Web Development

COMP 431 / COMP 531

Front-End Unit Testing

Scott E Pollack, PhD

September 29, 2016

Recap

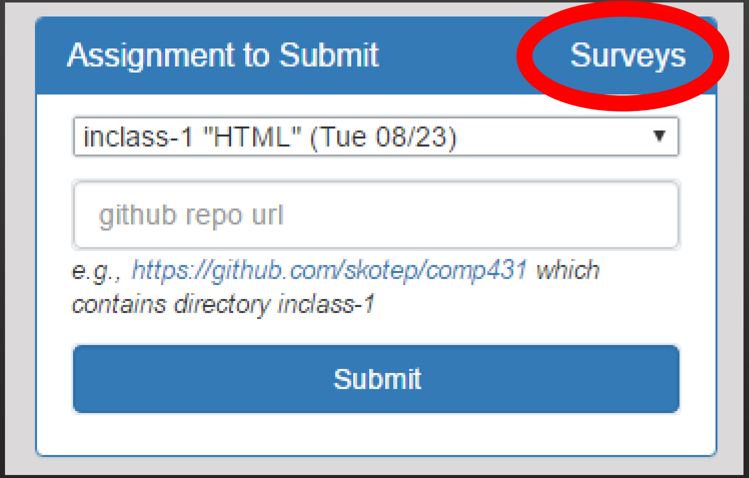
- HTML and HTML5, Storage, Canvas
- JavaScript and Scope
- Forms, CSS, Events
- jQuery, AJAX, and fetch
- Modern JS
- MVC, React and Redux

~~Modern JS~~
~~MVC~~
~~React & Redux~~
Testing

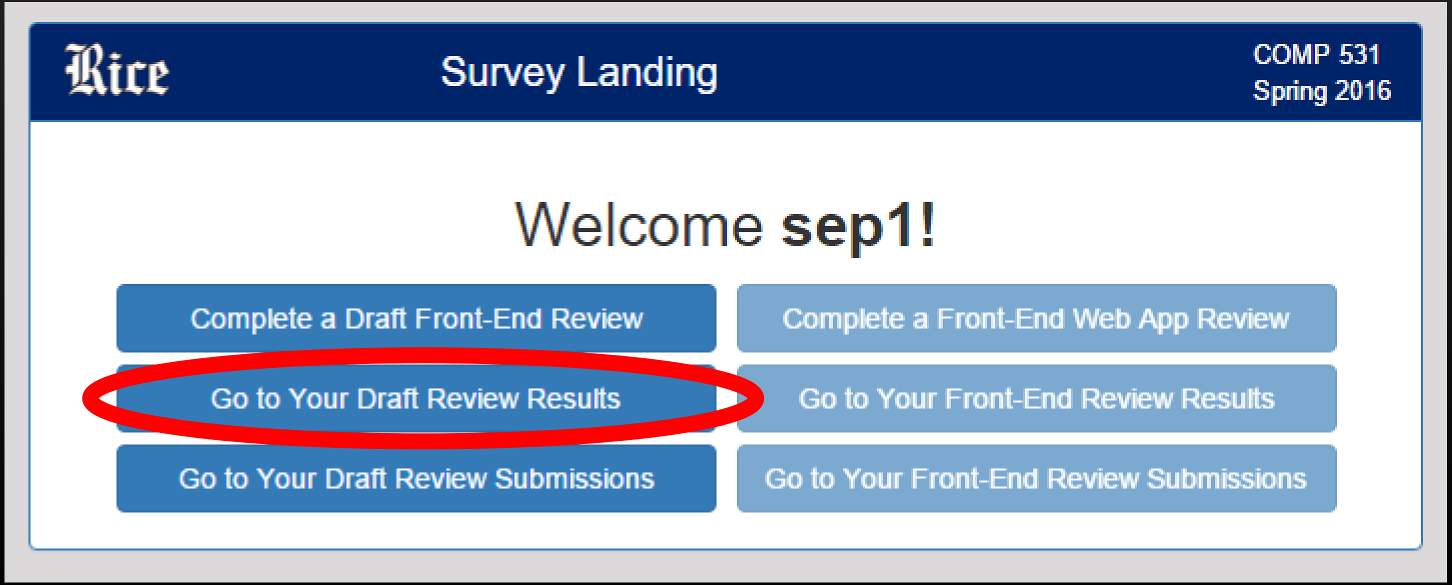
Homework Assignment 4
(JavaScript Game)
Due Thursday 10/6

Draft Front-End Review
User Feedback

- Go to the normal assignment submission page: https://webdev-rice.herokuapp.com/
- Click on "Surveys"
- Click on "Go to Your Draft Review Results"

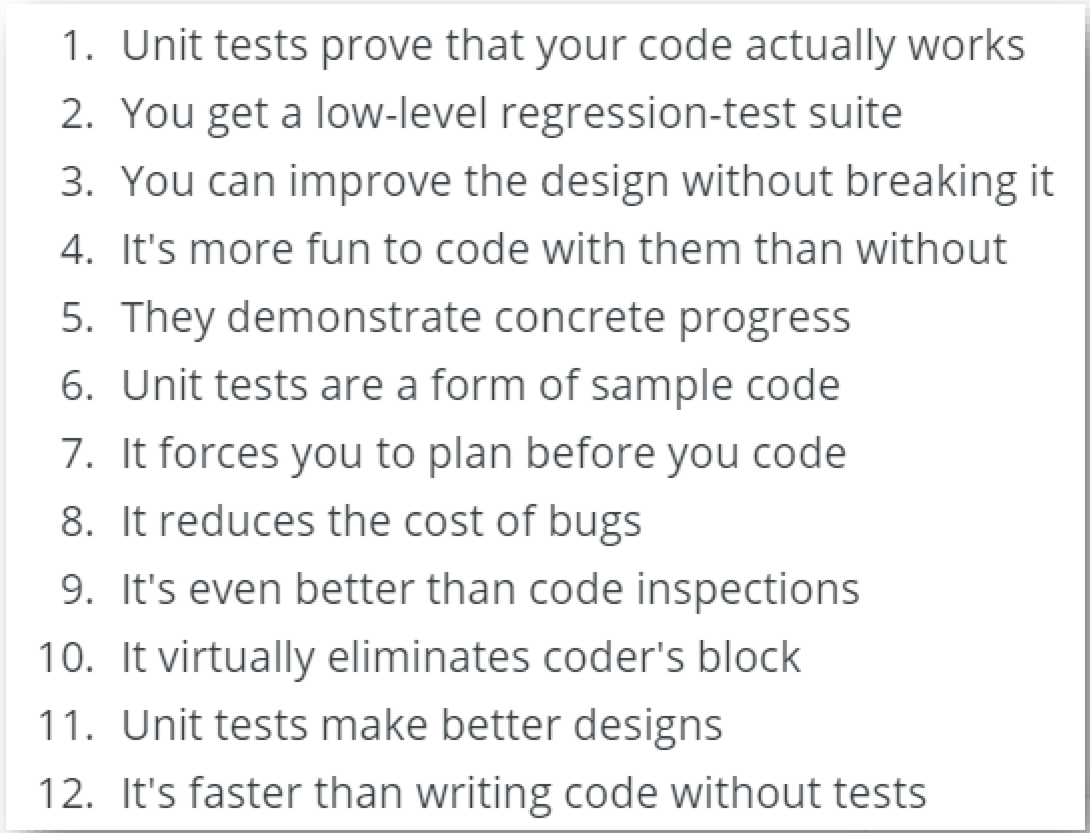


e.g., https://github.com/skotep/comp431 which contains directory inclass-1


Rice
Survey Landing
COMP 531
Spring 2016
Complete a Draft Front-End Review
Complete a Front-End Web App Review
Go to Your Draft Review Results
Go to Your Front-End Review Results
Go to Your Draft Review Submissions
Go to Your Front-End Review Submissions

Welcome **sep1!**

Testing



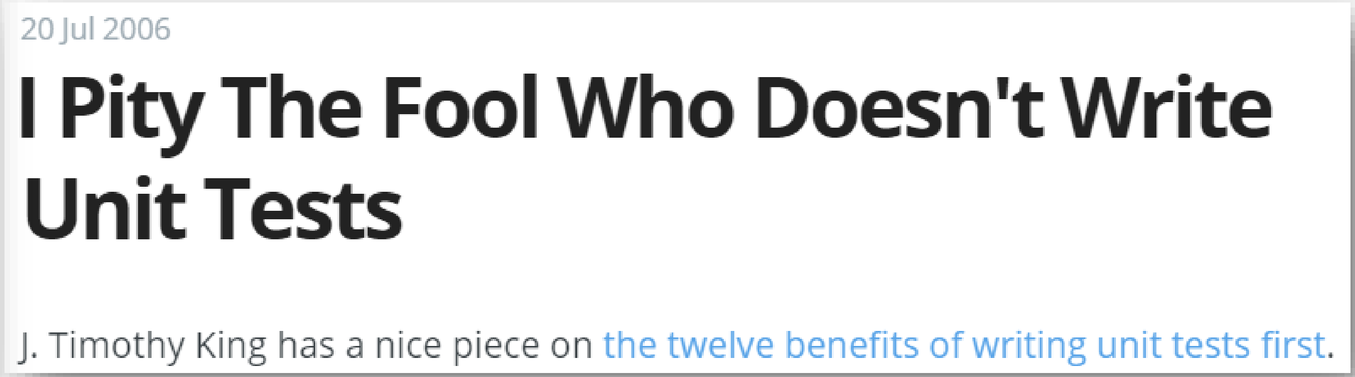
1. Unit tests prove that your code actually works
2. You get a low-level regression-test suite
3. You can improve the design without breaking it
4. It's more fun to code with them than without
5. They demonstrate concrete progress
6. Unit tests are a form of sample code
7. It forces you to plan before you code
8. It reduces the cost of bugs
9. It's even better than code inspections
10. It virtually eliminates coder's block
11. Unit tests make better designs
12. It's faster than writing code without tests

