Лабораторная работа

Создание меню

Краткие теоретические сведения

Меню - важная часть любого приложения.

Система Android предлагает достаточно простой интерфейс для создания стандартизированных прикладных меню и использования их в приложениях с разнообразной функциональностью. Для этого используется панель действий Action Bar

Action Bar идентифицирует приложение и обеспечивает более удобную навигацию. Этот элемент можно отключать, если в нем нет необходимости. Action Bar занимает гораздо больше места, чем традиционная строка заголовка, поэтому на устройствах с маленьким экраном его использование нецелесообразно.

Action Bar позволяет выводить опции меню (если оно реализовано в приложении) с текстом и значками (которая выглядит почти как строка главного меню в приложениях для настольных систем), что упрощает работу с приложением.

Типы меню

Android предлагает три основных типа меню.

- *Меню опций* (Options Menu) - набор пунктов меню, прикрепляемый к Activity. В старых телефонах есть отдельная кнопка Menu (), нажатие которой вызывает меню. В новых устройствах отдельную кнопку убрали, заменив на значок меню в виде трёх точек в вертикальной ориентации. Для меню выбора опций дополнительно существуют еще две разновидности меню:
  - *меню со значками* (Icon Menu) - расширение меню выбора опций, добавляющее значки к тексту в пункты меню. Меню может содержать максимум шесть пунктов. Этот тип меню - единственный, который поддерживает значки;
  - *расширенное меню* (Expanded Menu) - вертикальный выпадающий список пунктов меню. Расширенное меню появляется при наличии более шести пунктов меню. При этом в меню выбора опций появляется дополнительный пункт `More`. Расширенное меню добавляется автоматически системой Android. При нажатии пункта `More` показывается расширенное меню со списком пунктов, которые не поместились в основной части меню выбора опций.
- *Контекстное меню* (Context Menu) - всплывающий список пунктов меню, который появляется при касании сенсорного экрана в течение двух и более секунд (событие long-press).

– *Подменю (*Submenu*)* - всплывающий список пунктов меню, который привязан к конкретному пункту в меню выбора опций или в контекстном меню. Пункты подменю не поддерживают вложенные подменю.

Упражнение 1. *Создание простого меню*

Options Menu - наиболее распространенный тип меню в приложениях Android.

В шаблоне ***Empty Activity*** нет меню, поэтому будем создавать его сами. Это поможет понять принцип работы и получить общее представление о проекте.

В других шаблонах меню будет встроено и его можно сразу использовать.

1) создайте новый проект *Options menu* на основе Empty Activity. Никакого меню пока нет.
2) создайте несколько строковых ресурсов в файле **res/values/strings.xml**, которые будут отвечать за пункты меню:

```
<string name="app_name">Простое меню</string>
<string name="action_1">Open</string>
<string name="action_2">Edit</string>
<string name="action_3">Save</string>
<string name="action_4">Help</string>
<string name="action_5">Exit</string>
```

3) создайте новую папку *menu* в папке *res* (правый щелчок мыши на папке res | New | Directory).
4) далее создайте в этой папке файл *menu_main.xml* - имя указывает, что меню относится к основной активности MainActivity (правый щелчок мыши на папке menu | New | Menu Resource File). Когда создается приложение с несколькими экранами, то у каждой активности будет отдельное меню со своими настройками.
5) откройте файл *menu_main.xml* и добавьте в полученный шаблон свой код:

```
tools:context=".MainActivity">
<item
android:id="@+id/ action_0"
android:orderInCategory="0"
android:title="@string/action_1" />
```

В XML-файле меню есть три элемента:

– *<menu>* - корневой элемент файла меню;
– *<group>* - контейнерный элемент, определяющий группу меню;
– *<item>* - элемент, определяющий пункт меню.

Элементы *<item>* и *<group>* могут быть дочерними элементами. Конечно, корневой узел любого файла должен быть элементом меню.

6) откройте файл *MainActivity*. Сейчас в нём только один метод *onCreate*(). Добавьте новый метод *onCreateOptionsMenu()*. Именно данный метод отвечает за появление меню у активности. Сразу после метода *onCreate()* начинайте вводить первые символы метода и дальше студия сама покажет список подходящих методов.

