Unit Testing in JS

QUnit, Jasmin, Moca –framework

Test Driven Development

Behaviour Driven Dev

TDD – Qunit, JSUnit, Mocha

BDD – Jasmine, Mocha

Jasmine ! moje da e polezen za izpita

Mocha:

-bower/git download

-reporter – html reporter / karma reporter

-dw plugin for test syntax- chaiJS

-start writing

It-unit test

Opisanie na it - When empty array, expect to return 0

Describe(‘opisanie\_kakvo\_se\_testva’, func() {

It(‘opisanie\_kakvo\_se\_testva’, func(){

Var actual = sum([]);

Expect(actual).to.equal(0);

})

})

V htmla se referirat v tozi red

mocha.css

<body> < div id=’mocha’> tuka shte se podpuhvat unit testovete I infoto za tqh </div>

mocha js

chai js

mocha.setup(‘bdd‘);

expect = chai.expect;

myjsFile

myjsFiles.tests.js

<script> mocha.run();</script>

Reporters: visualize the results from unit tests

Karma: - can run in any browser env

Node js

Dw karma

* Install karma –g ---------global
* Dw karma command line interface
* Sled tova dw vsichki paketi nujni karma chai, karma chrome luncher, sus --save-dev sled jelaniq paket
* $ karma init and feel the needed data
* Start karma $ karma start

Npm install predi da pusnem demoto

Karma init v papkata na proekta

App/\*\*/\*.js

Test/\*\*/\*.js

Mocha with Chai

Chai Assertion

Assert.equal(var, ‘’)

Expect(var ).to.equal(value) 🡨------------

Var.should.equal(value);

JavaScript Pattern and Single-Page Applictaions

SammyJs / MustacheJs za domahna

SPA – web app

Architecture – server logic, client logic

Good performance, less traffic, reuse

* Server has to have good validation, router, moduli

SPA Application Architecture – building layers to meet separation of concerns

* UI layer
* Data layer
* Business layer

MV\* Patterns:

* MVC - controller
* MVVM - viewModel
* MVP – presenter

Model View Controller

Controller – contains business logic – it’s a middleman

Most of the JS frameworks implement the MVC pattern:

* AngularJS
* Backbone.js
* Ember.js
* Sammy.js
* Google Closure
* Batman.js

Model-View-ViewModel

viewModel – fires events

frameworks

* Kendo.UI
* Kendo.UI Mobile
* Knockout.js
* Knockback.js

Introduction to NodeJS – Exam Prep

For the exam:

Console 2 – Advanced console

NodeJS

* Google Chrome engine

Libuv – lib for events

V8 – google chrome – parse javascript

JavaScript > C++

Restoring Modules

NodeJS uses modules – packages

* Use npm to install package
* Use --save after package to save it , and after dw of the project and typing npm install will install all dependencies

Npm install

Npm runserver.js

Nice To Have:

SPA HW – the task Is just like the exam

Postman –REST client

chrome://apps

herokuapp – nodejs – free hosting