Второй курс, осенний семестр 2016/17
Конспект семинаров по программированию под Android

Собрано 15 октября 2016 г. в 20:41

Содержание

Глава #1
Activity
10 октября

1.1. Activity

Activity – это компонент приложения, который отвечает за взаимодействие с пользователем и представляет собой отдельное окно.

1.2. Жизненный цикл Activity

1. Resumed (Running) – можно взаимодействовать
2. Paused – видно, но взаимодействовать нельзя
3. Stopped – не видно

$$\cdot \xrightarrow{\text{onCreate()}} \text{Stopped} \xrightarrow{\text{onStart()}} \text{Paused} \xrightarrow{\text{onResume()}} \text{Resumed}$$
$$\text{Resumed} \xrightarrow{\text{onPause()}} \text{Paused} \xrightarrow{\text{onStop()}} \text{Stopped} \xrightarrow{\text{onDestroy()}} \cdot$$

1.3. Как создать Activity?

Нужно создать Java-класс, унаследованный от Activity. (Activity лежит в android.app.activity). Чтобы добавить Activity, нужно кликнуть правой кнопкой на манифест → New → Activity

MainActivity – та Activity, которая появляется при старте экрана.

1.4. Некоторые тонкости

1. В методах Activity нужно сперва вызывать этот метод у super.
2. При повороте экрана Activity уничтожается и создаётся с нуля.
3. Если из Activity вызвать другую Activity, то основная перейдёт в состояние Paused, а вторая создастся и перейдёт в состояние Stopped. Если же вернуться к основной Activity, то вторая будет уничтожена. Это происходит потому, что они находятся в одном task-е, который представляет собой стек.
4. Task ≠ приложение, в одном task-е могут лежать Activity из разных приложений.

Глава #2
Intent

10 октября

2.1. Что такое Intent и как его использовать

Intent (с английского - *"намерение"*) - класс, обеспечивающий универсальный способ общения различных частей приложения (или даже нескольких приложений).

Одно из частых применений Intent - использование его для создания Activity или Service (компонент, которые выполняют действия в фоновом режиме без пользовательского интерфейса). Для этого создаёте Intent с описанием того, что вам нужно сделать и передаёте в startActivity.

Также, в Intent может передаваться результат работы какого-то дочернего действия (подробнее об этом можно почитать в пункте про обратную связь).

2.2. Способы создания Intent и примеры использования

2.2.1. Явные объекты Intent

Явные объекты Intent используются для запуска конкретных компонентов приложения, например, определенной операции или службы. Чтобы создать явный объект Intent, жёстко задайте имя компонента. Для запуска одного из своих Activity рекомендуется использовать именно этот способ.

```
1 Intent intent = new Intent(this, MyActivity.class);
2 startActivity(intent);
```

Создаём Intent, в качестве контекста передаём себя. Вторым параметром передаём Activity, которое хотим создать

2.2.2. Неявные объекты Intent

Неявный объект Intent указывает действие, которым может быть вызвано любое имеющееся на устройстве приложение, способное выполнить это действие. Неявные объекты Intent используются, когда ваше приложение не может выполнить то или иное действие, зато какие-то другие приложения на устройстве, возможно, могут. При таком способе у вас нет гарантий, какое из приложений обработает ваш вызов.

Возможна ситуация, когда на устройстве пользователя не будет никакого приложения, которое может откликнуться на неявный объект Intent. В этом случае вызов закончится неудачей, а работа приложения аварийно завершится. Чтобы проверить наличие подходящего приложения, вызовите метод resolveActivity для своего объекта Intent. Если результатом будет значение, отличное от null, значит, имеется хотя бы одно приложение, которое способно откликнуться на объект Intent. Если же результатом будет null, объект Intent не следует использовать и по возможности следует отключить функцию, которая выдает этот объект Intent.

```
1 String uri = "geo:0,0?q=Saint-Petersburg";
2 Intent mapIntent = new Intent(Intent.ACTION_VIEW, Uri.parse(uri));
3 startActivity(mapIntent);
```

Создаём Intent. Первым параметром передаём тип действия, которое мы хотим. В нашем случае - показать пользователю какие-то данные. Дальнейшее происходящее зависит от второго параметра. Префикс "geo"говорит, что мы хотим показать место на карте. Например, этот Intent может обработать Google Maps.

Для некоторых действий необходимо заранее прописать в *AndroidManifest.xml* требование разрешения на это действие. Например, для обращение к книге контактов надо прописать:

```
1 <uses-permission android:name="android.permission.READ_CONTACTS" />
```

2.2.3. Передача дополнительной информации через Intent

Для передачи дополнительной информации можно использовать семейство методов putExtra (они перегружены для разных типов сохраняемых данных). Эти методы позволяют хранить данные по строковому ключу. Можно сохранить все примитивные типы, массивы примитивных типов, а также все объекты, наследующие Serializable. Обратно данные можно получить методами get[type]Extra.