Найдите нужный метод и заготовка будет создана автоматически.

```
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        return super.onCreateOptionsMenu(menu);
    }
```

7) добавьте в заготовку метод, который берёт данные из ресурсов меню и преобразует их в пункты меню на экране.

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    getMenuInflater().inflate(R.menu.menu_main, menu);
    return true;
}
```

8) запустите проект. Теперь в правой части заголовка появился значок из трёх точек, выстроенных в вертикальную линию (рис.1). Нажмите на значок, чтобы увидеть пункт меню **Open**. Как не трудно догадаться, элемент **item** отвечает за отдельный пункт меню.

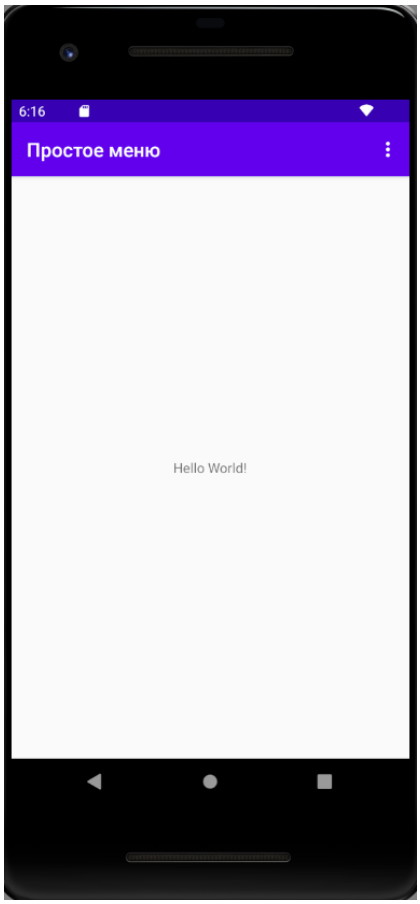

Рисунок 1

9) Добавьте в *menu_main.xml* ещё 4 пункта по такому же принципу, меняя только идентификатор и текст для меню:

```
<item
    android:id="@+id/action_cat1"
    android:orderInCategory="100"
    android:title="@string/action_2" />
```

```
<item
    android:id="@+id/action_cat2"
    android:orderInCategory="100"
    android:title="@string/action_3" />

