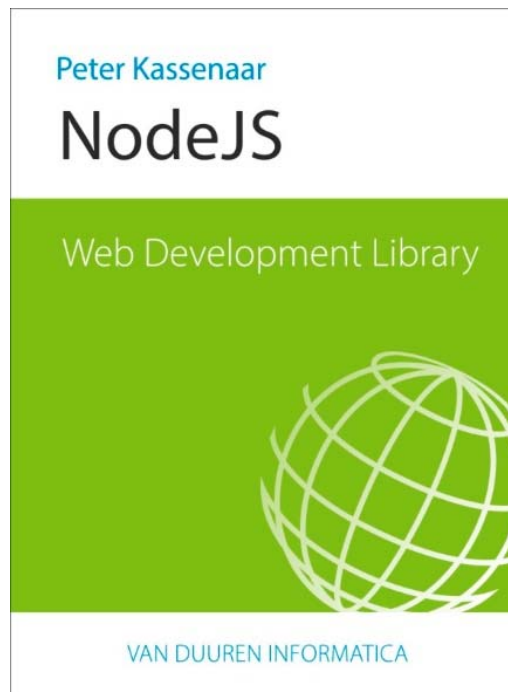


**Node.js**



Peter Kassenaar

Module 2 – Core Node.js modules

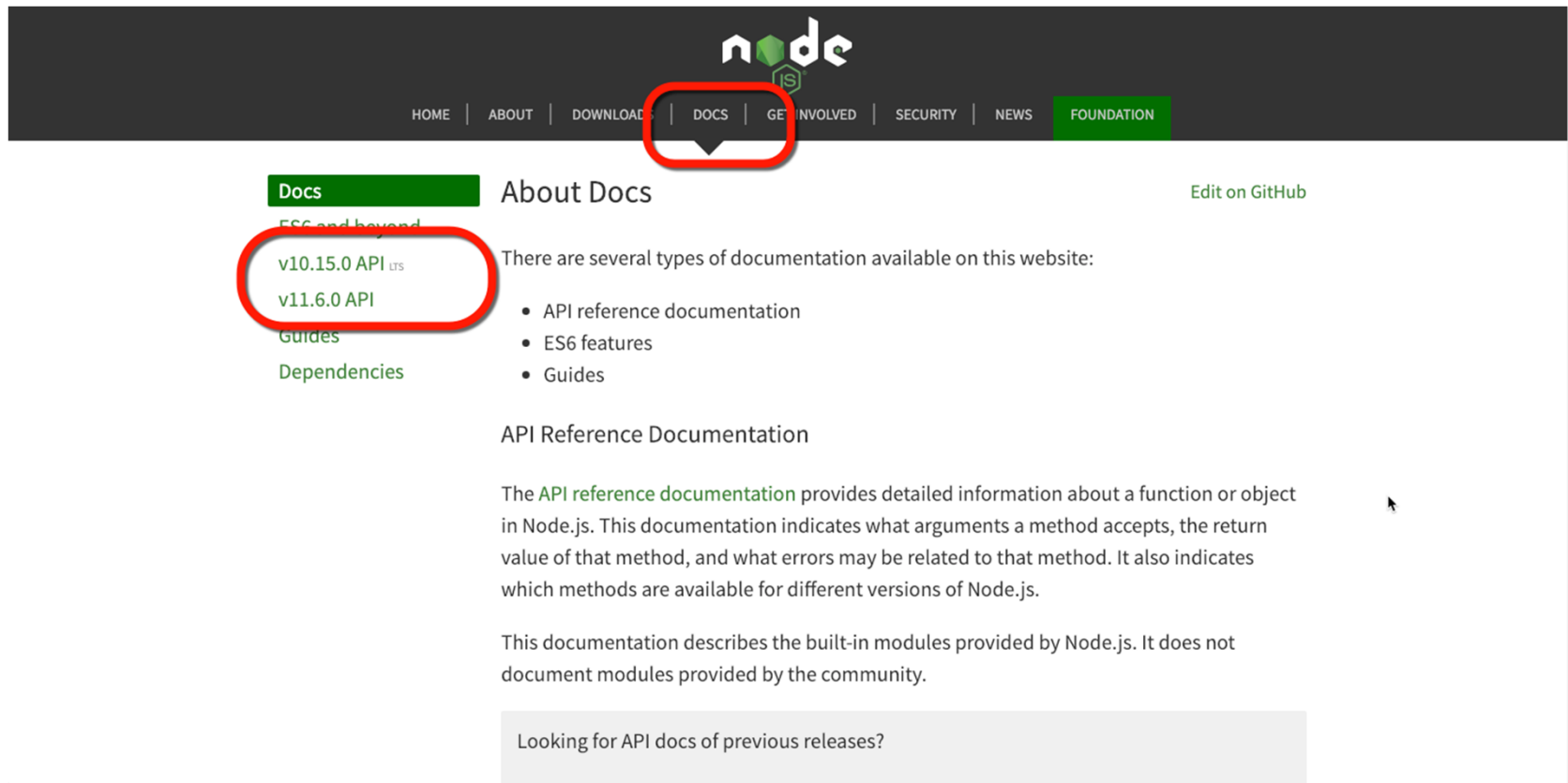


Hoofdstuk 4, p.74 en verder

# Ingebouwde Node.js-variabelen

- Console
- Timers
- `__filename`
- `__dirname`

# Check the docs!



The screenshot shows the Node.js website. The navigation bar at the top has links for HOME, ABOUT, DOWNLOAD, DOCS, GET INVOLVED, SECURITY, NEWS, and FOUNDATION. The 'DOCS' link is circled in red. Below the navigation bar, the 'About Docs' page is displayed. On the left sidebar, there are links for 'Docs', 'ES6 and beyond', 'v10.15.0 API LTS', 'v11.6.0 API', 'Guides', and 'Dependencies'. The 'v10.15.0 API LTS' link is circled in red. The main content area has the heading 'About Docs' and a link 'Edit on GitHub'. It states: 'There are several types of documentation available on this website:' followed by a list: 'API reference documentation', 'ES6 features', and 'Guides'. Below this, it says 'API Reference Documentation' and explains that the 'API reference documentation' provides detailed information about functions or objects in Node.js. It also mentions that this documentation describes built-in modules but not community-provided modules. At the bottom, there is a link: 'Looking for API docs of previous releases?'.

node

HOME | ABOUT | DOWNLOAD | DOCS | GET INVOLVED | SECURITY | NEWS | FOUNDATION

Docs

ES6 and beyond

v10.15.0 API LTS

v11.6.0 API

Guides

Dependencies

## About Docs

[Edit on GitHub](#)

There are several types of documentation available on this website:

