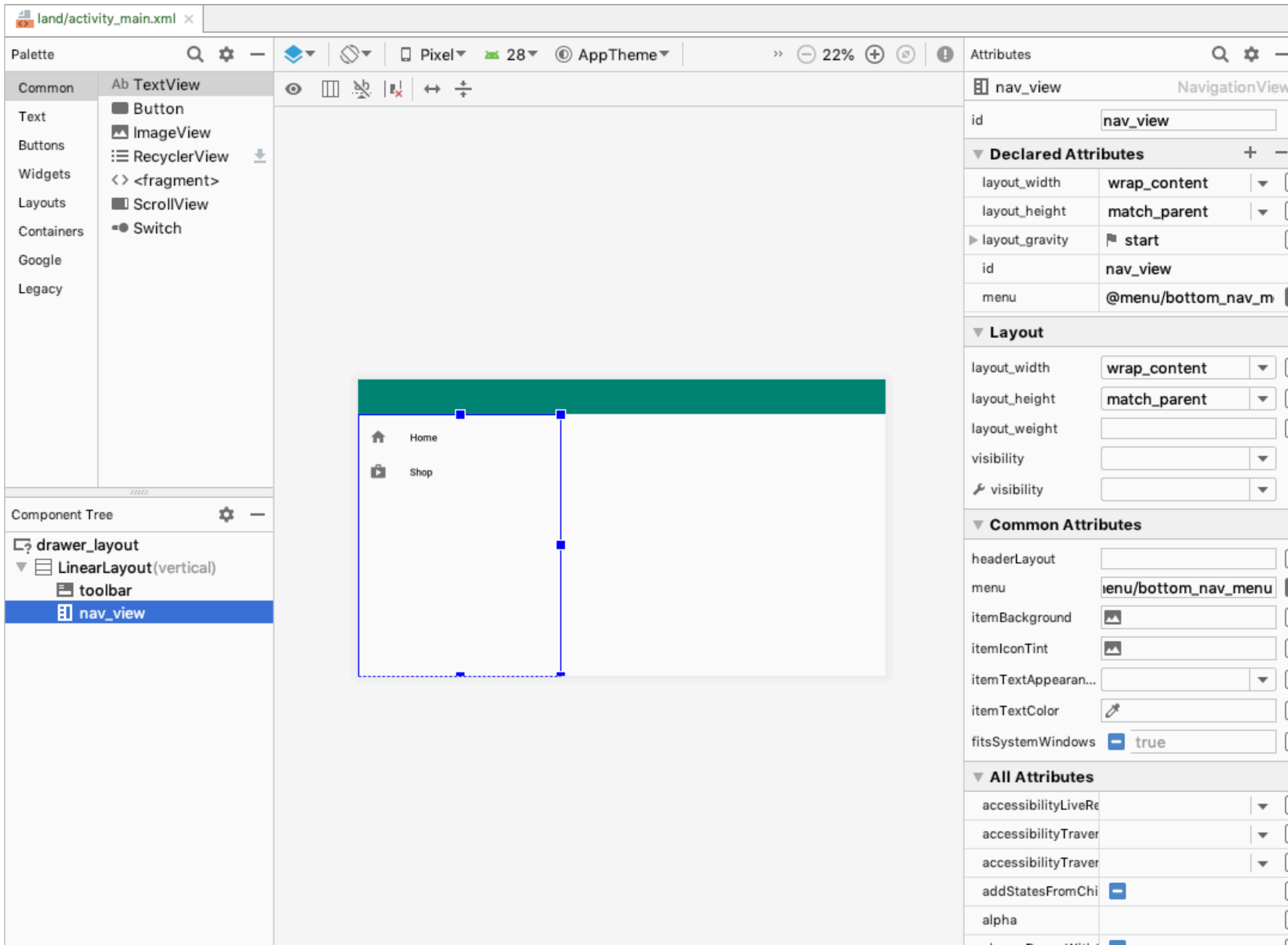
Wird automatisch ins build.gradle eingefügt

```
implementation 'com.google.android.material:material:1.0.0'
```

activity_main für Landscape erstellen



land/activity_main.xml



```
<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
```

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/colorPrimary"
        android:theme="@style/ThemeOverlay.AppCompat.Dark"/>

    <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"
        app:menu="@menu/bottom_nav_menu"/>
</LinearLayout>
```

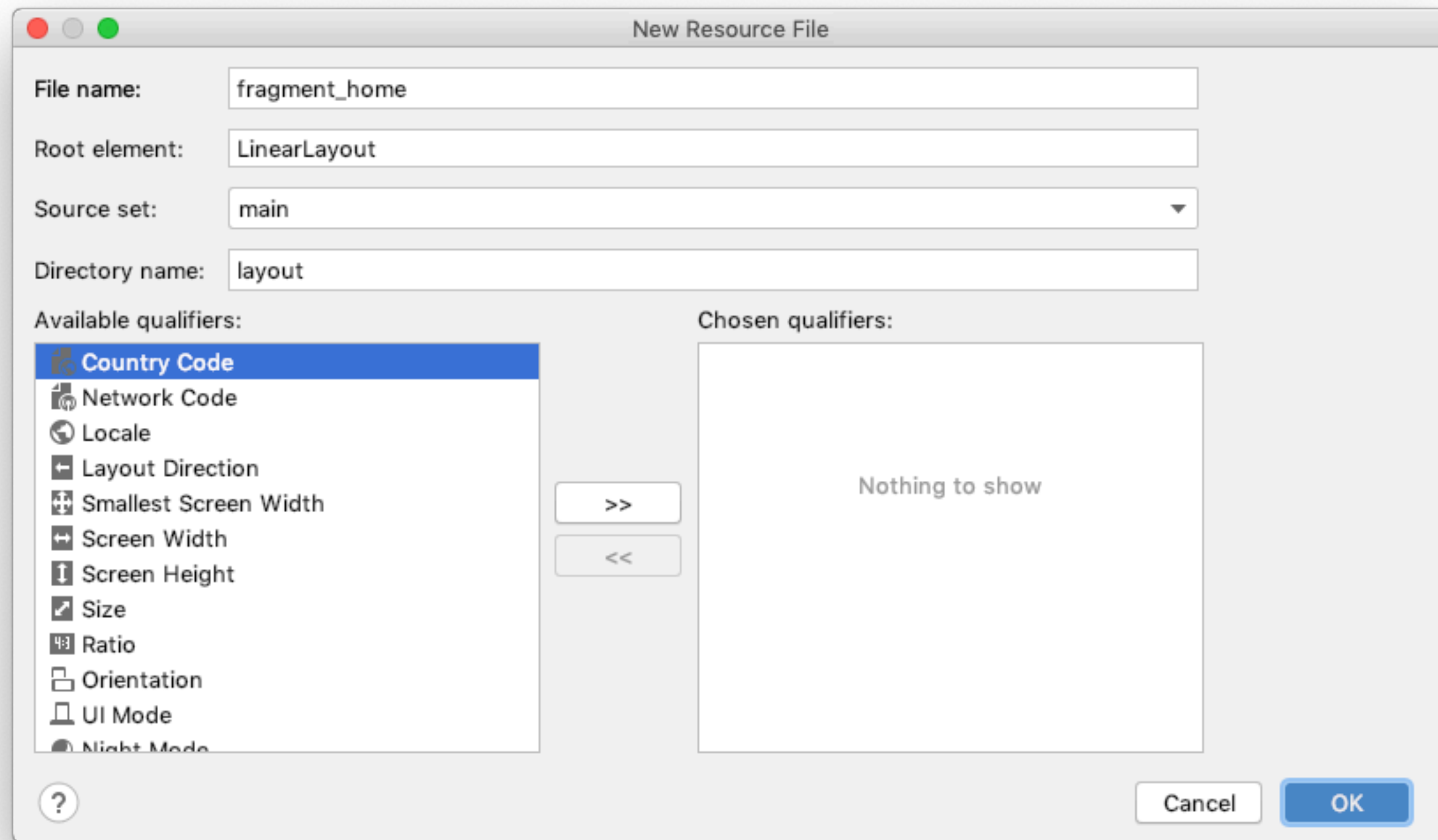
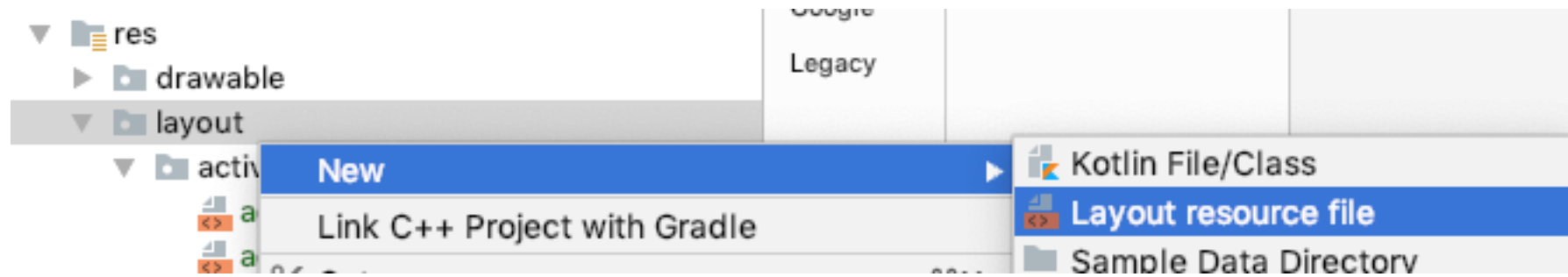
```
</androidx.drawerlayout.widget.DrawerLayout>
```

Was sind DrawerLayouts?

<https://youtu.be/DkT0vS14Um0>

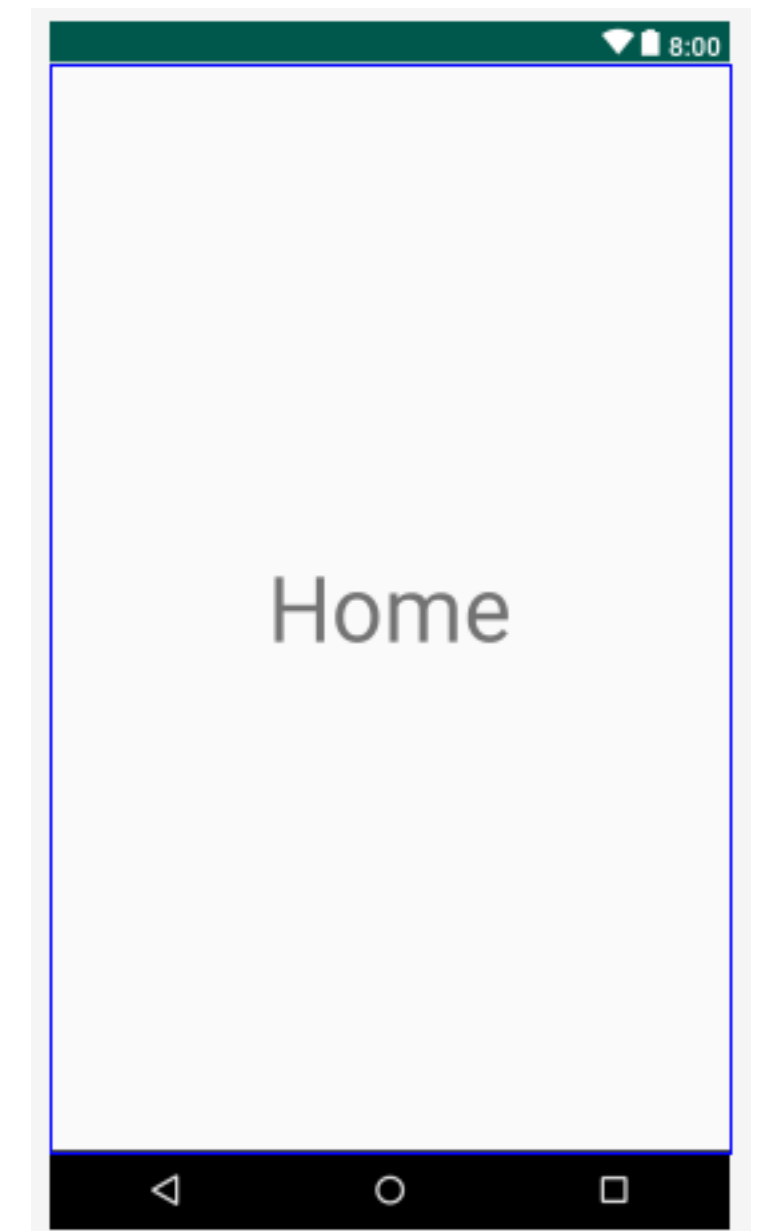
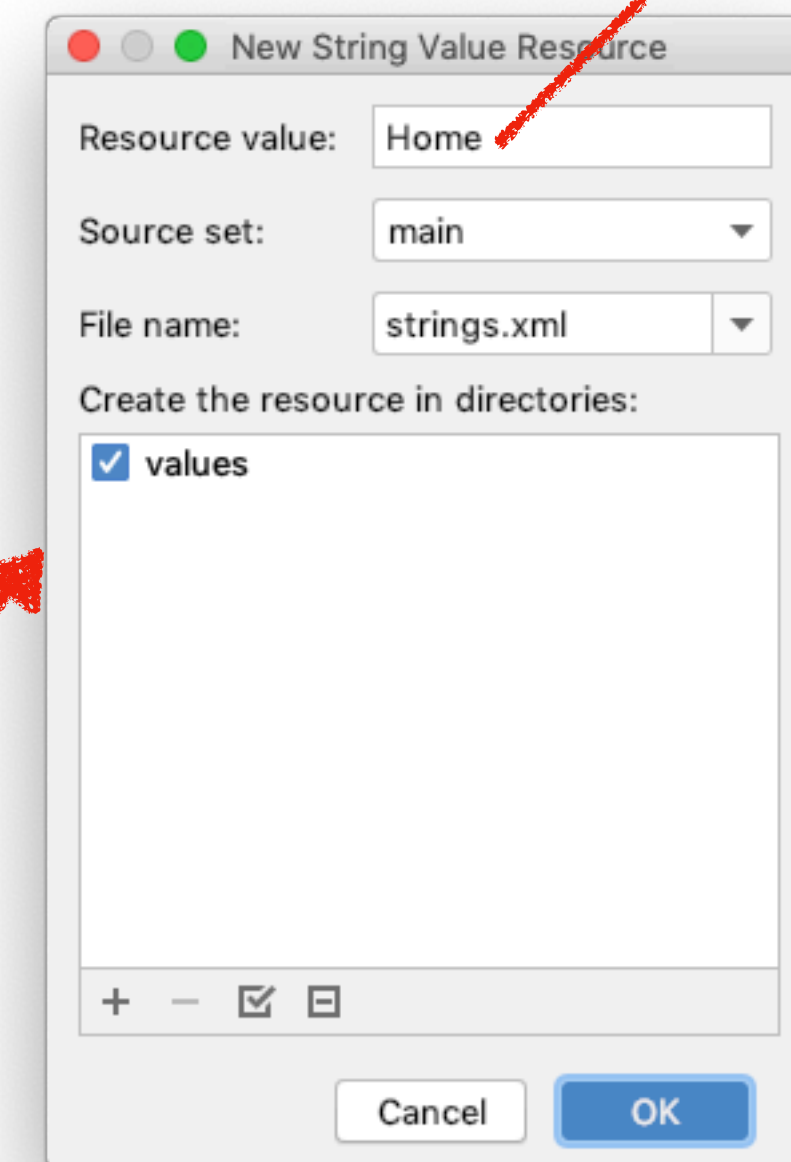
[Implementing an Android Navigation Drawer in Kotlin](#)

fragment_home.xml



