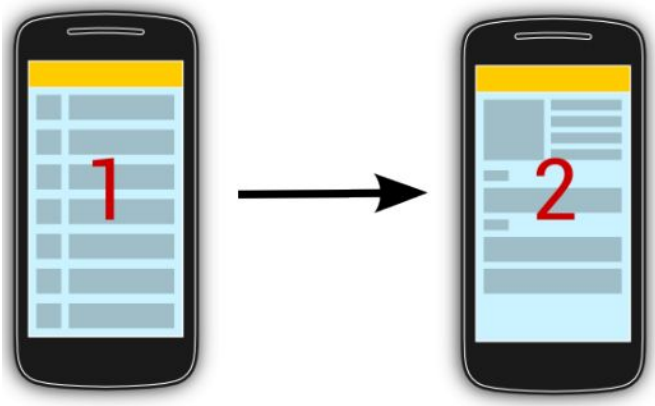
Домашнее задание

Цель задания:

В задании вы продолжите дорабатывать приложение, с которым работали на предыдущих уроках. Вы научитесь отображать фрагменты динамически, передавать аргументы внутрь фрагментов, работать с бекстеком, взаимодействовать с активити и использовать вложенные фрагменты.
В качестве дополнительного задания вам предстоит самостоятельно разобраться с паттерном master-detail и научиться отображать фрагменты с анимацией.

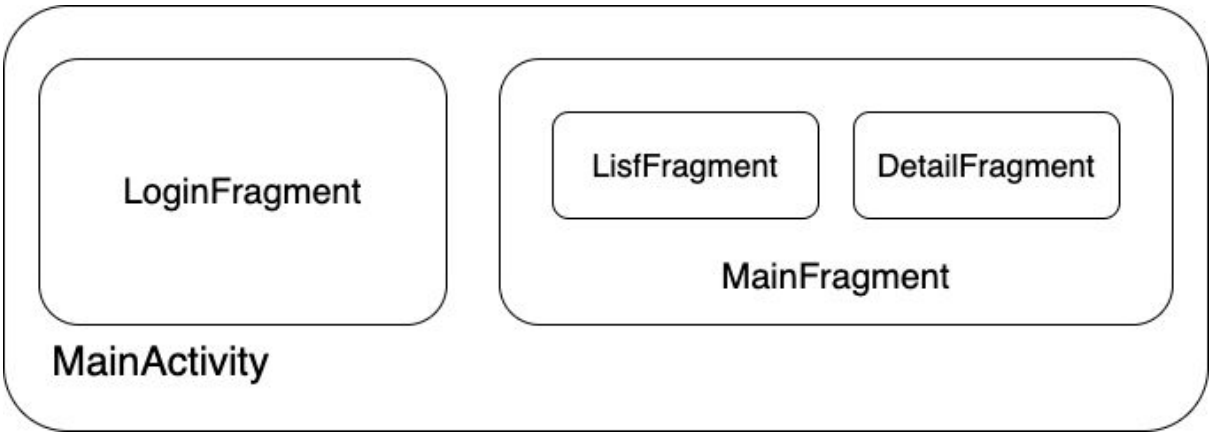
Что нужно сделать:

1. Создайте фрагмент LoginFragment на основе экрана логина из домашнего задания модуля “Жизненный цикл Activity”. Перенесите в него логику экрана логина из активити.
2. MainActivity должно выступать в качестве контейнера для фрагментов. Поэтому в разметке активити должен присутствовать только контейнер FrameLayout.
3. Используйте транзакции фрагментов, чтобы динамически менять их в активити. По запуску приложения в MainActivity должен открыться LoginFragment с формой логина. После успешного логина LoginFragment должен меняться на MainFragment.
4. MainFragment является родительским фрагментом для двух других фрагментов: списка (ListFragment) и детальной информации (DetailFragment). Для этого используйте вложенные фрагменты и childFragmentManager. При открытии MainFragment в его контейнер должен добавляться ListFragment со списком элементов.
5. По нажатию на элемент списка должен открыться фрагмент детальной информации. Также необходимо настроить обращение к родительскому фрагменту MainFragment и оповестить его, чтобы он открыл фрагмент детальной информации для выбранной сущности.



