**Меню**

**Меню**— важная часть любого приложения. Android предлагает простой интерфейс программирования для создания стандартизированных прикладных меню для приложений с разнообразной функциональностью.

**Android предлагает три базовых типа прикладных меню:**

* **Options Menu** (Меню выбора опций) — набор пунктов меню, прикрепляемый к деятельности. Меню появляется внизу экрана при нажатии клавиши <MENU> на мобильном устройстве. Для меню выбора опций дополнительно существует еще две разновидности меню:
* **Icon Menu** (Меню со значками) — расширение меню выбора опций, добавляющее значки к тексту в пункты меню. Меню может содержать максимум шесть пунктов. Этот тип меню — единственный, который поддерживает значки;
* **Expanded Menu** (Расширенное меню) — 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;
* **Context Menu** (Контекстное меню) — всплывающий список пунктов меню, которые появляются при касании сенсорного экрана в течение двух и более секунд (событие longpress);
* **Submenu** (Подменю) — всплывающий список пунктов меню, который привязан к конкретному пункту в меню выбора опций или в контекстном меню. Пункты подменю не поддерживают вложенные подменю.

**Контекстное меню**

**Контекетое 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 longpress).

**Обратите внимание:** *Пункты контекстного меню не поддерживают значки или быстрые клавиши (сочетания клавиш).*

Для создания контекстного меню необходимо реализовать в классе деятельности 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, ContextMenulnfo menulnfo) super.onCreateContextMenu(menu, menulnfo);

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 onContextltemSelected ()**, который необходимо определить в классе, реализующем деятельность. Этот метод обратного вызова 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 примерах меню.

public boolean onContextltemSelected(Menuitem item) CharSequence message;

switch (item.getltemid()) case IDM\_OPEN:

break; case IDM\_SAVE:

break;

default:

return super.onContextitemSelected(item);

Для примера приложения с контекстным меню создайте новый проект и в диалоге Create New Project введите следующие значения:

* Project name — ContextMenuApp;
* Application name — ContextMenu ample;
* Package name — сот. samples. contextmenu;
* Create Activity — ContextMenuActivity.

Откройте файл разметки и создайте структуру разметки подобно листингу:

**Файл разметки main.xml**

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

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

android:id="@+id/root"

android:orientation="vertical"

android:layout\_width="fill\_parent"

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

<TextView

android:id="@+id/view1"

android:layout\_height="wrap\_content"

android:layout\_width="wrap\_content"

android:text="Long-press for call a ContextMenu"

android:textStyle="bold"/>

</LinearLayout>

В классе ContextMenuActivity напишите код, как в листинге:

**Файл ContextMenuActivity.Java**

package com.samples.contextmenu;

import android.app.Activity;

import android.os.Bundle;

import android.view.ContextMenu;

import android.view.Gravity;

import android.view.Menu;

import android.view.MenuItem;

import android.view.View;

import android.view.ContextMenu.ContextMenuInfo;

import android.widget.LinearLayout;

import android.widget.Toast;

public class ContextMenuActivity extends Activity {

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

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

final LinearLayout edit = (LinearLayout)findViewById(R.id.root);

registerForContextMenu(edit);

}

@Override

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");

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");

}

@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);

}

Toast toast = Toast.makeText(this, message, Toast.LENGTH\_SHORT);

toast.setGravity(Gravity.CENTER, 0, 0);

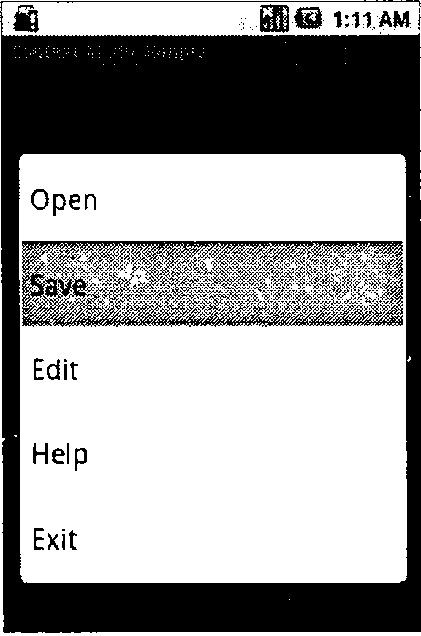
toast.show();

return true;

}

}

Запустите проект на выполнение. Внешний вид приложения с контекстным меню показан на рисунке:



*Пример контекстного меню*

**Меню выбора опций**

**Меню выбора опций** — наиболее распространенный тип меню в приложениях. Меню, как уже говорилось ранее, открывается при нажатии на мобильном устройстве клавиши <MENU>.

