**Как записать в файл на Java Примеры с FileWriter BufferedWriter Files и FileOutputStream**

Java предоставляет множество способов записи данных файл, поэтому сегодня мы рассмотрим на примере наиболее популярные способы.

**Как записать в файл. Теория**

**FileWriter — это самый простой способ записать информацию в файл.**Он предоставляет перегруженный метод write() для записи int, байтового массива и String в файл. FileWriter отлично подходит для записи небольших объемов данных.

**BufferedWriter — почти аналогичен FileWriter, но использует внутри себя буфер для записи данных в файл.** Если вам нужно записать в файл большие объемы информации, то ваш выбор должен пасть на BufferedWriter.

FileWriter и BufferedWriter хорошо справляются с записью текстовой информации в файл, но когда стоит задача записывать в файл данные потока, то желательно использовать**FileOutputStream**.

**Files** — вспомогательный класс для работы с файлами. Он содержит метод, который внутри себя использует OutputStream для  записи массива байтов в файл.

**Пишем в файл на Java. Практика**

Ниже приведена программа для записи информации в файл с помощью классов FileWriter, BufferedWriter, FileOutputStream и Files.

|  |
| --- |
| package ua.com.prologistic;    import java.io.BufferedWriter;  import java.io.File;  import java.io.FileOutputStream;  import java.io.FileWriter;  import java.io.IOException;  import java.io.OutputStream;  import java.nio.file.Files;  import java.nio.file.Paths;    public class WriteFile {        public static void main(String[] args) {          String data = "Это тестовая строка для записи в файл";          int noOfLines = 10000;            // пишем в файл с помощью FileWriter          writeUsingFileWriter(data);            // пишем в файл с помощью BufferdWriter          writeUsingBufferedWriter(data, noOfLines);            // пишем в файл с помощью Files          writeUsingFiles(data);            // пишем в файл с помощью OutputStream          writeUsingOutputStream(data);      }         // пишем в файл с помощью OutputStream       private static void writeUsingOutputStream(String data) {          OutputStream os = null;          try {              os = new FileOutputStream(new File("/Users/prologistic/file.txt"));              os.write(data.getBytes(), 0, data.length());          } catch (IOException e) {              e.printStackTrace();          }finally{              try {                  os.close();              } catch (IOException e) {                  e.printStackTrace();              }          }      }        // пишем в файл с помощью Files      private static void writeUsingFiles(String data) {          try {              Files.write(Paths.get("/Users/prologistic/files.txt"), data.getBytes());          } catch (IOException e) {              e.printStackTrace();          }      }        // пишем в файл с помощью BufferedWriter      private static void writeUsingBufferedWriter(String data, int noOfLines) {          File file = new File("/Users/prologistic/BufferedWriter.txt");          FileWriter fr = null;          BufferedWriter br = null;          String dataWithNewLine = data + System.getProperty("line.separator");          try{              fr = new FileWriter(file);              br = new BufferedWriter(fr);              for(int i = noOfLines; i>0; i--){                  br.write(dataWithNewLine);              }          } catch (IOException e) {              e.printStackTrace();          }finally{              try {                  br.close();                  fr.close();              } catch (IOException e) {                  e.printStackTrace();              }          }      }          // пишем в файл с помощью FileWriter      private static void writeUsingFileWriter(String data) {          File file = new File("/Users/prologistic/FileWriter.txt");          FileWriter fr = null;          try {              fr = new FileWriter(file);              fr.write(data);          } catch (IOException e) {              e.printStackTrace();          }finally{              try {                  fr.close();              } catch (IOException e) {                  e.printStackTrace();              }          }      }  } |

Обратите внимание, что в методе записи в файл с помощью BufferedWriter мы использовали разделитель пути файла "line.separator". Подробнее о нем читайте [здесь](http://prologistic.com.ua/razdelitel-puti-fajla-separatorchar-pathseparator-pathseparatorchar-opisanie-i-primery.html).

Это стандартные методы записи данных в файл на Java. Используйте нужный метод, отталкиваясь от особенностей вашего проекта. Подробнее о работе с файлами читайте в разделе [Java IO](http://prologistic.com.ua/category/java/java-io" \t "_blank).