CODING HORROR

programming and human factors

Copyright Jeff Atwood © 2015

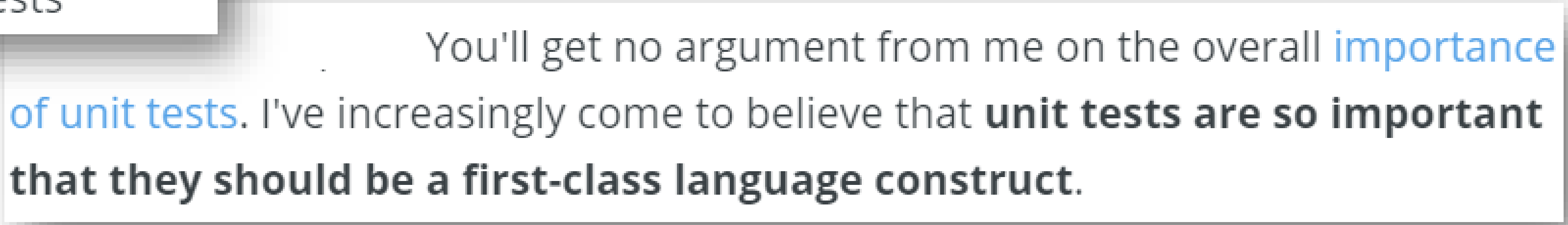
Logo image © 1993 Steven C. McConnell



20 Jul 2006

I Pity The Fool Who Doesn't Write Unit Tests

J. Timothy King has a nice piece on the twelve benefits of writing unit tests first.



You'll get no argument from me on the overall importance of unit tests. I've increasingly come to believe that **unit tests are so important that they should be a first-class language construct**.

http://blog.codinghorror.com/i-pity-the-fool-who-doesnt-write-unit-tests/
http://www.jtse.com/blog/2006/07/11/twelve-benefits-of-writing-unit-tests-first

Pieces of Testing

- Assertion
  - A single condition expected to be true
- Test Case
  - Executes an atomic component of the larger program
  - May contain multiple assertions
- Test Suite
  - Collection of Test Cases
- Test Runner
  - Executes the test suite

Testing Frameworks

Yesteryear: Jasmine

- Assertion
- “Describe it” for cases
- Suite = all in file
- Runner in page

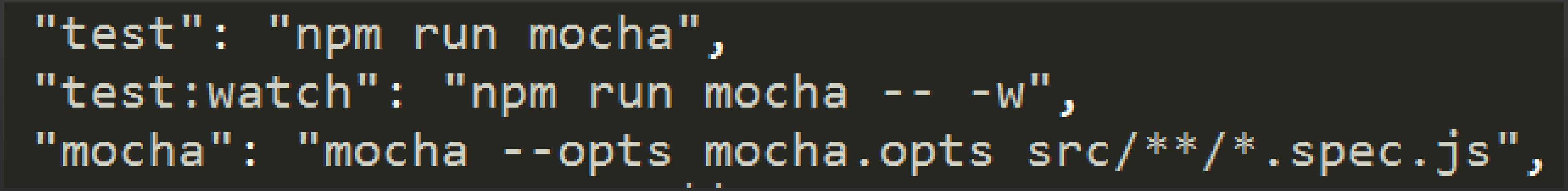
<u>state of the art</u>

Mocha (test)
Chai (assertion)
Sinon (mock)

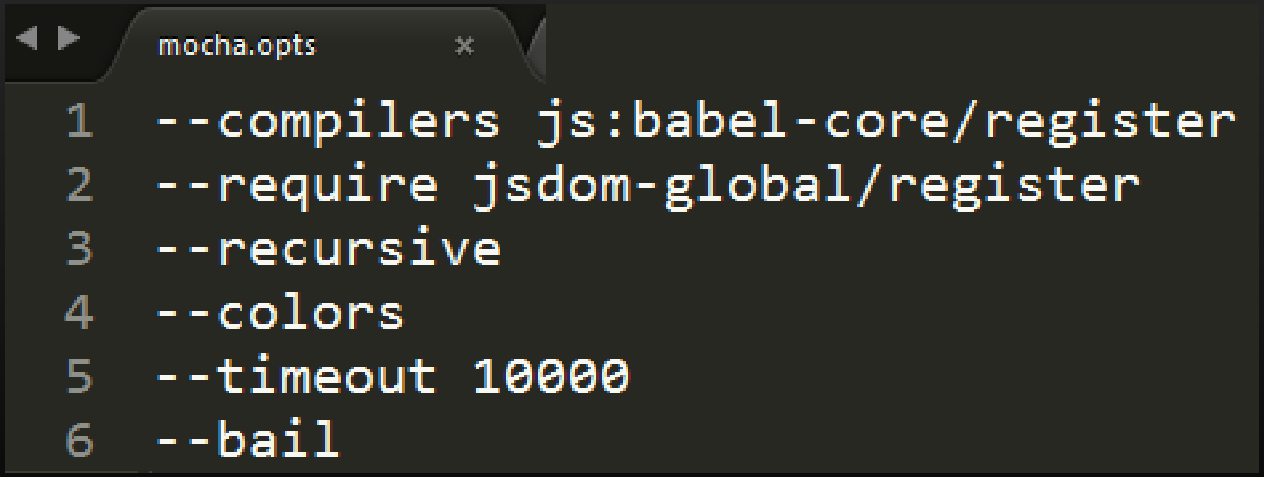
- QUnit (jQuery)
- Jest
- JsUnit
- Buster.js
- TestSwarm
- BrowserSwarm
- Intern
- Chutzpah
- Dojo Object Harness

- Pavlov
- jsTestDriver
- HtmlUnit
- Celerity
- Schnell
- Screw.Unit

Testing with Mocha



```
"test": "npm run mocha",
"test:watch": "npm run mocha -- -w",
"mocha": "mocha --opts mocha.opts src/**/*.spec.js",
```

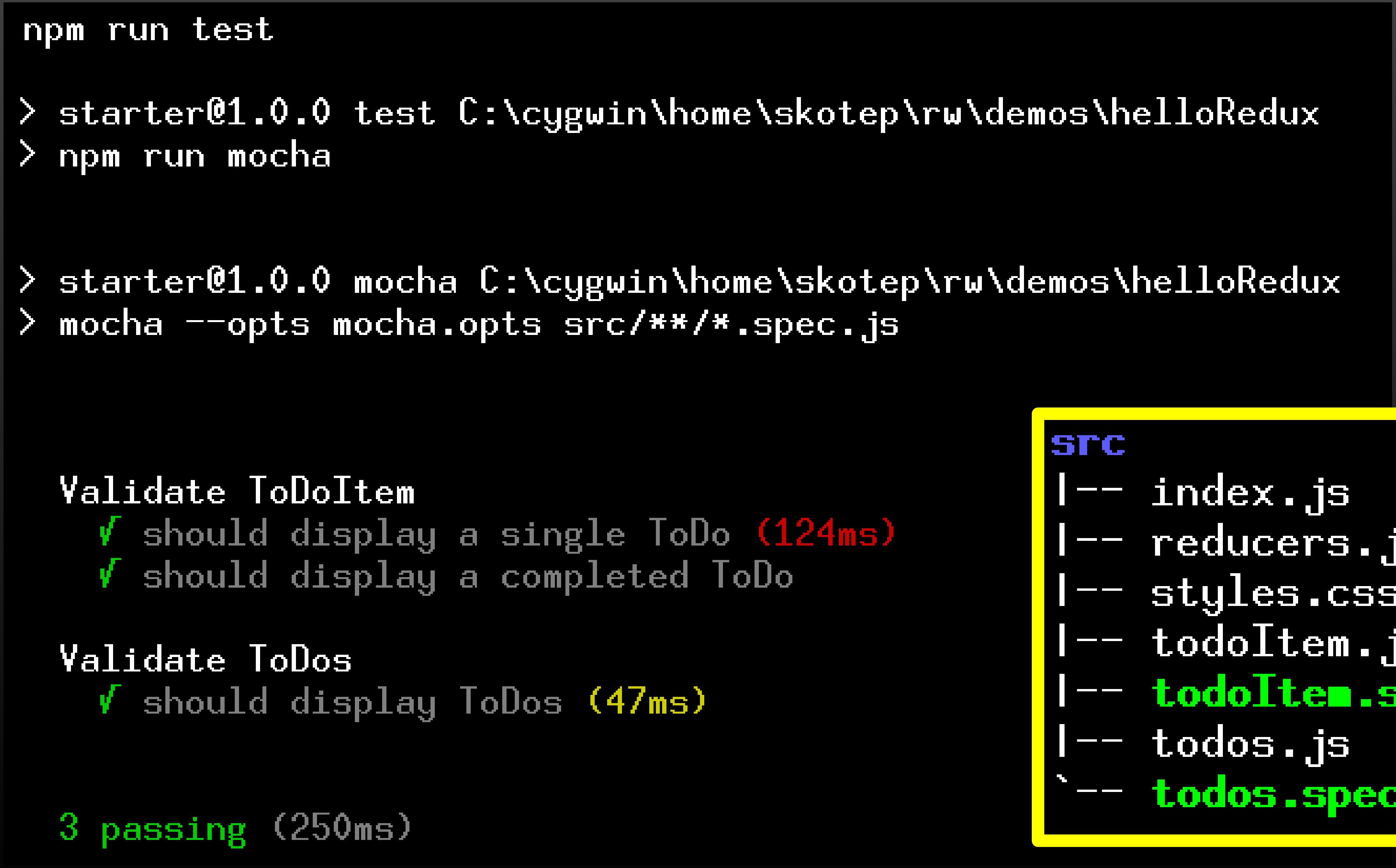


```
--compilers js:babel-core/register
--require jsdom-global/register
--recursive
--colors
--timeout 10000
--bail
```