<item
    android:id="@+id/action_cat3"
    android:orderInCategory="100"
    android:title="@string/action_4"/>
<item
    android:id="@+id/action_cat4"
    android:orderInCategory="100"
    android:title="@string/action_5"/>
```

Параметры **id** и **title** не нуждаются в объяснениях. Параметр **orderInCategory** позволяет задать свой порядок вывода пунктов меню. Предположим вы создали пять пунктов меню, но пока не определились с порядком их вывода на экране. Чтобы не перемещать постоянно целые блоки кода для пунктов меню в нужном порядке, можно воспользоваться данным параметром.

10) Запустите проект и попробуйте снова вызвать меню. Вы увидите 4 новых пункта. Вид приложения с развернутым меню представлен на рис. 2.

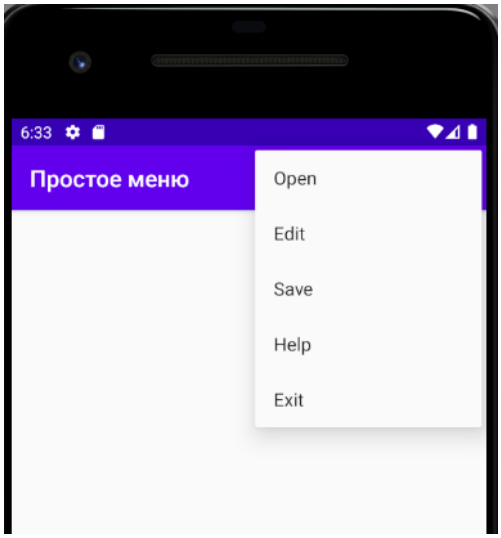

Рисунок 2 – Приложение с развернутым меню

Пока пункты меню не выполняют полезной работы. Любое нажатие на пункт просто закрывает меню без видимых последствий.

Упражнение 2. *Создание меню со значками*

В меню можно также отображать значки для каждого пункта. Значки добавляются с помощью атрибута `android:icon="@drawable/ic_menu_open"` при создании меню

1) создайте новый проект *Menu with icons* на основе Empty Activity.
2) файл компоновки для меню со значками возьмите из предыдущего примера и доработайте его.

Для отображения значков используется атрибут *android:icon*, в котором задается путь к графическому ресурсу.

Для меню можно задавать режим отображения с помощью атрибута *android:showAsAction*. Обычно значки выводятся на панель Action Bar.

Если места на Action Bar недостаточно, эти пункты меню переходят в раскрывающуюся часть меню. Но в раскрывающейся части меню может отображаться только текст без значков.

Если требуется вывести все значки на Action Bar, задается значение *app:showAsAction="always"*

Атрибут ***showAsAction*** задает режим показа элемента. Он может принимать значения:

- **never** – не показывать элемент
- **ifRoom** – показывать, если есть место
- **always** – всегда показывать

К этим значениям может быть добавлено еще одно – ***withText***. Актуально для элементов с указанной иконкой. В этом случае для элемента будет показана не только иконка, но и текст из ***title***.

Доработанный файл меню со значками представлен в листинге 1.

Листинг 1. Файл меню options.xml для проекта MenuWithIcons

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
tools:context=".MainActivity">

    <item android:id="@+id/action_1"
        android:title="Open"
        android:icon="@drawable/icon_open"
        android:orderInCategory="0"
        app:showAsAction="always" />
    <item android:id="@+id/action_2"
        android:title="Edit"
        android:icon="@drawable/icon_edit"
        android:orderInCategory="5"
        app:showAsAction="always" />
    <item android:id="@+id/action_3"
        android:title="Save"
        android:icon="@drawable/icon_save"
        android:orderInCategory="10"
        app:showAsAction="always" />
    <item android:id="@+id/action_4"
        android:title="Help"
        android:icon="@drawable/icon_help"
        android:orderInCategory="15"
```

```
        app:showAsAction="always" />
    <item android:id="@+id/action_5"
        android:title="Exit"
        android:icon="@drawable/icon_exit"
        android:orderInCategory="20"
        app:showAsAction="always" />
</menu>
```

Запустите проект на выполнение. При достаточно большом разрешении экрана в область Action Bar будут выведены все пункты меню (рис. 3).

Рисунок 3

Можно задать режим отображения значков: если на Action Bar не хватает места для отображения значка, он будет перемещаться в меню. При этом заголовок Activity всегда будет отображаться полностью. Такое поведение можно выставить, используя значение *ifRoom*: `app:showAsAction="ifRoom"`

Если запустить приложение при вертикальной ориентации экрана, то мы увидим, что из меню, в котором определены 5 пунктов, 2 первых пункта меню, Open и Save, переместились на Action Bar и отображаются в виде значков (рис. 4).

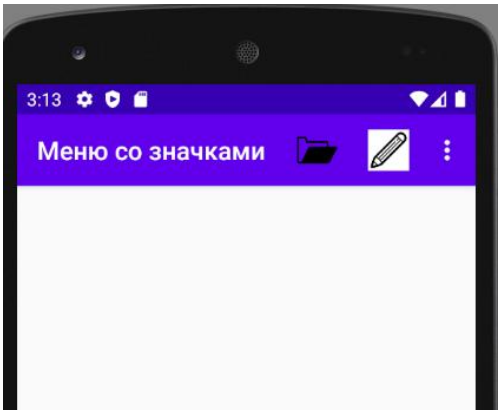

Рисунок 4 – Меню со значками

Можно также вывести на экран для каждой опции текст вместе со значком. Для этого нужно использовать константу *withText* в дополнение к *ifRoom*, соединив их знаком |:

```
app:showAsAction="ifRoom|withText"
```

Упражнение 3. *Создание меню в коде программы*

Для того, чтобы заставить пункт меню появиться в области Action Bar, при создании меню для каждого выводимого пункта надо добавить вызов метода *setShowAsAction*(), передав ему в качестве параметра константу SHOW_AS_ACTION_IF_ROOM, определенную в классе MenuItem:

```
private void CreateMenu(Menu menu) {
menu.add(Menu.NONE, IDM_OPEN, 1, "Open")
.setShowAsAction(MenuItem.SHOW_AS_ACTION_IF_ROOM); .
…
 return(super.onCreateOptionsMenu(menu));
 }
```

Метод setShowAsAction() дает указание системе Android выводить на экран пункт меню в правую область Action Bar, но при условии, что есть место для его отображения (это аналог XML-атрибута android:showAsAction). На экранах разного размера будет выведено различное количество опций меню.

