

برنامه نویسی دستگاه های سیار (CE364)

جلسه دوازدهم: مدیریت حرکت بین صفحات

سجاد شیرعلی شمرضا
پاییز 1401
چهارشنبه، 23 آذر 1401

● بخشهای مرتبط با این جلسه:

- Unit 3: Navigation:
 - Pathway 4: Advanced navigation app examples




سوال؟

تعیین نحوه حرکت بین قطعات

برنامه نهایی

Cupcake



Order Cupcakes

ONE CUPCAKE

SIX CUPCAKES

TWELVE CUPCAKES

Choose Flavor

☐ Vanilla

☒ Chocolate

☐ Red Velvet

☐ Salted Caramel

☐ Coffee

Subtotal \$27.00

CANCEL

NEXT

Choose Pickup Date

☐ Fri Dec 11

☒ Sat Dec 12

☐ Sun Dec 13

☐ Mon Dec 14

Subtotal \$24.00

CANCEL

NEXT

Order Summary

QUANTITY
12

FLAVOR
Chocolate

PICKUP DATE
Sat Dec 12

TOTAL \$24.00

SEND ORDER TO ANOTHER APP

CANCEL

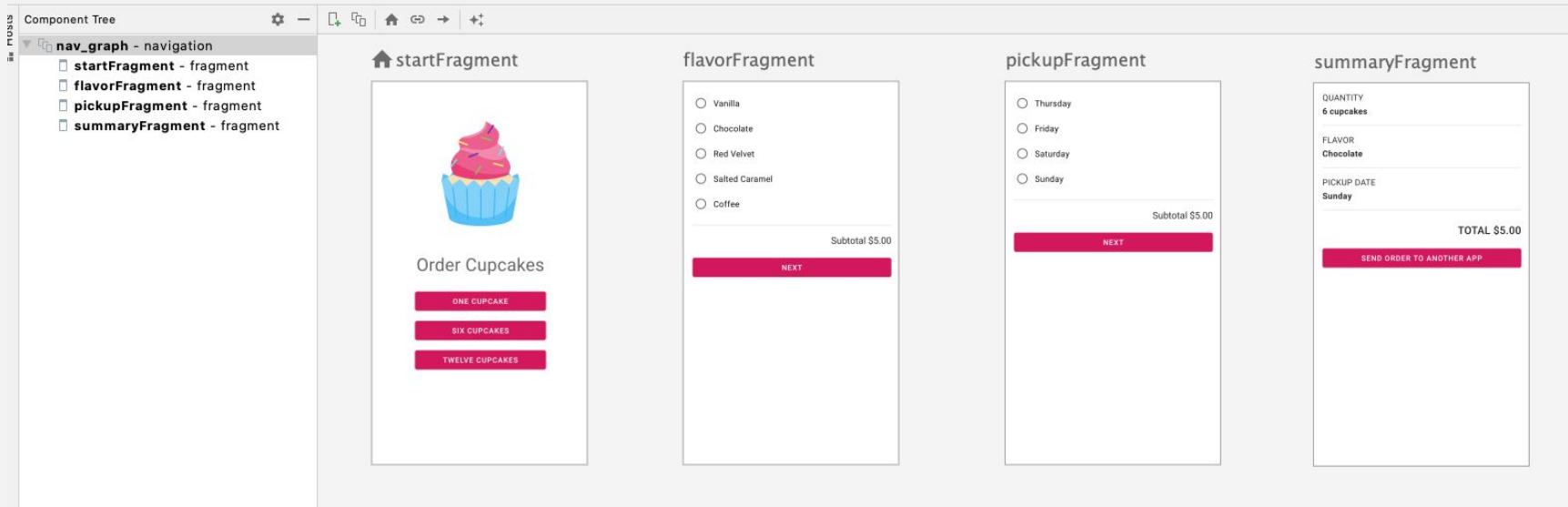
ساختار برنامه اولیه

- فعالیت ها تعریف شده اند
 - صفحه اولیه و انتخاب تعداد کیک ها: fragment_start.xml
 - انتخاب نوع و طعم کیک: fragment_flavor.xml
 - انتخاب روز گرفتن سفارش: fragment_pickup.xml
 - نمایش خلاصه سفارش: fragment_summary.xml
- عدم امکان حرکت بین فعالیت ها
- دکمه ها کار نمی کنند
- کلاس های متناظر با قطعه ها
 - تنها نمایش یک پیغام با فشار دادن هر کلیک

گراف حرکت بین صفحات

- باز کردن فایل گراف: `res > navigation > nav_graph.xml`
- انتخاب حالت تغییر گرافیکی

Code Split Design



اضافه کردن حرکت از یک قطعه به دیگر

↑ startFragment

flavorFragment



Order Cupcakes

ONE CUPCAKE

SIX CUPCAKES

TWELVE CUPCAKES

- ☐ Vanilla
- ☐ Chocolate
- ☐ Red Velvet
- ☐ Salted Caramel
- ☐ Coffee

Subtotal \$5.00

NEXT

- حرکت نمایشگر موشواره بر روی کادر خاکستری دور قطعه
- کشیدن از نقطه خاکستری نمایش داده شده تا قطعه مقصد

اضافه کردن حرکت ها


- تعیین نحوه حرکت بین قطعات:

startFragment -> flavorFragment

flavorFragment -> pickupFragment

pickupFragment -> summaryFragment

🏠 startFragment



Order Cupcakes

ONE CUPCAKE

SIX CUPCAKES

TWELVE CUPCAKES

flavorFragment

☐ Vanilla

☐ Chocolate

☐ Red Velvet

☐ Salted Caramel

☐ Coffee

Subtotal \$5.00

NEXT

حرکت به سمت جلو در گراف

🏠 startFragment



Order Cupcakes

ONE CUPCAKE

SIX CUPCAKES

TWELVE CUPCAKES

flavorFragment

- ☐ Vanilla
- ☐ Chocolate
- ☐ Red Velvet
- ☐ Salted Caramel
- ☐ Coffee

Subtotal \$5.00

NEXT

pickupFragment

- ☐ Thursday
- ☐ Friday
- ☐ Saturday
- ☐ Sunday

Subtotal \$5.00

NEXT

summaryFragment

QUANTITY
6 cupcakes

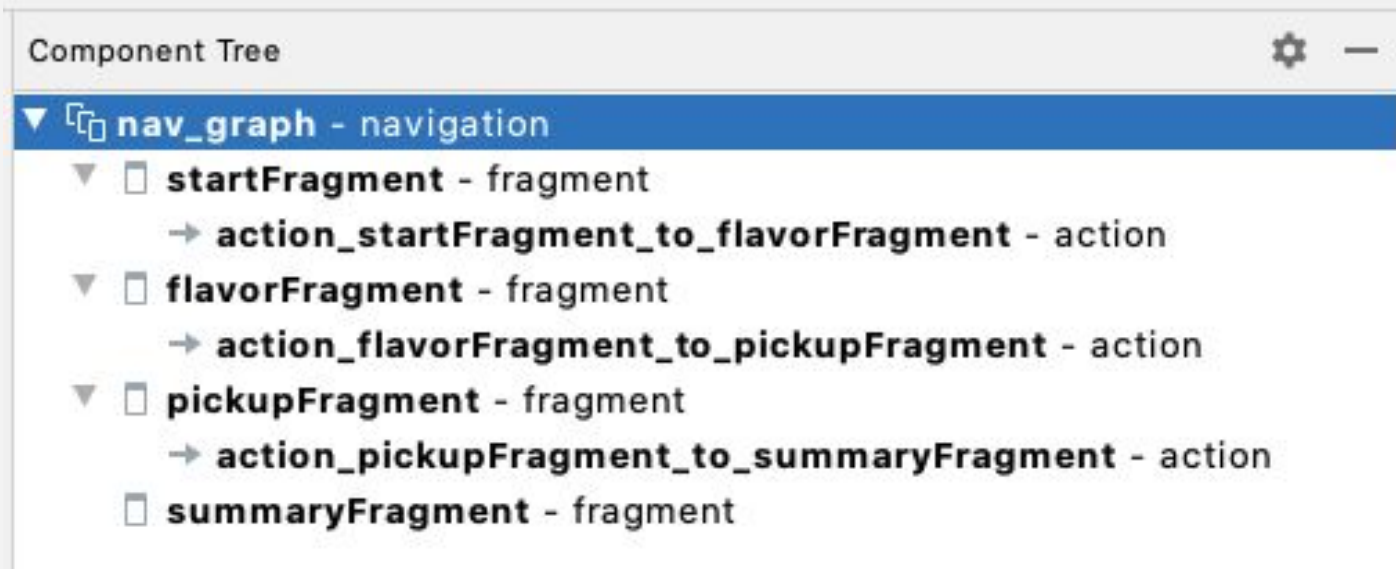
FLAVOR
Chocolate

PICKUP DATE
Sunday

TOTAL \$5.00

SEND ORDER TO ANOTHER APP

معادل حرکت در نمایش درختی



تعیین قطعه صفحه اول

🏠 startFragment

flavorFragment

pick

- ☐ Vanilla
- ☐ Chocolate
- ☐ Red Velvet
- ☐ Salted Caramel
- ☐ Coffee

NEXT

Edit

Add Action ▶

Move to Nested Graph ▶

Set as Start Destination

Scroll into view

✂ Cut ⌘X

📄 Copy ⌘C

📄 Paste ⌘V

Delete ⌘⌫

Go to XML

حرکت از صفحه اول به صفحه انتخاب طعم

- به روز رسانی onViewCreated فایل StartFragment در app>java>com.example.cupcake>

```
orderOneCupcake.setOnClickListener { orderCupcake(1) }  
orderSixCupcakes.setOnClickListener { orderCupcake(6) }  
orderTwelveCupcakes.setOnClickListener { orderCupcake(12) }
```

- تعریف تابع سفارش کیک

```
fun orderCupcake(quantity: Int) {  
    findNavController().navigate(R.id.action_startFragment_to_flavorFragment)  
}
```

حرکت به سمت جلو در صفحات دیگر

- در فایل `app > java > com.example.cupcake > PickupFragment.kt`

```
fun goToNextScreen() {  
    findNavController().navigate(R.id.action_flavorFragment_to_pickupFragment)  
}
```

- در فایل `app > java > com.example.cupcake > FlavorFragment.kt`

```
fun goToNextScreen() {  
    findNavController().navigate(R.id.action_pickupFragment_to_summaryFragment)  
}
```

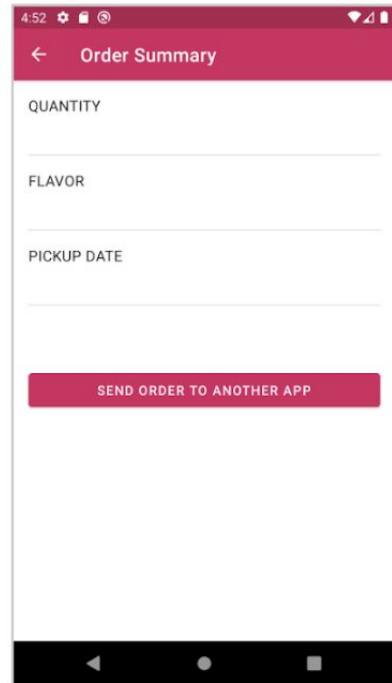
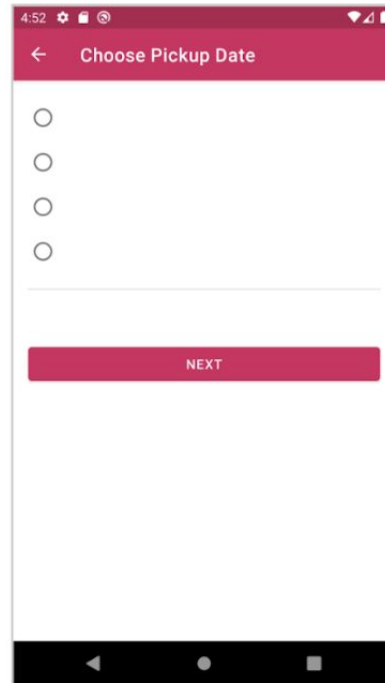
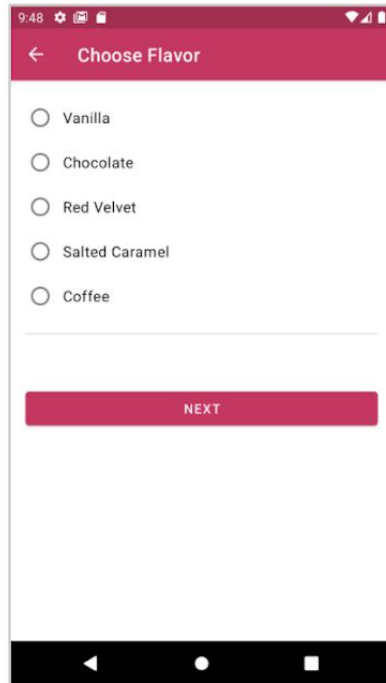
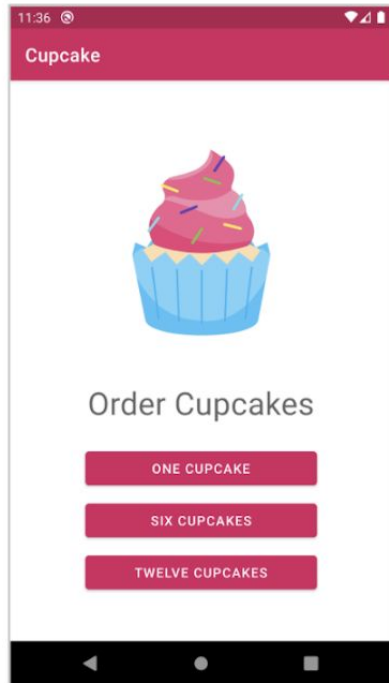
آماده سازی عنوان برنامه جهت تغییر آن

```
class MainActivity : AppCompatActivity(R.layout.activity_main) {  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
  
        val navHostFragment = supportFragmentManager  
            .findFragmentById(R.id.nav_host_fragment) as NavHostFragment  
        val navController = navHostFragment.navController  
  
        setupActionBarWithNavController(navController)  
    }  
}
```

