

Рассмотрим работу с FileWriter и FileReader:

- с помощью FileWriter мы можем создавать файлы
- с помощью FileReader - считывать их

Работая с ними, понадобится всегда помнить 3 важных момента:

## 1. Объявление

Перед тем, как вызывать какие-нибудь методы для работы с файлами, нужно объявить FileWriter/FileReader:

```
FileWriter fw= new FileWriter («MyFile.txt»);
```

 имя объекта

 имя расширение  
файла

ИЛИ

```
FileReader fr= new FileReader («MyFile.txt»);
```

 имя объекта

 имя расширение  
файла

Но среда может не распознать `FileReader/FileWriter` и начнет ругаться. Если такое произойдет, импортируйте библиотеку **java.io.\***. Для этого в самой первой строчке напишите:

```
import java.io.*;
```

## 2. Нужно закрыть поток

FileWriter/FileReader - это потоки, их нужно не только «открыть» (то-есть объявить), но и **«заккрыть»**. Представьте, что Вы открыли кран. Нельзя же уйти из дому, оставив воду литься?

Это **правило** работает и для других потоков - кроме стандартных System.in и System.out.

Закрывать поток можно с помощью `.close()`:

**fw.close();**  
↑  
имя объекта

или

**fr.close();**  
↑  
имя объекта

Пример:

```
1 import java.io.*;
2
3 class Test {
4
5     public static void main(String[] args) throws Exception {
6         FileWriter fw = new FileWriter( "sample1.txt" );
7         fw.close();
8
9         FileReader fr = new FileReader( "sample2.txt" );
10        fr.close();
11
12    }
13 }
```

### 3. Допишите "волшебную фразу".

В программировании очень важна **безопасность**. А работа с FileWriter/FileReader - это **небезопасно**, в процессе может возникнуть масса разных ошибок. Это беспокоит Eclipse (или IntelliJIdea - смотря чем пользуетесь), и программу она просто так не запустит. Помните, что к методу нужно дописать **«throws Exception»**:

```
1  import java.io.*;
2
3  public class Test2 {
4
5      public static void main(String[] args) throws Exception {
6          FileWriter fw = new FileWriter( "sample1.txt" );
7          FileReader fr = new FileReader( "sample2.txt" );
8
9          fw.close();
10         fr.close();
11     }
12 }
13
14
15
16
```

"волшебная фраза"

Итак, еще раз акцентируем внимание - всегда Вы должны помнить о 3 моментах:

1. **Объявить**
2. **Не забыть закрыть поток**
3. **Дописать «throws Exception»**

И еще, потоки `FileWriter` и `FileReader` воспринимают все файлы как текстовые:

---

## FileWriter

Теперь представим, что Вы начинаете использовать **FileWriter**.

### 1. Объявление.

Как Вы помните, нужно не забыть импортировать библиотеки **java.io.\*** и дописать "волшебную фразу" к методу, где Вы собираетесь объявить `FileWriter`.

Объявляем, как помните, почти как `Scanner`:

**FileWriter fw= new FileWriter («MyFile.txt»);**

↑  
имя объекта

↑   ↑  
имя   расширение  
файла

Объявили. А что теперь можно делать? Теперь пора пользоваться возможностями FileWriter!

Основной метод FileWriter - это метод **.write()**.

**fw.write("Hello World!");**

↑   ↑  
имя объекта   текст, который будет  
                  записан в файл

Мало? Да, но посмотрите, как много с ним можно сделать:

```
import java.io.*;

public class Test {

    public static void main(String[] args) throws Exception {
        int k1 = 2;
        int k2 = 9;
        newFile( k1, k2);
    }

    public static void newFile(int k1, int k2) throws Exception
    {
        FileWriter nFile = new FileWriter("file1.txt");

        for(int i = k1; i <= k2; i++) {

            nFile.write(i);

        }

        nFile.close();
    }
}
```

**\*обратите внимание - мы написали нашу "волшебную фразу" и в методе main, и в методе newFile.**

Так мы можем записать числа от k1 до k2, от 2 до 9, в наш файл **file1.txt**. Можно записывать только четные или нечетные числа, какой-нибудь текст, и многое другое.

