**Работа с файлами используя InputStream и OutputStream**

В данном уроке пойдет речь о потоках InputStream и OutputStream, а именно как с ними работать используя **Java IO API**.

В прошлом уроке мы познакомились с Java IO в этом уроке [Урок 1. Введение в Java IO](https://devcolibri.com/?p=2925).

**Шаг 1**

Что нам позволяют делать InputStream и OutputStream? Главная их цель прочитать и записать поток данных.

InputStream считывает поток байтов и позволяет манипулировать этими данными.

В папке C:\TMP\data я создал файл data.txt c текстом:

Hello World!

Давайте прочитаем файл, а потом перезапишем в него новую строку, используя InputStream и OutputStream, для этого создадим класс **InputOutputStreamExam.java**:

package examples;

// указываем, что будем использовать IO Api

import java.io.\*;

public class InputOutputStreamExam {

// класс для чтения файла

private InputStream inputstream;

// класс для записи в файл

private OutputStream outputStream;

// путь к файлу который будем читать и записывать

private String path;

public InputOutputStreamExam(String path) {

this.path = path;

}

// чтение файла используя InputStream

public void read() throws IOException {

// инициализируем поток на чтение

inputstream = new FileInputStream(path);

// читаем первый символ в байтах (ASCII)

int data = inputstream.read();

char content;

// по байтово читаем весь файл

while(data != -1) {

// преобразуем полученный байт в символ

content = (char) data;

// выводим посимвольно

System.out.print(content);

data = inputstream.read();

}

// закрываем поток

inputstream.close();

}

// запись в файл используя OutputStream

public void write(String st) throws IOException {

// инициализируем поток для вывода данных

// что позволит нам записать новые данные в файл

outputStream = new FileOutputStream(path);

// передаем полученную строку st и приводим её к byte массиву.

outputStream.write(st.getBytes());

// закрываем поток вывода

// только после того как мы закроем поток данные попадут в файл.

outputStream.close();

}

}

**Шаг 2**

Теперь в классе Main пишем следующее:

package examples;

import java.io.IOException;

public class Main {

// указываем путь к файлу с которым мы будем работать

private static final String PATH = "C:/TMP/data/data.txt";

// создаем экземпляр нашего класса который мы создали в первом шаге

private static InputOutputStreamExam streamExam;

public static void main(String... args) throws IOException {

// инициализируем наш класс для работы с файлом

streamExam = new InputOutputStreamExam(path);

// вызываем метод чтения файла

streamExam.read();

// вызываем метод записи в файл

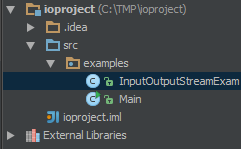
streamExam.write("Hello DevColibri!");

}

}

**Шаг 3**

После всего сделанного выше мы получим следующую структуру проекта:



Перед тем как перезаписать файл в нем была строка:

Hello World!

После того как мы вызвали метод **read()**; в консоли нам должен был вывестись этот текст, и после вызова **write(“Hello DevColibri”)**; файл перезаписался и в нем уже новая строка:

Hello DevColibri

Проект не прикрепляю, так как выложил листинг всех файлов в уроке.