**Меню Android**

Меню являются общим и хорошо известным компонентом интерфейса с пользователем для многих типов приложений. Чтобы обеспечить простой и непротиворечивый для пользователя способ общения с программой **Android**, Вы должны использовать [Menu](http://developer.android.com/reference/android/view/Menu.html) API для осуществления разных действий в окнах **Activity** [3].

[**Общий обзор меню Android**]

Начиная с Android 3.0 (API level 11) устройства на базе Android больше не обязаны предоставлять специально выделенную кнопку для меню. С таким изменением приложения Android должны уйти от зависимости к традиционной панели меню с 6 элементами, и вместо этого предоставить строку меню (action bar), чтобы в ней указывать общие возможные действия для пользователя.

Хотя дизайн (касаемо опыта пользователя) для некоторых элементов меню изменился, семантика для определения набора действий все еще основывается на функциях Menu API. Это руководство (перевод документации [1]) покажет, как создавать 3 фундаментальные типа меню или представления действий на всех версиях Android:

• **Меню опций и полоса меню** (options menu, action bar)

Меню опций является главной коллекцией элементов меню для экземпляра класса Activity [3] (этот класс олицетворяет окно программы, его вид и поведение). Меню опций - именно то место, куда нужно поместить действия, которые повлияют на все приложение в целом, такие как "Search" (поиск), "Compose email" (составить письмо) и "Settings" (настройки).

Если Вы разрабатываете приложение для Android 2.3 или более старой версии, пользователи могут отобразить панель меню путем нажатия кнопки Menu.

На Android 3.0 и более новых версиях элементы из меню опций (options menu) представлены в полосе меню (action bar [5]) в виде предложения действий, перекрывающих на экране основное окно программы. Начиная с Android 3.0, кнопка Menu устарела (на некоторых устройствах её может просто не быть), так что Вы должны мигрировать на использование action bar, чтобы предоставить доступ к разным операциям и другим опциям.

Подробнее см. раздел "Создание меню опций".

• **Контекстное меню и контекстные действия** (Context menu, contextual action mode)

Контекстное меню - это плавающее меню, которое появляется, когда пользователь выполнил долгое касание (long-click) на элементе. Оно предоставляет действия, касающиеся выбранного содержимого или контекстного окна.

Когда Вы разрабатываете для Android 3.0 и более новой версии, то Вы должны вместо этого использовать контекстные действия (contextual action mode), чтобы разрешить действия для выбранного содержимого. Этот режим отображает элементы действия, которые влияют на выделенный контент в полосе на верхней части экрана, и позволяют пользователю выбрать несколько элементов.

Подробнее см. раздел "Создание контекстного меню".

• **Всплывающее меню** (Popup menu)

Popup menu отображает вертикальный список элементов, привязанный к представлению View, которое вызвало меню. Это хорошо подходит для предоставления дополнительных действий, которые касаются определенного содержания или предоставляют опции для второй части команды. Действия в popup menu не должны непосредственно влиять на соответствующее содержание (для этого предназначено контекстное меню). Скорее popup menu предназначено для расширенных действий, которые связаны с областями контента в Вашей Activity.

Подробнее см. раздел "Создание всплывающего меню".

[**Определение меню в XML**]

Для всех типов меню Android предоставляет стандартный формат XML для определения элементов меню. Вместо того, чтобы строить меню в коде Вашей Activity, Вы должны задать меню и все его элементы в XML-ресурсе меню. Тогда Вы можете встроить ресурс меню (загрузить его как объект[Menu](http://developer.android.com/reference/android/view/Menu.html)) в Вашу Activity или фрагмент. Использование ресурса меню является хорошей практикой по нескольким причинам:

• В XML проще визуализировать структуру меню.  
• Это отделяет содержимое меню от кода, определяющего поведение Вашего приложения.  
• Это позволяет Вам создавать альтернативные конфигурации меню для разных версий платформ, размеров экрана и других конфигураций, усиливая фреймворк ресурсов приложения.

Чтобы определить меню, создайте файл XML в папке проекта res/menu/, и постройте меню из следующих элементов:

< menu > Создает Menu, которое является контейнером для элементов меню. Элемент < menu > должен быть корневым для файла и может содержать один или большее количество элементов < item > и < group >.

< item > Создает MenuItem, который представляет один элемент в меню. Этот элемент может содержать встроенный элемент < menu >, чтобы сделать подменю (submenu).

< group > Необязательный, невидимый контейнер для элементов < item >. Он позволяет разделить пункты меню на категории, так чтобы они могли использовать общие свойства, такие как активное состояние и видимость (visibility). Подробнее см. раздел "Создание групп меню".

Вот пример меню, сохраненного в файл res/menu/game\_menu.xml:

< ?xml version="1.0" encoding="utf-8"? >

**< menu** xmlns:android="http://schemas.android.com/apk/res/android" **>**

**< item** android:id="@+id/new\_game"

android:icon="@drawable/ic\_new\_game"

android:title="@string/new\_game"

android:showAsAction="ifRoom" **/>**

**< item** android:id="@+id/help"

android:icon="@drawable/ic\_help"

android:title="@string/help" **/>**

**< /menu >**

Элемент < item > поддерживает несколько атрибутов, которые Вы можете использовать для определения того, как выглядит элемент и как он будет себя вести. Элементы в вышеупомянутом меню могут включать в себя следующие атрибуты:

android:id Идентификатор ресурса (resource ID) который является уникальным для элемента меню. Идентификатор позволяет приложению распознать элемент меню, когда пользователь его выберет.

android:icon Ссылка на drawable-ресурс (картинка), который используется как иконка пункта меню.

android:title Ссылка на строку, которая используется как текстовое содержимое элемента меню.

android:showAsAction Задает, как этот элемент должен появиться в качестве элемента действия (action item) в полосе меню (action bar [5]).