Когда это меню открывается впервые, система Android вызывает **метод onCreateOptionsMenu()**, передавая в качестве параметра **объект Menu**. Этот метод необходимо реализовать в классе деятельности, где происходит вызов меню, и создать информационное наполнение для объекта Menu. Меню можно определить в XMLфайле или использовать **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()**, используемый в этом примере, принимает четыре параметра:

1. **идентификатор группы** — позволяет связывать этот пункт меню с группой других пунктов этого меню (о группах меню будет рассказано позже в этой главе);
2. **идентификатор пункта** для обработчика события выбора пункта меню (определяется в коде заранее);
3. **порядок расположения пункта в меню** — позволяет нам определять позицию пункта в меню. По умолчанию (значение Menu .none или 0) они будут отображены в соответствии с последовательностью добавления в коде;
4. **заголовок**— текст на пункте 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 ().***

При выборе пункта меню пользователем будет вызван **метод onOptionsitemSelected()**, который необходимо определить в классе, реализующем деятельность. Этот метод обратного вызова передает в программу ***объект Menuitem***— пункт меню, который был выбран пользователем. Идентифицировать выбранный пункт меню можно ***методом getitemid()***, который возвращает 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;

Для меню также возможно добавить "горячие" клавиши (или сочетания клавиш) для быстрого доступа, используя символы клавиатуры. Для добавления "горячих" клавиш в меню существует несколько методов:

* **setAlphabeticShortcut (char)** —добавляет символ;
* **setNumericShortcut (int)** —добавляет число;
* **setShortcut (char, int**)—добавляет комбинацию символа и числа. Например: *setAlphabeticShortcut('q');*

Теперь при открытии меню (или при удерживании **клавиши <MENU>**) нажатие клавиши <q> выберет этот пункт меню. Эта быстрая клавиша (или сочетание клавиш) будет показана как подсказка, отображающаяся ниже имени пункта меню.

Для примера приложения с меню создайте новый проект и в диалоге Create New Project введите следующие значения:

* Project name — OptionsMenuApp;
* Application name — OptionsMenu Sample;
* Package name — сот. samples. optionsmenu;
* Create Activity — OptionsMenuActivity.

Откройте файл разметки и создайте структуру разметки подобно листингу:

**Файл разметки main.xml**

<?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">

<TextView

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:text="Press MENU button..."

android:gravity="center"

android:textStyle="bold"/>

</LinearLayout>

В **классе optionsMenuActivity** реализуйте процедуру создания и обработки пользовательского взаимодействия с меню, приведенную ранее. Полный ***код класса OptionsMenuActivity*** приведен в листинге:

**Файл OptionsMenuActivity.Java**

package com.samples.optionsmenu;

import android.app.Activity;

import android.os.Bundle;

import android.view.Gravity;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.Toast;

public class OptionsMenuActivity extends Activity {

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

public void onCreate(Bundle icicle) {

super.onCreate(icicle);

setContentView(R.layout.main);

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

menu.add(Menu.NONE, IDM\_OPEN, Menu.NONE, "Open")

.setAlphabeticShortcut('o');

menu.add(Menu.NONE, IDM\_SAVE, Menu.NONE, "Save")

.setAlphabeticShortcut('s');

menu.add(Menu.NONE, IDM\_EDIT, Menu.NONE, "Edit")

.setAlphabeticShortcut('e');

menu.add(Menu.NONE, IDM\_HELP, Menu.NONE, "Help")

.setAlphabeticShortcut('h');

menu.add(Menu.NONE, IDM\_EXIT, Menu.NONE, "Exit")

.setAlphabeticShortcut('x');

return(super.onCreateOptionsMenu(menu));

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

CharSequence message;

switch (item.getItemId()) {

case IDM\_OPEN:

message = "Open item selected";

item.setChecked(true);

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);

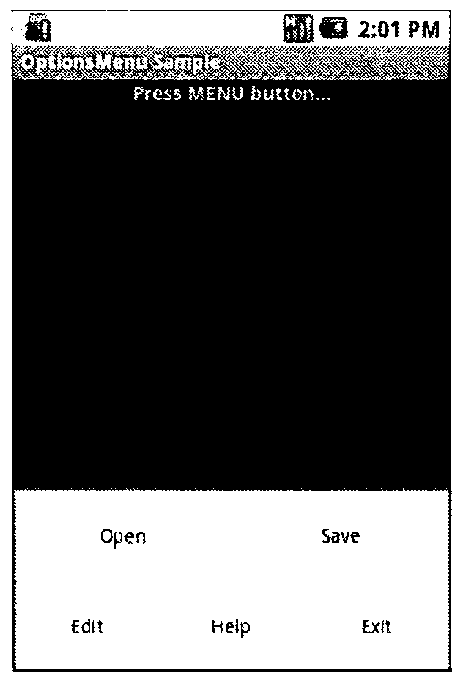
toast.show();

return true;

