CoGrammar

Welcome to this session:

Task Walkthrough - Node.js

The session will start shortly...

Questions? Drop them in the chat.
We'll have dedicated moderators
answering questions.

Safeguarding & Welfare

We are committed to all our students and staff feeling safe and happy; we want to make sure there is always someone you can turn to if you are worried about anything.

If you are feeling upset or unsafe, are worried about a friend, student or family member, or you feel like something isn't right, speak to our safeguarding team:

Ian Wyles
Designated Safeguarding Lead

Simone Botes

Nurhaan Snyman

Rafiq Manan

Ronald Munodawafa

Tevin Pitts

Scan to report a safeguarding concern

or email the Designated Safeguarding Lead:
Ian Wyles
safeguarding@hyperiondev.com

CoGrammar HyperionDev

Skills Bootcamp Cloud Web Development

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: **Questions**

Skills Bootcamp Cloud Web Development

- For all **non-academic questions**, please submit a query: ***www.hyperiondev.com/support***
- **Report a safeguarding incident: *www.hyperiondev.com/safeguardreporting***
- We would love your feedback on lectures: ***Feedback on Lectures***
- If you are hearing impaired, please kindly use your computer's function through Google chrome to enable captions.

Learning Outcomes

- **Initialize and set up an NPM package** by installing dependencies and creating custom scripts.
- **Utilise JavaScript packages**, such as Moment.js or Lodash, to enhance application functionality.
- **Demonstrate the use of package functions** to manipulate data effectively within a Node.js environment.
- **Create and execute custom NPM scripts** to automate tasks and streamline project workflows.

Lecture Overview

- → Presentation of the Task
- → Introduction to Node.js
- → Introduction to NPM
- → Task Walkthrough

Lodash Task

Imagine you're building a tool to organise your music library. 🎶 In this task, you'll create a package that uses Lodash to remove duplicate song entries from an array and adds a feature to count the total unique songs. This project will expand your skills in using NPM, Lodash, and script creation to solve real-life problems in a dynamic, engaging way.

You'll be using Lodash to clean up a playlist by filtering out repeated songs, so you're left with a list of unique tracks. Additionally, you'll count and display the total number of unique songs to see just how much you've slimmed down your playlist!

Moments Task

Imagine you’re building a Countdown Timer to track upcoming events, such as birthdays, holidays, or special occasions. 🎉 In this task, you’ll create a package that uses Moment.js to calculate the remaining days until each event. Your timer will display a list of events with their names and the countdown to each event, all in a user-friendly format.

By the end of this task, you’ll be comfortable with managing dates in JavaScript, calculating differences, and presenting time-sensitive information in a clear and engaging way!

What is NPM primarily used for in JavaScript?

A. Styling HTML pages
B. Package and dependency management
C. Running JavaScript in the browser
D. Encrypting data

CoGrammar

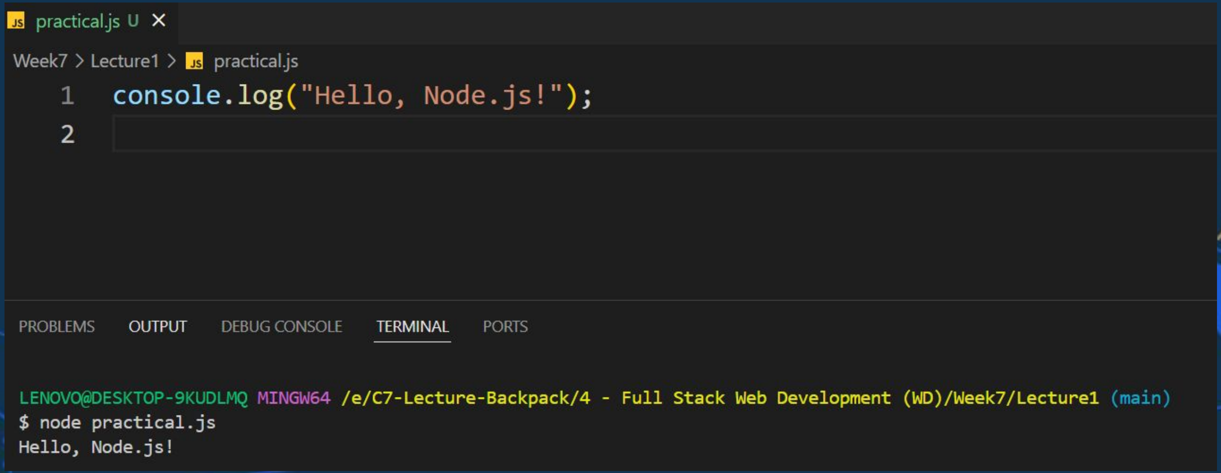
What does the Lodash uniq function do?