```
<?xml version="1.0" encoding="utf-8"?>  
<LinearLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent"  
    android:gravity="center">
```

```
<TextView  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:gravity="center"  
    android:textSize="55sp"  
    android:text="@string/home" />  
</LinearLayout>
```

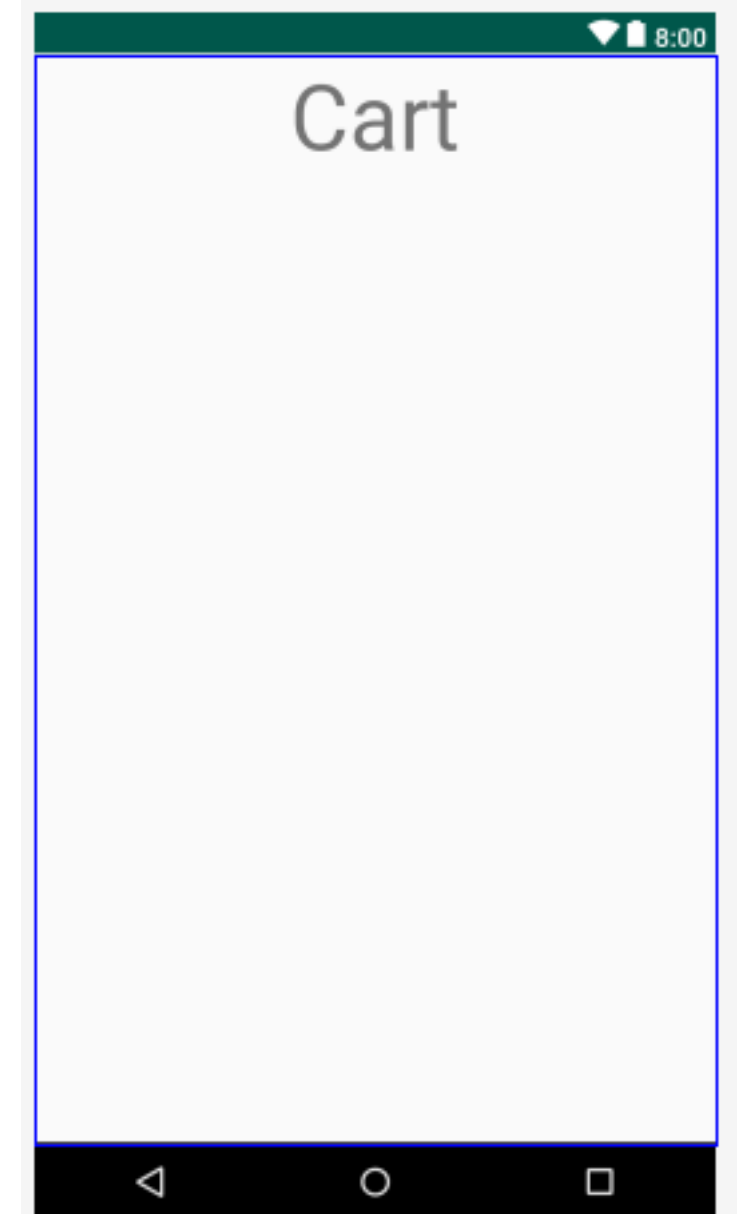


Show Intention Actions via `⌘↵` (Alt+Enter for Win/Linux)

fragment_cart.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="@string/cart"
        android:textSize="55sp"/>
</LinearLayout>
```



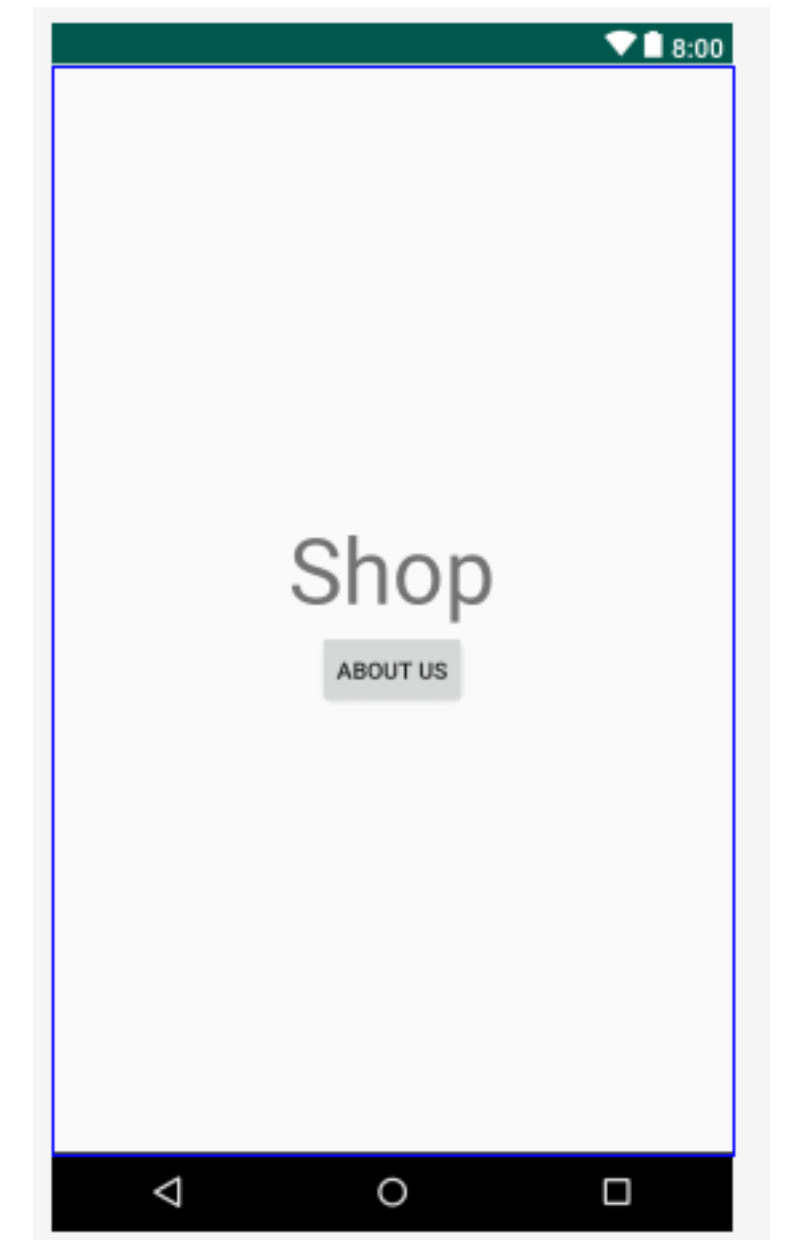
fragment_shop.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="vertical"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center">

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Shop"
        android:textSize="55sp"/>

    <Button
        android:id="@+id/btn_about"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="About Us"/>

