**Intent Android**

**Содержание**

* [1 Intent (Намерение)](http://doc.artofweb.ru/doc/Intent_Android#Intent_.28.D0.9D.D0.B0.D0.BC.D0.B5.D1.80.D0.B5.D0.BD.D0.B8.D0.B5.29)
  + [1.1 Неявные намерения и связывание во время выполнения программы](http://doc.artofweb.ru/doc/Intent_Android#.D0.9D.D0.B5.D1.8F.D0.B2.D0.BD.D1.8B.D0.B5_.D0.BD.D0.B0.D0.BC.D0.B5.D1.80.D0.B5.D0.BD.D0.B8.D1.8F_.D0.B8_.D1.81.D0.B2.D1.8F.D0.B7.D1.8B.D0.B2.D0.B0.D0.BD.D0.B8.D0.B5_.D0.B2.D0.BE_.D0.B2.D1.80.D0.B5.D0.BC.D1.8F_.D0.B2.D1.8B.D0.BF.D0.BE.D0.BB.D0.BD.D0.B5.D0)
  + [1.2 Константы действия](http://doc.artofweb.ru/doc/Intent_Android#.D0.9A.D0.BE.D0.BD.D1.81.D1.82.D0.B0.D0.BD.D1.82.D1.8B_.D0.B4.D0.B5.D0.B9.D1.81.D1.82.D0.B2.D0.B8.D1.8F)
  + [1.3 Методы](http://doc.artofweb.ru/doc/Intent_Android#.D0.9C.D0.B5.D1.82.D0.BE.D0.B4.D1.8B)
  + [1.4 Группы намерений](http://doc.artofweb.ru/doc/Intent_Android#.D0.93.D1.80.D1.83.D0.BF.D0.BF.D1.8B_.D0.BD.D0.B0.D0.BC.D0.B5.D1.80.D0.B5.D0.BD.D0.B8.D0.B9)
  + [1.5 Фильтры намерений и запуск заданий](http://doc.artofweb.ru/doc/Intent_Android#.D0.A4.D0.B8.D0.BB.D1.8C.D1.82.D1.80.D1.8B_.D0.BD.D0.B0.D0.BC.D0.B5.D1.80.D0.B5.D0.BD.D0.B8.D0.B9_.D0.B8_.D0.B7.D0.B0.D0.BF.D1.83.D1.81.D0.BA_.D0.B7.D0.B0.D0.B4.D0.B0.D0.BD.D0.B8.D0.B9)
  + [1.6 Принцип работы фильтров намерений](http://doc.artofweb.ru/doc/Intent_Android#.D0.9F.D1.80.D0.B8.D0.BD.D1.86.D0.B8.D0.BF_.D1.80.D0.B0.D0.B1.D0.BE.D1.82.D1.8B_.D1.84.D0.B8.D0.BB.D1.8C.D1.82.D1.80.D0.BE.D0.B2_.D0.BD.D0.B0.D0.BC.D0.B5.D1.80.D0.B5.D0.BD.D0.B8.D0.B9)
  + [1.7 Делегирование ответственности](http://doc.artofweb.ru/doc/Intent_Android#.D0.94.D0.B5.D0.BB.D0.B5.D0.B3.D0.B8.D1.80.D0.BE.D0.B2.D0.B0.D0.BD.D0.B8.D0.B5_.D0.BE.D1.82.D0.B2.D0.B5.D1.82.D1.81.D1.82.D0.B2.D0.B5.D0.BD.D0.BD.D0.BE.D1.81.D1.82.D0.B8)
    - [1.7.1 ACTION\_PICK](http://doc.artofweb.ru/doc/Intent_Android#ACTION_PICK)
* [2 Примеры использования намерений](http://doc.artofweb.ru/doc/Intent_Android#.D0.9F.D1.80.D0.B8.D0.BC.D0.B5.D1.80.D1.8B_.D0.B8.D1.81.D0.BF.D0.BE.D0.BB.D1.8C.D0.B7.D0.BE.D0.B2.D0.B0.D0.BD.D0.B8.D1.8F_.D0.BD.D0.B0.D0.BC.D0.B5.D1.80.D0.B5.D0.BD.D0.B8.D0.B9)
  + [2.1 Позвонить](http://doc.artofweb.ru/doc/Intent_Android#.D0.9F.D0.BE.D0.B7.D0.B2.D0.BE.D0.BD.D0.B8.D1.82.D1.8C)
  + [2.2 Получить контакт из телефонной книги](http://doc.artofweb.ru/doc/Intent_Android#.D0.9F.D0.BE.D0.BB.D1.83.D1.87.D0.B8.D1.82.D1.8C_.D0.BA.D0.BE.D0.BD.D1.82.D0.B0.D0.BA.D1.82_.D0.B8.D0.B7_.D1.82.D0.B5.D0.BB.D0.B5.D1.84.D0.BE.D0.BD.D0.BD.D0.BE.D0.B9_.D0.BA.D0.BD.D0.B8.D0.B3.D0.B8)
  + [2.3 Открыть ссылку в браузере](http://doc.artofweb.ru/doc/Intent_Android#.D0.9E.D1.82.D0.BA.D1.80.D1.8B.D1.82.D1.8C_.D1.81.D1.81.D1.8B.D0.BB.D0.BA.D1.83_.D0.B2_.D0.B1.D1.80.D0.B0.D1.83.D0.B7.D0.B5.D1.80.D0.B5)
  + [2.4 YouTube](http://doc.artofweb.ru/doc/Intent_Android#YouTube)
  + [2.5 Поисковый запрос](http://doc.artofweb.ru/doc/Intent_Android#.D0.9F.D0.BE.D0.B8.D1.81.D0.BA.D0.BE.D0.B2.D1.8B.D0.B9_.D0.B7.D0.B0.D0.BF.D1.80.D0.BE.D1.81)
  + [2.6 Отправить письмо](http://doc.artofweb.ru/doc/Intent_Android#.D0.9E.D1.82.D0.BF.D1.80.D0.B0.D0.B2.D0.B8.D1.82.D1.8C_.D0.BF.D0.B8.D1.81.D1.8C.D0.BC.D0.BE)
  + [2.7 Камера](http://doc.artofweb.ru/doc/Intent_Android#.D0.9A.D0.B0.D0.BC.D0.B5.D1.80.D0.B0)
  + [2.8 Голосовая команда](http://doc.artofweb.ru/doc/Intent_Android#.D0.93.D0.BE.D0.BB.D0.BE.D1.81.D0.BE.D0.B2.D0.B0.D1.8F_.D0.BA.D0.BE.D0.BC.D0.B0.D0.BD.D0.B4.D0.B0)
  + [2.9 Запустить Twitter](http://doc.artofweb.ru/doc/Intent_Android#.D0.97.D0.B0.D0.BF.D1.83.D1.81.D1.82.D0.B8.D1.82.D1.8C_Twitter)
  + [2.10 Проиграть MP4-файл (ACTION\_VIEW)](http://doc.artofweb.ru/doc/Intent_Android#.D0.9F.D1.80.D0.BE.D0.B8.D0.B3.D1.80.D0.B0.D1.82.D1.8C_MP4-.D1.84.D0.B0.D0.B9.D0.BB_.28ACTION_VIEW.29)