}

}

Запустите проект на выполнение. При нажатии клавиши <MENU> клавиатуры эмулятора на экране должно появиться меню из пяти пунктов: *Open, Save, Edit, Help, Exit* :



*Вызов меню в деятельности*

**Меню со значками**

В меню можно также отображать значки для каждого пункта. Значки добавляются методом **seticon()** При создании меню. Например:

menu.add(Menu.NONE, IDM\_OPEN, Menu.NONE, "Open")

.seticon(R.drawaWe.ic\_menu\_open);

menu.add(Menu.NONE, IDM\_SAVE, Menu.NONE, "Save")

.seticon(R.drawable.ic\_menu\_save);

В качестве примера приложения, имеющего меню со значками, создайте в Eclipse новый проект и в диалоге *Create New Project* заполните поля:

1. Project name— IconMenuApp;
2. Application name — IconMenu Sample;
3. Package name — com. samples. iconmenu;
4. Create Activity — IconMenuActivity.

Файл разметки для меню со значками оставьте таким же, как в предыдущем примере.

**Класс optionsiconMenuActivity**, реализующий меню со значками, представлен в листинге:

**Файл OptionsiconMenuActivity.java**

package com.samples.optionsiconmenu;

import android.app.Activity;

import android.os.Bundle;

import android.view.Gravity;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.Toast;

public class OptionsIconMenuActivity extends Activity {

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

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

menu.add(Menu.NONE, IDM\_OPEN, Menu.NONE, "Open")

.setIcon(R.drawable.ic\_menu\_open);

menu.add(Menu.NONE, IDM\_SAVE, Menu.NONE, "Save")

.setIcon(R.drawable.ic\_menu\_save);

menu.add(Menu.NONE, IDM\_EDIT, Menu.NONE, "Edit")

.setIcon(R.drawable.ic\_menu\_edit);

menu.add(Menu.NONE, IDM\_HELP, Menu.NONE, "Help")

.setIcon(R.drawable.ic\_menu\_help);

menu.add(Menu.NONE, IDM\_EXIT, Menu.NONE, "Exit")

.setIcon(R.drawable.ic\_menu\_exit);

return(super.onCreateOptionsMenu(menu));

}

@Override

public boolean onOptionsItemSelected(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 false;

}

Toast toast = Toast.makeText(this, message, Toast.LENGTH\_SHORT);

toast.setGravity(Gravity.CENTER, 0, 0);

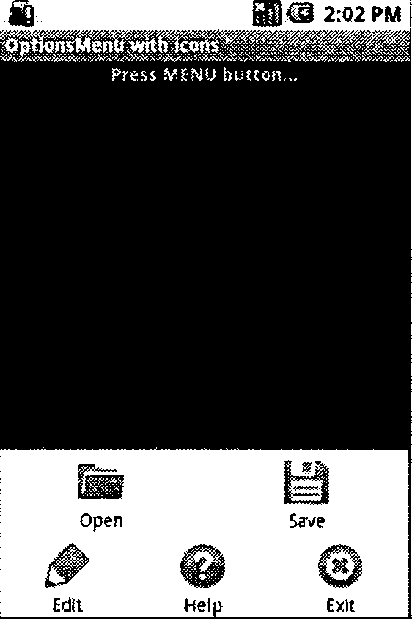
toast.show();

return true;

}

}

Запустите проект на выполнение. При нажатии клавиши **<MENU>** клавиатуры эмулятора должно появиться меню из пяти пунктов, такое же, как в предыдущем примере, но уже со значками для каждого пункта меню:



*Меню со значками*

**Расширенное меню**

**Расширенное меню,** как уже было сказано, появляется при количестве пунктов меню больше шести. Это меню добавляется автоматически системой Android при отображении меню на экране.

Для примера приложения с расширенным меню создайте новый проект и в диалоге Create New Project введите следующие значения:

* Project name — OptionsExpandedMenuApp;
* Application name — OptionsExpandedMenu Sample;
* Package name — сот. samples. optionsexpandedmenu;
* Create Activity — OptionsExpandedMenuActivity.