## 2. Переход на следующую строку

Но мы Вам кое-чего не сказали. Если запустить код из прошлого пункта, получится:

```
23456789|
```

Если понадобится вывести числа в столбик, понадобится добавить "**\n**" от "**new line**", новая строка. Запишем в файл стих:

```
import java.io.FileWriter;

public class Test {

    public static void main(String[] args) throws Exception {

        FileWriter nFile = new FileWriter("file1.txt");

        nFile.write("Хокку \nПодобен лучу самурайский клинок\nИ тот затупился \nПроклятая килька в томате!!");

        nFile.close();
    }

}
```

Получим:

```
Хокку
Подобен лучу самурайский клинок
И тот затупился
Проклятая килька в томате!!
```

Теперь вы знаете, как вывести числа с новой строки:

```
nFile.write(i+"\n");
```

## 3. Закрываем поток

После того, как Вы записали все необходимое, нужно не забыть закрыть поток. Это мы делали в каждом из приведенных примеров:

```

1 import java.io.*;
2
3 public class Test {
4
5     public static void main(String[] args) throws Exception {
6         int k1 = 2;
7         int k2 = 9;
8         newFile( k1, k2);
9     }
10
11
12     public static void newFile(int k1, int k2) throws Exception {
13         FileWriter nFile = new FileWriter("file1.txt");
14
15         for(int i = k1; i <= k2; i++) {
16
17             nFile.write(i);
18
19         }
20
21         nFile.close();
22     }
23 }

```

Поток закрыт

```

1 import java.io.FileWriter;
2
3 public class Test {
4
5     public static void main(String[] args) throws Exception {
6
7         FileWriter nFile = new FileWriter("file1.txt");
8
9         nFile.write("Хокку \nПодобен лучу самурайский клинок \nИ тот затупился \nПрок.");
10
11         nFile.close();
12     }
13 }

```

Поток закрыт

## FileReader

Теперь, рассмотрим пошагово работу с FileReader.

### 1. Объявление

Сначала FileReader, как и FileWriter, нужно **объявить**. Не забудьте про **библиотеку** и "**волшебную фразу**":

```

import java.io.FileReader;
import java.util.Scanner;

```

```

public class Test {

```

```

    public static void main(String[] args) throws Exception {

```

```
        FileReader fr= new FileReader("file1.txt");
        Scanner scan = new Scanner(fr);

        fr.close();

    }

}
```

## 2. FileReader + Scanner

Мы объявили не только FileReader, но и Scanner. Почему?

В отличии от FileWriter, FileReader не используется один:

Не вдаваясь в подробности, запомните, что FileReader и Scanner идут вместе. Но не забывайте их "связать" - для этого напишите название вашего объекта FileReader вместо "System.in" при объявлении Scanner:

```
FileReader fr= new FileReader("file1.txt");
Scanner scan = new Scanner(fr) ✓

FileReader fr= new FileReader("file1.txt");
Scanner scan = new Scanner(System.in) ✗
```

## 3. Методы

Тут уже больше методов. Рассмотрим методы **.nextLine()** и **.hasNextLine()**.

- **.nextLine()** - это метод, который считывает строку (до ENTER), и возвращает это значение
- **.hasNextLine()** - метод, который возвращает boolean - true или false, показывая, есть ли следующая строка.

Пример:

```
import java.io.FileReader;
import java.util.Scanner;

public class Test {

    public static void main(String[] args) throws Exception {

        FileReader fr= new FileReader("file1.txt");
        Scanner scan = new Scanner(fr);

        int i = 1;

        while (scan.hasNextLine()) {
            System.out.println(i + " : " + scan.nextLine());
            i++;
        }

        fr.close();

    }

}
```

Должен быть такой результат:

```
1 : Хокку
2 : Подобен лучу самурайский клинок
3 : И тот затупился
4 : Проклятая килька в томате!!
```

Обратите внимание: мы используем **.hasNextLine()** для того, чтобы избежать ошибки, и не заставлять **.nextLine()** считывать строку, которой не существует:


The screenshot shows a snippet of Java code. The line `while (scan.hasNextLine()) {` is circled in red. An arrow points from the text «Есть следующая строчка?» to the `scan.hasNextLine()` method call.

```
while (scan.hasNextLine()) {
    System.out.println(i + " : " + scan.nextLine());
    i++;
}

fr.close();
```



```
while (scan.hasNextLine()) {  
    System.out.println(i + " : " + scan.nextLine());  
    i++;  
}  
  
fr.close();
```




Тогда работай дальше!

#### 4. Закрываем поток.

Закрываем поток:

```
1 import java.io.FileReader;  
2 import java.util.Scanner;  
3  
4 public class Test {  
5  
6     public static void main(String[] args) throws Exception {  
7  
8         FileReader fr= new FileReader("file1.txt");  
9         Scanner scan = new Scanner(fr);  
10  
11         int i = 1;  
12  
13         while (scan.hasNextLine()) {  
14             System.out.println(i + " : " + scan.nextLine());  
15             i++;  
16         }  
17  
18         fr.close();  
19  
20  
21     }  
22 }
```



Поток закрыт

Готово. Теперь Вы знаете, как работать с FileWriter и FileReader.