Java

Класс File

Лекция #8

Пустовалова О.Г.
доцент. каф. мат.мод.
ИММиКН ЮФУ

Содержание

Класс File в Java

Класс File

Класс **File**, определенный в пакете java.io, не работает напрямую с потоками. Его задачей является управление информацией о файлах и каталогах.

Хотя на уровне операционной системы файлы и каталоги отличаются, но в Java они описываются одним классом **File**.

Можно использовать один из конструкторов для создания объекта:

File(String путь_к_каталогу)

File(String путь_к_каталогу, String имя_файла)

File(File каталог, String имя_файла)

Пример применения конструкторов для создания объектов

```
 // создаем объект File для каталога
File dir1 = new File("d://ClassFile");


// создаем объекты для файлов, которые находятся в каталоге
File file1 = new File("d://ClassFile", "file_001.txt");
File file2 = new File(dir1, "file_002.txt");
```

Методы класса File

boolean createNewFile():

создает новый файл по пути, который передан в конструктор.

В случае удачного создания возвращает **true**, иначе **false**

boolean delete():

удаляет каталог или файл по пути, который передан в конструктор.

При удачном удалении возвращает **true**.

boolean exists():

проверяет, существует ли по указанному в конструкторе пути файл или каталог.

И если файл или каталог существует, то возвращает **true**, иначе возвращает **false**

Методы класса File

String **getAbsolutePath**():

возвращает абсолютный путь для пути, переданного в конструктор объекта

String **getName**():

возвращает краткое имя файла или каталога

String **getParent**():

возвращает имя родительского каталога

Методы класса File

boolean isDirectory():

возвращает значение **true**, если по указанному пути располагается каталог

boolean isFile():

возвращает значение **true**, если по указанному пути находится файл

boolean isHidden():

возвращает значение **true**, если каталог или файл являются скрытыми

Методы класса File

`long length():`

возвращает размер файла в байтах

`long lastModified():`

возвращает время последнего изменения файла или каталога. Значение представляет количество миллисекунд, прошедших с начала эпохи Unix

Методы класса File

String[] list():
возвращает массив файлов и подкаталогов, которые находятся в определенном каталоге

File[] listFiles():
возвращает массив файлов и подкаталогов, которые находятся в определенном каталоге

boolean mkdir(): создает новый каталог и при удачном создании возвращает значение **true**

boolean renameTo(File dest): переименовывает файл или каталог

Пример. Получить список файлов и каталогов из каталога

```
import java.io.File;
public class Main {

    public static void main(String[] args) {

        // определяем объект для каталога
        File dir = new File("d://MyDir");

        // если объект представляет каталог
        if (dir.isDirectory()) {

            // получаем все вложенные объекты в каталоге
            for (File item : dir.listFiles()) {

                if (item.isDirectory()) {

                    System.out.println(item.getName() + "   \t folder");
                } else {

                    System.out.println(item.getName() + "\t file");

                }             }         }     } }
```

Пример. Удаление, переименование и создание каталогов

```
import java.io.File;
public class Main {

    public static void main(String[] args) {

     File dir = new File("D://Folder");

     boolean created = dir.mkdir();

     if(created)
        System.out.println("Folder has been created")
;
     // переименуем каталог
     File newDir = new File("D://NewFolder");

        dir.renameTo(newDir);

     // удалим каталог
     boolean deleted = newDir.delete();

    if(deleted)
        System.out.println("Folder has been deleted");
    }}
```

Работа с файлами

Пример. Часть 1 Получение информации о файле

```
import java.io.*;
public class Main {
public static void main(String... args) {

File file = new File("D://Folder//notes.txt");

System.out.println("Имя файла: " + file.getName());

System.out.println("Путь: " + file.getPath());

System.out.println("Абсолютный путь: " + file.getAbsolutePath());

System.out.println("Родительский каталог: " + file.getParent());
```

Пример. Часть 2. Получение информации о файле

```
System.out.println(file.exists ()
        ? "Файл/каталог существует." :
            "Файл/каталог не существует.");


System.out.println(file.canWrite()
        ? "Файл/каталог доступен для редактирования." :
            "Файл/каталог не доступен для редактирования.");


System.out.println(file.canRead()
        ? "Файл/каталог доступен для чтения." :
            "Файл/каталог не доступен для чтения.");
```

Пример. Часть 3. Получение информации о файле

```
System.out.println((file.isDirectory() ? "Каталог." :
                                          "Не каталог."));

System.out.println(file.isFile() ? "Файл." : "Не файл.");

System.out.println(file.isAbsolute()
        ? "Абсолютный путь." : "Не абсолютный путь.");

System.out.println("Дата последнего редактирования: " +
                                          file.lastModified());

System.out.println("Размер: " + file.length() + " байт."); }}
```

Пример. Часть 1. Работа с файлами

```
import java.io.*;
public class Main {
public static void main(String... args) {

    File myFile = new File("D://Folder//notes.txt");

    System.out.println("File name: " + myFile.getName());

    System.out.println("Parent folder: " + myFile.getParent());

    if(myFile.exists())
        System.out.println("File exists");
    else
        System.out.println("File not found");
```

getName

getParent

exists

Пример. Часть 2. Работа с файлами

```
System.out.println("File size: " + myFile.length());

    if(myFile.canRead())
        System.out.println("File can be read");

    else
        System.out.println("File can not be read");

    if(myFile.canWrite())
        System.out.println("File can be written");

    else
        System.out.println("File can not be written");
```

Длина файла

canRead

canWrite

Пример. Часть 3. Работа с файлами

```
// создадим новый файл
      File newFile = new File("D://Folder//myFile.txt");


   try
   {
     boolean created = newFile.createNewFile();


    if(created)
     System.out.println("File has been created");
      }
      catch(IOException ex){

         System.out.println(ex.getMessage());
      }
}}
```

При создании нового файла метод **createNewFile**() в случае неудачи выбрасывает исключение **IOException**

Это исключение нужно обрабатывать, например, в блоке **try**...**catch**.

Пример. Вывод всех файлов и папок в текущем каталоге

```
//объект-директория
File dir = new File("d://Folder");

//печать списка файлов и папок на экран
for (File file : dir.listFiles())
{
    System.out.println(file.getName());
}
```

Спасибо за внимание!