نمایش عنوان مناسب برای هر قطعه

```
<navigation ...>
  <fragment
    android:id="@+id/startFragment"
    ...
    android:label="@string/app_name" ... >
    <action ... />
  </fragment>
  <fragment
    android:id="@+id/flavorFragment"
    ...
    android:label="@string/choose_flavor" ... >
    <action ... />
  </fragment>
  <fragment
    android:id="@+id/pickupFragment"
    ...
    android:label="@string/choose_pickup_date" ... >
    <action ... />
  </fragment>
  <fragment
    android:id="@+id/summaryFragment"
    ...
    android:label="@string/order_summary" ... />
</navigation>
```


برنامه تا اینجا

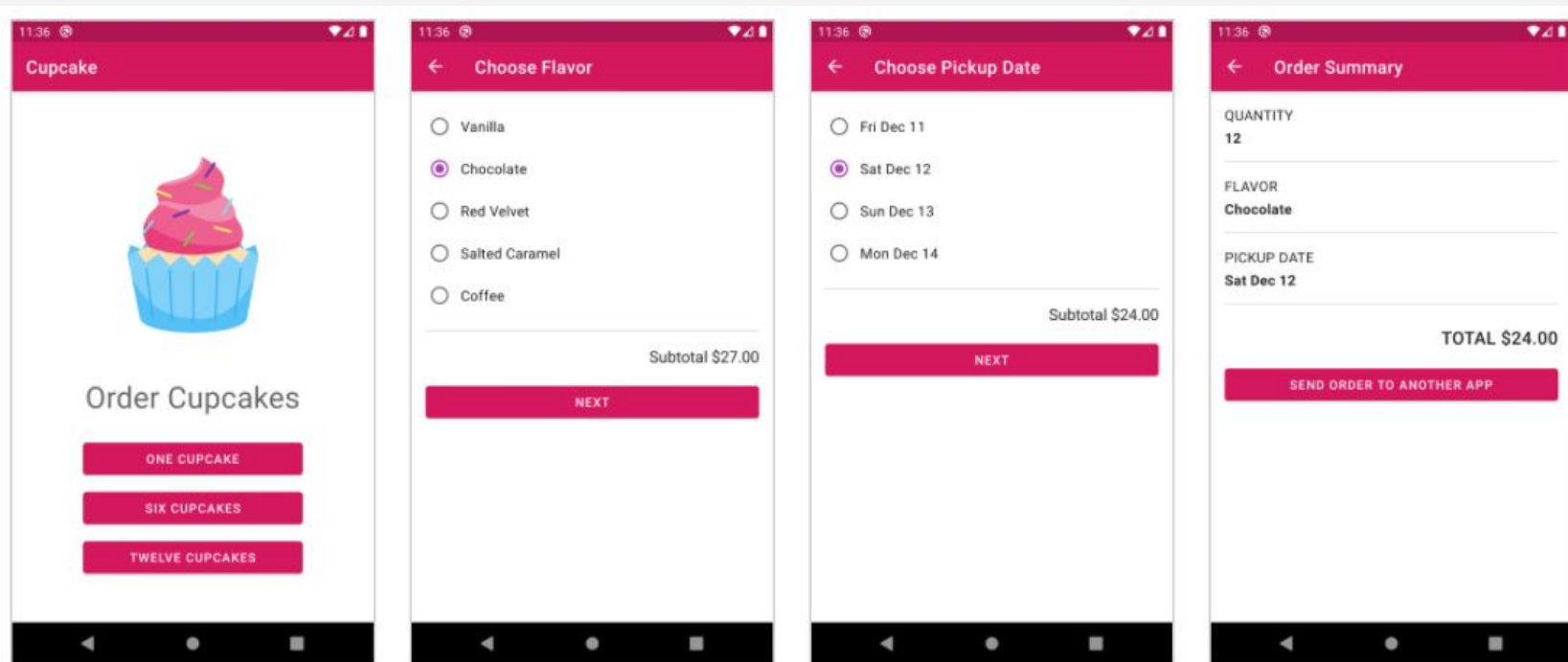




سوال؟

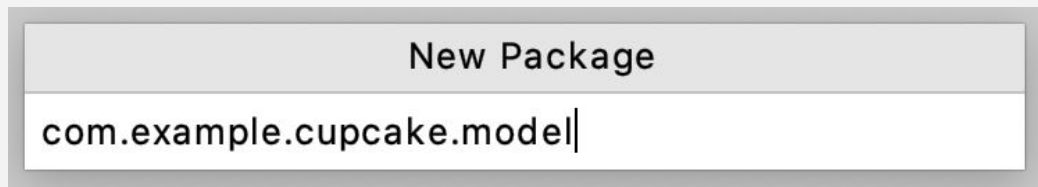
ایجاد مدل نمای مشترک

- داشتن یک نمای مدل مشترک بین قطعات جهت نگه داری اطلاعات

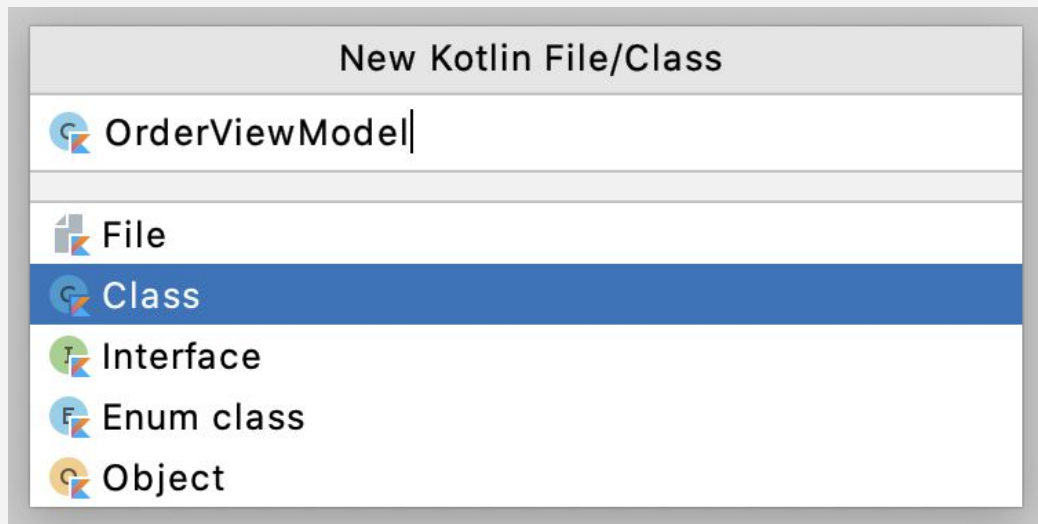


افزافه کردن مدل نما

- انتخاب گزینه یک بسته جدید (com.example.cupcake > New > Package)
- تعیین نام برای بسته



- تعیین نام کلاس مدل نما



تعیین داده ها در مدل نما

```
import androidx.lifecycle.ViewModel

class OrderViewModel : ViewModel() {

    private val _quantity = MutableLiveData<Int>(0)
    val quantity: LiveData<Int> = _quantity

    private val _flavor = MutableLiveData<String>("")
    val flavor: LiveData<String> = _flavor

    private val _date = MutableLiveData<String>("")
    val date: LiveData<String> = _date

    private val _price = MutableLiveData<Double>(0.0)
    val price: LiveData<Double> = _price
```

