

Desarrollo Web Front-End con AngularJS

¿Qué es AngularJS?

- Es un framework JavaScript cliente-servidor para agregar interactividad al documento HTML
- Es una mejor alternativa a jQuery para manipular datos del documento HTML
- Es parte del stack (tendencia) MEAN (**M**ongoDB/**M**ongoose + **E**xpressJS + **A**ngularJS + **N**odeJS)
- Para agregarlo en nuestro proyecto:

```
<script  
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.5.8/angu  
lar.min.js"></script>
```

¿Porqué AngularJS?

- Nos ayuda a organizar nuestro código JavaScript
- Nos ayuda a crear sitios responsivos de manera rápida
- Extiende HTML con nuevos atributos
- Es ideal para Single Page Applications (SPAs) y CRUD (90% apps)
- Es compatible con jQuery
- Es fácil de aprender
- Es fácil de testear ([Jasmine](#) y [Karma](#))

¿Porqué AngularJS?

- Estructurado. Aporta calidad y organización
- Liviano (<36KB comprimido y minificado)
- Free
- Modularidad
- Extensible y Mantenable
- Componentes reutilizables
- Two-way Data Binding – El modelo como fuente simple de confianza

Ejemplo con jQuery (no data binding)

HTML Fragment

```
<form>
  <p>
    Enter your name:
    <input type="text" id="name" required>
    <button type="button"
      disabled="disabled">Submit</button>
  </p>
  <p id="greet" style="display: none">
    Hello,
    <span id="name-repeat"></span>
  </p>
</form>
```

JavaScript Fragment

```
var name = $('#name'),
    nameRepeat = $('#name-repeat'),
    greet = $("#greet"),
    submit = $("button");

name.bind('keyup', function() {
  var disabled, value;
  value = name.val();
  if (value.length === 0) {
    submit.attr('disabled', 'disabled');
    greet.css('display', 'none');
  } else {
    submit.removeAttr('disabled');
    greet.css('display', 'block');
  }
  nameRepeat.innerHTML = value;
});

submit.bind('click', function() {
  $.ajax('/service?name=' + name.val());
});
```

Ejemplo con AngularJS

HTML Fragment

```
<form>
  <p>
    Enter your name:
    <input type="text" ng-model="name" required>
    <button type="button" ng-click="submit()"
      ng-disabled="!name">Submit</button>
  </p>
  <p ng-show="name">
    Hello, {{name}}
  </p>
</form>
```

JavaScript Fragment

```
$scope.submit = function() {
  $http.get('/service?name=' + $scope.name);
};
```

Hola Mundo!

jQuery

```
<p id="greeting2"></p>
<script>
$(function() {
  $('#greeting2').text('Hello World!');
});
</script>
```

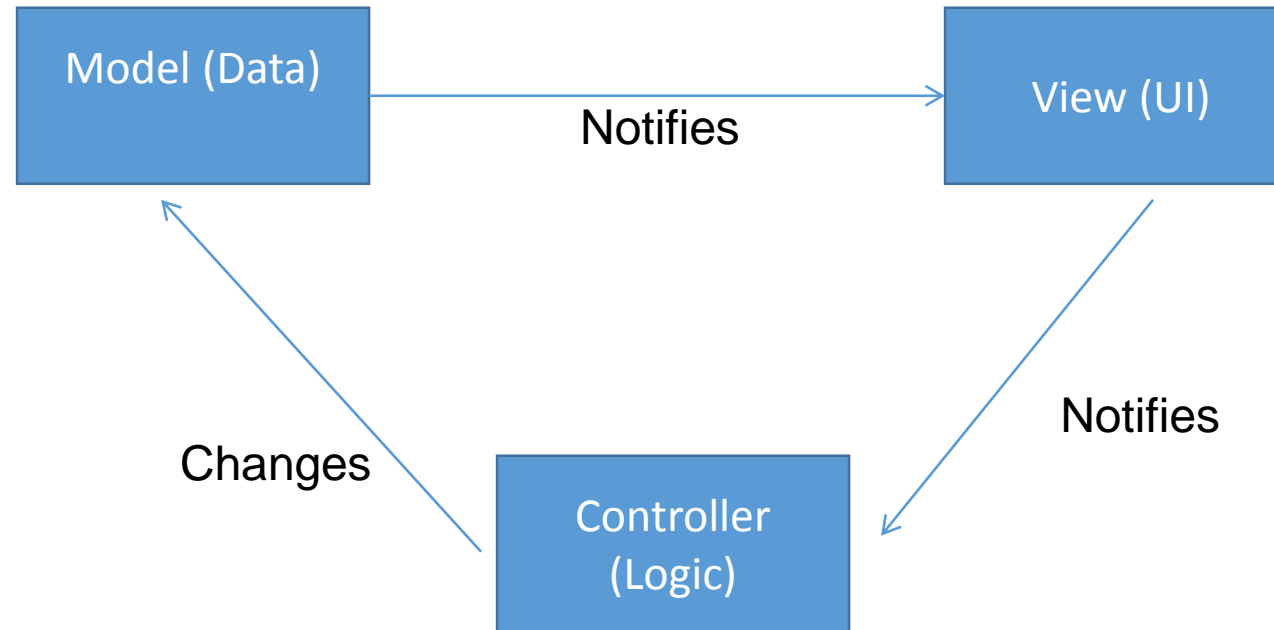
AngularJS

```
<p ng:init="greeting = 'Hello
World!'">{{greeting}}</p>
```

