Programming with NodeJS

Lesson 1: Setting up, the event loop and callback functions

Febuary 7th, 2017

What is NodeJS? What programming languages is it like?

What is NodeJS? What programming languages is it like?

Previous series:

What is NodeJS? What programming languages is it like?

Previous series:
https://github.com/Casper-Oakley/python-lessons

What is NodeJS? What programming languages is it like?

Previous series:
https://github.com/Casper-Oakley/python-lessons
This series:

What is NodeJS? What programming languages is it like?

Previous series:
https://github.com/Casper-Oakley/python-lessons
This series:
https://github.com/Casper-Oakley/nodejs-lessons

What is NodeJS? What programming languages is it like?

Previous series:
https://github.com/Casper-Oakley/python-lessons
This series:
https://github.com/Casper-Oakley/nodejs-lessons

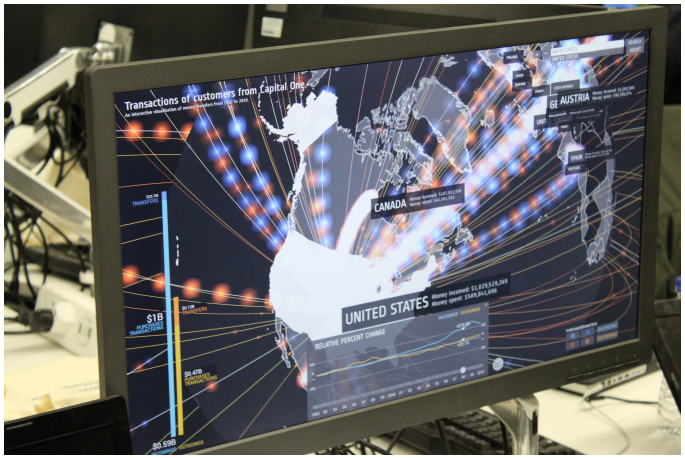

Figure: Image curtosy of HackSocNotts

Setting up your development environment

We will be using version: v6.9.5 (with npm version 3.10.10)

- Windows: **https://nodejs.org/en/download/**
- Mac OSX: **https://nodejs.org/en/download/**
- Linux: Either **sudo apt-get install nodejs** or **https://nodejs.org/en/download/**

If at any time you get stuck, just stick your hand up.

Setting up your development environment

You will also need a text editor. The choice of text editor is up to what you feel most comfortable using.

Setting up your development environment

You will also need a text editor. The choice of text editor is up to what you feel most comfortable using. Some options include:

Setting up your development environment

You will also need a text editor. The choice of text editor is up to what you feel most comfortable using. Some options include:

- notepad++

Setting up your development environment

You will also need a text editor. The choice of text editor is up to what you feel most comfortable using. Some options include:

- notepad++
- Vim

Setting up your development environment

You will also need a text editor. The choice of text editor is up to what you feel most comfortable using. Some options include:

- notepad++
- Vim
- GNU Emacs

Setting up your development environment

You will also need a text editor. The choice of text editor is up to what you feel most comfortable using. Some options include:

- notepad++
- Vim
- GNU Emacs
- atom

Setting up your development environment

You will also need a text editor. The choice of text editor is up to what you feel most comfortable using. Some options include:

- notepad++
- Vim
- GNU Emacs
- atom - Fun fact, this was actually written in part in NodeJS!

The basics of NodeJS

NodeJS is an extremely powerful language developed on top of the V8 javascript engine, written by google.

The basics of NodeJS

NodeJS is an extremely powerful language developed on top of the V8 javascript engine, written by google. It operates on a single thread, using an event loop to manage concurrency.

The basics of NodeJS

NodeJS is an extremely powerful language developed on top of the V8 javascript engine, written by google. It operates on a single thread, using an event loop to manage concurrency. It has been found to be particularly strong in certain areas:

The basics of NodeJS

NodeJS is an extremely powerful language developed on top of the V8 javascript engine, written by google. It operates on a single thread, using an event loop to manage concurrency. It has been found to be particularly strong in certain areas:

- I/O bound applications

The basics of NodeJS

NodeJS is an extremely powerful language developed on top of the V8 javascript engine, written by google. It operates on a single thread, using an event loop to manage concurrency. It has been found to be particularly strong in certain areas:

- I/O bound applications
- Data streaming applications

The basics of NodeJS

NodeJS is an extremely powerful language developed on top of the V8 javascript engine, written by google. It operates on a single thread, using an event loop to manage concurrency. It has been found to be particularly strong in certain areas:

- I/O bound applications
- Data streaming applications
- JSON APIs

The basics of NodeJS

NodeJS is an extremely powerful language developed on top of the V8 javascript engine, written by google. It operates on a single thread, using an event loop to manage concurrency. It has been found to be particularly strong in certain areas:

- I/O bound applications
- Data streaming applications
- JSON APIs
- As the backend in a MEAN stack web app

The basics of NodeJS

NodeJS is an extremely powerful language developed on top of the V8 javascript engine, written by google. It operates on a single thread, using an event loop to manage concurrency. It has been found to be particularly strong in certain areas:

- I/O bound applications
- Data streaming applications
- JSON APIs
- As the backend in a MEAN stack web app - This is the one we will focus on

Hello World!

Now that we have all prerequisites installed, load up PyCharm and start a new project.

Hello World!

Now that we have all prerequisites installed, load up PyCharm and start a new project.
We are going to build a simple program which prints the words "Hello, World!".

The *print* command prints something to the console:

Now that we have our code, we want to execute it. This can be done by hitting the green arrow in the top right corner.

Feel free to change the code inside the print to print whatever message you want!

Reading in input

A program that just prints a string is pretty rubbish. What'd be great would be if we could tell it what to print.

