**Создание и получение настроек**

Нередко приложению требуется сохранять небольшие кусочки данных для дальнейшего использования, например, данные о пользователе, настройки конфигурации и т.д. Для этого в Android существует концепция Preferences или настройки. Настройки представляют собой группу пар ключ-значение, которые используются приложением.

В качестве значений могут выступать данные следующих типов: Boolean, Float, Integer, Long, String, набор строк.

Настройки общими для всех activity в приложении, но также могут быть и настройки непосредственно для отдельных activity

Настройки хранятся в xml-файлах в незашифрованном виде в локальном хранилище. Они невидимы, поэтому для простого пользователя недоступны.

При работе с настройками следует учитывать следующие моменты. Так как они хранятся в незашифрованном виде, то не рекомендуется сохранять в них чувствительные данные типа пароля или номеров кредитных карт. Кроме того, они редставляют данные, ассоцииованные с приложением, и через панель управления приложением в Настроках ОС пользователь может удалить эти данные.

**Общие настройки**

Для работы с разделяемыми настройками в классе Activity (точнее в его базовом классе Context) имеется метод **getSharedPreferences()**:

import android.content.SharedPreferences;

//...........................

SharedPreferences settings = getSharedPreferences("PreferencesName", MODE\_PRIVATE);

Первый параметр метода указывает на название настроек. В данном случае название - "PreferencesName". Если настроек с подобным названием нет, то они создаются при вызове данного метода. Второй параметр указывает на режим доступа. В данном случае режим описан константой MODE\_PRIVATE

Класс android.content.SharedPreferences предоставляет ряд методов для управления настройками:

* contains(String key): возвращает true, если в настройках сохранено значение с ключом key
* getAll(): возвращает все сохраненные в настройках значения
* getBoolean (String key, boolean defValue): возвращает из настроек значение типа Boolean, которое имеет ключ key. Если элемента с таким ключом не окажется, то возвращается значение defValue, передаваемое вторым параметром
* getFloat(String key, float defValue): возвращает значение типа float с ключом key. Если элемента с таким ключом не окажется, то возвращается значение defValue
* getInt(String key, int defValue): возвращает значение типа int с ключом key
* getLong(String key, long defValue): возвращает значение типа long с ключом key
* getString(String key, String defValue): возвращает строковое значение с ключом key
* getStringSet(String key, Set<String> defValues): возвращает массив строк с ключом key
* edit(): возвращает объект SharedPreferences.Editor, который используется для редактирования настроек

Для управления настройками используется объект класса **SharedPreferences.Editor**, возвращаемый метод edit(). Он определяет следующие методы:

* clear(): удаляет все настройки
* remove(String key): удаляет из настроек значение с ключом key
* putBoolean(String key, boolean value): добавляет в настройки значение типа boolean с ключом key
* putFloat(String key, float value): добавляет в настройки значение типа float с ключом key
* putInt(String key, int value): добавляет в настройки значение int с ключом key
* putLong(String key, long value): добавляет в настройки значение типа long с ключом key
* putString(String key, String value): добавляет в настройки строку с ключом key
* putStringSet(String key, Set<String> values): добавляет в настройки строковый массив
* commit(): подтверждает все изменения в настройках
* apply(): также, как и метод commit(), подтверждает все изменения в настройках, однако измененный объект SharedPreferences вначале сохраняется во временной памяти, и лишь затем в результате асинхронной операции записывается на мобильное устройство

Рассмотрим пример сохранения и получения настроек в приложении. Определим в файле **activity\_main.xml** следующий пользовательский интерфейс:

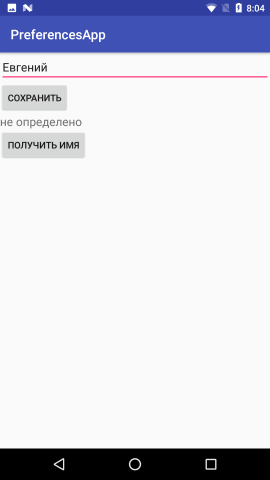
|  |
| --- |
| <LinearLayout xmlns:android="<http://schemas.android.com/apk/res/android>"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      android:orientation="vertical">        <EditText          android:id="@+id/nameBox"          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content"          android:hint="Введите имя"/>      <Button          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:text="Сохранить"          android:onClick="saveName"/>        <TextView          android:id="@+id/nameView"          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content" />      <Button          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:text="Получить имя"          android:onClick="getName"/>  </LinearLayout> |

На экране будут две кнопки - для сохранения и для вывода ранее сохраненного значения, а также поле для ввода и текстовое поля ля вывода сохраненной настройки.

