### Мобильная разработка

**Заниятие 16. Неявные объекты Intent**

**AndroidManifest.xml**

Файл AndroidManifest.xml – предоставляет подробную информацию о приложении. В каждом Android приложении вы должны в увидите этот файл, так как он является обязательным для создания приложения на ОС Android.

Этот файл автоматически создается при создании вашего проекта, он также автоматически редактируется, если вы, например, добавляете новую активность в ваш проект. Но редактировать файл манифеста можно и нужно в некоторых ситуациях вручную, изменяя XML-код.

Среди прочего в файле манифеста указывается следующая важная информация:

1. Указать имя Java-пакета приложения, который служит уникальным идентификатором;

2. Описать компоненты приложения, службы;

3. Указать список необходимых разрешений для обращения к защищенным частям API и взаимодействия с другими приложениями;

4. Разрешить доступ к ресурсам, которые сторонние приложения обязаны иметь для взаимодействия с компонентами данного приложения;

5. Указать минимальный и максимальный уровень API Android, необходимый для работы приложения.

Посмотрим наш файл манифеста. Первая активность (MainActivity), открывающаяся при старте приложения, является главной точкой входа. Действие «android.intent.action.MAIN» говорит о том, что это основная точка входа в приложение и не требуется ожидания никаких объектов Intent. Категория «android.intent.category.LAUNCHER» указывает на то, что значок данной активности следует поместить в средство запуска приложений системы. Если элемент <activity> не содержит указаний на конкретный значок с помощью тега icon, то система воспользуется значком из элемента манифеста <application>.

Если есть еще активности в нашем проекте, то они обязательно будут прописаны в этом файле, но по умолчанию без каких-либо параметров.

**Объекты URI, URN, URL**

Вы когда-нибудь обращали внимание на адресную строку в Вашем браузере? Что это? Адресб ссылка, URL? А что такое URI, URL или URN? Многие из нас не делают различий между URI, URL, URN, а кое-кто даже и не слышал терминов URI и URN, все просто пользуются термином URL.

### Определение

URI: Обозначает имя и адрес ресурса в сети. Как правило, делится на URL и URN, поэтому URL и URN это составляющие URI.

URL: Адрес некоторого ресурса в веб. URL определяет местонахождение ресурса и способ обращения к нему.

URN: Имя некоторого ресурса в веб. Смысл URN в том, что он определяет только название конкретного предмета, который может находится во множестве конкретных мест.

Итак, мы можем считать, что:

URI = URL или URI = URN или URI = URL + URN

Пример:

URI = http://handynotes.ru/2009/09/uri-url-urn.html

URL = http://myitschool.ru

URN = /book/mod/book/view.php?id=591&chapterid=397

В Java есть класс, который позволяет работать со строками, которые представляют собой ссылки на какие-то данные — интернет-ресурс или файл на компьютере. Строка, должна быть составлена в соответствии с этим документом:

<http://www.faqs.org/rfcs/rfc2396.html>

Этот класс Uri берет строку, разбирает ее на составляющие и хранит в себе эту информацию. Класс Uri имеет много методов, которые позволяют извлекать из распарсенной строки отдельные элементы.

Создадим проет и попробуем вывести поработать с классом Uri. Например возьмем такую строку:

Uri uri = Uri.parse("http://developer.android.com/reference/android/net/Uri.html");

И посмотрим, что нам возвращают методы:

uri.getScheme() - http  
uri.getSchemeSpecificPart() - //developer.android.com/reference/android/net/Uri.html  
uri.getAuthority() - developer.android.com  
uri.getHost() - developer.android.com  
uri.getPath() - reference/android/net/Uri.html  
uri.getLastPathSegment() - Uri.html

Понятия Scheme, Authority, Host, Path и пр. – взяты из RFC документа по ссылке выше. Там можно найти их полное описание, понять что они означают и свериться с тем, что нам вернул Uri.

Допустим, мы хотим загрузить видеоматериал в компонент VideoView. Само видео может находиться в ресурсах программы или на SD-карте. С помощью URI мы можем подсказать программе, откуда следует загрузить файл.