Reading in input

A program that just prints a string is pretty rubbish. What'd be great would be if we could tell it what to print.
The *input* command allows us to input what we want into the program. However, we need somewhere to put our input, before we can display it.

Now our program can display our name:

helloworld.py

```
print("Please input your name.")

user_input = input()

print('Hello, ' + user_input + '!')
```

Now our program can display our name:

helloworld.py

```
print("Please input your name.")

user_input = input()

print('Hello, ' + user_input + '!')
```

What have we actually done here?

Variables

Variables are a container for things. They can change when you want them to and have any name you want.

Variables

Variables are a container for things. They can change when you want them to and have any name you want.
In the previous example we specified a new variable called *name* where we stored the value we wanted to input.

Variables

Variables are a container for things. They can change when you want them to and have any name you want.
In the previous example we specified a new variable called *name* where we stored the value we wanted to input.
Variables can be many different things including:

Variables

Variables are a container for things. They can change when you want them to and have any name you want.
In the previous example we specified a new variable called *name* where we stored the value we wanted to input.
Variables can be many different things including:

- An integer (1,2,3 etc)

Variables

Variables are a container for things. They can change when you want them to and have any name you want.
In the previous example we specified a new variable called *name* where we stored the value we wanted to input.
Variables can be many different things including:

- An integer (1,2,3 etc)
- A character ('a','b','c' etc)

Variables

Variables are a container for things. They can change when you want them to and have any name you want.
In the previous example we specified a new variable called *name* where we stored the value we wanted to input.
Variables can be many different things including:

- An integer (1,2,3 etc)
- A character ('a','b','c' etc)
- A string ("Hello, World!", "I love python xo" etc)

Variables

Variables are a container for things. They can change when you want them to and have any name you want.
In the previous example we specified a new variable called *name* where we stored the value we wanted to input.
Variables can be many different things including:

- An integer (1,2,3 etc)
- A character ('a','b','c' etc)
- A string ("Hello, World!", "I love python xo" etc)
- A float (1.45345, 24.4562389, 7.4234 etc)

Variables

Variables are a container for things. They can change when you want them to and have any name you want.
In the previous example we specified a new variable called *name* where we stored the value we wanted to input.
Variables can be many different things including:

- An integer (1,2,3 etc)
- A character ('a','b','c' etc)
- A string ("Hello, World!", "I love python xo" etc)
- A float (1.45345, 24.4562389, 7.4234 etc)
- An array ([1,2,7,4,7], ['p', 'y', 't', 'h', 'o', 'n'] etc)

Variables

Variables are a container for things. They can change when you want them to and have any name you want.
In the previous example we specified a new variable called *name* where we stored the value we wanted to input.
Variables can be many different things including:

- An integer (1,2,3 etc)
- A character ('a','b','c' etc)
- A string ("Hello, World!", "I love python xo" etc)
- A float (1.45345, 24.4562389, 7.4234 etc)
- An array ([1,2,7,4,7], ['p', 'y', 't', 'h', 'o', 'n'] etc)
- Or more!

Variables

Variables are a container for things. They can change when you want them to and have any name you want.
In the previous example we specified a new variable called *name* where we stored the value we wanted to input.
Variables can be many different things including:

- An integer (1,2,3 etc)
- A character ('a','b','c' etc)
- A string ("Hello, World!", "I love python xo" etc)
- A float (1.45345, 24.4562389, 7.4234 etc)
- An array ([1,2,7,4,7], ['p', 'y', 't', 'h', 'o', 'n'] etc)
- Or more!

Some of these types can do things others can't. For example, we can subtract an integer from another, but we can't subtract a string from another.

A simple adding calculator

Using these variable types, we can convert our input into a different type! To convert a variable into a different type, we wrap the variable in the type we want to convert it to.

A simple adding calculator

Using these variable types, we can convert our input into a different type! To convert a variable into a different type, we wrap the variable in the type we want to convert it to.

```
print("Please input a number.")

input1 = int(input())
```

A simple adding calculator

Using these variable types, we can convert our input into a different type! To convert a variable into a different type, we wrap the variable in the type we want to convert it to.

```
print("Please input a number.")

input1 = int(input())
```

Here, the input we are asking for is being turned into an integer. Another name for this is we are **casting** the input to an integer.
Q: What happens if you put in something other than a number?

Our lovely calculator

Building on from that, we can combine what we have talked about to make a very simple calculator, which asks for two numbers, adds them together (using the $+$ operator), and then outputs them to the user.

Our lovely calculator

Building on from that, we can combine what we have talked about to make a very simple calculator, which asks for two numbers, adds them together (using the $+$ operator), and then outputs them to the user.

helloworld.py

```
print("Welcome to our number adder!!")
print("Please input a number.")

input1 = int(input())

print("Please input another number.")

input2 = int(input())

output = input1 + input2

print(str(input1) + " + " + str(input2) + " = " + str(output))
```

Our lovely calculator

Building on from that, we can combine what we have talked about to make a very simple calculator, which asks for two numbers, adds them together (using the $+$ operator), and then outputs them to the user.

helloworld.py

```
print("Welcome to our number adder!!")
print("Please input a number.")

input1 = int(input())

print("Please input another number.")

input2 = int(input())

output = input1 + input2

print(str(input1) + " + " + str(input2) + " = " + str(output))
```

Note we had to **cast** our integer variables to strings using *str()* for *print* to work.

That's all for tonight!

To summarise:

- We have installed Python and PyCharm
- We have learnt about variables and variable types
- We have learnt about printing and asking for input from the user
- We have made a sweet ass calculator!

For next week

Source code plus lecture slides will be available online soon after the lesson.
If you are new to HackSocNotts, please join us on *http://hacksocnotts.slack.com*.
If you have any questions, feel free to ask now or over slack.