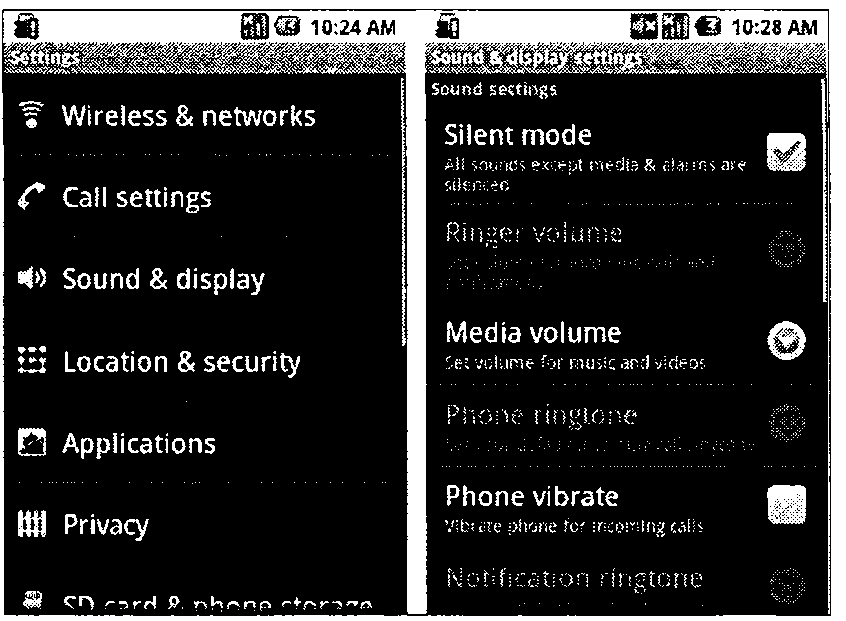
**Предпочтения**

**Предпочтение**—облегченный механизм для сохранения и восстановления пары ключзначение для примитивных типов данных. Предпочтения обычно используются для сохранения установок приложения, типа приветствия, размеров и стилей шрифта, звукового сопровождения, которые будут загружены при запуске приложения. В любом мобильном устройстве обязательно присутствует интерфейс для установки предпочтений (*на панели****Application Launcher****— значок****Settings***). Пример установки предпочтений и внешний вид окон показан на рисунке:



*Установка предпочтений для мобильного устройства*

**Android предоставляет** в распоряжение разработчиков ***Preferences Framework***, с помощью которого можно создавать индивидуальный набор предпочтений и встраивать их в приложения. ***Preferences Framework представляет собой набор классов*** для разработки.

**Предпочтения** — это отдельный экран в приложении, вызываемый из деятельности. Предпочтения определяются в отдельном XMLфайле. Корнем предпочтений в XML является элемент ***<PreferenceScreen>***, который представляет собой контейнер для предпочтений и также может содержать **дочерние элементы <PreferenceScreen>**. **Элемент <PreferenceCategory>**также является контейнерным элементом и предназначен для объединения предпочтений в группы.

Работа с этими предпочтениями и использование их в приложениях будет подробно описана далее в этом разделе.

Если набор предпочтений будет совместно употребляться с другими компонентами данного приложения, необходимо присвоить имя этому набору предпочтений. Если набор предпочтений будет использоваться только в одной деятельности, имя можно не присваивать и применять вызов ***метода Activity.getPreferences()***. Нельзя использовать предпочтения для обмена данными между приложениями (для этого существует контентпровайдер, который рассмотрен в следующем разделе).

* [CheckBoxPreference](http://androidfanclub.ru/programming/checkboxpreference)
* [EditTextPreference](http://androidfanclub.ru/programming/edittextpreference)
* [ListPreference](http://androidfanclub.ru/programming/listpreference)
* [PreferenceCategory](http://androidfanclub.ru/programming/preferencecategory)
* [PreferenceScreen](http://androidfanclub.ru/programming/preferencescreen)
* [RingtonePreference](http://androidfanclub.ru/programming/ringtonepreference)
* [Использование предпочтений](http://androidfanclub.ru/programming/%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5-%D0%BF%D1%80%D0%B5%D0%B4%D0%BF%D0%BE%D1%87%D1%82%D0%B5%D0%BD%D0%B8%D0%B9)

**CheckBoxPreference**

Это предпочтение сохраняет ***boolean-переменную*** в **SharedPreferences**. Это самое простое в использовании предпочтение. Для примера приложения с установкой предпочтений создайте новый проект и в окне *Create New Project* введите следующие значения:

* Project name — Preferences;
* Application name — Preferences Sample;
* Package name — com. samples. preferences;
* Create Activity — LaunchActivity.