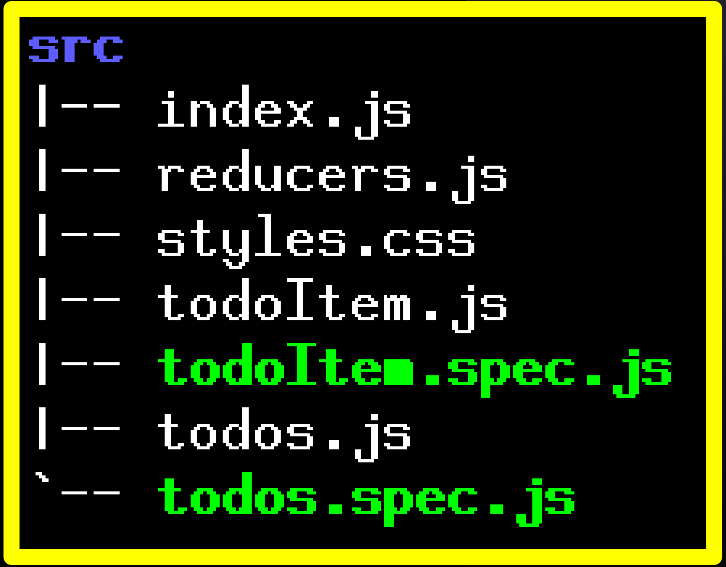
Use babel to transpile code while running
Use jsdom in global scope for DOM mocking
recurse directories looking for tests
Use colors in output
Timeout for tests is 10 seconds
Bail on problems

Console Reporter

html reporter
@see inclass fetch exercise



```
npm run test

> starter@1.0.0 test C:\cygwin\home\skotep\rw\demos\helloRedux
> npm run mocha


> starter@1.0.0 mocha C:\cygwin\home\skotep\rw\demos\helloRedux
> mocha --opts mocha.opts src/**/*.spec.js



  Validate ToDoItem
    √ should display a single ToDo (124ms)
    √ should display a completed ToDo

  Validate ToDos
    √ should display ToDos (47ms)


  3 passing (250ms)
```



```
src
|-- index.js
|-- reducers.js
|-- styles.css
|-- todoItem.js
|-- todoItem.spec.js
|-- todos.js
`-- todos.spec.js
```

Separation of Concerns

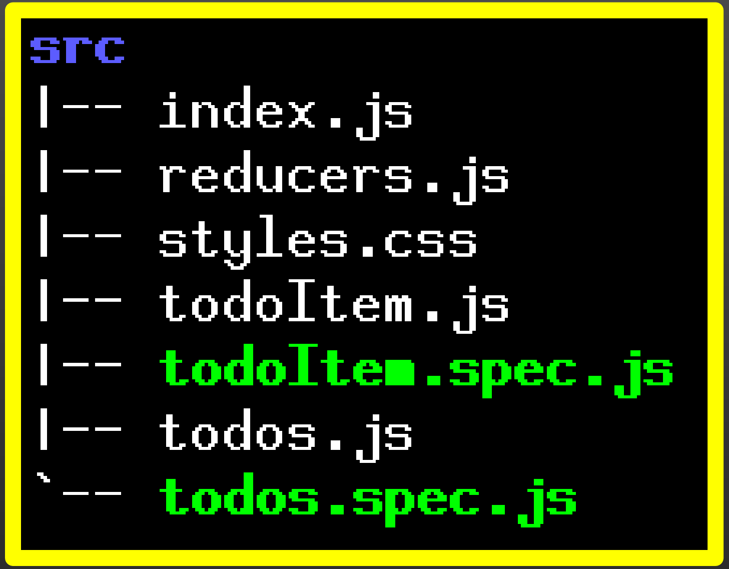
- What do we test and how?

```
src
|-- index.js
|-- reducers.js
|-- styles.css
|-- todoItem.js
|-- todoItem.spec.js
|-- todos.js
`-- todos.spec.js
```

Separation of Concerns

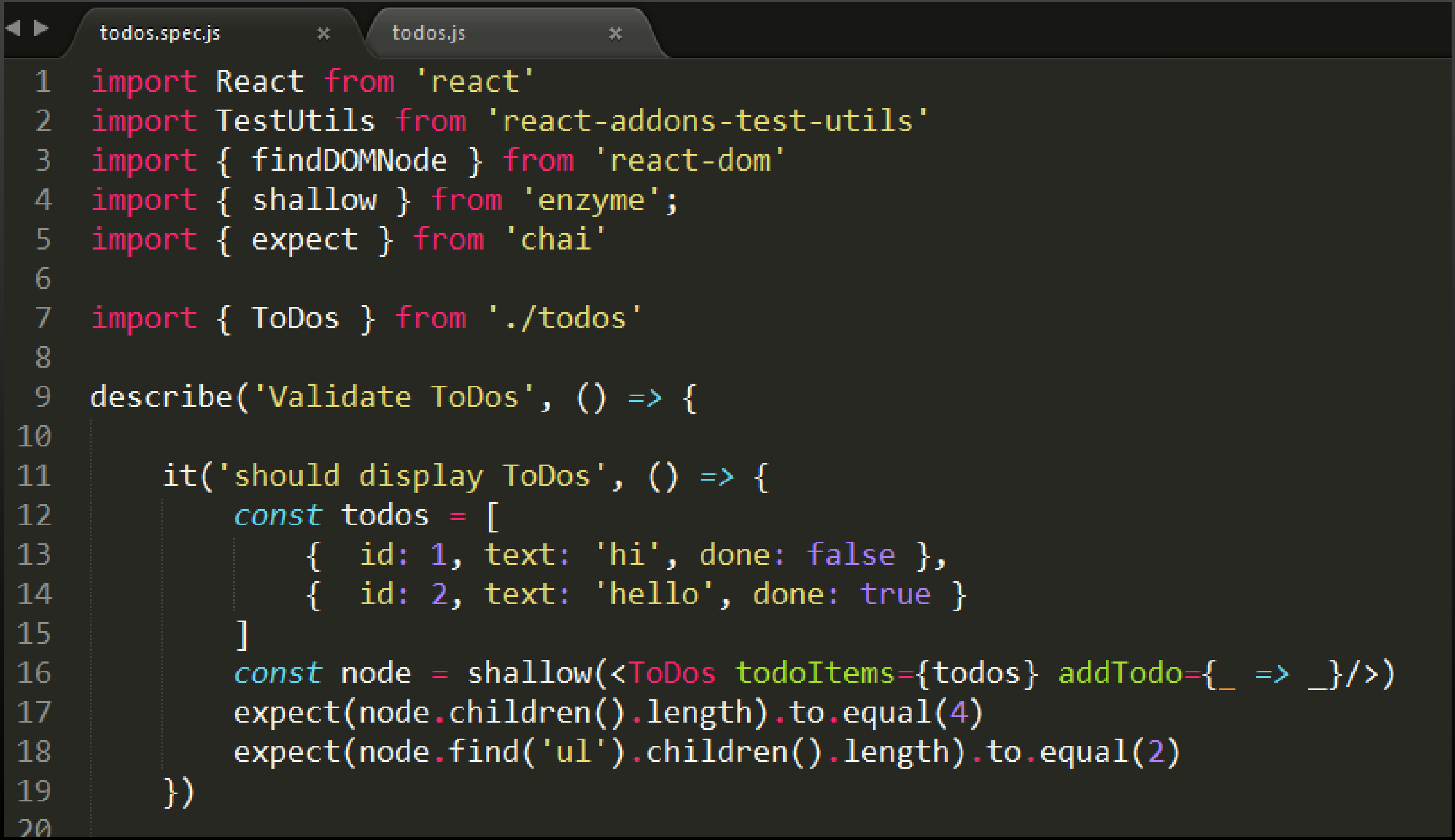
- What do we test and how?