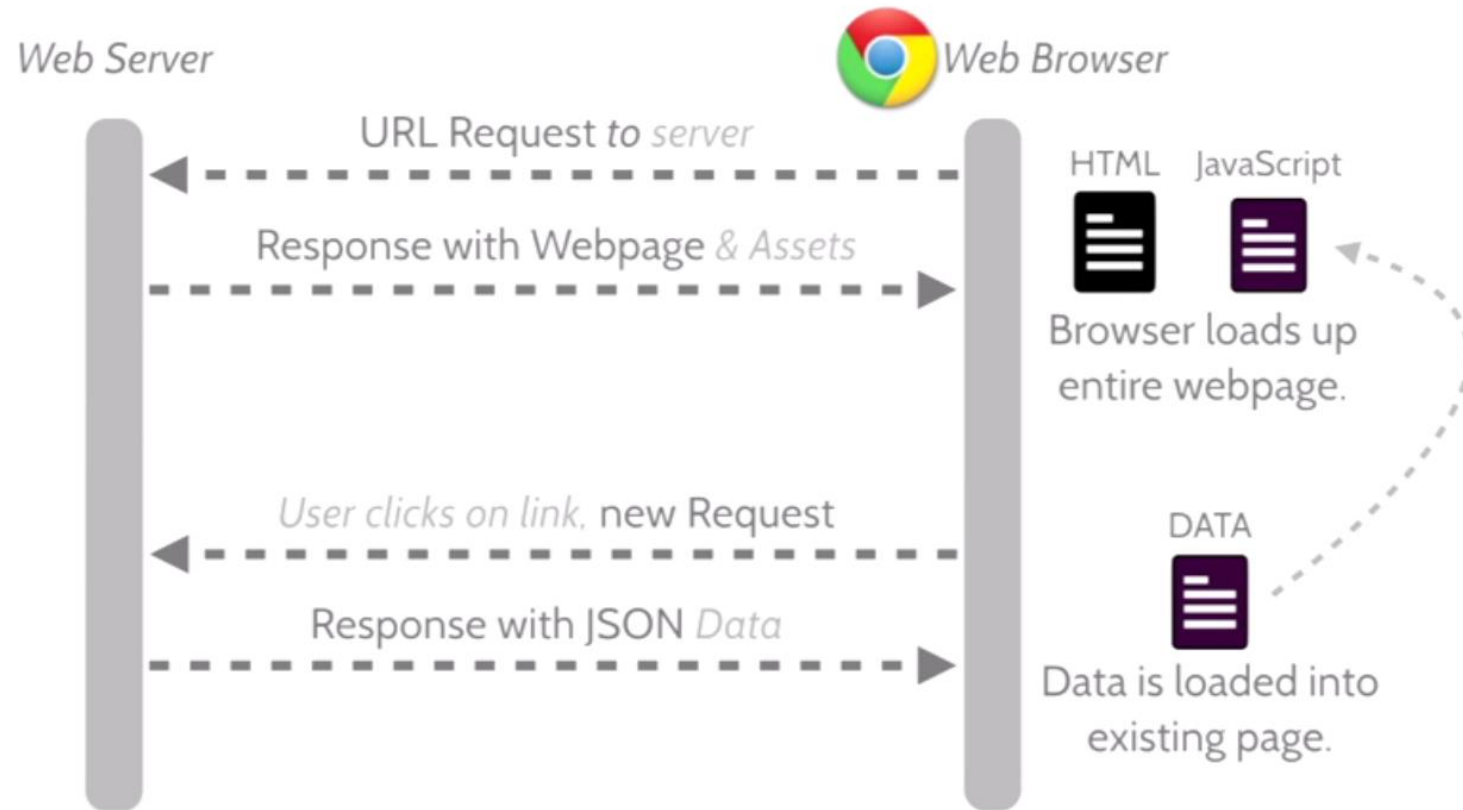
Model View Controller

- ¿Qué es MVC?
 - Model – los datos
 - View – la interfaz de usuario, lo que el usuario ve e interactúa
 - Controller – la interfaz entre el modelo y la vista
- El modelo no es necesariamente datos de la base de datos
- Utilizaremos MVC en el browser

Model View Controller



¿Cómo trabaja AngularJS?



¿Qué es el Data Binding?

- El Data-binding en aplicaciones AngularJS es la sincronización automática de los datos entre el modelo(Model) y la vista (View)
(<https://docs.angularjs.org/guide/databinding>)

```
<form>
  <p>
    Enter your name:
    <input type="text" ng-model="name" required>
    <button type="button" ng-click="submit()" ng-disabled="!name">Submit</button>
  </p>
  <p ng-show="name">
    Hello, {{name}}
  </p>
</form>
```

Expressions

- Angular expressions son trozos de código tipo JavaScript que son usualmente ubicados de la siguiente forma `{{ expression }}`. (<https://docs.angularjs.org/guide/expression>)

```
<form>
  <p>
    Enter your name:
    <input type="text" ng-model="name" required>
    <button type="button" ng-click="submit()" ng-disabled="!name">Submit</button>
  </p>
  <p ng-show="name">
    Hello, {{name}}
  </p>
</form>
```

Scope

- Es un objeto con las propiedades y metodos disponibles
- Es el vínculo entre el HTML (View) y el Javascript (Controller)
- Está disponible tanto en la vista como en el controlador

```
function foo() {  
    var name = "John";  
  
    function hello() {  
        var name = "Jack";  
        return "Hello, " + name;  
    }  
  
    function goodbye() {  
        return "Good bye, " + name;  
    }  
  
    hello();    //returns "Hello, Jack"  
    goodbye(); //returns "Good bye, John";  
}
```

Directive

- AngularJS nos permite extender nuestro HTML con nuevos atributos y nuevos tags llamados **Directives**
- AngularJS tiene un conjunto de directivas incorporadas que ofrecen funcionalidad a nuestras aplicaciones

```
<form>
  <p>
    Enter your name:
    <input type="text" ng-model="name" required>