Intent (Намерение)

Намерение (Intent) - это механизм для описания одной операции - выбрать фотографию, отправить письмо, сделать звонок, запустить браузер и перейти по указанному адресу. В Android-приложениях почти все работает благодаря намерениям.

Наиболее распространенный сценарий использования намерения - запуск другой активности в своём приложении.

Но это не единственный вариант использования намерения. Также можно использовать для объявления о запуске активности или сервиса, направленных на выполнение каких-либо действий (как правило, речь о работе с определенной частью данных) или для передачи уведомлений о том, что произошло некое событие (или действие).

Намерения могут применяться для трансляции сообщений по системе. Любое приложение способно зарегистрировать широковещательный приёмник и отслеживать эти намерения с возможностью на них реагировать. Это позволяет создавать приложения, использующие событийную модель, в основе которой лежат внутренние, системные или сторонние события, передаваемые внешними программами.

Android транслирует намерения для объявления о системных событиях, например об изменениях в состоянии сетевого подключения или в уровне заряда батареи. Системные приложения в Android, такие как программы дозвона или управления SMS, регистрируют компоненты, отслеживающие заданные намерения, например входящий звонок или получено новое SMS-сообщение, и соответствующим образом реагируют на них.

Неявные намерения и связывание во время выполнения программы

Неявные намерения — это механизм, позволяющий запрашивать анонимные компоненты приложений с помощью действий. Вы можете попросить систему запустить активность, выполняющую заданное действие, не зная ничего ни о самой активности, ни о её приложении.

