ENICARTHAGE
المدرسة الوطنية للمهندسين بقرطاج
Ecole Nationale d'Ingénieurs de Carthage

Testing an E-learning platform

Directed by:
zied Khayechi

Plan

Introduction

java Script

AngularJs

Unit test && End-to-End testing

Introduction :

° Full StackWeb Development

- ***Front-end and Back-end***

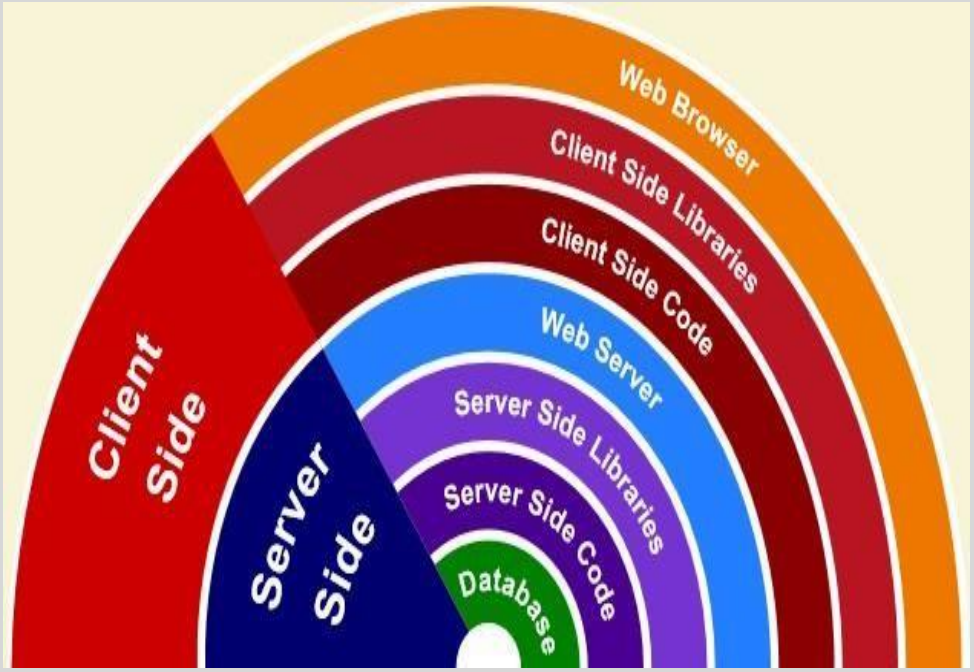
  - *Front end /Client-side*

    - HTML, CSS and Javascript

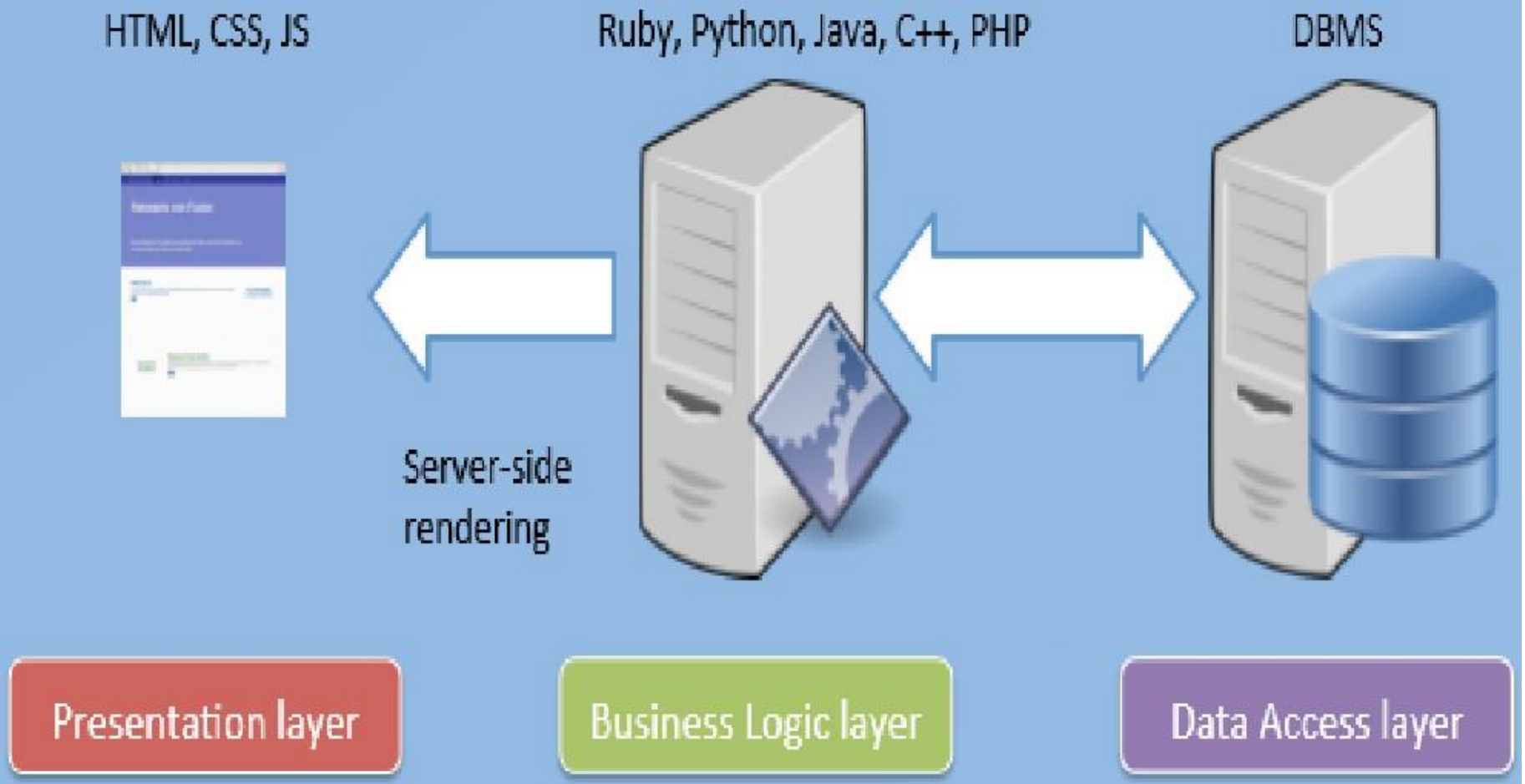
  - *Back end / Server-side*

