Заголовок

Автор

Дата или место проведения

Android Manifest

- Описание проекта, в том числе настройки и конфигурации, например версии.
- Прописаны permission и составляющие проекта.

Activity

- Первой в программе вызывается MainActivity(? не помню).
- Цикл жизни.
- Activity надо прописывать в манифесте.

BroadCastReceiver

- Один из стандартных классов андроид, завязан на несколько(может один) intent.
- Получает интент, к которому он привязан и быстро реагирует(с технической точки зрения при получении intent вызывается onReceive(), который должен быстро выполняться).
- Если один BroadCastReceiver соотетствует нескольким intent, то они построятся в цепочку и будут выполняться последовательно.
- Поэтому если по какому-то intent метод выполняется слишком долго, то его прикончат, чтобы другие intent не ждали.
- Так что обычно, его используют только чтобы послать какой-то другой intent или получить системное оповещение.
- Receiver должен быть прописан в Android манифесте.

BroadCastReceiver

- Подписку на intent можно тоже записать в Manifest.

```
1  <receiver android:name=".LocationChangedReceiver" />
```

- А можно это сделать и програмно. Explicit intent - явно указываем класс, к которому intent отправить. Бывает ещё inplicit intent. На самом деле здесь посылается не intent а наследник класса Intent - PendingIntent, в котором учитываются права приложения, из которого нас запустили.
- Intent - это такой способ общаться между частями приложения. В него можно записать дополнительную информацию.

```
1  startService.putExtra(LocationManager.
       KEY_LOCATION_CHANGED, location);
```

- Доставать информацию из Intent можно так:

```
1  Location location = (Location) intent.getExtras().
       get(locationKey);
```

- Класс Context - это базовый класс для частей приложения. К примеру, Activity - наследник класса Context. Он предоставляет доступ ко всем андроидовским ресурсам. BroadCastReceiver не является наследником класса Context, поэтому он туда передаётся

BroadCastReceiver

Пример onReceive():

```
1   @Override
2   public void onReceive(Context context, Intent intent) {
3       final String locationKey = LocationManager.KEY_LOCATION_CHANGED;
4
5       if (intent.hasExtra(locationKey)) {
6           Location location =
7                   (Location) intent.getExtras().get(locationKey);
8
9           Intent startService =
10                      new Intent(context, ForecastUpdateService.class);
11          startService.putExtra(LocationManager.KEY_LOCATION_CHANGED,
                location);
12          context.startService(startService);
13      }
14  }
```

Log

- Записи в log выводятся на экран Logcat.
- Логирование происходит так:

```
1       ...
2  private static final String TAG =
3                  SMSReceiver.class.getSimpleName();
4       ...
5  Log.d(TAG, "SMS received");
```

Permissions

- Чтобы работать с сетью надо подключить соотетсвующие permission.

```
1  <uses-permission android:name="android.permission.
       INTERNET" />
2  <uses-permission  android:name="android.permission.
       ACCESS_NETWORK_STATE"/>
```

Service

- Service также надо прописывать в манифесете:

```
1  <service
2       android:name=".DatabaseService"
3       android:exported="false" >
4  </service>
```

- Для того, чтобы делать продолжительные действия BroadCastReceiver не подходит. Для этого нужно использовать Service.
- Ключик exported задаёт, позволено ли другим приложениям запускать наш сервис, Activity и тд.

Service

- Если к нему обращаются много intent, они также становятся в очередь.
- Запускают intent к service через команду startService(...);
- При обработке intent запускается метод onHandleIntent(Intent intent);

Connection

- Самый простой способ работать с сетью - это HttpURLConnection.
- Возвращается методом .openConnection():

```
1  URL url = new URL(uri.toString());
2  connection =
3          (HttpURLConnection) url.openConnection();
```

- Протокол http может обрабатывать различные запросы. Например, запрос на получение информации. Если мы хотим получить информацию - пишем следующее:

```
1  connection.setRequestMethod("GET");
2  connection.connect();
```

Connection

Пример считывания информации:

```
1   InputStream inputStream = connection.getInputStream();
2   StringBuilder buffer = new StringBuilder();
3   if (inputStream == null) {
4       return;
5   }
6   reader = new BufferedReader(new InputStreamReader(inputStream));
7   String line;
8   while ((line = reader.readLine()) != null) {
9       buffer.append(line);
10      buffer.append("\n");
11  }
12
13  if (buffer.length() == 0) {
14      return;
15  }
```