INPUT/OUTPUT STREAM

Разбор концепта обертки в Buffered: http://stackoverflow.com/questions/9648811/specific-difference-between-bufferedreader-and-filereader В чем преимущество обертки: http://stackoverflow.com/questions/18600331/why-is-using-bufferedinputstream-to-read-a-file-byte-by-byte-faster-than-using-f В чем отличие inputstreamreader от filereader: http://stackoverflow.com/questions/7991770/inputstreamreader-vs-filereader <http://stackoverflow.com/questions/5155226/fileinputstream-vs-filereader>

Если не вызвать метод сlose возможна утечка памяти. Например если мы открыли поток для работы с файлом и не закрыли его, даже после того как перестала работать Jvm возможно доступа к файлу с стороны операционной системы не будет.

Как записать строку в файл.

Есть много способов.

С помощью метода FileWriter:

String s = **"Some Text and new message"**;  
FileWriter fileWriter = **new** FileWriter(**"D:/1.txt"**);  
fileWriter.write(s);  
fileWriter.flush();  
fileWriter.close();

Но правильней и быстрее будет завернуть все в BufferedWriter.

BufferedWriter fileWriter = **new** BufferedWriter(**new** FileWriter(**"D:/1.txt"**));  
fileWriter.write(s);  
fileWriter.flush();  
fileWriter.close();

НО! Все советую использовать OutputStreamWriter, для того чтоб можно біло задать кодировку вручную. Personally I would avoid FileReader altogether though, as it always uses the system default character encoding. Instead, use InputStreamReader around a FileInputStream... but only when you want to deal with text.

BufferedWriter out = **new** BufferedWriter(**new** OutputStreamWriter(**new** FileOutputStream(**"D:/1.txt"**),**"CP1251"**));

С помощью FileOutputStream:

String s = **"Hello amigos"**;  
BufferedOutputStream out = **new** BufferedOutputStream(**new** FileOutputStream(**"D:/1.txt"**));  
out.write(s.getBytes());  
out.flush();  
out.close();

Как записать информацию с файла в файл

Базовые классы – InputStream, OutputStream (для байтов), Reader, Writer (для текста);

ВАЖНО!!!! КОДИРОВКУ МОЖНО ЗАДАТЬ ТОЛЬКО В OutputStreamReader(Writer)

Преемущества использования InputStreamReader:

First, InputStreamReader can handle all input streams, not just files. Other examples are network connections, classpath resources and ZIP files.

Second, FileReader does not allow you to specify an encoding and instead uses the plaform default encoding, which makes it pretty much useless as using it will result in corrupted data when the code is run on systems with different platform default encodings.

**In short, forget that FileReader exists.**

BufferedWriter out = **new** BufferedWriter(**new** OutputStreamWriter(**new** FileOutputStream(**"D:/1.txt"**),**"CP1251"**));

Если FileReader завернуть в BufferedReader то будет так: файлридер будет считывать всю строку в RAM, а после этого уже в java приложение. Это убирает необходимость обращатся к файлу за двумя байтами.

Обертака Bufered служит для того чтоб использовать буфер во время чтения и записи строк.

FileReader – больше создан для работы с текстовыми файлами, он считывает сразу по два байта за раз.

InputStreamReader(Writer) – это мост между байтовыми потоками и символьными. Он очень одобен тем, что можно задать кодировку явно. Все советуют использовать его вместе с оберткой Buffered.