Создавая новое неявное намерение для передачи в метод startActivity, необходимо назначить действие, которое должно выполниться, а также при желании указать вспомогательный путь URI к тем данным, что нужно обработать. Вы также можете передать дополнительные данные в целевую Активность, используя параметр намерения extras.

При применении этого намерения для запуска активности система во время работы приложения сама найдёт класс компонента, который лучше всего подходит для заданного действия, учитывая указанный тип данных. Это значит, что вы можете создавать проекты, используя возможности других приложений и не зная при этом, как они называются и какую функциональность предоставляют.

Простой пример создания намерения с предопределенным действием ACTION\_VIEW для запуска браузера и перехода на нужный адрес:

Uri address = Uri.parse("[http://developer.alexanderklimov.ru](http://developer.alexanderklimov.ru/)"); Intent openlink = new Intent(Intent.ACTION\_VIEW, address); startActivity(openlink);

В данном случае действие ACTION\_VIEW означает просмотр чего-либо. Мы указали нужные данные (адрес), и происходит запуск новой активности (браузера). При этом исходная активность приостанавливается и переходит в фоновый режим. Когда пользователь нажимает на кнопку Back, то он возвращается к исходной активности.

В каждом случае Android находит соответствующую активность, чтобы ответить на намерение, инициализируя ее в случае необходимости.

Приложения могут также создавать свои собственные типы намерений и позволять другим приложениям вызывать.

Объект Intent содержит информацию, представляющую интерес для компонента, который получает намерение, и данные, которые передаются этому компоненту. Кроме того, объект Intent содержит информацию, представляющую интерес для системы Android, — имя компонента, который должен обработать намерение и набор параметров запуска этого компонента.

Как правило, объект Intent может содержать следующие поля:

* Имя компонента - имя, который должен **обработать намерение**. Используется объект ComponentName, который является комбинацией полного имени класса целевого компонента (например, "MainActivity") и набора имени пакета в файле манифеста приложения, где компонент постоянно находится (например, "com.samples.yourproject"). Составляющее имя является дополнительным. Если оно установлено, объект Intent поставляет образцу определяемого класса. Если имя не установлено, Android использует другую информацию в объекте Intent, чтобы определить местонахождение подходящего адресата. Составляющее имя устанавливается методами setComponent(), setClass() или setClassName() и читается методом getComponent();
* Действие - определяет действие, которое **будет выполнено**. Класс Intent содержит множество констант действия. Название метода определяет ряд параметров и возвращаемое значение. Вы можете также определить собственные действия для активизации активности. В этом случае вы должны включать имя пакета приложения в качестве префикса, например com.samples.yourproject.CUSTOM\_ACTiON. Действие в объекте Intent устанавливается в методе setAction() и читается методом getAction();
* Данные - это **URI данных** и тип MIME для этих данных. Разные активности соединены с разными видами спецификаций данных.
* Категория - строка, содержащая дополнительную информацию о **виде компонента**, который должен обработать намерение. В объект Intent можно поместить любое количество описаний категорий. Класс Intent определяет несколько констант CATEGORY, например, CATEGORY\_BROWSABLE
* Дополнения - пары ключ-значения для дополнительной **информации**, которую **нужно** поставить **компоненту**, **обращающемуся с намерением**. Например, действие ACTION\_TIMEZONE\_CHANGED имеет дополнение time-zone, которое идентифицирует новый часовой пояс, ACTION\_HEADSET\_PLUG имеет дополнение state, указывающее, включены ли наушники или отключены, а также дополнение name для типа наушников. Объект Intent имеет ряд методов put...() для вставки различных типов дополнительных данных и подобного набора методов get...() для чтения данных. Дополнения устанавливаются и читаются как объекты Bundle с использованием методов putExtras() и getExtras();
* Флаги - указывают системе, как запускать активность (например, какому заданию должна принадлежать активность) и как обработать это после того, как активность запустили (например, принадлежит ли она списку недавних активностей). Все флаги определены в классе Intent.

Система и приложения, которые идут в комплекте с платформой, также используют объекты Intent для активизации определенных системой компонентов (например, различных приложений и служб при загрузке системы).

