Le module FS (FileSystem)

Agenda

- **Présentation**
- **Création d'un fichier**
- **Lecture d'un fichier**
- **Synchrone ou asynchrone ?**
- **Suppression d'un fichier (asynchrone)**
- **Suppression d'un fichier (synchrone)**
- **Ordre des opérations**
- Gestion des dossiers

Présentation

- Le module du core « fs » de Node.js fournit une API qui permet d'interagir avec le système de fichiers et d'effectuer certaines opérations IO comme créer, lire ou écrire un fichier.
- Comme « fs » est un module du core, nous n'avons pas besoin de la configurer avant de l'utiliser.
  Pour le charger, comme tout autre module CommonJS, on utilisera l'instruction require() :
- var fs = require("fs");

Création d’un fichier

- Plusieurs methodes pour créer un fichier : write, writeFile ou createWriteStream. Le plus simple est d’utiliser la function writeFile() du module :

- ```
  var fs = require('fs');
  fs.writeFile("./test", "Hey there!", function(err) {
  if(err) {
  return console.log(err);
  }
  console.log("The file was saved!");
  });
  ```
  ```

Lecture d’un fichier: readFile

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

fs.readFile('./test', 'utf8', function(err, contents) {

    console.log(contents);

});

console.log('after calling readFile');
```

Synchrone ou asynchrone ?

- Toutes les fonctions de ce module peuvent être utilisées de manière **synchrone** ou **asynchrone**.
- Si vous utilisez la forme asynchrone, vous devrez utiliser une fonction de callback.
- Les arguments passés à cette fonction de callback dépendent de la fonction utilisée.
- Cependant, de manière générale, le premier argument est réservé à une exception. Si l'opération
  s'est bien passée, la valeur retournée par cet argument sera null ou undefined.
- Les fonctions synchrone reprennent souvent le nom de la fonction asynchrone suivi de « Sync ».

Synchrone ou asynchrone ?

- Ce code d'exemple supprime le fichier /tmp/hello. On utilise la fonction unlink() du module « fs ».

- ```
  const fs = require('fs');
  fs.unlink('/tmp/hello', (err) => {
  if (err) {
  throw err;
  }
  console.log('successfully deleted
  /tmp/hello');
  });
  ```

- ```
  const fs = require('fs');
  fs.unlinkSync('/tmp/hello');
  console.log('successfully
  deleted /tmp/hello');
  ```

Ordre des opérations

- Avec des fonctions asynchrones, l'ordre des opération n'est pas garantie !
- fs.stat pourrait très bien être exécuté avant fs.rename.
- Donc le code suivant est sujet à erreurs

```
fs.rename('/tmp/hello', '/tmp/world', (err) => {
    if (err) throw err;
    console.log('renamed complete');
});

fs.stat('/tmp/world', (err, stats) => {
    if (err) throw err;
    console.log(`stats: ${JSON.stringify(stats)}`);
});
```

Ordre des opérations

- En programmation asynchrone, vous devez toujours vous baser sur les callbacks.
- Utiliser une chaine de callback entre rename et stat.

```
fs.rename('/tmp/hello', '/tmp/world', (err) => {
    if (err) {
        throw err;
    }
    fs.stat('/tmp/world', (err, stats) => {
        if (err) {
            throw err;
        }
        console.log(`stats:
        ${JSON.stringify(stats)}`);
    });
});
```

Gestion des dossiers

- Créer
- Lister le contenu
- Renommer
- Supprimer

Créer un nouveau dossier

```
const fs = require('fs')

const folderName = '/Users/joe/test'

try {
  if (!fs.existsSync(folderName)) {
    fs.mkdirSync(folderName)
  }
} catch (err) {
  console.error(err)
}
```

Lire le contenu d'un dossier

```
const fs = require('fs')
const path = require('path')

const folderPath = '/temp'

console.log(fs.readdirSync(folderPath))
```

* Fichiers uniquement

```
const isFile = fileName => {
    return fs.lstatSync(fileName).isFile()
  }

  fs.readdirSync(folderPath).map(fileName => {
    return path.join(folderPath, fileName)
  })
  .filter(isFile)
```

Renommer un dossier

```
const fs = require('fs')

fs.rename('/Users/joe', '/Users/roger', err => {
  if (err) {
    console.error(err)
    return
  }

})
```

Supprimer un dossier

N.B: Installer le module fs-extra pour supprimer un dossier non vide: npm install fs-extra

```
// synchrone

const fs = require('fs-extra')

const folder = '/Users/joe'

fs.remove(folder, err => {
  console.error(err)
})
```

```
// asynchrone

async function removeFolder(folder) {
    try {
      await fs.remove(folder)
      //done
    } catch (err) {
      console.error(err)
    }
  }

  const folder = '/Users/joe'
  removeFolder(folder)
```