Это самые важные из атрибутов, которые Вы должны использовать, однако есть еще и многие другие. Для подробной информации по всем поддерживаемым атрибутам см. [2].

Вы можете добавить подменю к элементу в любом месте меню (за исключением подменю), путем добавления элемента < menu > как корневого в < item >. Подменю полезны, когда Ваше приложение имеет некоторые функции, которые можно организовать по темам, наподобие полосы меню программ PC (File, Edit, View, и т. д.). Пример:

< ?xml version="1.0" encoding="utf-8"? >

**< menu** xmlns:android="http://schemas.android.com/apk/res/android" **>**

**< item** android:id="@+id/file"

android:title="@string/file" **>**

*< !-- Подменю "file" -- >*

**< menu** **>**

**< item** android:id="@+id/create\_new"

android:title="@string/create\_new" **/>**

**< item** android:id="@+id/open"

android:title="@string/open" **/>**

**< /menu >**

**< /item >**

**< /menu >**

Для использования меню в Вашей Activity нужно встроить ресурс меню (преобразовать ресурс XML в программируемый объект), используя MenuInflater.inflate(). В последующих секциях будет рассмотрено, как встраивать меню каждого типа.

[**Создание меню опций**]

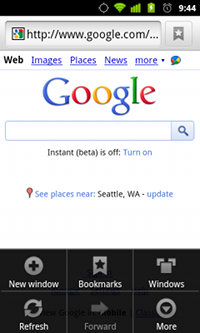


Рис. 1. Меню опций в браузере,  
как оно отображается в Android 2.3.

Меню опций (Options Menu) это то место, где Вы должны добавить действия и разные настройки, которые относятся к текущему контексту активности приложения, такие как Поиск, Составить письмо и Настройки. Место, где элементы будут появляться в Вашем меню опций на экране, зависит от версии Android, для которой разработано приложение:

• Если Вы разрабатываете приложение для Android 2.3.x (API level 10) или более старой версии, содержимое меню опций появится в нижней части экрана, когда пользователь нажмет кнопку Menu, как это показано на рис. 1. После появления меню первой видимой частью меню будет меню со значками, которое может содержать до 6 пунктов (элементов) меню. Если Ваше меню содержит больше 6 элементов, то Android разместит шестой элемент и остальные элементы в перекрывающее меню (overflow menu), которое пользователь может открыть, выбрав последний правый пункт More (Еще).

• Если Вы разрабатываете приложение для Android 3.0 (API level 11) и более новой версии, то элементы в меню опций доступны в полосе меню (action bar). По умолчанию система размещает все элементы в перекрывающее окно действий (action overflow), которое пользователь может отобразить с помощью иконки действия (action overflow icon) в правой части полосы меню (или нажатием кнопки Menu на устройстве, если эта кнопка имеется). Чтобы обеспечить быстрый доступ к важным действиям, Вы может продвинуть несколько элементов для появления их в полосе меню (action bar) путем добавления атрибута android:showAsAction="ifRoom" к соответствующим элементам < item > (см. рис. 2).

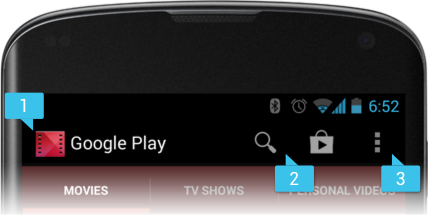


Рис. 2. Полоса меню (action bar) из приложения галереи Honeycomb Gallery, которая показывает закладки навигации (navigation tabs) и элемент действия с камерой (плюс кнопку действия action overflow button).

Дополнительную информацию об элементах действий и поведении полосы меню см. руководство Action Bar.

Примечание: даже если Вы не разрабатываете для Android 3.0 или более новых версий Android, Вы можете создать свою собственную полосу меню (action bar layout) с похожим эффектом. Для примера, как можно поддерживать старые версии Android с полосой меню, см. пример Action Bar Compatibility.

Вы можете задекларировать элементы в для меню опций либо из подкласса Activity приложения, либо из подкласса Fragment [4]. Если и activity, и fragment(ы) декларируют элементы меню опций, то в интерфейсе пользователя они будут скомбинированы. Элементы меню из activity появятся первыми, за ними будут следовать элементы от каждого fragment-а в том порядке, как каждый fragment был добавлен к activity. Если необходимо, Вы можете реорганизовать элементы меню с атрибутом android:orderInCategory в каждом < item >, который нужно передвинуть.

Чтобы указать меню опций для activity, переопределите onCreateOptionsMenu() (fragment-ы предоставляют свой собственный onCreateOptionsMenu() callback). В этом методе Вы можете встроить ресурс меню (заданный в XML) в Menu, предоставленный в методе обратного вызова (callback) onCreateOptionsMenu. Пример:

@Override

**public** boolean onCreateOptionsMenu(Menu menu)

{

MenuInflater inflater = getMenuInflater();

inflater.inflate(R.menu.game\_menu, menu);

**return** **true**;

}

Вы можете также добавить элементы меню используя add() и запросить элементы с помощьюfindItem(), чтобы сделать ревизию их свойств через MenuItem API.