- تعیین کلاس پایه برای کلاس جدید

- تعریف داده ها

افزافه کردن توابع تعیین مقدار

- برای قیمت نیازی نیست، چون به صورت خودکار حساب می شود

```
fun setQuantity(numberCupcakes: Int) {  
    _quantity.value = numberCupcakes  
}
```

```
fun setFlavor(desiredFlavor: String) {  
    _flavor.value = desiredFlavor  
}
```

```
fun setDate(pickupDate: String) {  
    _date.value = pickupDate  
}
```

نحوه دسترسی به مدل نمای مشترک

- استفاده از مدل نمای موجود در فعالیت (به جای خود قطعه)
 - برای مدل نمای خود قطعه: `viewModels`
 - برای مدل نمای فعالیت: `activityViewModels`
- گرفتن یک ارجاع به مدل نمای مشترک در کلاس `:StartFragment`

```
private val sharedViewModel: OrderViewModel by activityViewModels()
```

- ذخیره تعداد کیک انتخاب شده:

```
fun orderCupcake(quantity: Int) {  
    sharedViewModel.setQuantity(quantity)  
    findNavController().navigate(R.id.action_startFragment_to_flavorFragment)  
}
```


تعیین طعم پیش فرض

- چک کردن طعم انتخاب شده در OrderViewModel

```
fun hasNoFlavorSet(): Boolean {  
    return _flavor.value.isNullOrEmpty()  
}
```

- تعیین طعم پیش فرض

```
fun orderCupcake(quantity: Int) {  
    sharedViewModel.setQuantity(quantity)  
    if (sharedViewModel.hasNoFlavorSet()) {  
        sharedViewModel.setFlavor(getString(R.string.vanilla))  
    }  
    findNavController().navigate(R.id.action_startFragment_to_flavorFragment)  
}
```

به روز کردن طعم از روی مدل نما

```
<layout ...>
```

```
  <data>
```

```
    <variable
```

```
      name="viewModel"
```

```
      type="com.example.cupcake.model.OrderViewModel" />
```

```
  </data>
```

```
  <ScrollView ...>
```

```
    ...
```

- به روز رسانی fragment_flavor.xml

- متصل کردن مدل نما در تابع onCreateView کلاس FlavorFragment

```
binding?.apply {  
    viewModel = sharedViewModel  
    ...  
}
```

ساختار apply در زبان کاتلین

- اجرای کد در حوزه یک شی

```
clark.apply {  
    firstName = "Clark"  
    lastName = "James"  
    age = 18  
}
```

// The equivalent code without apply scope function would look like the following.

```
clark.firstName = "Clark"  
clark.lastName = "James"  
clark.age = 18
```

انتخاب گزینه مناسب در لیست گزینه ها

```
<RadioGroup
...>

<RadioButton
    android:id="@+id/vanilla"
    ...
    android:checked="@{viewModel.flavor.equals(@string/vanilla)}"
.../>

<RadioButton
    android:id="@+id/chocolate"
    ...
    android:checked="@{viewModel.flavor.equals(@string/chocolate)}"
.../>

<RadioButton
    android:id="@+id/red_velvet"
    ...
    android:checked="@{viewModel.flavor.equals(@string/red_velvet)}"
.../>

<RadioButton
    android:id="@+id/salted_caramel"
    ...
    android:checked="@{viewModel.flavor.equals(@string/salted_caramel)}"
.../>

<RadioButton
    android:id="@+id/coffee"
    ...
    android:checked="@{viewModel.flavor.equals(@string/coffee)}"
.../>
</RadioGroup>
```

تغییر مدل نما با تغییر گزینه انتخابی

```
<RadioGroup
...>

<RadioButton
    android:id="@+id/vanilla"
    ...
    android:onClick="@{() -> viewModel.setFlavor(@string/vanilla)}"
.../>

<RadioButton
    android:id="@+id/chocolate"
    ...
    android:onClick="@{() -> viewModel.setFlavor(@string/chocolate)}"
.../>

<RadioButton
    android:id="@+id/red_velvet"
    ...
    android:onClick="@{() -> viewModel.setFlavor(@string/red_velvet)}"
.../>

<RadioButton
    android:id="@+id/salted_caramel"
    ...
    android:onClick="@{() -> viewModel.setFlavor(@string/salted_caramel)}"
.../>

<RadioButton
    android:id="@+id/coffee"
    ...
    android:onClick="@{() -> viewModel.setFlavor(@string/coffee)}"
.../>
</RadioGroup>
```



سوال؟

تاریخ دریافت سفارش

پردازش و نمایش تاریخ

- هدف: نمایش 4 تاریخ آماده شدن سفارش و انتخاب توسط کاربر
- استفاده از کلاس SimpleDateFormat:
 - فرمت کردن و پردازش تاریخ
 - تبدیل تاریخ به متن و متن به تاریخ
 - متناسب با اطلاعات محلی سازی شده کاربر
- ایجاد یک نمونه به صورت
 - روز: d
 - ماه: M
 - سال: y
 - روز هفته: E

```
SimpleDateFormat("E MMM d", Locale.getDefault())
```


اطلاعات محلی (locale)

- نمایش دهنده یک ناحیه جغرافیایی، فرهنگی، و یا سیاسی
- استفاده برای تعیین نوع نمایش اطلاعات
- گرفتن محل تعیین شده توسط کاربر به وسیله : `Locale.getDefault`
- در اندروید: ترکیب زبان و کشور
 - زبان یک رشته دو حرفی کوچک مشخص کننده زبان: مثلا en و یا fa
 - کشور یک رشته دو حرفی بزرگ مشخص کننده کشور: مثلا US و یا IR

ایجاد تاریخ های موجود

```
private fun getPickupOptions(): List<String> {  
    val options = mutableListOf<String>()  
    val formatter = SimpleDateFormat("E MMM d", Locale.getDefault())  
    val calendar = Calendar.getInstance()  
    // Create a list of dates starting with the current date and the following 3 dates  
    repeat(4) {  
        options.add(formatter.format(calendar.time))  
        calendar.add(Calendar.DATE, 1)  
    }  
    return options  
}
```