Файл разметки главной деятельности main.xml будет таким же, как и в предыдущем приложении с текстовым редактором. Просто скопируйте его в новый проект.

Так как предпочтения имеют имена, к которым будут обращаться из файла предпочтений и программного кода, эти имена целесообразно хранить в файле *res/values/strings.xml*. Наше первое предпочтение назовем **OpenMode**. Поскольку в это приложение мы будем постепенно добавлять предпочтения других типов, можете сразу записать их имена в файл. Содержимое **файла strings.xml** для полного набора предпочтений, которые будут добавлены в этой главе, показано в листинге:

***Файл strings.xml***

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

<resources>

<string name="app\_name">Preferences Sample</string>

<string name="pr\_openmode">OpenMode</string>

<string name="pr\_color">Color</string>

<string name="pr\_color\_black">ColorBlack</string>

<string name="pr\_color\_red">ColorRed</string>

<string name="pr\_color\_green">ColorGreen</string>

<string name="pr\_color\_blue">ColorBlue</string>

<string name="pr\_size">Size</string>

<string name="pr\_style">Style</string>

<string name="pr\_style\_regular">StyleRegular</string>

<string name="pr\_style\_bold">StyleBold</string>

<string name="pr\_style\_italic">StyleItalic</string>

<string name="pr\_tone">Tone</string>

</resources>

Это предпочтение будет или сразу загружать файл в поле редактирования, если установлен флажок, или открывать пустое поле, если флажок не установлен. Код файла предпочтений **preferences.xml** показан в листинге:

**Файл preferences.xml**

<PreferenceScreen

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

<CheckBoxPreference

android:key="@string/pr\_openmode"

android:title="Open file"

android:summary="To open а file at start application"/>

</PreferenceScreen>

Учитывая, что предпочтения объявляются в XMLфайле, для отображения предпочтения необходимо использовать встроенную деятельность. В классе деятельности для предпочтений внутри метода обратного вызова **onCreate()** нужно только вызвать ***метод addPreferencesFromResource()*** и загрузить XML pecypc (в нашем примере — файл preferences.xml), содержащий предпочтения. **Код класса деятельности PreferencesActivity** для вызова предпочтений представлен в листинге:

**Файл класса деятельности PreferencesActlvity.java**

package com.samples.preferences;

import android.os.Bundle;

import android.preference.PreferenceActivity;

public class PreferencesActivity extends PreferenceActivity {

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

addPreferencesFromResource(R.xml.preferences);

}

}

Не забудьте добавить объявление деятельности **PreferencesActivity** в файл манифеста приложения.

**Файл AndroidManifest.xml**

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

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

package="com.samples.preferences">

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

<activity

android:name=".EditorActivity"

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

<intent-filter>

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

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

</intent-filter>

</activity>

<activity

android:name=".PreferencesActivity"

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

</activity>

</application>

</manifest>

За основу класса главной деятельности возьмем **класс EditorActivity** из предыдущего примера. В класс добавим новый пункт меню — **Settings**, который будет открывать окно предпочтений. В метод обратного вызова **onOptionsltemSelected()** добавим в ***структуру switch*** новый ***элемент case*** для вызова окна предпочтений:

case IDM\_PREF:

Intent intent new Intent();

intent.setClass(this, PreferencesActivity.class);

startActivity (intent) ;

break;

Чтение установок предпочтений производится в ***методе onResume()*** . Этот метод вызывается системой Android как во время запуска приложения, так и после закрытия окна предпочтений и возврата главной деятельности на передний план. Код для чтения установки предпочтения в ***onResume()*** должен выглядеть так:

SharedPreferences prefs

PreferenceManager.getDefaultSharedPreferences(this);

// читаем установленное значение из CheckBoxPreference

if (prefs.getBoolean(getString(R.string.pr\_openmode), false)) openFile(FILENAME);

В методе getBoolean() второй параметр (false) означает значение по умолчанию для возвращаемого значения предпочтения, если запрос на чтение установленного значения закончится неудачей.