Если Вы разрабатываете приложение для Android 2.3.x и более старых версий Android, система вызовет onCreateOptionsMenu() для создания меню опций, когда пользователь откроет меню впервые. Если же Вы разрабатываете для Android 3.0 и более новых версий, система вызовет onCreateOptionsMenu() при старте activity, чтобы показывать элементы в полосе меню (action bar).

**Обработка событий клика на элементах меню**

Когда пользователь выбирает элемент из меню опций (включая элементы из полосы меню action bar), система вызовет метод активности onOptionsItemSelected(). Этот метод передает выбранный MenuItem. Вы можете идентифицировать элемент меню путем вызова getItemId(), который возвратит уникальный ID для этого элемента меню (этот идентификатор определен атрибутом android:id в ресурсе меню или целым числом, переданным методом add()). Вы можете сравнить этот ID с известными для Вас идентификаторами меню, чтобы выполнить в ответ нужное действие. Пример:

@Override

**public** boolean onOptionsItemSelected(MenuItem item)

{

*// Обработка выбранного элемента меню.*

**switch** (item.getItemId())

{

**case** R.id.new\_game:

newGame();

**return** **true**;

**case** R.id.help:

showHelp();

**return** **true**;

**default**:

**return** **super**.onOptionsItemSelected(item);

}

}

Когда Вы успешно обработали элемент меню, то должно быть возвращено true. Если Вы не обработали элемент меню, то Вы должны вызвать реализацию метода onOptionsItemSelected() из суперкласса (реализация по умолчанию вернет false).

Если Ваша activity включает в себя fragment-ы, то система сначала вызовет onOptionsItemSelected() для activity, затем для каждого fragment-а (в том порядке, в котором фрагменты были добавлены), пока один из них не вернет true, или пока все фрагменты не будут обработаны.

Совет: Android 3.0 добавил возможность задать поведение по клику для элемента меню в XML, используя атрибут android:onClick. Значением этого атрибута должно быть имя метода, определенного в activity, использующем данное меню. Метод должен быть определен как public, и принять один параметр MenuItem — когда система вызовет этот метод, она передаст ему выбранный элемент меню. Подробнее описание примера см. в [2].

Совет: если Ваше приложение содержит несколько activity и некоторые из них предоставляют одно и то же меню опций, рассмотрите возможность создания activity, которая не реализует ничего, за исключением методов onCreateOptionsMenu() и onOptionsItemSelected(). Затем расширьте (extend) этот класс для каждой activity, которая должна использовать то же самое меню опций. Таким способом Вы можете писать только один код, который будет обрабатывать действия меню для каждого дочернего класса, который будет наследовать поведение меню. Если Вы хотите добавить элементы меню к одной из дочерних activity, перезадайте (override) метод onCreateOptionsMenu() в этой activity. Вызовите super.onCreateOptionsMenu(menu) для оригинальных пунктов созданного меню, затем добавьте новые элементы меню вызовом menu.add(). Вы можете также перезадать поведение суперкласса для отдельных элементов меню.

**Изменение элементов меню во время выполнения (runtime)**

После того, как система вызовет onCreateOptionsMenu(), она сохранит экземпляр Menu, которое Вы заполняете, и не будет вызывать onCreateOptionsMenu() снова за исключением случаев, когда меню будет сделано недействительным (invalidated) по некоторой причине. Однако Вы должны использовать onCreateOptionsMenu() только чтобы создать начальное состояние меню, и не вносить изменения в течение всего жизненного цикла activity.

Если Вы хотите модифицировать меню опций на базе событий, которые случаются во время жизненного цикла activity, Вы можете делать это в методе onPrepareOptionsMenu(). Этот метод передает объект Menu в качестве текущего существующего, так что Вы можете его модифицировать: добавлять, удалять или запрещать пункты меню (фрагменты также предоставляют onPrepareOptionsMenu() callback).

На Android 2.3.x и более старых версиях система вызывает onPrepareOptionsMenu() каждый раз, когда пользователь открывает меню опций (presses the Menu button). На Android 3.0 и более новых версиях считается, что меню опций открыто всегда, когда элементы меню присутствуют в полосе меню (action bar). Когда происходит событие, и Вы хотите обновить меню, то должны вызватьinvalidateOptionsMenu() для запроса к системе, чтобы она вызвала onPrepareOptionsMenu().

Примечание: Вы никогда не должны изменять элементы в меню опций на базе View, находящегося сейчас в фокусе. Когда программа находится в режиме касания (когда пользователь не использует трекбол или d-pad), представления View не могут получить фокус, так что Вы не должны использовать фокус в качестве базы для модификаций элементов в меню опций. Если Вы хотите предоставить элементы меню, которые являются контекстно-чувствительными для View, то используйте другой вид меню - контекстное (Context Menu, см. далее).

