TESTING

App Testing

- Testing should occur on a number of levels
  - *Unit testing*
  - *End to End*

Q-Unit

- Unit Testing Framework designed to work with JavaScript
- Tests are defined along with correct results
- All unit tests can be run
- May not be exhaustive, but instead give a basic check that code is nominally correct.

- (ref)

Q-Unit (Example)

- Builds a driver for the function.
- Description of test
- An assertion of the call and result

```
function add(a, b)
    { return a + b; }

QUnit.module('add', hooks =>
    { QUnit.test('should add two numbers', assert =>
        { assert.equal(add(1, 1), 2); }); });
```

Q-Unit

- Add a test directory to your project.
- Inside define a test file
- Export the function that you want to test and require it in the test file

```
const add require('../add.js')

QUnit.module('add')

QUnit.test('should add two numbers', assert =>
        { assert.equal(add(1, 1), 2); });
```

Running Q-Unit

- Add scripts inside your package.json.
- Then you can run the tests via npm

```
{ "scripts": {
        "test": "qunit" }
 }
```

npm test

Q-Unit Assertions

- Besides the standard equal which compares two expressions there are many options. A few of them follow
  - *Assert.true() is an expression true*
  - *Assert.rejects() does the promise reject*
  - *Assert.throws() does the expression throw an error*
  - *Assert.timeout() set the time in milliseconds to wait on the test of an asynchronous expression being tested.*

Q-Unit Setup

- Can define code that will be run before each test
- There are hooks that you can use to define callbacks that are triggered at certain points in the unit testing process.

References

- An introduction to JS Unit testing

Higher level testing

- Functionality Testing
  - *Test for broken links*
  - *Test forms*
    - Default values
    - Error messages
    - Submission
  - *Test Cookies*
    - Session cookies expire and are deleted
  - *Test HTML/CSS*

Higher level testing

- Usability Testing
  - *Focus group*
  - *Give tasks and see how long a person takes to complete the task.*
  - *Check navigation*
  - *Check content*

Higher level testing

- System Testing
  - *Requests to the data base are sent*
  - *DB integrity is maintained*
  - *Results are displayed*
  - *Results are correct*
  - *Results are delivered within required time*
  - *Errors are caught and shown to DB admin only*
  - *Stress test*
  - *Check against denial of service attacks*

Higher level testing

- Compatibility Testing
  - *Check that the web site functions correctly on different browsers.*
  - *Check functionality against browser versions*
  - *Check functionality on mobile platforms.*
  - *Use an automated tool*
    - A starting point for tools

Higher level testing

- Security Testing
  - *Check secured pages*
  - *Check private files not accessible*
  - *Check session/cookie timeouts.*
  - *Check SSL*

Tool: Selenium

- Build tests that check for the proper function of a web site.
- Tests define actions and expected results at the browser level.
- Has a playback tool
- Test are separate from the code/server.
- Single test can be run on multiple browsers/versions.