Полный код класса главной деятельности приложения **EditorActivity:**

**Файл класса деятельности EditorActivity.java**

package com.samples.preferences;

import java.io.BufferedReader;

import java.io.InputStream;

import java.io.InputStreamReader;

import java.io.OutputStreamWriter;

import android.app.Activity;

import android.content.Intent;

import android.content.SharedPreferences;

import android.graphics.Color;

import android.graphics.Typeface;

import android.os.Bundle;

import android.preference.PreferenceManager;

import android.text.method.NumberKeyListener;

import android.view.Menu;

import android.view.MenuItem;

import android.widget.EditText;

import android.widget.Toast;

public class EditorActivity extends Activity {

static final int IDM\_OPEN = 101;

static final int IDM\_SAVE = 102;

static final int IDM\_PREF = 103;

static final int IDM\_EXIT = 104;

private final static String FILENAME = "file.txt";

private EditText mEdit;

@Override

public void onCreate(Bundle icicle) {

super.onCreate(icicle);

setContentView(R.layout.main);

mEdit = (EditText)findViewById(R.id.edit);

}

@Override

public void onResume() {

super.onResume();

SharedPreferences prefs =

PreferenceManager.getDefaultSharedPreferences(this);

// читаем установленное значение из CheckBox Preference

if (prefs.getBoolean(getString(R.string.pr\_openmode), false)) {

openFile(FILENAME);

}

int color = Color.BLACK;

if (prefs.getBoolean(getString(R.string.pr\_color\_red), false)) {

color += Color.RED;

}

if (prefs.getBoolean(getString(R.string.pr\_color\_green), false)) {

color += Color.GREEN;

}

if (prefs.getBoolean(getString(R.string.pr\_color\_blue), false)) {

color += Color.BLUE;

}

float fSize = Float.parseFloat(

prefs.getString(getString(R.string.pr\_size), "20"));

String regular = prefs.getString(getString(R.string.pr\_style), "");

int typeface = Typeface.NORMAL;

if (regular.contains("Bold")) {

typeface += Typeface.BOLD;

}

if (regular.contains("Italic")) {

typeface += Typeface.ITALIC;

}

mEdit.setTextSize(fSize);

mEdit.setTextColor(color);

mEdit.setTypeface(null, typeface);

}

@Override

public boolean onCreateOptionsMenu(Menu menu) {

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

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

.setAlphabeticShortcut('o');

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

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

.setAlphabeticShortcut('s');

menu.add(Menu.NONE, IDM\_PREF, Menu.NONE, "Settings")

.setIcon(R.drawable.ic\_menu\_preferences)

.setAlphabeticShortcut('t');

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

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

.setAlphabeticShortcut('x');

return(super.onCreateOptionsMenu(menu));

}

@Override

public boolean onOptionsItemSelected(MenuItem item) {

switch (item.getItemId()) {

case IDM\_OPEN:

openFile(FILENAME);

break;

case IDM\_SAVE:

saveFile(FILENAME);

break;

case IDM\_PREF:

Intent i = new Intent();

i.setClass(this, PreferencesActivity.class);

startActivity(i);

break;

case IDM\_EXIT:

finish();

break;

default:

return false;

}

return true;

}

private void openFile(String fileName) {

try {

InputStream inStream = openFileInput(FILENAME);

if (inStream != null) {

InputStreamReader tmp =

new InputStreamReader(inStream);

BufferedReader reader = new BufferedReader(tmp);

String str;

StringBuffer buffer = new StringBuffer();

while ((str = reader.readLine()) != null) {

buffer.append(str + "\n");

}

inStream.close();

mEdit.setText(buffer.toString());

}

}

catch (Throwable t) {

Toast.makeText(getApplicationContext(),

"Exception: " + t.toString(), Toast.LENGTH\_LONG)

.show();

}

}

private void saveFile(String FileName) {

try {

OutputStreamWriter outStream =

new OutputStreamWriter(openFileOutput(FILENAME, 0));

outStream.write(mEdit.getText().toString());

outStream.close();

}

catch (Throwable t) {

Toast.makeText(getApplicationContext(),

"Exception: " + t.toString(), Toast.LENGTH\_LONG)

.show();