    - Various technologies and approaches

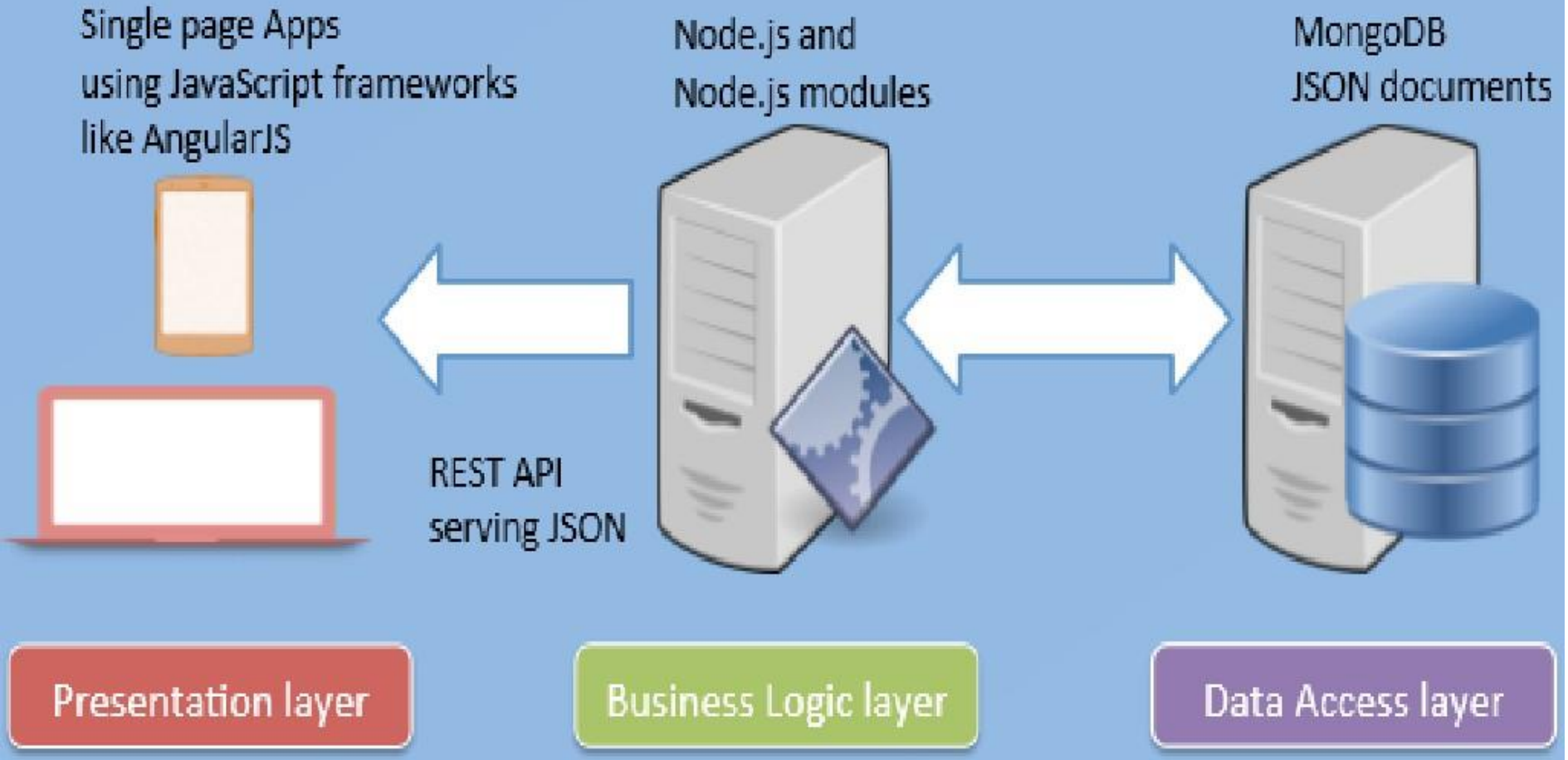
    - PHP, Java, ASP.NET, Ruby, Python



Traditional Web Development



Full Stack JavaScript Development



Full Stack Web Development

BaaS

Node.js Modules

Node.js

MongoDB

Presentation layer

Business Logic layer

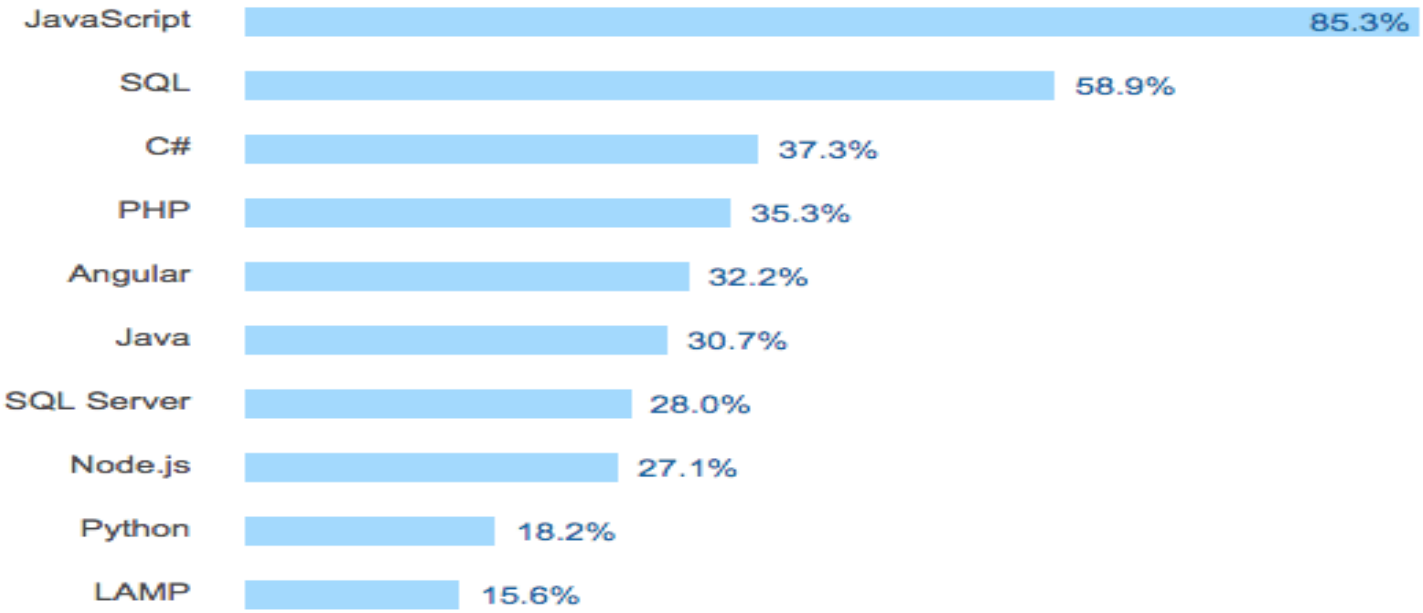
Data Access layer

java Script

JavaScript

JavaScript (not to be confused with Java) is an interpreted language with object oriented