Cypress

Cypress Overview

- "Fast, easy and reliable testing for anything that runs in a browser"
  - https://www.cypress.io/
- Supports end-to-end (E2E) tests
- Only runs in Chrome, no other browsers
- Documentation at https://docs.cypress.io/

Cypress With Vue ...

- Install options
  - use Vue CLI to create a new application and select "E2E Testing"
  - add to an existing application with **`vue add @vue/cli-plugin-e2e-cypress`**
- Either way these files are added
  - **`tests/e2e/.eslintrc.js`** configures ESLint to understand Cypress tests
  - **`tests/e2e/specs/test.js`** contains an example test
  - **`cypress.json`** registers plugin file in next bullet
  - **`tests/e2e/plugins/index.js`** configures Cypress plugin
  - two files under **`tests/e2e/support`** that support adding Cypress commands, but none are added
  - one file under **`tests/e2e/plugins`** that specifies some folder and file locations

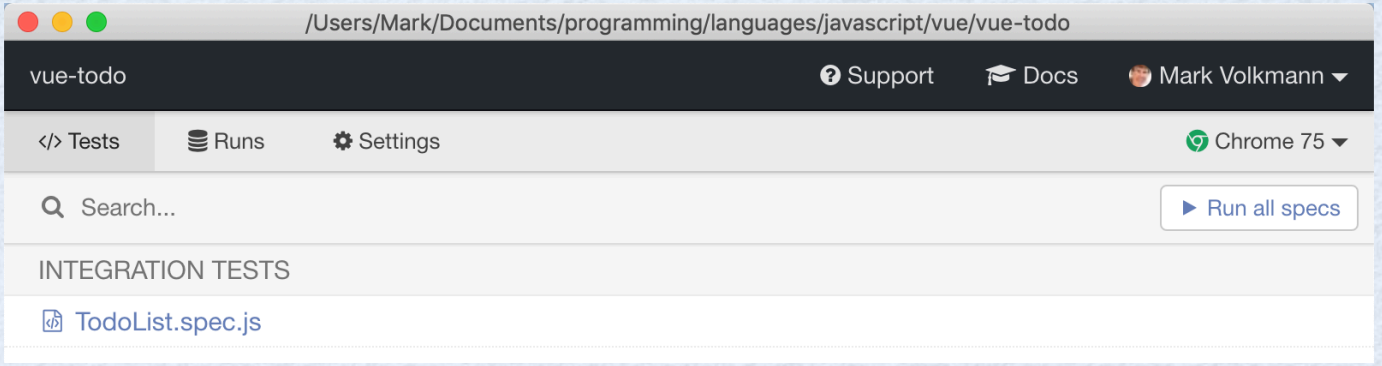
don't need to modify any of these; delete **`tests/e2e/specs/test.js`** after creating your own test files in that directory