Когда это меню открывается впервые, система Android вызывает метод *onCreateOptionsMenu*(), передавая в качестве параметра объект *Menu*. Этот метод необходимо реализовать в классе Activity, где происходит вызов меню, и создать информационное наполнение для объекта *Menu*. Далее необходимо вызвать метод *add*() для последовательного присоединения каждого пункта меню, например:

```
// Сначала определяем идентификаторы для создаваемых пунктов меню
private static final int IDM_OPEN = 101;
 private static final int IDM_SAVE = 102;
...
public boolean onCreateOptionsMenu(Menu menu) {
// Добавляем пункты меню
menu.add(Menu.NONE, IDM_OPEN, Menu.NONE, "Open");
menu.add(Menu.NONE, IDM_SAVE, Menu.NONE, "Save");
}
```

Метод ***add()*** принимает четыре параметра:

– *идентификатор группы* - позволяет связывать данный пункт меню с группой других пунктов этого меню;

- *идентификатор пункта* для обработчика события выбора пункта меню (определяется в коде заранее);
- *порядок расположения пункта в меню* - позволяет определять позицию пункта в меню. По умолчанию (значение `Menu.NONE` или 0) пункты меню будут отображены в соответствии с последовательностью добавления в коде;
- *заголовок* - текст пункта меню (он может также быть строковым ресурсом, если необходимо создавать локализованные приложения).

Этот метод возвращает объект `MenuItem`, который можно использовать для установки дополнительных свойств, например значка, "горячих" клавиш и других параметров настройки для этого пункта меню.

Метод `onCreateOptionsMenu`() вызывается системой только один раз - при создании меню. Если требуется обновлять меню каждый раз при его вызове из программы, необходимо определить в программе метод обратного вызова `onPrepareOptionsMenu`().

В классе, реализующем Activity, создадим меню из пяти пунктов, аналогичное тому, что вы уже создавали в XML-файле, но только теперь - в классе MainActivity, добавив к каждому пункту по вызову `setShowAsAction`(), как представлено в листинге 2.

Листинг 2. Файл класса окна приложения MainActivity.java

```
public class MainActivity extends AppCompatActivity {
    private static final int IDM_OPEN = 1001;
    private static final int IDM_SAVE = 1002;
    private static final int IDM_EDIT = 1003;
    private static final int IDM_HELP = 1004;
    private static final int IDM_EXIT = 1005;

    private ActionBar actionBar;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        actionBar = this.getActionBar();
    }
    public boolean onCreateOptionsMenu(Menu menu) {
        menu.add(Menu.NONE, IDM_OPEN, 1, "Open")
                .setIcon(R.drawable.icon_open)
                .setShowAsAction(MenuItem.SHOW_AS_ACTION_IF_ROOM);
        menu.add(Menu.NONE, IDM_SAVE, 2, "Save")
                .setIcon(R.drawable.icon_save)
                .setShowAsAction(MenuItem.SHOW_AS_ACTION_IF_ROOM);
        menu.add(Menu.NONE, IDM_EDIT, 3, "Edit")
                .setIcon(R.drawable.icon_edit)
                .setShowAsAction(MenuItem.SHOW_AS_ACTION_IF_ROOM);
        menu.add(Menu.NONE, IDM_HELP, 4, "Help")
                .setIcon(R.drawable.icon_help)
                .setShowAsAction(MenuItem.SHOW_AS_ACTION_IF_ROOM);
        menu.add(Menu.NONE, IDM_EXIT, 5, "Exit")
                .setIcon(R.drawable.icon_exit)
                .setShowAsAction(MenuItem.SHOW_AS_ACTION_IF_ROOM);
        return (super.onCreateOptionsMenu(menu));
```

```
    }
}
```

Внешний вид приложения будет такой же, как и на рис. 4.