}

}

class FloatKeyListener extends NumberKeyListener {

private static final String CHARS="0123456789-.";

protected char[] getAcceptedChars() {

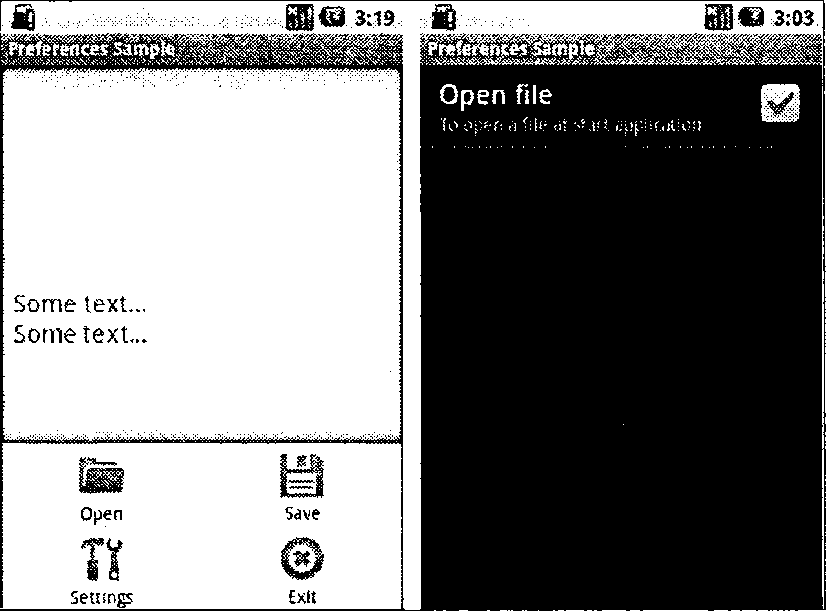
return(CHARS.toCharArray());

}

}

}

Скомпилируйте и запустите проект. В результате у вас получилась записная книжка с пока единственной опцией установки предпочтений.



*Установка предпочтенияфлажка CheckBoxPreference*

**EditTextPreference**

**Каркас предпочтений в Android** предоставляет также предпочтение с текстовым полем — ***EditTextPreference***. Это предпочтение позволяет фиксировать текст, вводимый пользователем в свободной форме. Чтобы продемонстрировать работу этого элемента, добавим в наше приложение дополнительные возможности — в текстовом поле пользователь будет устанавливать размер шрифта для редактора.

Файл предпочтений для нашего приложения с дополнительным элементом <EditTextPreference> представлен в листинге:

**Файл preferences.xml**

<PreferenceScreen

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

<CheckBoxPreference

android:key="@string/pr\_openmode"

android:title="Open file"

android:summary="To open а file at start application"/>

<EditTextPreference

android:key="@string/pr\_size"

android:title="Text Size"

android:summary="Set text size"

android:defaultValue="l4"

android:dialogTitle="Enter text size (from 10 to 32)"/>

</PreferenceScreen>

В **метод onResume()** добавим код для чтения установленного значения размера шрифта:

Метод обратного вызова **onResume()** в ***классе EditorAotivity***

@Override

public void onResume() super.onResume();

SharedPreferences prefs

PreferenceManager.getDefaultSharedPreferences(this);

//читаем открытия файла из CheckBoxPreference

if (prefs.getBoolean(getString(R.string.pr\_openmode),false))

openFile(FILENAME);

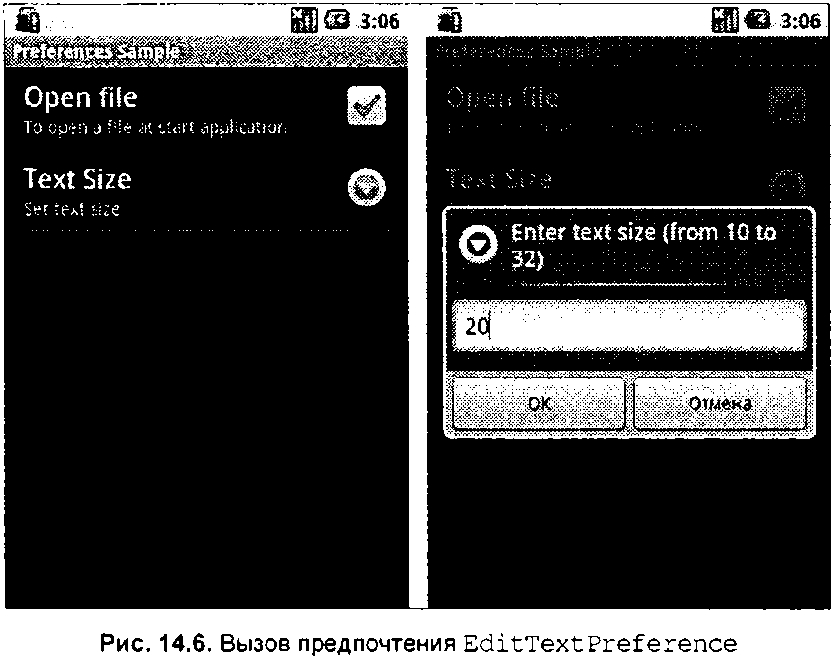
//читаем размер текста из EditTextPreference float fSize Float.parseFloat(

prefs.getString(getString(R.string.pr\_size), "20"));

// меняем настройки в EditText mEdit.setTextSize(fSize);

Скомпилируйте и запустите проект. Теперь в нашем редакторе появилась опция установки размера текста в виде диалогового окна с текстовым полем ввода.