Например, если видеофайл “play.3gp» находится в папке /res/raw, то получить адрес для загрузки можно следующим образом:

Uri uri = Uri.parse("android.resource://your.app.package/" + R.raw.play);

Если файл хранится на внешней карточке, то код будет следующим (опустим правильное определение имени карточки):

Uri uri = Uri.parse("file:///sdcard/playcat.3gp"); // файл в корне карточки

С помощью объекта Uri можно не только хранить ссылки на данные, но и записывать какие-то конкретные данные, это позволяет хранить данные в удобном формате и быстро их извлекать.

Таки образом с помощью класса Uri можно определить и координаты:

Uri uri = Uri.parse("geo:54.354183,37.34011");

В этом случае метод uri.getScheme() вернёт geo, а метод uri.getSchemeSpecificPart() - 54.354183,37.34011.

Или номер телефона:

U*ri uri = Uri.parse("tel:12345");*

uri.getScheme(): tel  
uri.getSchemeSpecificPart():12345

**Неявные намерения**

Неявные объекты Intent не содержат имени конкретного класса. Вместо этого они включают действие (action), которое требуется выполнить. Неявный Intent может включать дополнительно наименование категории (category) и тип данных (data). Такой набор параметров позволяют компоненту из другого приложения обработать этот запрос.

Отличным примером служит открытие ссылки из мессенжера. При нажатии на ссылку формируется намерение открыть в браузере страницу с адресом, указанным в ссылке. Если на телефоне установлен только один браузер, то ссылка откроется именно через него. Но если на телефоне несколько браузеров, то система автоматически предложит вам выбрать именно тот, в котором вы хотите открыть ссылку.

Таким образом, при создании объекта Intent вместо явного указания класса компоненты заполняются определенными значениями следующие параметры — действие, категория, данные. Комбинация этих значений определяют цель, которую нужно достичь. Например: отправка письма, распознавание текста, просмотр картинки, телефонный звонок и т. д.

Когда android получает неявный объект Intent для выполнения, то система ищет подходящие компоненты путем сравнения содержимого Intent с фильтрами Intent других приложений, зарегистрированных в файлах манифестов. Если параметры объекта Intent совпадают с параметрами одного из фильтров Intent, то система запускает этот компонент и передает ему объект Intent. Следует отметить, что поиск идет по всем Activity всех приложений в системе. Если нашлась только одна активность, подходящая по условиям, то она и запускается. Если находится несколько соответствующих условию активностей, то система предоставляет вам выбор, какой именно программой вы хотите воспользоваться.

## Действие намерения, action

Действие action объекта Intent определяет, что нужно выполнить, например просмотр фото (view) или выбор фото (pick). В значительной степени действие определяет, каким образом описана остальная часть намерения Intent, в частности, что именно содержится в разделе данных. Действие для объекта Intent можно указать методом setAction() или определить в конструкторе Intent.

В таблице предствлены несколько часто используемых действий с описанием:

|  |  |
| --- | --- |
| ACTION\_ANSWER | Открывает активность, связанную с входящими звонками. |
| ACTION\_CALL | Открывает активность, инициализирующую обращение к телефону. |
| ACTION\_HEADSET\_PLUG | Подключение наушников. |
| ACTION\_SEND | Вызов активности для отправки данных, указанных в намерении. Получатель должен быть определен с помощью полученной активности. Для указания типа передаваемых данных MIME используйте метод setType. Данные передаются параметром намерения extras с ключами EXTRA\_TEXT или EXTRA\_STREAM в зависимости от типа. При использовании электронной почты стандартное приложение принимает дополнительные параметры по ключам EXTRA\_EMAIL, EXTRA\_CC, EXTRA\_BCC и EXTRA\_SUBJECT. Действие ACTION\_SEND следует использовать только в тех случаях, когда данные необходимо передать удаленному адресату, а не другой программе на том же устройстве. |
| ACTION\_SENDTO | Открывает активность для отправки сообщений контакту, указанному в пути URI, который определяется в намерении. |
| ACTION\_VIEW | Наиболее распространенное общее действие просмотра чего-либо. Выбор приложения зависит от схемы (протокола) данных : стандартные адреса http будут открывать браузер, адреса tel вызовут приложение для дозвона, geo откроет программу Google Maps, а данные с контактами откроют приложении для управления контактной информацией. |
| ACTION\_WEB\_SEARCH | Открывает активность поиска информации в интернете, основываясь на передаваемых данных с помощью пути URI; как правило, при этом запускается браузер. |