</LinearLayout>
```



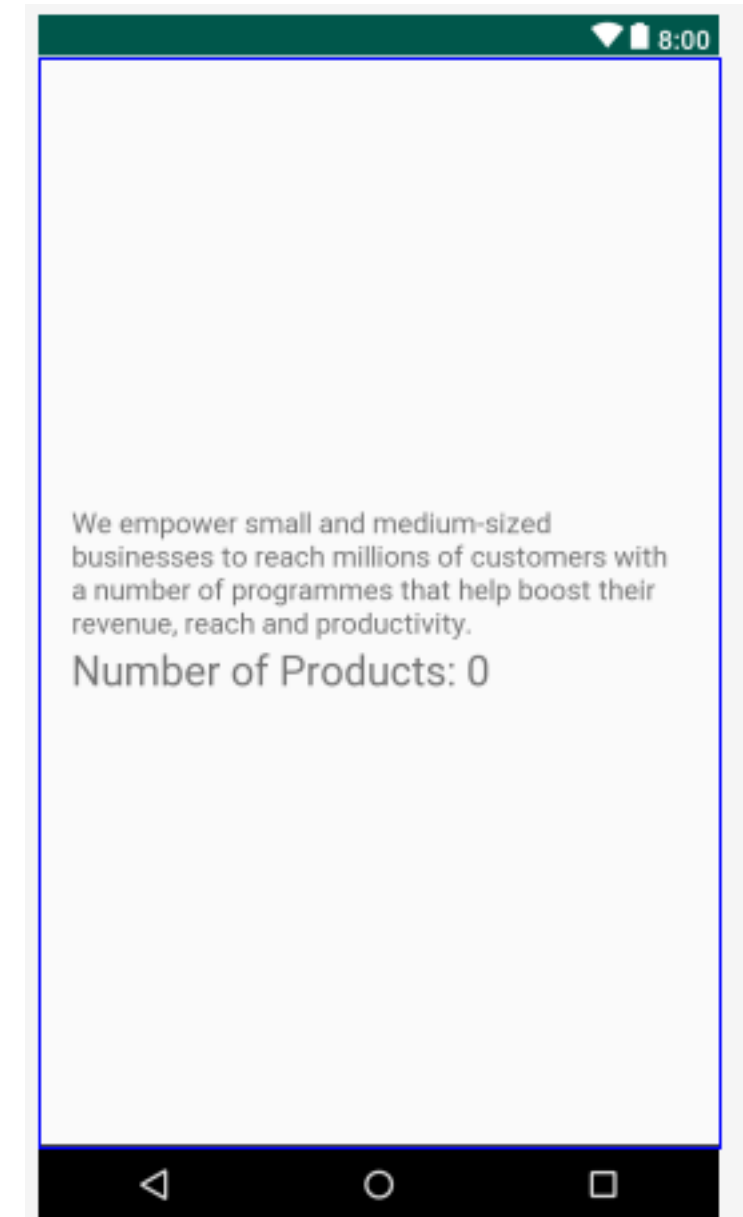
fragment_about.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="center"
    android:orientation="vertical"
    android:padding="20dp">

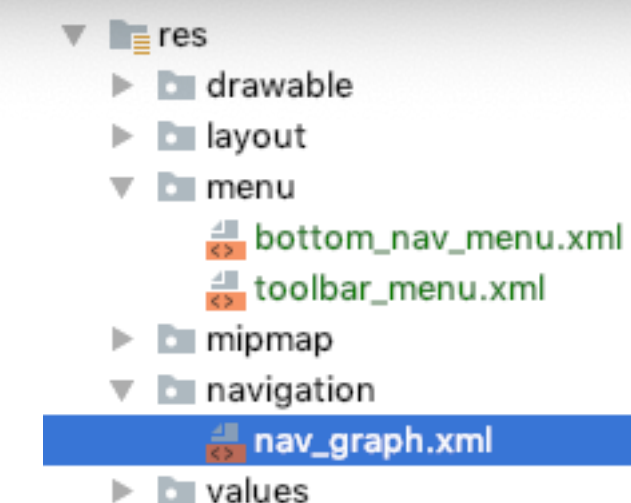
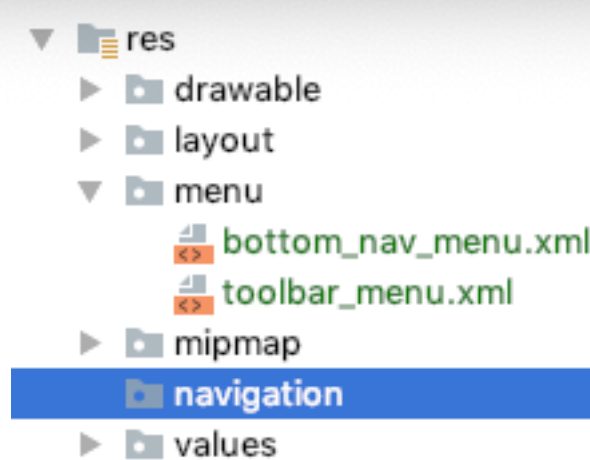
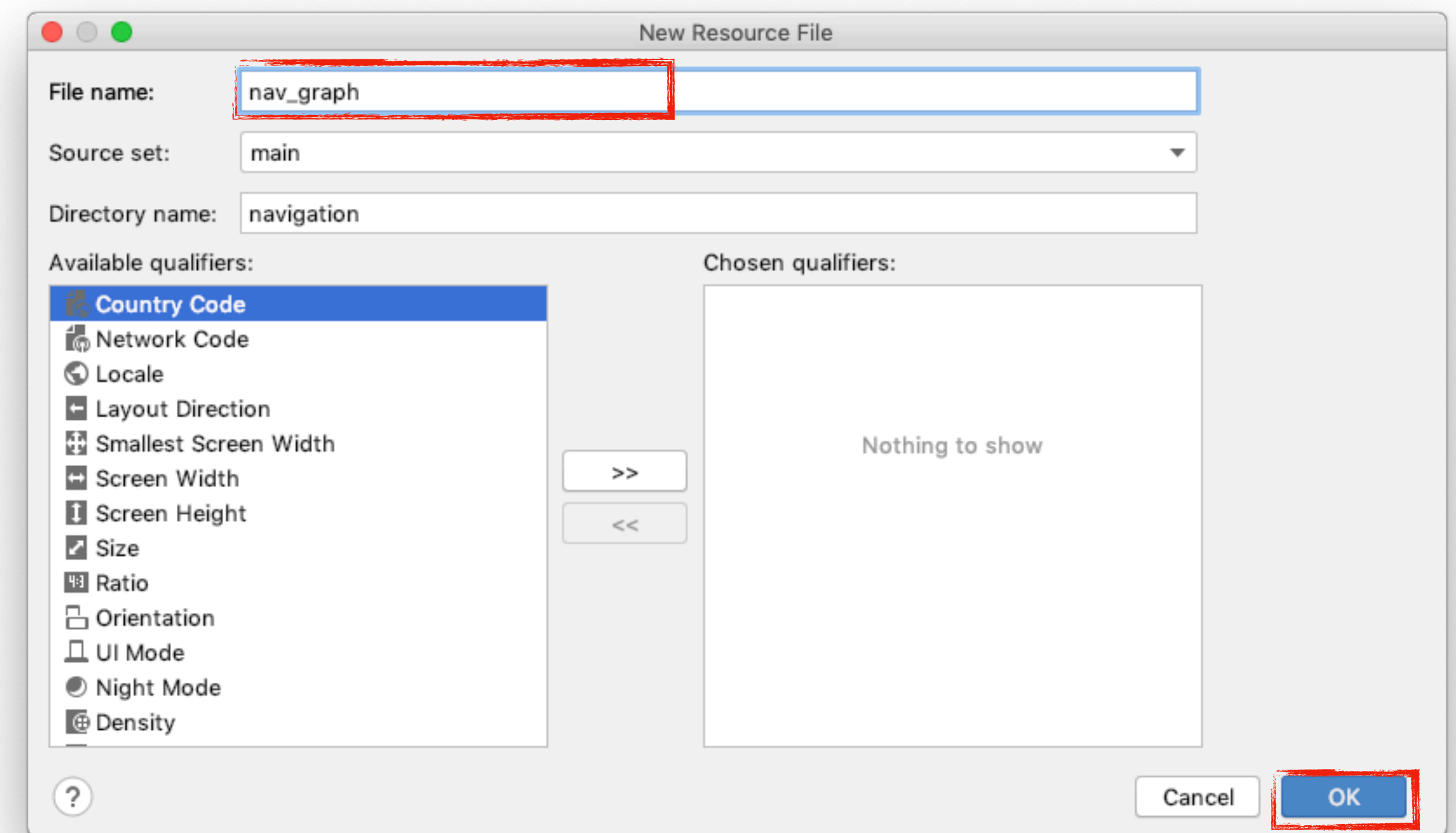
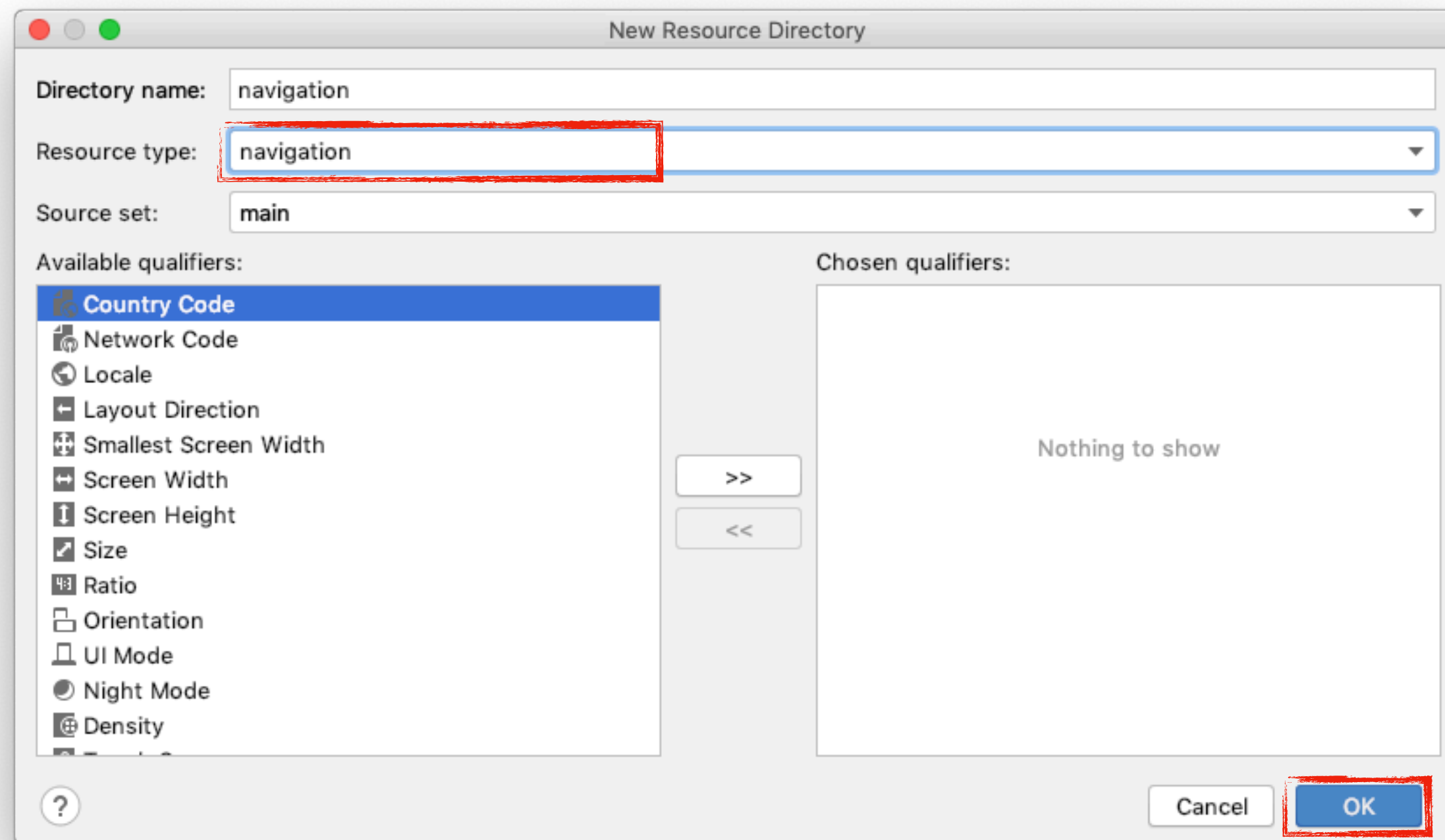
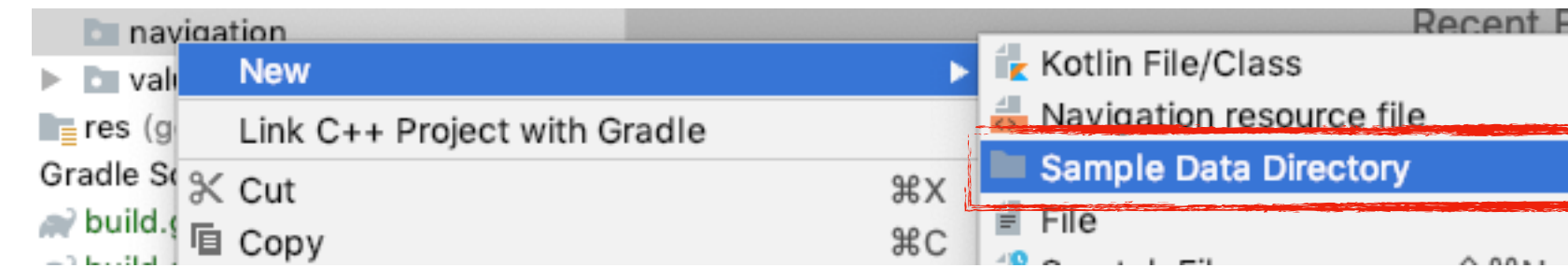
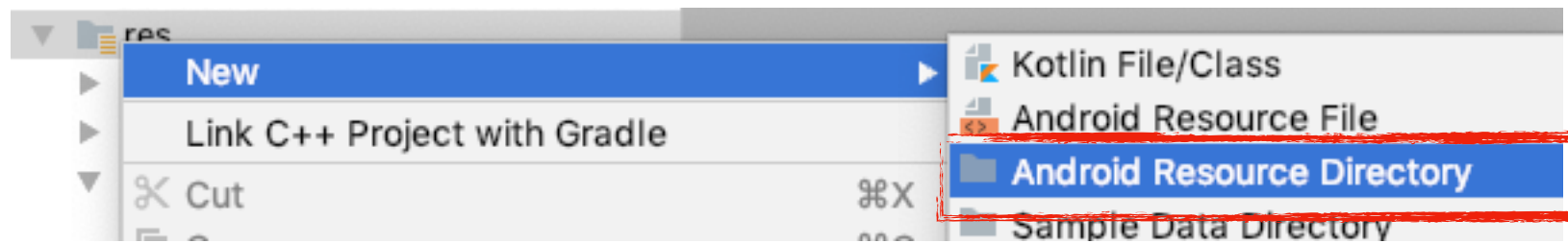
    <TextView
        android:id="@+id/tv_details"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="start"
        android:text="@string/detail"
        android:textSize="17sp"/>

    <TextView
        android:id="@+id/tv_product_count"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="start"
        android:text="Number of Products: 0"
        android:textSize="25sp"/>

</LinearLayout>
```



Erstellen des Navigations-Files



AboutFragment.kt

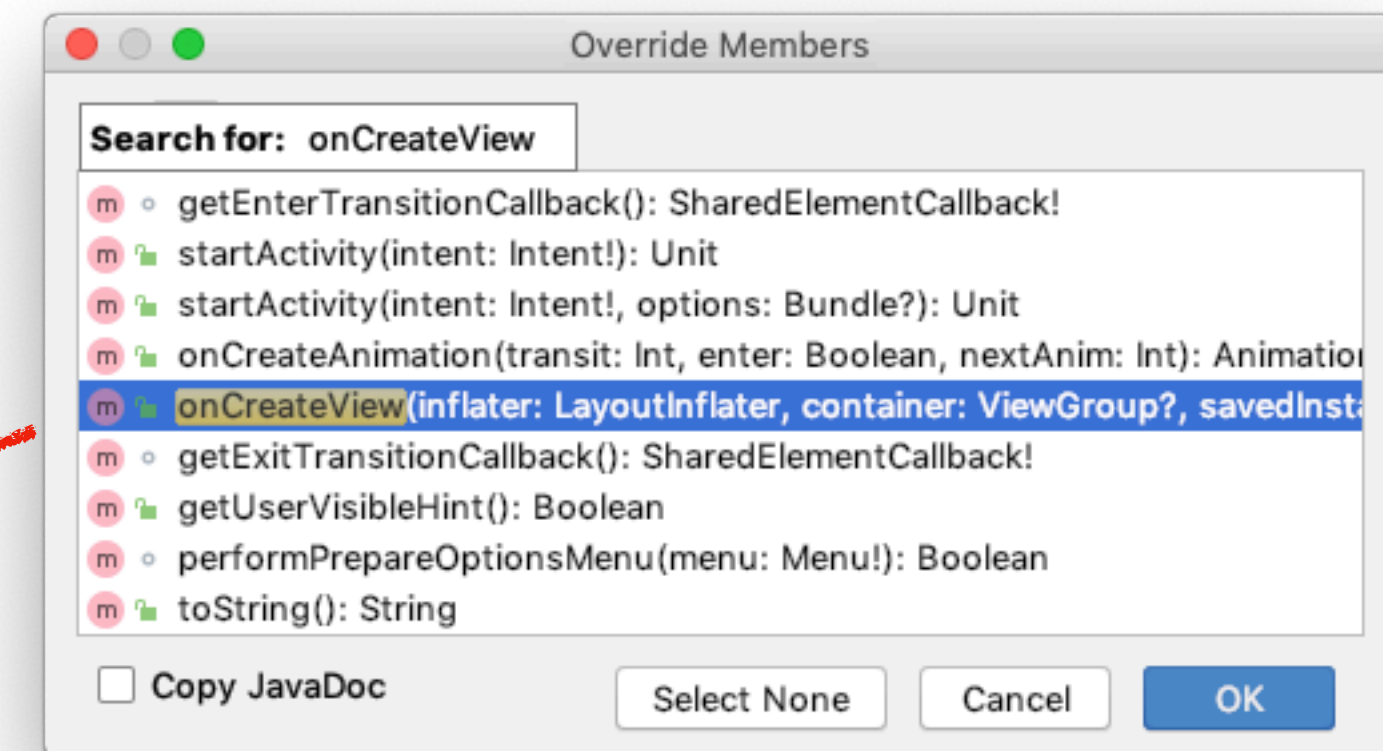


```
package at.htl.shop
```

```
import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.fragment.app.Fragment
```

```
class AboutFragment : Fragment() {
```

```
    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_about, container, false)
    }
}
```



CartFragment.kt



```
package at.htl.shop

import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.fragment.app.Fragment

class CartFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_cart, container, false)
    }
}
```

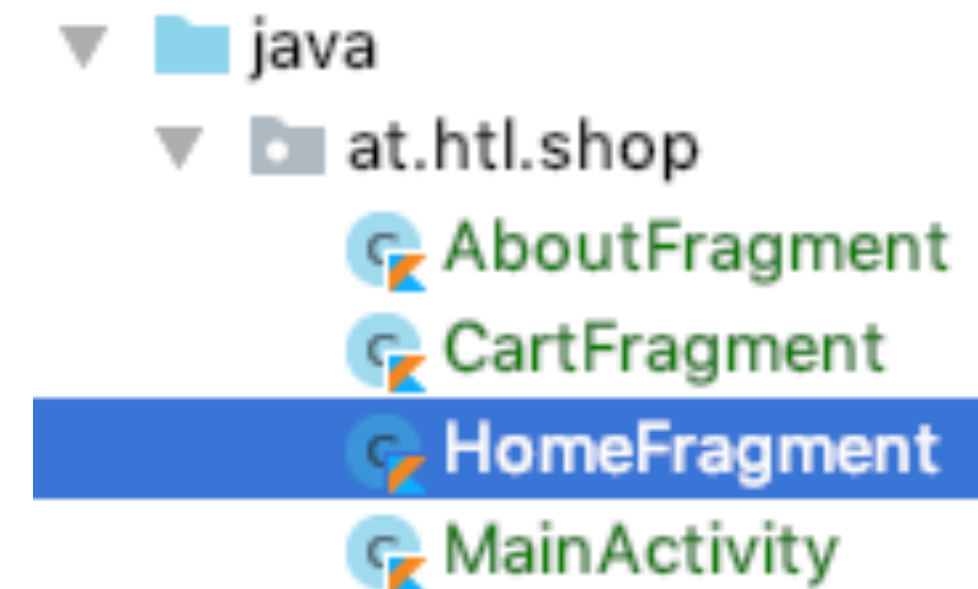
HomeFragment.kt

```
package at.htl.shop

import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.fragment.app.Fragment

class HomeFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_home, container, false)
    }
}
```