- Rendering logic
  - React components
- Business logic
  - Reducers
  - Complex Actions



```
src
|-- index.js
|-- reducers.js
|-- styles.css
|-- todoItem.js
|-- todoItem.spec.js
|-- todos.js
`-- todos.spec.js
```

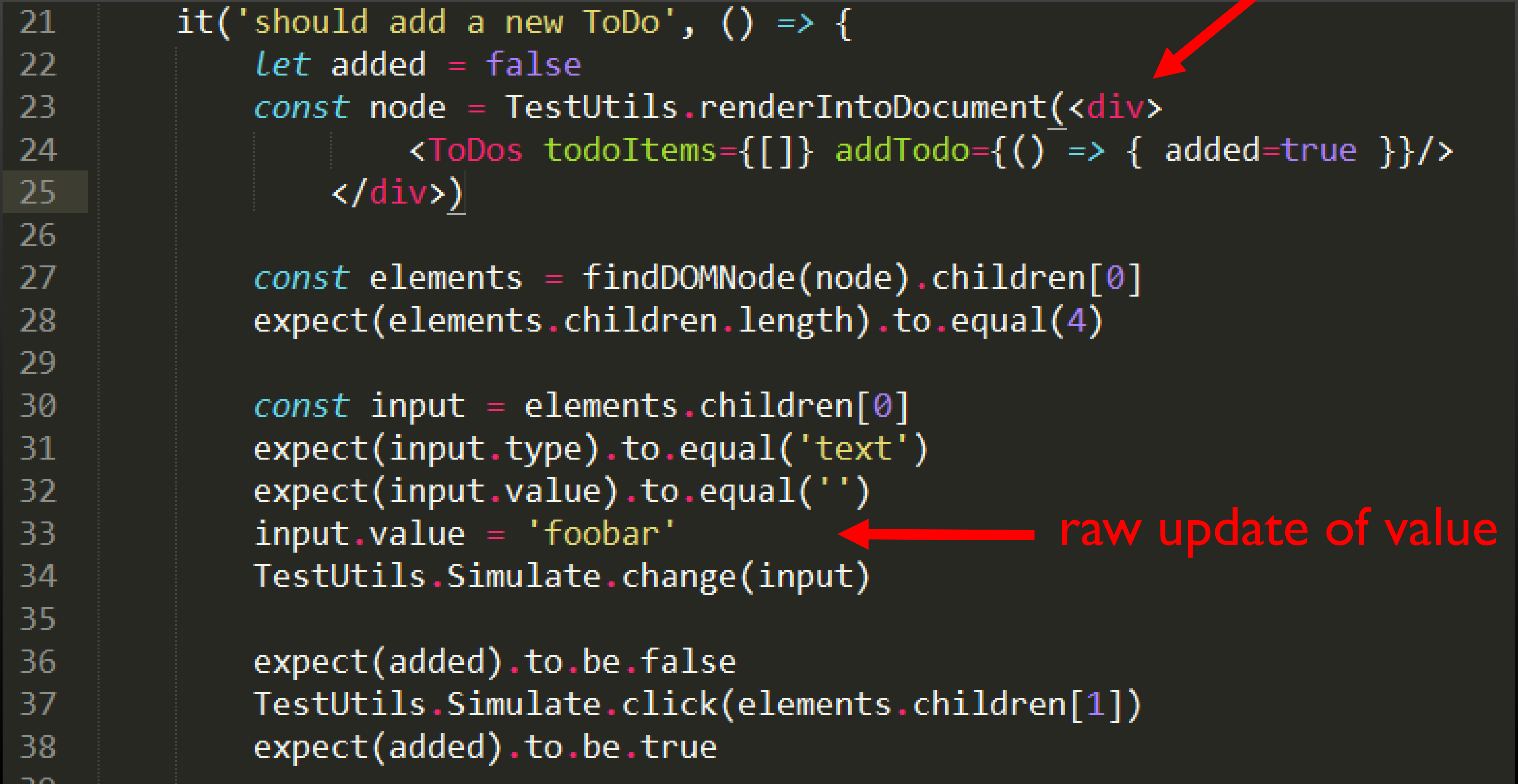
Rendering Tests: shallow



```
import React from 'react'
import TestUtils from 'react-addons-test-utils'
import { findDOMNode } from 'react-dom'
import { shallow } from 'enzyme';
import { expect } from 'chai'

import { ToDos } from './todos'

describe('Validate ToDos', () => {

    it('should display ToDos', () => {
        const todos = [
            {  id: 1, text: 'hi', done: false },
            {  id: 2, text: 'hello', done: true }
        ]
        const node = shallow(<ToDos todoItems={todos} addTodo={_ => _}/>)
        expect(node.children().length).to.equal(4)
        expect(node.find('ul').children().length).to.equal(2)
    })
```

Rendering Tests: deep

wrapper for simple component



```
it('should add a new ToDo', () => {
    let added = false
    const node = TestUtils.renderIntoDocument(<div>
            <ToDos todoItems={[]} addTodo={() => { added=true }}/>
        </div>)

    const elements = findDOMNode(node).children[0]
    expect(elements.children.length).to.equal(4)

    const input = elements.children[0]
    expect(input.type).to.equal('text')
    expect(input.value).to.equal('')
    input.value = 'foobar'
    TestUtils.Simulate.change(input)

    expect(added).to.be.false
    TestUtils.Simulate.click(elements.children[1])
    expect(added).to.be.true
```

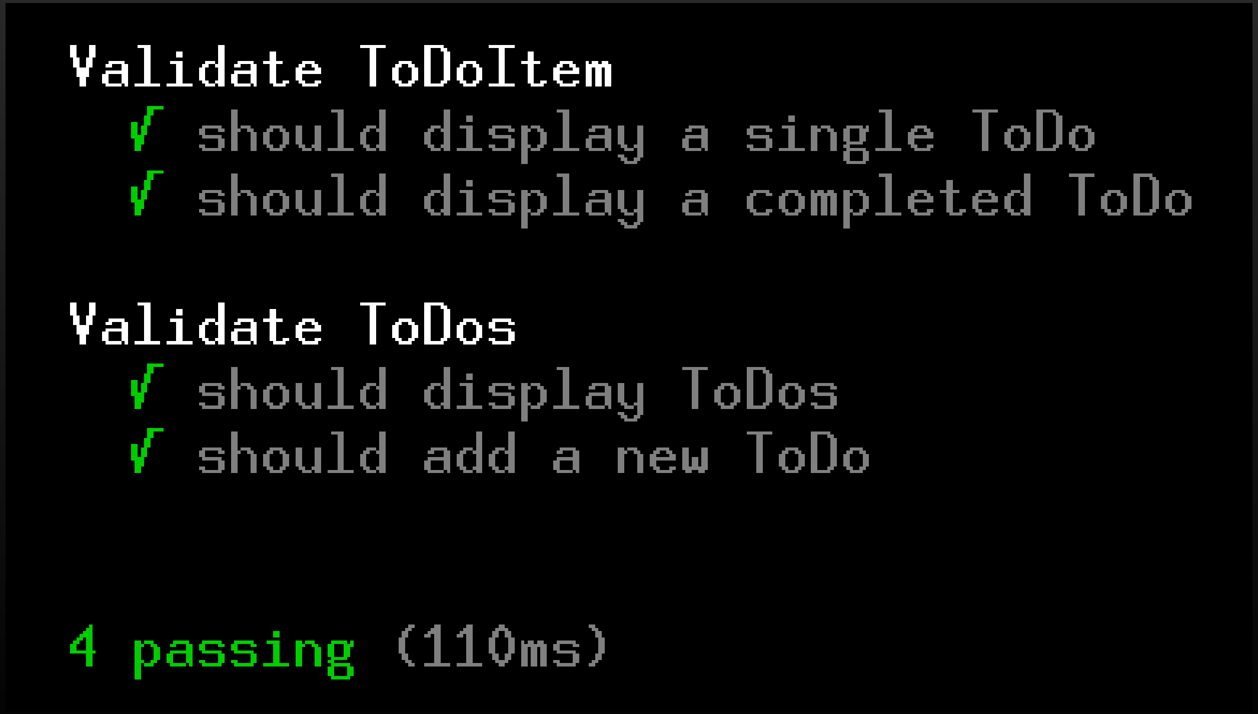
raw update of value

Test Driven Development

- Add filter functionality:
  - all tasks
  - uncompleted tasks
  - completed tasks.