Поскольку в **текстовое поле EditTextPreference** разрешен свободный ввод любого текста, желательно проверять пользовательский ввод. Попробуйте усовершенствовать приложение, добавив проверку правильности вводимых данных.



**ListPreference**

**Предпочтение ListPreference** представляет собой диалоговое окно со списком. Для формирования предпочтения требуется строковый ресурс для заголовка диалогового окна и массив строк для списка значений. Индекс выбранной строки списка определяет, какое значение сохранено как предпочтение в **SharedPreferences**.

Для нашего приложения сделаем список для выбора стиля текста. В списке будет четыре опции: Regular, Bold, Italic, Bold+Italic. Для массива значений **списка ListPreference** необходимо создать новый ***файл arrays.xml*** в каталоге *res/values/*. Содержимое файла представлено в листинге.

**Файл arrays.xml**

<?xml version="1.0" encoding="utf8"?>

<resources>

<stringarray name="text\_style">

<item>Regular</item>

<item>Bold</item>

<item>Italic</item>

<item>Bold+Italic</item>

</stringarray>

</resources>

В **файл preferences.xml** добавим дополнительный ***элемент <ListPreference>***, в котором определим атрибуты заголовка окна, привязку к массиву значений и значение по умолчанию:

**Файл preferences.xml**

<PreferenceScreen

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

<CheckBoxPreference

android:key="@string/pr\_openmode"

android:title="Open file"

android:summary="To open а file at start application"/>

<EditTextPreference

android:key="@string/pr\_size"

android:title="Text Size"

android:summary="Set text size"

android:defaultValue="14"

android:dialogTitle="Enter text size (from 10 to 32)"/>

<ListPreference

android:key="@string/pr\_style"

android:title="Text Style"

android:summary="Set text style"

android:defaultValue="l"

android:entries="@array/text\_style"

android:entryValues="@array/text\_style"

android:dialogTitle="Choose text style"/>

</PreferenceScreen>

В листинге приводится код метода onResume().

@Override

public void onResume() super.onResume();

SharedPreferences prefs

PreferenceManager.getDefaultSharedPreferences(this);

// читаем открытия файла из CheckBoxPreference

if (prefs.getBoolean(

getString(R.string.pr\_openmode),false)){openFile(FILENAME);

// читаем размер текста из EditTextPreference

float fSize Float.parseFloat(

prefs.getString(getString(R.string.pr\_size), "20"));

// читаем стили текста из ListPreference

String regular prefs.getString(

getString(R.string.pr\_style),int typeface Typeface.NORMAL;

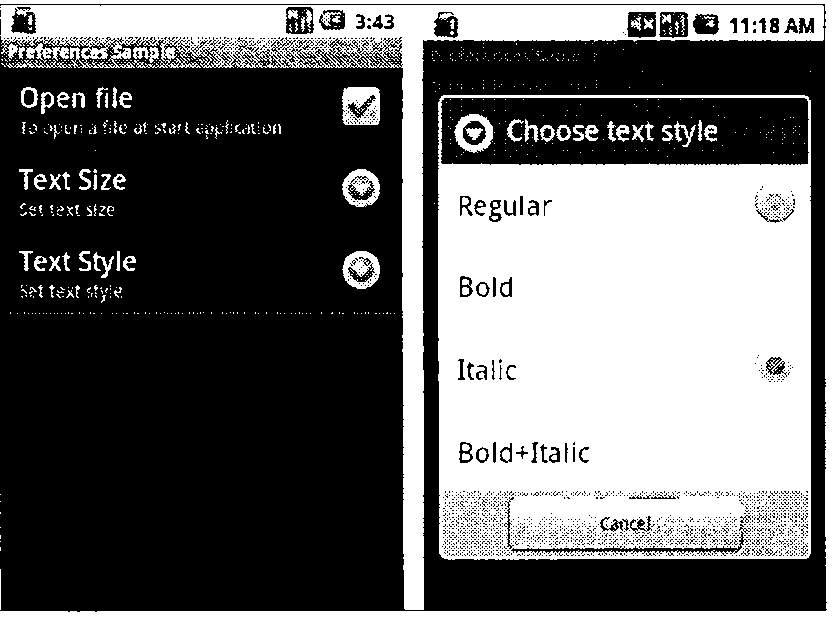
if (regular.contains("Bold")) typeface += Typeface.BOLD;

if (regular.contains ("Italic")) typeface += Typeface.ITALIC;