Константы действия

* ACTION\_ANSWER — Открывает активность, которая связана с входящими звонками. Это действие обрабатывается стандартным экраном для приема звонков;
* ACTION\_CALL — инициализирует обращение по телефону;
* ACTION\_DELETE — Запускает активность, с помощью которой можно удалить данные, указанные в пути URI внутри намерения;
* ACTION\_EDIT — Отображает данные для редактирования пользователем;
* ACTION\_INSERT — Открывает активность для вставки в Курсор (Cursor) нового элемента, указанного с помощью пути URI. Дочерняя активность, вызванная с этим действием, должна вернуть \*URI, ссылающийся на вставленный элемент;
* ACTION\_HEADSET\_PLUG - Подключение наушников;
* ACTION\_MAIN — Запускается как начальная активность задания;
* ACTION\_PICK - Загружает дочернюю Активность, позволяющую выбрать элемент из источника данных, указанный с помощью пути URI. При закрытии должен возвращаться URI, ссылающийся на выбранный элемент. Активность, которая будет запущена, зависит от типа выбранных данных, например при передаче пути content://contacts/people вызовется системный список контактов;
* ACTION\_SEARCH — Запускает активность для выполнения поиска. Поисковый запрос хранится в виде строки в дополнительном параметре намерения по ключу SearchManager.QUERY;
* ACTION\_SEND — Загружает экран для отправки данных, указанных в намерении. Контакт-получатель должен быть выбран с помощью полученной активности. Используйте метод setType, чтобы указать тип MIME для передаваемых данных. Эти данные должны храниться в параметре намерения extras с ключами EXTRA\_TEXT или EXTRA\_STREAM, в зависимости от типа. В случае с электронной почтой стандартное приложение в Android также принимает дополнительные параметры по ключам EXTRA\_EMAIL, EXTRA\_CC, EXTRA\_BCC и EXTRA\_SUBJECT. Используйте действие \*ACTION\_SEND только в тех случаях, когда данные нужно передать удаленному адресату (а не другой программе на том же устройстве);
* ACTION\_SENDTO — Открывает активность для отправки сообщений контакту, указанному в пути URI, который передаётся через намерение;
* ACTION\_SYNC — Синхронизирует данные сервера с данными мобильного устройства;
* ACTION\_TIMEZONE\_CHANGED - Смена часового пояса;
* ACTION\_VIEW — Наиболее распространенное общее действие. Для данных, передаваемых с помощью пути URI в намерении, ищется наиболее подходящий способ вывода. Выбор приложения зависит от схемы (протокола) данных. Стандартные адреса http: будут открываться в браузере, адреса tel: — в приложении для дозвона, geo: — в программе Google Maps, а данные о контакте — отображатся в приложении для управления контактной информацией;
* ACTION\_WEB\_SEARCH — Открывает активность, которая ведет поиск в интернете, основываясь на тексте, переданном с помощью пути URI (как правило, при этом запускается браузер);

Константы категорий

* CATEGORY\_BROWSABLE — активность может быть безопасно вызвана браузером, чтобы отобразить ссылочные данные, например, изображение или почтовое сообщение;
* CATEGORY\_HOME — активность отображает Home Screen, первый экран, который пользователь видит после включения устройства и загрузки системы, или когда нажимает клавишу HOME;
* CATEGORY\_LAUNCHER — активность может быть начальной деятельностью задания из списка приложений в группе Application Launcher устройства

Методы

Для работы с категориями в классе Intent определена группа методов:

* addCategory() — помещает категорию в объект Intent;
* removeCategory() — удаляет категорию, которая была добавлена ранее;
* getCategories() — получает набор всех категорий, находящихся в настоящее время в объекте Intent;

Группы намерений

Намерения могут быть разделены на две группы:

* явные намерения — определяют целевой компонент по имени (составляющее поле имени, имеет набор значения). Явные намерения обычно используются для сообщений внутри приложения, например, когда одна активность запускает другую активность из этого приложения.
* неявные намерения — не называют адресата (поле для составляющего имени — пробел). Неявные намерения часто используются, чтобы активизировать компоненты в других приложениях. Неявные намерения используют для запуска компонентов других приложений. В файле манифеста приложений декларируется фильтр намерений. В отсутствие определяемого адресата система просматривает фильтры намерений всех приложений и находит компонент (активность, службу или приемник широковещательных сообщений), фильтр намерений которого является наиболее подходящим для выполнения данного неявного намерения.