    <button type="button" ng-click="submit()" ng-disabled="!name">Submit</button>
  </p>
  <p ng-show="name">
    Hello, {{name}}
  </p>
</form>
```

Directive Naming

- Cuando definimos directivas in JavaScript el nombre deber ser en formato camelCase

```
module.directive('myDirective', [function() { ... }]);
```

- Cuando activamos directivas utilizamos la forma lowercase:

```
<my-directive></my-directive>
```

```
<div my-directive></div>
```

- AngularJS normaliza el HTML para encontrar las directivas por:
`<div data-my-directive></div>` is recognized as the directive **myDirective**

Service

- ¿Qué es un servicio?
 - El suministro del proveedor de utilidades y comodidades como el agua, la electricidad, el gas, etc. requeridas y demandas por el publico.
- En programación... ¿Qué es un servicio?
 - El suministro del proveedor de utilidades y comodidades como funciones, valores, objetos requeridas y demandas por una expression
- En AngularJS podemos definir servicios que provean funcionalidades para ser utilizadas repetidamente en nuestro código


Dependency Injection

Cuando una función es dependiente de un dato o funcionalidad, esa función debe ser accessible.

- Option 1: acceso global

```
var a = 4, b = 2;

function divide() {
  return a / b;
}
```



- Option 2: Usando dependency injection

```
function divide(a, b) {
  return a / b;
}
```

Dependency Injection

- La inyección de dependencia requiere datos específicos por cada parámetro.
- No podemos utilizar la función divide como se muestra a continuación:

```
function divide(a, b) {  
    return a / b;  
}  
  
divide('hello', { x: 2 });
```

Modules

- ¿Qué es un módulo?
 - Es un **contenedor de código** para las diferentes partes de nuestras aplicaciones
- Un modulo es usado para definir servicios que son reusables tanto por el document HTML como por otros módulos.
- Contienen elementos como:
 - Controller
 - Directive
 - Constant, Value
 - Factory, Provider, Service
 - Filter
- **Buena práctica:** Dividir nuestro código en módulos con funcionalidades diferentes. “Divide et impera”.

