**Express.js static Middleware**

The express.static() middleware is a built-in middleware function in Express.js used to serve **static files** like **HTML, CSS, JavaScript, images, videos, and other assets**.

**1. Basic Usage**

To serve static files from a folder (e.g., public), use:

const express = require('express');

const app = express();

// Serve static files from "public" directory

app.use(express.static('public'));

app.listen(3000, () => console.log('Server running on port 3000'));

📌 **Folder Structure Example:**

/public

├── index.html

├── styles.css

├── script.js

├── images/

├── logo.png

📌 **Accessing Static Files:**

* http://localhost:3000/index.html
* http://localhost:3000/styles.css
* http://localhost:3000/images/logo.png

**2. Custom Static File Path (Virtual Path)**

You can add a **virtual path prefix** so that files aren’t served directly from the root.

app.use('/static', express.static('public'));

📌 **Now, access files like this:**

* http://localhost:3000/static/index.html
* http://localhost:3000/static/styles.css

**3. Serve Multiple Static Directories**

You can serve multiple static directories using multiple express.static() calls.

app.use(express.static('public'));

app.use(express.static('assets'));

📌 If both public/logo.png and assets/logo.png exist, Express will **serve the first match**.

**4. Restricting File Access (Custom Middleware)**

By default, express.static() serves all files in the directory. If you want to **restrict** access to certain files, you can use middleware before serving static files.

app.use((req, res, next) => {

if (req.url.includes('.private')) {

return res.status(403).send('Access Forbidden');

}

next();

});

app.use(express.static('public'));

📌 This prevents files like secret.private from being accessed.

**5. Cache Control (Disable Caching)**

By default, express.static() enables caching for performance. To disable caching, pass an options object:

app.use(express.static('public', { cacheControl: false, etag: false }));

📌 This prevents browsers from caching static files.

**6. Setting a Custom Cache Expiry Time**

You can set a custom maxAge (in milliseconds) for caching.

app.use(express.static('public', { maxAge: 86400000 })); // 1 day caching

📌 This tells browsers to cache static files for **1 day**.

**7. Handling 404 for Static Files**

If a file is missing, Express does **not** return a custom 404 by default. You can define a **catch-all** route.

app.use(express.static('public'));

app.use((req, res) => {

res.status(404).send('File Not Found');

});

**Summary**

| **Feature** | **Example** |
| --- | --- |
| Serve static files | app.use(express.static('public')) |
| Virtual static path | app.use('/static', express.static('public')) |
| Multiple directories | app.use(express.static('public')); app.use(express.static('assets')) |
| Restrict access | Custom middleware to block certain files |
| Disable caching | app.use(express.static('public', { cacheControl: false })) |
| Custom cache expiry | app.use(express.static('public', { maxAge: 86400000 })) |
| Custom 404 page | Handle missing static files |

Would you like me to expand on anything? 🚀