prototype. Mainly used in interactive web pages but also for servers with use (for example) of Node.JS It is mainly used on the client side web. That is, it is the browser that runs the code. Unlike the PHP or ASP style query languages that are run on the server side

Nowdays JavaScript is the most popular (most used by full stack developers)



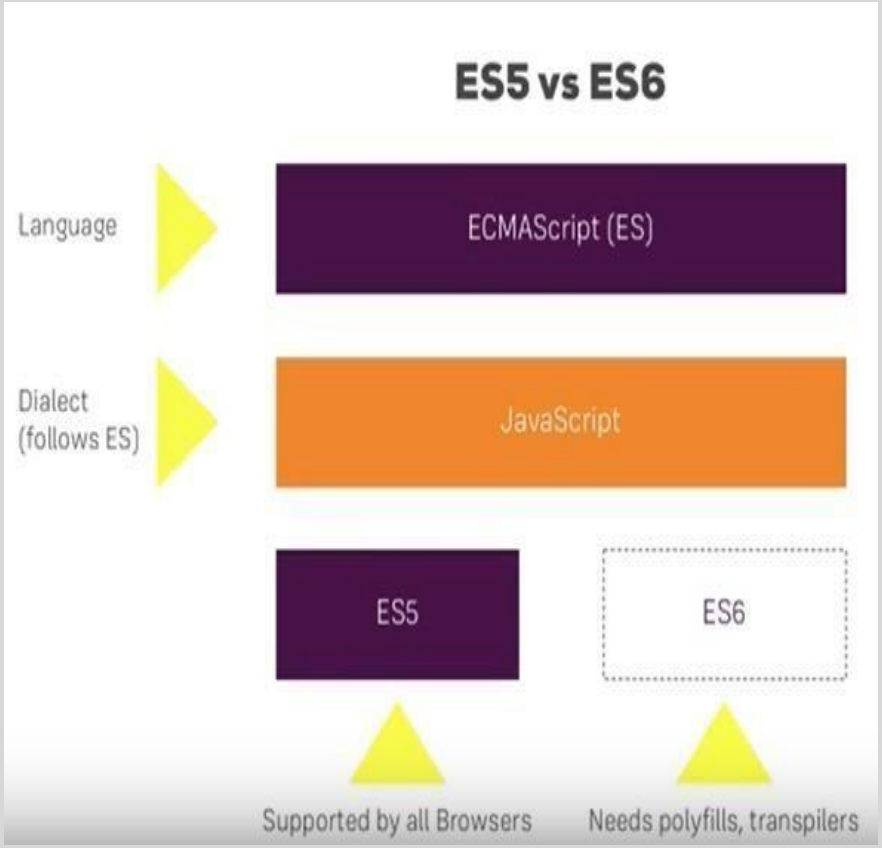
What is the relationship between JavaScript and ECMAScript?

JavaScript is ECMAScript or almost.