Рассмотрим простой пример формирования неявного намерения для открытия определенной страницы сайта :

*Intent intent = new Intent(Intent.ACTION\_VIEW, Uri.parse("http://developer.android.com"));  
if (intent.resolveActivity(getPackageManager())!=null)  
 startActivity(intent);  
else  
 Toast.makeText(getApplicationContext(), "Intent (ACTION\_VIEW) не обработан!", Toast.LENGTH\_LONG).show();*

В данном примере создается намерение с действием ACTION\_VIEW, означающим просмотр чего-либо. При вызове данного намерения текущая активность приостанавливает своё действие и переходит в фоновый режим. При нажатии пользователем кнопки Back исходная активность восстанавливется.

В случае, если система не сможет выполнить неявное намерение при вызове метода startActivity или startActivityForResult, то приложение завершится с ошибкой. Чтобы не допустить этого следует использовать метод намерения resolveActivity:

*if (intent.resolveActivity(getPackageManager())!=null)*

*startActivity(intent);*

Необходимо понимать, что все приложения в системе заточены под конкретные действия с конкретными типами данных. И если вы попробуете позвонить на адрес сайта, или открыть на карте номер телефона – то система просто не найдет приложения, способные на это.

## Фильтры намерений Intent в файле AndroidManifest.xml

Чтобы вызвать активность любого приложения, не важно ваше это приложение или стандартное, для выполнения какого-либо действия, у этой активности в файле AndroidManifest.xml должны быть указаны фильтры.

Фильтр Intent представляет собой секцию в файле манифеста приложения, описывающее типы объектов Intent, которые компонент мог бы выполнить. Таким образом, наличие фильтра Intent в описании активности в манифесте позволяет другим приложениям напрямую запускать данную операцию с помощью некоторого объекта Intent. Если фильтр Intent не описан, то операцию можно будет запустить только с помощью явного объекта Intent.

В секциях фильтров намерений в файле манифеста можно объявить только три составляющих объекта Intent : действие, данные, категория.

Про действия мы уже поговорили выше, остается отметить только то, что действие, определяемое в манифесте, включает наименование пакета и не использует префикс "ACTION\_", например, «android.intent.action.VIEW». С другими константными действиями можно познакомиться в справочнике:

<https://developer.android.com/reference/android/content/Intent.html>.

## Категория намерения, category

Категория – это строка, содержащая дополнительные сведения о том, каким компонентом должна выполняться обработка объекта Intent. В объект Intent можно поместить любое количество категорий. Однако большинству объектов Intent описание категории не требуется.

Класс Intent для работы с категориями имеет группу методов :

* addCategory() — добавить категорию в объект Intent;
* removeCategory() — удалить ранее добавленную категорию из объекта Intent;
* getCategories() — получить набор категорий объекта Intent.

При описании категории в файле манифеста используются стандартные значения, предоставляемые системой :

|  |  |
| --- | --- |
| BROWSABLE | Операция стартует веб-браузер для отображения данных, указанных по ссылке (url), например, изображение или сообщение электронной почты. |
| LAUNCHER | Операция (активность) является начальной операцией задачи. Активность с данной категорией помещается в окно для запуска приложений. |
| DEFAULT | Данная категория позволяет объявить компонент обработчиком по умолчанию для действия, выполняемого с указанным типом данных внутри фильтра намерений. |
| HOME | Активность с данной категорией отображает главный экран (Home Screen), который открывается после включения устройства и загрузки системы, или когда нажимается клавиша HOME. |

Константные наименования категории, используемые в java-коде, имеют префикс "CATEGORY\_". С другими константными категориями можно познакомиться в справочнике: <https://developer.android.com/reference/android/content/Intent.html> .