Файл разметки для расширенного меню оставьте таким же, как в предыдущем примере.

Для класса, реализующего деятельность, возьмите код из предыдущего примера и добавьте в него три дополнительных идентификатора для пунктов меню:

private static final int IDM\_FIND\_REPLACE 106;

private static final int IDM\_FIND\_NEXT 107;

private static final int IDM FIND PREV = 108;

В **методе onCreateOptionsMenu()** добавьте новые пункты меню (теперь в меню будет восемь пунктов):

public boolean onCreateOptionsMenu (Menu menu)

menu.add(Menu.NONE, IDM\_FIND\_REPLACE, Menu.NONE, "Find/Replace");

menu.add(Menu.NONE, IDM\_FIND\_NEXT, Menu.NONE, "Find Next");

menu.add(Menu.NONE, IDM\_FIND\_PREV, Menu.NONE, "Find Previous");

return (super. onCreateOptionsMenu (menu) );

**Обратите внимание:***Значки для дополнительных пунктов меню не добавляются, поскольку они все равно не будут отображаться в пунктах расширенного меню.*

Добавьте также в **метод onOptionsitemSelected()** обработку событий выбора новых пунктов меню. Полный код класса OptionsExpandedMenuActivity приведен в листинге:

**Файл разметки OptionsExpandedMenuActivity.Java**

package com.samples.optionsexpandedmenu;

import android.app.Activity;

import android.os.Bundle;

import android.view.Gravity;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.Toast;

public class OptionsExpandedMenuActivity extends Activity {

public static final int OPEN = 101;

public static final int SAVE = 102;

public static final int EDIT = 103;

public static final int HELP = 104;

public static final int EXIT = 105;

public static final int FIND\_REPLACE = 106;

public static final int FIND\_NEXT = 107;

public static final int FIND\_PREV = 108;

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

menu.add(Menu.NONE, OPEN, Menu.NONE, "Open")

.setIcon(R.drawable.ic\_menu\_open);

menu.add(Menu.NONE, SAVE, Menu.NONE, "Save")

.setIcon(R.drawable.ic\_menu\_save);

menu.add(Menu.NONE, EDIT, Menu.NONE, "Edit")

.setIcon(R.drawable.ic\_menu\_edit);

menu.add(Menu.NONE, HELP, Menu.NONE, "Help")

.setIcon(R.drawable.ic\_menu\_help);

menu.add(Menu.NONE, EXIT, Menu.NONE, "Exit")

.setIcon(R.drawable.ic\_menu\_exit);

menu.add(Menu.NONE, FIND\_REPLACE, Menu.NONE, "Find/Replace");

menu.add(Menu.NONE, FIND\_NEXT, Menu.NONE, "Find Next");

menu.add(Menu.NONE, FIND\_PREV, Menu.NONE, "Find Previous");

return(super.onCreateOptionsMenu(menu));

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

CharSequence message;

switch (item.getItemId()) {

case OPEN:

message = "Open item selected";

break;

case SAVE:

message = "Save item selected";

break;

case HELP:

message = "Help item selected";

break;

case EDIT:

message = "Edit item selected";

break;

case EXIT:

message = "Exit item selected";

break;

case FIND\_REPLACE:

message = "Find/Replace item selected";

break;

case FIND\_NEXT:

message = "Find Next item selected";

break;

case FIND\_PREV:

message = "Find Previous item selected";

break;

default:

return false;

}

Toast toast = Toast.makeText(this, message, Toast.LENGTH\_SHORT);

toast.setGravity(Gravity.CENTER, 0, 0);

