

Developing REST APIs with Node.js

VY102 (Web-based)

Course description

This course teaches developers how to create, test, and deploy applications with Node.js.The Node.js runtime environment is a highly scalable server-side application platform. In this course, you learn how to develop REST applications with Express, a popular web application framework for Node. You design callback functions to handle asynchronous network events. You also install and manage Node features with npm, the packaging manager for Node modules.You build, test, and deploy the lab exercises in your own workstation. You optionally deploy the application on the IBM Cloud with your own IBM Cloud account.

For information about other related courses, see the IBM Training website:

**ibm.com**/training

General information

Delivery method

Web-based

Course level

ERC 6.0

Product and version

Node.js LTS release, Version 10.15.3 or newer

Audience

This course is designed for API developers who want to build REST applications with the Node.js server runtime environment.

Learning objectives

After completing this course, you should be able to:

* Install, validate, and test the Node runtime environment on your local workstation
* Install Node packages with npm
* Develop REST API operations with Express
* Develop callback functions to handle asynchronous events
* Perform static code analysis of the application with ESLint
* Run Mocha and Supertest unit tests on Node applications
* Debug Node applications with the Google Chrome browser with Node inspector
* Package Node applications
* Deploy Node applications to IBM Cloud with the IBM Cloud command-line utility
* Run Node applications on IBM Cloud

Prerequisites

* Working knowledge of JavaScript programming
* Familiarity with web application architecture and REST API concepts

Duration

2 days

Skill level

Intermediate

|  |  |
| --- | --- |
| Classroom (ILT) setup requirements | |
| Processor | Intel Pentium 2.26 GHz or faster |
| GB RAM | 4.0 |
| GB free disk space | 20 |
| Network requirements | LAN / Internet / DHCP |
| Other requirements | This course runs as a BYOD/BYOE (bring your own device/environment) course. It can be run on any of the Windows, Linux, or MacOS environments. Processor requirements are different for each operating environment. The hardware requirements shown here are for the Windows operating system. |

Notes

The following unit and exercise durations are estimates, and might not reflect every class experience. If the course is customized or abbreviated, the duration of unchanged units will probably increase.

This course is an update of the following previous course:

* VY102 ERC 5.0: *Developing REST APIs with Node.js for IBM Bluemix*

Course agenda

|  |
| --- |
| Course introduction  Duration: 15 minutes |

|  |  |
| --- | --- |
| Unit 1. Introduction to Node.js  Duration: 45 minutes | |
| Overview | This unit explains the motivation and purpose of Node.js, an event-driven JavaScript web application server framework, and runtime. It explores how to easily write web applications in JavaScript, and describes how to create a simple web server in Node.js on your workstation. |
| Learning objectives | After completing this unit, you should be able to:   * Explain the origin and purpose of the Node.js JavaScript framework * Write a simple web server with JavaScript * Import Node.js modules into your script |

|  |  |
| --- | --- |
| Exercise 1. Installing, verifying, and developing a Node application  Duration: 45 minutes | |
| Overview | In this exercise, you set up the Node.js runtime environment on your own workstation. With the Node.js runtime, you can develop and test Node applications in a local environment. The Node.js runtime environment is a prerequisite to creating APIs with the LoopBack framework. |
| Learning objectives | After completing this exercise, you should be able to:   * Install the Node.js runtime environment on a local workstation * Verify the setup of the Node.js runtime environment * Verify the setup of the Node package manager, npm * Update the Node package manager on your workstation * Define a package manifest file * Install a third-party package in a Node application * Start the Node interactive shell * Run a Node application |

|  |  |
| --- | --- |
| Unit 2. Developing a REST API in Node  Duration: 45 minutes | |
| Overview | The Node.js framework relies on callback functions to handle network calls in an asynchronous manner. In this unit, you learn how to write anonymous callback functions to act on network events. You learn how to listen and intercept network traffic, and parse network traffic with socket programming. |
| Learning objectives | After completing this unit, you should be able to:   * Define a package dependency * Explain the features of the Express Node framework * Handle HTTP method calls with the Express framework * Call remote services with the Request package * Create a callback function to handle responses from remote services * Parse XML data with the xml2js package |

|  |  |
| --- | --- |
| Exercise 2. Developing a REST API with Node.js  Duration: 1 hour | |
| Overview | In this exercise, you develop a REST API as a Node application. You build a web application with the Express framework that handles HTTP method requests on web resources. In the implementation of your Node web application, you call remote services with the Request package. You also develop a callback function to handle the response and error message from remote services. |
| Learning objectives | After completing this exercise, you should be able to:   * Install the Express node package * Define an Express web application * Handle requests to web resources with Express * Call remote services with the Request package * Create a callback function to handle responses from remote calls * Handle errors with a callback return parameter * Test a callback function in a Node application |