- Start by spinning up mocha in watch mode:

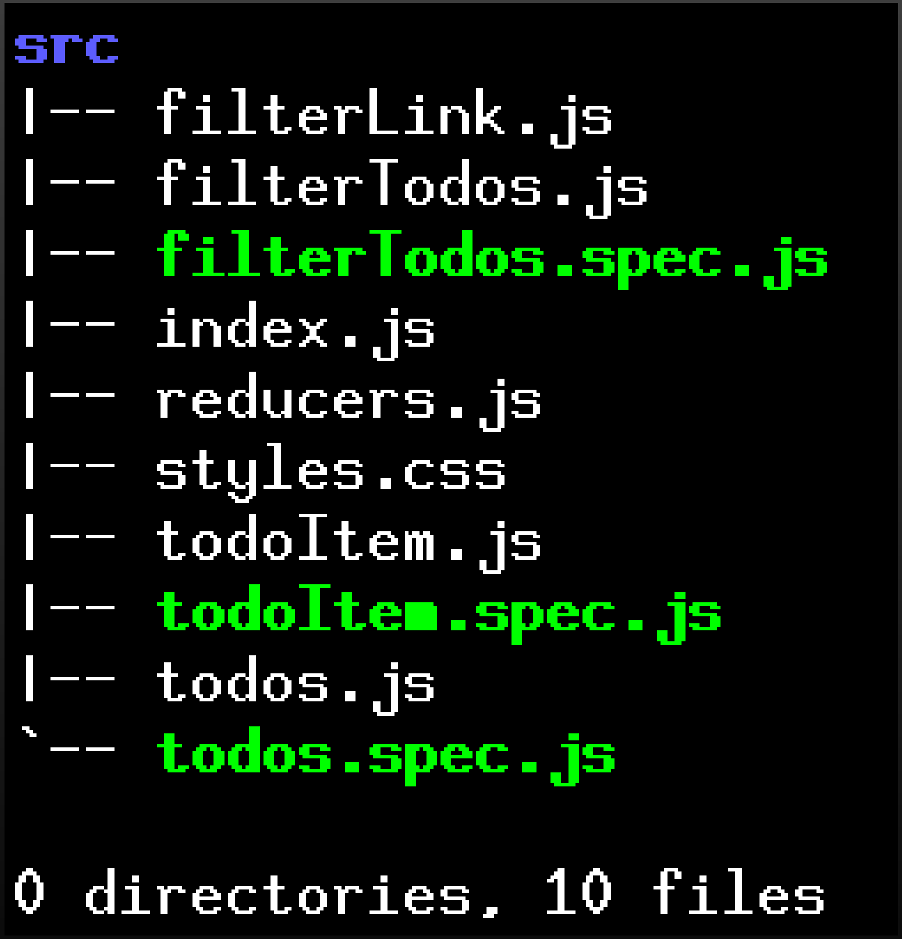
```
npm run test:watch
```



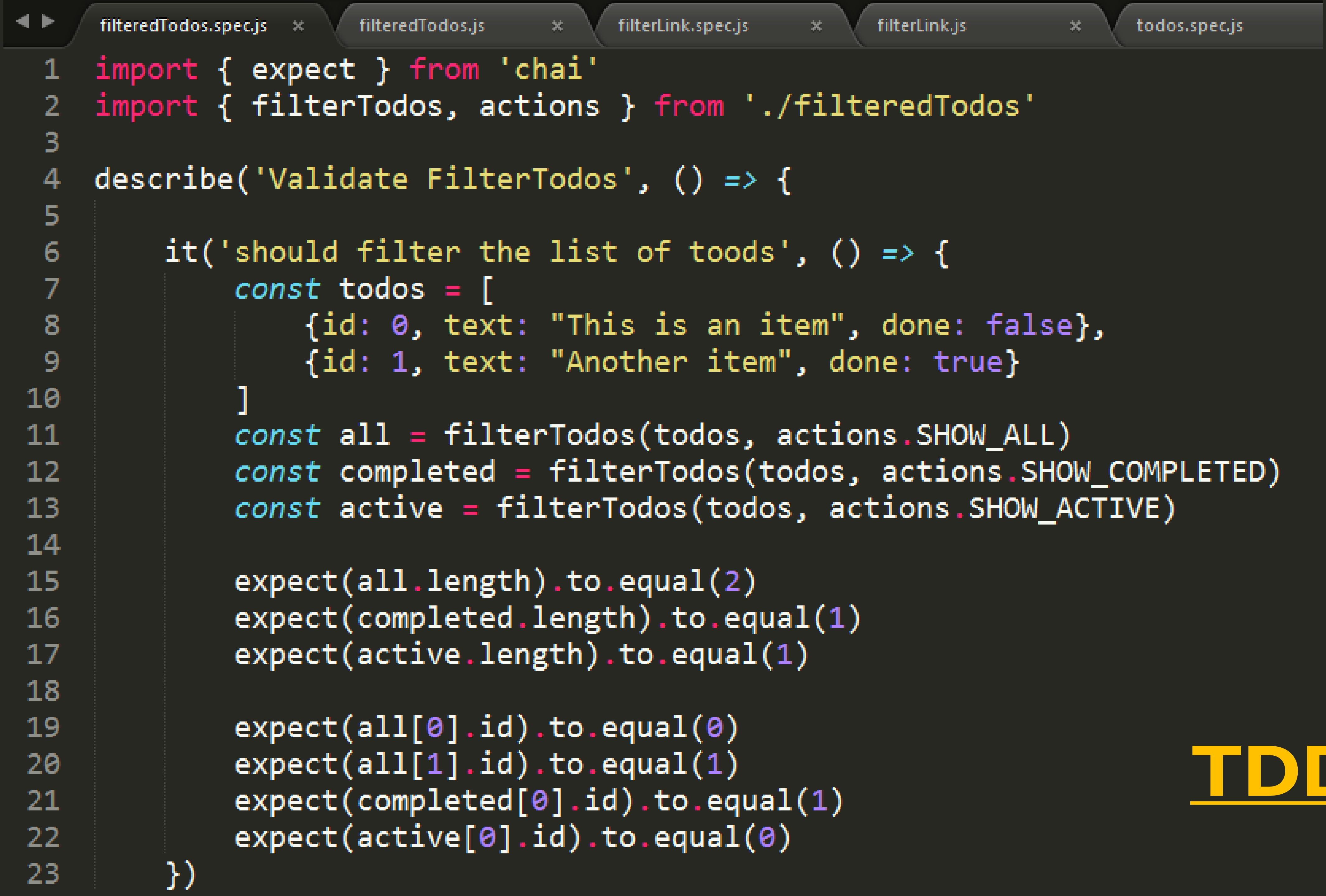
```
Validate ToDoItem
  √ should display a single ToDo
  √ should display a completed ToDo

Validate ToDos
  √ should display ToDos
  √ should add a new ToDo


4 passing (110ms)
```

Write Tests First

- We will need:
  - Three Links to click
  - Link dispatches action to update the filtered list of todos
  - FilteredTodo wraps ToDo with filtered list
- Tests:
  - Filter action properly filters list



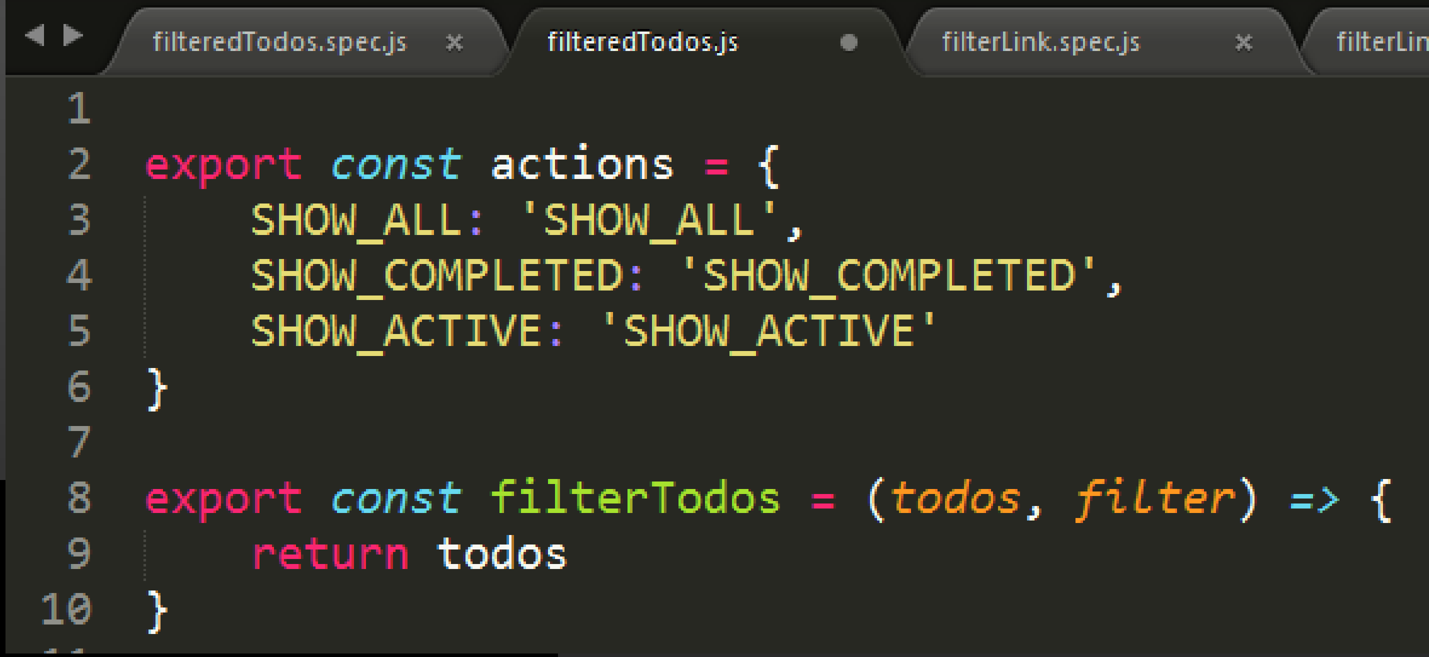
```
src
|-- filterLink.js
|-- filterTodos.js
|-- filterTodos.spec.js
|-- index.js
|-- reducers.js
|-- styles.css
|-- todoItem.js
|-- todoItem.spec.js
|-- todos.js
`-- todos.spec.js

0 directories, 10 files
```