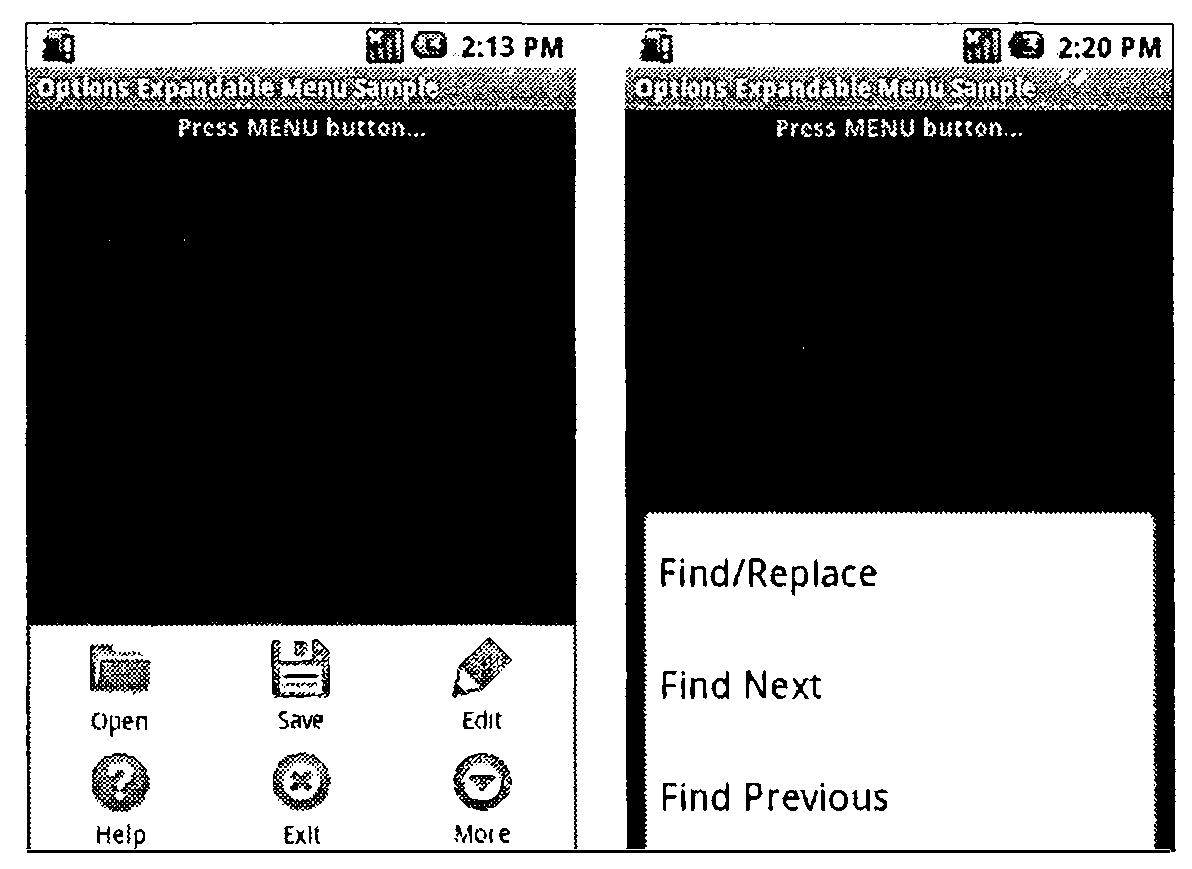
toast.show();

return true;

}

}

Запустите проект на выполнение. При нажатии клавиши <MENU> клавиатуры эмулятора должно появиться меню из пяти пунктов и дополнительным пунктом More для расширенного меню. При выборе пункта More появится расширенное меню с дополнительными пунктами.



*Расширенное меню*

**Подменю**

**Подменю** можно добавить в любое меню, кроме другого подменю. Они очень полезны, когда ваше 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()**, определяемого в классе деятельности. 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") ;

Для примера приложения с подменю создайте новый проект и в диалоге *Create New Project*введите следующие значения:

* Project name — SubMenuApp;
* Application name — SubMenu Sample;
* Package name — com. samples. submenu;
* Create Activity — SubMenuActivity.

В примере будет меню из трех пунктов: *File, Edit и Help*. Для первых двух пунктов определим подменю:

1. File — New, Open, Save;
2. Edit — Cut, Сору, Paste.

Полный код класса SubMenuActivity представлен в листинге:

**Файл SubMenuActivity.Java**

package com.samples.submenu;

import android.app.Activity;

import android.os.Bundle;

import android.view.Gravity;

import android.view.Menu;

import android.view.MenuItem;

import android.view.SubMenu;

import android.widget.Toast;

public class SubMenuActivity extends Activity {

public static final int HELP = 101;

public static final int NEW = 201;

public static final int OPEN = 202;

public static final int SAVE = 203;

public static final int CUT = 301;

public static final int COPY = 302;

public static final int PASTE = 303;

@Override

public void onCreate(Bundle icicle) {

super.onCreate(icicle);

setContentView(R.layout.main);

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

SubMenu subMenuFile = menu.addSubMenu("File");

subMenuFile.add(Menu.NONE, NEW, Menu.NONE, "New");

subMenuFile.add(Menu.NONE, OPEN, Menu.NONE, "Open");

subMenuFile.add(Menu.NONE, SAVE, Menu.NONE, "Save");