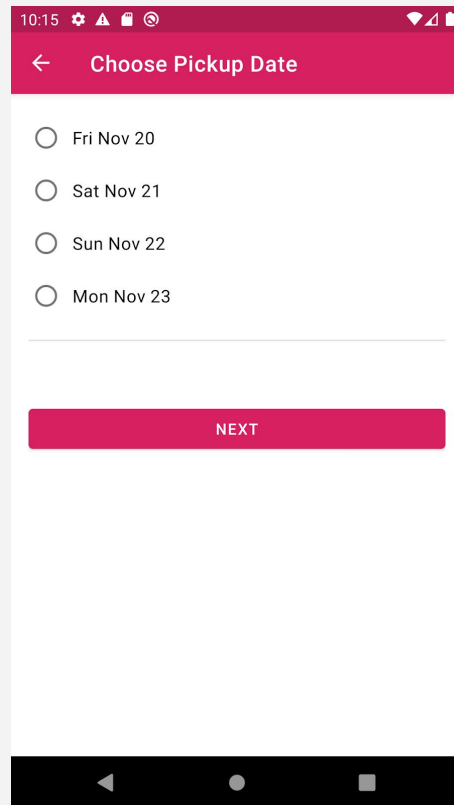
نمایش تاریخ های انتخابی به کاربر

```
<RadioButton
    android:id="@+id/option0"
    ...
    android:checked="@{viewModel.date.equals(viewModel.dateOptions[0])}"
    android:onClick="@{() -> viewModel.setDate(viewModel.dateOptions[0])}"
    android:text="@{viewModel.dateOptions[0]}"
    ...
/>

<RadioButton
    android:id="@+id/option1"
    ...
    android:checked="@{viewModel.date.equals(viewModel.dateOptions[1])}"
    android:onClick="@{() -> viewModel.setDate(viewModel.dateOptions[1])}"
    android:text="@{viewModel.dateOptions[1]}"
    ... />

<RadioButton
    android:id="@+id/option2"
    ...
    android:checked="@{viewModel.date.equals(viewModel.dateOptions[2])}"
    android:onClick="@{() -> viewModel.setDate(viewModel.dateOptions[2])}"
    android:text="@{viewModel.dateOptions[2]}"
    ... />

<RadioButton
    android:id="@+id/option3"
    ...
    android:checked="@{viewModel.date.equals(viewModel.dateOptions[3])}"
    android:onClick="@{() -> viewModel.setDate(viewModel.dateOptions[3])}"
    android:text="@{viewModel.dateOptions[3]}"
    ... />
```



ریست کردن اطلاعات وارد شده

```
fun resetOrder() {  
    _quantity.value = 0  
    _flavor.value = ""  
    _date.value = dateOptions[0]  
    _price.value = 0.0  
}
```

```
init {  
    resetOrder()  
}
```

```
private val _quantity = MutableLiveData<Int>()  
val quantity: LiveData<Int> = _quantity
```

```
private val _flavor = MutableLiveData<String>()  
val flavor: LiveData<String> = _flavor
```

```
private val _date = MutableLiveData<String>()  
val date: LiveData<String> = _date
```

```
private val _price = MutableLiveData<Double>()  
val price: LiveData<Double> = _price
```



سوال؟

نمایش خلاصه سفارش

استفاده از مدل نما

- اضافه کردن مدل نما به fragment_summary.xml

```
<layout ...>

    <data>
        <variable
            name="viewModel"
            type="com.example.cupcake.model.OrderViewModel" />
    </data>

    <ScrollView ...>

        ...
```

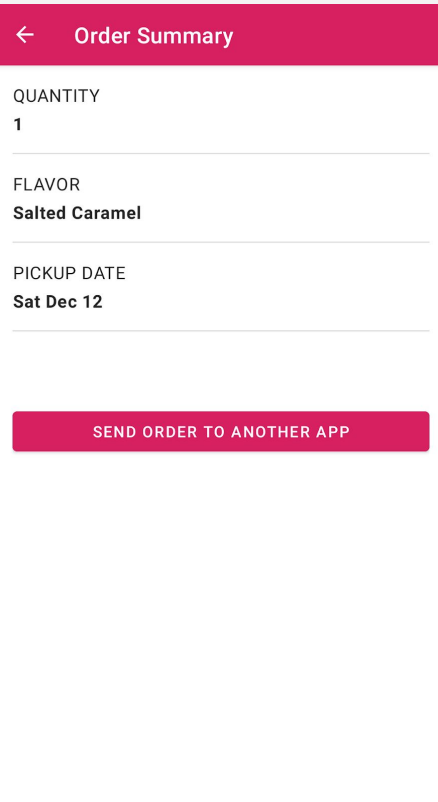
- مقدار اولیه دادن (binding.viewModel) در onCreateView

نمایش اطلاعات در قطعه خلاصه

```
<TextView
    android:id="@+id/quantity"
    ...
    android:text="@{viewModel.quantity.toString()}"
    ... />

<TextView
    android:id="@+id/flavor"
    ...
    android:text="@{viewModel.flavor}"
    ... />

<TextView
    android:id="@+id/date"
    ...
    android:text="@{viewModel.date}"
    ... />
```



محاسبه قیمت

```
package ...
```

```
import ...
```

```
private const val PRICE_PER_CUPCAKE = 2.00
```

```
class OrderViewModel : ViewModel() {  
    ...
```

```
private fun updatePrice() {  
    _price.value = (quantity.value ?: 0) * PRICE_PER_CUPCAKE  
}
```

```
fun setQuantity(numberCupcakes: Int) {  
    _quantity.value = numberCupcakes  
    updatePrice()  
}
```

- قیمت هر کیک 2 دلار
- هزینه گرفتن سفارش در همان روز 3 دلار

نمایش قیمت در هر صفحه

```
<layout ...>
```

```
    <data>
```

```
        <variable
```

```
            name="viewModel"
```

```
            type="com.example.cupcake.model.OrderViewModel" />
```

```
    </data>
```

```
    <ScrollView ...>
```

```
        ...
```

```
    ...
```

```
    <TextView
```

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

```
        android:text="@{@string/subtotal_price(viewModel.price)}"
```

```
        ... />
```

```
    ...
```

```
    <string name="subtotal_price">Subtotal %s</string>
```

• در `onViewCreated`

```
binding?.apply {  
    viewModel = sharedViewModel  
    ...  
}
```

هزینه اضافه گرفتن سفارش در همان روز

```
private const val PRICE_FOR_SAME_DAY_PICKUP = 3.00
```

```
private fun updatePrice() {  
    var calculatedPrice = (quantity.value ?: 0) * PRICE_PER_CUPCAKE  
    // If the user selected the first option (today) for pickup, add the surcharge  
    if (dateOptions[0] == _date.value) {  
        calculatedPrice += PRICE_FOR_SAME_DAY_PICKUP  
    }  
    _price.value = calculatedPrice  
}
```

```
fun setDate(pickupDate: String) {  
    _date.value = pickupDate  
    updatePrice()  
}
```

به روز کردن نما پس از تغییر قیمت




- اضافه کردن در `onViewCreated` قطعات برای استفاده از داده زنده (`LiveData`)

```
binding?.apply {  
    lifecycleOwner = viewLifecycleOwner  
    ...  
}
```

- تبدیل قیمت قبل از نمایش

```
private val _price = MutableLiveData<Double>()  
val price: LiveData<String> = Transformations.map(_price) {  
    NumberFormat.getCurrencyInstance().format(it)  
}
```