- API reference documentation
- ES6 features
- Guides

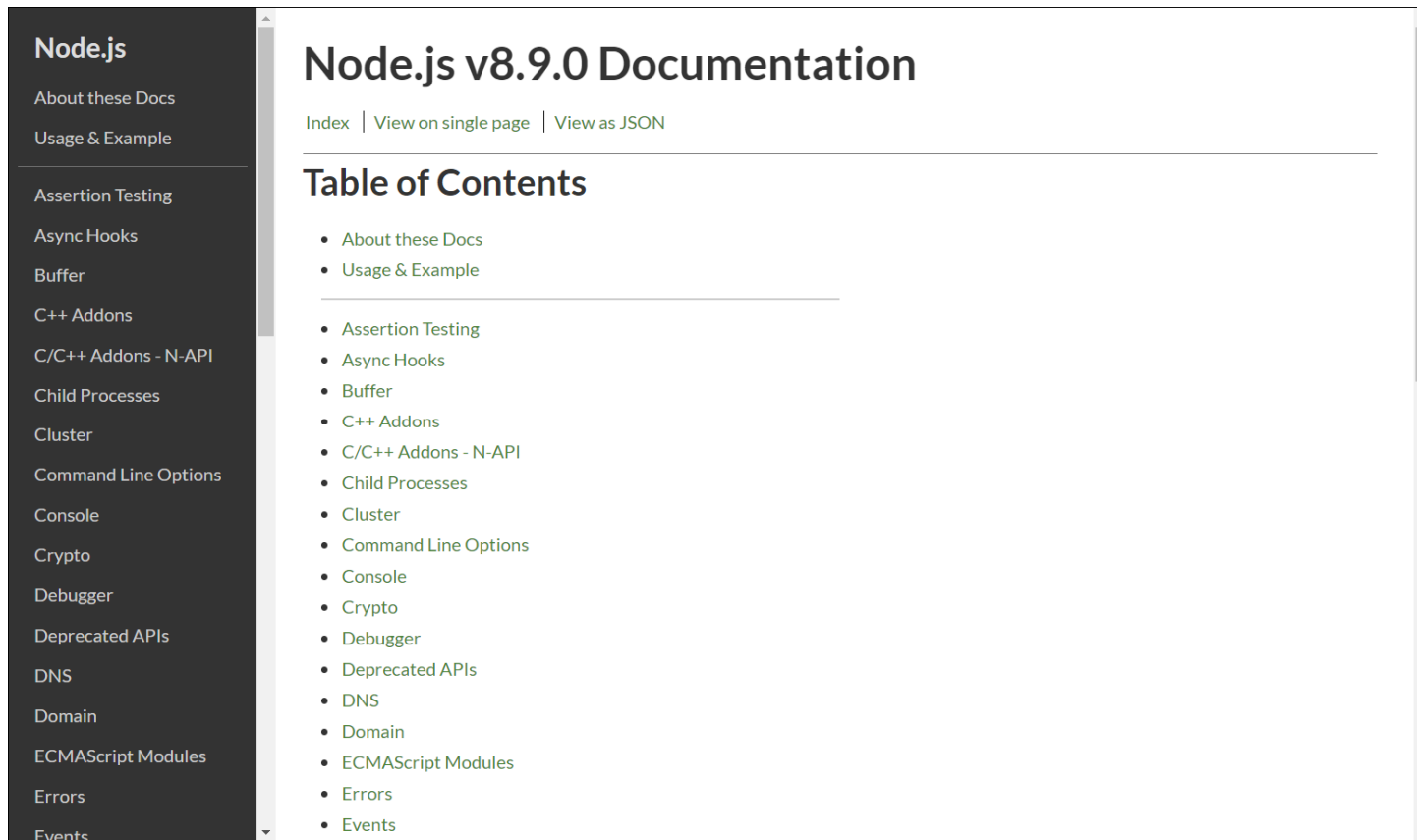
### API Reference Documentation

The [API reference documentation](#) provides detailed information about a function or object in Node.js. This documentation indicates what arguments a method accepts, the return value of that method, and what errors may be related to that method. It also indicates which methods are available for different versions of Node.js.

This documentation describes the built-in modules provided by Node.js. It does not document modules provided by the community.

[Looking for API docs of previous releases?](#)

# API Documentation



**Node.js**

About these Docs  
Usage & Example

Assertion Testing  
Async Hooks  
Buffer  
C++ Addons  
C/C++ Addons - N-API  
Child Processes  
Cluster  
Command Line Options  
Console  
Crypto  
Debugger  
Deprecated APIs  
DNS  
Domain  
ECMAScript Modules  
Errors  
Events

## Node.js v8.9.0 Documentation

[Index](#) | [View on single page](#) | [View as JSON](#)

### Table of Contents

- [About these Docs](#)
- [Usage & Example](#)
- [Assertion Testing](#)
- [Async Hooks](#)
- [Buffer](#)
- [C++ Addons](#)
- [C/C++ Addons - N-API](#)
- [Child Processes](#)
- [Cluster](#)
- [Command Line Options](#)
- [Console](#)
- [Crypto](#)
- [Debugger](#)
- [Deprecated APIs](#)
- [DNS](#)
- [Domain](#)
- [ECMAScript Modules](#)
- [Errors](#)
- [Events](#)

<https://nodejs.org/dist/latest-v12.x/docs/api/>

# Console

- Meldingen tonen in stdout / consolevenster
- `console.log()`, `console.error()`, `console.info()`
  - Standaard meldingen
- `console.time()` – `console.timeEnd()`
  - Meten van prestaties



# Timers

- Eenmalig taak uitvoeren
  - `setTimeout(callbackFn, delay)`
- Herhaald taak uitvoeren
  - `setInterval(callbackFn, delay)`
- Clear timer
  - `clearTimeout()`, `clearInterval()`

Zoals bij alle JavaScript-timers: execution wordt niet gestopt.

