Стандартная библиотека Java: ввод/вывод

Алексей Владыкин

17 октября 2012

java.io.File

- API к файловой системе до Java 6 включительно
- Представляет файл *или директорию*
- Файл идентифицируется путем, специфичным для ОС/ФС
  - `\\server\share\`
  - `C:\Program Files\Java`
  - `/usr/bin/ls`

Работа с путями

```
File java = new File("/usr/bin/java");
java.isAbsolute();   // true
java.getPath();      // "/usr/bin/java"
java.getName();      // "java"
java.getParent();    // "/usr/bin"
```

- Файла по этому пути может не быть
- Поддерживаются абсолютные и относительные пути

Работа с путями

- `String getAbsolutePath()`
  `File getAbsoluteFile()`

  Если путь файла не абсолютный, то он разрешается относительно текущей директории — `System.getProperty("user.dir")`

- `String getCanonicalPath()`
  `File getCanonicalFile()`

  Путь преобразуется в абсолютный, убираются компоненты `.` и `..`, разрешаются символические ссылки

Работа с файлами

```
File java = new File("/usr/bin/java");
java.exists();          // true
java.isFile();          // true
java.canRead();         // true
java.length();          // 1536
java.lastModified();// 1231914805000
```

- Если файла нет, `length()` и `lastModified()` возвращают 0

Работа с директориями

```
File usrbin = new File("/usr/bin");
usrbin.exists();       // true
usrbin.isDirectory(); // true
usrbin.list();         // array of Strings
usrbin.listFiles();    // array of Files
```

- Если директория не существует, `list()` вернет `null`
- Есть `list()` и `listFiles()` с фильтром

Модификация файловой системы

- `boolean createNewFile()`

  Создает пустой файл, если он еще не существует

- `boolean delete()`

  Удаляет файл. Если это директория, то она должна быть пустой

- `boolean renameTo(File dest)`

  Перемещает файл. Перемещение на другую ФС может не работать. Перемещение поверх существующего файла тоже.

- `boolean mkdir()`
  `boolean mkdirs()`

  Создает директорию или директории.

java.nio.file.*

- Новый API для работы с файловой системой, начиная с Java 7
- Центральная сущность — `java.nio.file.Path` — представляет путь в файловой системе
- Определяет синтаксические операции над путями
- Доступ к файловой системе обеспечивает класс `java.nio.file.Files`
- `File.toPath()`, `Path.toFile()`

Работа с путями

```
Path java = Paths.get("/usr/bin/java");
java.isAbsolute();    // true
java.toString();      // /usr/bin/java
java.getFileName();   // java
java.getParent();     // /usr/bin
java.getNameCount();  // 3
java.getName(1);      // bin
```

- Основные операции с путями реализованы, не надо вручную возиться с разными разделителями на разных ОС/ФС

Работа с файлами

```
Path java = Paths.get("/usr/bin/java");
Files.exists(java);                 // true
Files.isRegularFile(java);          // true
Files.isReadable(java);             // true
Files.size(java);                   // 1536
Files.getLastModifiedTime(java)
        .toMillis();                // 1231914805000
```

- Если файла нет, size() и getLastModifiedTime() бросают NoSuchFileException

Работа с директориями

```
Path usrbin = Paths.get("/usr/bin");
Files.exists(usrbin);          // true
Files.isDirectory(usrbin);     // true

try (DirectoryStream<Path> dirStream =
         Files.newDirectoryStream(usrbin)) {
    for (Path child : dirStream) {
        System.out.println(child);
    }
}
```

- Если директория не существует, `newDirectoryStream()` бросит `NoSuchFileException`
- Есть вариант с фильтром

Модификация файловой системы

- `Path createFile(Path path)`

  Создает пустой файл, если он еще не существует

- `void delete(Path path)`

  Удаляет файл. Если это директория, то она должна быть пустой

- `Path move(Path source, Path target)`
  `Path copy(Path source, Path target)`

  Перемещает или копирует файл

- `Path createDirectory(Path dir)`
  `Path createDirectories(Path dir)`

  Создает директорию или директории

Что осталось за кадром

- Работа с атрибутами файлов
  (права доступа, владелец и т. д.)
- Работа с файловой системой в целом
  (размер, свободное место и т. д.)
- Отслеживание изменений файловой системы
  (создание, удаление, изменение файлов)