[**Создание контекстного меню**]

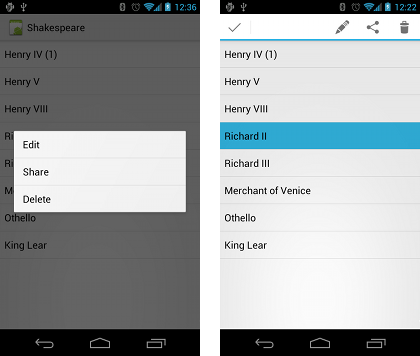


Рис. 3. Скриншоты плавающего контекстного меню (слева)  
и контекстного action bar (справа).

Контекстное меню (Context Menu) предоставляет действия, которые касаются отдельных элементов интерфейса пользователя или контекста его окон (фреймов). Вы можете предоставить контекстное меню для любого View (это любой виджет программы - кнопка, галочка, строка ввода и т. п.), однако чаще всего контекстное меню применяют для ListView,GridView или других коллекций View, в которых пользователь может выполнить конкретные действия с каждым элементом коллекции.

Есть два метода предоставления контекстных действий:

• В плавающем контекстном меню. Меню появится в виде списка элементов поверх окна программы (наподобие диалога), когда пользователь выполнил долгое касание (long-click, сделал нажатие и удерживает) на View, который декларировал поддержку контекстного меню. Пользователи могут выполнить контекстные действия по одному элементу за раз.  
• В режиме контекстных действий (contextual action mode). Этот режим является системной реализацией ActionMode, который отображает контекстную полосу меню (contextual action bar) в верхней части экрана с действиями, которые влияют на выбранные элементы интерфейса. Когда активен этот режим, пользователи могут выполнять действия с несколькими элементами меню сразу (если приложение это предусматривает).

Совет: contextual action mode доступен на Android 3.0 (API level 11) и более новых версиях, и это предпочтительная техника для отображения контекстных действий, когда она доступна. Если Ваше приложение поддерживает версии Android ниже 3.0, то вы должны откатиться к использованию плавающих контекстных меню.

**Создание плавающего контекстного меню**

Для предоставления плавающего контекстного меню выполните следующие шаги:

**1**. Зарегистрируйте View, с которым должно быть ассоциировано контекстное меню путем вызова метода registerForContextMenu() и передачи ему View.

Если Ваша activity использует ListView или GridView, и Вы хотите для каждого предоставить одинаковое контекстное меню, то зарегистрируйте все элементы для контекстного меню путем передачи ListView или GridView в метод registerForContextMenu().

**2**. Реализуйте метод onCreateContextMenu() в теле Activity или Fragment. Когда зарегистрированный View принимает событие долгого клика (long-click event), система вызывает Ваш метод onCreateContextMenu(). Это то место, где Вы определяете элементы меню, обычно путем встраивания ресурса меню. Пример:

@Override

**public** void onCreateContextMenu(ContextMenu menu, View v,

ContextMenuInfo menuInfo)

{

**super**.onCreateContextMenu(menu, v, menuInfo);

MenuInflater inflater = getMenuInflater();

inflater.inflate(R.menu.context\_menu, menu);

}

MenuInflater позволяет Вам встроить контекстное меню из ресурса меню (который как обычно задается в XML). Параметры callback-метода включают в себя View, на котором пользователь сделал long-click и объект ContextMenu.ContextMenuInfo, который предоставляет дополнительную информацию о выбранном элементе. Если Ваша activity имеет несколько представлений View (виджетов интерфейса), каждое из которых предоставляет свое отличающееся контекстное меню, то Вы можете использовать эти параметры, чтобы определить, какое контекстное меню встраивается.

**3**. Реализуйте метод onContextItemSelected(). Когда пользователь выбирает элемент меню, система вызовет этот метод, так что Вы можете предпринять соответствующее действие в коде. Пример:

@Override

**public** boolean onContextItemSelected(MenuItem item)

{

AdapterContextMenuInfo info = (AdapterContextMenuInfo)item.getMenuInfo();

**switch** (item.getItemId())

{

**case** R.id.edit:

editNote(info.id);

**return** **true**;

**case** R.id.delete:

deleteNote(info.id);

**return** **true**;

**default**:

**return** **super**.onContextItemSelected(item);

}

}

Метод getItemId() опрашивает ID для выбранного элемента меню. Уникальный идентификатор ID должен быть назначен для каждого элемента в XML ресурса меню с использованием атрибутаandroid:id, как это было показано в секции "Определение меню в XML". Когда Вы успешно обработали элемент меню, то верните из onContextItemSelected значение true. Если Вы не обработали элемент меню, то должны передать элемент в реализацию обработки меню суперкласса. Если в Вашей activity имеются добавленные fragment-ы, то сначала вызовется callback onContextItemSelected этой activity. Когда будет вызван метод суперкласса для необработанного элемента меню (из onContextItemSelected активности Вы вернули false), то система передаст событие по цепочке к соответствующему методу callback в каждом fragment-е, по одному за раз (в том порядке, как fragment-ы были добавлены к activity), пока не будет возвращено из callback значение false, или пока fragment-ы не закончатся. Реализация по умолчанию для Activity иandroid.app.Fragment возвратит false, так что Вы должны всегда вызвать суперкласс, когда событие не было обработано.