اثر تبدیل قیمت

10:50   

← Order Summary




QUANTITY
6

FLAVOR
Red Velvet

PICKUP DATE
Sun Dec 13

TOTAL 12.0

SEND ORDER TO ANOTHER APP

11:44   

← Choose Flavor

☐ Vanilla

☒ Chocolate




☐ Red Velvet

☐ Salted Caramel

☐ Coffee

Subtotal \$24.00

CANCEL NEXT

11:44   

← Choose Pickup Date

☐ Mon Dec 7




☐ Tue Dec 8

☒ Wed Dec 9

☐ Thu Dec 10

Subtotal \$24.00

CANCEL NEXT

11:44   

← Order Summary

QUANTITY
12

FLAVOR
Chocolate

PICKUP DATE
Wed Dec 9

TOTAL \$24.00

SEND ORDER TO ANOTHER APP

CANCEL

پردازش فشردن کلیدها در صفحه اول

```
<layout ...>

    <data>
        <variable
            name="startFragment"
            type="com.example.cupcake.StartFragment" />
        </data>
        ...
    <ScrollView ...>
```

```
override fun onViewCreated(view: View, savedInstanceState: Bundle?) {
    super.onViewCreated(view, savedInstanceState)
    binding?.startFragment = this
}
```

```
<Button
    android:id="@+id/order_one_cupcake"
    android:onClick="@{() -> startFragment.orderCupcake(1)}"
    ... />

<Button
    android:id="@+id/order_six_cupcakes"
    android:onClick="@{() -> startFragment.orderCupcake(6)}"
    ... />

<Button
    android:id="@+id/order_twelve_cupcakes"
    android:onClick="@{() -> startFragment.orderCupcake(12)}"
    ... />
```

اتصال داده زنده در بقیه قطعات (مثلا FlavorFragment)

```
<layout ...>
```

```
    <data>
```

```
        <variable
```

```
            ... />
```

```
        <variable
```

```
            name="flavorFragment"
```

```
            type="com.example.cupcake.FlavorFragment" />
```

```
    </data>
```

```
    <ScrollView ...>
```

```
    <Button
```

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

```
        android:onClick="@{() -> flavorFragment.goToNextScreen()}" }
```

```
    ... />
```

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

```
    binding?.apply {
```

```
        lifecycleOwner = viewLifecycleOwner
```

```
        viewModel = sharedViewModel
```

```
        flavorFragment = this@FlavorFragment
```

```
    }
```



سوال؟

پشتیبانی از دکمه برگشت

- پشتیبانی از برگشت به مرحله قبل (با فشردن دکمه برگشت در بالا)
- لغو سفارش و شروع از ابتدا (با فشردن دکمه لغو)
- اشتراک گذاری سفارش با برنامه های دیگر

دکمه بازگشت

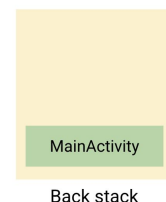
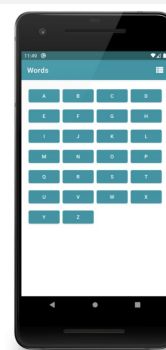
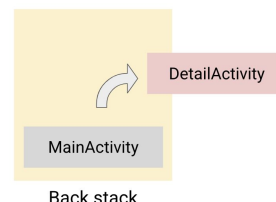
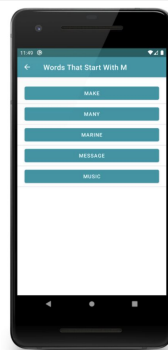
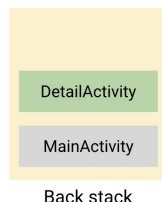
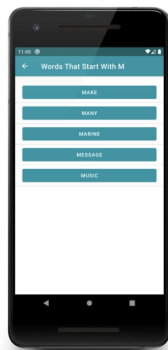
- یک فلش به سمت چپ در بالا در سمت چپ صفحه
- معروف به دکمه "Up"

تغییر نحوه پردازش دکمه بازگشت

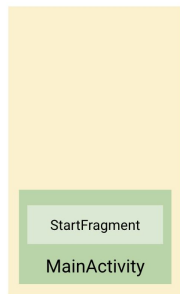
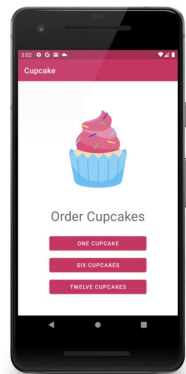
```
class MainActivity : AppCompatActivity(R.layout.activity_main) {  
  
    private lateinit var navController: NavController  
  
    override fun onCreate(savedInstanceState: Bundle?) {  
        super.onCreate(savedInstanceState)  
  
        val navHostFragment = supportFragmentManager  
            .findFragmentById(R.id.nav_host_fragment) as NavHostFragment  
        navController = navHostFragment.navController  
  
        setupActionBarWithNavController(navController)  
    }  
}  
  
override fun onSupportNavigateUp(): Boolean {  
    return navController.navigateUp() || super.onSupportNavigateUp()  
}
```

کار (task)

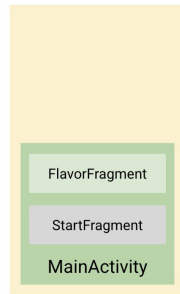
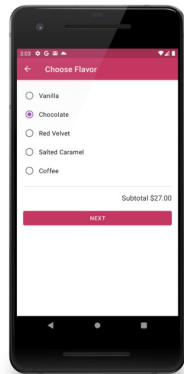
- هر فعالیت در قالب یک کار اجرا می شود
- فشردن دکمه برگشت باعث حذف کار فعلی و استفاده از کار قبلی می شود



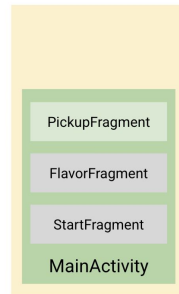
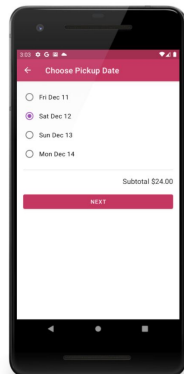
برگشت در برنامه سفارش کیک



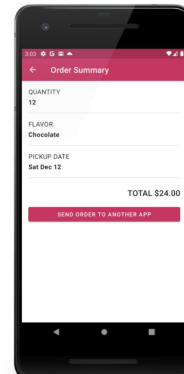
Back stack



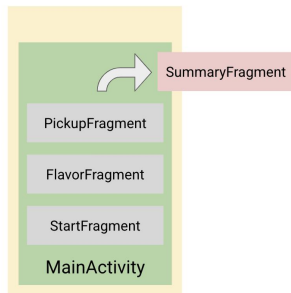
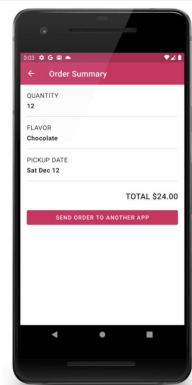
Back stack



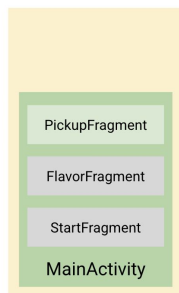
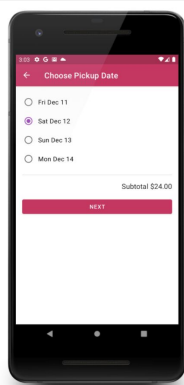
Back stack



