

ONE PAGE SUMMARY

IT & Software > Backend Development > Express.js

Introduction to Express.js

Master Express.js for efficient backend development!

DEFINITION

Express.js is a minimal and flexible Node.js web application framework that simplifies server-side development. It provides a robust set of features for building web and mobile applications.

KEY CONCEPTS

Middleware

Middleware functions are functions that execute during the request-response cycle, allowing you to modify requests and responses.

Routing

Routing refers to defining the endpoints of your application and how they respond to client requests.

Request and Response

Express.js handles HTTP requests and responses, allowing you to send data back to clients easily.

Static Files

You can serve static files like images, CSS, and JavaScript using Express, making it easy to deliver content.

Error Handling

Express provides a way to handle errors gracefully, ensuring users receive appropriate feedback when issues arise.

EXAMPLES

```
→ app.get('/api/users', (req, res) => {
  res.send(users); })
→ app.use(express.static('public'))
→ app.post('/api/login', (req, res) => { /*
  login logic */ })
```

MEMORY TIPS

- ★ Remember 'M' for Middleware: Think of it as the 'middle' layer in requests.
- ★ Use 'R' for Routing: Routes direct traffic like roads.
- ★ Think 'E' for Error Handling: Errors are part of the journey, handle them well.

COMMON MISTAKES

- ✗ Forgetting to use 'next()' in middleware functions.
- ✗ Not defining routes before using them in the app.
- ✗ Neglecting to handle errors properly, leading to crashes.

QUICK RECAP

Express.js is essential for building web applications with Node.js. Focus on middleware, routing, and error handling to create robust applications.