ShopFragment.kt

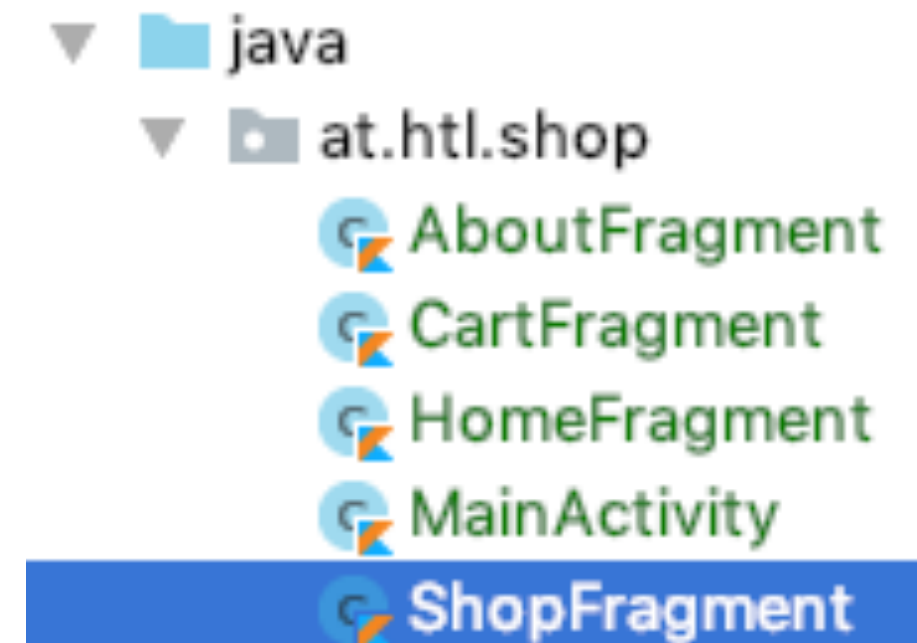
```
package at.htl.shop
```

```
import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.fragment.app.Fragment
```

```
class ShopFragment : Fragment() {
```

```
    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_shop, container, false)
    }
}
```

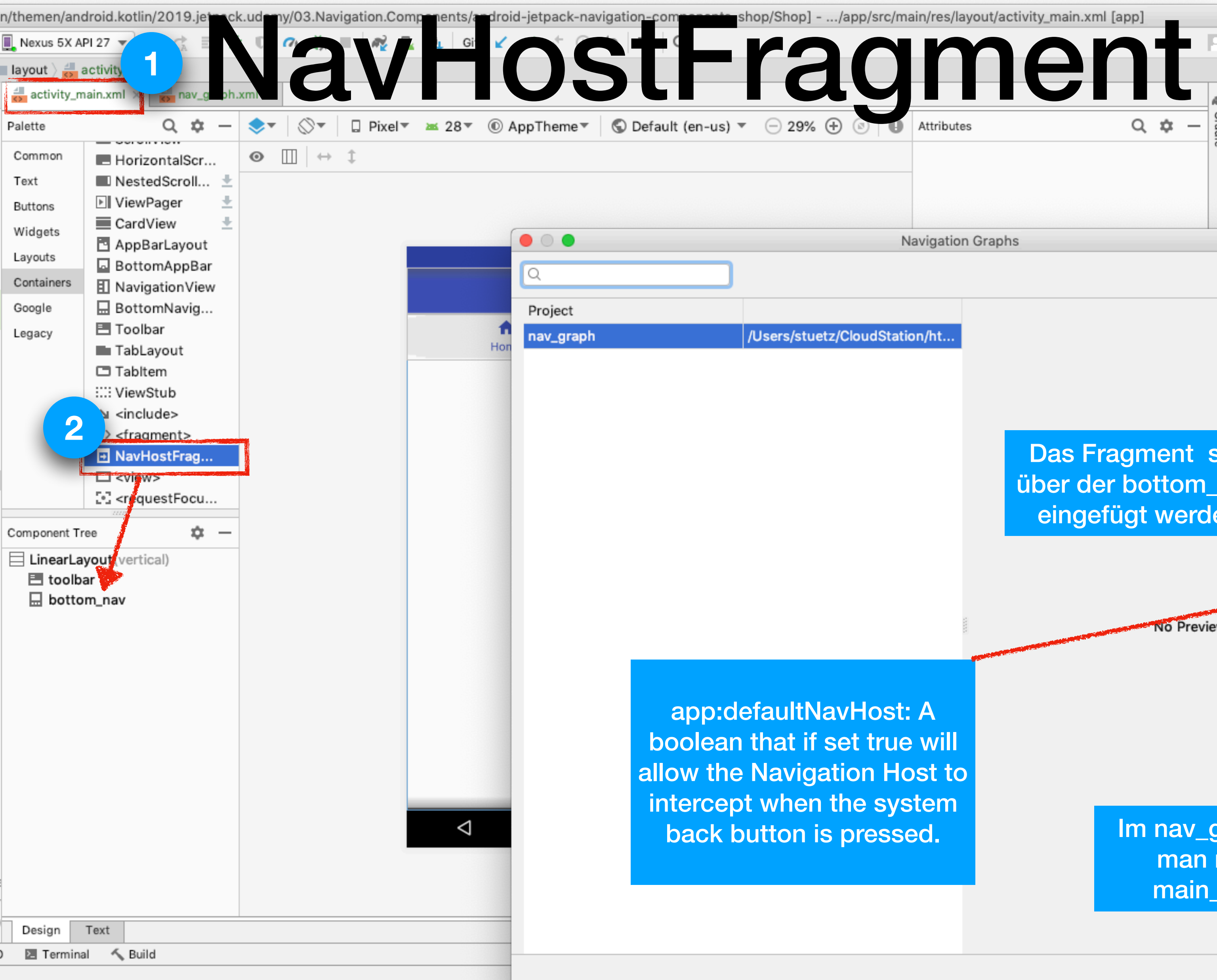
```
}
```



Navigation Components

- Navigation graph – contains all navigated information in centralized location
- NavHostFragment – special widget to add in layout
- NavController – Kotlin or Java object that keeps track of current position

1 NavHostFragment hinzufügen



2

1

4 Das Fragment soll über der bottom_nav eingefügt werden

app:defaultNavHost: A boolean that if set true will allow the Navigation Host to intercept when the system back button is pressed.

5 Im nav_graph sieht man nun die main_activity

```
<?xml version="1.0" encoding="utf-8"?>
<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="match_parent"
    android:orientation="vertical">

    <androidx.appcompat.widget.Toolbar
        android:id="@+id/toolbar"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:background="@color/colorPrimary"
        android:theme="@style/ThemeOverlay.AppCompat.Dark"/>

    <fragment
        android:id="@+id/nav_host_fragment"
        android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        app:defaultNavHost="true"
        app:navGraph="@navigation/nav_graph"/>

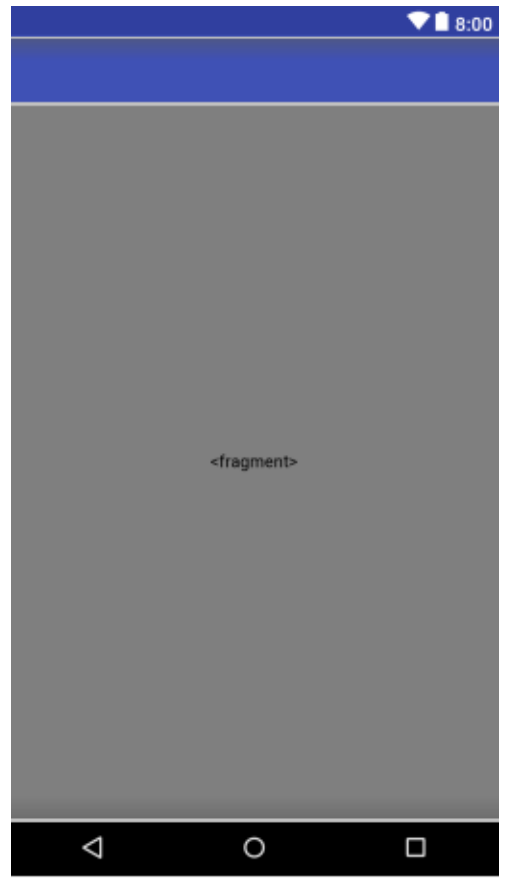
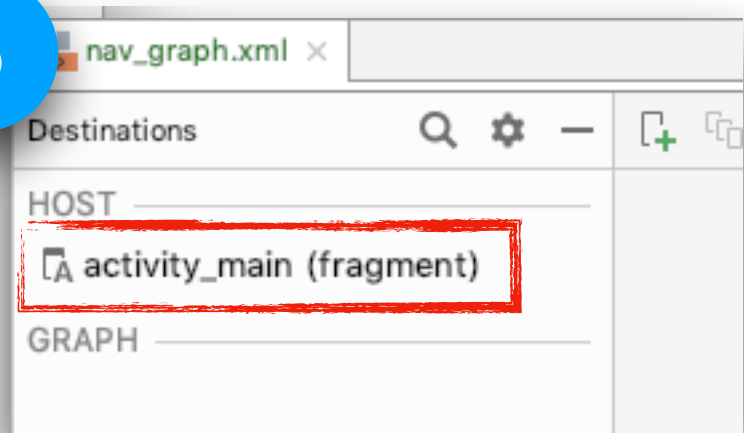
    <com.google.android.material.bottomnavigation.BottomNavigationView
        android:id="@+id/bottom_nav"
        app:menu="@menu/bottom_nav_menu"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"/>

</LinearLayout>
```

