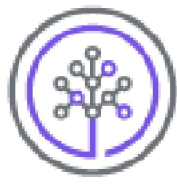


Developing Back-End Apps with Node.js and Express

Module 1 Cheat Sheet: Introduction to Server-Side JavaScript

| Package/Method | Description | Code Example |
|--------------------------|--|--|
| http.createServer | <p>http package is used for establishing remote connections to a server or to create a server which listens to client.</p> <p>createServer - Takes a requestListener, a function which takes request and response parameters, where request is the handle to the request from the client and response is the handle to be sent to the client.</p> | <pre>const http = require('http'); const requestListener = function(req, res) { res.writeHead(200); res.end('Hello, World!'); } const port = 8080; const server = http.createServer(requestListener); console.log('server listening on port: ' + port); server.listen(port);</pre> |
| new Date() | <p>new Date() method returns the current date as an object. You can invoke methods on the date object to format it or change the timezone.</p> | <pre>module.exports.getDate = function getDate() { let aestTime = new Date().toLocaleString("en-US", {timeZone: "Australia/Brisbane"}); return aestTime; }</pre> |
| import() | <p>The import statement is used to import modules that some other module has exported. A file that includes reusable code is known as a module.</p> | <pre>// addTwoNos.mjs function addTwo(num) { return num + 4; } export { addTwo }; // app.js import { addTwo } from './addTwoNos.mjs'; // Prints: 8 console.log(addTwo(4));</pre> |
| require() | <p>The built-in NodeJS method require() is used to incorporate external modules that are included in different files. The require() statement essentially reads and executes a JavaScript file before returning the export object.</p> | <pre>module.exports = 'Hello Programmers'; let msg = require('./messages.js'); console.log(msg);</pre> |



Skills Network