Exercise (Instructions):Node Modules: Callbacks and Error Handling

Objectives and Outcomes

In this exercise, you will learn about callbacks, JavaScript closures and error handling in Node applications. At the end of this exercise, you will be able to:

- Using Callbacks in Node applications
- Error handling in Node applications

Using Callbacks and Error Handling

- Update *rectangle.js* as shown below:

```
module.exports = (x,y,callback) => {

    if (x <= 0 || y <= 0)

        setTimeout(() =>

            callback(new Error("Rectangle dimensions should be greater than zero

              : l = "

                + x + ", and b = " + y),

            null),

            2000);

    else

        setTimeout(() =>

            callback(null, {

                perimeter: () => (2*(x+y)),

                area:() => (x*y)

            }),
```

```
            2000);

}
```

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

```
. . .

function solveRect(l,b) {

    console.log("Solving for rectangle with l = "

                + l + " and b = " + b);

    rect(l,b, (err,rectangle) => {

        if (err) {

          console.log("ERROR: ", err.message);

      }

        else {

            console.log("The area of the rectangle of dimensions l = "

                + l + " and b = " + b + " is " + rectangle.area());

            console.log("The perimeter of the rectangle of dimensions l = "

                + l + " and b = " + b + " is " + rectangle.perimeter());

        }

    });

    console.log("This statement after the call to rect()");

};

. . .
```

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

Conclusions

In this exercise, you learnt about using Callbacks and error handling in Node applications. In addition you learnt about using external Node modules.