меняем настройки в EditText mEdit.setTextSize(fSize);

mEdit.setTypeface(null, typeface);

Флажок может быть включен или выключен пользователем. Теперь при запуске приложения и выборе ***опции TextStyle*** появляется диалог выбора предпочтения для стиля текста. Обратите внимание, что в диалоговом окне нет кнопки сохранения, только кнопка Cancel. Изменения сохраняются сразу при выборе опции списка. Использование ListPreference для выбора стиля текста.



*Использование ListPreference для выбора стиля текста*

**PreferenceCategory**

Если приложение содержит много пользовательских предпочтений, иметь их всех в одном большом списке может стать неудобным для восприятия. **Preferences Framework** позволяет группировать предпочтения, распределяя их по категориям.

Категории добавляются через ***элемент <PreferenceCategory>*** в *XMLфайле* предпочтений и используются для группировки связанных предпочтений. Вместо того, чтобы иметь все предпочтения как дочерние записи **корневого элемента <PreferenceScreen>**, можно поместить в *XMLфайл* предпочтений ***дополнительные элементы <PreferenceCategory>*** под корневым элементом **<PreferenceScreen>**, а затем установить предпочтения в соответствующие категории.

Файл предпочтений, сгруппированный по категориям предпочтений, представлен в листинге.

**Файл preferences.xml**

<PreferenceScreen

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

<PreferenceCategory android:title="Open file mode preferences">

<CheckBoxPreference

android:key="@string/pr\_openmode"

android:title="Open file"

android:summary="То open a file at start application"/>

</PreferenceCategory>

<PreferenceCategory android:title="Text preferences">

<EditTextPreference

android:key="@string/pr\_size"

android:title="Text Size"

android:summary="Set text size"

android:defaultValue="14"

android:dialogTitle="Enter text size (from 10 to 32)"/>

<ListPreference

android:key="@string/pr\_style"

android:title="Text Style"

android:summary="Set text style"

android:defaultValue="1"

android:entries="@array/text\_style"

android:entryValues="@array/text\_style"

android:dialogTitle="Choose text style"/>

</PreferenceCategory>

<PreferenceCategory android:title="Other Preferences">

<RingtonePreference

android:key="@string/pr\_tone"

android:title="Tone"

android:showDefault="true"

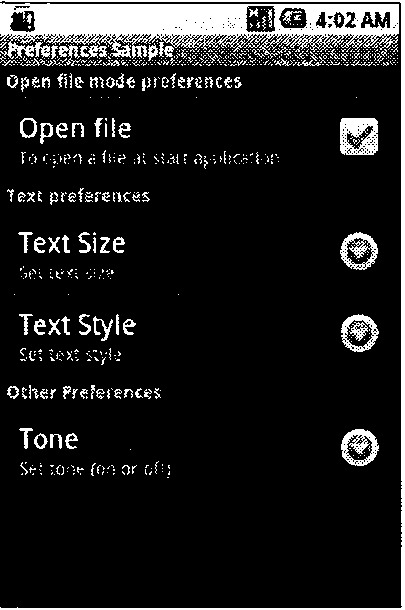
android:showSilent="true"

android:summary="Set tone (on or off)"/>

</PreferenceCategory>

</PreferenceScreen>

Результатом применения **предпочтения <PreferenceCategory>** является список элементов предпочтений, сгруппированных по категориям. Визуально это добавляет разделитель с заголовком категории между группами предпочтений:



**PreferenceScreen**

**Корневой элемент <PreferenceScreen>** позволяет вложение ***дочерних элементов <PreferenceScreen>***. Любые дочерние записи **вложенного элемента <PreferenceScreen>**будут отображаться на отдельном экране. ***Родительский экран <PreferenceScreen>*** в этом случае будет отображать в своем списке предпочтений вход для запуска дочернего экрана предпочтений.