filteredTodos.spec.js × filteredTodos.js × filterLink.spec.js × filterLink.js × todos.spec.js

```
import { expect } from 'chai'
import { filterTodos, actions } from './filteredTodos'

describe('Validate FilterTodos', () => {

    it('should filter the list of toods', () => {
        const todos = [
            {id: 0, text: "This is an item", done: false},
            {id: 1, text: "Another item", done: true}
        ]
        const all = filterTodos(todos, actions.SHOW_ALL)
        const completed = filterTodos(todos, actions.SHOW_COMPLETED)
        const active = filterTodos(todos, actions.SHOW_ACTIVE)

        expect(all.length).to.equal(2)
        expect(completed.length).to.equal(1)
        expect(active.length).to.equal(1)

        expect(all[0].id).to.equal(0)
        expect(all[1].id).to.equal(1)
        expect(completed[0].id).to.equal(1)
        expect(active[0].id).to.equal(0)
    })
```

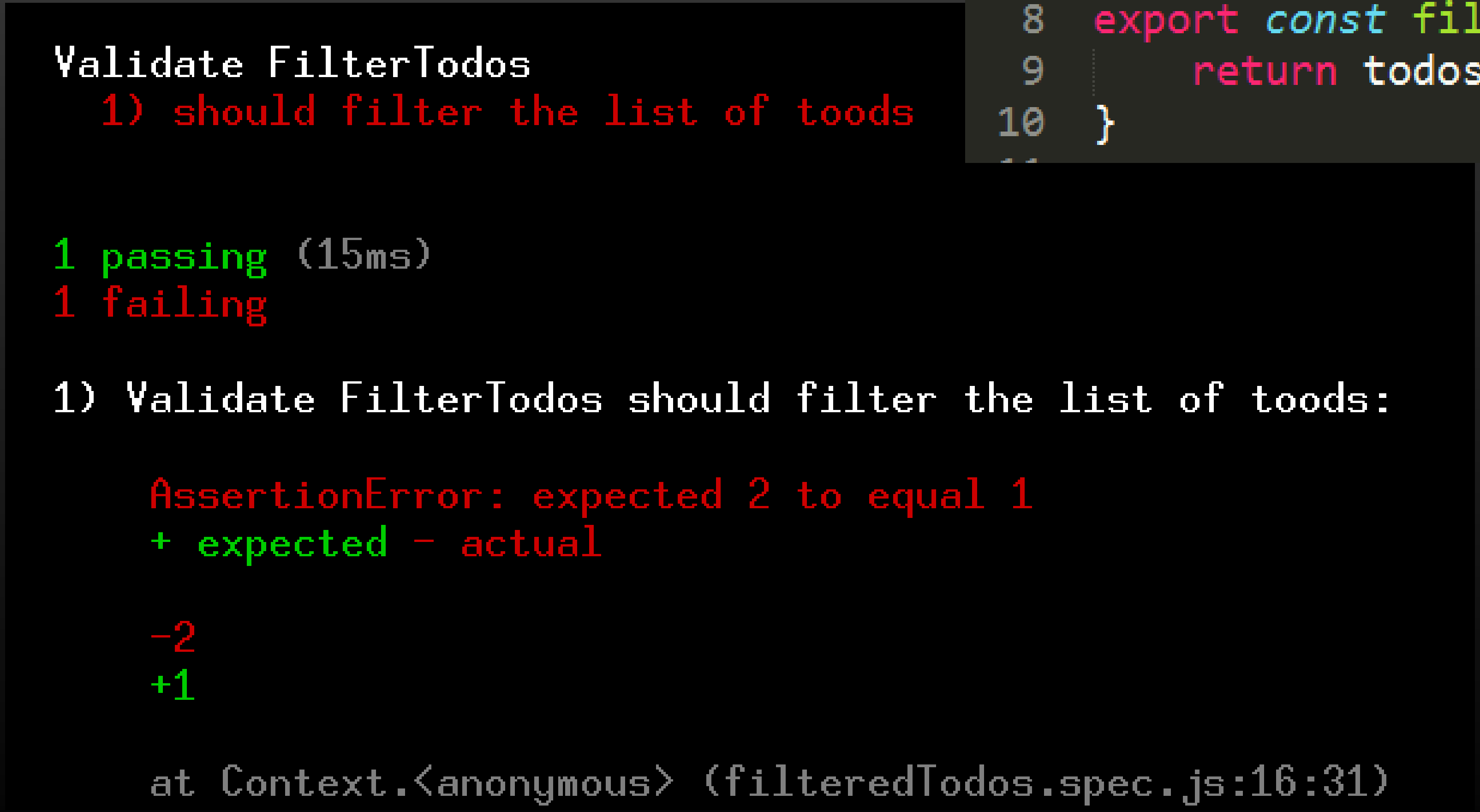
TDD

Tests initially fail



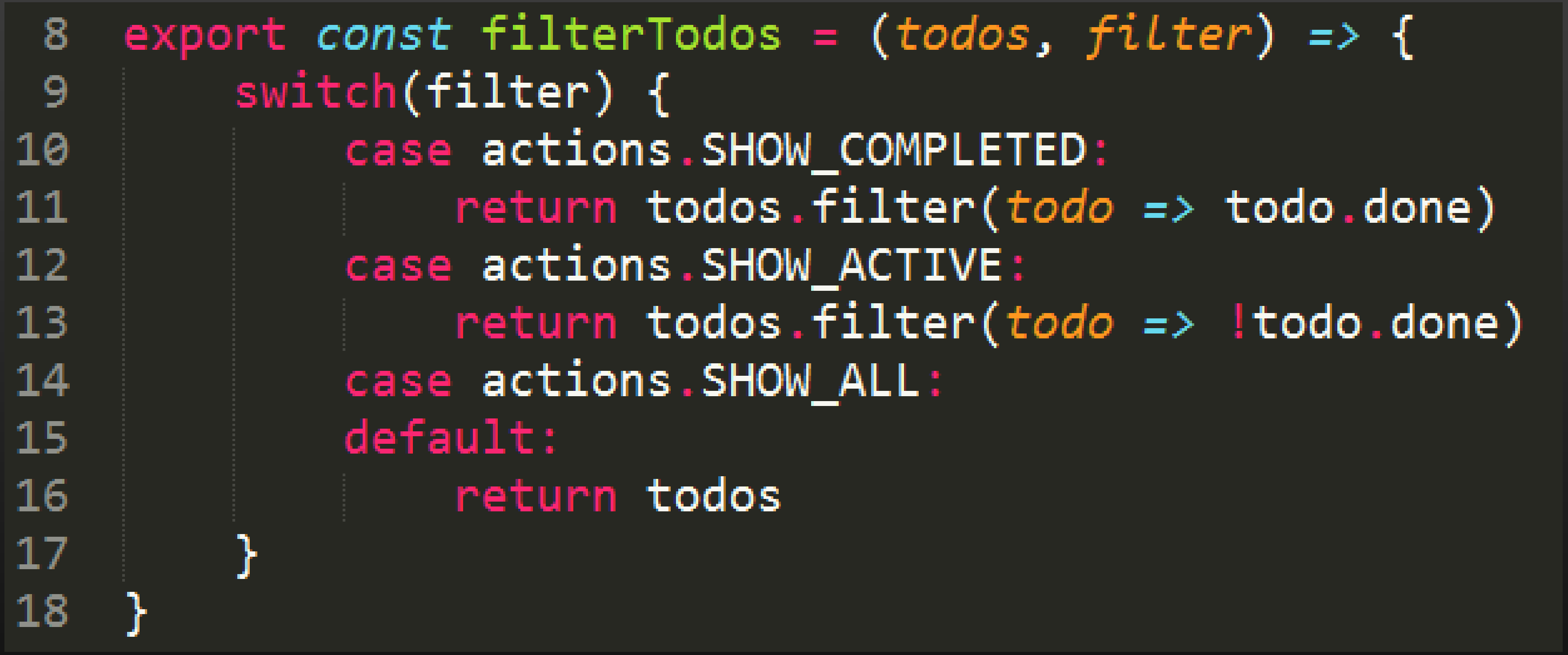
filteredTodos.spec.js × | filteredTodos.js ● | filterLink.spec.js × | filterLin

```
export const actions = {
    SHOW_ALL: 'SHOW_ALL',
    SHOW_COMPLETED: 'SHOW_COMPLETED',
    SHOW_ACTIVE: 'SHOW_ACTIVE'
}

export const filterTodos = (todos, filter) => {
    return todos
}
```