**Использование режима контекстных действий (contextual action mode)**

Режим contextual action является системной реализацией ActionMode, фокусирующий взаимодействие с пользователем на выполнении контекстных действий. Когда пользователь разрешает этот режим выбором соответствующего элемента, в верхней части экрана появляется полоса меню (action bar), и она предоставляет действия которые пользователь может выполнить на текущем выбранном элементе (или элементах) интерфейса пользователя. Когда разрешен этот режим, пользователь может в меню выбрать сразу несколько элементов (если в приложении это предусмотрено и разрешено), отменить выбор элементов, и продолжить навигацию по activity (как Вы готовы это позволить). Режим action mode запрещен и contextual action bar пропадает с экрана, когда пользователь отменяет выбор всех элементов, нажимает кнопку BACK (назад), или когда выберет действие Done в левой части полосы меню (action bar).

Примечание: contextual action bar необязательно связывается с action bar. Он работает независимо, несмотря на то, что contextual action bar визуально перекрывает позицию action bar.

Если Вы разрабатываете для Android 3.0 (API level 11) или для более свежих версий, то Вы обычно должны использовать contextual action mode для предоставления контекстных действий вместо использования плавающего контекстного меню.

Для представлений View, которые предоставляют контекстные действия, Вы должны обычно применить contextual action mode с одним из двух событий (или с обоими сразу):

• Пользователь выполнил долгий клик на каком-то элементе интерфейса.  
• Пользователь выбрал чекбокс или подобный элемент интерфейса в пределах view.

От Вашего дизайна зависит, как Ваше приложение будет вызывать contextual action mode, и какое будет поведение для каждого действия. Есть два базовых принципа дизайна:

• Для контекстных действий с отдельными, произвольными представлениями View.  
• Для пакетных контекстных действий (batch contextual actions) на группах элементов в ListView или GridView (это позволяет пользователю выбрать несколько элементов и выполнить действие на всех выбранных).

Далее будет показано, какие действия нужно предпринять для каждого сценария.

**Как разрешить работу contextual action mode для отдельных View**

Если Вы хотите использовать contextual action mode, когда пользователь выбирает только некоторые отдельные элементы интерфейса (View), то нужно выполнить следующее:

**1**. Реализовать интерфейс ActionMode.Callback. В методе callback-а Вы можете указать действия для контекстного action bar, в ответ на события клика на элементах действия, и обработать другие события времени жизни для action mode.  
**2**. Вызвать startActionMode(), когда Вы хотите показать полосу меню (в таких случаях, когда пользователь сделал долгий клик на View).

Пример:

**1**. Реализация интерфейса ActionMode.Callback:

**private** ActionMode.Callback mActionModeCallback = **new** ActionMode.Callback()

{

*// Вызывается, когда создан action mode. Был вызван startActionMode().*

@Override

**public** boolean onCreateActionMode(ActionMode mode, Menu menu)

{

*// Встраивание ресурса меню, предоставляющего элементы*

*// контекстного меню.*

MenuInflater inflater = mode.getMenuInflater();

inflater.inflate(R.menu.context\_menu, menu);

**return** **true**;

}

*// Вызывается каждый раз, когда показывается action mode. Вызывается*

*// всегда после onCreateActionMode, но может быть вызван несколько*

*// раз, если режим переделывался (invalidated).*

@Override

**public** boolean onPrepareActionMode(ActionMode mode, Menu menu)

{

**return** **false**; *// Возврат false, если обработки не было.*

}

*// Вызывается, когда пользователь выбрал элемент контекстного меню.*

@Override

**public** boolean onActionItemClicked(ActionMode mode, MenuItem item)

{

**switch** (item.getItemId())

{

**case** R.id.menu\_share:

shareCurrentItem();

*//Это закроет contextual action bar:*

mode.finish();

**return** **true**;

**default**:

**return** **false**;

}

}

*// Вызывается, когда пользователь выходит из action mode.*

@Override

**public** void onDestroyActionMode(ActionMode mode)

{

mActionMode = **null**;

}

};

Имейте в виду, что эти callback-и обработки событий обычно те же самые, что и callback-и для меню опций, за исключением каждого из них, которые также передают объект ActionMode, связанный с событием. Вы можете использовать ActionMode API для того, чтобы сделать различные изменения в CAB (contextual action bar), такие как изменения строки заголовка (title) вызовом setTitle() иsetSubtitle() (полезно, чтобы показать, сколько элементов выбрано).

Также имейте в виду, что вышеприведенный пример устанавливает переменную mActionMode в null, когда уничтожается action mode. На следующем шаге Вы увидите, как может быть полезным инициализировать и сохранить ActionMode как переменную класса в Вашей activity или fragment-е.

**2**. Вызов startActionMode(), чтобы разрешить contextual action mode, когда это необходимо, как например в ответ на long-click на элементе View:

someView.setOnLongClickListener(**new** View.OnLongClickListener()

{

*// Вызывается, когда пользователь сделал long-click на*

*// каком-нибудь View*

**public** boolean onLongClick(View view)

{

**if** (mActionMode != **null**)

{

**return** **false**;

}

*// Запуск CAB с использованием ActionMode.Callback,*

*// который был определен выше.*

mActionMode = getActivity().startActionMode(mActionModeCallback);

view.setSelected(**true**);

**return** **true**;

}

});

Когда Вы вызвали startActionMode(), система вернет созданный ActionMode. Путем сохранения его в переменную класса Вы можете делать изменения в contextual action bar в ответ на другие события. В примере, показанном выше, ActionMode используется, чтобы экземпляр ActionMode не был пересоздан, если он уже активен, путем проверки на null перед запуском action mode.