2.2.4. Bundle

Класс Bundle представляет из себя нечто вроде HashMap, в котором по String можно сохранять сразу множество разных данных. Аналогично Intent, Bundle имеет методы putExtra и get[type]Extra. Через метод putExtras можно скопировать в Intent сразу все данные из Bundle. Также можно дополнительно передавать Bundle в startActivity и startActivityForResult.

2.2.5. Примеры использования

Больше примеров можно найти здесь

2.3. Обратная связь при запуске Activity

2.3.1. startActivityForResult

Если вы хотите получить какой-то результат работы Activity (например, вызываете менеджер файлов, чтобы пользователь выбрал какой-нибудь файл), то следует запускать его методом startActivityForResult. Кроме Intent надо передать ещё некоторое число requestCode. По нему можно будет понять, когда придёт результат выполнения. Подробнее об этом в пункте про onActivityResult.

2.3.2. Получение возвращаемого значения

После завершения дочерней Activity, запущенной через startActivityForResult, у вашего Activity будет вызван метод onActivityResult. Это метод принимает три параметра:

1. requestCode - число, которое было передано в startActivityForResult при запуске дочерней Activity
2. resultCode - код, с которым завершилась дочерняя Activity
3. data - Intent, в котором сохранена вся возвращаемая информация

2.3.3. Как вернуть результат?

Это делается методом setResult. В него надо передать код завершения и возвращаемый Intent. Обратите внимание, что эта команда не завершает работу Activity и, следовательно, не вызывает onActivityResult у родителя. Это произойдёт только после команды finish.

2.3.4. Пример

В основном Activity запускаем дочернюю Activity

```
1 private static final int MY_ACTIVITY_REQUEST = 7;
2
3 ...
4
5 Intent intent = new Intent(this, MyActivity.class);
6 intent.putExtra("data", "Hello world");
7 startActivityForResult(intent, MY_ACTIVITY_REQUEST);
```

В дочерней Activity при завершении работы

```
1 String s = getIntent().getStringExtra("data"); // "Hello world"
2
3 ...
4
5 Intent result = new Intent();
6 result.putExtra("res", "Hello!");
7 setResult(RESULT_OK, result);
8 finish();
```

В основном после этого вызовется

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == MY_ACTIVITY_REQUEST) {
        if (resultCode == RESULT_OK) {
            String s = data.getStringExtra("res"); // "Hello!"
        }
    }
}
```

2.4. Фильтры

2.4.1. Зачем?

Как было сказано выше, неявные объекты Intent не гарантируют, какое Activity их обработает. Как же определяется, кто может обработать конкретный Intent? Для этого и существуют Intent-фильтры. У каждого Activity может быть несколько фильтров. Intent может достаться Activity, если подходит хотя бы под один из её фильтров.

2.4.2. Как задать фильтр

Фильтры задаются в файле *AndroidManifest.xml*. Если вы уже создавали проекты в AndroidStudio, то могли видеть автоматически сгенерированный фильтр для главного Activity. Он выглядел примерно так:

```
1 <activity
2     android:name=".MainActivity"
3     android:label="@string/app_name"
4     android:theme="@style/AppTheme.NoActionBar">
5     <intent-filter>
6             <action android:name="android.intent.action.MAIN" />
7             <category android:name="android.intent.category.LAUNCHER" />
8     </intent-filter>
9 </activity>
```

Конкретно фильтр занимает тут строчки с 5 по 8

Разберём опции фильтра на другом примере:

```
1 <activity android:name="ShareActivity">
2     <intent-filter>
3         <action android:name="android.intent.action.SEND"/>
4         <category android:name="android.intent.category.DEFAULT"/>
5         <data android:mimeType="text/plain"/>
6     </intent-filter>
7 </activity>
```

Итак, вот что мы можем задавать:

1. action - определяет действие, которое мы умеем обрабатывать (здесь - отправлять сообщения)
2. category - категория, к которой относится наша Activity (текстовый редактор, плеер, браузер и т.п.)
3. data - тип данных, которые мы ожидаем получить (здесь - просто текст)

Для получения неявных объектов Intent необходимо включить категорию "android.intent.category.DEFAULT"в фильтр. Это связано с тем, что методы startActivity

и startActivityForResult работают с неявными Intent так, как будто они требуют указанную категорию. Стартовое Activity должно иметь фильтр с категорией "android.intent.category.LAUNCHER".