JavaScript Runtime

On Server Side

File & Stream

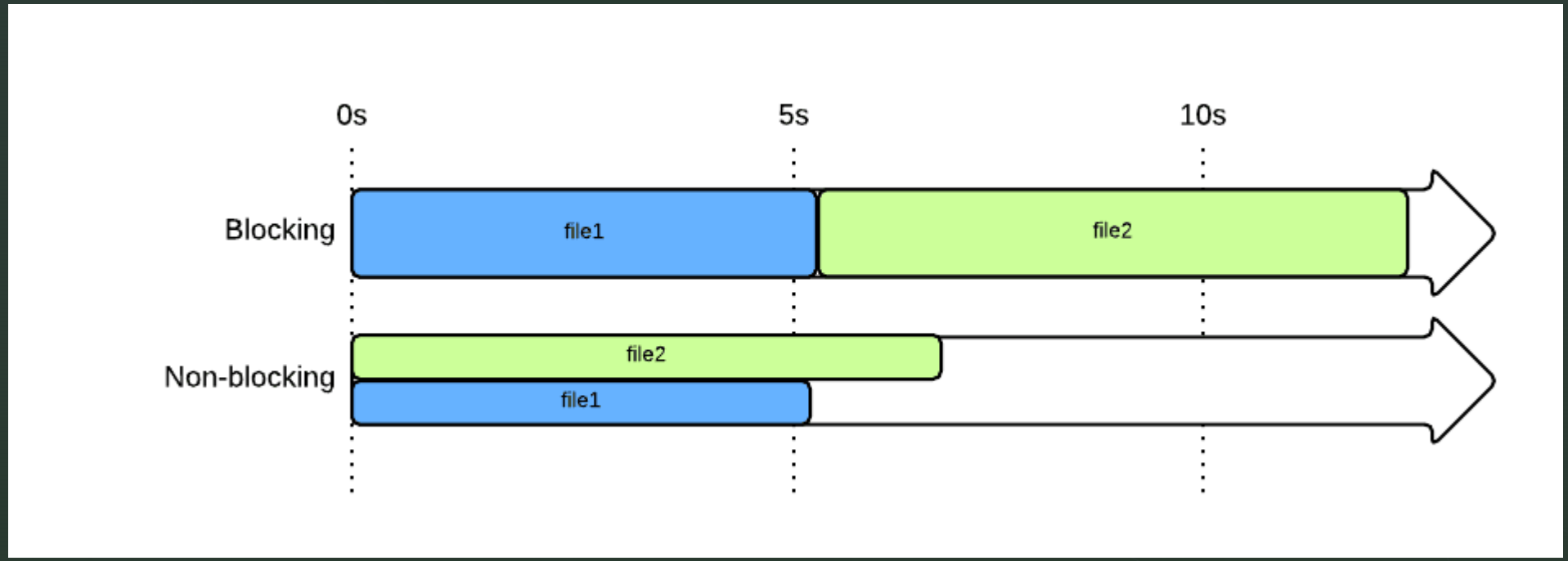
Nodemy

Đọc ghi file trong Node.js

- Xử lý file đồng bộ & bất đồng bộ
- Nên sử dụng phương thức không đồng bộ
- Sử dụng module `fs` để thao tác với File

Nodemy

Lưu ý khi đọc, ghi file



Nodemy

Một ví dụ cơ bản

```
var fs = require("fs");

// Phuong thuc doc file khong dong bo
fs.readFile('input.txt', function (err, data) {
   if (err) {
       return console.error(err);
   }
   console.log("Phuong thuc doc file khong dong bo: " + data.toString());
});

// Phuong thuc doc file dong bo
var data = fs.readFileSync('input.txt');
console.log("Phuong thuc doc file dong bo: " + data.toString());

console.log("Ket thuc chuong trinh");
```

Nodemy

FS API

- Mở một File trong Node.js
  - fs.open(path, flags[, mode], callback)

FS API

- Mở một File trong Node.js
  - fs.open(path, flags[, mode], callback)
- Lấy thông tin File
  - fs.stat(path, callback)
- Xóa file
  - fs.unlink(path, callback)
- Tạo thư mục
  - fs.mkdir(path[, mode], callback)

Phương thức	Miêu tả
stats.isFile()	Trả về true nếu đó là một file
stats.isDirectory()	Trả về true nếu đó là một thư mục
stats.isBlockDevice()	Trả về true nếu đó là một Block Device.
stats.isCharacterDevice()	Trả về true nếu đó là một Character Devi
stats.isSymbolicLink()	Trả về true nếu đó là một Symbolic Link.
stats.isFIFO()	Trả về true nếu đó là một kiểu FIFO.
stats.isSocket()	Trả về true nếu đó là một kiểu Socket.