Фильтры намерений и запуск заданий

Если намерение запрашивает выполнение какого-либо действия с определенным набором данных, то системе нужно уметь выбрать приложение (или компонент) для обслуживания этого запроса. На помощь приходят фильтры намерений, которые используются для регистрации активностей, сервисов и широковещательных приёмников в качестве компонентов, способных выполнять заданные действия с конкретным видом данных. С помощью этих фильтров также регистрируются широковещательные приёмники, настроенные на трансляцию намерением заданного действия или события.

Задействуя Фильтры намерений, приложения объявляют, что они могут отвечать на действия, запрашиваемые любой другой программой, установленной на устройстве. Чтобы зарегистрировать компонент приложения в качестве потенциального обработчика намерений, нужно добавить тег <intent-fiiter> в узел компонента в манифесте.

В фильтре намерений декларируется только три составляющих объекта Intent: действие, данные, категория. Дополнения и флаги не играют никакой роли в принятии решения, какой компонент получает намерение.

Например, в любом приложении есть главная активность, которая устанавливается как точка входа для задания:

<activity

android:name=".HelloWorldActivity"

android:label="@string/app\_name">

<intent-filter>

<action android:name="android.intent.action.MAIN"

<category android:name="android.intent.category.LAUNCHER"

</intent-filter>

</activity>

Фильтр такого вида в элементе **<action>** помечает активность, как **запускаемую по умолчанию**. Элемент **<category>** заставляет **значок и метку** для деятельности отображаться **на панели Application Launcher**, давая пользователям возможность запускать задание и возвращаться к этому заданию в любое время после того, как оно было запущено.

Внутри узла Фильтра намерений можно использовать такие теги (и соответствующие атрибуты).

* action - Использует атрибут android:name, чтобы задать название действия, которое будет обслуживаться. Каждый фильтр намерений должен иметь один (и **только один**) **тег action**. Действия должны быть представлены в виде уникальных строк, которые сами себя описывают. Рекомендуется выбирать названия, основываясь на соглашении об именовании пакетов в Java.
* сategory - Использует атрибут android:name, чтобы указать, при каких обстоятельствах должно обслуживаться действие. Каждый тег intent-filter способен содержать **несколько тегов category**. Вы можете задать собственные категории или же брать стандартные значения, предоставляемые системой
* ALTERNATIVE - Наличие данной категории говорит о том, что действие должно быть **доступно в качестве альтернативного тому**, которое выполняется по умолчанию для элемента этого типа данных. Например, если действие по умолчанию для контакта — просмотр, то в качестве альтернативы его также можно редактировать
* SELECTED\_ALTERNATIVE - То же самое, что и ALTERNATIVE, но вместо одиночного действия с использованием утверждения намерения, которое описано выше, применяется в тех случаях, когда нужен **список различных возможностей**. Одной из функций фильтра намерений может стать **динамическое заполнение** контекстного меню с помощью действий.
* BROWSABLE - Говорит о том, что действие **доступно из браузера**. Когда намерение срабатывает в браузере, оно всегда содержит данную категорию. Если вы хотите, чтобы приложение реагировало на действия, инициированные браузером (такие как перехват ссылок на конкретный сайт), то должны добавить в его манифест категорию BROWSABLE.
* DEFAULT - Установите эту категорию, чтобы сделать компонент обработчиком по умолчанию для действия, выполняемого с указанным типом данных внутри Фильтра намерений. Это необходимо и для Активностей, которые запускаются с помощью явных Намерений
* GADGET - Наличие этой категории указывает на то, что **данная активность может запускаться внутри другой активности**.
* HOME - Устанавливая эту категорию и не указывая при этом действия, вы **создаете альтернативу для стандартного домашнего экрана**.
* LAUNCHER - Используя эту категорию, вы **помещаете Активность в окно для запуска приложений**.
* data - Этот тег дает возможность указать **тип данных**, с которым может взаимодействовать ваш компонент. При необходимости можно задать несколько тегов data. Чтобы указать, какие именно данные поддерживает ваш компонент, используйте сочетание следующих атрибутов:
  + android:host — задает доступное **имя удаленного сервера** (например, google.com);
  + android:mimetype — позволяет указать **тип данных**, которые ваш компонент **способен обрабатывать**. Для примера и будет соответствовать любому Курсору в Android:

<type android: value="vnd.android.cursor.dir/\*"/>

* + android:path — **задает** доступные значения для **пути URI** (например, /transport/boats/);
  + android:port — **указывает** доступные **порты** для заданного сервера;
  + android:scheme — требует **указать** конкретную **схему** (например, content или http).