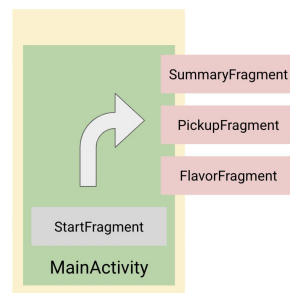
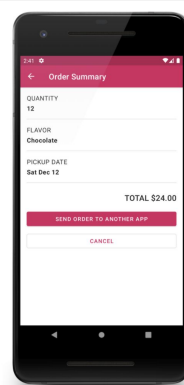
Back stack



Back stack

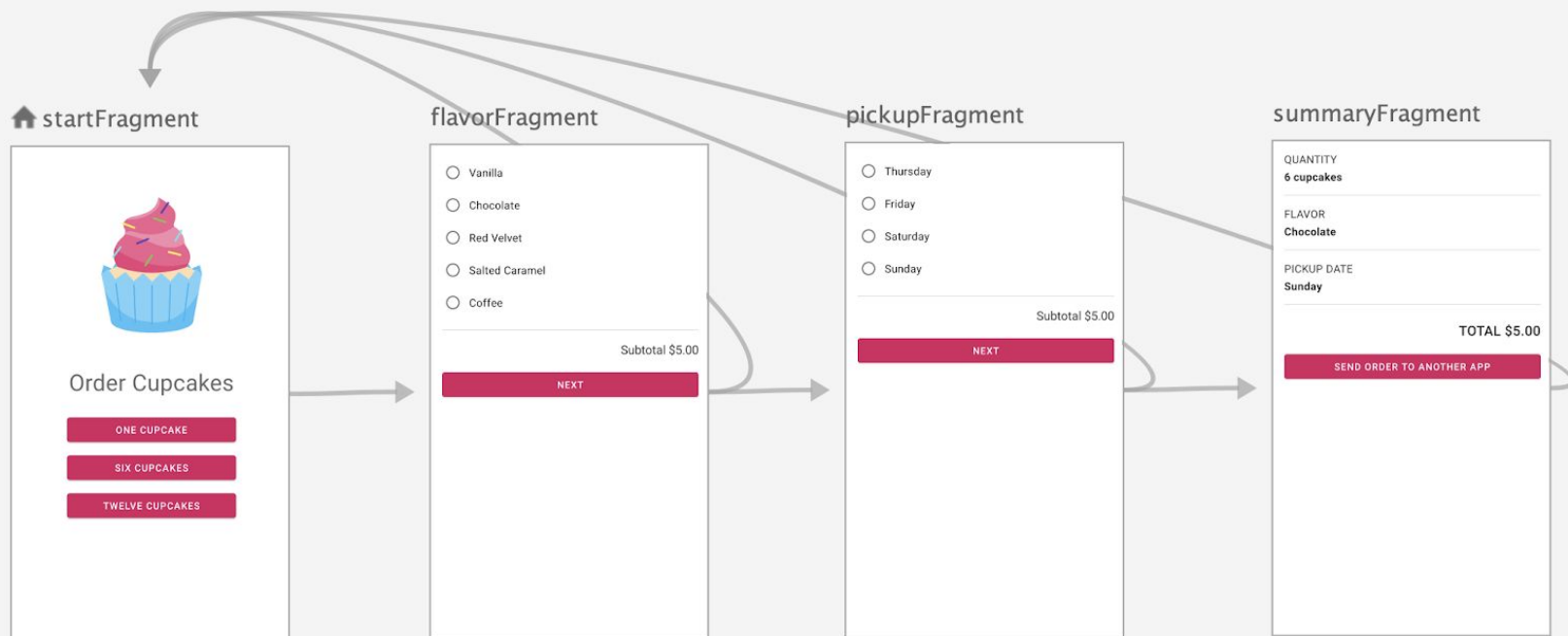


Back stack



Back stack

اضافه کردن حرکت برای برگشت به صفحه اول



CANCEL

NEXT

اضافه کردن دکمه لغو

```
...  
  
<TextView  
    android:id="@+id/subtotal" ... />  
  
<Button  
    android:id="@+id/cancel_button"  
    android:layout_width="0dp"  
    android:layout_height="wrap_content"  
    android:text="@string/cancel"  
    app:layout_constraintEnd_toStartOf="@id/next_button"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toTopOf="@id/next_button" />  
  
<Button  
    android:id="@+id/next_button" ... />  
  
...
```

```
...  
  
<Button  
    android:id="@+id/cancel_button"  
    android:layout_marginEnd="@dimen/side_margin" ... />  
  
<Button  
    android:id="@+id/next_button"  
    app:layout_constraintStart_toEndOf="@id/cancel_button" ... />  
  
...  
  
<Button  
    android:id="@+id/cancel_button"  
    style="?attr/materialButtonOutlinedStyle" ... />
```


دکمه لغو در صفحه خلاصه سفارش

...

```
<Button  
    android:id="@+id/send_button" ... />
```

```
<Button  
    android:id="@+id/cancel_button"  
    style="?attr/materialButtonOutlinedStyle"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:layout_marginTop="@dimen/margin_between_elements"  
    android:text="@string/cancel" />
```

```
</LinearLayout>
```

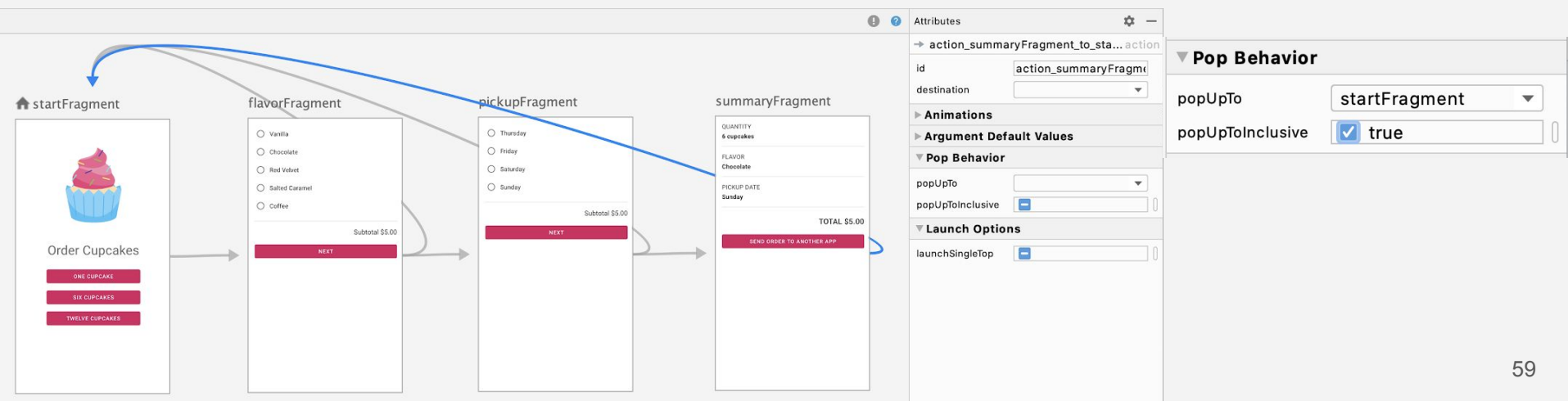
پردازش دکمه لغو

```
fun cancelOrder() {  
    sharedViewModel.resetOrder()  
    findNavController().navigate(R.id.action_flavorFragment_to_startFragment)  
}
```

```
<Button  
    android:id="@+id/cancel_button"  
    android:onClick="@{() -> flavorFragment.cancelOrder()}" ... />
```