A bit of history does not hurt: Brendan Eich initially developed a scripting language server side, called LiveScript

Netscape and SUN work together to wear LiveScript on the browser. Thus released in 1995 a new version of the language, the first to be widely disseminated which is then called JavaScript.
Netscape submits his language to Ecma International to make a standard. The first drafts of the standard, Microsoft comes out JScript. Adobe takes them and creates ActionScript. The Standard is called ECMAScript. It deals with the language itself.

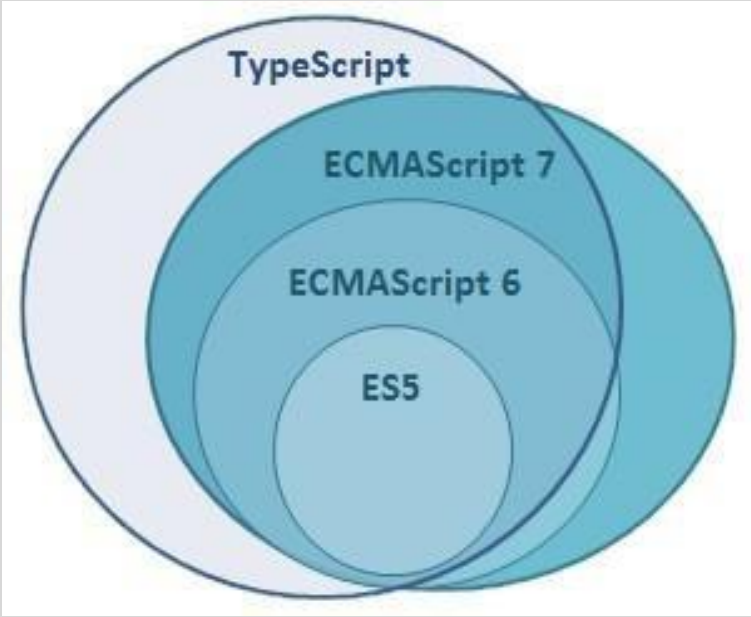
JavaScript, like all the combinations that have been born before the standard, since revised to conform.
JavaScript is an implementation of ECMAScript.



TypeScript

Type Script:
type script is typed superset of javaScript that compiles to plain javascript
➔ we need type script to compiles java script

(java , c++ ,c#): OOP➔classes ⟶ Function
inheritance ⟶ Prototype
…… …..
ES6 ⟶ ES5

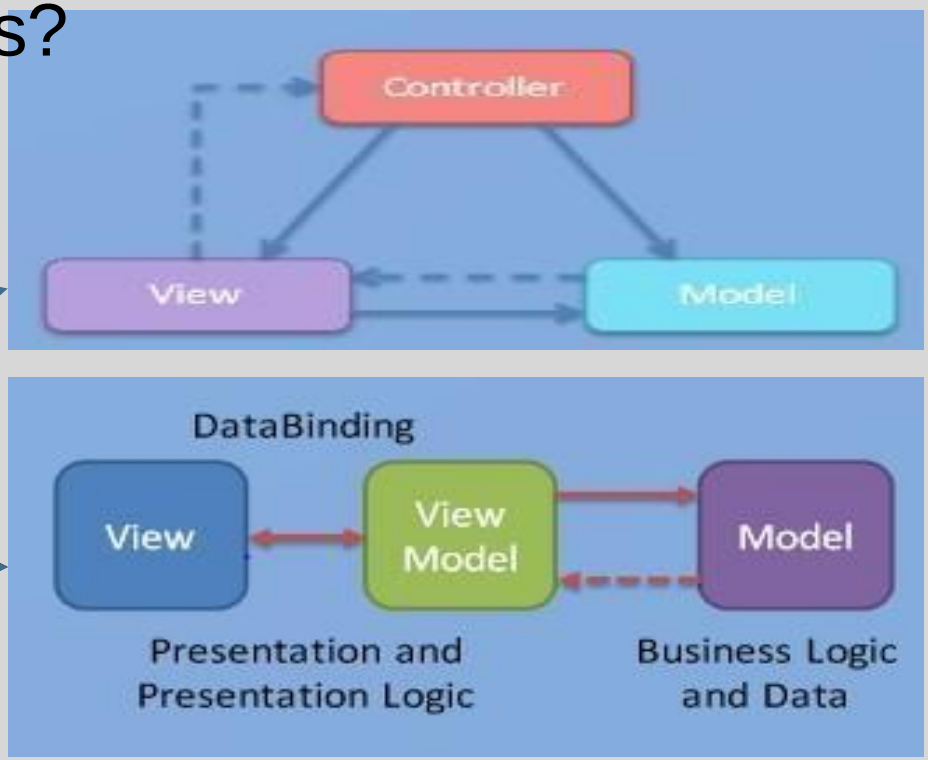


Why JavaScript Frameworks?