**Как разрешить batch contextual actions в ListView или GridView**

Пакетный режим контекстных действий (batch contextual actions) нужен, если у Вас есть набор элементов в ListView или GridView (или в другом расширении от AbsListView), и Вы хотите выполнить пакетные действия на них. Для этого требуется:

• Реализовать интерфейс AbsListView.MultiChoiceModeListener, и установить его для группы View вызовом setMultiChoiceModeListener(). В прослушивающих callback-методах Вы можете указать действия для CAB (contextual action bar), отвечающие на события клика на элементах действий, и обработать другие callback-и, унаследованные от интерфейса ActionMode.Callback.  
• Вызвать setChoiceMode() с аргументом CHOICE\_MODE\_MULTIPLE\_MODAL.

Пример:

ListView listView = getListView();

listView.setChoiceMode(ListView.CHOICE\_MODE\_MULTIPLE\_MODAL);

listView.setMultiChoiceModeListener(**new** MultiChoiceModeListener()

{

@Override

**public** void onItemCheckedStateChanged(ActionMode mode, int position,

long id, boolean checked)

{

*// Здесь Вы можете что-то сделать, когда элементы выбираются*

*// или их выбор отменяется, как например можно обновить*

*// заголовок в CAB.*

...

}

@Override

**public** boolean onActionItemClicked(ActionMode mode, MenuItem item)

{

*// Код обработки кликов на действиях в CAB.*

**switch** (item.getItemId())

{

**case** R.id.menu\_delete:

deleteSelectedItems();

mode.finish(); *// действие выбрано, закроем CAB*

**return** **true**;

**default**:

**return** **false**;

}

}

@Override

**public** boolean onCreateActionMode(ActionMode mode, Menu menu)

{

*// Встраивание меню для CAB*

MenuInflater inflater = mode.getMenuInflater();

inflater.inflate(R.menu.context, menu);

**return** **true**;

}

@Override

**public** void onDestroyActionMode(ActionMode mode)

{

*// Здесь Вы можете сделать нужные обновления для activity,*

*// когда CAB удален. По умолчанию, выбор элементов*

*// отменяется (для тех, которые были выбраны).*

...

}

@Override

**public** boolean onPrepareActionMode(ActionMode mode, Menu menu)

{

*// Здесь Вы можете выполнить обновления для CAB в ответ*

*// на запрос invalidate().*

**return** **false**;

}

});

Теперь, когда пользователь выберет элемент с долгим кликом, система вызовет методonCreateActionMode() и отобразит CAB (contextual action bar) с указанными действиями. Пока CAB отображается на экране, пользователи могут выбирать дополнительные элементы.

В некоторых случаях, когда контекстные действия предоставляют общие элементы действий, Вы можете захотеть добавить чекбокс или другой подобный элемент UI, который позволяет пользователям выбрать элементы, потому что они не могли бы обнаружить долгий клик. Когда пользователь выбирает чекбокс, Вы можете задействовать contextual action mode путем установки помеченного состояния (checked) на соответствующем элементе списка вызовом setItemChecked().

[**Создание всплывающего меню**]

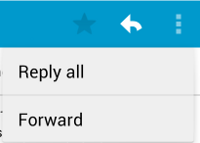


Рис. 4. Popup меню в приложении  
Gmail, привязанное к oweflow-кнопке  
справа вверху.

Всплывающее меню (Popup Menu) является модальным меню, которое привязано к View (появилось начиная с API level 11). Оно появляется ниже View (если есть место), или поверх (если места нет). Popup Menu полезно для:

• Предоставления меню в стиле перекрывания для действий, которые относятся к отдельному содержимому (такому как заголовки писем Gmail, как это показано на рис. 4).

Примечание: это не то же самое, что и контекстное меню, которое обычно применяется для действий, влияющих на выбранный контент. Для действий, которые влияют на выбранный контент используйте contextual action mode или плавающее контекстное меню.

• Предоставления второй части для последовательности команды (такой как кнопка с пометкой "Add", которая предоставляет popup menu с разными опциями для добавления).

• Предоставления выпадающего списка (drop-down) наподобие Spinner, который не сохраняет постоянный выбранный вариант.

Если Вы определили меню в XML, вот шаги для отображения popup menu:

**1**. Создайте конструктором экземпляр PopupMenu, который получит текущий контекст приложенияContext и View, к которому меню должно быть привязано.

**2**. Используйте MenuInflater для встраивания ресурса Вашего меню в объект Menu, который вернет PopupMenu.getMenu(). На API level 14 и более новом Вы можете использовать вместо этогоPopupMenu.inflate().

**3**. Вызовите PopupMenu.show().

Например, вот кнопка с атрибутом android:onClick, которая покажет popup menu:

**< ImageButton**

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:src="@drawable/ic\_overflow\_holo\_dark"

android:contentDescription="@string/descr\_overflow\_button"

android:onClick="showPopup" **/>**

Примерно так activity может отобразить popup menu:

**public** void showPopup(View v)

{

PopupMenu popup = **new** PopupMenu(**this**, v);

MenuInflater inflater = popup.getMenuInflater();

inflater.inflate(R.menu.actions, popup.getMenu());

popup.show();

}

На API level 14 и более свежем Вы можете объединить две строки, которые встраивают меню, вызовом PopupMenu.inflate().

