## Создание главного меню приложения

### Цель работы

Познакомиться с инструментами и приемами создания меню мобильного приложения и способами взаимодействия с ним.

### Методические указания

#### Простое меню

Откроем MainActivity.java. За создание меню отвечает метод onCreateOptionsMenu. На вход ему подается объект типа Menu, в который мы и будем добавлять свои пункты.

Добавьте в Activity этот метод:

public boolean onCreateOptionsMenu(Menu menu) {

// TODO Auto-generated method stub

menu.add("menu1");

menu.add("menu2");

menu.add("menu3");

menu.add("menu4");

return super.onCreateOptionsMenu(menu);

}

Пункты меню добавляются методом add. На вход методу подается текст пункта меню. Добавим 4 пункта.

Метод onCreateOptionsMenu должен вернуть результат типа boolean. True – меню показывать, False – не показывать. Т.е. можно было бы накодить проверку какого-либо условия, и по итогам этой проверки не показывать меню передавая False. Пока нам это не нужно, поэтому поручаем этот выбор методу суперкласса, по умолчанию он возвращает True.

#### Создание обработчика нажатия

Появилось 4 пункта меню. Нажатие на них ни к чему не приводит, т.к. не реализован обработчик. Обработчиком является Activity, а метод зовется onOptionsItemSelected. На вход ему передается пункт меню, который был нажат – MenuItem. Определить, какое именно меню было нажато можно по методу getTitle. Давайте выводить всплывающее сообщение с текстом нажатого пункта меню. На выходе метода надо возвращать boolean. И мы снова предоставляем это суперклассу.

public boolean onOptionsItemSelected(MenuItem item) {

// TODO Auto-generated method stub

Toast.makeText(this, item.getTitle(), Toast.LENGTH\_SHORT).show();

return super.onOptionsItemSelected(item);

}

Определять нажатый пункт меню по тексту – это не самый лучший вариант. Далее будем делать это по ID. Но для этого надо немного по другому создавать меню.

#### Меню с группировками

Рассмотрим другую реализацию этого метода - add(int groupId, int itemId, int order, CharSequence title). У этого метода 4 параметра на вход:

- groupId - идентификатор группы, частью которой является пункт меню

- itemId - ID пункта меню

- order - для задания последовательности показа пунктов меню

- title - текст, который будет отображен

Чтоб показать как используются все эти параметры, создадим приложение. На экране будет TextView и CheckBox:

- TextView будет отображать какой пункт меню был выбран

- CheckBox будет определять показывать обычное меню или расширенное. Это будет реализовано с помощью групп меню.

Откроем main.xml, присвоим ID существующему TextView, сотрем его текст и создадим CheckBox. Код:

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

<LinearLayout

xmlns:android="http://schemas.android.com/apk/res/android"

android:orientation="vertical"

android:layout\_width="fill\_parent"

android:layout\_height="fill\_parent">

<CheckBox

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/chbExtMenu"

android:text="расширенное меню">

</CheckBox>

<TextView

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/textView">

</TextView>

</LinearLayout>

Открываем MainActivity.java и класс MainActivity заполняем следующим кодом:

public class MainActivity extends Activity {

// Элементы экрана

TextView tv;

CheckBox chb;

/\*\* Called when the activity is first created. \*/

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

// находим элементы

tv = (TextView) findViewById(R.id.textView);

chb = (CheckBox) findViewById(R.id.chbExtMenu);

}

// создание меню

@Override

public boolean onCreateOptionsMenu(Menu menu) {

// TODO Auto-generated method stub

// добавляем пункты меню

menu.add(0, 1, 0, "add");

menu.add(0, 2, 0, "edit");

menu.add(0, 3, 3, "delete");

menu.add(1, 4, 1, "copy");

menu.add(1, 5, 2, "paste");

menu.add(1, 6, 4, "exit");

return super.onCreateOptionsMenu(menu);

}

// обновление меню

@Override

public boolean onPrepareOptionsMenu(Menu menu) {

// TODO Auto-generated method stub

// пункты меню с ID группы = 1 видны, если в CheckBox стоит галка

menu.setGroupVisible(1, chb.isChecked());

return super.onPrepareOptionsMenu(menu);

}

// обработка нажатий

@Override

public boolean onOptionsItemSelected(MenuItem item) {

// TODO Auto-generated method stub

StringBuilder sb = new StringBuilder();

// Выведем в TextView информацию о нажатом пункте меню

sb.append("Item Menu");

sb.append("\r\n groupId: " + String.valueOf(item.getGroupId()));

sb.append("\r\n itemId: " + String.valueOf(item.getItemId()));

sb.append("\r\n order: " + String.valueOf(item.getOrder()));

sb.append("\r\n title: " + item.getTitle());

tv.setText(sb.toString());

return super.onOptionsItemSelected(item);

}

}

Давайте разбирать написанное. Мы используем следующие методы:

* onCreateOptionsMenu - вызывается только при первом показе меню. Создает меню и более не используется. Здесь мы добавляем к меню пункты.
* onPrepareOptionsMenu - вызывается каждый раз перед отображением меню. Здесь мы вносим изменения в уже созданное меню, если это необходимо
* onOptionsItemSelected - вызывается при нажатии пункта меню. Здесь мы определяем какой пункт меню был нажат.

