



Coding
Bootcamps



Intro to Node.js, Express.js and MongoDB

By Kaustubh Ghadge from [Coding Bootcamps](#)

Session Outline

1. Refresher on Essential Concepts
2. What is Node.js?
3. Why does Node.js exist?
4. What is NPM?
5. Fundamentals of Node.js
6. Error Handling

Hello!

**I am Kaustubh Ghadge
("KG")**

A senior instructor at
coding-bootcamps.com



Recap

An overview of what we discussed in last session

1.

Refresher Essential Concepts



We will cover

- Asynchronous programming and callbacks
- Timers
- Promises
- Async and Await
- Closures
- The Event Loop

JavaScript Training

You should take the below course first if you are new to JS:

- [Intro to JavaScript](#)

Callbacks

- Callback is an asynchronous equivalent for a function.
- A callback function is called at the completion of a given task. Node makes heavy use of callbacks.
- All the APIs of Node are written in such a way that they support callbacks.

Asynchronous Programming

- Asynchronous programming is a means of parallel programming in which a unit of work runs separately from the main application thread and notifies the calling thread of its completion, failure or progress

Timers

- `setTimeout()`
- `setInterval()`
- `setImmediate()`

Promises

- A promise is commonly defined as a proxy for a value that will eventually become available.
- Promises are one way to deal with asynchronous code, without getting stuck in callback hell.

Async and Await

- There's a special syntax to work with promises in a more comfortable fashion, called "async/await".
- When we use **async/await**, we rarely need **.then**, because **await** handles the waiting for us. And we can use a regular **try..catch** instead of **.catch**.

Closures

A closure is a function that has access to its outer function scope even after the outer function has returned.

Event Loop

- The event loop continuously checks the call stack to see if there's any function that needs to run.
- While doing so, it adds any function call it finds to the call stack and executes each one in order.

More JavaScript Training

For more in-depth JS training, the below course is highly recommended:

- [Intermediate JS with jQuery, JSON and Ajax](#)

2.

What is Node.js?

Node.js

- Node.js is an open-source JavaScript runtime environment.
- Node.js runs the V8 JavaScript engine.
- It is asynchronous
- Event driven

3.

**Why does Node.js
exist?**

Seriously, why?

- Node.js was written initially by Ryan Dahl in 2009.
- Its development and maintenance was led by Dahl and later sponsored by Joyent.
- Apache HTTP Server had some limitations with regards to handling concurrent connections.

How does it work?

- Compared to traditional web-serving techniques where each connection (request) spawns a new thread, taking up system RAM and eventually maxing-out at the amount of RAM available, Node.js operates on a single-thread, using non-blocking I/O calls, allowing it to support tens of thousands of concurrent connections (held in the event loop).

4.

What is NPM?

npm

- **npm** is the standard package manager for Node.js.
- **npm** manages downloads of dependencies of your project.
- **npm** makes it easy for JavaScript developers to share and reuse code, and it makes it easy to update the code that you're sharing.

Managing dependency with npm

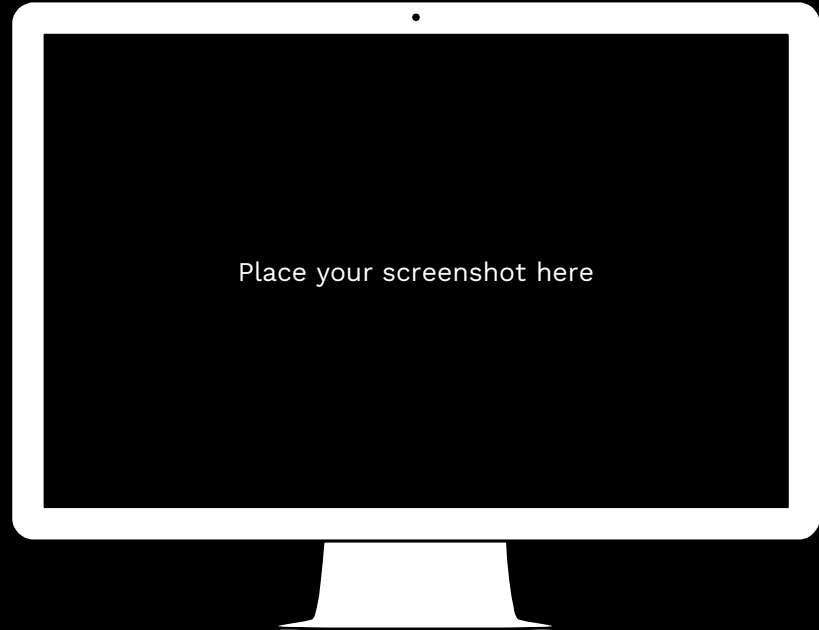
- **npm** can install packages in local or global mode.
- In local mode it installs the package in a `node_modules` folder in your parent working directory.
- Global packages are installed in `{prefix}/lib/node_modules/` which is owned by root