Nodemy

FS API

- Đọc thư mục và liệt kê files
  - fs.readdir(path, callback)
- Xóa thư mục
  - fs.rmdir(path, callback)
- Đọc & ghi file
  - fs.readFile(path[, options], callback)
  - fs.writeFile(file, data[, options], callback)

Nodemy

Stream

- Stream là các đối tượng cho phép bạn đọc dữ liệu từ một nguồn và ghi dự liệu đến một đích.
- Có 4 loại:
  - Readable - sử dụng để cho hoạt động đọc
  - Writable - sử dụng cho hoạt động ghi
  - Duplex - sử dụng cho cả mục đích ghi và đọc
  - Transform - là một kiểu Duplex Stream, khác ở chỗ là kết quả đầu ra được tính toán dựa trên dữ liệu nhập vào.

Stream

- Mỗi loại Stream là một sự thể hiện của đối tượng **EventEmitter**
  - **data** - Sự kiện khi dữ liệu là có sẵn cho hoạt động đọc.
  - **end** - Sự kiện khi không còn dữ liệu nào để đọc nữa.
  - **error** - Sự kiện xảy ra bất kỳ lỗi nào trong việc đọc và ghi dữ liệu.
  - **finish** - Sự khi tất cả dữ liệu đã được chuyển hết tới vùng hệ thống cơ sở

Nodemy

Đọc dữ liệu từ Stream

```
var fs = require("fs");
var data = '';

// Tao mot Readable Stream
var readerStream = fs.createReadStream('input.txt');

// Thiet lap encoding la utf8.
readerStream.setEncoding('UTF8');

// Xu ly cac su kien lien quan toi Stream --> data, end, va
readerStream.on('data', function(chunk) {
   data += chunk;
});

readerStream.on('end',function(){
   console.log(data);
});

readerStream.on('error', function(err){
   console.log(err.stack);
});

console.log("Ket thuc chuong trinh");
```

Nodemy

Ghi dữ liệu vào Stream

```
var fs = require("fs");
var data = 'VietNamVoDoi';

// Tao mot Writable Stream
var writerStream = fs.createWriteStream('output.txt');

// Ghi du lieu toi Stream theo ma hoa utf8
writerStream.write(data,'UTF8');

// Danh dau diem cuoi cua file (end of file)
writerStream.end();

// Xu ly cac su kien lien quan toi Stream --> finish, va
writerStream.on('finish', function() {
    console.log("Ket thuc hoat dong ghi.");
});

writerStream.on('error', function(err){
   console.log(err.stack);
});

console.log("Ket thuc chuong trinh");
```

Piping Stream

- Pipe là một kỹ thuật cung cấp kết quả đầu ra của một Stream để làm dữ liệu đầu vào cho một Stream khác
- Quá trình piping có thể vẫn tiếp tục nếu vẫn còn đường ống phía sau
- Ví dụ đọc dữ liệu từ một file, sau đó ghi dữ liệu đó tới một file khác

Piping Stream

```
var fs = require("fs");

// Tao mot Readable Stream
var readerStream = fs.createReadStream('input.txt');

// Tao mot Writable Stream
var writerStream = fs.createWriteStream('output.txt');

// Piping hoat dong Readable va Writable
// Doc du lieu tu input.txt va ghi du lieu toi output.txt
readerStream.pipe(writerStream);

console.log("Ket thuc chuong trinh");
```

Nodemy

Chaining Stream

- Chaining là kỹ thuật để kết nối kết quả đầu ra của một Stream tới một Stream khác, tạo một chuỗi bao gồm nhiều hoạt động Stream
- Thường được sử dụng với các hoạt động Piping
- Ví dụ đọc một file, sau đó nén file đó

Nodemy

Chaining Stream

```
var fs = require("fs");
var zlib = require('zlib');

// Nen input.txt thanh input.txt.gz
fs.createReadStream('input.txt')
  .pipe(zlib.createGzip())
  .pipe(fs.createWriteStream('input.txt.gz'));

console.log("File duoc nen thanh cong.");
```

Nodemy

References

- https://nodejs.org/api/fs.html
- https://nodejs.org/api/stream.html