Module Definition

- Definir un módulo:

```
var module = angular.module('myModule', []);
```

- Definir un modulo con dependencias de otro módulos:

```
var module = angular.module('myModule', ['otherModule']);
```

- Obtener un modulo existente:

```
var module = angular.module('myModule');
```

Application Module

AngularJS prove un camino para vincular nuestro modulo principal con el document HTML utilizando la directive **ng-app**

- HTML fragment
- Javascript fragment

```
<div ng-app='myApp'>  
  ...  
</div>
```

```
angular.module('myApp', []);
```

Controller Definition and Assignment

Facilitan la comunicación entre el modelo y la vista

Los controllers vinculan el modelo con la vista utilizando el servicio de AngularJS: \$scope

- HTML Fragment

```
<div ng-app='myApp' ng-controller="myController">
  <p>Hello, {{name}}!</p>
  <p>{{greet()}}</p>
</div>
```

- JavaScript fragment

```
var module = angular.module('myApp', []);

module.controller('myController', [
  '$scope', function($scope) {

    $scope.name = 'John Smith';

    $scope.greet = function() {
      return 'Hello, ' + $scope.name + '!';
    };

  }]);
```

Filters

- ¿Qué es un filtro?:
 - Un filtro formatea el valor de una expresión para mostrar al usuario
- Los filtros pueden ser utilizados en HTML mediante la notación por barra o en JavaScript inyectando el servicio \$filter

Filters

- HTML Example

```
<div ng-app='myApp' ng-controller="myController">
  <p>{{name | uppercase}}</p>
  <p>{{uppercaseName()}}</p>
</div>
```

- JavaScript Example

```
var module = angular.module('myApp', []);

module.controller('myController', [
  '$scope', '$filter', function($scope, $filter) {

    $scope.name = 'John Smith';

    $scope.uppercaseName = function() {
      return $filter('uppercase')($scope.name);
    };

  }]);
```


Filters with Parameters

- HTML Example

```
{{ expression | filterName : param1 : param2 }}
```

- JavaScript example

```
$filter('filterName')(expression, param1, param2);
```

Core Filters

- AngularJS tiene varios filtros incorporados:

- currency
- date
- filter
- json
- limitTo
- lowercase
- number
- orderBy
- uppercase

```
{{ '2015-03-19T19:00:00.000Z' | date : 'MMMM yyyy' }}
```

```
$filter('date')(new Date(), 'MMMM yyyy');
```

Service

- En AngularJS, un servicio es una función, o un objeto, que está disponible y limitado a nuestra aplicación AngularJS
- El servicio genera un singleton de un objeto instanciado

```
//define a service
myModule.service('person', [function() {
  this.first = 'John';

  this.last = 'Jones';

  this.name = function() {
    return this.first + ' ' + this.last;
  };
}]);

//inject the person service
myModule.controller('myController', ['$scope', 'person', function($scope, person) {
  $scope.name = person.name();
}]);
```

Service

- AngularJS tiene cerca de 30 servicios incorporados. Algunos de ellos:
 - `$location`.
 - `$timeout`
 - `$http`
 - `$interval`
- AngularJS supervisa constantemente tu aplicación, y para manejar los cambios y eventos de manera apropiada, AngularJS prefiere `$location` en lugar de `window.location`

Routes

- \$routeProvider – used for dealing with routes

Modified app.js

```
angular.module('F1FeederApp', [  
  'F1FeederApp.services',  
  'F1FeederApp.controllers',  
  'ngRoute'  
]).  
config(['$routeProvider', function($routeProvider) {  
  $routeProvider.  
    when("/drivers", {templateUrl: "partials/drivers.html", controller: "driversController"}).  
    when("/drivers/:id", {templateUrl: "partials/driver.html", controller: "driverController"}).  
    otherwise({redirectTo: '/drivers'});  
}]);
```

Your First App

- HTML Document

```
<!doctype html>
<html lang="en" ng-app="app">
  <head>
    <title>App Title</title>
    <link rel="stylesheet" href="css/styles.css">
  </head>
  <body>
    <script src="js/angular.js"></script>
    <script src="js/app.js"></script>
    <script src="js/moduleA.js"></script>
  </body>
</html>
```

- APP.js

```
var module = angular.module('app', ['moduleA']);
```