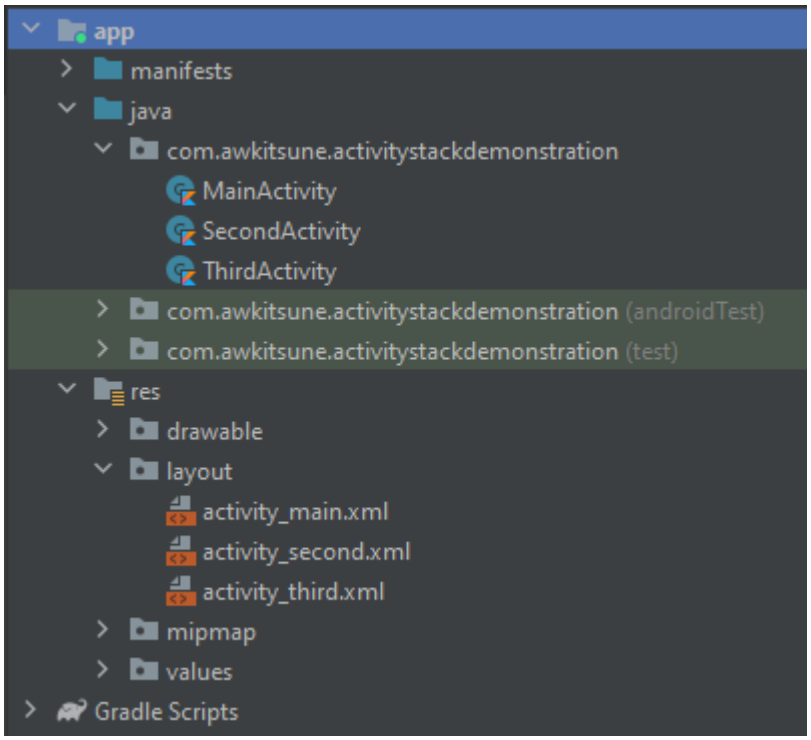
Лекция #21. Взаимодействие между Activity

В прошлых темах мы рассмотрели жизненный цикл **activity** и запуск новых **activity** с помощью объекта **Intent**. Теперь рассмотрим некоторые особенности взаимодействия между **activity** в одном приложении. Допустим, у нас есть три **activity**: **MainActivity**, **SecondActivity** и **ThirdActivity**.



С помощью **Intent**, например, по нажатию кнопки **MainActivity** запускает **SecondActivity**:

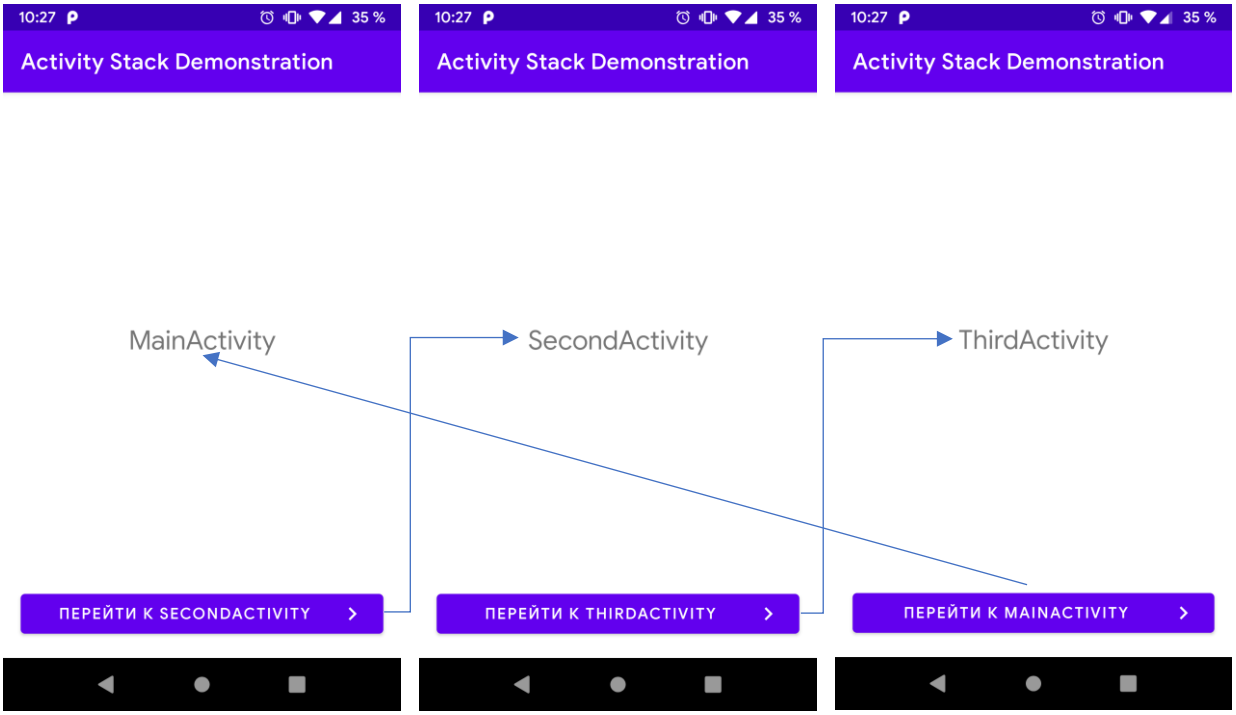
```
val intent: Intent =
        Intent(this@MainActivity, SecondActivity::class.java)
startActivity(intent)
```

