**Чтение и сохранение файлов**

Работа с настройками уровня activity и приложения позволяет сохранить небольшие данные отдельных типов (string, int), но для работы с большими массивами данных, такими как графически файлы, файлы мультимедиа и т.д., нам придется обращаться к файловой системе.

Приложение Android сохраняет свои данные в каталоге */data/data/<название\_пакета>/* и, как правило, относительно этого каталога будет идти работа.

Для работы с файлами абстрактный класс android.content.Context определяет ряд методов:

* deleteFile(String name): удаляет определенный файл
* fileList(): получает все файлы, которые содержатся в подкаталоге */files* в каталоге приложения
* getCacheDir(): получает ссылку на подкаталог *cache* в каталоге приложения
* getDir(String dirName, int mode): получает ссылку на подкаталог в каталоге приложения, если такого подкаталога нет, то он создается
* getExternalCacheDir(): получает ссылку на папку */cache* внешней файловой системы устройства
* getExternalFilesDir(): получает ссылку на каталог */files* внешней файловой системы устройства
* getFileStreamPath(String filename): возвращает абсолютный путь к файлу в файловой системе
* openFileInput(String filename): открывает файл для чтения
* openFileOutput (String name, int mode): открывает файл для записи

Все файлы, которые создаются и редактируются в приложении, как правило, хранятся в подкаталоге */files* в каталоге приложения.

Для непосредственного чтения и записи файлов применяются также стандартные классы Java из пакета *java.io*

Итак, применим функционал чтения-записи файлов в приложении. Пусть у нас будет следующая примитивная разметка layout:

|  |
| --- |
| <LinearLayout xmlns:android="<http://schemas.android.com/apk/res/android>"      xmlns:tools="<http://schemas.android.com/tools>"      android:layout\_width="match\_parent"      android:layout\_height="match\_parent"      android:weightSum="10"      android:orientation="vertical">      <EditText          android:id="@+id/save\_text"          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content"          android:gravity="top|left"          android:layout\_weight="4"/>      <Button          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:layout\_gravity="center"          android:onClick="saveText"          android:text="Сохранить"/>        <TextView          android:layout\_marginTop="80dp"          android:id="@+id/open\_text"          android:layout\_width="match\_parent"          android:layout\_height="wrap\_content"          android:layout\_weight="4"/>      <Button          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:layout\_gravity="center"          android:onClick="openText"          android:text="Открыть"/>    </LinearLayout> |

Поле EditText предназначено для ввода текста, а TextView - для вывода ранее сохраненного текста. Для сохранения и восстановления текста добавлены две кнопки.

Теперь в коде Activity пропишем обработчики кнопок с сохранением и чтением файла:

|  |
| --- |
| package com.example.eugene.filesystemapp;    import android.support.v7.app.ActionBarActivity;  import android.os.Bundle;  import android.view.Menu;  import android.view.MenuItem;    import android.widget.TextView;  import android.widget.EditText;  import android.view.View;  import android.widget.Toast;    import java.io.IOException;  import java.io.FileOutputStream;  import java.io.FileInputStream;      public class MainActivity extends ActionBarActivity {        FileInputStream fin;      FileOutputStream fos;        @Override      protected void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.activity\_main);      }      // сохранение файла      public void saveText(View view){            try {              EditText textBox = (EditText) findViewById(R.id.save\_text);              String text = textBox.getText().toString();                fos = openFileOutput("content.txt", MODE\_PRIVATE);              fos.write(text.getBytes());              Toast.makeText(this, "Файл сохранен", Toast.LENGTH\_SHORT).show();          }          catch(IOException ex) {                Toast.makeText(this, ex.getMessage(), Toast.LENGTH\_SHORT).show();          }          finally{              try{                  if(fos!=null)                      fos.close();              }              catch(IOException ex){                    Toast.makeText(this, ex.getMessage(), Toast.LENGTH\_SHORT).show();              }          }      }      // открытие файла      public void openText(View view){            TextView textView = (TextView) findViewById(R.id.open\_text);          try {              fin = openFileInput("content.txt");              byte[] bytes = new byte[fin.available()];              fin.read(bytes);              String text = new String (bytes);              textView.setText(text);          }          catch(IOException ex) {                Toast.makeText(this, ex.getMessage(), Toast.LENGTH\_SHORT).show();          }          finally{                try{                  if(fin!=null)                      fin.close();              }              catch(IOException ex){                    Toast.makeText(this, ex.getMessage(), Toast.LENGTH\_SHORT).show();              }          }      }  } |

При нажатии на кнопку сохранения будет создаваться поток вывода FileOutputStream fos = openFileOutput("content.txt", MODE\_PRIVATE)

В данном случае введенный текст будет сохраняться в файл "content.txt". При этом будет использоваться режим MODE\_PRIVATE

Система позволяет создавать файлы с двумя разными режимами:

* **MODE\_PRIVATE**: файлы могут быть доступны только владельцу приложения (режим по умолчанию)
* **MODE\_APPEND**: данные могут быть добавлены в конец файла

Поэтому в данном случае если файл "content.txt" уже существует, то он будет перезаписан. Если же нам надо было дописать файл, тогда надо было бы использовать режим MODE\_APPEND:

|  |  |
| --- | --- |
| 1 | FileOutputStream fos = openFileOutput("content.txt", MODE\_APPEND); |

Для чтения файла применяется поток ввода **FileInputStream**:

|  |  |
| --- | --- |
| 1 | FileInputStream fin = openFileInput("content.txt"); |

Подробнее про использование потоков ввода-вывода можно прочитать в руководстве по Java: <http://metanit.com/java/tutorial/6.3.php>

В итоге после нажатия кнопки сохранения весь текст будет сохранен в файле /data/data/название\_пакета/files/content.txt