A. Filters an array based on a callback
B. Sorts an array in ascending order
C. Removes duplicates from an array
D. Maps each element to a new value

CoGrammar

What is Node.js?

- Node.js is a runtime environment that allows you to run JavaScript code on the server-side.
- It uses an event-driven, non-blocking I/O model, making it efficient for handling asynchronous operations.

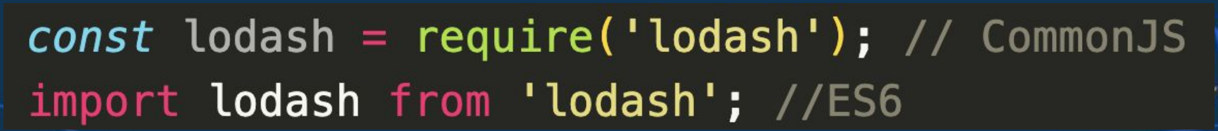


```javascript
console.log("Hello, Node.js!");
```

```
LENOVO@DESKTOP-9KUDLMQ MINGW64 /e/C7-Lecture-Backpack/4 - Full Stack Web Development (WD)/Week7/Lecture1 (main)
$ node practical.js
Hello, Node.js!
```

What are Modules?

- Modules in Node.js are encapsulated units of functionality that can be reused throughout your application.
- They promote code organization, maintainability, and reusability.
- Node.js provides several core modules like http, fs, and path, which can be used without installation.
- User-defined modules are created by developers to encapsulate specific functionality.



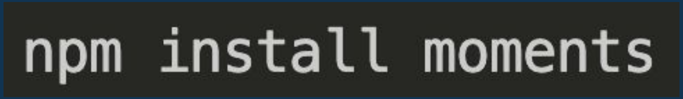
```
const lodash = require('lodash'); // CommonJS
import lodash from 'lodash'; //ES6
```

Node Package Manager

- NPM is the default package manager for Node.js, used for installing, managing, and sharing packages of JavaScript code.
- It provides access to a vast repository of open-source packages and tools for Node.js development.

Managing Dependencies with NPM

- Use npm init to initialize a new NPM package in your project directory and generate a package.json file interactively or with default values.
- package.json serves as the manifest for your project, documenting project metadata, dependencies, and scripts.
- Define project dependencies in the package.json file.
- Use npm install to install dependencies listed in package.json.

