the Master Course

{CODENATION}

Intermediate JavaScript

Node.js and NPM Introduction

{CODENATION}

{CODENATION}

Learning Objectives

To explore what NodeJS is used for.
To be able to run your javascript files in a Node Environment.
To be familiar with modularity in Javascript.
To be able to export and require modules.
To explore what NPM is and how to use it.

Node.js

A JavaScript runtime environment

An environment which understands
JavaScript outside of the browser.

Node is a huge reason for JavaScript's surge in popularity because it lets us do things we could never do before.

JavaScript was designed to run in the browser, which limited its wider application

Node takes JavaScript server-side.

About Node

- It is fast.
- It is event driven.
- It is non-blocking (can run JavaScript asynchronously)
- It is powered by Googles V8 engine.
- It is written in C++
- It takes the JavaScript and converts it into machine code.

We can run our JavaScript files in node.
We simply use the node keyword and
the path to the JavaScript file we want
to run.

Example:
node app.js

JavaScript can add interactivity to a web page in a browser but can't do much else

With Node we can connect databases and access the computer's file system. Basically, what traditional languages like C and Java can do.

Libraries and Frameworks

There are many JavaScript libraries and frameworks on the web. They are JavaScript files that other developers have written. We can use these files in our own projects as if we had written the code ourselves.

Node Module System

Node modules are blocks of encapsulated code that communicate with an external application.

There are many core modules built into node and there are many more that we can install too. We can also make our own modules.

There are three types of modules

Local modules (created by us)
Core modules (built into node)
Third party modules (made by other people)

Local Modules

Create modules to better organise and structure your projects.

Break down your programs into smaller, manageable, JavaScript files.

```
let add = (num1, num2) => {
    let result = num1 + num2;
    return result;
};

let subtract = (num1, num2) => {
    let result = num1 - num2;
    return result;
};

module.exports = { add, subtract };
```

We can use the module.exports object to export functions, arrays, variables etc that we want to use in other files/ folders.

```
const myFunctions = require('./exp1.js');

console.log(myFunctions.add(2, 5));
console.log(myFunctions.subtract(5, 2));
```

To include a module, use the require method and store it in a variable.

```
const { add, subtract } = require('./exp1.js');

console.log(add(2, 5));
console.log(subtract(5, 2));
```

We can also require the exact functions directly using curly braces, object destructuring.

Core Modules

The built in modules are developed by the node team and are part of the language itself.

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

let userDetails = os.userInfo().username;

fs.appendFile('oh-hi.txt', `Hello ${userDetails}`, (err) => {
    if (err) {
        console.log('oops');
    }
});
```

Let's use the os and fs core modules. What does this do?

Read the Documentation

os and **fs** have different methods, see what else you can do.

https://nodejs.org/api/os.html

https://nodejs.org/api/fs.html

Third Party Modules

Third-party modules are modules that are available online using the Node Package Manager.

Node Package Manager

We can use the Node Package Manager (NPM) to install third party packages. NPM is also included in node, so we do not have to install it!

Using NPM

First we need to initialise our project. In the terminal, make sure you are in the correct folder and run the **command:**

npm init -y

```
{
  "name": "advanced-javascript-wk4",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}
```

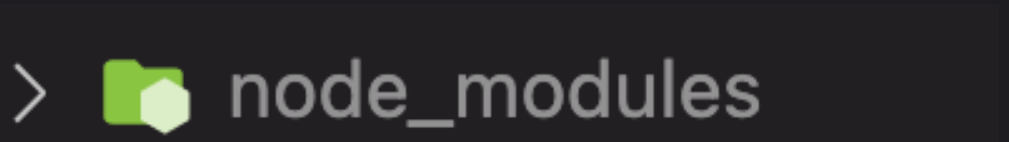
We now have a package.json file which will store our module information.

Installing a package

In the terminal run the command: `npm install inquirer`

Node Modules

Third party modules we use are referred to as dependencies. When we install them they are included in a folder called node_modules.

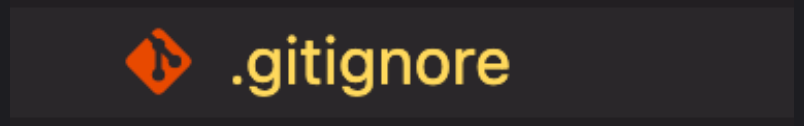


```
{
  "name": "advanced-javascript-wk4",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "inquirer": "^8.2.4"
  }
}
```

Our package.json file now includes this package as a dependency.

Git Ignore

node_modules should not be pushed to git. Create a folder named .gitignore and add the node_modules inside the file.



Package.json

Our package.json lists any dependencies that the project relies on to run. If you clone a repository from Github, remember to run npm install in the terminal to get the node_modules to run the project.

Activity

Use the Inquirer package in your Cyber Pet Challenge.
The Inquirer package is a collection of common interactive command line interfaces.

Require the package into your file.

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

As with any third party package, you **must** read the documentation.

https://www.npmjs.com/package/inquirer

{CODENATION}

Learning Objectives

To explore what NodeJS is used for.
To be able to run your javascript files in a Node Environment.
To be familiar with modularity in Javascript.
To be able to export and require modules.
To explore what NPM is and how to use it.