## Console:

```
// verschillende voorbeelden van console-meldingen
console.log('Algemene logging-melding');
console.info('Algemene informatie-melding');
console.time('Tijdsduur van for-loop');
var j = 0;
for (var i = 0; i < 10000000; i += 1) {
    j += i;
}
console.log('j: ', j);
console.timeEnd('Tijdsduur van for-loop');
```

## Timer:

```
// #1 - met inline anonieme functie
setTimeout(function () {
    console.log('Er zijn drie seconden verstreken!')
}, 3000);
```



## \_\_filename en \_\_dirname

- Globale variabele voor huidige bestandsnaam en padnaam
- Handig voor serveren statische files en lezen/schrijven bestanden

*// Magic variable names: \_\_filename en \_\_dirname*

```
console.log('De bestandsnaam is: ', __filename);
```

```
console.log('De huidige directory is: ', __dirname);
```

# De module `File System`

- Globaal, maar wel insluiten : `require('fs')`
- Werken met bestanden: lezen, schrijven, verwijderen
- Standaard: asynchrone acties

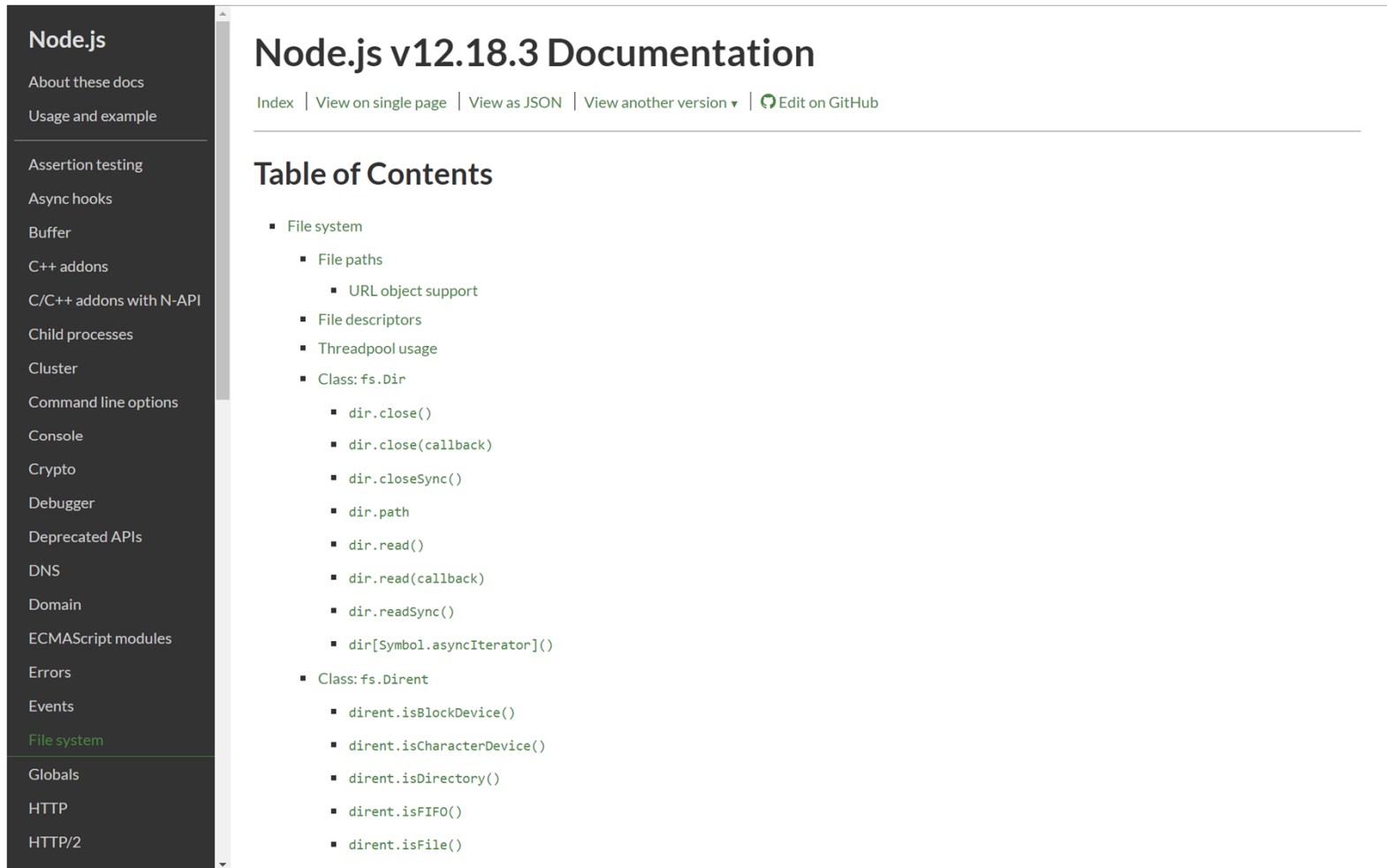
```
fs.readFile('filename', [options], callback(err, data){  
    ...  
}
```