حذف تمامی قطعات قبلی هنگام لغو

- اضافه کردن `app:popUpTo` به عمل رفتن به صفحه اول
- `app:popUpTo="@id/startFragment"`
- برای اینکه خود قطعه اول را نیز بردارد:
- `app:popUpToInclusive="true"`



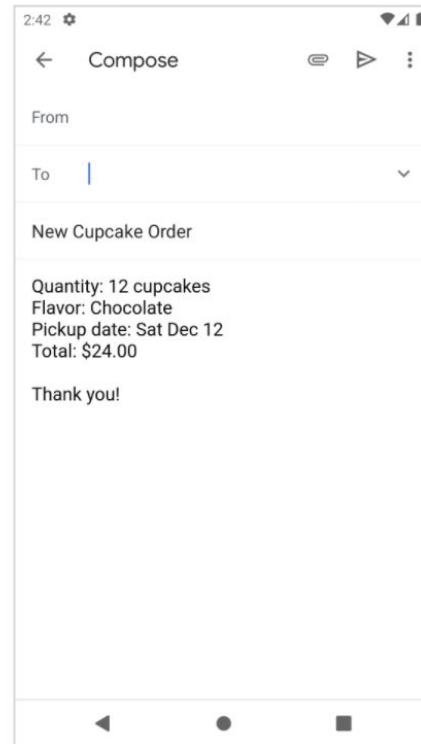
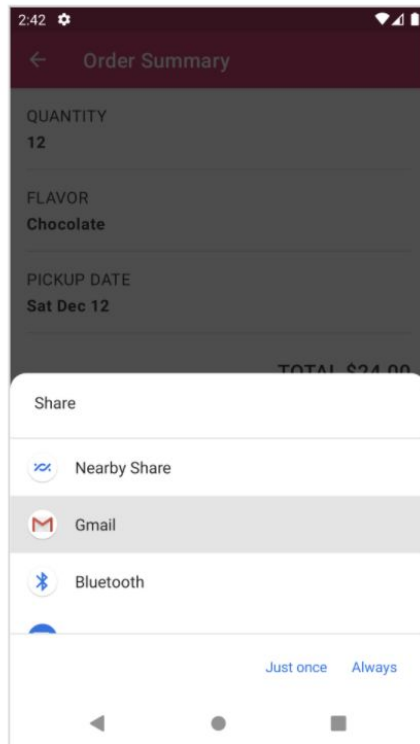
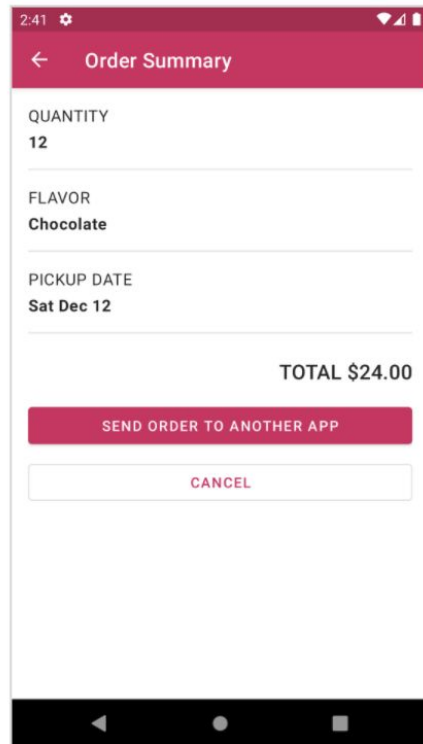
گراف حرکت پس از اضافه کردن این موارد

```
<navigation
  android:id="@+id/nav_graph" ...>
  <fragment
    android:id="@+id/startFragment" ...>
    ...
  </fragment>
  <fragment
    android:id="@+id/flavorFragment" ...>
    ...
    <action
      android:id="@+id/action_flavorFragment_to_startFragment"
      app:destination="@id/startFragment"
      app:popUpTo="@id/startFragment"
      app:popUpToInclusive="true" />
    </fragment>
  <fragment
    android:id="@+id/pickupFragment" ...>
    ...
    <action
      android:id="@+id/action_pickupFragment_to_startFragment"
      app:destination="@id/startFragment"
      app:popUpTo="@id/startFragment"
      app:popUpToInclusive="true" />
    </fragment>
  <fragment
    android:id="@+id/summaryFragment" ...>
    <action
      android:id="@+id/action_summaryFragment_to_startFragment"
      app:destination="@id/startFragment"
      app:popUpTo="@id/startFragment"
      app:popUpToInclusive="true" />
    </fragment>
</navigation>
```



سوال؟

فرستادن سفارش



تعریف داده برای ارسال

- اضافه کردن در فایل strings.xml

```
<string name="new_cupcake_order">New Cupcake Order</string>
<string name="order_details">Quantity: %1$s cupcakes \n Flavor: %2$s \n Pickup
```

- داده هایی که باید جایگذاری شوند با %1 تا %4 مشخص شده اند.
- نحوه جایگذاری اطلاعات:
- `getString(R.string.order_details, "12", "Chocolate", "Sat Dec 12", "$24.00")`

```
Quantity: 12 cupcakes
Flavor: Chocolate
Pickup date: Sat Dec 12
Total: $24.00
```

```
Thank you!
```


آماده سازی ارسال اطلاعات

```
val orderSummary = getString(  
    R.string.order_details,  
    sharedViewModel.quantity.value.toString(),  
    sharedViewModel.flavor.value.toString(),  
    sharedViewModel.date.value.toString(),  
    sharedViewModel.price.value.toString()  
)
```

- اطلاعات ارسالی:

```
val intent = Intent(Intent.ACTION_SEND)  
    .setType("text/plain")  
    .putExtra(Intent.EXTRA_SUBJECT, getString(R.string.new_cupcake_order))  
    .putExtra(Intent.EXTRA_TEXT, orderSummary)
```

```
<plurals name="cupcakes">  
    <item quantity="one">%d cupcake</item>  
    <item quantity="other">%d cupcakes</item>  
</plurals>
```

- ارسال اطلاعات در قصد ضمنی:

- پردازش درست جمع و مفرد

تابع ارسال اطلاعات

```
fun sendOrder() {  
    val numberOfCupcakes = sharedViewModel.quantity.value ?: 0  
    val orderSummary = getString(  
        R.string.order_details,  
        resources.getQuantityString(R.plurals.cupcakes, numberOfCupcakes, numberOfCupcakes),  
        sharedViewModel.flavor.value.toString(),  
        sharedViewModel.date.value.toString(),  
        sharedViewModel.price.value.toString()  
    )  
  
    val intent = Intent(Intent.ACTION_SEND)  
        .setType("text/plain")  
        .putExtra(Intent.EXTRA_SUBJECT, getString(R.string.new_cupcake_order))  
        .putExtra(Intent.EXTRA_TEXT, orderSummary)  
  
    if (activity?.packageManager?.resolveActivity(intent, 0) != null) {  
        startActivity(intent)  
    }  
}
```



سوال؟