




Welcome to the **Co**Grammar Routing and Middleware in Express

The session will start shortly...

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



Full Stack Web Development Session Housekeeping

- 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.
- 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](#)

Full Stack Web Development Session Housekeeping cont.

- 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](#)
- You can turn on live captions in your browser's settings for clear translation on what's being lectured

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



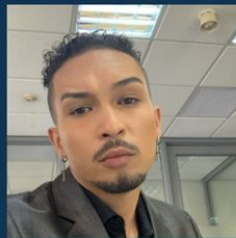
Nurhaan Snyman



Rafiq Manan



Ronald Munodawafa



Tevin Pitts

Scan to report a
safeguarding concern



or email the Designated
Safeguarding Lead:
Ian Wyles

safeguarding@hyperiondev.com

Polls

Please have a look at the poll notification and select an option.

Which of the following is NOT a valid HTTP method in Express.js routing?

- A. GET
- B. POST
- C. FETCH
- D. DELETE

Lesson Objectives

❖ By the end of this topic, students will:

- Remember the core concepts of routing and middleware
- Apply routing and middleware in basic express
- Analyze the request-response cycle and middleware execution
- Evaluate the effectiveness of routing and middleware in real-world scenarios

**SKILLS
FOR LIFE**

SKILLS BOOTCAMPS



Department
for Education

CoGrammar

Routing and Middleware

January 2025

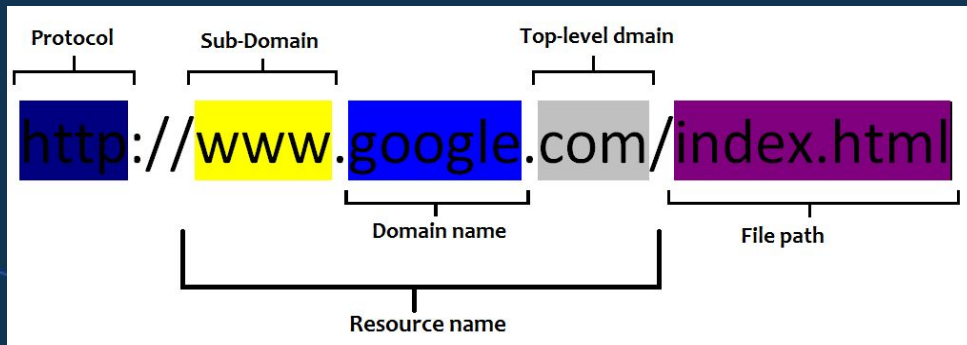
Introduction to Routing



Routes

Determining how an application responds to a client's request to a particular endpoint.

- ❖ Since we have the server configured, we need a routing mechanism to react to users' requests.
- ❖ Having a brief of routes and URLs will assist. Below is an example of a full URL.



Routes

URL Parts

- ❖ **Protocol (Scheme):** Specifies the protocol or method used to access the resources such as **HTTP, HTTPS, FTP** e.t.c
- ❖ **Subdomain:** A prefix to the main domain name indication a specific department within the domain. **www**
- ❖ **Domain:** The main part of the URL that identifies the website or web service.
- ❖ **Top Level Domain:** Used to categorize the internet domain space into different groups based on their purpose of location.

Routes

URL Parts

- ❖ **Port:** Identifies a specific endpoint within a server separated from the domain by a colon. example.com:**8080**
- ❖ **Path:** specifies the location of a specific resource or file within the domain directory structure. E.g example.com/**path/file**
- ❖ We'll be more interested in the path section of the URL by identifying the path that was requested and providing a response based on that path.

Creating a route for your application

- ❖ In Express.js (or any backend frameworks) there are routing methods that specify the type of requests.
- ❖ Common routing methods we'll use in express.js:
 - **GET:** Retrieves data from server
 - **POST:** Submit data to be processed in the server
 - **PUT:** Updates or replaces data existing in the server with submitted data.
 - **DELETE:** Delete a specified resource from the server.

Creating a route for your application

- ❖ We'll create our first path with the GET method.
- ❖ From the app variable, we can call the `app.get()` which takes in two main arguments. (**The path** and **a callback function**).
- ❖ The callback function in this case becomes the route handler, it determined the kind of response the user will get after making a request to a specific path on the server.



Router Parameters

- ❖ Router Parameters are named URL segments used to capture values at specific positions in the URL.



Routes example

What is the difference between a static route and a dynamic route from React?

index.js

```
5 app.get('/', (req, res) => {  
6   res.send("Hello World")  
7 })  
8  
9 app.get('/users/:userId', (req, res) => {  
10   res.send(`User Id: ${req.params.userId}`)  
11 })
```

Snipped

MiddleWare

- ❖ Middleware functions are functions that have access to the request object (req), the response object (res), and the next middleware function in the application's request-response cycle.
- ❖ Key concepts:
 - Next Function: Passes control to the next middleware function
 - Order Matters: Middleware functions are executed in the order they are added.

Types of middleware

- ❖ **Application-level Middleware:**
 - Bound to the app object using `app.use()`
- ❖ **Router level middleware**
 - Bound to an instance of `express.Router()`
- ❖ **Error-handling middleware:**
 - Defined with four arguments (`err`, `req`, `res`, `next`)
- ❖ **Built-in Middleware:**
 - `express.json()`, `express.urlencoded()`, `express.static()`
- ❖ **Third-party middleware:**
 - `Body-parser`, `cors` e.t.c

Middleware example

Basic middleware structure.

index.js

```
5  app.use((req, res, next)=>{
6    console.log(`${req.method} ${req.url}`)
7    next()
8  })
9
10 app.use((err, req, res, next)=>{
11   console.error(err)
12   res.status(500).send('Something went wrong')
13 })
14
```

Snipped

Polls

Please have a look at the poll notification and select an option.

What is the purpose of the `next()` function in Express.js middleware?

- A. To terminate the request-response cycle
- B. To pass control to the next middleware function in the stack
- C. To send a response back to the client
- D. To log errors in the application

Polls

Please have a look at the poll notification and select an option.

What is the correct order of middleware execution in Express.js?

- A. Error-handling middleware -> Application-level middleware -> Router-level middleware
- B. Application-level middleware -> Router-level middleware -> Error-handling middleware
- C. Router-level middleware -> Application-level middleware -> Error-handling middleware
- D. Error-handling middleware -> Router-level middleware -> Application-level middleware

Questions and Answers



Thank you for attending



Department
for Education

CoGrammar