Принцип работы фильтров намерений

При использовании метода startActivity() передаваемое неявное намерение, как правило, доходит лишь до одной активности. Если для выполнения заданного действия с указанными данными годятся сразу несколько активностей, пользователю предоставляется список выбора. Процесс, когда решается, какую активность лучше запустить, называется Утверждением намерений. Его цель — найти наиболее подходящий фильтр намерений. В целом весь алгоритм работает следующим образом. Android собирает список всех доступных Фильтров намерений из установленных пакетов.

Фильтры, которые не соответствуют действию или категории Намерения, удаляются из списка. Совпадение происходит только в том случае, если Фильтр намерений содержит указанное действие (или если действие для него вовсе не задано). Совпадения не произойдет, только если ни одно из действий Фильтра намерений не будет эквивалентно тому, которое задано в Намерении. Для категорий процесс соответствия более строгий. Фильтр намерений должен включать в себя все категории, заданные в полученном Намерении. Фильтр, для которого категории не указаны, может соответствовать только таким же Намерениям (нет категорий).

Наконец, каждая часть пути URI из Намерения сравнивается с тегом data Фильтра намерений. Если в Фильтре указаны схема (протокол), сервер/принадлежность, путь или тип MIME, все эти значения проверяются на соответствие пути URI из Намерения. При любом несовпадении Фильтр будет удален из списка. Если в Фильтре намерений не указано ни одного параметра data, его действие будет распространяться на любые данные.

* MIME — тип данных, который должен совпасть. При сравнении типов данных вы можете использовать маски, чтобы охватывать все подтипы (например, cats/\*). Если в Фильтре намерения указан тип данных, он должен совпасть с тем, который значится в намерении, при отсутствии тега data подойдет любой тип.
* Схема — это протокольная часть пути URI, например http:, mailto: или tel:.
* Имя сервера (или принадлежность данных) — часть URI между схемой и самим путем (например, www.google.com). Чтобы совпало имя сервера, схема Фильтра намерений также должна подойти.

После имени сервера идет путь к данным (например, /ig). Путь пройдет проверку только после схемы и имени сервера, содержащихся в теге.

Когда вы неявным образом запускаете Активность и вышеописанный процесс возвращает более одного совпадения, пользователю выводится список со всеми вариантами

Поиск и использование намерения, с помощью которого была запущена активность

Когда программный компонент запускается с помощью неявного намерения, он должен найти действие, которое необходимо осуществить, и данные для него.

Используйте метод getIntent() (как правило, внутри метода onCreate), чтобы извлечь намерение, с помощью которого компонент запущен.

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

Intent intent = getIntent();

}

Применяйте методы getAction() и getData(), чтобы найти действие и данные, связанные с намерением. Для извлечения дополнительной информации, хранящейся в параметре extras, используйте типизированные методы get<тип>Extra.

String action = intent.getAction();

Uri data = intent.getData();

Делегирование ответственности

Задействуйте метод startNextMatchingActivity() для делегирования ответственности за обработку действия следующему компоненту, который лучше всего для этого подходит.

Intent intent = getIntent();

if (isDuringBreak)

startNextMatchingActivity(intent);

Это позволяет добавлять дополнительные условия для компонентов, которые ограничат сферу их применения рамками утверждения Намерений. В некоторых случаях ваш компонент может выполнить некоторые операции или предложить пользователю выбор, прежде чем передать Намерение другому компоненту.

**ACTION\_PICK**

До сих пор мы в основном рассматривали намерения или действия, которые обычно активируют другую активность, не ожидая получить в ответ на это результат. Теперь рассмотрим более сложное действие, которое возвращает значение после того, как будет активировано. ACTION\_PICK - это обобщенное название для таких действий.

Принцип ACTION\_PICK заключается в том, чтобы запустить активность, отображающую список элементов. После этого активность должна предоставлять пользователю возможность выбора элемента из этого списка. Когда пользователь выберет элемент, активность возвратит URI выбранного элемента вызывающей стороне. Таким образом, можно многократно использовать функцию UI для выбора нескольких элементов определенного типа.