Cypress With Vue

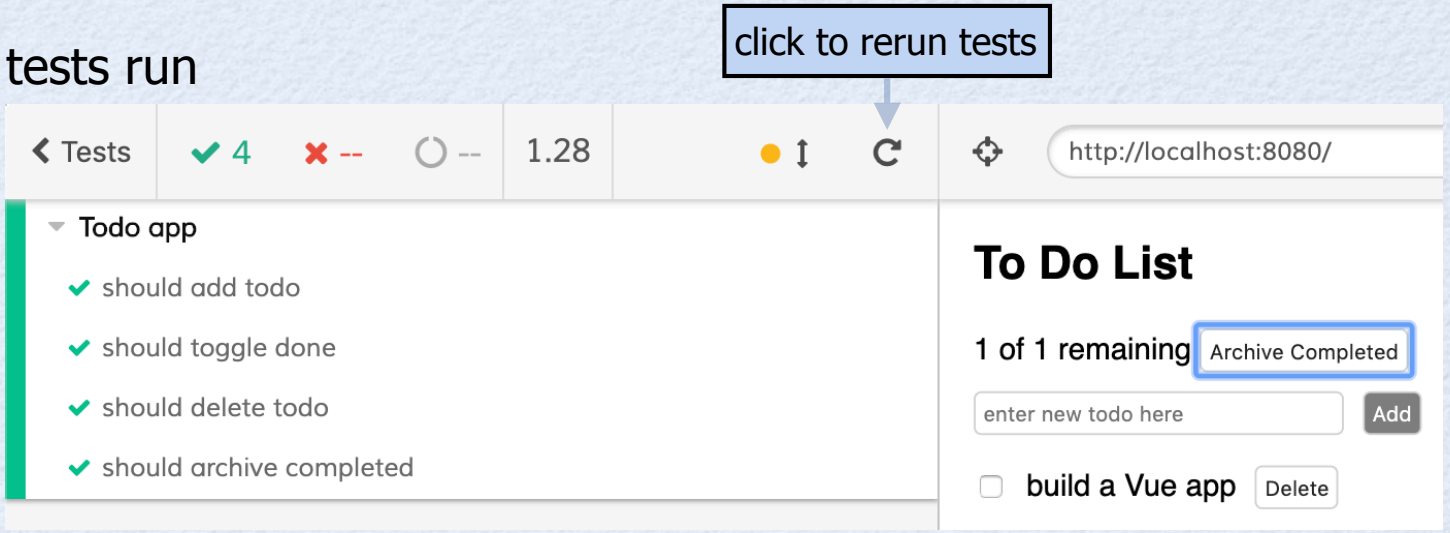
- Tests can be written in any `.js` files under `e2e/specs`
- To run tests
  - start Cypress with `npm run test:e2e`
  - will start application server for you
  - click "Run all specs" button

Cypress UI

- Initial Cypress window
  - press "Run all specs" button



- Cypress window after tests run
  - passing tests are green, failing are red
  - click a test description to see details on what it did



- Automatically reruns tests when test files are modified, but not when application files are modified

Writing Tests

- Use CSS selectors to find page elements to test
- Tests use familiar methods found in other test libraries like Jest
  - `describe`, `before`, `beforeEach`, `after`, `afterEach`, `it`, ...
- `cy` is a global variable
  - refers to an object with many methods
- Queries to find elements on page wait for them appear
  - by default waits 4 seconds

Todo App Cypress Test ...

```
describe('Todo app', () => {
  it('should add todo', () => {
    cy.visit('/');
    cy.contains('1 of 2 remaining');
    // "Add" button should be disabled until text is entered.
    cy.contains('Add')
      .as('addBtn')
      .should('be.disabled');

    // Enter todo text.
    const todoText = 'buy milk';
    cy.get('.todo-input')
      .as('todoInput')
      .type(todoText);

    cy.get('@addBtn').should('not.be.disabled');
    cy.get('@addBtn').click();

    cy.get('@todoInput').should('have.value', ''); // cleared
    cy.get('@addBtn').should('be.disabled');
    cy.contains(todoText);
    cy.contains('2 of 3 remaining');
  });
```

allows referring to this button again later with `@addBtn`

... Todo App Cypress Test ...

```
it('should toggle done', () => {
  cy.visit('/');
  cy.contains('1 of 2 remaining');

  // Find first checkbox and toggle it.
  cy.get('input[type=checkbox]')
    .first()
    .as('cb1')
    .click();
  cy.contains('2 of 2 remaining');

  // Toggle same checkbox again.
  cy.get('@cb1').check();
  cy.contains('1 of 2 remaining');
});
```

... Todo App Cypress Test ...

```
it('should delete todo', () => {
  cy.visit('/');
  cy.contains('1 of 2 remaining');

  const todoText = 'learn Vue'; // first todo
  cy.contains('ul', todoText);

  // Click first "Delete" button.
  cy.contains('Delete').click();
  cy.contains('ul', todoText).should('not.exist');
  cy.contains('1 of 1 remaining');
});
```

... Todo App Cypress Test

```
  it('should archive completed', () => {
    cy.visit('/');

    const todoText = 'learn Vue'; // first todo
    cy.contains('ul', todoText);

    // Click "Archive Completed" button.
    cy.contains('Archive Completed').click();
    cy.contains('ul', todoText).should('not.exist');
    cy.contains('1 of 1 remaining');
  });
});
```