• Complexity of managing DOM manipulation and data updat manually • Well defined application architectures

– Model View Controller **MVC**
Model View View Model ~ MVVM
Model View Whatever

– Binding of model and view:
controllers, view , models



Software Library

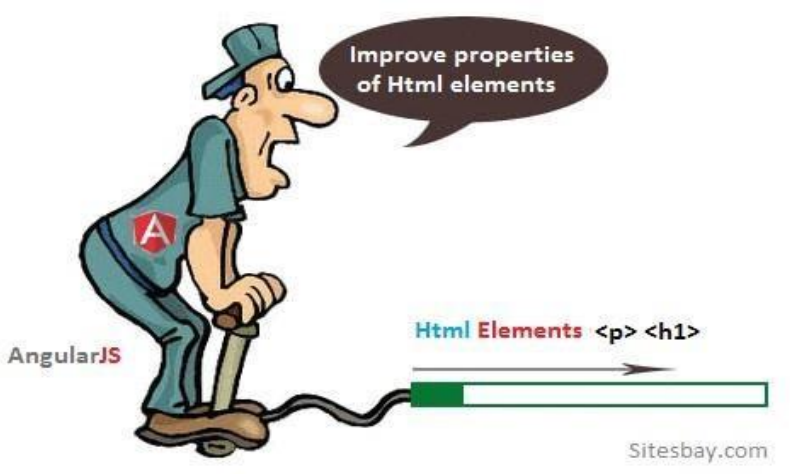
• Collection of implementations of behavior (well-defined interface)
• Reuse of behavior • Modularity • E.g., jQuery

But softwre Library don't support complexity ➔ turn to the software framework

Library	Framework
-collection of functions which are useful when writing web apps _code is in charge and it calls into the library when it sees fit. E.g.,jQuery - In control :call functions	-a particular implementation of a web application ➔ set of code -User ➔add more code ➔ Assume more control -Call function created when the app specify that ➔Hollywood Principe(Don't call us,we will call you!!) -Inversion of control : the users don't decide how the code is executed • Single Page Application – Rich InternetApplications • Model-View-Controller(MVC) – Data, binding,routing • Scalable, Reusable, Maintanable ➔ JS code • Test driven development

JavaScript Frameworks:

•Angular•Ember•Backbone•React•Aurelia•Meteor•Polymer•Knockout•Vue•Mercury

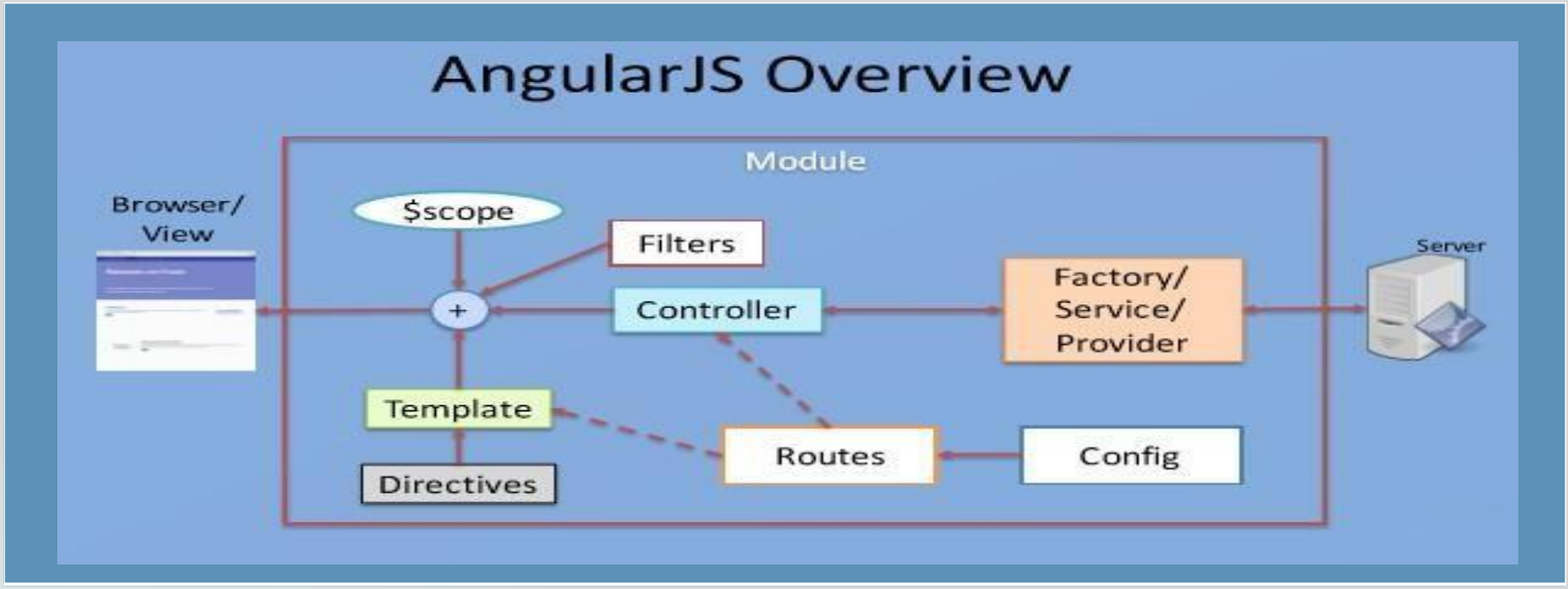


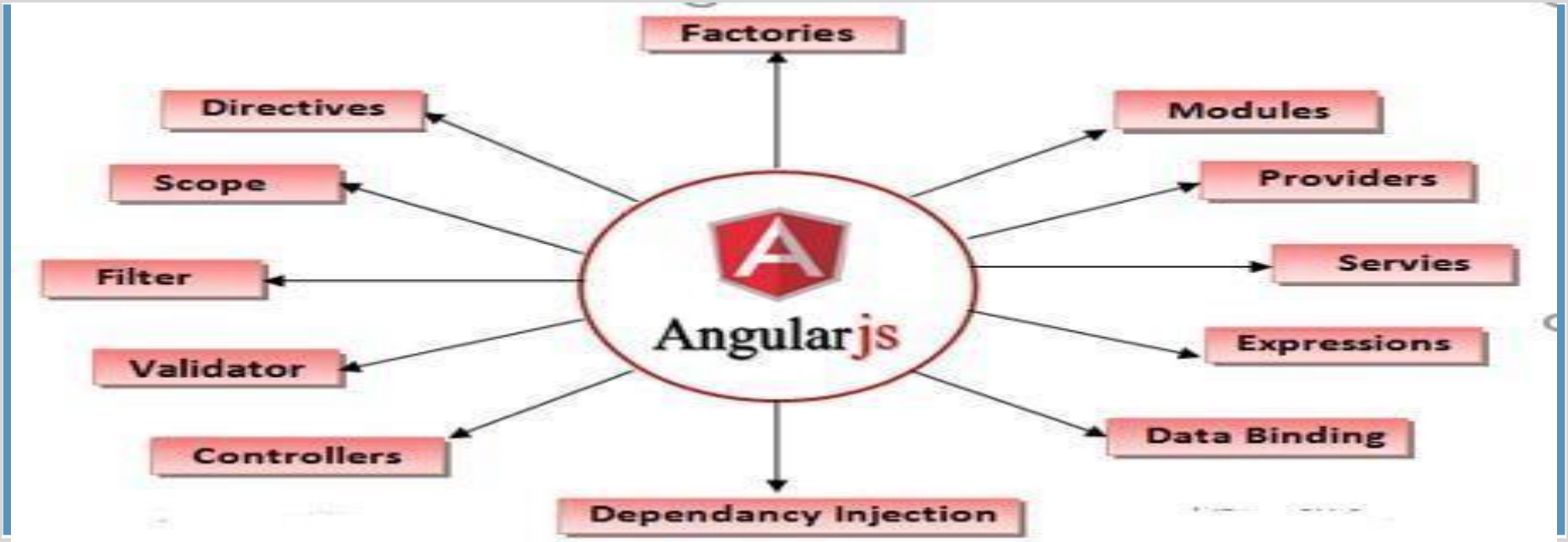
AngularJs

ANGULARJS
by Google

AngularJS :

Structural JavaScript framework for dynamic web applications:
– HTML only does static documents
– Angular fills in the gap to support dynamic applications
• Solving the impedance mismatch
• Designed with CRUD applications (data-driven) in mind
• Declarative approach





MVC

AngularJS is a framework of JavaScript which work on MVC model. MVC stand for Model View Controller. Model View Controller is a software design pattern for developing web application. It given software application into three interconnected parts; Model, View and Controller.

Dependency Injection

AngularJS has a built-in dependency injection subsystem that helps the developer by making the application easier to develop, understand, and test.

Validation

AngularJS provides client side validation same like JavaScript. Using AngularJS you can

create your own validation.

Filter

Filter are mainly used for modify the data. Filters can be added to expressions and directives using a pipe (|) character.

Directives

A directive is something that introduces new syntax. It improve the feature or functionality of html elements. Directives are markers on a DOM element which attach a special behavior to it. For example, static HTML does not know how to create and display a date picker widget. To teach HTML this new syntax we need a directive. AngularJS directives are extended HTML attributes with the prefix ng-.

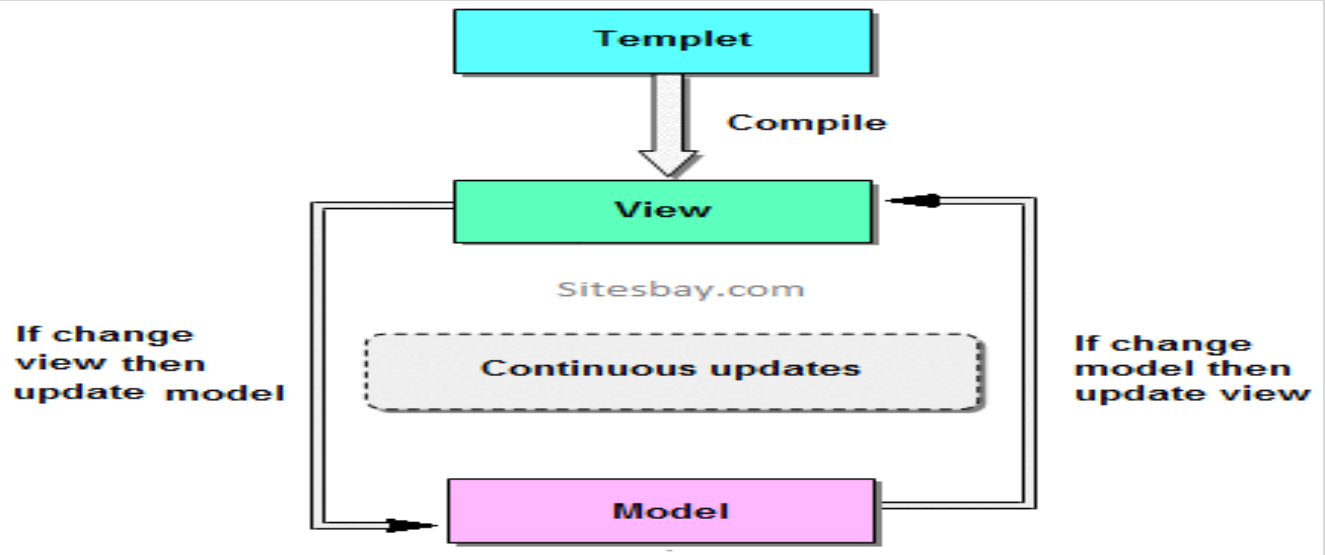
like ng-app, ng-init, ng-model and ng-repeat