В методе onCreateOptionsMenu мы добавляем 6 пунктов меню. Обратим внимание на параметры метода Add.

Первый параметр – ID группы. В первых трех пунктах он равен нулю, в оставшихся трех – 1. Т.е. пункты меню copy, paste и exit объединены в группу с ID = 1. Визуально это никак не проявляется - они не отличаются цветом или еще чем-либо. ID группы мы будем использовать в реализации onPrepareOptionsMenu.

Второй параметр – ID пункта меню. В обработчике используется для определения какой пункт меню был нажат. Будем использовать его в onOptionsItemSelected.

Третий параметр – определяет позицию пункта меню. Этот параметр используется для определения порядка пунктов при отображении меню. Используется сортировка по возрастанию, т.е. от меньшего order к большему.

Четвертый параметр – текст, который будет отображаться на пункте меню.

В метод onPrepareOptionsMenu передается объект Menu и мы можем работать с ним. В данном примере вызываем setGroupVisible. Этот метод позволяет скрывать\отображать пункты меню. На вход подается два параметра – ID группы и boolean-значение. В качестве ID группы мы пишем – 1 (та самая группа с ID = 1, в которой находятся пункты copy, paste и exit), а в качестве boolean параметра используем состояние CheckBox. Если он включен, то пункты меню (из группы с ID = 1) будут отображаться, если выключен – не будут.

В зависимости от состояния CheckBox в меню видно 3 или 6 пунктов.

Обратите внимание на порядок пунктов. Они отсортированы по параметру order по возрастанию. Если order у нескольких пунктов совпадает, то эти пункты размещаются в порядке их создания в методе onCreateOptionsMenu.

При нажатии на какой-либо пункт меню срабатывает метод onOptionsItemSelected. В нем мы выводим в TextView информацию о нажатом пункте. Можете сверить эту информацию с тем, что мы кодили при создании пунктов меню. Все параметры должны совпадать. Порядок, для удобства, я сделал такой же как и в методе add: groupId, itemId, order, title.

#### Создание меню в отдельном файле

Есть еще один, более удобный и предпочтительный способ создания меню - с использованием xml-файлов, аналогично layout-файлам при создании экрана. Чтобы получить меню, которые мы создавали программно на этом уроке, надо создать в папке res/menu файл mymenu.xml:

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

<menu

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

<item

android:id="@+id/menu\_add"

android:title="add">

</item>

<item

android:id="@+id/menu\_edit"

android:title="edit">

</item>

<item

android:id="@+id/menu\_delete"

android:orderInCategory="3"

android:title="delete">

</item>

<group

android:id="@+id/group1">

<item

android:id="@+id/menu\_copy"

android:orderInCategory="1"

android:title="copy">

</item>

<item

android:id="@+id/menu\_paste"

android:orderInCategory="2"

android:title="paste">

</item>

<item

android:id="@+id/menu\_exit"

android:orderInCategory="4"

android:title="exit">

</item>

</group>

</menu>

Если в папке res нет папки меню, создайте ее. Правой кнопкой на res, выбирайте New > Android Resource Directory, в Resource type выбирайте menu и жмите OK.

item - это пункт меню, group - группа. В атрибутах ID используем ту же схему, что и в ID экранных компонентов, т.е. пишем @+id/<your\_ID> и Eclipse сам создаст эти ID в R.java. Атрибут orderInCategory - это порядок пунктов, а title - текст.

В методе onCreateOptionsMenu нам теперь не надо вручную кодить создание каждого пункта, мы просто свяжем menu, который нам дается на вход и наш xml-файл.

public boolean onCreateOptionsMenu(Menu menu) {

getMenuInflater().inflate(R.menu.mymenu, menu);

return super.onCreateOptionsMenu(menu);

}

С помощью метода getMenuInflater мы получаем MenuInflater и вызываем его метод inflate. На вход передаем наш файл mymenu.xml из папки res/menu и объект menu. MenuInflater берет объект menu и наполняет его пунктами согласно файлу mymenu.xml.

Если захотите скрыть группу, выполняете тот же метод setGroupVisible и передаете туда R.id.group1 в качестве ID группы.

### Дополнительные задания

1. Попробуйте добавить еще несколько пунктов в меню, чтобы их стало больше шести. И обратите внимание, как они отобразятся.
2. Реализуйте меню с пунктом “выход”, при выборе которого приложение должно закрыться.
3. Модифицируйте игру “Угадай число” таким образом, чтобы в меню можно было выбирать уровень сложности, начинать новую игру и выходить из приложения.