Node Modules

✔ **Done:** View

Learn about the concept of Node modules and how to use them.

4. Exercise: Callbacks and Error Handling In Node

https://www.youtube.com/watch?v=oR-73NaWGKg

Click here if you wish to open this video in its own window.

What You'll Do

- More clearly understand the use of callbacks in Node.
- Learn about a common approach for handling errors in Node.

Instructions

- Update **rectangle.js** as shown below:

```
module.exports = (x, y, callback) => {
    if (x <= 0 || y <= 0) {
        callback(new Error(`Rectangle dimensions must be greater than zero. Received: ${x}, ${y}`));
    } else {
        setTimeout(() =>
            callback(null, {
                perimeter: () => 2 * (x + y),
                area: () => x * y
            }),
            2000
        );
    }
};
```

- Then, update **app.js** as shown below:

```
const rect = require('./rectangle');

function solveRect(l, w) {
    console.log(`Solving for rectangle with dimensions: ${l}, ${w}`);

    rect(l, w, (err, rectangle) => {
        if (err) {
            console.log('ERROR:', err.message);
        } else {
            console.log(`Area of rectangle with dimensions ${l}, ${w} is: ${rectangle.area()}`);
            console.log(`Perimeter of rectangle with dimensions ${l}, ${w} is: ${rectangle.perimeter()}`);
        }
    });
    console.log('This statement is logged after the call to rect()');
}

solveRect(2, 4);
solveRect(3, 5);
solveRect(0, 5);
solveRect(5, -3);
```

- Run the Node application as before and see the result.
- **Optional:** Do a Git commit with the message "Node Callbacks and Error Handling".

Summary

- In this exercise, you learned about using callbacks to delay an operation, and error handling in Node applications.

Additional Resources

- NodeJS - What are callbacks?
- NodeJS - What are the error conventions?
- MDN - JavaScript Global Objects - Error

« PREVIOUS ACTIVITY
Prerequisites: Postman and Node

NEXT ACTIVITY
Code Challenge: Node Modules »