**Дополненный файл preferences.xml с дочерним окном предпочтений выбора цвета текста. Файл preferences.xml.**

<PreferenceScreen

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

<PreferenceCategory android:title="Open file mode preferences">

<CheckBoxPreference

android:key="@string/pr\_openmode"

android:title="Open file"

android:summary="То open a file at start application"/>

</PreferenceCategory>

<PreferenceCategory android:title="Text preferences">

<PreferenceScreen

android:key="@string/pr\_color"

android:title="Text Color"

android:summary="Set text color">

<CheckBoxPreference

android:key="@string/pr\_color\_black"

android:title="Black"

android:defaultValue="true"

android:summary="Set black color"/>

<CheckBoxPreference

android:key="@string/pr\_color\_red"

android:title="Red"

android:summary="Set red color"/>

<CheckBoxPreference

android:key="@string/pr\_color\_green"

android:title="Green"

android:summary="Set green color"/>

<CheckBoxPreference

android:key="@string/pr\_color\_blue"

android:title="Blue"

android:summary="Set blue color"/>

</PreferenceScreen>

<EditTextPreference

android:key="@string/pr\_size"

android:title="Text Size"

android:summary="Set text size"

android:defaultValue="14"

android:dialogTitle="Enter text size (from 10 to 32)"/>

<ListPreference

android:key="@string/pr\_style"

android:title="Text Style"

android:summary="Set text style"

android:defaultValue="1"

android:entries="@array/text\_style"

android:entryValues="@array/text\_style"

android:dialogTitle="Choose text style"/>

</PreferenceCategory>

<PreferenceCategory android:title="Other Preferences">

<RingtonePreference

android:key="@string/pr\_tone"

android:title="Tone"

android:showDefault="true"

android:showSilent="true"

android:summary="Set tone (on or off)"/>

</PreferenceCategory>

</PreferenceScreen>

Для экрана предпочтений выбора цвета текста необходимо добавить код обработки выбора цвета в **метод onResume()** **класса EditorActivity**. Выбор нескольких цветов из группы суммирует значения каждого выбранного цвета, позволяя получить дополнительные цвета текста.

Измененный **метод onResume() класса Edi torActivity** представлен в листинге.

Метод обратного вызова onResume() в классе **EditorActivity**

@Override

public void onResume() {

super.onResume();

SharedPreferences prefs =

PreferenceManager.getDefaultSharedPreferences(this);

// читаем установленное значение из CheckBox Preference

if (prefs.getBoolean(getString(R.string.pr\_openmode), false)) {

openFile(FILENAME);

}

int color = Color.BLACK;

if (prefs.getBoolean(getString(R.string.pr\_color\_red), false)) {

color += Color.RED;

}

if (prefs.getBoolean(getString(R.string.pr\_color\_green), false)) {

color += Color.GREEN;

}

if (prefs.getBoolean(getString(R.string.pr\_color\_blue), false)) {

color += Color.BLUE;

}

float fSize = Float.parseFloat(

prefs.getString(getString(R.string.pr\_size), "20"));

String regular = prefs.getString(getString(R.string.pr\_style), "");

int typeface = Typeface.NORMAL;

if (regular.contains("Bold")) {

typeface += Typeface.BOLD;

}

if (regular.contains("Italic")) {

typeface += Typeface.ITALIC;

}

mEdit.setTextSize(fSize);

mEdit.setTextColor(color);

mEdit.setTypeface(null, typeface);

}

Окончательный вид приложения со всеми созданными в этой главе предпочтениями и отдельным входом Text Color для дочернего окна предпочтений показан на рисунке:



*Добавление дочернего контейнера PreferenceScreen для выбора цвета текста*

# RingtonePreference

**Предпочтение RingtonePreference**— это специализированное предпочтение для установки мелодии звонка. В нашем приложении в использовании этого предпочтения особой необходимости нет, мы его добавим только для примера.

**Файл preferences.xml**

<PreferenceScreen

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

<PreferenceCategory android:title="Open file mode preferences">

<CheckBoxPreference

android:key="@string/pr\_openmode"

android:title="Open file"

android:summary="То open a file at start application"/>

<EditTextPreference

android:key="@string/pr\_size"

android:title="Text Size"

android:summary="Set text size"

android:defaultValue="14"

android:dialogTitle="Enter text size (from 10 to 32)"/>

<ListPreference

android:key="@string/pr\_style"

android:title="Text Style"

android:summary="Set text style"

android:defaultValue="1"

android:entries="@array/text\_style"

android:entryValues="@array/text\_style"

android:dialogTitle="Choose text style"/>

</PreferenceCategory>

<PreferenceCategory android:title="Other Preferences">

<RingtonePreference

android:key="@string/pr\_tone"

android:title="Tone"

android:showDefault="true"

android:showSilent="true"

