1. Introduction to Node.js and Command Line Interface

Web Frameworks

- After building multiple projects, the biggest thing is you realise that the same code is being written over and over again.
- Especially in **Node.js** where there are routes. The same route sends html files and sometimes you just write the thing again.
- Web Frameworks provide APIs and commands that automate web app task and make your coding time shorter.
- Web frameworks introduced many concepts that have been adopted in most frameworks today:
  - Convention of Configuration (CoC)
  - Don’t repeat yourself (DRY)
  - Using REST
  - MVC
  - **Don’t Re-invent the Wheel**

Don’t Re-invent the Wheel

In Node.js, Don’t Re-invent the Wheel is the largest convention. This is why Node.js has a package management system called NPM. This is to decrease the amount of work you would have to do normally. This also helps the Don’t Repeat Yourself convention as well.

Express.js is a package that handles your routes and middle-wear, where usually is very annoying to set up, you can accomplish this in a few lines of code.

What is Node.js?

- **Node.js** is a platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications.

- **Node.js** uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

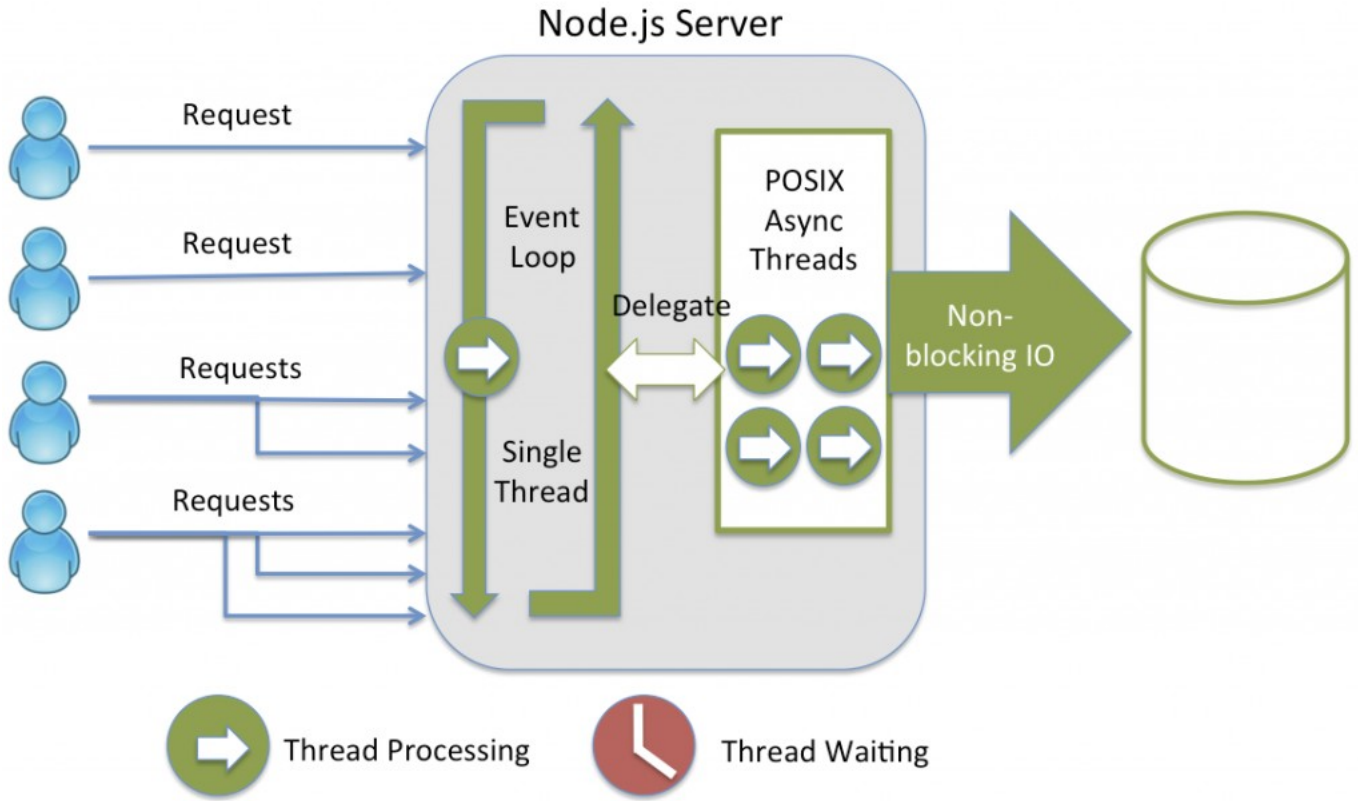
What is Node.js?

- **Node.js** is a server-side platform built on Google Chrome's JavaScript Engine (V8 Engine). Node.js was developed by Ryan Dahl in 2009

- It is an open source, cross-platform runtime environment for developing server-side and networking applications. **Node.js** applications are written in JavaScript, and can be run within the Node.js runtime on OS X, Microsoft Windows, and Linux.

Node.js System Architechture

Node.js is an event driven server side language that manages the events in a queue as an event loop. Then with the Event loop it is delegated into asynchronous tasks that manages to create a non-blocking IO (input output) to the user.



Difference to PHP/Laravel?

PHP/Laravel
Pros:

- Synchronous codebase, so easy to understand.
- Fully Object-Oriented Programming.
- Scaffolding for authentication (user login and signup can be easily create
- Easier generation of SQL queries with Eloquent ORM
- Easier management of table relationships and complex joins
- Asynchronous and easy management of tasks and jobs with the Task Sch (PHP7)
- Artisan generators for generating models, migrations, controllers and CR
- controllers

Difference to PHP/Laravel?

Nodejs
Pros:

- Overall, NodeJS offers much faster execution of scripts and code
- Asynchronous execution of code with Promises and Observables
- NodeJS scales much better for applications of a larger user base and of larger database query operations
- Thousands of node modules available to help decrease development tim
- Lightweight frameworks available such as Express or FeathersJS compar Laravel or CodeIgniter/CakePHP
- Improvements to JavaScript such as ES6 and TypeScript offer methods a functions which allow easier manipulation of arrays and objects
- Better access to new and improved DBMS (database management syste such as MongoDB

How to install Nodejs

Windows:

- Go to https://nodejs.org and press the download link and install.

Mac:

- First install Homebrew by following the instructions from here:
  - https://brew.sh/
  - In your terminal type:
    - Brew install node

Now you have Node.js on your computer!

Command line

Node.js utilizes the command line interface to run your awesome projects!

Different to other programming languages and IDE’s there is no “run” button to start your program.

To run your program type in Power Shell (Windows) or Terminal (Mac):

```
node <file_name>.js
```

Package Manager

Node comes with a dependency manager called NPM (Node Package Manager). NPM is used to install and manage project libraries, which can be separated into largely two different types:

- Global packages that can be used as development/build tools, such as Nodemon and Knex
- Or local packages that are used in each individual project.

NPM commands

Here are some example NPM command lines.
To start your node project:

```
npm init
```

To install packages globally:

```
npm install –g <package_name>
```

To install pckages locally and save into your project:

```
npm install –save <package_name>
```

To install packages in one go:

```
npm -i
```

Development environments

Node.js is a multiplatform language that can be run by writing normal JavaScript. Javascript can be written in any IDE (Intergrated Development Environment) even notepad if you wish.

However there are limitations with node.js

Atom or VS Code will be used to show code in the lectures, however if you desire to use a IDE that you like please use that. In the future when programming is required we will show you some of them.