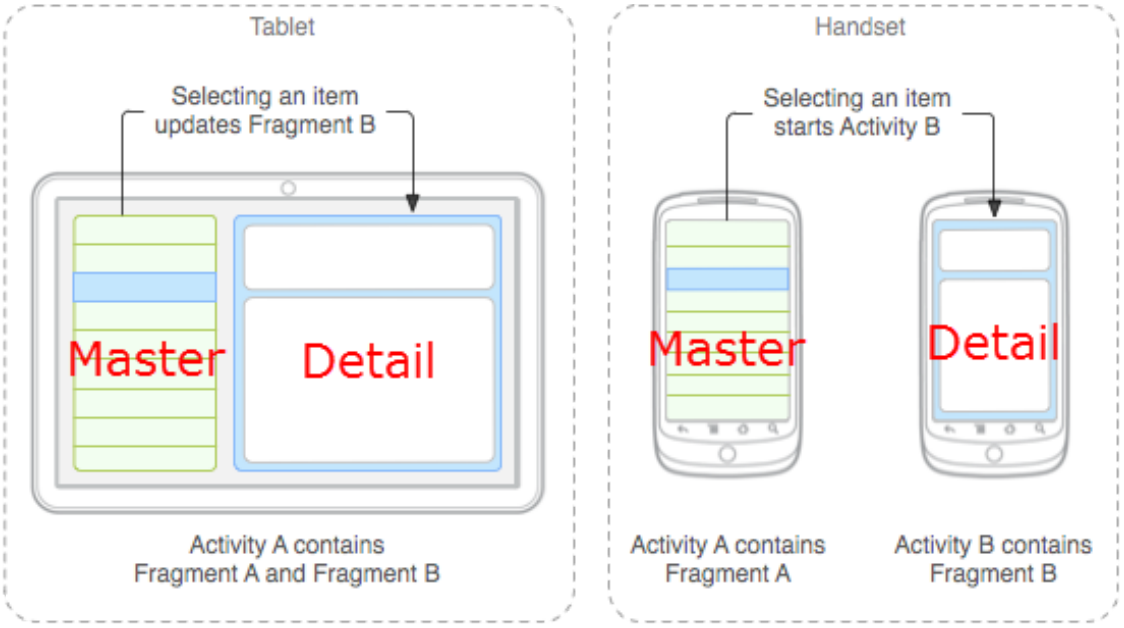
6. По нажатию кнопки «Назад» в DetailFragment приложение возвращается на ListFragment. По нажатию «Назад» на ListFragment происходит выход из приложения. Используйте fragment backstack.

В итоге должна получиться следующая структура:



Дополнительное задание:

1. Настройте анимацию при смене фрагментов.
2. Используйте паттерн master-detail, чтобы отобразить список элементов и детальную информацию на одном экране, если приложение запущено на планшете.
   Для этого::
   a. Создайте альтернативный ресурс разметки MainFragment для планшета.
   b. В ресурсе разметки планшета должен быть статический фрагмент с ListFragment в левой части экрана и контейнер для динамического фрагмента справа. Установите id динамического контейнера аналогично равным id контейнера в обычной вёрстке MainFragment из основной части домашнего задания.
   c. При запуске MainFragment проверяйте, запущено ли приложение на планшете. Если да — ничего не делайте, так как ListFragment будет отображен автоматически. Если нет — настройте открытие ListFragment, как в основной части домашнего задания.
   d. При нажатии на элемент списка должен открываться фрагмент детальной информации в правой части экрана. Этот пункт должен заработать автоматически при корректной реализации основной части домашнего задания и пунктов, указанных выше.



Рекомендации:

Используйте репозиторий **learning_materials / android_basic**
Скачайте изменения в репозитории на локальную машину.
Выполните домашнее задание в папке **Fragments.**
Перед выполнением скопируйте проект из папки **ActivityLifecycle,** после этого сделайте первый коммит, выполните домашнее задание и сделайте второй коммит (или несколько коммитов). Отправьте коммиты в удалённый репозиторий.

Критерии оценки:

1. Код оформлен в соответствии с правилами https://kotlinlang.org/docs/reference/coding-conventions.html.
2. Соблюдены принципы инкапсуляции с помощью модификаторов доступа.
3. Классы не являются финальными (open, abstract) только по необходимости.
4. Текстовые строки не являются захардкоженными и используются из ресурсов.
5. Фрагменты взаимодействуют друг с другом с помощью обращения к родительской сущности: активити или родительскому фрагменту.
6. При обращении к родительской сущности используется её приведение к интерфейсу, а не к конкретному классу.
7. Данные передаются во фрагмент с помощью аргументов.
8. Фрагмент с аргументами создаётся в функции внутри companion object класса фрагмента: fun newInstance(args): Fragment.