npm Scripts

- **npm** script is a very powerful concept - with the help of them you can build small utilities or even compose complex build systems.
- The most common ones are the start and the test scripts.

Let's Code 1

Hello World!



5.

Fundamentals of Node.js

Understanding Modules

- Dependency Management
- Modularization
- Structuring your Node project code

Sharing modules

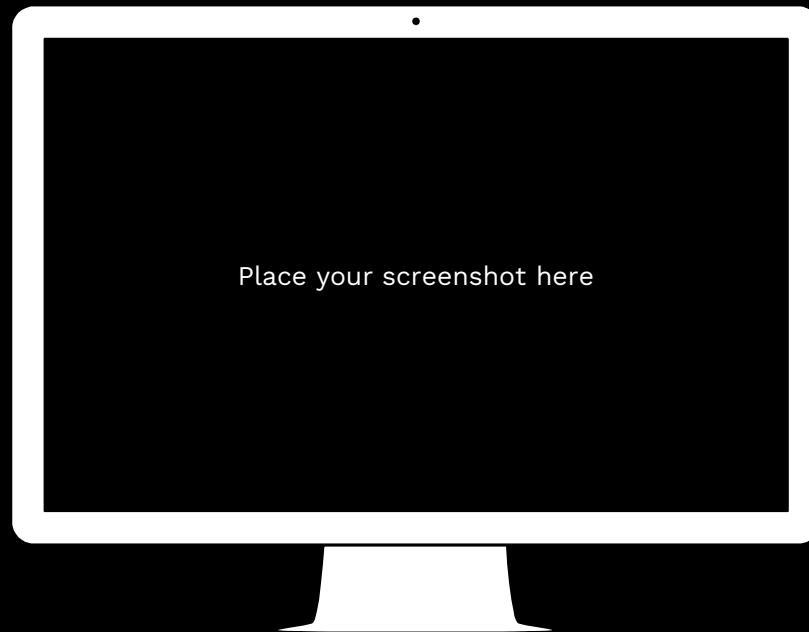
- `require`
- `exports`
- `module.exports`

fs + path Modules

- The fs module provides functionality to access and manipulate the file system.
- The path module provides a lot of very useful functionality to access and interact with the file system.
- All the methods are asynchronous by default
- Reading and writing to files