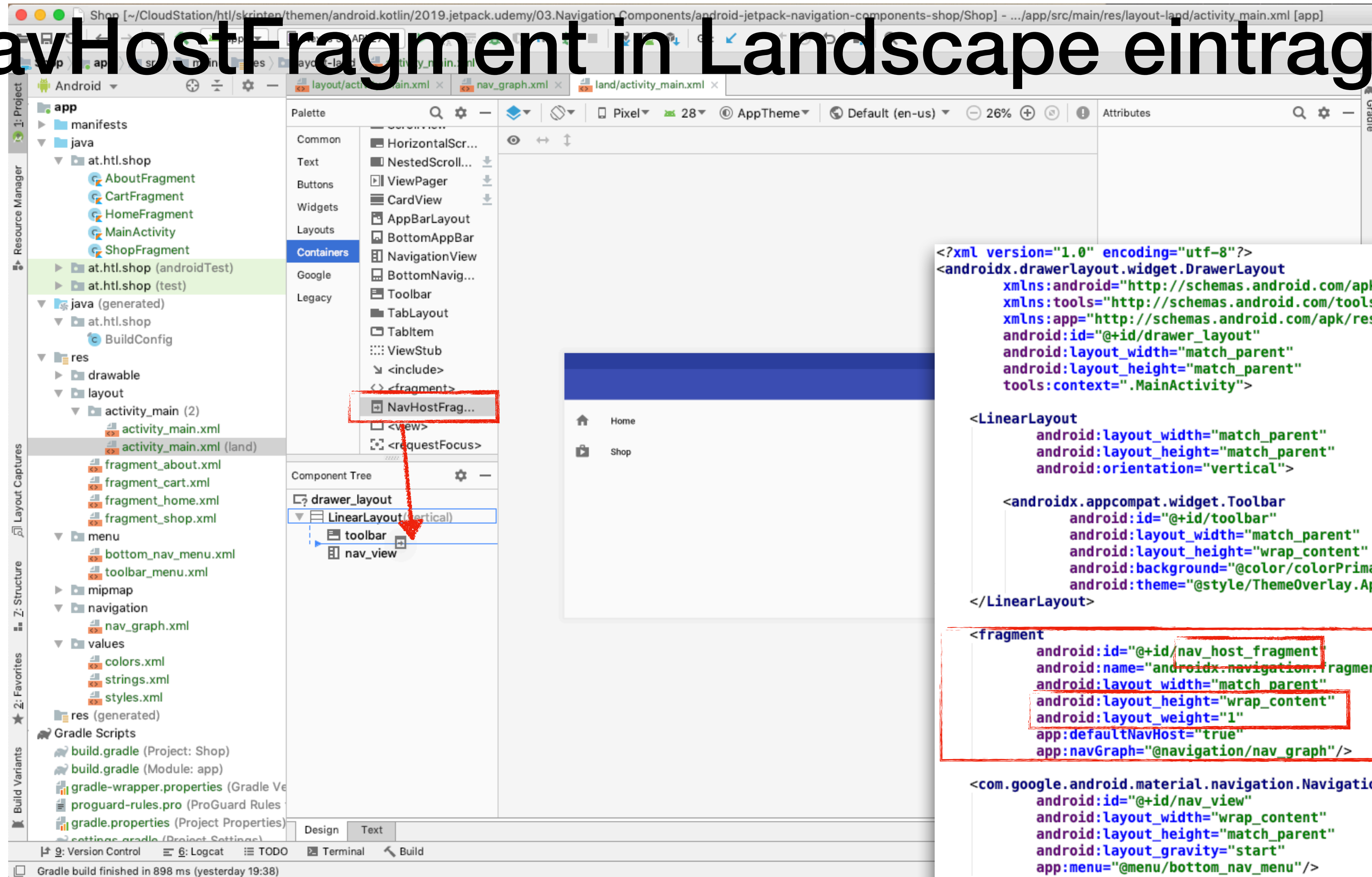
4

5

3



NavHostFragment in Landscape eintragen



```
<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

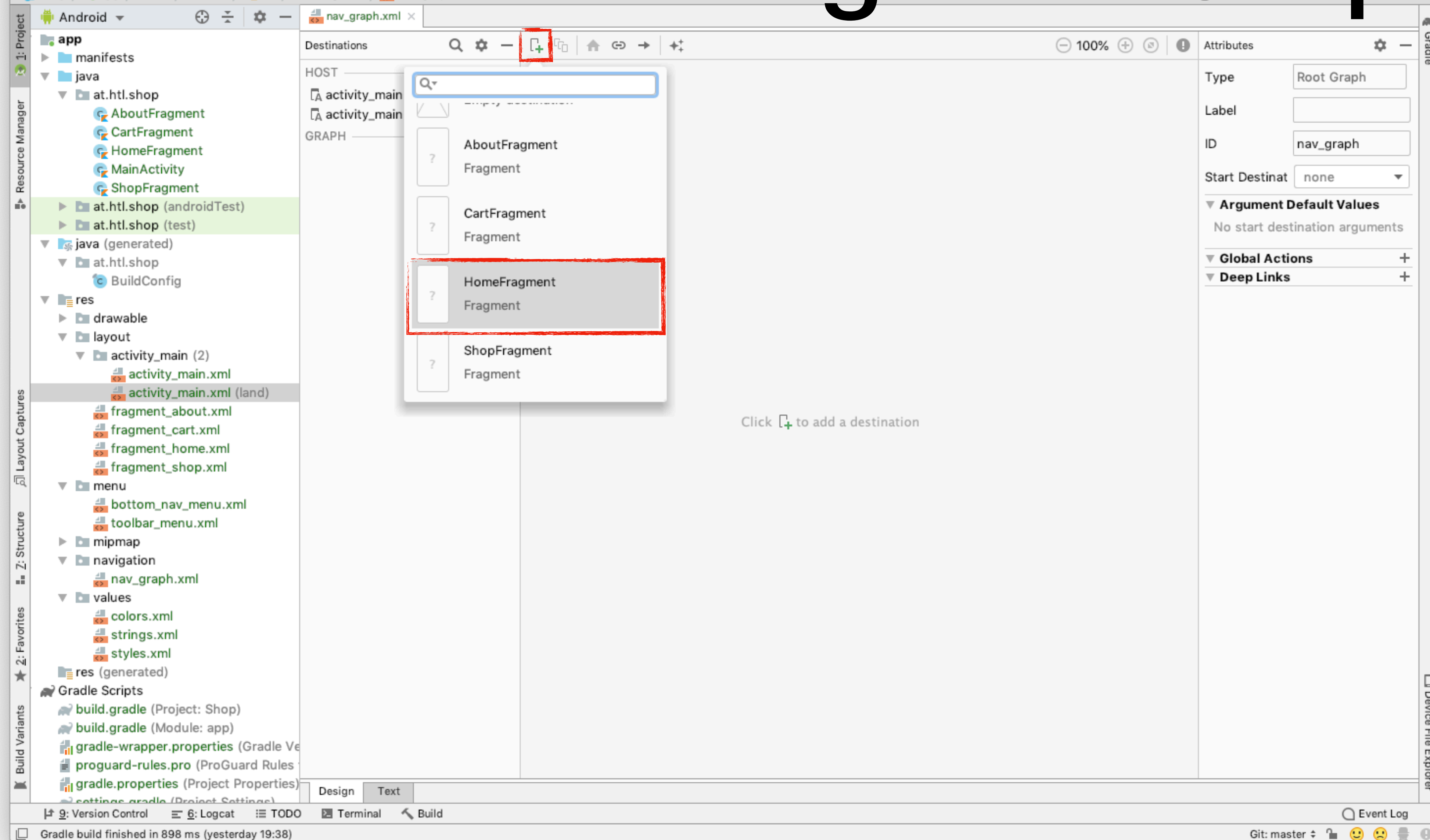
    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

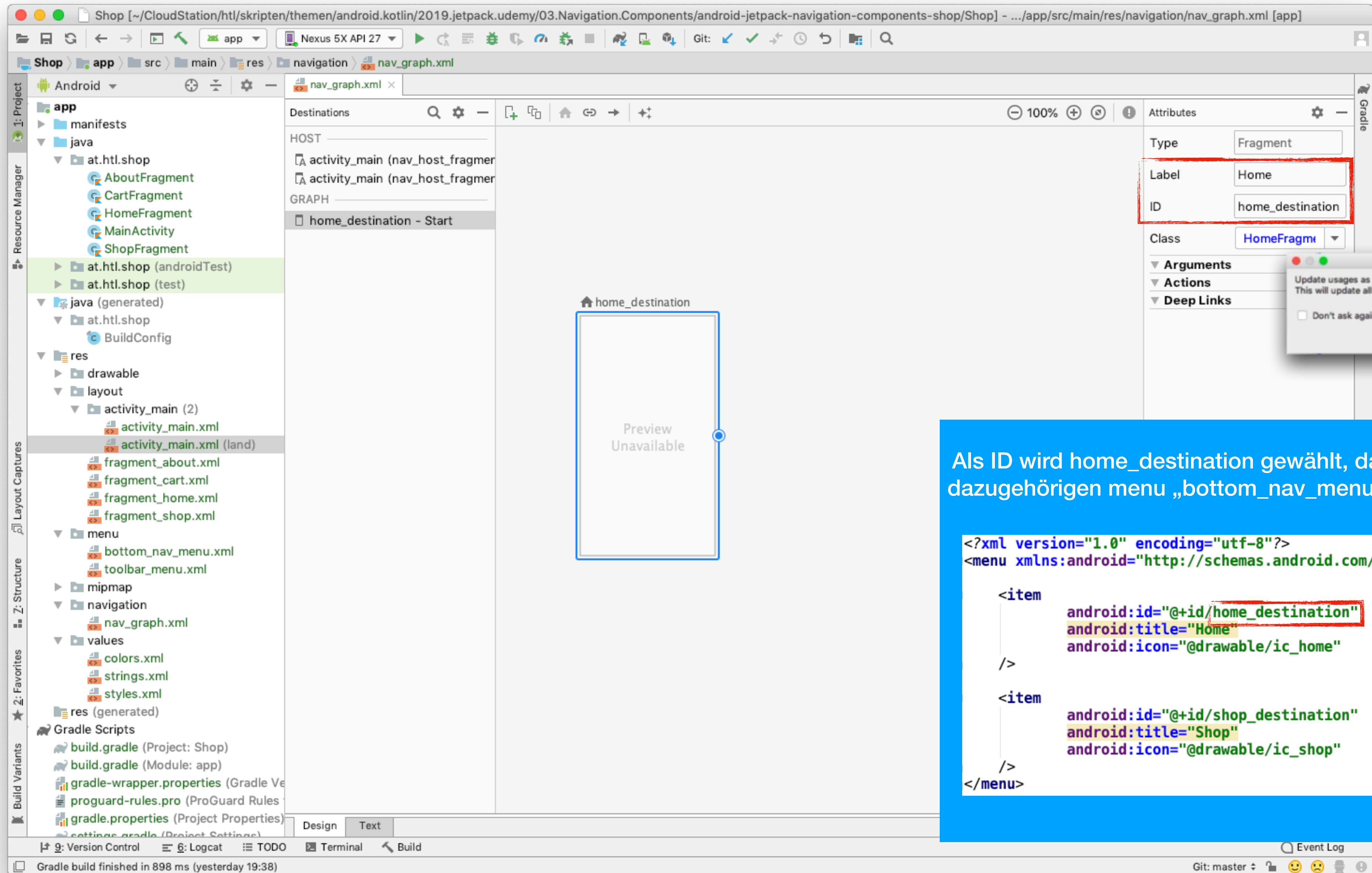
        <androidx.appcompat.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:background="@color/colorPrimary"
            android:theme="@style/ThemeOverlay.AppCompat.Dark"/>
    </LinearLayout>

    <fragment
        android:id="@+id/nav_host_fragment"
        android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        app:defaultNavHost="true"
        app:navGraph="@navigation/nav_graph"/>

    <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"
        app:menu="@menu/bottom_nav_menu"/>
</androidx.drawerlayout.widget.DrawerLayout>
```

Erstellen des Navigation Graphs





Als ID wird home_destination gewählt, da dies bereits im dazugehörigen menu „bottom_nav_menu“ eingetragen ist

Shop [~/CloudStation/html/skripten/themen/android.kotlin/2019.jetpack.udemy/03.Navigation.Components/android-jetpack-navigation-components-shop/Shop] - .../app/src/main/res/navigation/nav_graph.xml [app]

Shop > app > src > main > res > navigation > nav_graph.xml

1: Project

Resource Manager

Destinations

HOST

- activity_main (nav_host_fragment)
- activity_main (nav_host_fragment)

GRAPH

- home_destination - Start
- shop_destination
- about_destination
- cart_destination

home_destination

shop_destination

about_destination

cart_destination

Preview Unavailable

Preview Unavailable

Preview Unavailable

Preview Unavailable

Attributes

Type: Fragment

Label: CartFragment

ID: cart_destination

Class: (at.htl.shop)

Arguments: +

Actions: +

Deep Links: +

Design Text

Version Control Logcat TODO Terminal Build

Gradle build finished in 898 ms (yesterday 19:38)

```

<?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"
    app:startDestination="@id/home_destination">

    <fragment
        android:id="@+id/home_destination"
        android:name="at.htl.shop.HomeFragment"
        android:label="Home"/>
    <fragment
        android:id="@+id/shop_destination"
        android:name="at.htl.shop.ShopFragment"
        android:label="Shop"/>
    <fragment
        android:id="@+id/about_destination"
        android:name="at.htl.shop.AboutFragment"
        android:label="About"/>
    <fragment
        android:id="@+id/cart_destination"
        android:name="at.htl.shop.CartFragment"
        android:label="Cart"/>
</navigation>

```


MainActivity.kt 1

In der MainActivity werden folgende Komponenten konfiguriert:
BottomNavigationBar
Toolbar
NavigationDrawer

```
class MainActivity : AppCompatActivity() {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
        setContentView(R.layout.activity_main)  
  
        val navController =  
            Navigation.findNavController(this, R.id.nav_host_fragment)  
  
        setUpBottomNav(navController)  
        setUpSideNav(navController)  
        setUpActionBar(navController)  
    }  
}
```

Lassen Sie sich die Methoden generieren

```
private fun setUpBottomNav(navController: NavController) {  
}  
  
private fun setUpSideNav(navController: NavController) {  
}  
  
private fun setUpActionBar(navController: NavController) {  
}  
}
```


MainActivity.kt 2

activity_main.xml

```
<com.google.android.material.bottomnavigation.BottomNavigationView  
    android:id="@+id/bottom_nav"  
    app:menu="@menu/bottom_nav_menu"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"/>
```

Der safe-call-Operator '?' wird verwendet, da das Landscape-Layout kein bottom_nav enthält
<https://kotlinlang.org/docs/reference/null-safety.html#safe-calls>

NavigationUI

Class which hooks up elements of your application such as global navigation patterns like a navigation drawer or bottom nav bar with your NavController.

```
private fun setUpBottomNav(navController: NavController) {  
    bottom_nav?.let {  
        NavigationUI.setupWithNavController(it, navController)  
    }  
}
```

```
private fun setUpSideNav(navController: NavController) {  
    nav_view?.let {  
        NavigationUI.setupWithNavController(it, navController)  
    }  
}
```

```
private fun setUpActionBar(navController: NavController) {  
    NavigationUI.setupActionBarWithNavController(  
        this,  
        navController,  
        drawer_layout  
    )  
}
```

let { ... }

2. This vs. it argument

If we look at `T.run` and `T.let`, both functions are similar except for one thing, the way they accept the argument. The below shows the same logic for both functions.

```
stringVariable?.run {
    println("The length of this String is $length")
}

// Similarly.

stringVariable?.let {
    println("The length of this String is ${it.length}")
}
```

If you check the `T.run` function signature, you'll notice the `T.run` is just made as extension function calling `block: T.()`. Hence all within the scope, the `T` could be referred as `this`. In programming, `this` could be omitted most of the time. Therefore in our example above, we could use `$length` in the `println` statement, instead of `${this.length}`. I call this as sending in *this as argument*.

However for `T.let` function signature, you'll notice that `T.let` is sending itself into the function i.e. `block: (T)`. Hence this is like a lambda argument sent it. It could be referred within the scope function as `it`. So I call this as sending in *it as argument*.

MainActivity.kt 3

Erstellen des Option-Menus

```
override fun onCreateOptionsMenu(menu: Menu?): Boolean {  
    menuInflater.inflate(R.menu.toolbar_menu, menu)  
    return true  
}
```

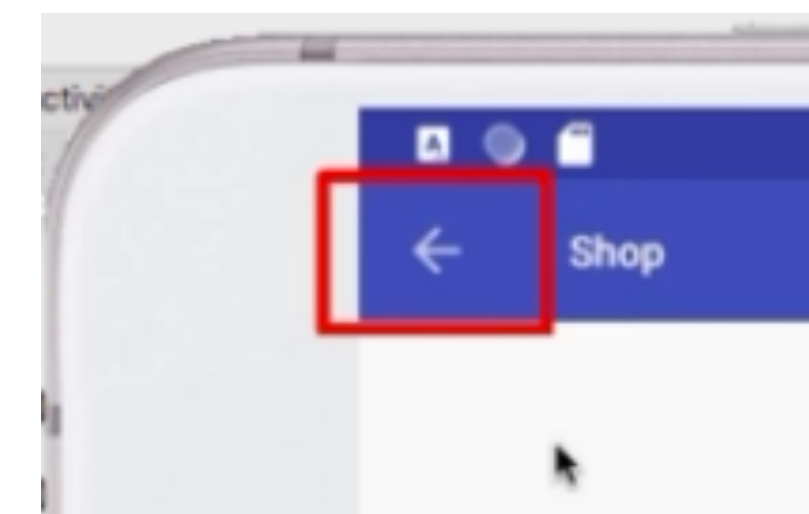
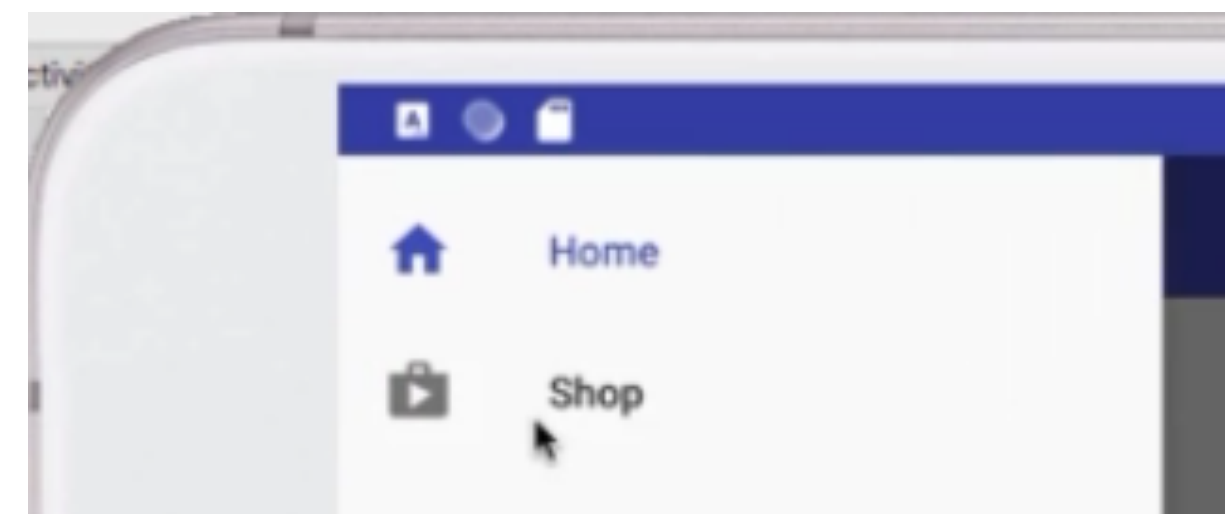
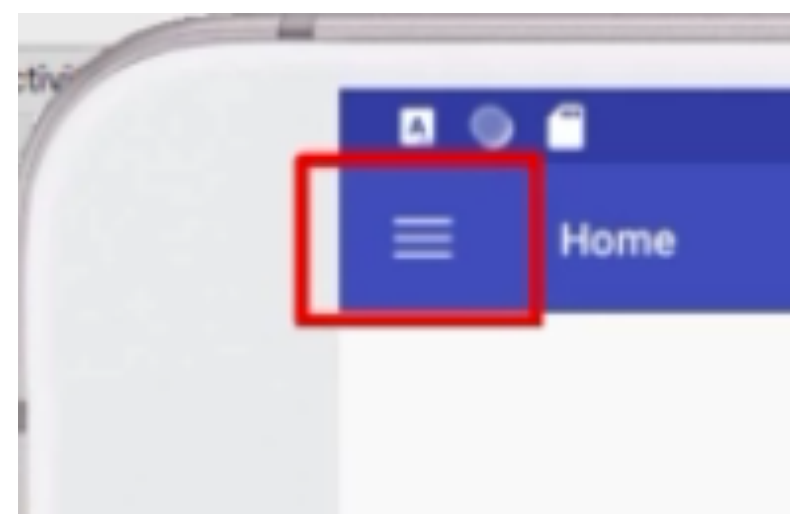
```
override fun onOptionsItemSelected(item: MenuItem?): Boolean {  
    val navController = Navigation.findNavController(this, R.id.nav_host_fragment)  
    val navigated = NavigationUI.onNavDestinationSelected(item!!, navController)  
    return navigated || super.onOptionsItemSelected(item)  
}
```

```
override fun onSupportNavigateUp(): Boolean {  
    return NavigationUI.navigateUp(  
        Navigation.findNavController(this, R.id.nav_host_fragment), drawer_layout  
    )  
}
```