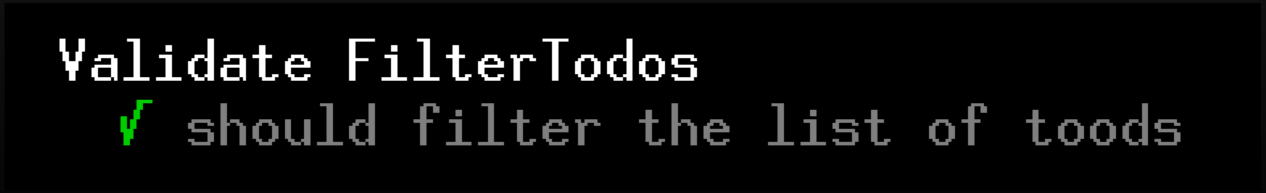


```
Validate FilterTodos
  1) should filter the list of toods


1 passing (15ms)
1 failing

1) Validate FilterTodos should filter the list of toods:

    AssertionError: expected 2 to equal 1
    + expected - actual

    -2
    +1

    at Context.<anonymous> (filteredTodos.spec.js:16:31)
```

Implement Functionality

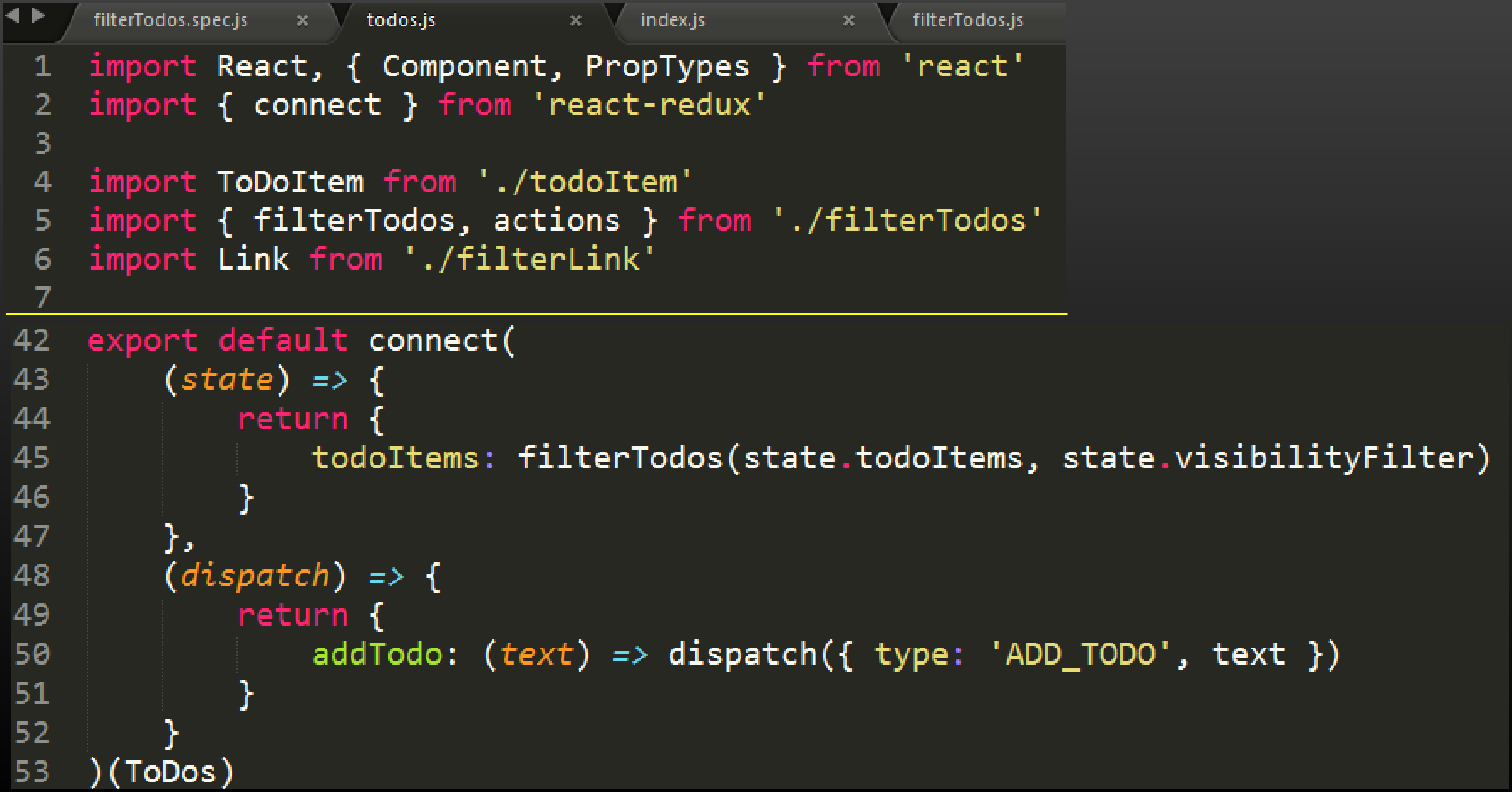


```
export const filterTodos = (todos, filter) => {
    switch(filter) {
        case actions.SHOW_COMPLETED:
            return todos.filter(todo => todo.done)
        case actions.SHOW_ACTIVE:
            return todos.filter(todo => !todo.done)
        case actions.SHOW_ALL:
        default:
            return todos
    }
}
```

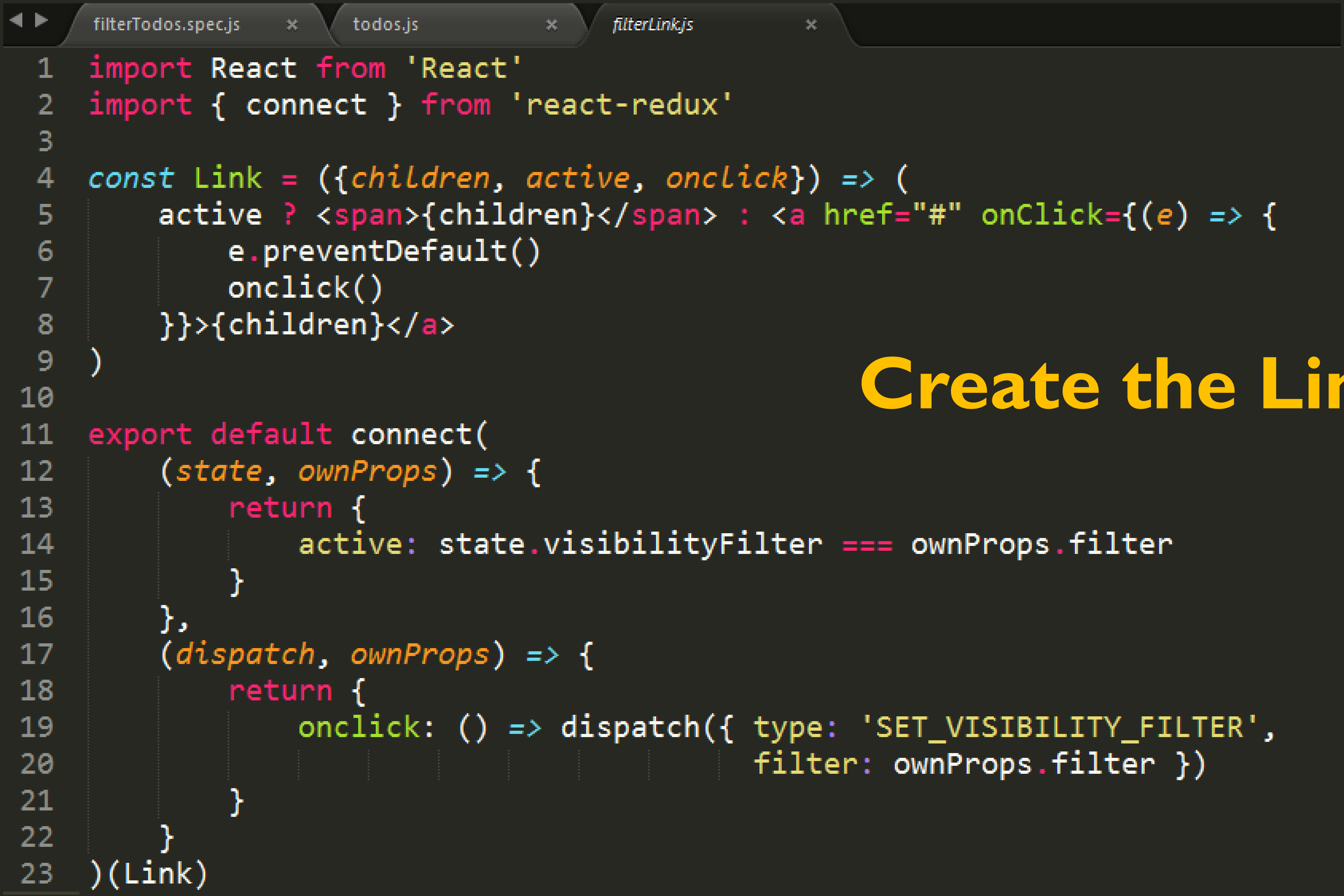


```
Validate FilterTodos
  √ should filter the list of toods
```