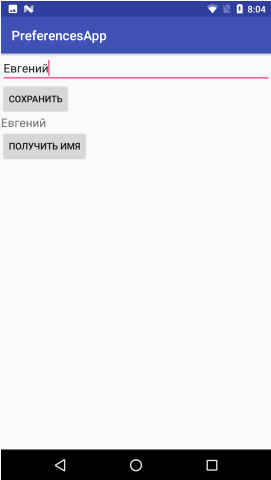
Определим методы обработчики кнопок в классе **MainActivity**:

|  |
| --- |
| package com.example.eugene.preferencesapp;    import android.content.SharedPreferences;  import android.support.v7.app.AppCompatActivity;  import android.os.Bundle;  import android.view.View;  import android.widget.EditText;  import android.widget.TextView;    public class MainActivity extends AppCompatActivity {        private static final String PREFS\_FILE = "Account";      private static final String PREF\_NAME = "Name";      SharedPreferences settings;      @Override      protected void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.activity\_main);          settings = getSharedPreferences(PREFS\_FILE, MODE\_PRIVATE);      }        public void saveName(View view) {          // получаем введенное имя          EditText nameBox = (EditText) findViewById(R.id.nameBox);          String name = nameBox.getText().toString();          // сохраняем его в настройках          SharedPreferences.Editor prefEditor = settings.edit();          prefEditor.putString(PREF\_NAME, name);          prefEditor.apply();      }        public void getName(View view) {          // получаем сохраненное имя          TextView nameView = (TextView) findViewById(R.id.nameView);          String name = settings.getString(PREF\_NAME,"не определено");          nameView.setText(name);      }  } |

При отсутствии настроек при попытке их получить, приложение выведет значение по умолчанию:



Теперь сохраним и выведем заново сохраненное значение:



Нередко возникает задача автоматически сохранять вводимые данные при выходе пользователя из activity. Для этого мы можем переопределить метод onPause:

|  |
| --- |
| package com.example.eugene.preferencesapp;    import android.content.SharedPreferences;  import android.support.v7.app.AppCompatActivity;  import android.os.Bundle;  import android.view.View;  import android.widget.EditText;    public class MainActivity extends AppCompatActivity {        private static final String PREFS\_FILE = "Account";      private static final String PREF\_NAME = "Name";      EditText nameBox;      SharedPreferences settings;      SharedPreferences.Editor prefEditor;      @Override      protected void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.activity\_main);            nameBox = (EditText) findViewById(R.id.nameBox);          settings = getSharedPreferences(PREFS\_FILE, MODE\_PRIVATE);            // получаем настройки          String name = settings.getString(PREF\_NAME,"");          nameBox.setText(name);      }        @Override      protected void onPause(){          super.onPause();            EditText nameBox = (EditText) findViewById(R.id.nameBox);          String name = nameBox.getText().toString();          // сохраняем в настройках          prefEditor = settings.edit();          prefEditor.putString(PREF\_NAME, nameBox.getText().toString());          prefEditor.apply();      }        public void saveName(View view) {        }        public void getName(View view) {        }  } |

**Приватные настройки**

Кроме общих настроек каждая activity может использовать приватные, к которым доступ из других activity будет невозможен. Для получения настроек уровня activity используется метод **getPreferences(MODE\_PRIVATE)**:

|  |  |
| --- | --- |
| 1  2  3 | import android.content.SharedPreferences;  //........................  SharedPreferences settings = getPreferences(MODE\_PRIVATE); |

То есть в отличие от общих настроек здесь не используется название группы настроек в качестве первого параметра, как в методе getSharedPreferences(). Однако вся остальная работа по добавлению, получению и изменению настроек будет аналогична работает с общими настройками.