Меню будет отменено, когда пользователь выберет элемент меню или коснется в любом месте вне зоны меню. Вы можете прослушивать события отмены меню (dismiss event) с использованиемPopupMenu.OnDismissListener.

**Обработка кликов в меню**

Чтобы предпринимать действия в ответ на выбор пользователя в меню, Вы должны реализовать интерфейс PopupMenu.OnMenuItemClickListener и зарегистрировать его вызовомsetOnMenuItemclickListener(). Когда пользователь выберет элемент в меню, система вызоветonMenuItemClick() callback в Вашем интерфейсе. Пример:

**public** void showMenu(View v)

{

PopupMenu popup = **new** PopupMenu(**this**, v);

*// Здесь activity реализует OnMenuItemClickListener.*

popup.setOnMenuItemClickListener(**this**);

popup.inflate(R.menu.actions);

popup.show();

}

@Override

**public** boolean onMenuItemClick(MenuItem item)

{

**switch** (item.getItemId())

{

**case** R.id.archive:

archive(item);

**return** **true**;

**case** R.id.delete:

delete(item);

**return** **true**;

**default**:

**return** **false**;

}

}

[**Создание групп меню**]

Группа меню (Menu Group) является коллекцией элементов меню, которые имеют определенные общие черты. С группами Вы можете:

• Показать или спрятать все элементы группы вызовом setGroupVisible().  
• Разрешить или запретить все элементы группы вызовом setGroupEnabled().  
• Указать, должны ли все элементы группы иметь возможность установки галочки (checkable) вызовом setGroupCheckable().

Вы можете создать группу путем вложения элементов < item > внутрь элемента < group > в ресурсе меню (если Вы задаете структуру меню через XML), или путем указания идентификатора группы (group ID) в методе add() (если создайте меню программно).

Вот пример ресурса меню, который имеет группу:

< ?xml version="1.0" encoding="utf-8"? >

**< menu** xmlns:android="http://schemas.android.com/apk/res/android" **>**

**< item** android:id="@+id/menu\_save"

android:icon="@drawable/menu\_save"

android:title="@string/menu\_save" **/>**

*< !-- menu group -- >*

**< group** android:id="@+id/group\_delete" **>**

**< item** android:id="@+id/menu\_archive"

android:title="@string/menu\_archive" **/>**

**< item** android:id="@+id/menu\_delete"

android:title="@string/menu\_delete" **/>**

**< /group >**

**< /menu >**

Элементы, которые в находятся в группе, появятся на одном и том же уровне, как и первый элемент — все три элемента в меню одноуровневые. Однако Вы можете модифицировать черты этих двух элементов в группе одновременно, обращаясь к ним через group ID и используя методы, которые были упомянуты ранее. Система также никогда не будет разделять группированные элементы. Например, если Вы задекларируете android:showAsAction="ifRoom" для каждого элемента, то они оба появятся в action bar или оба появятся в action overflow.

**Использование элементов меню с галочками (checkable)**

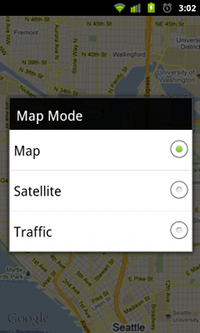


Рис. 5. Скриншот подменю   
с выбираемыми элементами.

Меню может быть полезно для организации интерфейса, в котором опции просто включаются и выключаются с использованием чекбокса, или radio-кнопок для групп, в которых опции взаимно исключают друг друга. На рис. 5 показано submenu с элементами, имеющими радиокнопки.

Примечание: элементы меню в Icon Menu (из меню опций) не может отобразить чекбокс или радиокнопки. Если Вы решите сделать элементы в Icon Menu выбираемыми (с галочками или радиокнопками), то Вы должны вручную показывать выбранное состояние для нужных элементов меню с помощью картинок (путем смены иконок меню) или текста для каждого из выбранных измененных состояний.

Вы можете задать выбираемое (checkable) поведение для отдельных элементов меню с использованием атрибута android:checkable в элементе < item >, или для всей группы с помощью атрибутаandroid:checkableBehavior в элементе < group >. Например, все элементы в этой группе меню будут выбираемые, с радиокнопками:

< ?xml version="1.0" encoding="utf-8"? >

**< menu** xmlns:android="http://schemas.android.com/apk/res/android" **>**

**< group** android:checkableBehavior="single" **>**

**< item** android:id="@+id/red"

android:title="@string/red" **/>**

**< item** android:id="@+id/blue"

android:title="@string/blue" **/>**

**< /group >**

**< /menu >**

Атрибут android:checkableBehavior допускает следующие варианты:

single только один элемент в группе может быть выбран (это радиокнопки).  
all все элементы в меню могут быть выбраны (это чекбоксы).  
none нет выбираемых элементов в меню.

Вы можете применить выбранное состояние по умолчанию с помощью атрибута android:checked в элементе < item >, и поменять это в коде (runtime) методом setChecked().

