**пример работы с внутренней памятью**

В этом уроке мы рассмотрим пример работы с внутренней памятью (Android Internal Storage), а именно хранение и чтение данных.

**Внутренняя память Android. Теория**

Любое устройство на Android позволяет хранение личных данных во внутренней памяти. Если одно приложение сохраняет или загружает файл на внутреннюю память, то другие приложения не смогут получить доступ к этим файлам. **Этими файлами может распоряжаться только это приложение**. Когда пользователь удаляет его, то все внутренние файлы этого приложения также удаляются.

Тем не менее, обратите внимание, что некоторые пользователи могут получить root-доступ на свои Android устройства. В этом случае, пользователи смогут получать доступ **к любым файлам**.

**Создание и чтение файлов с внутренней памяти Android**

Android использует методы openFileInput() и openFileOutput() классов [Java IO](https://javadevblog.com/category/java/java-io" \t "_blank) для работы с локальными файлами.

Обратите внимание, что эти методы **не принимают** на вход путь к файлу (например, **path/to/fileName.txt**), а просто получают имена файлов «**fileName.txt**«.

Метод openFileOutput() используется для создания и сохранения файла. Пример использования приведен ниже:

Java

|  |  |
| --- | --- |
| 1 | FileOutputStream fileOutputStream = openFileOutput("fileName.txt",Context.MODE\_PRIVATE); |

Метод openFileOutput() возвращает экземпляр FileOutputStream . После этого мы можем записать данные в файл. Пример:

Java

|  |  |
| --- | --- |
| 1  2  3 | String string = "какой-то текст, который мы будем записывать в файл";  fileOutputStream.write(str.getBytes());  fileOutputStream.close(); |

Метод openFileInput() используется для чтения данных из файла. Он возвращает экземпляр FileInputStream. Пример использования:

Java

|  |  |
| --- | --- |
| 1 | FileInputStream fileInputStream = openFileInput(file); |

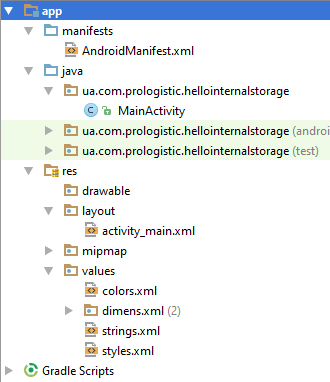
Далее мы посимвольно считываем данные из файла в строку. Например, так:

Java

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7 | int ch;  String resultString = "";  while((ch = fileInputStream.read()) != -1){     resultString = resultString + Character.toString((char)ch);  }    fileInputStream.close(); |

**Пример работы с внутренней памятью (Android Internal Storage). Практика**

Ниже представлена структура нашего простого Android проекта:

[](https://javadevblog.com/wp-content/uploads/2016/03/projStruc.png)

Код макета activity\_main.xml содержит кнопки writeBtn и readBtn для записи данных в файл и чтения. Обратите внимание, что мы определили метод OnClick в макете, а не в коде (можете сами назначить слушатель в коде — это ничего не меняет. Просто мы учим Android, поэтому пробуем различные способы управлять поведением виджетов):

Java

|  |
| --- |
| <?xml version="1.0" encoding="utf-8"?>  <RelativeLayout 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:paddingBottom="@dimen/activity\_vertical\_margin"      android:paddingLeft="@dimen/activity\_horizontal\_margin"      android:paddingRight="@dimen/activity\_horizontal\_margin"      android:paddingTop="@dimen/activity\_vertical\_margin"      tools:context="ua.com.prologistic.hellointernalstorage.MainActivity">        <TextView          android:id="@+id/textView\_title"          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:layout\_alignParentLeft="true"          android:layout\_alignParentRight="true"          android:padding="3dp"          android:text="Чтение/запись в Android Internal Storage"          android:textSize="24sp"          android:textStyle="bold" />        <EditText          android:id="@+id/edit\_field"          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:layout\_alignParentLeft="true"          android:layout\_alignParentRight="true"          android:layout\_below="@+id/textView\_title"          android:layout\_margin="3dp"          android:layout\_marginTop="20dp"          android:hint="Введите текст и нажмите 'Запись'"          android:inputType="text"          android:minLines="3">          <!--фокус ввода данных будет в этом виджете-->          <requestFocus />      </EditText>        <Button          android:id="@+id/writeBtn"          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:layout\_alignEnd="@+id/edit\_field"          android:layout\_alignRight="@+id/edit\_field"          android:layout\_alignTop="@+id/readBtn"          android:onClick="writeToFile"          android:text="Запись в файл" />        <Button          android:id="@+id/readBtn"          android:layout\_width="wrap\_content"          android:layout\_height="wrap\_content"          android:layout\_alignLeft="@+id/edit\_field"          android:layout\_alignStart="@+id/edit\_field"          android:layout\_centerVertical="true"          android:onClick="readFromFile"          android:text="Чтение из файла" />    </RelativeLayout> |

Чтобы назначать метод OnClick() в макете, мы должны определить метод в коде с параметром View, иначе Android выдаст ошибку, что метод имеет неправильную сигнатуру (в параметрах метода нет View).

Класс MainActivity содержит реализацию методов чтения и записи файлов:

Java

|  |
| --- |
| package ua.com.prologistic.hellointernalstorage;    import android.os.Bundle;  import android.support.v7.app.AppCompatActivity;  import android.view.View;  import android.widget.EditText;  import android.widget.Toast;    import java.io.FileInputStream;  import java.io.FileOutputStream;  import java.io.InputStreamReader;  import java.io.OutputStreamWriter;    public class MainActivity extends AppCompatActivity {        private EditText editText;      // определяем размер буфера при считывании с файла      private static final int READ\_BLOCK\_SIZE = 100;    *@Override*      protected void onCreate(Bundle savedInstanceState) {          super.onCreate(savedInstanceState);          setContentView(R.layout.activity\_main);            editText = (EditText) findViewById(R.id.edit\_field);      }        // метод для записи в файл      public void writeToFile(View v) {          // открываем файл по названию          try {              FileOutputStream fileOutputStream = openFileOutput("simplefile.txt", MODE\_PRIVATE);              OutputStreamWriter outputWriter = new OutputStreamWriter(fileOutputStream);              outputWriter.write(editText.getText().toString());              outputWriter.close();                // создаем всплывающее окно c результатом выволнения записи в файл              Toast.makeText(getBaseContext(), "Запись в файл успешно проведена!",                      Toast.LENGTH\_LONG).show();            } catch (Exception e) {              e.printStackTrace();          }      }        // метод для чтения из файла      public void readFromFile(View v) {          try {              FileInputStream fileInputStream = openFileInput("simplefile.txt");              InputStreamReader reader = new InputStreamReader(fileInputStream);                char[] inputBuffer = new char[READ\_BLOCK\_SIZE];              String s = "";              int charRead;                // цикл читает данные из файла,              while ((charRead = reader.read(inputBuffer)) != -1) {                  // конвертируем char в строку                  String rString = String.copyValueOf(inputBuffer, 0, charRead);                  s += rString;              }              reader.close();              editText.setText(s);              // создаем всплывающее окно c результатом выволнения чтения из файла              Toast.makeText(getBaseContext(), "Чтение из файла успешно проведено!",                      Toast.LENGTH\_LONG).show();            } catch (Exception e) {              e.printStackTrace();          }      }  } |