Упражнение 4. *Обработка событий меню*

При выборе пункта меню пользователем будет вызван метод *onOptionsItemSelected*(), который необходимо определить в классе, реализующем Activity. Этот метод обратного вызова передает в программу объект *MenuItem* - пункт меню, который был выбран пользователем. Идентифицировать выбранный пункт меню можно методом *getItemId*(), возвращающим целое число, являющееся идентификатором пункта меню, который был назначен ему в методе *add*() при создании меню в *onCreateOptionsMenu*(). После идентификации пункта меню можно написать код, реализующий обработку события выбора меню.

Обработчик события выбора пункта меню будет выглядеть примерно так:

```
public boolean onOptionsItemSelected(MenuItem item) {
switch (item.getItemId() ) {
case IDM_OPEN:
...
return true;
case IDM_SAVE:
...
return true;
}
return false;
}
```

Добавим метод *onOptionsItemSelected*() в программу из листинга 2. При выборе пункта меню на экран будет выводиться уведомление, показывающее этот пункт. Код метода представлен в листинге 3.

Листинг 3. Метод onOptionsItemSelected()

```
@Override public boolean onOptionsItemSelected(MenuItem item) {
CharSequence message = "DEF";
switch (item.getItemId()) {
case IDM_OPEN:
message = "Open item selected";
break;
case IDM_SAVE:
```

```
message = "Save item selected";
break;
case IDM_HELP:
message = "Help item selected";
break;
case IDM_EDIT:
message = "Edit item selected";
break;
case IDM_EXIT:
message = "Exit item selected";
break;
default:
return false;
}
Toast toast = Toast.makeText(this, message, Toast.LENGTH_SHORT);
toast.setGravity(Gravity.CENTER, 0, 0);
 toast.show();
return true;
 }
```

Запустите проект на выполнение. Теперь при выборе одного из пунктов меню будет появляться соответствующее всплывающее сообщение.

Упражнение 5. *Добавление флажков и переключателей меню*

Для расширения функциональности в пункты меню можно добавить флажки или переключатели. Например, чтобы добавить флажок для отдельного элемента меню, необходимо использовать метод `setCheckable`().

Если есть необходимость добавить несколько пунктов меню с флажками или переключателями, целесообразно объединять их в группы меню.

...

Атрибут `android:checkableBehavior` может принимать значение `single` - это будут переключатели или `all` - тогда будет группа пунктов меню с флажками.

Если какой-либо пункт надо пометить по умолчанию, то следует для этого пункта добавить атрибут `android:checked`: `android:checked="true"`

1) создайте новый проект `MenuWithRadioGroup` на основе Empty Activity.
2) файл компоновки для меню используйте из упражнения 1.

В приложении создайте меню из 4 пунктов: Item A, Item B, Item C и Item other. Первые 3 пункта меню будут группой. В файле компоновки меню main.xml создайте группу из первых трех пунктов, а четвертый пункт оставьте независимым (листинг 4).

Листинг 4. Файл компоновки меню main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">