## Данные намерения

При создании объекта Intent в некоторых случая необходимо определить данные (data). Например, для действия ACTION\_EDIT, данные должны включать URI документа, который требуется отредактировать. Объект Uri ссылается на данные и/или тип этих данных MIME. Так, например, операция, которая выводит на экран изображения, скорее всего, не сможет воспроизвести аудио/видео файл, даже если и у тех, и у других данных будут одинаковые форматы URI. Поэтому указание типа данных MIME помогает системе Android найти наиболее подходящий компонент для выполнения объекта Intent.

Чтобы задать только URI данных, используйте метод setData(). Чтобы определить только тип MIME, вызовите setType(). Чтобы определить URI и тип MIME следует использовать метод setDataAndType(). Не вызывайте методы setData() и setType() поочередно, поскольку каждый из этих методов аннулирует результат выполнения другого.

В файле манифеста данные в фильтре определяются в виде атрибутов тега <data>. При необходимости можно определить несколько тегов data. Чтобы указать, какие именно данные поддерживает компонент, используются следующие атрибуты :

* android:host – определение имени удалённого сервера, например, google.com;
* android:port – определение порта удалённого сервера;
* android:path – определение пути URI;
* android:mimetype – определение типа данных, которые компонент может обрабатывать;
* android:scheme – необходимо использовать конкретную схему (например, content или http).

#### Дополнительные данные

Данные, содержащие дополнительную информацию, необходимую для выполнения запрошенного действия, определяются в виде пары "ключ-значение". Добавлять дополнительные данные можно с помощью различных методов putExtra(), каждый из которых принимает два параметра : имя и значение ключа. Также можно создать объект Bundle с дополнительными данными и вставить его в Intent с помощью метода putExtras(). Например, для отправки сообщения электронной почты адрес получателя можно указать с помощью ключа EXTRA\_EMAIL, а тему сообщения ― с помощью ключа EXTRA\_SUBJECT.

Класс Intent имеет несколько констант EXTRA\_\* для стандартных типов данных. Если необходимо использовать собственные дополнительные ключи для объектов Intent, которые принимает только ваше приложение, обязательно указывайте в качестве префикса имя пакета своего приложения. Например :

static final String EXTRA\_WATT = "com.example.EXTRA\_WATT";

Создадим вторую активность Main2Activity. Несмотря на то, что пользователи могут входить в эту активность из MainActivity, они также могут открыть Main2Activity напрямую из другого приложения, которое создаст неявный объект Intent, с параметрами, соответствующими параметрам Main2Activity.

Попробуем добавить нашу вторую активность с список браузеров. Для этого в файле AndroidManifest.xml добавим секцию для описания намерений, с помощью которых можно открыть нашу вторую активность:

*<intent-filter>*

*<action android:name="android.intent.action.VIEW"></action>  
<category android:name="android.intent.category.DEFAULT" />  
<category android:name="android.intent.category.BROWSABLE"/>  
<data android:scheme="http"></data>*

*</intent-filter>*

## Определение намерения, запустившего активность

При старте активности с помощью неявного намерения необходимо определить действие, которое следует выполнить, и данные для этого действия. Метод getIntent активности позволяет получить объект намерения :

*@Override*

*public void onCreate(Bundle savedInstanceState)*

*{*

*super.onCreate(savedInstanceState);*

*setContentView(R.layout.main);*

*Intent intent = getIntent();*

*}*

Методы getAction() и getData() позволяют определить действие и данные, связанные с намерением. Для получения дополнительной информации, хранящейся в параметре extras, используйте типизированные методы get<Type>Extra.

*String action = intent.getAction();*

*Uri data = intent.getData();*

Реализуем во второй активности простейший браузер

*@Override  
protected void onCreate(Bundle savedInstanceState) {  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.activity\_main2);  
 Uri uri = getIntent().getData();  
 String url = "http://stackoverflow.com";  
 WebView wv = findViewById(R.id.webView);  
 wv.setWebViewClient(new WebViewClient(){  
 @Override  
 public boolean shouldOverrideUrlLoading(WebView view, String url)  
 {  
 view.loadUrl(url);  
 return true;  
 }  
 });  
  
 WebSettings set = wv.getSettings();  
 set.setJavaScriptEnabled(true);  
 wv.loadUrl(uri.toString());  
}*