Update the view



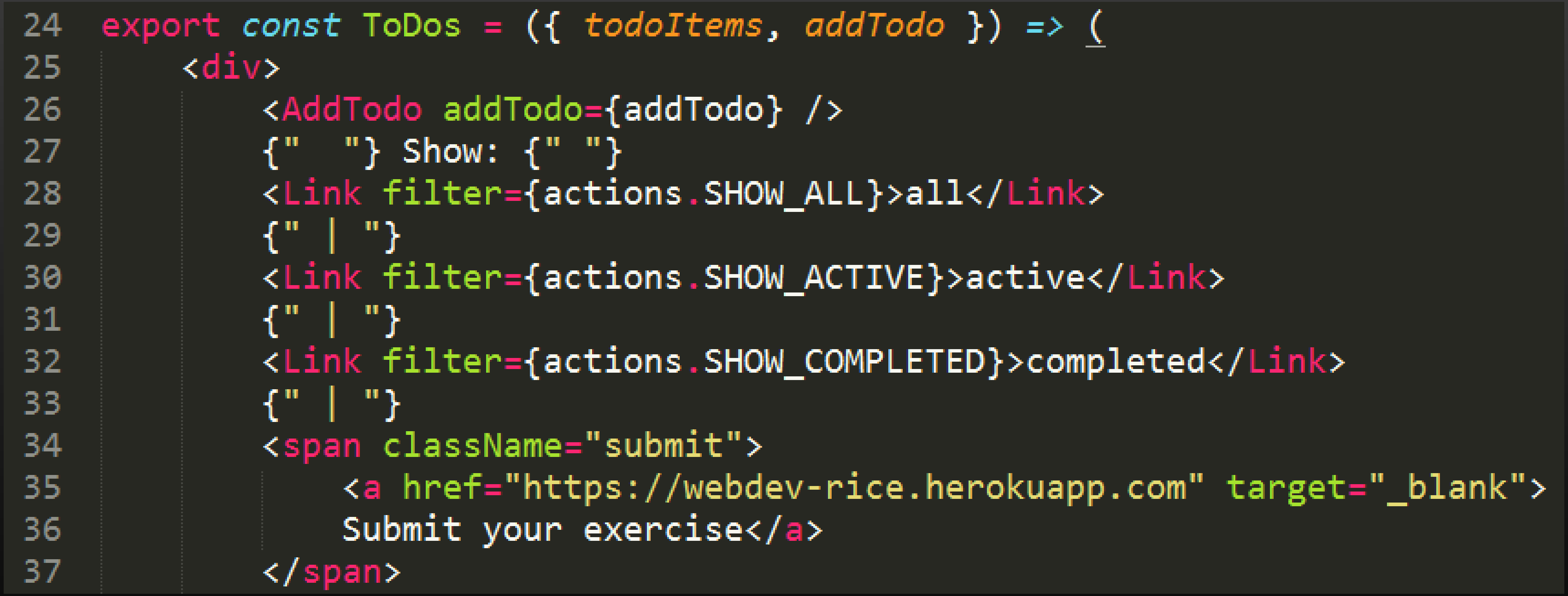
```
import React, { Component, PropTypes } from 'react'
import { connect } from 'react-redux'

import ToDoItem from './todoItem'
import { filterTodos, actions } from './filterTodos'
import Link from './filterLink'

export default connect(
    (state) => {
        return {
            todoItems: filterTodos(state.todoItems, state.visibilityFilter)
        }
    },
    (dispatch) => {
        return {
            addTodo: (text) => dispatch({ type: 'ADD_TODO', text })
        }
    }
)(ToDos)
```


filterTodos.spec.js × todos.js × filterLink.js ×

```
import React from 'React'
import { connect } from 'react-redux'

const Link = ({children, active, onclick}) => (
    active ? <span>{children}</span> : <a href="#" onClick={(e) => {
        e.preventDefault()
        onclick()
    }}>{children}</a>
)

export default connect(
    (state, ownProps) => {
        return {
            active: state.visibilityFilter === ownProps.filter
        }
    },
    (dispatch, ownProps) => {
        return {
            onclick: () => dispatch({ type: 'SET_VISIBILITY_FILTER',
                                    filter: ownProps.filter })
        }
    }
)(Link)
```

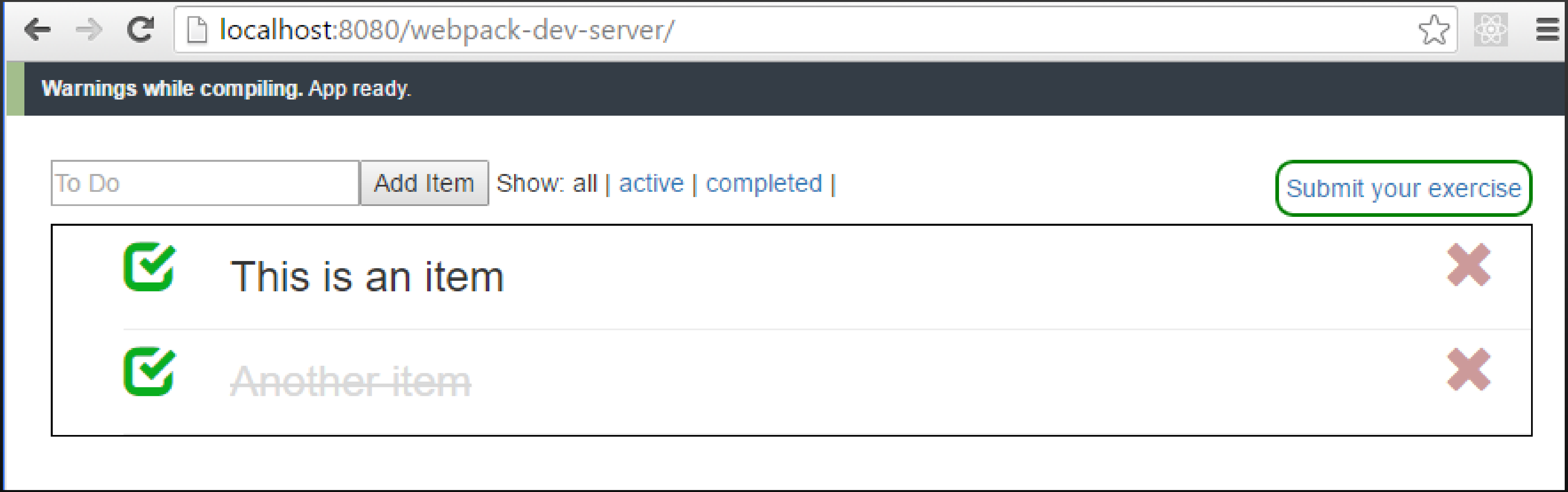
Create the Link

Add the Links to the View



```
export const ToDos = ({ todoItems, addTodo }) => (
    <div>
        <AddTodo addTodo={addTodo} />
        {"  "} Show: {" "}
        <Link filter={actions.SHOW_ALL}>all</Link>
        {" | "}
        <Link filter={actions.SHOW_ACTIVE}>active</Link>
        {" | "}
        <Link filter={actions.SHOW_COMPLETED}>completed</Link>
        {" | "}
        <span className="submit">
            <a href="https://webdev-rice.herokuapp.com" target="_blank">
            Submit your exercise</a>
        </span>
```

Final Product



localhost:8080/webpack-dev-server/

Warnings while compiling. App ready.

This is an item

~~Another item~~

In-Class Exercise: Frontend Testing

https://www.clear.rice.edu/comp431/sample/helloRedux-testing.zip

- Download and unpack the archive

```
npm install
```

- Implement the four tests in

  todoItem.spec.js

1. Display a todo (check proper rendering of class and content)
2. Toggle button functions
3. Remove button functions
4. Display a completed todo

/inclass-12/todoItem.spec.js

https://webdev-rice.herokuapp.com