Let's Code 2

Explore fs + path
modules

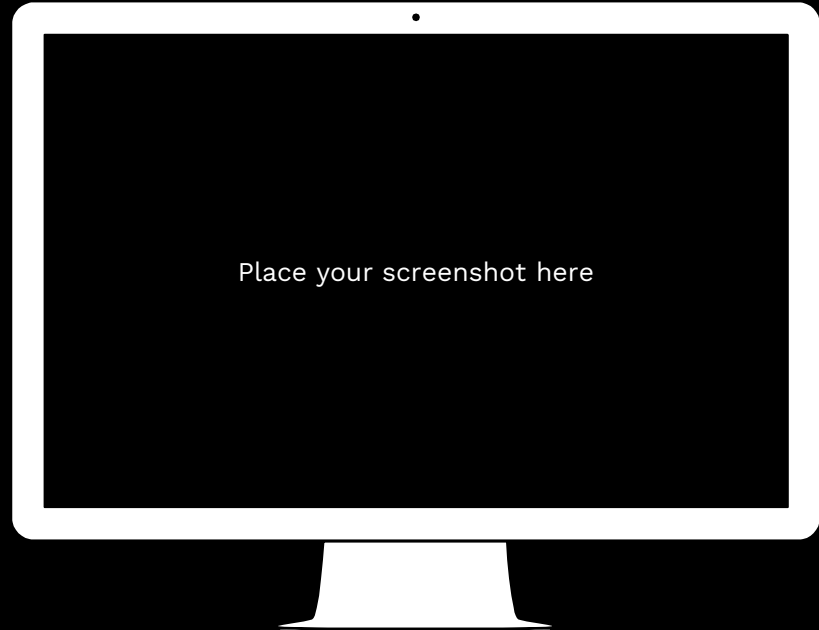


http module

- The HTTP core module is a key module to Node.js networking.
- The module provides some properties and methods, and some classes.

Let's Code 3

Creating an HTTP Server



6.

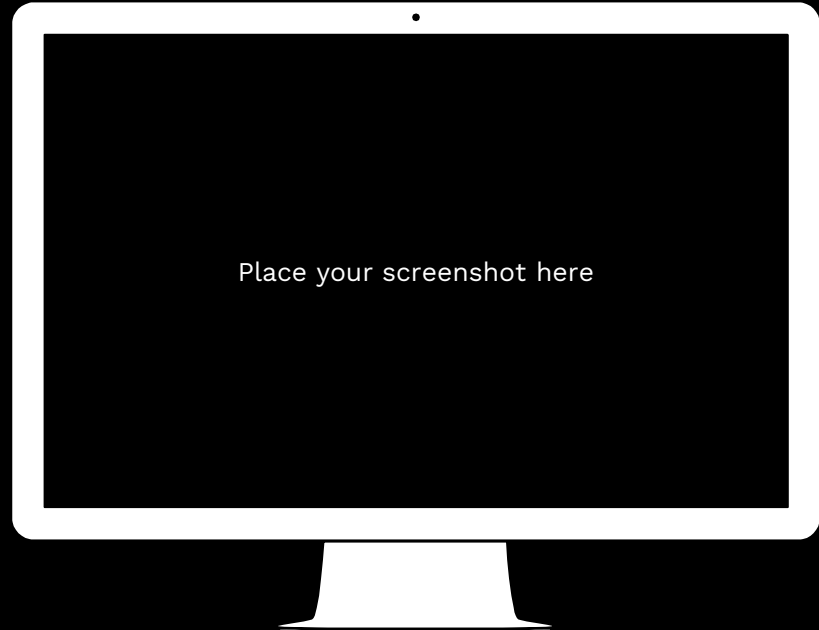
Error Handling

Error Handling in Node.js

- Creating exceptions
- Error objects
- Handling exceptions
- Catching uncaught exceptions
- Exceptions with promises
- Error handling with async/await

Let's Code 4

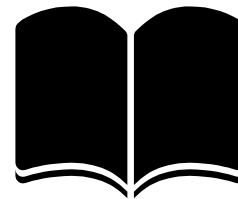
Error Handling



RECAP

Next Session

- What is Express.js?
- Features of Express.js
- Express.js installation
- Fundamental Concepts of Express.js



Resources

Reading

- [7 reasons why you should use Node.JS](#)

Case Studies

- [Node.js Helps NASA Keep Astronauts Safe and Data Accessible](#)
- [Capital One uses Node.js in its aggregation tier](#)

Live Private Coaching Sessions

- [Web design and development tutoring sessions- Weekly and monthly plans](#)
- [JavaScript, jQuery, Node.JS, MongoDB, and Express.JS- Private tutoring sessions](#)

More JavaScript Courses

- [Intermediate JS with jQuery, JSON and Ajax](#)
- [Intro to Angular.JS Framework](#)
- [Intro to React.JS Framework](#)
- [Vue.JS Framework](#)

Thanks!

Any questions?

You can find me at

- [@kaustubh_ghadge](https://twitter.com/kaustubh_ghadge)
- kaustubh.ghadge@gmail.com

coding-bootcamps.com



Coding Bootcamps

