**Создание временных файлов в Java**

Сегодня мы научимся создавать временные файлы в Java. Как обычно, сначала немного теории и общих понятий, а потом на практике все закрепим.

**Создание временных файлов в Java. Теория**

Иногда нам нужно создать временный файл, который будет использоваться только в нашем приложении. Есть два способа с помощью которых мы можем создать временный файл. Все решения присутствуют в стандартном классе java.io.File:

**Метод createTempFile(String prefix, String suffix, File directory):**

* Этот метод создает файл с определенным суффиксом и префиксом в указанной папке.
* Аргумент File directory должен уже существовать и должен быть каталогом, иначе будет брошено исключение.
* Имя файла создается по формуле prefix + random\_long\_no + suffix. Это сделано для безопасности приложения, так как никто не будет знать имя файла и ваша программа одна будет работать временным файлом.
* prefix должен быть минимум три символа.
* Если suffix является null, то по умолчанию используется «.tmp«.
* Если в параметр directory передан null, то временный файл создается в стандартном каталоге для временных файлом операционной системы.

**Метод createTempFile(String prefix, String suffix):**

Это самый простой способ создать временный файл в стандартной папке для временных файлом операционной системы.

**Создание временных файлов в Java. Практика**

Вот небольшой пример, показывающий создание временного файла.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25 | package ua.com.prologistic;    import java.io.File;  import java.io.IOException;    public class TempFile {        public static void main(String[] args) {          try {              File tmpFile = File.createTempFile("data", null);              File newFile = File.createTempFile("text", ".temp", new File("/Users/prologistic/tmp"));              System.out.println(tmpFile.getCanonicalPath());              System.out.println(newFile.getCanonicalPath());                // работайте с временным файлом как с обычным файлом                // удаляем файл после того, как ваша программа закончила работу              tmpFile.deleteOnExit();              newFile.deleteOnExit();          } catch (IOException e) {              e.printStackTrace();          }      }    } |

Следует отметить, что мы можем работать с временными файлами также, как и с обычными файлами. Временные файлы должны быть удалены после того, как программа закончила свою работу