SubMenu subMenuEdit = menu.addSubMenu("Edit");

subMenuEdit.add(Menu.NONE, CUT, Menu.NONE, "Cut");

subMenuEdit.add(Menu.NONE, COPY, Menu.NONE, "Copy");

subMenuEdit.add(Menu.NONE, PASTE, Menu.NONE, "Paste");

menu.add(Menu.NONE, HELP, Menu.NONE, "Help");

return super.onCreateOptionsMenu(menu);

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

CharSequence message;

switch (item.getItemId()) {

case NEW:

message = "New item selected";

break;

case OPEN:

message = "Open item selected";

break;

case SAVE:

message = "Save item selected";

break;

case CUT:

message = "Cut item selected";

break;

case COPY:

message = "Copy item selected";

break;

case PASTE:

message = "Paste item selected";

break;

case HELP:

message = "Help item selected";

break;

default:

return false;

}

Toast toast = Toast.makeText(this, message, Toast.LENGTH\_SHORT);

toast.setGravity(Gravity.CENTER, 0, 0);

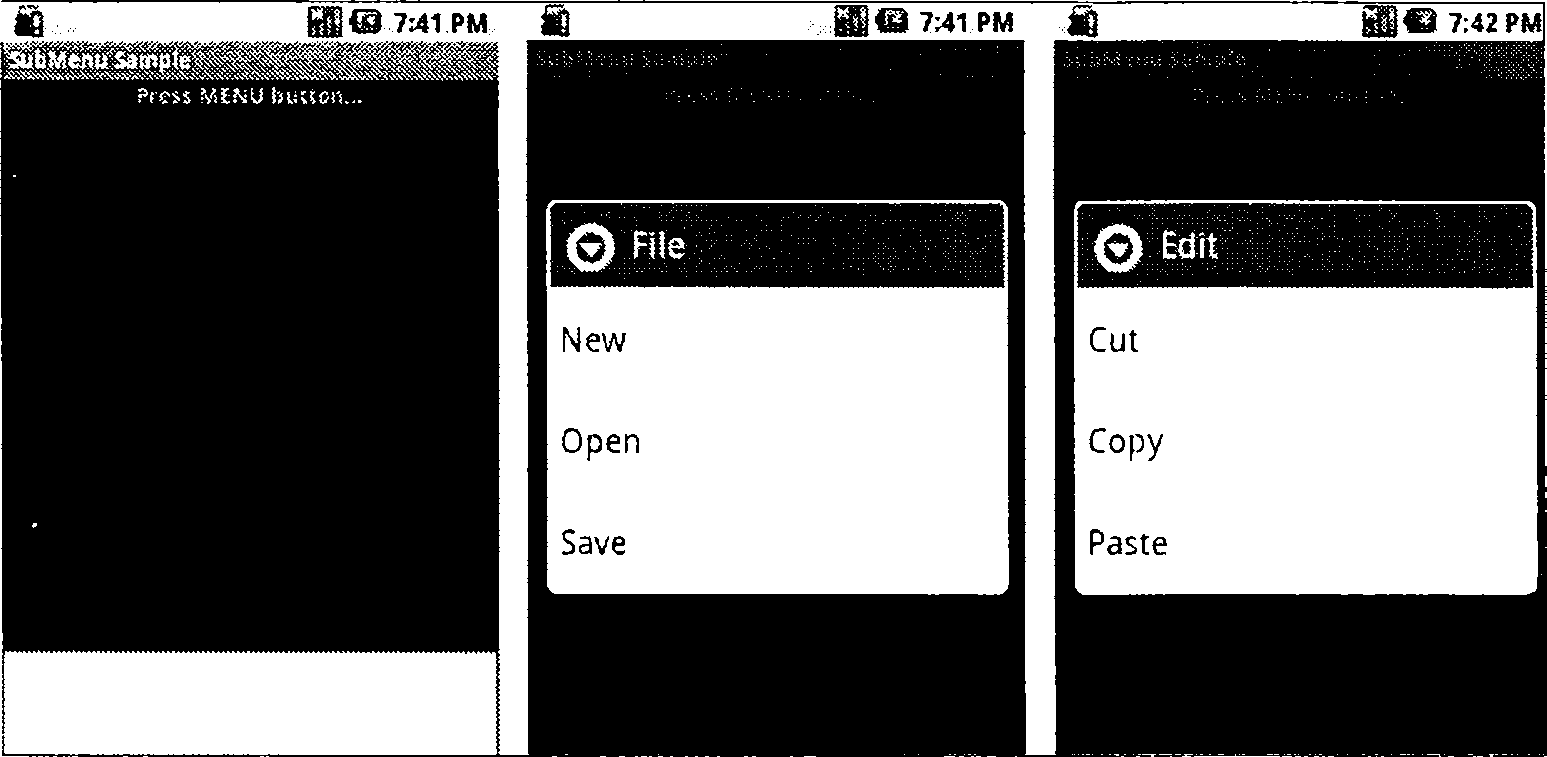
toast.show();

return true;

