COURSEWARE

Istanbul

Contents

Overview

JS library designed to calculate test coverage.

Installation

```
npm i --save-dev nyc
```

This will install the **nyc** cli to your project as a **DevDependency**.

Calculating Coverage

Add a **coverage** script that runs `npm test` with the **nyc** CLI.

```
  "scripts": {
    "start": "node index.js",
    "test": "mocha",
    "coverage": "nyc npm test"
  }
```

This can be called using `npm run coverage` and will give you an output similar to this:

```
---------------|---------|----------|---------|---------|-------------------------
File           | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
---------------|---------|----------|---------|---------|-------------------------
All files      |   58.89 |    18.18 |   25.93 |   58.89 |
 backend       |      90 |       50 |   33.33 |      90 |
  server.js    |      90 |       50 |   33.33 |      90 | 21,25
 backend/config|   91.67 |       50 |     100 |   91.67 |
  db.js        |   91.67 |       50 |     100 |   91.67 | 35
 backend/routes|   41.38 |     6.25 |   21.74 |   41.38 |
  list.js      |   53.33 |     12.5 |   41.67 |   53.33 | 9,16,24-28,34-38,44-48
  todo.js      |   28.57 |        0 |       0 |   28.57 | 6-9,14-18,24-28,34-38,44-48
---------------|---------|----------|---------|---------|-------------------------
```

Looking at the above image, we can see that the project is lacking testing in a lot of areas, now we can identify which files require attention.

Tutorial

1. Import `nyc` package from npm
2. Amend your `package.json` to include a `coverage` script that points to the command `nyc npm test`
3. Write some tests in a folder that correlates to a project you are working on.
4. Run `npm run coverage` and observe the output then go back and amend your tests as needed

Exercises

Revisit a previous project you have worked on and run the coverage on the tests you've written.