„item is not null“
Es wird ein NPE geworfen, falls item
doch null ist

<https://kotlinlang.org/docs/reference/null-safety.html#the-null-operator>

navigateUp sorgt dafür, dass
der Hamburger durch einen
Pfeil (icon 'up') ersetzt wird,
wenn zu einem anderen
Fragment gewechselt wird



MainActivity.kt 4

```
package at.htl.shop
```

```
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.view.Menu
import android.view.MenuItem
import androidx.navigation.NavController
import androidx.navigation.Navigation
import androidx.navigation.ui.NavigationUI
import kotlinx.android.synthetic.main.activity_main.*
```

```
class MainActivity : AppCompatActivity() {

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

        val navController =
            Navigation.findNavController(this, R.id.nav_host_fragment)

        setUpBottomNav(navController)
        setUpSideNav(navController)
        setUpActionBar(navController)
    }
}
```

```
private fun setUpBottomNav(navController: NavController) {
    bottom_nav?.let {
        NavigationUI.setupWithNavController(it, navController)
    }
}
```

```
private fun setUpSideNav(navController: NavController) {
    nav_view?.let {
        NavigationUI.setupWithNavController(it, navController)
    }
}
```

```
private fun setUpActionBar(navController: NavController) {
    NavigationUI.setupActionBarWithNavController(
        this,
        navController,
        drawer_layout
    )
}
```

```
override fun onCreateOptionsMenu(menu: Menu?): Boolean {
    menuInflater.inflate(R.menu.toolbar_menu, menu)
    return true
}
```

```
override fun onOptionsItemSelected(item: MenuItem?): Boolean {
    val navController = Navigation.findNavController(this, R.id.nav_host_fragment)
    val navigated = NavigationUI.onNavDestinationSelected(item!!, navController)
    return navigated || super.onOptionsItemSelected(item)
}
```

```
override fun onSupportNavigateUp(): Boolean {
    return NavigationUI.navigateUp(
        Navigation.findNavController(this, R.id.nav_host_fragment), drawer_layout
    )
}
```


Überblick

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

    val navController =
        Navigation.findNavController(this, R.id.nav_host_fragment)

    setUpBottomNav(navController)
    setUpSideNav(navController)
    setUpActionBar(navController)
}
```

In der MainActivity werden die Komponenten konfiguriert

```
private fun setUpBottomNav(navController: NavController) {
    bottom_nav?.let {
        NavigationUI.setupWithNavController(it, navController)
    }
}
```

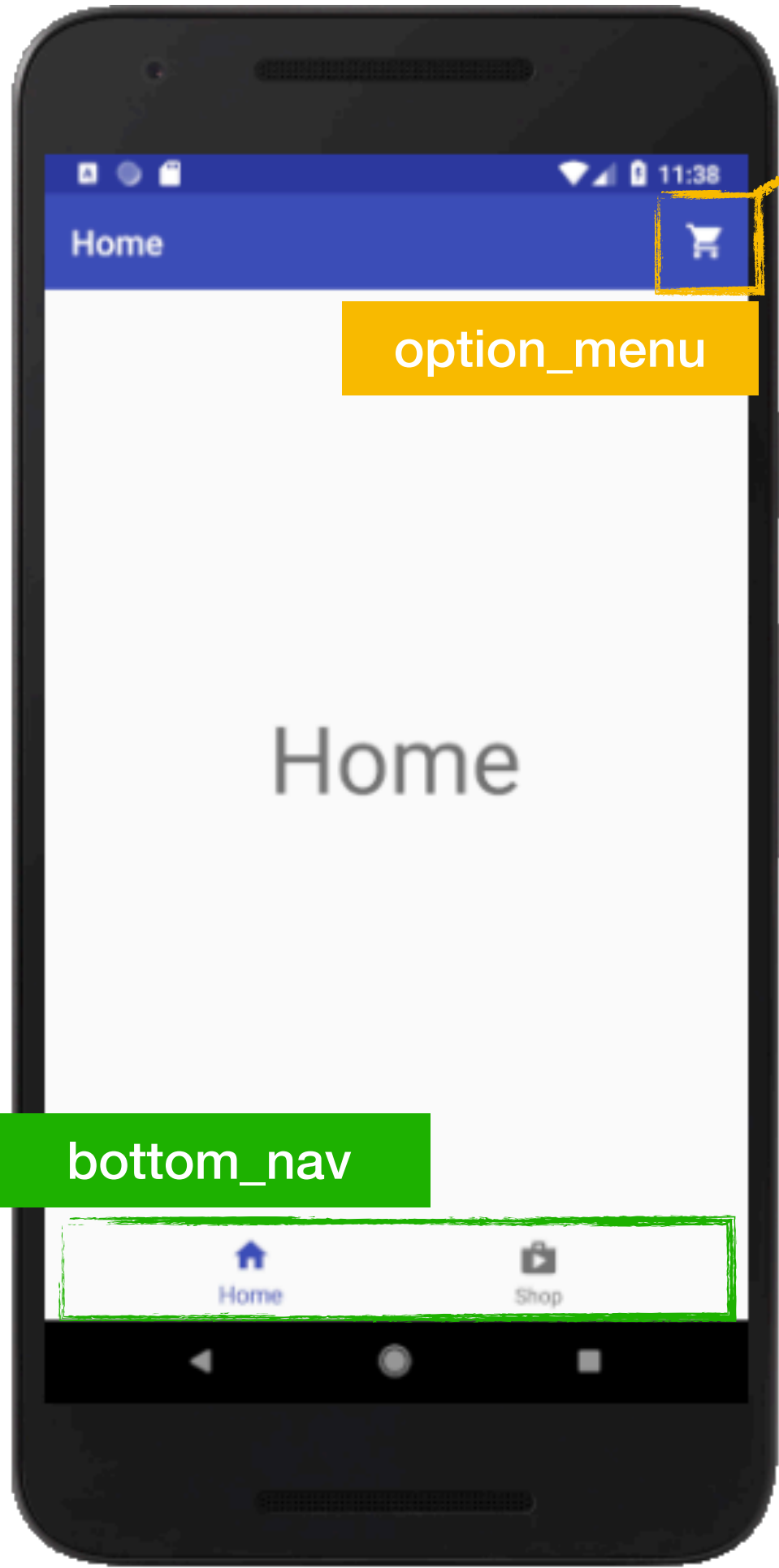
```
private fun setUpSideNav(navController: NavController) {
    nav_view?.let {
        NavigationUI.setupWithNavController(it, navController)
    }
}
```

```
private fun setUpActionBar(navController: NavController) {
    NavigationUI.setupActionBarWithNavController(
        this,
        navController,
        drawer_layout
    )
}
```

bezieht sich auf die Landscape activity_main.xml
android:id="@+id/drawer_layout"

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/home_destination"
        android:title="Home"
        android:icon="@drawable/ic_home"
    />
    <item
        android:id="@+id/shop_destination"
        android:title="Shop"
        android:icon="@drawable/ic_shop"
    />
</menu>
```

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/cart_destination"
        android:icon="@drawable/ic_shopping_cart"
        android:title="Cart"
        app:showAsAction="ifRoom"
    />
</menu>
```



```
<?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"
    app:startDestination="@id/home_destination">
    <fragment
        android:id="@+id/home_destination"
        android:name="at.htl.shop.HomeFragment"
        android:label="Home"/>
    <fragment
        android:id="@+id/shop_destination"
        android:name="at.htl.shop.ShopFragment"
        android:label="Shop"/>
    <fragment
        android:id="@+id/about_destination"
        android:name="at.htl.shop.AboutFragment"
        android:label="About"/>
    <fragment
        android:id="@+id/cart_destination"
        android:name="at.htl.shop.CartFragment"
        android:label="Cart"/>
</navigation>
```

```
class CartFragment : Fragment() {
    override fun onCreateView(...): View? {
        return inflater.inflate(R.layout.fragment_cart, container, false)
    }
}
```

Im CartFragment werden die Fragmente am Screen dargestellt (inflated)

onClickListener implementieren



```
package at.htl.shop

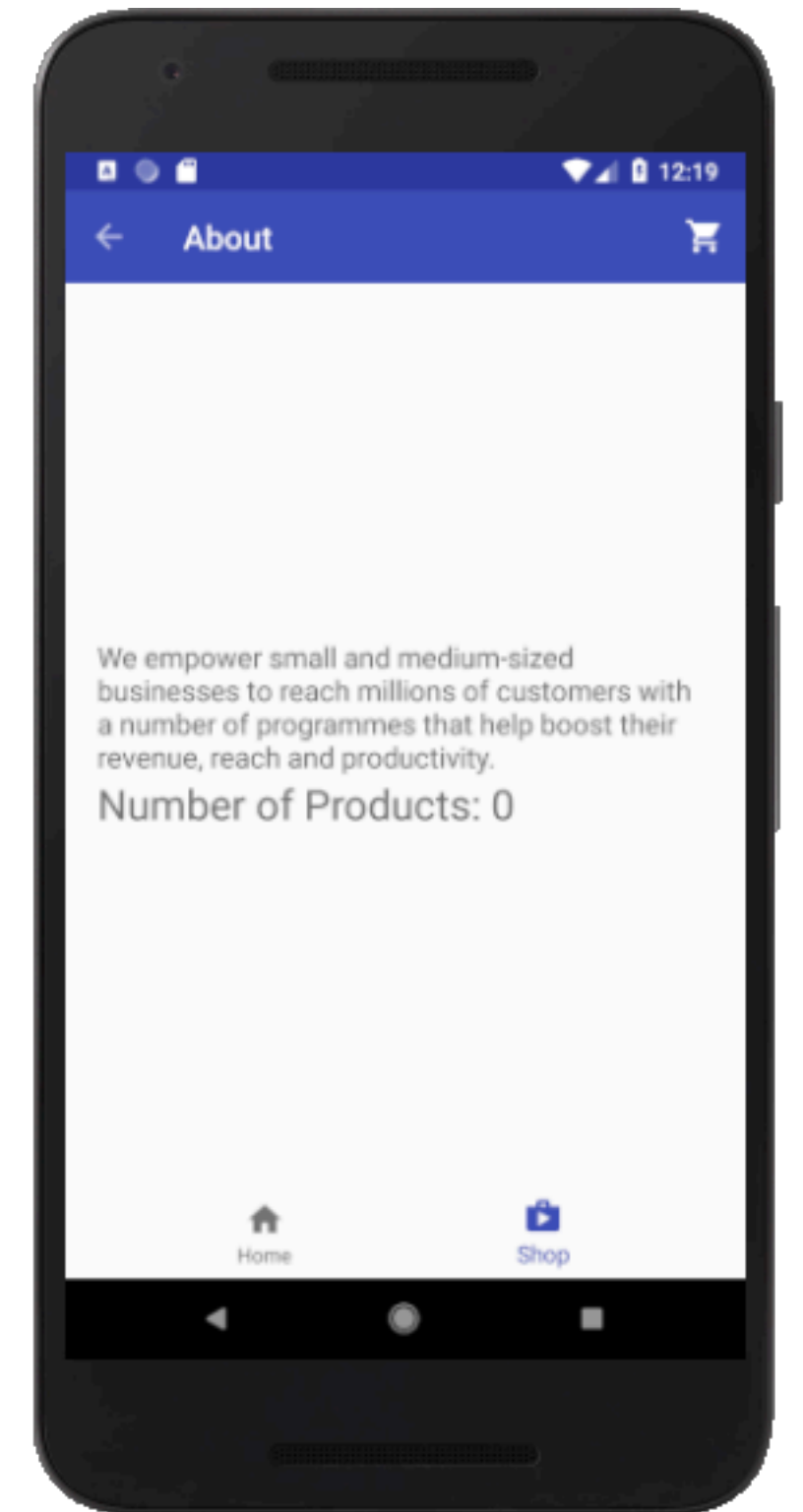
import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.fragment.app.Fragment
import androidx.navigation.Navigation
import kotlinx.android.synthetic.main.fragment_shop.*

class ShopFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_shop, container, false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)

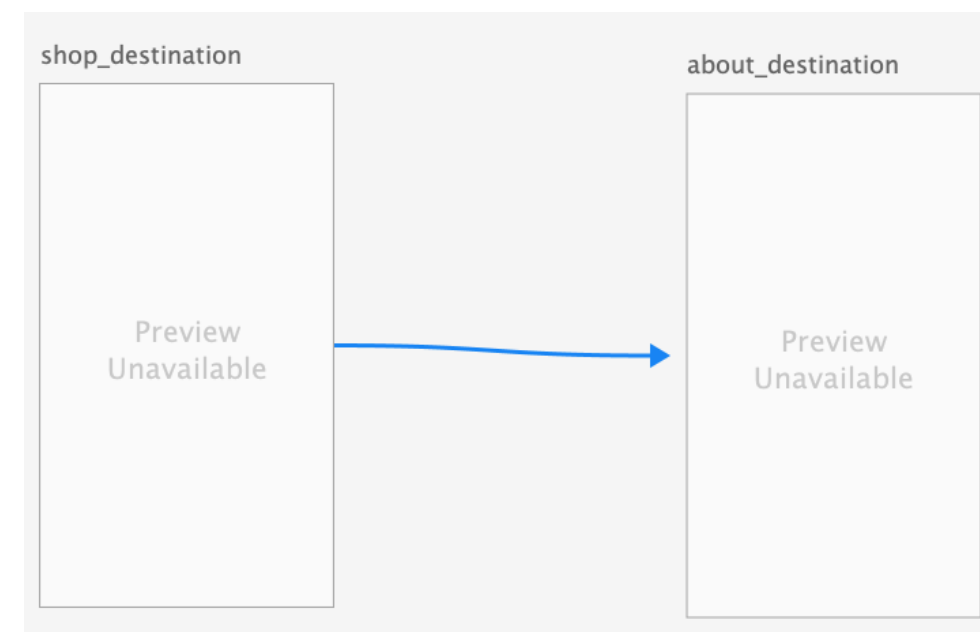
        btn_about.setOnClickListener {
            Navigation.findNavController(it).navigate(R.id.about_destination)
        }
    }
}
```



Um zum About Fragment zu gelangen, müssen wir den Button konfigurieren

Ergebnis

Navigation Actions



Setzen der ersten Action

The screenshot shows the Android Studio interface with the navigation graph editor open. The graph displays four destinations: home_destination, shop_destination, about_destination, and cart_destination. A blue arrow points from shop_destination to about_destination. The right-hand pane shows the 'Attributes' for the selected 'next_action' element, with the 'Destination' dropdown set to 'about_destination'. Below this, the XML code for the navigation graph is displayed, with the action element highlighted in red:

```
<?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"
    app:startDestination="@id/home_destination">

    <fragment
        android:id="@+id/home_destination"
        android:name="at.htl.shop.HomeFragment"
        android:label="Home"/>

    <fragment
        android:id="@+id/shop_destination"
        android:name="at.htl.shop.ShopFragment"
        android:label="Shop">
        <action
            android:id="@+id/next_action"
            app:destination="@id/about_destination"/>
    </fragment>

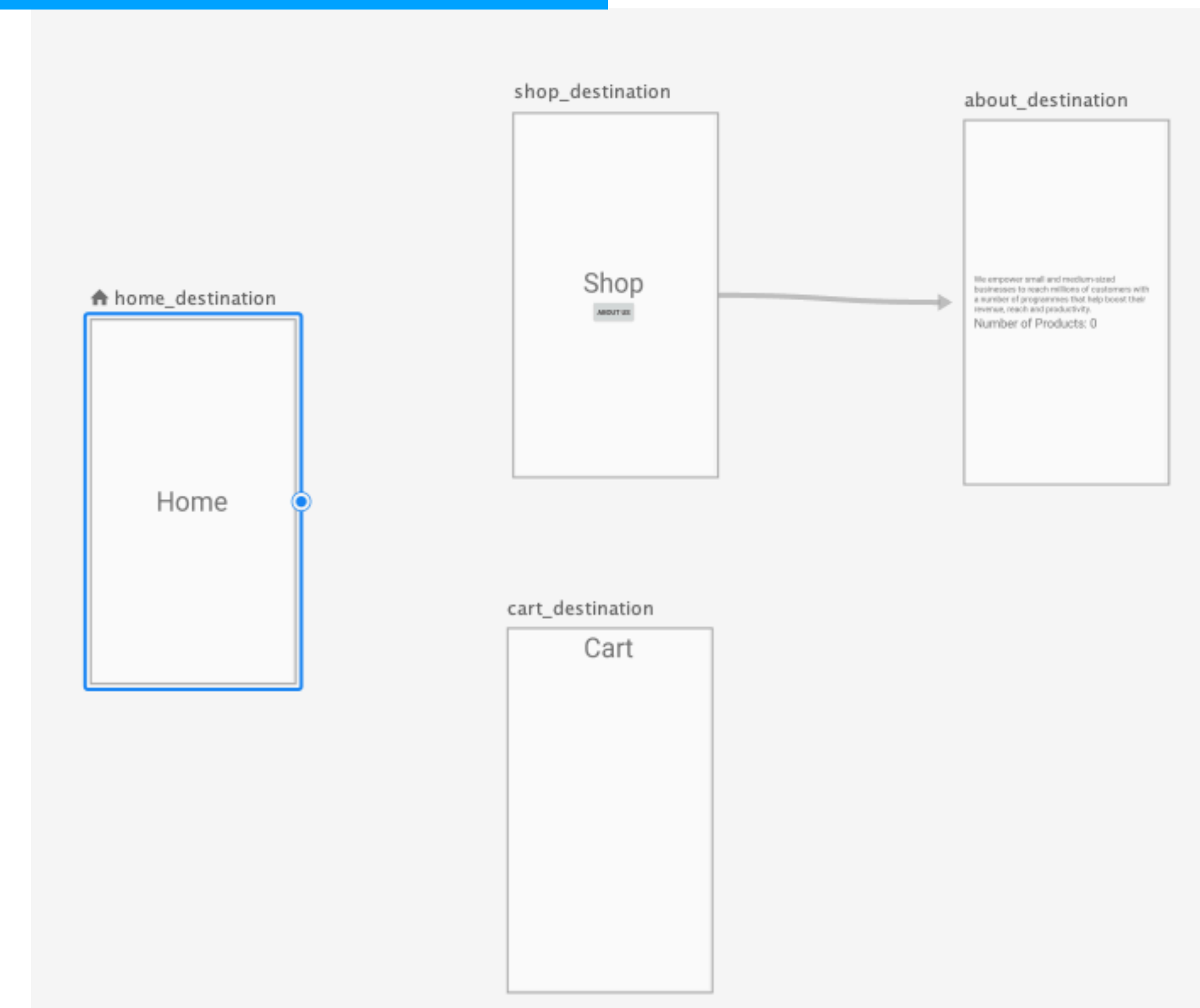
    <fragment
        android:id="@+id/about_destination"
        android:name="at.htl.shop.AboutFragment"
        android:label="About"/>

    <fragment
        android:id="@+id/cart_destination"
        android:name="at.htl.shop.CartFragment"
        android:label="Cart"/>
</navigation>
```

nav_graph.xml: Layout eintragen

```
<?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_graph"
  app:startDestination="@id/home_destination">
  <fragment
    android:id="@+id/home_destination"
    android:name="at.htl.shop.HomeFragment"
    android:label="Home"
    tools:layout="@layout/fragment_home"/>
  <fragment
    android:id="@+id/shop_destination"
    android:name="at.htl.shop.ShopFragment"
    android:label="Shop"
    tools:layout="@layout/fragment_shop">
    <action
      android:id="@+id/next_action"
      app:destination="@id/about_destination"/>
  </fragment>
  <fragment
    android:id="@+id/about_destination"
    android:name="at.htl.shop.AboutFragment"
    android:label="About"
    tools:layout="@layout/fragment_about"/>
  <fragment
    android:id="@+id/cart_destination"
    android:name="at.htl.shop.CartFragment"
    android:label="Cart"
    tools:layout="@layout/fragment_cart"/>
</navigation>
```

Nun werden die fragment-Elemente mit ihren Layouts ergänzt. Dadurch sieht man im Navigations-Editor das jeweilige Layout



Verwenden der Action beim onClickListener

```
package at.htl.shop

import ...

class ShopFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_shop, container, false)
    }
}
```



funktioniert!

```
    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)

        btn_about.setOnClickListener {
            // Navigation.findNavController(it).navigate(R.id.about_destination)

            Navigation.findNavController(it).navigate(R.id.next_action)
        }
    }
}
```

Pass Data between Destinations Using SafeArgs

Documentation

[OVERVIEW](#)
[GUIDES](#)
[REFERENCE](#)
[SAMPLES](#)
[DESIGN & QUALITY](#)
[Adding components to your Project](#)
[Data Binding Library](#)
[Handling Lifecycles](#)
[LiveData](#)
[Navigation](#)
[Overview](#)
[Implement Navigation](#)
[Update UI components with NavigationUI](#)
[Nested graphs](#)
[Pass data between](#)

Pass data between destinations



Navigation allows you to attach data to a navigation operation by defining arguments for a destination. For example, a user profile destination might take a user ID argument to determine which user to display.

In general, you should strongly prefer passing only the minimal amount of data between destinations. For example, you should pass a key to retrieve an object rather than passing the object itself, as the total space for all saved states is limited on Android. If you need to pass large amounts of data, consider using a [ViewModel](#) as described in [Share data between fragments](#).

Contents

[Define destination arguments](#)
[Override a destination argument in an action](#)
[Use Safe Args to pass data with type safety](#)
[Use Safe Args with a global action](#)
[Pass data between destinations with Bundle objects](#)

<https://developer.android.com/topic/libraries/architecture/navigation/navigation-pass-data>

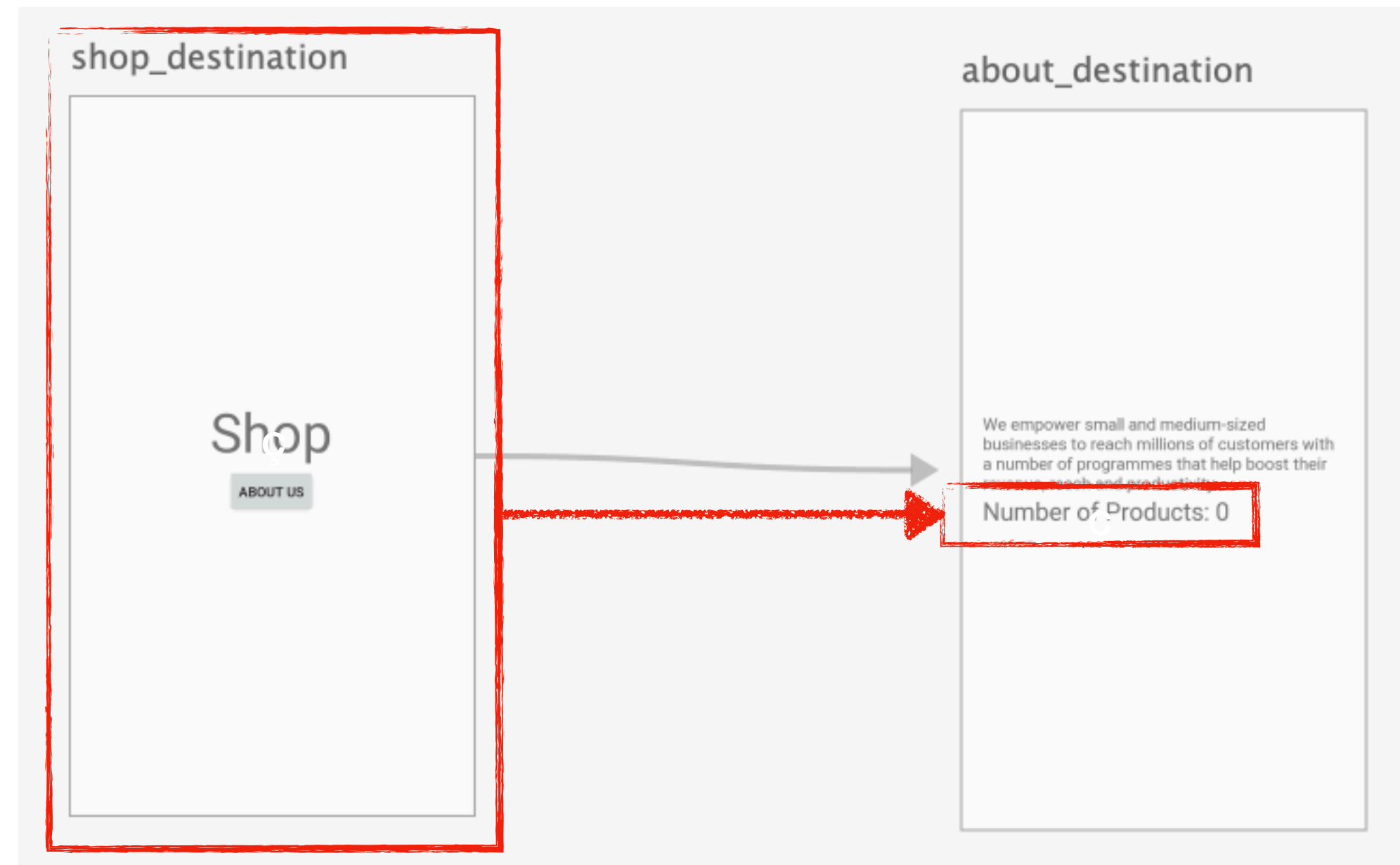
Konfigurieren von SafeArgs

```
Shop x
Configure project in Project Structure dialog.
1 // Top-level build file where you can add configuration options common to all sub-projects/m
2
3 buildscript {
4     ext.kotlin_version = '1.3.21'
5     repositories {
6         google()
7         jcenter()
8     }
9 }
10 dependencies {
11     classpath 'com.android.tools.build:gradle:3.5.0-alpha04'
12     classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin_version"
13     classpath "android.arch.navigation:navigation-safe-args-gradle-plugin:1.0.0-beta02"
14
15     // NOTE: Do not place your application dependencies here; they belong
16     // in the individual module build.gradle files
17 }
18 }
19
20 allprojects {
21     repositories {
22         google()
23         jcenter()
24     }
25 }
26 }
27
28 task clean(type: Delete) {
29     delete rootProject.buildDir
30 }

app x
Configure project in Project Structure dialog.
1 apply plugin: 'com.android.application'
2
3 apply plugin: 'kotlin-android'
4
5 apply plugin: 'kotlin-android-extensions'
6
7 apply plugin: "androidx.navigation.safeargs.kotlin"
8
9 android {
10     compileSdkVersion 28
11     defaultConfig {
12         applicationId "at.htl.shop"
13         minSdkVersion 23
14         targetSdkVersion 28
15         versionCode 1
16         versionName "1.0"
17         testInstrumentationRunner "androidx.test.runner.AndroidJUnitRunner"
18     }
19     buildTypes {
20         release {
21             minifyEnabled false
22             proguardFiles getDefaultProguardFile('proguard-android-optimize.txt')
23         }
24     }
25 }
26
27 dependencies {
28     implementation fileTree(dir: 'libs', include: ['*.jar'])
```

https://developer.android.com/jetpack/androidx/releases/navigation#declaring_dependencies

Aufgabenstellung



Eine Zufallszahl wird vom shop- zum about-fragment übergeben

Parameter eintragen

The screenshot shows the Android Studio interface with the navigation graph editor. The graph shows a transition from 'shop_destination' to 'about_destination'. The 'about_destination' fragment is highlighted with a red box and a blue circle labeled '1'. The 'Add Argument Link' dialog is open, with 'productCount' entered in the 'Name' field, 'Integer' selected in the 'Type' dropdown, and '0' in the 'Default Value' field. The 'Add' button is highlighted with a blue circle labeled '4'. The 'Attributes' panel on the right shows the configuration for the 'about_destination' fragment, including 'Type: Fragment', 'Label: About', 'ID: about_destination', and 'Class: (at.htl.shop)'. The 'Arguments' section shows 'productCount: integer (0)'. A code snippet at the bottom shows the XML for the fragment and argument:

```
<fragment
  android:id="@+id/about_destination"
  android:name="at.htl.shop.AboutFragment"
  android:label="About"
  tools:layout="@layout/fragment_about">
  <argument
    android:name="productCount"
    app:argType="integer"
    android:defaultValue="0"/>
</fragment>
```


ShopFragment.kt