На **SecondActivity** тоже есть кнопка, которая запускает **ThirdActivity**:

```
val intent: Intent =
       Intent(this@SecondActivity, ThirdActivity::class.java)
startActivity(intent)
```

На **ThirdActivity** также есть кнопка, которая возвращается к первой **activity** - **MainActivity**:

```
val intent: Intent =
       Intent(this@ThirdActivity, MainActivity::class.java)
startActivity(intent)
```



Если мы последовательно запустим все activity: из главной **MainActivity** запустим **SecondActivity**, из **SecondActivity** - ThirdActivity, то в итоге у нас сложится следующий стек activity:

1. **ThirdActivity**
2. **SecondActivity**
3. **MainActivity**

Если после этого из ThirdActivity мы захотим обратиться к MainActivity, то метод startActivity() запустит новый объект MainActivity (а не вернется к уже существующему), и стек уже будет выглядеть следующим образом:

1. **MainActivity**
2. **ThirdActivity**
3. **SecondActivity**
4. **MainActivity**

То есть у нас будут две независимые копии **MainActivity**. Такое положение нежелательно, если мы просто хотим перейти к существующей. И этот момент надо учитывать.

Если мы нажмем на кнопку **Back (Назад)**, то текущая **activity**, которая находится на вершине стека, удаляется из стека, и предыдущая activity оказывается на вершине стека и возобновляет свою работу. И таким образом с помощью кнопки **Back** (Назад) мы сможем перейти к предыдущей activity в стеке. Например, в случае выше если мы нажмем на кнопку Назад, то **MainActivity** на вершине стека завершает свою работу, и на экране начинает отображаться **ThirdActivity**

1. **ThirdActivity**
2. **SecondActivity**
3. **MainActivity**

Тем не менее иногда возникает необходимость упавлять переходом между **activity**. Например, в данном случае нам нежелательно при нажатии на кнопку в **ThirdActivity** запускать новую копию **MainActivity** вместо того, чтобы просто перейти к **MainActivity**, которая была запущена первой и находится в самом низу стека. Рассмотрим, какие возможности предоставляет нам **Android**.

Управление стеком activity

Для управления стеком из activity Android предлагает нам использовать флаги - константы, определенные в классе **Intent**. Применение определенного флага позволит нам определенным образом изменить положение в стеке для определенных **activity**.

Например, возьмем предыдущую задачу, когда после нажатия на кнопку в **ThirdActivity** запускается новый экземпляр **MainActivity**. Но мы хотим не запускать новую, а перейти к уже существующей.

1. **MainActivity**
2. **ThirdActivity**
3. **SecondActivity**
4. **MainActivity**

Чтобы выйти из этой ситуации, мы можем использовать флаг **Intent.FLAG_ACTIVITY_REORDER_TO_FRONT**:

```
val intent: Intent =
```

```
        Intent(this@ThirdActivity, MainActivity::class.java)
intent.addFlags(Intent.FLAG_ACTIVITY_REORDER_TO_FRONT)
startActivity(intent)
```

Флаг **Intent.FLAG_ACTIVITY_REORDER_TO_FRONT** перемещает activity, к которой осуществляется переход на вершину стека, если она ужее есть в стеке. И в этом случае после перехода из **ThirdActivity** к **MainActivity** стек будет выглядеть следующим образом:

1. **MainActivity**
2. **ThirdActivity**
3. **SecondActivity**

Если же нам просто надо перейти из ThirdActivity к MainActivity, как если бы мы перешли назад с помощью кнопки Back, то мы можем использовать флаги **Intent.FLAG_ACTIVITY_CLEAR_TOP** и **Intent.FLAG_ACTIVITY_SINGLE_TOP**:

```
val intent: Intent =
        Intent(this@ThirdActivity, MainActivity::class.java)
intent.addFlags(
        Intent.FLAG_ACTIVITY_CLEAR_TOP or
        Intent.FLAG_ACTIVITY_SINGLE_TOP)
startActivity(intent)
```

Флаг **Intent.FLAG_ACTIVITY_CLEAR_TOP** очищает все activity кроме той, которая запускается (если она уже есть в стеке). А флаг **Intent.FLAG_ACTIVITY_SINGLE_TOP** указывает, что если в вершине стеке уже есть activity, которую надо запустить, то она НЕ запускается (то она может существовать в стеке только в единичном виде).

В этом случае после перехода из ThirdActivity к MainActivity стек будет полностью очищен, и там останется одна MainActivity.

Еще один флаг - **Intent.FLAG_ACTIVITY_NO_HISTORY** позволит не сохранять в стеке запускаемую activity. Например, при запуске SecondActivity мы не хотим ее сохранять в стеке:

```
val intent: Intent =
        Intent(this@MainActivity, SecondActivity::class.java)
intent.addFlags(Intent.FLAG_ACTIVITY_NO_HISTORY)
startActivity(intent)
```

В этом случае при переходе по цепочке MainActivity -> SecondActivity -> ThirdActivity стек будет выглядеть следующим образом:

1. **MainActivity**
2. **ThirdActivity**