# Node.js Fundamentals

Introduction to Node.js

---

## What is Node.js?

- JavaScript outside of a browser
  - Good support for latest ECMAScript features
- You can build...
  - Console applications (e.g. CLI for managing Angular apps)
  - Scalable network applications (e.g. backend for Angular apps)
- Fast and robust
  - Single-threaded (child processes are possible)
  - Non-blocking
- Well suited for web development
- Not so well suited for CPU-intensive applications

---

## Basic Console App

```
for (let i = 0; i < 10; i++) {
  console.log(`${i + 1}: Hello World!`);
}
```

- Run it with: `node app.js`
- Note template string

---

## I/O with *Console* and *Readline*

```
// Read more about `console` at https://nodejs.org/api/console.html
console.log('This is a log message');
console.info('This is an info message');
console.error('This is an error message');

// Read more abour `readline` at https://nodejs.org/api/readline.html
const readline = require('readline');
const rl = readline.createInterface(process.stdin, process.stdout);
rl.question('Please enter your name: ', answer => {
  console.log(`Hi ${answer}`);
  rl.close();
});
```

- Note how callback is used to process input from `stdin`
- Run this program with `node app.js`

---

## Excursus: Arrow Functions

```
const numbers = [1, 2, 3, 4, 5];

let newNumbers = numbers.map(function(number) {
  return number * number;
});
console.log(newNumbers.join(', '));

// Note removing of `function` keyword
newNumbers = numbers.map((number) => {
  return number * number;
});
console.log(newNumbers.join(', '));

// Note removing of `return` keyword
newNumbers = numbers.map(number => number * number);
console.log(newNumbers.join(', '));
```

---

## Basic Web API

```
const http = require('http');

const server = http.createServer((req, res) => {
  if (req.url.startsWith('/api')) {
    res.setHeader('Content-Type', 'application/json');
    res.statusCode = 200;
    res.write(JSON.stringify({foo: 'bar', answer: 42}));
    res.end();
  } else {
    res.statusCode = 404;
    res.write('Sorry, do not know what you mean...');
    res.end();
  }
});

const port = 8000;
console.log(`Listening on port ${port}...`);
server.listen(port);
```

More about core module *http*

---

## Excursus: JSON

- Lightweight data-interchange format
- Easy to read and write
- Familiar to people knowing C-family of languages
- Implemented in all major programming platforms
  - JavaScript: `JSON.parse` and `JSON.stringify` (docs)
  - Json.NET
  - JSON-java

## Basic File System Operations

```
// Use file name from command line or `greetings.txt` if no arguments were
↪  specified
const fileName = process.argv[2] || 'greeting.txt'; // Note *coalesce*
↪  operation

// Open the file for reading
const fs = require('fs');
fs.open(fileName, 'r', (err, fd) => {
  if (err) {
    console.log(`Error while opening ${fileName}: ${err.message}`);
  } else {
    // Allocate a buffer and read content of file into it.
    const buffer = new Buffer(1024);
    fs.read(fd, buffer, 0, 1024, 0, (err, bytesRead, buffer) => {
      if (err) {
        console.log(`Error: ${err.message}`);
      } else {
        console.log(`Read ${bytesRead} bytes: ${buffer.toString('utf8', 0,
        ↪  bytesRead)}`);
      }
    });
  }
});
```

More about core modules *fs* and *process*

---

## Modules (1/2)

- In Node.js, each file is a separate module
- Import modules using `require`
- Export members from a module using `exports`
- Use `__filename` to get path and file name of a module
- Use `__dirname` to get path of a module

---

## Modules (2/2)

`math.js`:

```
exports.square = x => x * x;
exports.double = x => x * 2;
exports.abs = x => {
    if (x < 0) return x * (-1);
    return x;
};
exports.demoModuleWrapper = () => {
    console.log(__filename);
    console.log(__dirname);
}
```

`app.js`:

```
const math = require('./math');
console.log(math.square(2));
math.demoModuleWrapper();
```

## Further Readings and Exercises

- Want to know more? Read/watch...
  - Node.js built-in API
  - First two lessons of *Einführung in Node.js* (German)