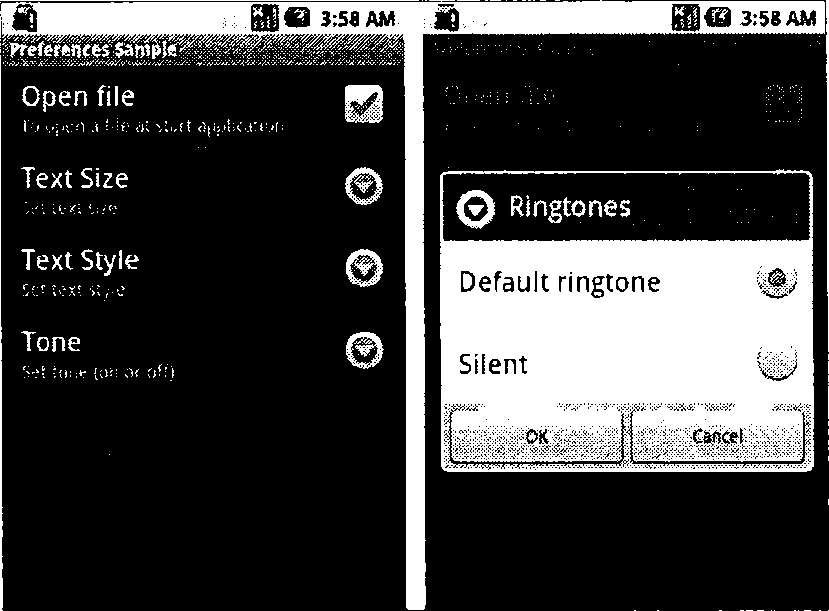
android:summary="Set tone (on or off)"/>

</PreferenceCategory>

</PreferenceScreen>

**Предпочтение <RingtonePreference>**предоставляет диалоговое окно выбора мелодии звонка со списком опций .

Список в диалоговом окне отображает мелодии для звонка, уведомлений, тонового набора, доступные на мобильном устройстве. При необходимости добавить тихий режим в ***элементе <RingtonePreference>*** предусмотрен атрибут **android: showSilent,** если его значение выставить в ***true,*** в списке мелодий появится дополнительная опция Silent.



Список ***RingtonePreference***

**Использование предпочтений**

Android позволяет действиям и приложениям сохранять предпочтение в виде пары ключ-значение. В предпочтении можно сохранять любые данные, если они имеют **тип string** или являются примитивными типами данных (**boolean, int**и т. д.).

Предпочтение может быть доступно для единственной деятельности, а может быть общим для всех деятельностей приложения. Предпочтение возможно также сделать общим для нескольких приложений.

**Для получения доступа к предпочтению в коде приложения используются три метода:**

* **getPreferences()** — внутри деятельности, чтобы обратиться к определенному для деятельности предпочтению;
* **getSharedPreferences()** — внутри деятельности, чтобы обратиться к предпочтению на уровне приложения;
* **getDefaul tSharedPreferences()** — из объекта PreferencesManager, чтобы получить общедоступное предпочтение, предоставляемое системой Android.

Первые два берут параметр режима безопасности, обычно установленный в 0. **Метод getSharedPreferences()** также принимает название ряда предпочтения — **getPreferences()**. Метод **getDefaultSharedPreferences()** в качестве параметра принимает объект **Context**(например, текущей деятельности).

Все эти методы возвращают ***экземпляр класса SharedPreferences***, из которого можно получить соответствующее предпочтение с помощью ряда методов:

* getBoolean(String key, boolean defValue);
* getFloat (String key, float defValue);
* getint (String key, int defValue);
* getLong (String key, long defValue);
* getString (String key, String defValue).

Второй параметр в **методах get()** — это значение по умолчанию, возвращаемое при невозможности прочитать выбранное значение предпочтения.

Например, так можно получить значение для предпочтения с***типом boolean***:

SharedPreferences prefs

PreferenceManager.getDefaultSharedPreferences(this);

boolean val prefs.getBoolean(getString(R.string.pref\_item),false))

Создание предпочтений для приложения мы опробуем на практике, взяв за основу предыдущий проект для чтениязаписи файла, и постепенно будем совершенствовать его, добавляя различные виды предпочтений.