前端单元测试(Vue)

1. 什么是单元测试

单元测试允许你将独立单元的代码进行隔离测试，其目的是为开发者提供对代码的信心。通过编写细致且有意义的测试。

为一个 Vue 应用做单元测试并没有和为其它类型的应用做测试有什么明显的区别。

1. 测试框架(Mocha + chai/Jest)

安装好Mocha后，使用npm run test指令，在终端进行单元测试，执行tests/unit下的所有.spec.js文件，在spec.js文件中引入相应模块，使用对应的js代码语句进行功能测试

基本语法：

describe（‘测试套件名称’,() => {

It(‘测试用例名称’, ()) => {

expect(‘测试相应的内容’).to.be.xxx(‘期望测试内容的值’)

}

}）

测试文件中应包括一个或多个describe，每个describe中可以有一个或多个it，每个it中可以有一个或多个expect，执行npm run test时，如果有测试用例未通过，则会给出报错提示和原因

expect就是判断源码的实际执行结果与预期结果是否一致，如果不一致就抛出一个错误。

1. Chai断言库

断言：一种判断代码预期结果与实际结果的代码，比如 a === b ===就是断言，chai常用的断言有3种expect should assert

expect(vm.$el.querySelector(‘.aa’).innerText()).to.be.equal(‘12’)

vm.$el.querySelector(‘.aa’).innerText().should.be.equal(‘12’)

expect(‘获取相应的内容’).to.be.断言内容(‘期望输出的值’)

Chai官网: <https://www.chaijs.com>

1. vue/test-utils

Vue.js官方的单元测试实用工具库，支持主流js测试运行器，是一个基于包裹器的API，同步触发事件

可以使用mount方法挂载对应组件,然后通过vm获取组件内容

import { mount } from '@vue/test-utils'

let wrapper = mount(Test)

console.log(wrapper.vm.num)

文档地址：<https://v1.test-utils.vuejs.org>

常用方法:

1. wrapper.setData(setProps)设置组件状态,或者在mount的时候使用参数添加数据 :

let wrapper = mount(Test, {

  propsData: {

    aProp: 'someValue'

  }

})

注意:操作dom的操作需要要在断言之前使用await异步处理,否则获取到操作前的值

1. wrapper.find(‘dom选择器’)
2. Wrapper.setData({a:’text’})设置组件的数据，只能修改已有的数据，不可给组件data添加数据
3. Wrapper.exists()判断一个内容是否存在(通常是dom节点)， expect(wrapper.exist()).to.be(true)
4. wrapper.text()获取文本

6.Wrapper.trigger():trigger带有一个可选的options对象。Options对象内的属性会被添加到事件上，trigger会返回一个promise，promise解决时，确保组件的内容已经更新