**Чтение и запись файлов**

[Текстовые файлы](http://developer.alexanderklimov.ru/android/io.php#textfile)  
[Графические файлы](http://developer.alexanderklimov.ru/android/io.php#imgfile)

В пакете **java.io.\*** находятся классы, необходимые для работы с файлами: **FileInputStream**, **FileOutputStream**, **InputStream**, **OutputStream** и др.

Во многих проектах мы уже не раз сталкивались с примерами чтения и записи файлов, поэтому данный материал можно считать закрепляющим.

**Текстовые файлы**

**Чтение файла**

Для чтения файла используются классы **FileInputStream** (для открытия) и **InputStreamReader** (для организации чтения):

FileInputStream fis = new FileInputStream("cats1098.txt");

InputStreamReader isr = new InputStreamReader(fis. "UTF-8");

Класс **InputStreamReader** позволяет задавать кодировку файла. Если в этом нет необходимости, то можно использовать класс **FileReader**.

Напишем пример для чтения содержимого текстового файла. Создадим специальный метод, который будет возвращать содержимое файла. Методу нужно будет передать объект класса**File**.

// функция для чтения текстового файла

static public String getContent(File file) {

StringBuilder builder = new StringBuilder();

try {

// Считываем по одной строке

BufferedReader = input = new BufferedReader(new FileReader(file));

try {

String line = null;

while ((line = input.readLine()) != null){

// прочитанную строку добавляем в буфер

// после каждой строки добавляем разделитель строк

builder.append(line);

builder.append(System.getProperty("line.separator"));

}

}

finally {

input.close();

}

}

catch (IOException e){

e.printStackTrace();

}

// возвращаем буфер строк как одну больщую строку

return contents.toString();

}

Можно переделать метод, чтоб он принимал имя файла - *static public String getContent(String file){...*, поскольку конструктор класса **FileReader** принимает как строку, так и объект класса **File**:

FileReader(String filePath)

FileReader(File fileObj)

**Запись в файл**

Напишем метод для записи в файл. Ему нужно передать два параметра - файл и строку, которую нужно записать.

static public void setContent(File file, String content)

throws FileNotFoundException, IOException {

if (file == null) {

throw new IllegalArgumentException("File should not be null.");

}

if (!file.exists()) {

throw new FileNotFoundException ("File does not exist: " + file);

}

if (!file.isFile()) {

throw new IllegalArgumentException("Should not be a directory: " + file);

}

if (!file.canWrite()) {

throw new IllegalArgumentException("File cannot be written: " + file);

}

// используем буферизацию

Writer output = new BufferedWriter(new FileWriter(file));

try {

//FileWriter always assumes default encoding is OK!

output.write(content);

}

finally {

output.close();

}

}

Чтобы прочитать и внести изменения в файл при помощи написанных функций:

// Определяем файл

File myFile = new File("cats1098.txt");

// Получаем содержимое файла

String old\_content = getContent(myFile);

// Зададим новый текст для файла

String new\_content = "Cats, kitten";

// Записываем в файл

setContent(myFile, new\_content);

**Построчное добавление в файл**

В предыдущем примере мы сразу меняли весь текст. Можно вносить изменения построчно через метод **append()**. Допустим, мы хотим создать на SD-карте CSV-файл:

package ru.alexanderklimov.csvdemo;

import java.io.File;

import java.io.FileWriter;

import java.io.IOException;

import android.app.Activity;

import android.os.Bundle;

import android.os.Environment;

public class MainActivity extends Activity {

@Override

public void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.main);

generateCsvFile("aboutandroid.csv");

}

private static void generateCsvFile(String sFileName)

{

try

{

File root = Environment.getExternalStorageDirectory();

File gpxfile = new File(root, sFileName);

FileWriter writer = new FileWriter(gpxfile);

writer.append("SDK Name");

writer.append(',');

writer.append("SDK Number");

writer.append('\n');

writer.append("Eclairs");

writer.append(',');

writer.append("2.1");

writer.append('\n');

writer.append("Froyo");

writer.append(',');

writer.append("2.2");

writer.append('\n');

writer.flush();

writer.close();

}

catch(IOException e)

{

e.printStackTrace();

}

}

}

**Графические файлы**

Рассмотрим работу с графическими файлами

**Чтение графического файла**

// имя файла на SD-карте

String fname = "/sdcard/myImages/cat.jpg";

//Создаем объекта класса File

File imageFile = new File(fname);

// Если файл существует

if(imageFile.exists()){

// Загружаем изображение

Bitmap bmp = BitmapFactory.decodeFile(fname);

// Куда выводить изображение

ImageView myImage = (ImageView) findViewById(R.id.imageView1);

// Выводим картинку

myImage.setImageBitmap(bmp);

}

**Сохранение изображения в файл**

Например, мы сохраняем изображение с камеры:

Bitmap takenPicture;

// Устанавливаем имя файла для изображения

FileOutputStream out = openFileOutput("mycat.png", Context.MODE\_WORLD\_WRITEABLE);

// Устанавливаем метод сжатия

takenPicture.compress(CompressFormat.PNG, 100, out);

// Сбрасываем содержимое буфера

out.flush();

// Закрываем поток

out.close();

Стоит отметить, что в Android 6.0 усложнили работу с файлами и требуется другой подход. Пока эта система не получила широкого распространения, поэтому пока материал не обновлялся.