Следующий код выведет диалоговое окно со списком всех возможных программ, которые могут запустить активность с данными, так как мы не указываем конкретный тип (setType("\*/\*")):

// Показываем все программы для запуска Intent intent = new Intent(Intent.ACTION\_PICK); intent.setType("\*/\*"); startActivityForResult(intent, 1);

Если мы укажем конкретный тип данных, которые нас интересует, то список программ существенно сузится. Например, если указать intent.setType("image/\*");, то можем увидеть только системную программу Галерея для просмотра картинок (у пользователя могут быть установлены программы, которые также предназначены для просмотра картинок, поэтому список может отличаться).

При работе с действиями, возвращающими данные, нельзя использовать метод startActivity(), поскольку startActivity() не возвращает результат, так как этот метод открывает новую активность в виде модального диалогового окна в отдельном потоке, а основнои поток оставляет свободным для служебных событий. Другими словами, startActivity() - это асинхронный вызов, с которым не применяются обратные вызовы и, следовательно, невозможно узнать, что произошло с активированным действием. Но если вы хотите вернуть данные, то можете использовать метод startActivityForResult(), приспособленный для работы с обратными вызовами.

Очень подробно про намерения написано в книге Разработка приложений для Android

На странице <http://www.openintents.org/> собрана информация о доступных намерениях в разных программах.

Примеры использования намерений

* Запуск карт Google
* Загрузить изображения в ImageView из галереи
* Запускаем приложение Галерея и выбираем картинку
* Включить Bluetooth
* Управление настройками Wi-Fi-соединениями
* Сообщить системе о включение/выключении автономного режима (Airplane Mode)
* Запуск приложения Батарея
* Открыть окно «Моё местоположение» из системного приложения Настройки
* Открыть окно «Экран» из системного приложения Настройки
* Отправить SMS через намерение ACTION\_VIEW
* Отправить SMS через намерение ACTION\_SENDTO
* Выбрать мелодию (action.RINGTONE\_PICKER)

**Позвонить**

Запустить стандартную звонилку без набора номера (ACTION\_DIAL)

Открыть звонилку и набрать заданный номер. Требуются разрешения android.permission.CALL\_PHONE, CALL\_PRIVILEGED (для экстренных вызовов)

public static void dialNumber(Activity activity) {

Intent intent = new Intent(Intent.ACTION\_CALL);

intent.setData(Uri.parse("tel:" + "5551222"));

activity.startActivity(intent);

}

**Получить контакт из телефонной книги**

Intent pickIntent = new Intent(Intent.ACTION\_PICK,

android.provider.ContactsContract.Contacts.CONTENT\_URI);

startActivityForResult(pickIntent, PICK\_RESULT);

Открывает стандартное приложение контактов. Метод startActivityForResult позволяет получить URI выбранного контакта. Вы увидите сразу все контакты пользователя, включая почтовые. Если необходимо выбрать только контакт из телефонной книги, можно воспользоваться следующим кодом:

Intent pickIntent = new Intent(Intent.ACTION\_GET\_CONTENT); pickIntent.setType(ContactsContract.CommonDataKinds.Phone.CONTENT\_ITEM\_TYPE); startActivityForResult(pickIntent, PICK\_RESULT);

Разрешения: READ\_CONTACTS.

**Открыть ссылку в браузере**

Uri address = Uri.parse("http://developer.alexanderklimov.ru/android");

Intent openlink = new Intent(Intent.ACTION\_VIEW, address);

startActivity(openlink);

Данный код запустит браузер с указаной ссылкой. Eсли ссылка ведет на YouTube, то откроется стандатное приложение для YouTube. Также можно открыть стандартный клиент Android Market с ссылкой на какое-либо приложение. Для этого необходимо в параметр URI передать ссылку вида market://search?q=search query, где search query необходимо заменить соответствующим поисковым запросом.

Разрешения: не требуется.

