Testing Library

“The more your tests resemble the way your software is used, the more confidence they can give you.”

- Kent C. Dodds

- Test DOM nodes, not components
- Tests should work the way the application is to be used

https://testing-library.com/docs/guiding-principles
Testing Library

Query variants

- **getBy* / getAllBy**
  - queries the DOM for the first matching node/array of matches
  - throws error if none are found
  - single version throws error if more than one is found

- **queryBy* / queryAllBy**
  - queries the DOM for the first matching node/array of matches
  - returns null or empty [] if none are found
  - single version throws error if more than one is found

- **findBy* / findAllBy**
  - returns a Promise which resolves when a matching node(s) is found
  - throws error after 1000ms if none are found
  - single version throws error if more than one is found

Testing Library

Query types (partial list)

- **ByText**
  - looks for an element based on the text contents of the node

- **ByRole**
  - search based on the role of the component (e.g., listitem, button, textbox, etc...)
  - can narrow the search with options like the aria-label

- **ByTestId**
  - looks for specific components based on data-testid value
  - this is basically the cheat code and not really in the spirit of the library

A query is a variant + a type
e.g., queryByText() or findAllByRole

Testing Library

Tools

- **screen**
  - allows us to query the entire DOM (screen.findByText('example'))
  - screen.debug() prints out the virtual DOM

- **fireEvent.type** (component, event properties)
  - allows us to simulate user interaction
    - type is any kind of HTML event: click, change, drag, drop, keyDown, etc...

Example

renders the component

uses mock function for callback

finds the section

finds all titles

tests that each title is visible
Mocking fetch
fetch-mock-jest

```javascript
beforeAll(() => {
  localFilms = films.map((film) => ({ ...film }));
  fetchMock.reset();
  fetchMock.get('http://basin.cs.middlebury.edu:3042/api/films', () => localFilms);
  fetchMock.put('http://basin.cs.middlebury.edu:3042/api/films/102', (url, options) => {
    const id = 102;
    const modifiedFilm = JSON.parse(options.body);
    localFilms = localFilms.map((film) => {
      if (id === film.id) {
        return modifiedFilm;
      } else {
        return film;
      }
    });
    return modifiedFilm;
  });
});
```