De inhoud van het file wordt als `data` doorgegeven aan de callbackfunctie

# Sync vs. Async

- Indien gewenst: synchrone variant aanwezig
  - Liever niet gebruiken: blocking
  - `fs.readFile()` vs. `fs.readFileSync()`
  - `fs.writeFile()` vs. `fs.writeFileSync()`
  - Enzovoort
- Voor onbekende bestandsgrootten: Streams gebruiken
  - Data leveren zodra gereed. Niet wachten tot complete file is gelezen.
  - Werken met events (`data`, `end`, `error`)
  - `fs.createReadStream()`, `fs.createWriteStream()`

# Tal van functies



The screenshot shows the Node.js v12.18.3 Documentation page for the File system module. The left sidebar contains a list of navigation links, with 'File system' highlighted in green. The main content area displays the 'Table of Contents' for the File system module, listing various functions and classes.

**Node.js**

About these docs  
Usage and example

Assertion testing  
Async hooks  
Buffer  
C++ addons  
C/C++ addons with N-API  
Child processes  
Cluster  
Command line options  
Console  
Crypto  
Debugger  
Deprecated APIs  
DNS  
Domain  
ECMAScript modules  
Errors  
Events  
**File system**  
Globals  
HTTP  
HTTP/2

## Node.js v12.18.3 Documentation

[Index](#) | [View on single page](#) | [View as JSON](#) | [View another version ▼](#) | [Edit on GitHub](#)

### Table of Contents

- File system
  - File paths
    - URL object support
  - File descriptors
  - Threadpool usage
  - Class: `fs.Dir`
    - `dir.close()`
    - `dir.close(callback)`
    - `dir.closeSync()`
    - `dir.path`
    - `dir.read()`
    - `dir.read(callback)`
    - `dir.readSync()`
    - `dir[Symbol.asyncIterator]()`
  - Class: `fs.Dirent`
    - `dirent.isBlockDevice()`
    - `dirent.isCharacterDevice()`
    - `dirent.isDirectory()`
    - `dirent.isFIFO()`
    - `dirent.isFile()`

<https://nodejs.org/dist/latest-v12.x/docs/api/fs.html>

# Voorbeeld File System

*// Verschillende methodes om te werken met het file system*

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

```
var msg = 'Hello World';
```

*// #1. Bestand opslaan*

```
fs.writeFile('hello.txt', msg, function () {
```

```
    console.log('bestand opgeslagen!')
```

```
});
```

*// #2. Bestand inlezen en in de console tonen*

```
fs.readFile('hello.txt', 'utf8', function (err, data) {
```

```
    if (err) {
```

```
        console.log('Error: ', err);
```

```
    } else {
```

```
        console.log('bestand ingelezen: ', data);
```

```
    }
```

```
});
```

## Voorbeeld Stream

```
var stream = fs.createReadStream(fileName);
stream.on('data', function (chunk) {
    res.write(chunk);
});
stream.on('end', function () {
    res.end();
});
stream.on('error', function (err) {
    console.log('error: ' + err);
});
```

# Pipes

*"The pipe() function reads data from a readable stream as it becomes available and writes it to a destination writable stream."*