Когда выбран checkable-элемент меню, система вызовет соответствующий метод callback для выбранного элемента (такой как onOptionsItemSelected()). Это то место, где Вы должны установить состояние чекбокса, потому что чекбокс или радиокнопка не меняют свое состояние автоматически. Вы можете опросить текущее состояние элемента меню (например до того, как пользователь выбрал элемент меню) методом isChecked() и установить состояние методом setChecked(). Пример:

@Override

**public** boolean onOptionsItemSelected(MenuItem item)

{

**switch** (item.getItemId())

{

**case** R.id.vibrate:

**case** R.id.dont\_vibrate:

**if** (item.isChecked()) item.setChecked(**false**);

**else** item.setChecked(**true**);

**return** **true**;

**default**:

**return** **super**.onOptionsItemSelected(item);

}

}

Если Вы не установите checked-состояние таким способом, то видимое состояние элемента меню (чекбокс или радиокнопка) не изменятся, когда пользователь выберет его. Когда Вы устанавливаете состояние checked, activity сохранит это состояние до того момента, как пользователь позже снова откроет меню опций, и checked-состояние, которое было установлено, будет видимым.

Примечание: выбираемые (checkable) элементы меню предназначены только для использования на базе сессии программы, и checkable-состояния элементов меню не сохраняются, после того как приложение уничтожено в оперативной памяти. Если Вы хотите сохранять настройки приложения для пользователя, то нужно специально сохранять данные с использованием [Shared Preferences](http://developer.android.com/guide/topics/data/data-storage.html" \l "pref).

[**Добавление пунктов меню на базе Intent**]

Иногда Вам может понадобиться, чтобы в пункте меню был элемент, запускающий activity с использованием Intent (независимо от того, относится эта activity Вашему приложению или другому приложению). Когда Вы знаете intent, который хотите использовать, и у Вас есть специальный элемент меню для инициализации intent, то Вы можете запустить intent через startActivity() в соответствующем методе callback выбранного элемента меню (таком как onOptionsItemSelected()callback).

Однако если Вы не уверены, какое именно приложение (обрабатывающее intent) содержится на устройстве пользователя, то добавление элемента меню может привести к тому, что этот пункт меню работать не будет, потому что intent может не найти activity. Чтобы исправить это, Android позволяет Вам динамически добавлять элементы к Вашему меню, когда Android находит разные activity на устройстве, которые обрабатывают Ваш intent.

Чтобы добавить элементы меню, основываясь на доступных activity, которые принимают intent:

**1**. Определите intent с категорией CATEGORY\_ALTERNATIVE и/или CATEGORY\_SELECTED\_ALTERNATIVE, плюс любые дополнительные требования.

**2**. Вызовите Menu.addIntentOptions(), тогда Android будет искать любые приложения, которые могут выполнить intent, и добавит их в Ваше меню.

Если нет приложений, которые могут удовлетворить intent, то соответствующие пункты меню добавлены не будут.

Примечание: CATEGORY\_SELECTED\_ALTERNATIVE используется для обработки текущего выбранного элемента на экране. Так что он должен использоваться только к когда создается меню вызовом onCreateContextMenu().

Пример:

@Override

**public** boolean onCreateOptionsMenu(Menu menu)

{

**super**.onCreateOptionsMenu(menu);

*// Создается Intent, который описывает требования для добавления*

*// в наше меню. Предложенное приложение должно включать значение*

*// категории Intent.CATEGORY\_ALTERNATIVE.*

Intent intent = **new** Intent(**null**, dataUri);

intent.addCategory(Intent.CATEGORY\_ALTERNATIVE);

*// Поиск и расширение меню с допустимыми предоставленными приложениями.*

menu.addIntentOptions(

R.id.intent\_group, *// Группа меню, в которую будут добавлены новые элементы*

0, *// Уникальный ID (отсутствует)*

0, *// Порядок для элементов (не указан)*

**this**.getComponentName(), *// Имя текущей activity*

**null**, *// Отдельные элементы, которые будут размещены первыми (их нет)*

intent, *// Intent, созданный ранее, который описывает наши требования*

0, *// Дополнительные флаги для управления элементами (не указаны)*

**null**); *// Массив MenuItem, который коррелирует со специальными элементами (отсутствует)*

**return** **true**;

}

Для каждой найденной activity, которая предоставляет intent filter, подходящий заданному intent, будет добавлен элемент меню с использованием значения android:label в intent filter в качестве текста в пункте меню и иконка приложения в качестве иконки пункта меню. Метод addIntentOptions()возвратит количество добавленных пунктов меню.

Примечание: когда Вы вызываете addIntentOptions(), он переопределяет все пункты меню в группе, указанной в первом аргументе.

**Разрешение для Вашей activity быть добавленной в другие меню**

Вы также можете предоставить сервисы Вашей activity для других приложений, так что Ваше приложение может быть добавлено в меню других программ (обратная роль к роли, описанной выше).

Чтобы можно было добавить приложение в меню других приложений, Вам нужно определить intent filter как обычно, но убедиться, что добавлены значения CATEGORY\_ALTERNATIVE и/илиCATEGORY\_SELECTED\_ALTERNATIVE для категории intent filter. Например:

**< intent-filter** label="@string/resize\_image" **>**

...

**< category** android:name="android.intent.category.ALTERNATIVE" **/>**

**< category** android:name="android.intent.category.SELECTED\_ALTERNATIVE" **/>**

...

**< /intent-filter>**

Дополнительную информацию по написанию фильтров intent см. в документе Intents and Intent Filters [6]. Для получения примера этой техники см. пример кода [Note Pad](http://developer.android.com/resources/samples/NotePad/src/com/example/android/notepad/NoteEditor.html).