



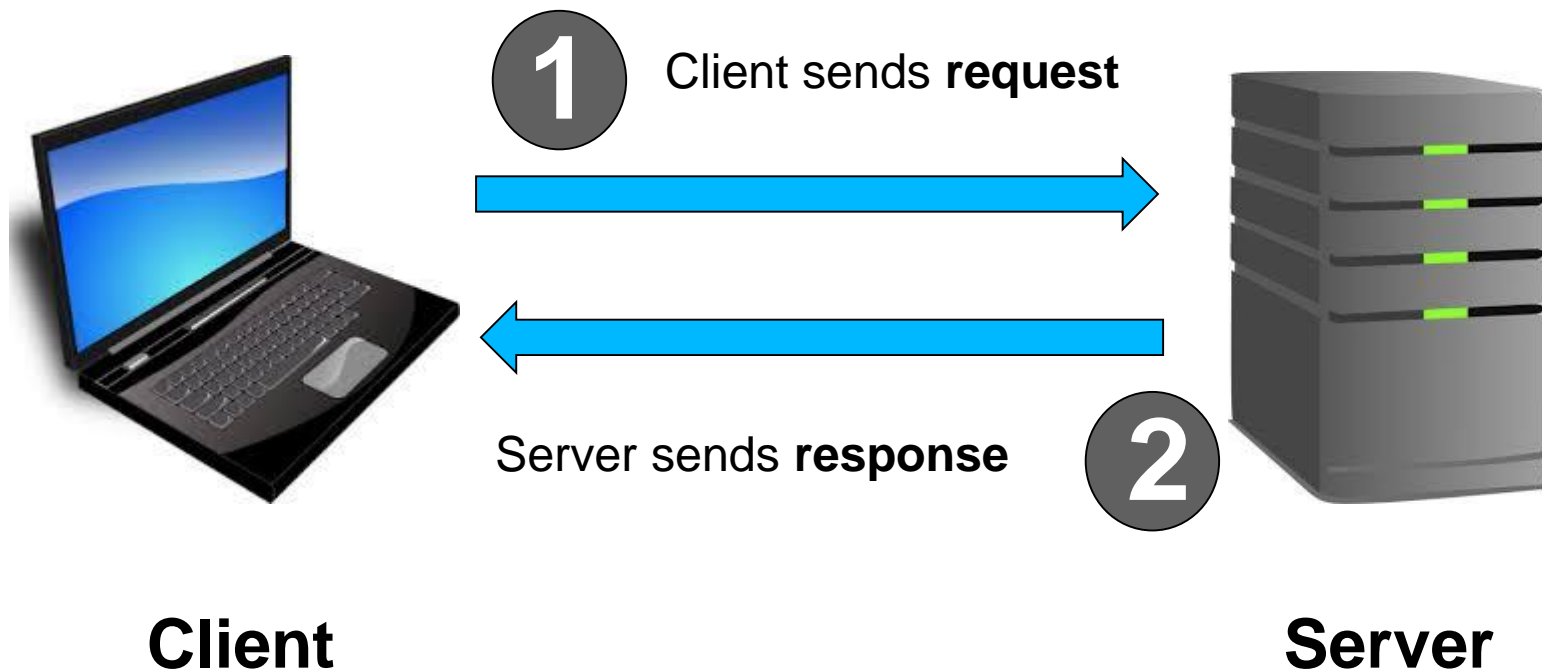
Video 4.1

Intro to Node.js Environment Setup

Chris Murphy

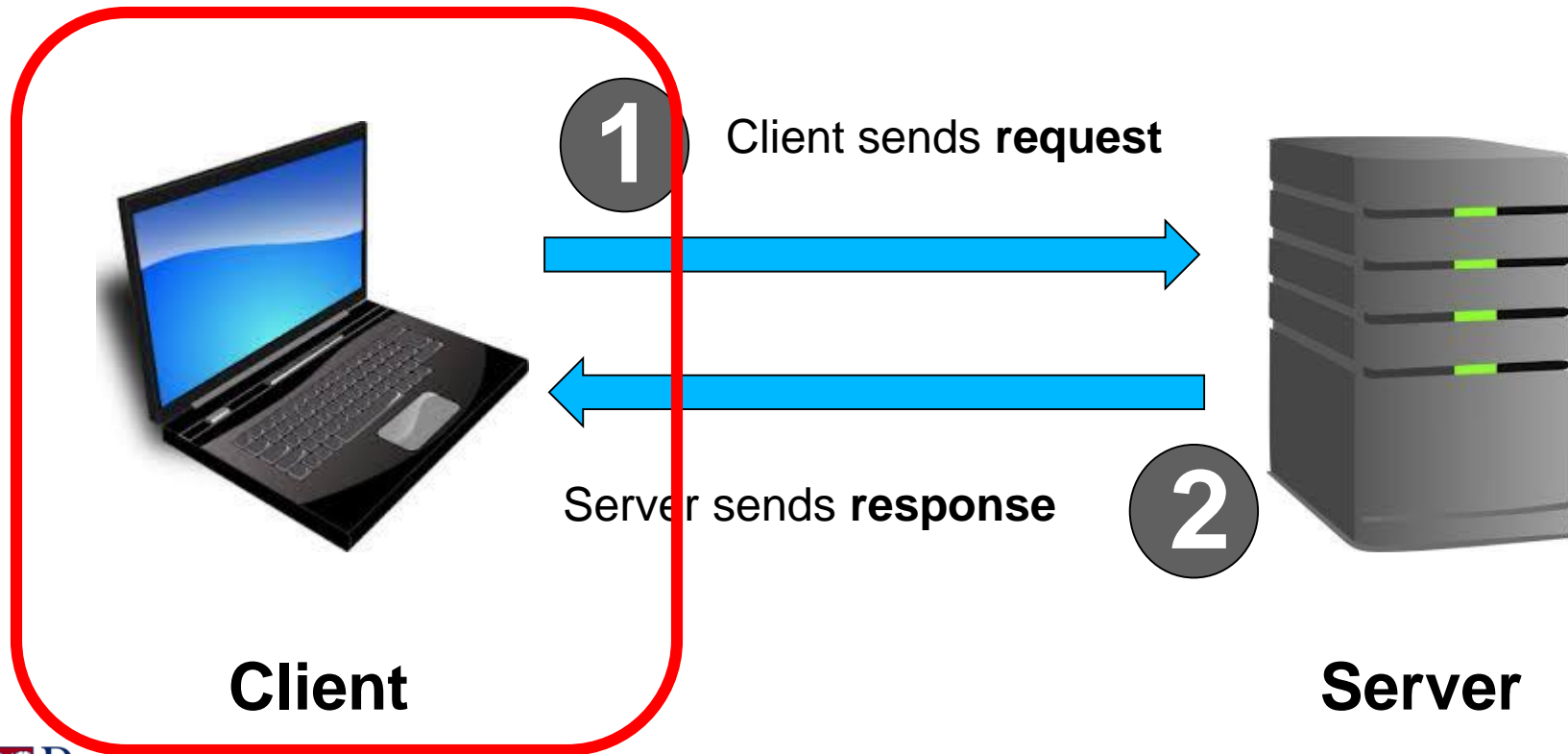
Review: How does a Web Browser Work?

- The World Wide Web utilizes **Hypertext Transfer Protocol (HTTP)** to transfer documents