}

}

выводим уведомление о выбранном пункте меню Toast toast Toast.makeText(this, message, Toast.LENGTH\_SHORT); toast.setGravity(Gravity.CENTER, О, 0); toast.show(); return true;



*Меню и его подменю в приложении*

Запустите проект на выполнение. При нажатии клавиши **<MENU>**клавиатуры эмулятора должно появиться меню из трех пунктов. При выборе пункта File появится его подменю. Пункт меню Edit также имеет собственное подменю.

**Добавление флажков и переключателей в меню**

Для расширения функциональности в пункты меню возможно добавление флажков или переключателей.

Чтобы добавить флажок для отдельного элемента меню, **необходимо использовать метод setCheckable()**

Menuitem item rnenu.add(O, IDM\_FORMAT\_BOLD, О, "Bold")

item. setCheckable(true);

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

Группа меню определяется идентификатором (целым числом). Пункт меню можно добавить к группе, используя один из вариантов метода **add()**, передав ему в качестве первого параметра идентификатор группы меню. Например. пусть в коде объявлены идентификаторы для группы меню Color и элементов меню для установки цвета:

pubic static final int IDM\_COLOR\_GROUP 200;

pubic static final int IDM\_COLOR\_RED 201;

public static final int IDM\_COLOR\_GREEN 202;

public static final int IDM\_COLOR\_BLUE 203;

Тогда для создания группы меню с флажками необходимо назначить тот же самый идентификатор группы на каждый пункт меню и вызвать метод **setGroupCheckable()** для всей группы. В этом случае нет необходимости вызывать **метод setCheckable()** для каждого пункта меню:

SubMenu subMenuFile menu.addSubMenu("Color");

subMenuFile.add(IDM\_COLOR\_GROUP,IDM\_COLOR\_RED,Menu.NONE, "Red");

subMenuFile.add(IDM\_COLOR\_GROUP,IDM\_COLOR\_GREEN,Menu.NONE,"Green");

subMenuFile.add(IDM\_COLOR\_GROUP,IDM\_COLOR\_BLUE, Menu.NONE,"Blue");

subMenuFile.setGroupCheckable(IDM\_COLOR\_GROUP,true, false);

В **метод setGroupCheckable()**передаются три параметра:

* **первый** — идентификатор группы меню;
* **второй** — true, если в группе разрешены переключатели или флажки;
* **третий** — устанавливает единственный (true) или множественный (false) выбор пунктов меню. Этот параметр фактически определяет внешний вид меню — это будет меню с радиокнопками или флажками.

Для управления состоянием флажков и переключателей необходимо написать дополнительный код в обработчике события выбора пункта меню. Например, изменение состояния флажка будет выглядеть так:

public boolean onOptionsltemSelected(Menuitem item)

switch (item.getltemid())

case IDM\_COLOR\_RED:

// инвертируем состояние флажка item.setChecked(!item. isChecked());

break;

Для примера использования групп меню в приложении создайте новый проект и в диалоге *Create New Project*введите следующие значения:

* **Project name** — CheckableSubMenuApp;
* **Application name** — CheckableSubMenu Sample;
* **Package name** — com. samples. checkablesubmenu;
* **Create Activity** — CheckableSubMenuActivity.

В приложении создайте меню из трех пунктов: **Color, Font Style, Help.** Пункты **Color** и **Font Style**являются группами меню. Для этих групп определим подменю:

* **Color** — Red, Green, Blue;
* **Font Style** — Regular, Bold, Italic.