- Чтение данных
  `java.io.InputStream`
- Вывод данных
  `java.io.OutputStream`
- Производные классы для конкретных случаев
- Бросают `java.io.IOException` в случае ошибок

java.io.InputStream

- `int read()`

  Возвращает следующий байт из потока или `-1` в случае конца потока

- `int read(byte b[])`
  `int read(byte b[], int off, int len)`

  Считывает байты из потока в массив, возвращает количество считанных байт

- `void close()`

  Закрывает поток и освобождает ресурсы

java.io.InputStream — подклассы

- java.io.FileInputStream

```
new FileInputStream(new File("input.data"))
```

- java.io.ByteArrayInputStream

```
new ByteArrayInputStream(new byte[] {1, 2, 3})
```

- java.io.DataInputStream

```
new DataInputStream(anotherInputStream)
```

- java.util.zip.DeflaterInputStream

```
new DeflaterInputStream(anotherInputStream)
```

java.io.OutputStream

- `void write(int b)`

  Записывает один байт (младшие 8 бит числа) в поток

- `void write(byte b[])`
  `void write(byte b[], int off, int len)`

  Записывает данные из массива в поток

- `void flush()`

  Сбрасывает буферизованные данные в поток

- `void close()`

  Закрывает поток и освобождает ресурсы

java.io.OutputStream — подклассы

- java.io.FileOutputStream

```
new FileOutputStream(new File("output.data"))
```

- java.io.ByteArrayOutputStream

```
new ByteArrayOutputStream()
```

- java.io.DataOutputStream

```
new DataOutputStream(anotherOutputStream)
```

- java.util.zip.DeflaterOutputStream

```
new DeflaterInputStream(anotherOutputStream)
```

- Чтение данных
  `java.io.Reader`
- Вывод данных
  `java.io.Writer`
- Производные классы для конкретных случаев
- Бросают `java.io.IOException` в случае ошибок

java.io.Reader

- `int read()`

  Возвращает следующий символ из потока или `-1` в случае конца потока

- `int read(char cbuf[])`
  `int read(char cbuf[], int off, int len)`

  Считывает символы из потока в массив, возвращает количество считанных символов

- `void close()`

  Закрывает поток и освобождает ресурсы

java.io.Reader — подклассы

- java.io.InputStreamReader

  ```
  new InputStreamReader(inputStream, "UTF-8")
  ```

- java.io.CharArrayReader

  ```
  new CharArrayReader(new char[] {'a', 'b', 'c'})
  ```

- java.io.BufferedReader

  ```
  new BufferedReader(anotherReader)
  ```

  добавляет буферизацию и String readLine()

java.io.Writer

- `void write(int c)`

  Записывает один символ (младшие 16 бит числа) в поток

- `void write(char cbuf[])`
  `void write(char cbuf[], int off, int len)`

  Записывает символы из массива в поток

- `void flush()`

  Сбрасывает буферизованные данные в поток

- `void close()`

  Закрывает поток и освобождает ресурсы

java.io.Writer — подклассы

- java.io.OutputStreamWriter

  ```
  new OutputStreamWriter(outputStream, "UTF-8")
  ```

- java.io.CharArrayWriter

  ```
  new CharArrayWriter()
  ```

- java.io.BufferedWriter

  ```
  new BufferedWriter(anotherWriter)
  ```

  добавляет буферизацию и `void newLine()`

java.io.PrintStream и java.io.PrintWriter

- Являются подклассами java.io.OutputStream и java.io.Writer соответственно
- Добавляют методы print() и println() для всех примитивных типов, строк и объектов
- Добавляют метод printf()

```
printer.printf("e = %+.4f", Math.E) //e = +2,7183
```

- Вместо исключения устанавливают флаг ошибки

System.out

```
public final class System {
    public static final InputStream in;
    public static final PrintStream out;
    public static final PrintStream err;
}
```

Разбор текста на токены

- `java.io.StreamTokenizer`
  - умеет разбирать текст на «слова» и «числа»
- `java.util.Scanner`
  - добавлен в Java 5
  - умеет разбирать все примитивные типы, а также искать токены по произвольному регулярному выражению
  - поддерживает локали

Что сегодня узнали

- Как ходить по файловой системе, получать списки файлов и директорий, читать их атрибуты
- Как читать и писать двоичные данные при помощи потоков байт
- Как читать и писать текстовые данные при помощи потоков символов