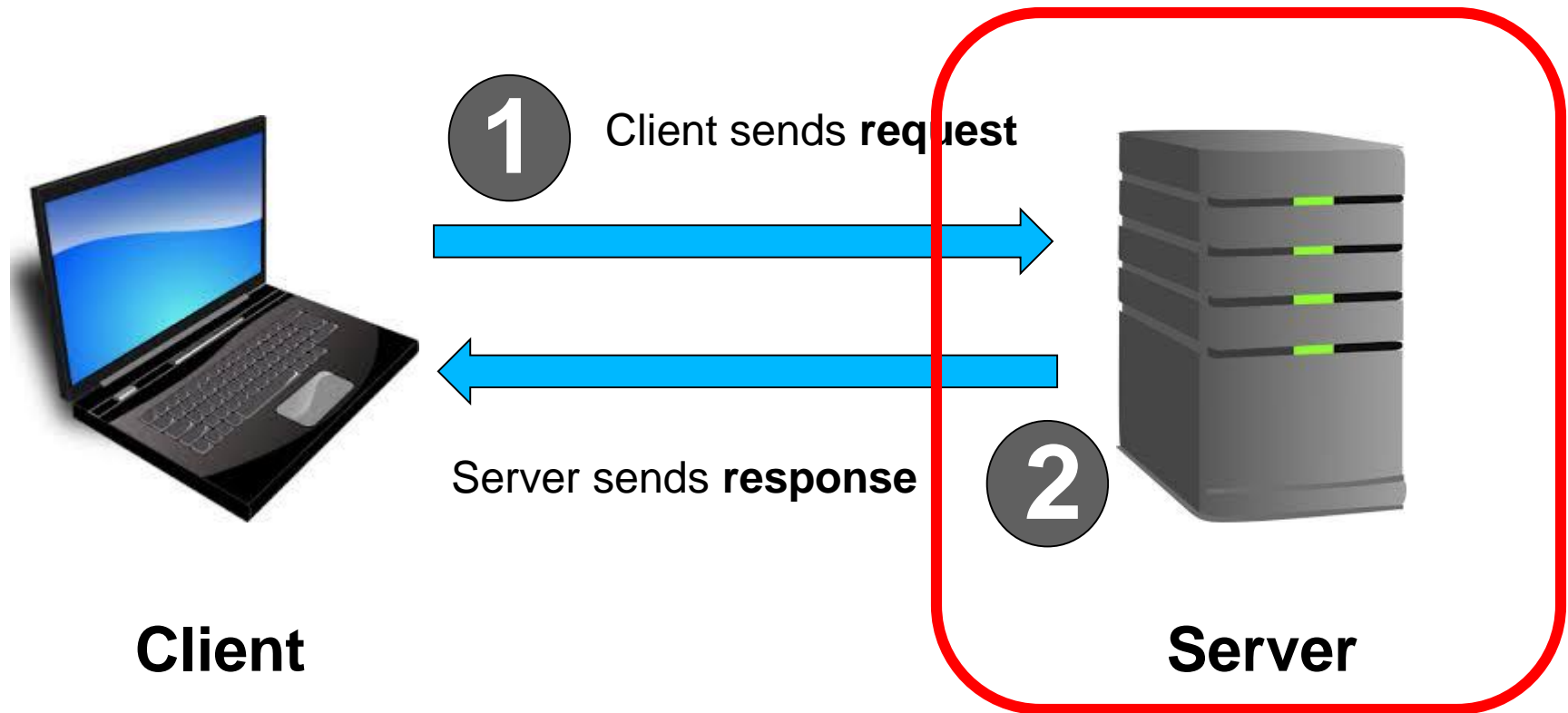
Review: How does a Web Browser Work?

- The World Wide Web utilizes **Hypertext Transfer Protocol (HTTP)** to transfer documents



Review: How does a Web Browser Work?

- The World Wide Web utilizes **Hypertext Transfer Protocol (HTTP)** to transfer documents



What does the Web Server do?

- Listen for and accept incoming HTTP requests
- Parse the HTTP request to determine what is being requested
- Locate (and/or create) the resource being requested
- Construct and send back the HTTP response

Node.js

- Asynchronous, event-driven JavaScript runtime environment for building web applications
- Treats HTTP requests as events that invoke callback functions/handlers that construct the HTTP response
- Also includes a package manager to simplify the deployment of JavaScript apps



Installing Node.js

- You can install Node.js by downloading, running, and finishing the package installer available here:
 - <https://nodejs.org/en/download/>
- Check that installation is correct using: **node -v**
- Update modules using: **npm install npm -g**

Setting up a new project

- Create a new folder for your project
- Use Terminal, Command Prompt, etc. to navigate to that folder
- Set up a new project by running: **npm init**
 - You will be prompted to enter some information about your project
 - Specify “index.js” as your entry point

Setting up a new project

- Your project folder should now have a **package.json** configuration file

```
{
  "name": "helloworld",
  "version": "1.0.0",
  "description": "A basic hello world app",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "edX Learner",
  "license": "ISC"
}
```