Полный **код класса CheckableSubMenuActivity** представлен в листинге:

**Файл CheckableSubMenuActivity.Java**

package com.samples.checkablesubmenu;

import android.app.Activity;

import android.os.Bundle;

import android.view.Gravity;

import android.view.Menu;

import android.view.MenuItem;

import android.view.SubMenu;

import android.widget.Toast;

public class CheckableSubMenuActivity extends Activity {

public static final int HELP = 101;

public static final int COLOR\_MENU\_GROUP = 200;

public static final int RED = 201;

public static final int GREEN = 202;

public static final int BLUE = 203;

public static final int FONT\_MENU\_GROUP = 300;

public static final int REGULAR = 301;

public static final int BOLD = 302;

public static final int ITALIC = 303;

@Override

public void onCreate(Bundle icicle) {

super.onCreate(icicle);

setContentView(R.layout.main);

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

SubMenu subMenuFile = menu.addSubMenu("Color");

subMenuFile.add(COLOR\_MENU\_GROUP, RED, Menu.NONE, "Red");

subMenuFile.add(COLOR\_MENU\_GROUP, GREEN, Menu.NONE, "Green");

subMenuFile.add(COLOR\_MENU\_GROUP, BLUE, Menu.NONE, "Blue");

subMenuFile.setGroupCheckable(COLOR\_MENU\_GROUP, true, false);

SubMenu subMenuEdit = menu.addSubMenu("Font Style");

subMenuEdit.add(FONT\_MENU\_GROUP, REGULAR, Menu.NONE, "Regular")

.setChecked(true);

subMenuEdit.add(FONT\_MENU\_GROUP, BOLD, Menu.NONE, "Bold");

subMenuEdit.add(FONT\_MENU\_GROUP, ITALIC, Menu.NONE, "Italic");

subMenuEdit.setGroupCheckable(FONT\_MENU\_GROUP, true, true);

menu.add(Menu.NONE, HELP, Menu.NONE, "Help");

return super.onCreateOptionsMenu(menu);

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

CharSequence message;

switch (item.getItemId()) {

case RED:

item.setChecked(!item.isChecked());

message = "Red item selected";

break;

case GREEN:

item.setChecked(!item.isChecked());

message = "Green item selected";

break;

case BLUE:

item.setChecked(!item.isChecked());

message = "Blue item selected";

break;

case REGULAR:

item.setChecked(true);

message = "Regular item selected";

break;

case BOLD:

item.setChecked(true);

message = "Bold item selected";

break;

case ITALIC:

item.setChecked(true);

message = "Italic item selected";

break;

case HELP:

message = "Help item selected";

break;

default:

return false;

}

Toast toast = Toast.makeText(this, message, Toast.LENGTH\_SHORT);

toast.setGravity(Gravity.CENTER, 0, 0);

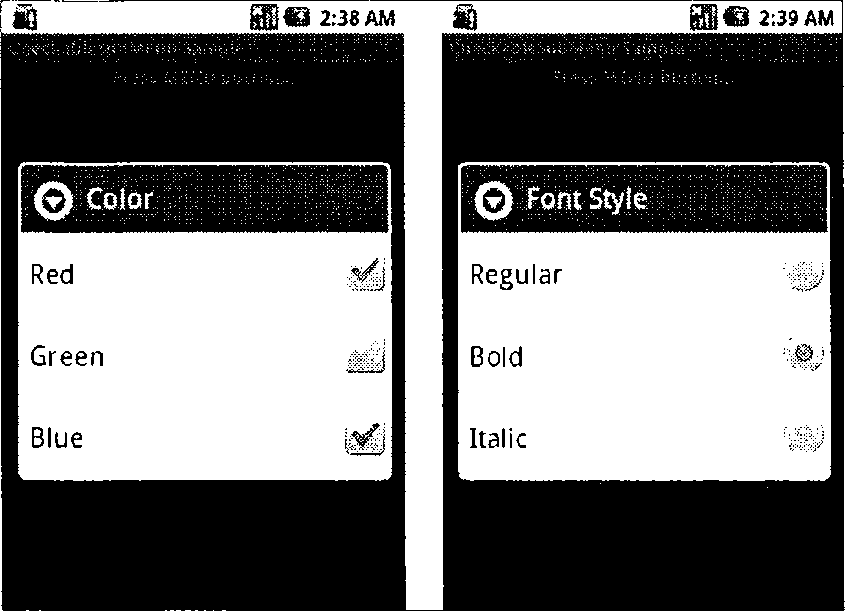
toast.show();

return true;

}

}

Запустите проект на выполнение. При нажатии клавиши **<MENU>** клавиатуры эмулятора на экране должно появиться меню из трех пунктов. При выборе ***пункта Color***появится его подменю выбора цвета с флажками, а для Font Style— подменю с радиокнопкам и. Состояние флажков и радиокнопок обрабатывается в коде программы и сохраняется при повторных вызовах меню.



*Группы подменю с флажками и радиокнопками*