 <group android:checkableBehavior="single"
        android:orderInCategory="0">

<item android:id="@+id/menuitemA"
            android:title="Item A"
            android:checked="true"/>
        <item android:id="@+id/menuitemB"
            android:title="Item B" />
        <item android:id="@+id/menuitemC"
            android:title="Item C"/>
    </group>
    <item android:id="@+id/menuitemOther"
        android:title="Item other"
        android:orderInCategory="10"
        app:showAsAction="never" />
</menu>
```

Внешний вид приложения с переключателями в меню показан на рис. 5.

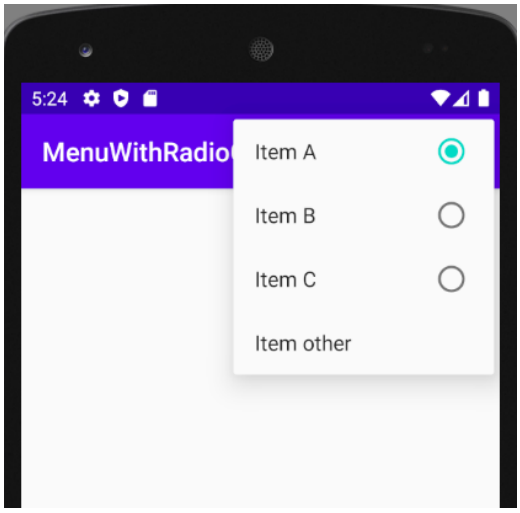

Рисунок 5 – Меню с переключателями

Если требуется отображать пункты меню с флажками, надо просто поменять значение *checkableBehavior* на *all*, как показано на рис. 6.

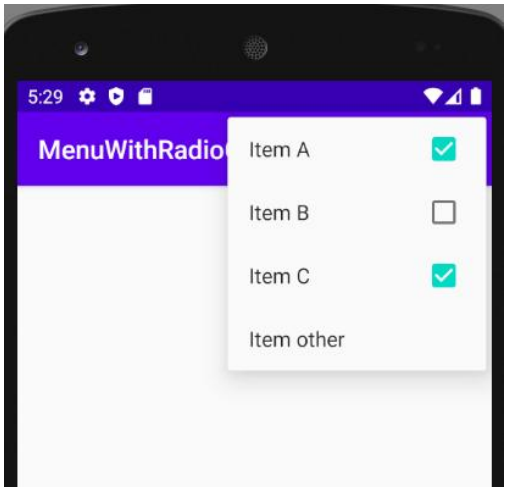

Рисунок 6 – Меню с флажками

Упражнение 6. *Добавление подменю*

Подменю можно добавить в любое меню, кроме другого подменю. Они очень полезны, когда приложение имеет много функций, которые должны быть организованы в разделы подобно пунктам в главном меню приложений для настольных систем (File, Edit, View и т. д.).

Подменю создается в методе обратного вызова `onCreateOptionsMenu`(), определяемом в классе, реализующем Activity. Подменю добавляется для уже существующего пункта меню с помощью метода `addSubMenu`(), который возвращает объект `SubMenu`.

В объект `SubMenu` можно добавить дополнительные пункты к этому меню, используя метод `add`(). Например:

```
public boolean onCreateOptionsMenu(Menu menu) {
SubMenu subMenuFile = menu.addSubMenu("File");

 subMenuFile.add(Menu.NONE, IDM_NEW, Menu.NONE, "New");
 subMenuFile.add(Menu.NONE, IDM_OPEN, Menu.NONE, "Open");
 subMenuFile.add(Menu.NONE, IDM_SAVE, Menu.NONE, "Save");
...
}
```

Самостоятельно программно создайте меню из трех пунктов: File, Edit и Help. Для первых двух пунктов определите подменю: File — New, Open, Save; Edit — Cut, Copy, Paste. Добавьте уведомление о выбранном пункте меню

Упражнение 7. *Создание контекстного меню*

Контекстное меню в Android напоминает контекстное меню в настольных системах, появляющееся при нажатии правой кнопки мыши. Меню вызывается при нажатии на объект в течение двух секунд (событие *long-tap*).

Для создания контекстного меню необходимо реализовать в классе Activity метод обратного вызова меню *onCreateContextMenu*(). В методе *onCreateContextMenu*() можно добавить пункты меню, используя один из методов add() и метод обратного вызова *onContextItemSelected*().

Код для создания контекстного меню может выглядеть следующим образом:

```
public void onCreateContextMenu(ContextMenu menu, View v,
ContextMenuInfo menuInfo) {
 super.onCreateContextMenu(menu, v, menuInfo);
menu.add(Menu.NONE, IDM_OPEN, Menu.NONE, "Open");
menu.add(Menu.NONE, IDM_SAVE, Menu.NONE, "Save");
...
}
```

При выборе пользователем пункта меню будет вызван метод *onContextItemSelected*(), который необходимо определить в классе, реализующем Activity. Этот метод передает в программу объект *MenuItem* — пункт меню, который был выбран пользователем. Для обработки события используются те же процедуры идентификации выбранного пункта меню, что и в предыдущих упражнениях.