Two Way Data-Binding

• Binding an HTML or CSS property to a JavaScript variable:

–When the value of the variable is updated, the HTML/CSS property is also updated

–Also works vice-versa with <input> fields

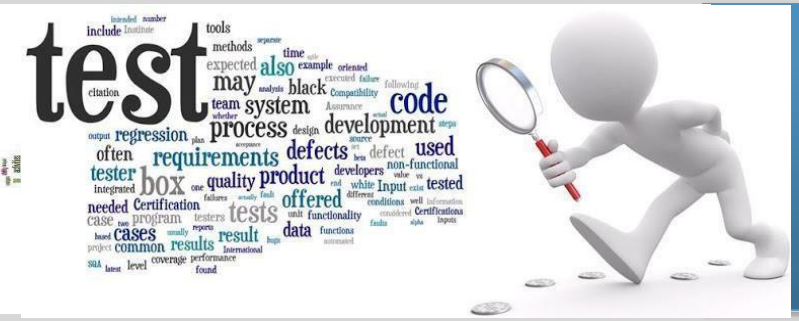


Testing Angular Applications :

- Angular (designed) ➔ facilitate testing –Modular
- Module implementation (controllers, filters, factories, services and providers)
  boost the fonctionnality .
- Dependency injection : separated Modules ➔ fonctionnality can be injected

Test-Driven Development

- Write an automated test case defining the desired functionality
- Write application code to pass the test
- Refactor the code to meet coding standards



Unit test
&&
End-to-End testing

Unit Testing

JavaScript is a dynamically typed language (great power of expressionno help from the compiler).

For this reason any code written in JavaScript needs to come with a set of tests.

➔features into AngularJS make testing easy .

Separation of Concerns

Unit Test ➔test individual units of code " isolate the unit of code under test "

we don't want to be forced into creating related pieces such as the DOM elements, or making any XHR calls to fetch the data to sort.
XHR :dependency injection ➔requests simulated
DOM: abstract ➔ testing the model without having to manipulate the DOM directly

Dependency Injection

AngularJS ➔dependency injection built-in ➔we can pass in a component's dependencies and stub or mock them.

without having to mess with any global variables that could inadvertently affect anothertest.

Additional tools for testing AngularJS applications

**Jasmine **Karma **angular-mocks

Jasmine

- Behavior driven development framework for JavaScript
  - –Adopted to test Angular applications
  - –Use *"describe"* function to group our tests
  - –Use *"it"* function to define individual tests

Jasmine Example :

```
describe('sorting the list of users', function() {
    it('sorts in descending order by default', function() { // your test assertion goes here }); });
```

Specs : defined by calling the global jasmine function it (string ,function)

➔contain one or more Expectation (true –false)

ALL

Passing Failed

Expect(….):take the actual value ,which is chained with a Matcher function (the expected value)

Matcher function

Boolean comparaison ,responsible to report to jasmine the value of the expectation

Jasmine has a rich of matched included :

expect{ . Tobe(….); .not.tobe(…); .toEqual(…); .tomatch (…); tobedefined (…); tobeunder (…); tobenull (…); tobetruthy (…);
. tobefalsy (…); tobelessthan (….); tobegreaterthan (…); tobecloseto (…); tocontain (…); tothrow (…); …….. etc }

Jasmine Example:

```
describe('Controller: MenuController', function (){
it('should create "dishes« with 2 dishesfetched from xhr',
function() {  expect(scope.showMenu).toBeTruthy();
expect(scope.dishes).toBeDefined();  expect(scope.dishes.length).toBe(2);
    });
});
```

Karma

- JavaScript based command line tool (NodeJS application)
  - –Spawns a web server to load your application's source code
  - –Executes your tests in the browsers (confident that the App works on all browsers you need to support)

Angular-mocks

Angular ngMockmodule provides mocking support for your tests

- –Inject and mock Angular services within unit tests
- –Make asynchronous modules execute synchronously to make it easier to execute tests
- –$httpBackend lets us mock XHR requests in tests

Angular Mocks Example

- // load the controller's module

  beforeEach(module('confusionApp'));

  beforeEach: before the test; module('confusionApp'): Angular Mocks Module

- var MenuController, scope, $httpBackend; → Variables
- // Initialize the controller and a mock scope

  beforeEach(inject(→ used in the test

  (function($controller,_$httpBackend_,$rootScope,menuFactory) {

- // place here mocked dependencies

  $httpBackend=_$httpBackend_; (test:ignored) (JS code)

  $httpBackend.expectGET('http://localhost:3000/dishes').respond([….]);

  'http://localhost:3000/dishes': URL

- scope=$rootScope.$new(); → new mock scope
- MenuController= $controller('MenuController',{