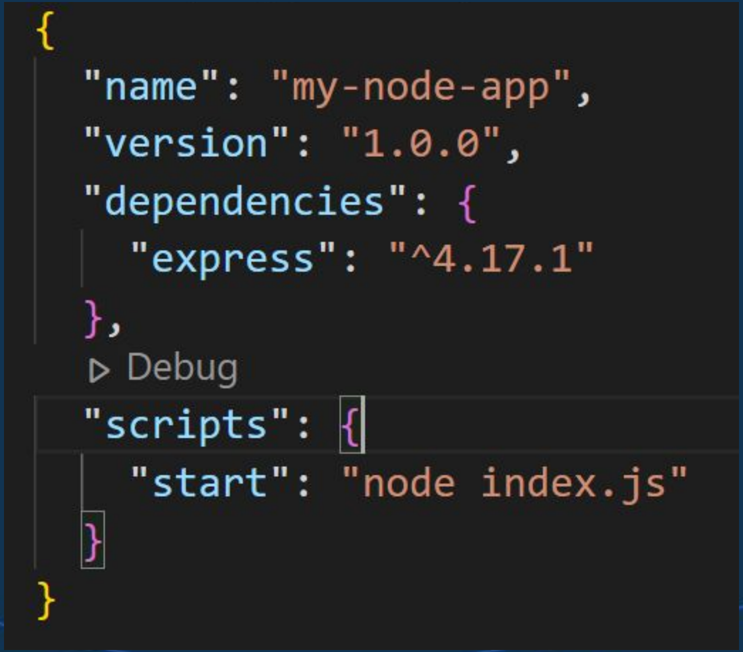


```
npm install moments
```

Understanding package.json Structure

- **name:** The name of the project.
- **version:** The version of the project.
- **dependencies:** List of project dependencies and their version specifications.
- **scripts:** Custom scripts for tasks like testing, building, and deployment.

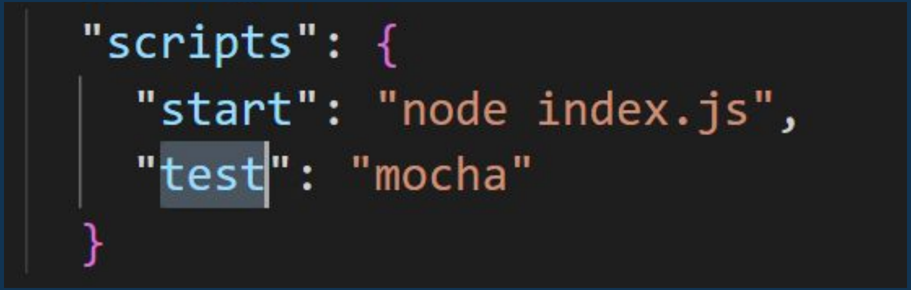
Understanding package.json Structure



```json
{
  "name": "my-node-app",
  "version": "1.0.0",
  "dependencies": {
    "express": "^4.17.1"
  },
  "scripts": {
    "start": "node index.js"
  }
}
```

Managing Scripts in package.json

- Use the **scripts** field in **package.json** to define custom scripts.
- Scripts can be executed using **npm run <script-name>**.



```
"scripts": {
  "start": "node index.js",
  "test": "mocha"
}
```

Lodash Task

Imagine you’re building a tool to organise your music library. 🎶 In this task, you’ll create a package that uses Lodash to remove duplicate song entries from an array and adds a feature to count the total unique songs. This project will expand your skills in using NPM, Lodash, and script creation to solve real-life problems in a dynamic, engaging way.

You’ll be using Lodash to clean up a playlist by filtering out repeated songs, so you’re left with a list of unique tracks. Additionally, you’ll count and display the total number of unique songs to see just how much you’ve slimmed down your playlist!

Moments Task

Imagine you’re building a Countdown Timer to track upcoming events, such as birthdays, holidays, or special occasions. 🎉 In this task, you’ll create a package that uses Moment.js to calculate the remaining days until each event. Your timer will display a list of events with their names and the countdown to each event, all in a user-friendly format.

By the end of this task, you’ll be comfortable with managing dates in JavaScript, calculating differences, and presenting time-sensitive information in a clear and engaging way!

Which command initialises a new package in NPM?

A. npm install
B. npm start
C. npm init
D. npm run

CoGrammar

What is the purpose of the package.json file in a Node.js project?

A. To store user data and settings
B. To manage project dependencies and scripts
C. To style HTML elements
D. To create an executable binary

CoGrammar

Q & A SECTION

Please use this time to ask any questions relating to the topic, should you have any.

Thank you for attending

CoGrammar

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education