### Ввод-вывод

Ввод-вывод осуществляется за счёт модулей:

* io — интерфейс сервера ввода-вывода. Операции являются высокоуровневыми и позволяют проводить построчные чтения, форматированные выводы и т.д.  
  Операции работают как со стандартными потоками, так и с прочими девайсами ввода-вывода (например, файлами);
* file — интерфейс файловой системы;
* filelib — высокоуровневый интерфейс файловой системы.

### Работа с файлами

«Regarding filename encoding, the Erlang VM can operate in two modes. The current mode can be queried using function [native\_name\_encoding/0](https://www.erlang.org/doc/man/file.html#native_name_encoding-0). It returns latin1 or utf8.

In latin1 mode, the Erlang VM does not change the encoding of filenames. In utf8 mode, filenames can contain Unicode characters greater than 255 and the VM converts filenames back and forth to the native filename encoding (usually UTF-8, but UTF-16 on Windows).

The default mode depends on the operating system. Windows, MacOS X and Android enforce consistent filename encoding and therefore the VM uses utf8 mode.

On operating systems with transparent naming (for example, all Unix systems except MacOS X), default is utf8 if the terminal supports UTF-8, otherwise latin1. The default can be overridden using +fnl (to force latin1 mode) or +fnu (to force utf8 mode) when starting [erl](https://www.erlang.org/doc/man/erl.html).»

В Erlang для работы с файлами предусмотрен стандартный модуль file.

Функции, упоминаемые без имени модуля ниже находятся в модуле file.

Перед выполнением действий нужно файл открыть:

|  |
| --- |
| file:open(Path, Mode) -> {ok, IoDevice} | {error, Reason} |

Где:

* Path — путь до требуемого файла;
* Mode — режим работы с файлом.

Таблица 1 — Некоторый перечень режимов работы с файлом

| **Режим** | **Действие** |
| --- | --- |
| read | Файл, который должен существовать, открывается для чтения |
| write | Файл открывается для записи.  Он создается, если он не существует.  Если файл существует, и если запись не объединена с чтением, файл будет усечен |
| append | Файл будет открыт для записи, и он будет создан, если он не существует. Каждая операция записи в файл, открытый с помощью append, будет иметь место в конце файла |
| exclusive | Файл, когда он открыт для записи, создается, если он не существует.  Если файл существует, open вернет {error, exist} |
| numberOfBytes | Количество байтов информации, которое необходимо прочитать из файла |

После использования файла его нужно закрыть:

|  |
| --- |
| close(IoDevice) -> ok | {error, Reason} |

Перечень доступных действий для обработки файлов:

Таблица 2 — некоторый перечень действий, доступных для файлов

| **Функция** | **Вывод** | **Что делает** |
| --- | --- | --- |
| **write(IoDevice, Bytes)** | **ok | {error, Reason}** | Запись байтов в указанный io-девайс |
| **io:format(IoDevice, Format, Data)** | **ok** | Форматированная запись в io-девайс |
| **read(IoDevice, Number)** | **{ok, Data} | eof | {error, Reason}** | Чтение определённого числа байтов из файла. |
| **io:get\_line(IoDevice, Prompt)** | [Data | server\_no\_data()](https://www.erlang.org/doc/man/io.html#type-server_no_data) | Чтение строки из указанного io-девайса |
| **copy(Source, Destination)** | **{ok, BytesCopied} | {error, Reason}** |  |
| **rename(Source, Destination)** | **ok | {error, Reason}** |  |
| **delete(Filename)** | **ok | {error, Reason}** |  |
| **make\_dir(Dir)** | **ok | {error, Reason}** |  |
| **list\_dir(Dir)** | **{ok, Filenames} | {error, Reason}** |  |
| **filelib:is\_dir(Name)** | boolean |  |
| **del\_dir(Dir)** | **ok | {error, Reason}** |  |
| **del\_dir\_r(File)** | **ok | {error, Reason}** |  |
|  |  |  |