NODE FILE SYSTEM

FS

READFILE

```
const fs = require(‘fs’);

// sync version
const content = fs.readFileSync('blogpost.json', ‘utf8');
console.log(content);
```

```
// async version
fs.readFile('blogpost.json', 'utf8', (err, content) => {
     if (err) throw err;
     console.log(content);
});
```

Node fs

WRITEFILE

```
const fs = require('fs');
const data = {
    title : ‘My blog post'
};

fs.writeFile('blogpost.json', JSON.stringify(data), (err) => {
    if (err) {
        throw err;
    } else {
        console.log('written file succesfully');
    }
});
```

Node fs

WRITEFILESYNC

```
const fs = require('fs');
const data = {
    title : ‘My blog post'
};

fs.writeFileSync('blogpost.json', JSON.stringify(data));

console.log('written file succesfully');
```

Node fs

MKDIR

```
const fs = require('fs');
const path = require('path');
const data = {
     title: 'My blog post'
};
const filePath = 'dist/blogpost.json';

fs.writeFile(filePath, JSON.stringify(data), (err) => {
    if (err) {
        throw err;
    } else {
        console.log('written file succesfully');
    }
});
```

Node fs

MKDIR

```
const fs = require('fs');
const path = require('path');
const data = {
    title: ‘My blog post'
};
const filePath = 'dist/blogpost.json';
const dirname = path.dirname(filePath);

if (!fs.existsSync(dirname)) {
    fs.mkdir(dirname, () => {
        writeFile(filePath, data);
    });
} else {
    writeFile(filePath, data);
}

function writeFile(filePath, data) {
    fs.writeFile(filePath, JSON.stringify(data), (err) => {
        if (err) {
            throw err;
        } else {
            console.log('written file succesfully');
        }
    });
}
```

Node fs

MKDIRP

```
const fs = require('fs');
const path = require('path');
const mkdirp = require('mkdirp');
const data = {
    title: ‘My blog post'
};
const filePath = 'dist/blogpost/blogpost.json';
const dirname = path.dirname(filePath);
mkdirp.sync(dirname);

fs.writeFile(filePath, JSON.stringify(data), (err) => {
    if (err) {
        throw err;
    } else {
        console.log('written file succesfully');
    }
});
```

NPM mkdirp

MORE FS

- **fs.rename()** // rename
- **fs.unlink()** // delete
- **fs.watch()** // watch for changes
- **fs.createReadStream()** // read a file as stream
- **fs.createWriteStream()** // write a file as stream
- **fs.Stats** // returns object with e.g. **isDirectory()**, **isFile()** and more
- **and more...**

Node fs

PATH

PATH

utilities for working with file and directory paths

- **path.basename**(‘/content/blogposts/my-blogpost.json’);

returns ‘my-blog-post.json’

- **path.dirname**(‘/content/blogposts/my-blogpost.json’);

returns ‘/content/blogposts’

- **path.parse**(‘/content/blogposts/my-blogpost.json’);

returns {

root: “/”,

dir: “/content/blogposts”,

base: ”my-blogpost.json”,

ext: “json”,

name: “my-blogpost”

}

Node path

PATH

utilities for working with file and directory paths

- **path.join**(‘content‘, ‘//blogpost/‘, ‘my-blogpost/media’);

returns ‘/content/blogpost/my-blogpost/media‘

- **path.normalize**(‘/content//blogposts’);

returns ‘/content/blogposts‘

- **path.relative**(‘/dist/my-blogpost/index.html‘, ‘/dist/assets/css/main.css’);

returns ‘../assets/css/main.css‘

- and more

Node path

GLOB

GLOB

Match files using the patterns the shell uses, like stars and stuff

```
- /file.json
- /folder/nestedFile.json
- /folder/nestedFolder/deeplyNestedFile.json
```

```
const glob = require("glob");

glob("*.json", function (err, files) {
    // returns [‘file.json’]
});

glob("*/*.json", function (err, files) {
    // returns [‘file.json’,
    //          ‘folder/nestedFile.json’]
});

glob("**/*.json", function (err, files) {
    // returns [‘file.json’,
    //          ‘folder/nestedFile.json’,
    //          ‘folder/nestedFolder/deeplyNestedFile.json’]
});
```

NPM glob

DE VOORHOEDE

front-end developers