```
public boolean onContextItemSelected(MenuItem item) {
CharSequence message;
switch (item.getItemId()) {
case IDM_OPEN: ... break;
case IDM_SAVE: ... break;
...
default: return super.onContextItemSelected(item);
} }
```

1) создайте новый проект *Context Menu Sample*.
2) разместите на экране текстовое поле со словами «Длительное нажатие для вызова контекстного меню»
3) в классе MainActivity напишите код, определив метод onCreateContextMenu(), как в листинге 5

Листинг 5. Файл класса окна приложения MainActivity.java для проекта ContextMenu

```
public class MainActivity extends AppCompatActivity {
    // Идентификаторы пунктов меню
    public static final int IDM_OPEN = 101;
    public static final int IDM_SAVE = 102;
    public static final int IDM_EDIT = 103;
    public static final int IDM_HELP = 104;
    public static final int IDM_EXIT = 105;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        final LinearLayout edit = (LinearLayout)findViewById(R.id.root);
        registerForContextMenu(edit);
    }
    @Override
    public void onCreateContextMenu(ContextMenu menu, View v, ContextMenu.ContextMenuInfo
menuInfo) {
        super.onCreateContextMenu(menu, v, menuInfo);
        menu.add(Menu.NONE, IDM_OPEN, Menu.NONE, "Open");
        menu.add(Menu.NONE, IDM_SAVE, Menu.NONE, "Save");
        menu.add(Menu.NONE, IDM_EDIT, Menu.NONE, "Edit");
        menu.add(Menu.NONE, IDM_HELP, Menu.NONE, "Help");
        menu.add(Menu.NONE, IDM_EXIT, Menu.NONE, "Exit");
    }
    // Обработчик события выбора пункта меню
    @Override
    public boolean onContextItemSelected(MenuItem item) {
        CharSequence message;
        switch (item.getItemId()) {
            case IDM_OPEN:
                message = "Open item selected";
                break;
            case IDM_SAVE:
                message = "Save item selected";
                break;
            case IDM_HELP:
                message = "Help item selected";
                break;
            case IDM_EDIT:
                message = "Edit item selected";
                break;
            case IDM_EXIT:
                message = "Exit item selected";
                break;
            default:
                return super.onContextItemSelected(item);
        }
        // Выводим уведомление о выбранном пункте меню
        Toast toast = Toast.makeText(this, message, Toast.LENGTH_SHORT);
        toast.setGravity(Gravity.CENTER, 0, 0);
        toast.show();
        return true;
    }
}
```

4) Запустите проект на выполнение. После запуска программы при нажатии левой кнопки мыши и удержании ее в течение 1–2 секунд должно появиться контекстное меню из пяти пунктов. Внешний вид приложения с контекстным меню показан на рис. 7.

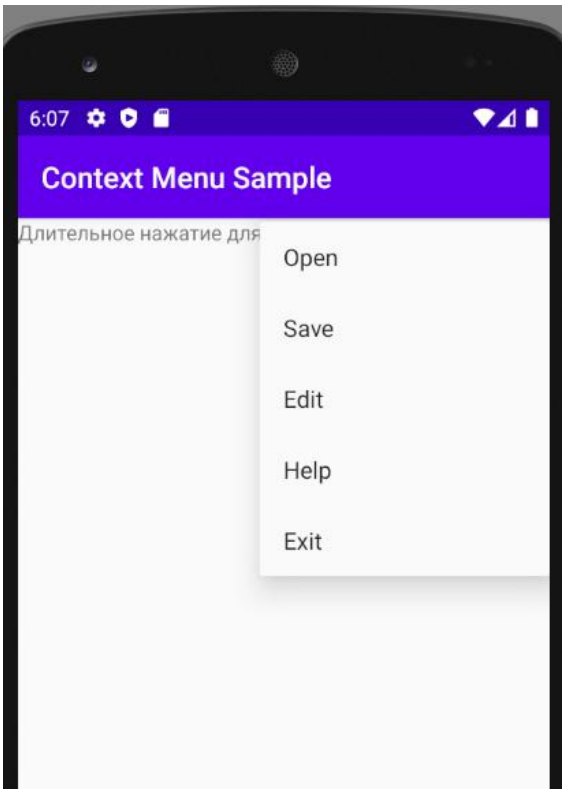

Рисунок 7 - Пример контекстного меню