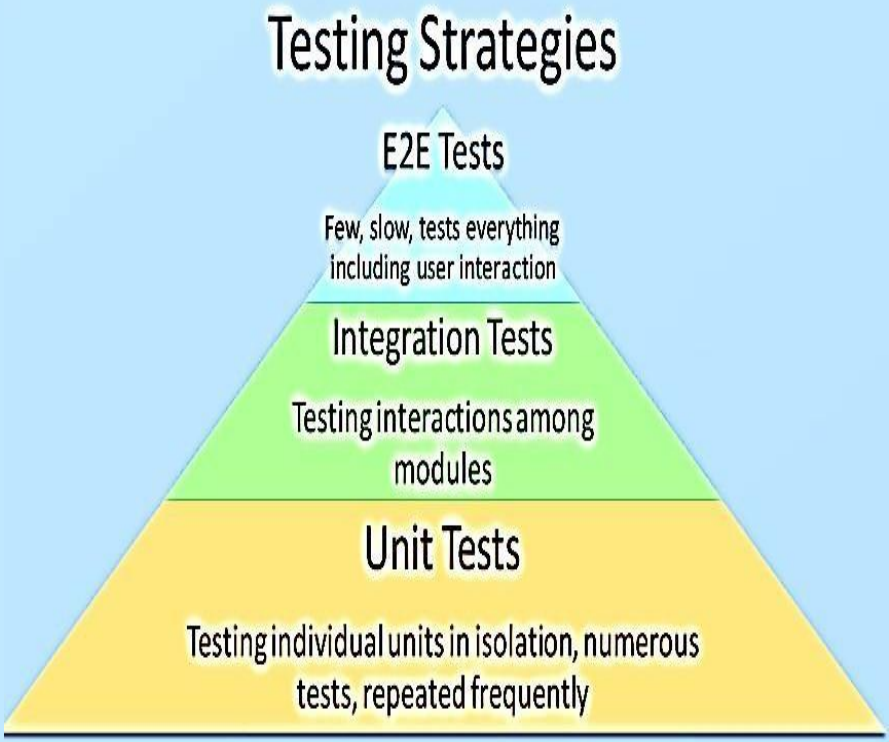
  $scope:scope , menuFactory:menuFactory });
- $httpBackend.flush(); → Data pushed to the server
- }));

End-to-End Testing

- Unit Testing is great for testing the units in isolation
  - Frequent repeated tests
  - Fast
  - Does not test the interaction among the units
  - Small test coverage scope
- Need integration && end-to-end tests
  - Covers large group of module interactions
  - Slow, so not repeated frequently

- Unit and IntegrationTests

_testing interaction between the controller, service and mock back end using $httpBackend



Protractor

- Node program that enables running of end-to-end tests
  - Runs tests against your application running in a browser and interacting with it like a real user

- Uses WebDriver to control browsers to carry out the tests
  - Selenium browser automation framework
  - Can use Direct Connect to test with Chrome and Firefox

- Uses Jasmine for expressing the test syntax

ProtractorConfiguration

```
exports.config= {
allScriptsTimeout: 11000,
specs:   ['e2e/*.js'],

capabilities:{'browserName':'chrome'},

baseUrl: 'http://localhost:3001/',
framework:        'jasmine',

directConnect:    true,
jasmineNodeOpts:
{defaultTimeoutInterval:    30000    }
};
```

Time to creat an error

Specification in test

Use the selinum driver

Acess App

Specifing the test

Chrome is directly connect

If the test fail

```
'use strict';
describe('conFusion App E2E Testing', function() {

    it('should automatically redirect to / when location hash/fragment is empty',
function() {
        browser.get('index.html');
        expect(browser.getLocationAbsUrl()).toMatch("/");
    });

});
```

Test the configuration

```
describe('menu 0 item', function() {
    beforeEach(function() {
        browser.get('index.html#/menu/0');
    });

    it('should have a name', function() {
        var name = element(by.binding('dish.name'));
        expect(name.getText()). toEqual('Uthapizza Hot $4.99');
    });
});
```

Carry out test : fetch Data

Extract elment

TEST THE ELEMENT

// Created by ZIED on 10/07/2017.

```
**********_configjs_**********
exports.config ={
   seleniumAddress: 'http://localhost:4444/wd/hub',
   specs: ['esp1.js'],
   allScriptsTimeout: 20000,

 jasmineNodeOpts: {
showColors: true,
defaultTimeoutInterval: 250000,
isVerbose: true
 }
};
**********____test____**********
describe('Add points ', function () {
   browser.driver.manage().window().maximize();
   it(' student exist ??', function () {
      browser.waitForAngularEnabled(false);
      browser.get('http://...../');
      browser.element(by.css('[name="username"]')).sendKeys('super@user.com');
```

```
browser.element(by.css('[name="password"]')).sendKeys('test1234');

browser.sleep( 1000 );

browser.element(by.css('[class="submit-row"]')).click();

var count= 0;

element(by.xpath('//*[@id="changelist-form"]/div[1]/span')).getAttribute('data-actions-icnt')

.then(nbr=>

{ console.log('this is the number', nbr);

   count=nbr;
    var i = 1 ;

    var test =true;

    while( i <= count && test == true )

           {

      var k = i.toString();

      var str1 ='//*[@id="result_list"]/tbody/tr[';

      var str2 =k;

      str3 = ']/th/a';

      const res = str1.concat(str2, str3);

      console.log ( res);

       element(by.xpath( res )).getText().then( elt =>

      {    console.log ( elt);
            var email ='s_____@__.com';

           if ( elt === email ) {
```

```
                browser.element(by.xpath( res )).click();
                browser.sleep( 1000 );
                 ok = true ;
                console.log('show me',ok);
                browser.element(by.xpath('//*[@id="id_points"]')).clear();
                browser.element(by.xpath('//*[@id="id_points"]')).sendKeys(1000);
                browser.sleep(4000);
                element(by.xpath('//*[@id="student_form"]/div/div/input[1]')).click();
                browser.sleep(2000);
              console.log('ok!!!!');
              var test = false;
            } else {  console.log(" no student has this email !! "); }
          });
          i++ ;
        }
      //expect(count).toBe('100');
    });
  });
});
```

thank you for your attention