|  |  |
| --- | --- |
| Unit 3. Static code analysis and unit testing  Duration: 45 minutes | |
| Overview | This unit describes how to validate JavaScript code with static code analysis tools. The unit also describes some of the testing tools that are used to test Node applications. |
| Learning objectives | After completing this unit, you should be able to:   * Describe how to validate JavaScript code with ESLint * Describe how to unit test your application with Mocha * Explain how to create HTTP unit tests with Supertest |

|  |  |
| --- | --- |
| Exercise 3. Static code analysis and unit testing  Duration: 1 hour and 30 minutes | |
| Overview | In this exercise, you validate and test your Node application REST API implementation. You validate the application source code with the ESLint package, and develop and run a suite of unit test cases on the Node application functions with Mocha and Supertest. |
| Learning objectives | After completing this exercise, you should be able to:   * Explain the purpose of static code analysis * Explain the purpose of unit testing * Perform static code analysis with ESLint * Create and run a function test suite in Mocha * Create and run a web application test suite in Supertest |

|  |  |
| --- | --- |
| Unit 4. Debugging and building Node applications  Duration: 45 minutes | |
| Overview | This unit describes some of the utilities and techniques that are used to debug and package Node applications. |
| Learning objectives | After completing this unit, you should be able to:   * Describe the Read, Evaluate, Print, and Loop (REPL) capability of Node * Describe the use of the console to log diagnostic information * Describe the standard command-line debug feature of Node * Examine the Node Inspector graphical debug tool * Describe the purpose of the npm shrinkwrap utility * Describe the use of npm as a build tool |

|  |  |
| --- | --- |
| Exercise 4. Debugging and building Node applications  Duration: 1 hour | |
| Overview | In this exercise, you work with various Node.js debug utilities on your own workstation. You work with the command-line and graphical debug utilities. You also create scripts in the package.json file that npm uses to build and run node applications. |
| Learning objectives | After completing this exercise, you should be able to:   * Use the standard node debug utility of Node * Enable Node Inspector * Work with the Node Inspector graphical debug tool * Use package lock to set node module versions * Use script objects and npm to build node applications |

|  |  |
| --- | --- |
| Unit 5. Deploying and testing Node applications on IBM Cloud  Duration: 1 hour | |
| Overview | This unit describes the architecture of IBM Cloud. Describe the features of the IBM Cloud Development Tools that are used to enable and deploy cloud applications. Explain how to deploy a Node.js application to IBM Cloud. |
| Learning objectives | After completing this unit, you should be able to:   * Describe the architecture of IBM Cloud * Describe the features of the IBM Cloud Developer Tools * Describe what is Cloud Foundry * Describe how to install the IBM Cloud Developer Tools command-line interface (CLI) * Explain how to deploy a Node application to IBM Cloud * Describe how to view and manage the resources in your cloud account from the IBM Cloud dashboard |

|  |  |
| --- | --- |
| Exercise 5. Deploying a REST API on IBM Cloud (optional)  Duration: 45 minutes | |
| Overview | In this exercise, you deploy a packaged REST API into your IBM Cloud account. You learn how to install the IBM Cloud Developer Tools command-line interface (CLI) utility, and connect and manage your IBM Cloud account with the Cloud CLI. You also learn how to deploy your REST API as an IBM Cloud Node.js application. |
| Learning objectives | After completing this exercise, you should be able to:   * Register for an IBM Cloud account * Install the IBM Cloud command-line interface (CLI) * Enable an existing application for IBM Cloud * Build and run an application on a local container that is built with the IBM Cloud Developer Tools * Deploy a node application into IBM Cloud * Retrieve the application logs from an IBM Cloud application |

|  |  |
| --- | --- |
| Unit 6. Course summary  Duration: 10 minutes | |
| Overview | This unit summarizes the course and provides information for future study. |
| Learning objectives | After completing this unit, you should be able to:   * Explain how the course met its learning objectives * Access the IBM Training website * Identify other IBM Training courses that are related to this topic * Locate appropriate resources for further study |

For more information

To learn more about this course and other related offerings, and to schedule training, see **ibm.com**/training

To learn more about validating your technical skills with IBM certification, see **ibm.com**/certify

To stay informed about IBM training, see the following sites:

IBM Training News: http://bit.ly/IBMTrainEN

YouTube: youtube.com/IBMTraining

Facebook: facebook.com/ibmtraining

Twitter: twitter.com/websphere\_edu