Node.js alapozó guide

1 Node File read and write howto

1.1 Version 1

Fájl olvasás és kiírás a konzolra

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

fs.readFile(__filename, {encoding: 'utf8'}, gotFileContents);

function gotFileContents(err, result) {
  if(err) {
    return console.error(err);
  }

  console.log(result);
}
```

1.2 Version 2

Fájl olvasás és kiírás fájlba

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

var dir = path.join(__dirname, 'temp');
var source = __filename;
var target = path.join(dir, 'target.js');

fs.mkdir(dir, function(err, result) {
  if(err) {
    return handleError(err);
  }

  fs.readFile(source, {encoding: 'utf8'}, function(err, result) {
    if(err) {
      return handleError(err);
    }

    fs.writeFile(target, result, function(err, result) {
      if(err) {
        return handleError(err);
      }
      console.log("All done.");
    })
  });
});

function handleError(err) {
  console.error(err);
};
```

1.3 Version 3

Függvények kiemelése, error kezelővel

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

var dir = path.join(__dirname, 'temp');
var source = __filename;
var target = path.join(dir, 'target.js');

fs.mkdir(dir, handlingError(folderCreated));

function folderCreated(result) {
  fs.readFile(source, {encoding: 'utf8'}, handlingError(fileOpened));
}

function fileOpened(result) {
  fs.writeFile(target, result, handlingError(fileWritten))
}

function fileWritten(result) {
  console.log("All done.");
}

function handlingError(callback) {
  return function(err, result) {
    if(err) {
      return handleError(err);
    }
    callback(result);
  }
}

function handleError(err) {
  console.error(err);
};
```

1.4 Version 4

Promise megoldás

```
var fs = require('fs');
var path = require('path');
var Promise = require('promise');

var dir = path.join(__dirname, 'temp');
var source = __filename;
var target = path.join(dir, 'target.js');

createFolder(dir)
  .then(openFile)
  .then(writeFile)
  .then(function() {
    console.log('All done');
  })
  .catch(handleError);

function createFolder(dirName) {
  return new Promise(function(resolve, reject) {
    fs.mkdir(dirName, handlingError(resolve, reject));
  });
}

function openFile() {
  return new Promise(function(resolve, reject) {
    fs.readFile(source, {encoding: 'utf8'}, handlingError(resolve, reject));
  });
}

function writeFile(content) {
  return new Promise(function(resolve, reject) {
    fs.writeFile(target, content, handlingError(resolve, reject))
  });
}

function handlingError(resolve, reject) {
  return function(err, result) {
    if(err) {
      return reject(err);
    }
    resolve(result);
  }
}

function handleError(err) {
  console.error(err);
};
```

1.5 Version 5

Denodify-os megoldás

```
var fs = require('fs');
var path = require('path');
var Promise = require('promise');

var dir = path.join(__dirname, 'temp');
var source = __filename;
var target = path.join(dir, 'target.js');

var fs_mkdir = Promise.denodeify(fs.mkdir);
var fs_readFile = Promise.denodeify(fs.readFile);
var fs_writeFile = Promise.denodeify(fs.writeFile);

fs_mkdir(dir)
  .then(openFile)
  .then(writeFile)
  .then(function() {
    console.log('All done');
  })
  .catch(handleError);

function openFile() {
  return fs_readFile(source, {encoding: 'utf8'});
}

function writeFile(content) {
  return fs_writeFile(target, content);
}

function handleError(err) {
  console.error(err);
};
```

2 Stream howto

2.1 Első lépés

1. Projekt base letöltése:
2. CMD elindítása, mappa megnyitás
3. Adatforrás generálás
    a. cd data-source
    b. node create-source.js
4. Lib mappa átnézése

2.2 Read stream

2.2.1 Version 1

```
var ReadStream = require('./lib/readStream.js');
var stream = new ReadStream();
stream.on('readable', function() {
  while (null !== (record = stream.read())) {
    console.log('received: ' + JSON.stringify(record));
  }
});
stream.on('end', function() {
  console.log('done');
});
```

2.2.2 Version 2

```
var ReadStream = require('./lib/readStream.js');
var stream = new ReadStream();
stream.on('data', function(record) {
  console.log('received: ' + JSON.stringify(record));
});
stream.on('end', function() {
  console.log('done');
});
```

2.2.3 Version 3

```
var ReadStream = require('./lib/readStream.js');
var stream = new ReadStream();
stream.on('data', function(record) {
  console.log('received: ' + JSON.stringify(record));
  console.log('pausing stream for 2 seconds');
  stream.pause();
  setTimeout(function() {
    console.log('resuming stream');
    stream.resume();
  },2000);
});
stream.on('end', function() {
  console.log('done');
});
```

2.3 Write stream

Nézzük meg mit csinált a writeStream

2.3.1 Version 1

```
var ReadStream = require('./lib/readStream.js'),
  WriteStream = require('./lib/writeStream.js');

var rs = new ReadStream();
var ws = new WriteStream();

rs.pipe(ws);
```

2.3.2 Version 2

Nézzük meg mit csinál a writeStream 2

```
var ReadStream = require('./lib/readStream.js'),
  WriteStream = require('./lib/writeStream_v2.js');

var rs = new ReadStream();
var ws = new WriteStream();

rs.pipe(ws);
```

2.4 Transform stream

Nézzük meg mit csinál a transformStream

```
var ReadStream = require('./lib/readStream.js'),
  WriteStream = require('./lib/writeStream.js'),
  TransformStream = require('./lib/transformStream.js');

var rs = new ReadStream();
var ws = new WriteStream();
var ts = new TransformStream();

rs.pipe(ts).pipe(ws);
```

3 Testing

3.1.1 Projekt inicializálás

1. npm init
2. npm install -g mocha
3. npm install chai –save-dev
4. Amit tesztelni fogunk, tags.js
    a. Snippet bemásolás

3.2 Egyszerű teszt írás

1. tags.spec.js létrehozás

```
var expect = require("chai").expect;
var tags = require("./tags.js");

describe("Tags", function(){
  describe("#parse()", function() {
    it("should parse long formed tags", function(){
      var args = ["--depth=4", "--hello=world"];
      var results = tags.parse(args);
      expect(results).to.have.property("depth", 4);
      expect(results).to.have.property("hello", "world");
```

```
    });
  });
});
```

3.3 Lehesssen megadni default értéket

1. Először írjuk meg az új tesztet

```
describe("Tags", function(){
  describe("#parse()", function() {
    var args;
    before(function() {
      args = ["--depth=4", "--hello=world"];
    });

    it("should parse long formed tags", function(){
      var results = tags.parse(args);
      expect(results).to.have.property("depth", 4);
      expect(results).to.have.property("hello", "world");
    });

    it("should fallback to defaults", function(){
      var defaults = { depth: 2, foo: "bar" };
      var results = tags.parse(args, defaults);

      var expected = {
        depth: 4,
        foo: "bar",
        hello: "world"
      };

      expect(results).to.deep.equal(expected);
    });
  });
});
```

2. Futtassuk, el fog failelni
3. Javítsuk ki a tags.js-t

```
exports.parse = function(args, defaults) {
  // Kiadni ------------------------------------
  var options = {};
  // v2 ----------------------------------------
  if (typeof defaults === "object" && !(defaults instanceof Array)) {
    options = defaults
```