```
package at.htl.shop

import ...

class ShopFragmentDirections private constructor() {
    private data class NextAction(val productCount: Int = 0) : NavDirections {
        override fun getActionId(): Int = at.htl.shop.R.id.next_action

        override fun getArguments(): Bundle {
            val result = Bundle()
            result.putInt("productCount", this.productCount)
            return result
        }
    }

    companion object {
        fun nextAction(productCount: Int = 0): NavDirections = NextAction(productCount)
    }
}
```

```
package at.htl.shop

import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.fragment.app.Fragment
import androidx.navigation.Navigation
import kotlinx.android.synthetic.main.fragment_shop.*
import kotlin.random.Random

class ShopFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_shop, container, false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)

        btn_about.setOnClickListener {
            // Navigation.findNavController(it).navigate(R.id.about_destination)

            //Navigation.findNavController(it).navigate(R.id.next_action)

            val nextAction = ShopFragmentDirections.nextAction(Random.nextInt(200))
            Navigation.findNavController(it).navigate(nextAction)
        }
    }
}
```

AboutFragment.kt

```
package at.htl.shop

import ...

data class AboutFragmentArgs(val productCount: Int = 0) : NavArgs {
    fun toBundle(): Bundle {
        val result = Bundle()
        result.putInt("productCount", this.productCount)
        return result
    }

    companion object {
        @JvmStatic
        fun fromBundle(bundle: Bundle): AboutFragmentArgs {
            bundle.setClassLoader(AboutFragmentArgs::class.java.classLoader)
            val __productCount : Int
            if (bundle.containsKey("productCount")) {
                __productCount = bundle.getInt("productCount")
            } else {
                __productCount = 0
            }
            return AboutFragmentArgs(__productCount)
        }
    }
}
```

```
package at.htl.shop

import android.os.Bundle
import android.view.LayoutInflater
import android.view.View
import android.view.ViewGroup
import androidx.fragment.app.Fragment
import kotlinx.android.synthetic.main.fragment_about.*

class AboutFragment : Fragment() {

    override fun onCreateView(
        inflater: LayoutInflater,
        container: ViewGroup?,
        savedInstanceState: Bundle?
    ): View? {
        // inflate the layout for this fragment
        return inflater.inflate(R.layout.fragment_about, container, false)
    }

    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)

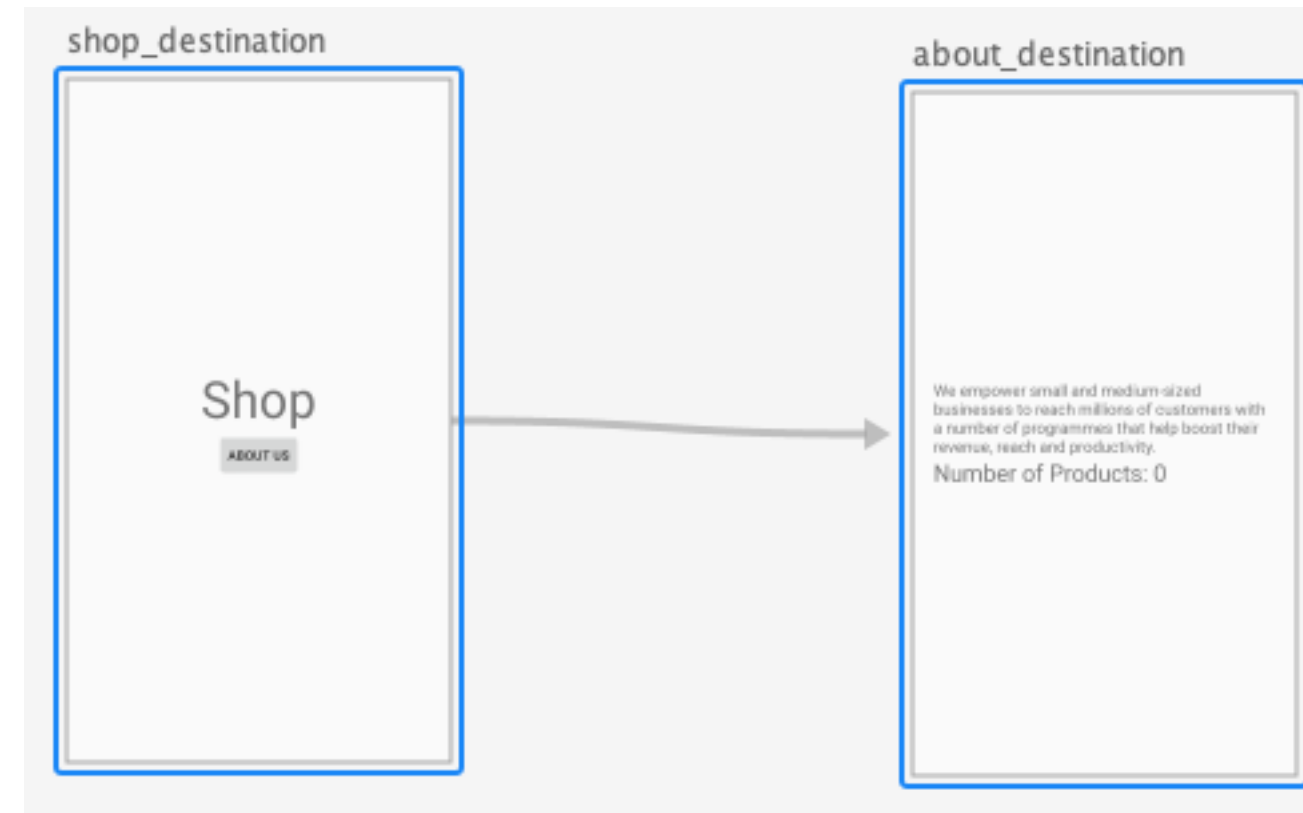
        arguments?.let {
            val safeArgs = AboutFragmentArgs.fromBundle(it)
            tv_product_count.text = "Total Products Available: ${safeArgs.productCount}"
        }
    }
}
```


SENDER

SafeArgs

EMPFÄNGER

```
class ShopFragment : Fragment() {
    override fun onCreateView(...): View? { ... }
    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        btn_about.setOnClickListener {
            val nextAction = ShopFragmentDirections
                .nextAction(Random.nextInt(200))
            Navigation.findNavController(it).navigate(nextAction)
        }
    }
}
```



```
class AboutFragment : Fragment() {
    override fun onCreateView(...): View? { ... }
    override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
        super.onViewCreated(view, savedInstanceState)
        arguments?.let {
            val safeArgs = AboutFragmentArgs.fromBundle(it)
            tv_product_count.text =
                „Total Products Available: ${safeArgs.productCount}“
        }
    }
}
```

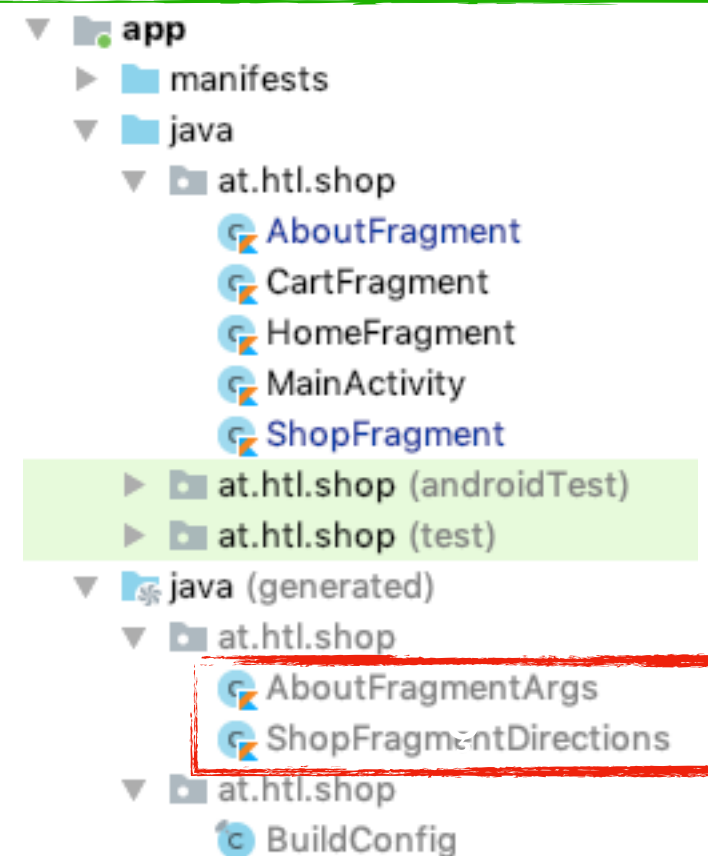
automatisch generierte Klassen

ShopFragmentDirections

NextAction

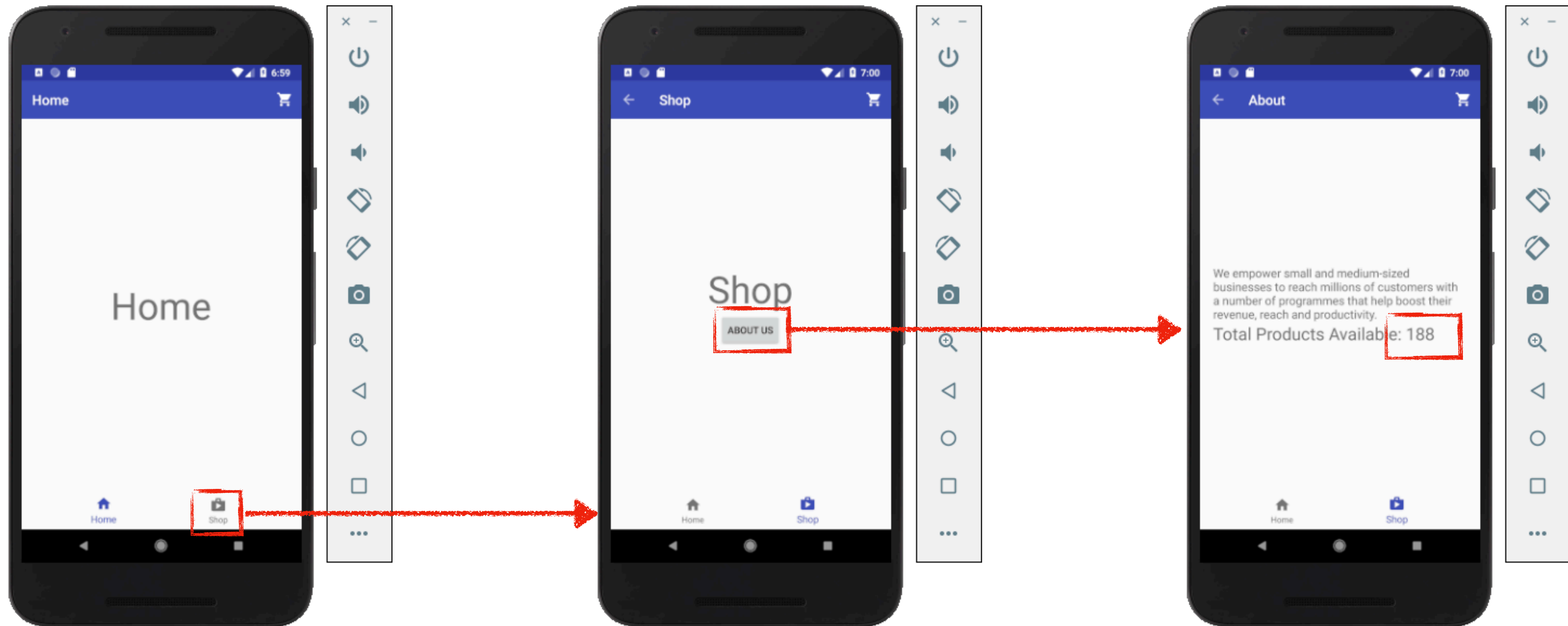
AboutFragmentArgs

```
class ShopFragmentDirections private constructor() {
    private data class NextAction(val productCount: Int = 0) : NavDirections {
        override fun getActionId(): Int = at.htl.shop.R.id.next_action
    }
    override fun getArguments(): Bundle {
        val result = Bundle()
        result.putInt("productCount", this.productCount)
        return result
    }
}
companion object {
    fun nextAction(productCount: Int = 0): NavDirections = NextAction(productCount)
}
```



```
data class AboutFragmentArgs(val productCount: Int = 0) : NavArgs {
    fun toBundle(): Bundle {
        val result = Bundle()
        result.putInt("productCount", this.productCount)
        return result
    }
}
companion object {
    @JvmStatic
    fun fromBundle(bundle: Bundle): AboutFragmentArgs {
        bundle.setClassLoader(AboutFragmentArgs::class.java.classLoader)
        val __productCount : Int
        if (bundle.containsKey("productCount")) {
            __productCount = bundle.getInt("productCount")
        } else {
            __productCount = 0
        }
        return AboutFragmentArgs(__productCount)
    }
}
```


Testlauf



Wie gehts weiter?

- <https://codelabs.developers.google.com/codelabs/android-room-with-a-view/#0>
-



Noch
Fragen?

Source

<https://www.udemy.com/android-jetpack-architecture-components/>

The screenshot shows the UdeMy website interface. At the top, there is a navigation bar with the UdeMy logo, a search bar, and links for 'UdeMy for Business', 'Become an instructor', 'Log In', and 'Sign Up'. Below the navigation bar, the breadcrumb trail reads 'Development > Mobile Apps > Android Game Development'. The main content area features the course title 'Android Jetpack Architecture Components' in large white text on a dark background. Below the title is a subtitle: 'Utilize Android Jetpack Architecture components to make your Android application development flexible and maintainable'. There is a 'NEW' badge, a star rating of 0.0 (0 ratings), and '4 students enrolled'. The course is created by Packt Publishing and was last updated in 2/2019. It is available in English and English [Auto-generated]. To the right of the course title, there are buttons for 'Gift This Course' and 'Wishlist'. Below the course title, there is a video player with a play button and the text 'Preview this course'.

What you'll learn

- ✓ Get introduced to Android architecture components
- ✓ Provide stability in your app by handling life cycles, view models, and live data
- ✓ Load data gradually and gracefully in RecyclerView by using the Paging library
- ✓ Explore how to perform CRUD operations in the Room database
- ✓ Use the Data Binding library to bind data to the UI
- ✓ Implement effective in-app navigation by using the Navigation architecture component
- ✓ Implement a local database to store structured data by using the Room database
- ✓ Schedule tasks asynchronously by using Work Manager

€10.99 ~~€124.99~~ 91% off

🕒 5 hours left at this price!

Add to cart

Buy now

30-Day Money-Back Guarantee

Includes

- 📺 3 hours on-demand video
- 📄 1 downloadable resource
- 🌐 Full lifetime access
- 📱 Access on mobile and TV
- 🏆 Certificate of Completion