**YouTube**

Можно запустить конкретно приложение YouTube

// constant value of package & class name of YouTube app

public static final String YOUTUBE\_PACKAGE\_NAME = "com.google.android.youtube";

public static final String YOUTUBE\_CLASS\_NAME = "com.google.android.youtube.WatchActivity";

//Launch Youtube to watch an URL

public static void launchYoutube(Context context, String url) {

Intent intent = new Intent(android.content.Intent.ACTION\_VIEW, Uri.parse(url));

intent.setClassName(YOUTUBE\_PACKAGE\_NAME, YOUTUBE\_CLASS\_NAME);

context.startActivity(intent);

}

**Поисковый запрос**

Можно отправить поисковый запрос в Гугл:

public static void invokeWebSearch(Activity activity) {

String q = "котёнок";

Intent intent = new Intent(Intent.ACTION\_WEB\_SEARCH);

intent.putExtra(SearchManager.QUERY, q);

activity.startActivity(intent);

}

**Отправить письмо**

Добавьте на форму текстовые поля для ввода адресата, темы письма и сообщения. Далее сам код:

final EditText edittextEmailAddress = (EditText) findViewById(R.id.email\_address);

final EditText edittextEmailSubject = (EditText) findViewById(R.id.email\_subject);

final EditText edittextEmailText = (EditText) findViewById(R.id.email\_text);

Button buttonSendEmail\_intent = (Button) findViewById(R.id.sendemail\_intent);

buttonSendEmail\_intent.setOnClickListener(new Button.OnClickListener() {

@Override

public void onClick(View arg0) {

// TODO Auto-generated method stub

String emailAddress = edittextEmailAddress.getText().toString();

String emailSubject = edittextEmailSubject.getText().toString();

String emailText = edittextEmailText.getText().toString();

String emailAddressList[] = { emailAddress };

Intent intent = new Intent(Intent.ACTION\_SEND);

intent.setType("plain/text");

intent.putExtra(Intent.EXTRA\_EMAIL, emailAddressList);

intent.putExtra(Intent.EXTRA\_SUBJECT, emailSubject);

intent.putExtra(Intent.EXTRA\_TEXT, emailText);

startActivity(Intent.createChooser(intent,

"Choice App to send email:"));

}

});

На устройстве должна быть настроена программа, отвечающая за отправку писем. Если таких программ будет несколько, то пользователь сможет выбрать одну из них при помощи диалогового окна.

**Камера**

Программное включение приложения Камера

Программно включить приложение Камера, сделать снимок и получить результат

Запустить приложение Камера в режиме видеозаписи и получить результат Распознавание речи

Intent speechIntent = new Intent(RecognizerIntent.ACTION\_RECOGNIZE\_SPEECH);

speechIntent.putExtra(RecognizerIntent.EXTRA\_LANGUAGE\_MODEL,

RecognizerIntent.LANGUAGE\_MODEL\_FREE\_FORM);

speechIntent.putExtra(RecognizerIntent.EXTRA\_PROMPT, "Можно мяукать сейчас");

startActivityForResult(speechIntent, RESULT\_SPEECH\_TO\_TEXT);

Этот код запускает приложение распознавания речи. Если произнести что-либо, система обработает голос и вернет результат в метод onActivityResult:

@Override

protected void onActivityResult(int requestCode, int resultCode, Intent data) {

if (requestCode == RESULT\_SPEECH\_TO\_TEXT && resultCode == RESULT\_OK) {

ArrayList<String> matches = data.getStringArrayListExtra(RecognizerIntent.EXTRA\_RESULTS);

TextView tvSpeech = (TextView) findViewById(R.id.tv\_speech);

tvSpeech.setText(matches.get(0));

}

}

**Голосовая команда**

Я не проверял.

Intent intent = new Intent(Intent.ACTION\_VOICE\_COMMAND);

startActivity(intent);

**Запустить Twitter**

Twitter можно запустить через намерение. Если на устройстве установлено родное приложение, то запустится оно, иначе запустится веб-версия:

try {

startActivity(new Intent(

Intent.ACTION\_VIEW,

Uri.parse("twitter://user?screen\_name=" + twitter\_user\_name)));

} catch (Exception e) {

startActivity(new Intent(Intent.ACTION\_VIEW,

Uri.parse("https://twitter.com/#!/" + twitter\_user\_name)));

}

Вместо переменной twitter\_user\_name необходимо подставить нужный аккаунт. Отсюда.

**Проиграть MP4-файл (ACTION\_VIEW)**

// файл выбирается из списка

@Override

protected void onListItemClick(ListView l, View v,

int position, long id) {

// TODO Auto-generated method stub

//super.onListItemClick(l, v, position, id);

String uriString = l.getItemAtPosition(position).toString();

Uri intentUri = Uri.parse(uriString);

Intent intent = new Intent();

intent.setAction(Intent.ACTION\_VIEW);

intent.setDataAndType(intentUri, "video/mp4");

startActivity(intent);

}