Express

- Express is a web application framework that sits on top of a Node.js server
- Express helps you modularize and streamline your web application
- Within Express, you can organize your app in many ways:
 - Define separate modules that have different responsibilities
 - Handle requests via different routes and routers
 - Split each step in the processing of a request into Middlewares

Adding Express

- To use Express, run the following from the folder where you created your Node.js app:
npm install express --save
- The Express **package** will be downloaded to the project and added to your package.json file as a **dependency**
 - **Package**: a package is a module of JavaScript code, usually with a specific purpose, that can be re-used and assembled with other modules
 - **Dependency**: A dependency is a piece of code that your program relies on to work correctly

Express Configuration

- Your package.json file will now have a new section called dependencies
- npm can refer to this in the future and re-download or update your packages as needed

```
{
  "name": "helloworld",
  "version": "1.0.0",
  "description": "A basic hello world app",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "edX Learner",
  "license": "ISC",
  "dependencies": {
    "express": "^4.15.3"
  }
}
```

Express Configuration

- Your package.json file will now have a new section called dependencies
- npm can refer to this in the future and re-download or update your packages as needed

```
{
  "name": "helloworld",
  "version": "1.0.0",
  "description": "A basic hello world app",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "edX Learner",
  "license": "ISC",
  "dependencies": {
    "express": "^4.15.3"
  }
}
```

Express Configuration

- Your package.json file will now have a new section called dependencies
- npm can refer to this in the future and re-download or update your packages as needed

```
{
  "name": "helloworld",
  "version": "1.0.0",
  "description": "A basic hello world app",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "edX Learner",
  "license": "ISC",
  "dependencies": {
    "express": "^4.15.3"
  }
}
```

Express Configuration

- Your package.json file will now have a new section called dependencies
- npm can refer to this in the future and re-download or update your packages as needed

```
{
  "name": "helloworld",
  "version": "1.0.0",
  "description": "A basic hello world app",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "edX Learner",
  "license": "ISC",
  "dependencies": {
    "express": "^4.15.3"
  }
}
```

Express Configuration

- Your package.json file will now have a new section called dependencies
- npm can refer to this in the future and re-download or update your packages as needed

```
{
  "name": "helloworld",
  "version": "1.0.0",
  "description": "A basic hello world app",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "edX Learner",
  "license": "ISC",
  "dependencies": {
    "express": "^4.15.3"
  }
}
```


Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');
var app = express();

app.use('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(3000, () => {
  console.log('Listening on port 3000');
});
```

Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');  
var app = express();  
  
app.use('/', (req, res) => {  
  res.send('Hello World!');  
});  
  
app.listen(3000, () => {  
  console.log('Listening on port 3000');  
});
```

Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');  
var app = express();  
  
app.use('/', (req, res) => {  
  res.send('Hello World!');  
});  
  
app.listen(3000, () => {  
  console.log('Listening on port 3000');  
});
```

Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');
var app = express();

app.use('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(3000, () => {
  console.log('Listening on port 3000');
});
```

Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');
var app = express();

app.use('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(3000, () => {
  console.log('Listening on port 3000');
});
```

Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');
var app = express();

app.use('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(3000, () => {
  console.log('Listening on port 3000');
});
```

Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');
var app = express();

app.use('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(3000, () => {
  console.log('Listening on port 3000');
});
```

Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');
var app = express();

app.use('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(3000, () => {
  console.log('Listening on port 3000');
});
```


Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');
var app = express();

app.use('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(3000, () => {
  console.log('Listening on port 3000');
});
```

Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');
var app = express();

app.use('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(3000, () => {
  console.log('Listening on port 3000');
});
```

Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');
var app = express();

app.use('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(3000, () => {
  console.log('Listening on port 3000');
});
```

Hello World

- Create an file named **index.js** in your Node.js project root directory with the following contents:

```
var express = require('express');
var app = express();

app.use('/', (req, res) => {
  res.send('Hello World!');
});

app.listen(3000, () => {
  console.log('Listening on port 3000');
});
```

Running Express

- In the project folder, run: **node index.js**
- When the server starts, you should see “Listening on port 3000” written to the console/screen
- Open a browser on the same computer and go to **<http://localhost:3000/>**

Hello World!

Looking Ahead

- How can the server send different responses for different requests?
- How can the server dynamically generate responses?
- How does the server interact with external data sources?