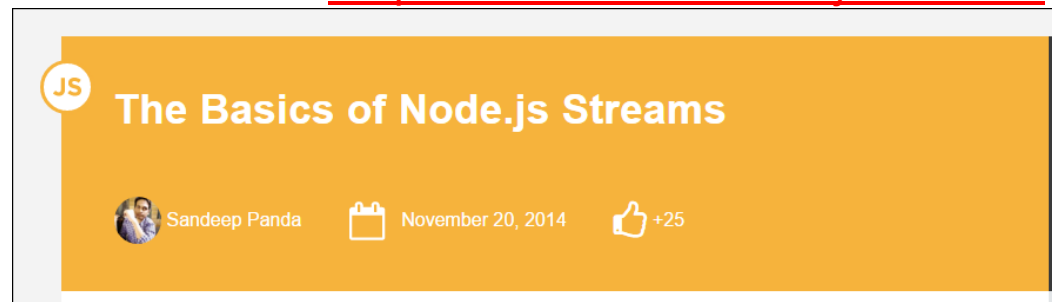
```
// File inlezen
```

```
var read = fs.createReadStream('lorem.txt');
```

```
var write = fs.createWriteStream('lorem_copy.txt');
```

```
read.pipe(write);
```

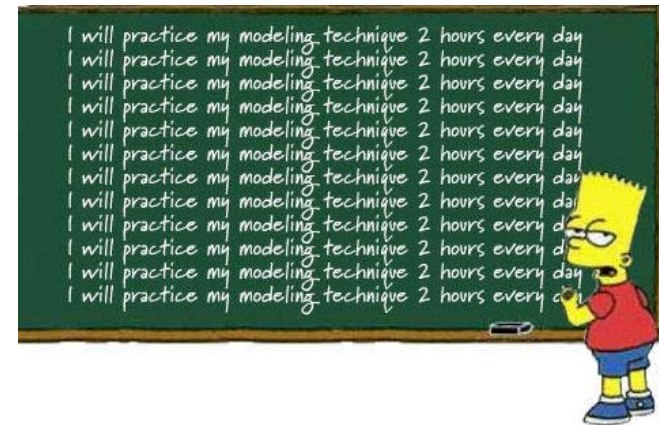
Zie ook [sitepoint.com/basics-node-js-streams/](http://sitepoint.com/basics-node-js-streams/)



# Checkpoint

- Er zijn tal van core modules
- Belangrijk: console, file system, timers, http, OS, path, Stream
- File operations: preferably async
- Streams: werken met events.

## Oefening....





# Serving static files

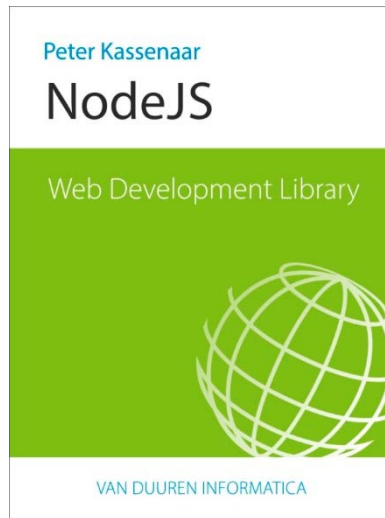
- Webserver == serving static files over http.
- Dus minimaal includen: `http` en `fs`.
  - Rest == sugar

```
var http = require('http'),  
    fs = require('fs'),  
    path = require('path'),  
    root = __dirname + '/public/'; // magic variable
```

# Stappenplan

1. Maak server (`http.createServer`)
2. Onderzoek binnenkomend request
3. Check of bestand bestaat.
  - Zo ja – serveren
  - Zo nee – 404 terugsturen





p.83 - 93

Demo files: \H04\0406\_Server\server\_01.js tot en met server\_03.js

# Regels voor modules in Node.js

## 1. Kijken naar ingebouwde module

```
require('http')
```

## 2. Ga kijken in `node_modules` naar de aangegeven dependencies in `package.json`

```
require('moment')
```

## 3. Ga kijken in aangegeven directory

```
require('./moment')
```

## 4. Crash – “module not found”

# Checkpoint

- Er zijn tal van core modules
- Belangrijk: console, file system, timers, http, OS, path, Stream
- File operations: preferably async
- Streams: werken met events.

## Oefening....

