

## Agenda: Angular JS

sábado, 16 de septiembre de 2017 18:33

The graphic features a large teal 'U' shape on the right side. Inside the 'U', there's a white rectangular area containing text and logos. At the top of this area are four teal rounded squares of varying sizes. Below them is a logo for 'ESCUELA: TECNOLOGÍA' featuring a stylized gear icon. The main title 'DESARROLLO WEB AVANZADO CON ANGULAR JS' is centered in bold black text. Below the title is the date '18, 19 y 20 de Septiembre de 2017'. At the bottom left is the 'indra' logo with a circular icon. At the bottom right is the 'icono3 TRAINING CONSULTING' logo.

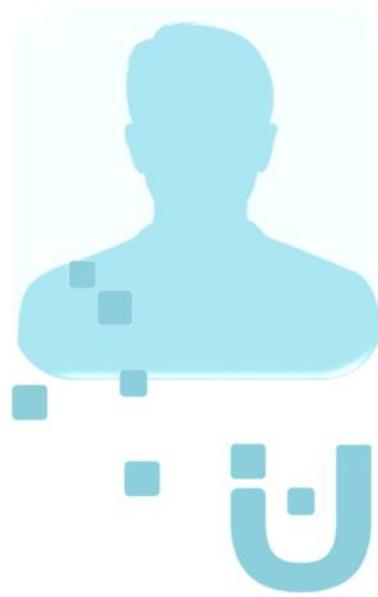
Duración: 24 horas

## ÍNDICE

Objetivo: Aportar a los alumnos reglas, principios, documentación y ejemplos suficientes para que puedan incorporar AngularJS en sus desarrollos.

- Fundamentos de AngularJS
- Creación de Proyectos para AngularJS
- Directivas y Plantillas (Vistas)
- Extendiendo el DOM
- Formularios y validaciones
- Servicios en AngularJS

Requisitos: Tener experiencia en proyectos de desarrollo de Software. Conocimientos básicos de lenguajes HTML, CSS y fundamentos de lenguaje JavaScript.



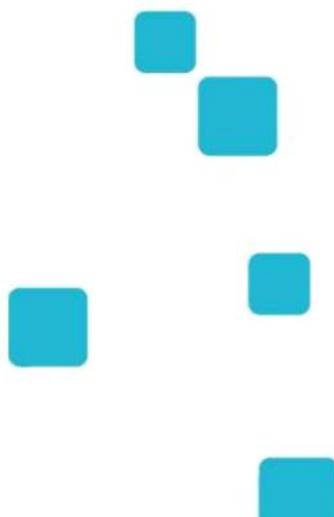
- NOMBRE APELLIDO PROFESOR  
Alejandro Cerezo

- VER PERFIL COMPLETO:

- [linkedin.com/company/icono-training-Consulting](https://linkedin.com/company/icono-training-Consulting)
- [linkedin.com/in/alejandrocerezo](https://linkedin.com/in/alejandrocerezo)

- CONTACTO

- [training@iconotc.com](mailto:training@iconotc.com)
- [alce65@hotmail.es](mailto:alce65@hotmail.es)



Avda. de Bruselas 35  
28108 Alcobendas,  
Madrid España  
T +34 91 480 50 00  
F +34 91 480 50 80  
[www.indracompany.com](http://www.indracompany.com)

## Desarrollo Avanzado Web con AngularJS

### Fundamentos de AngularJS

### Tecnologías implicadas

- Frameworks en JS
- Presentación de AngularJS
  - Versiones de AngularJS. AngularJS 1.5. Angular 2.x - 4.x
  - Líneas maestras. MVC SPA
  - Elementos de AngularJS
    - Vista y controlador. Doble *binding*
    - Módulos y estilos
  - Evolución: Angular 1.5. Componentes
- Entorno de trabajo
  - Navegadores y Servidor Web
  - Editores de código
  - Gestión del proyecto
    - Instalaciones: Node, npm
    - Versiones : Git, GitHub
    - Arquitectura () y despliegue
- ECMAScript 6 (ES6)
  - Funciones Arrow, Clases, Promises

### Creación de Proyectos para AngularJS

#### Desarrollo imperativo

- Módulos
- Modularidad y estilos
- Controladores
- Componentes

#### Arquitectura de proyectos

Estructura adecuada de un proyecto basado en Angular

- **Módulos (Modules)**
- *components*
- *Services*
- *Mocks*

- **MVC y \$Scope: Modelo y Controller**
- Formas de creación del controlador (*controller*):  
Estilos y buenas prácticas de Angular

### Directivas y Plantillas (Vistas)

#### Desarrollo declarativo: Vistas

- Directivas básicas. Doble *binding*
- Expresiones
- Iteraciones y Condicionales
- Filtros
- Directivas y CSS

### Formularios y validaciones

#### Formularios

- Validación

### Extendiendo el DOM

#### Ampliación de las vistas

- Creación de directivas
- Creación de componentes
- Creación de filtros

#### Scope Avanzado. Jerarquía y Herencia

- Seguimiento de cambios
- Angular Local-Storage

- Formas de invocación
- Eventos

## Servicios en AngularJS

Elementos inyectables. Diferencias

- Servicio \$log
- Servicio \$locale

Enrutamiento

- ngRoute
- Ui-router

Promesas y Datos REST

- Promesas
- Servicio \$http
- Servicio \$Resource

Inyectables

Creación

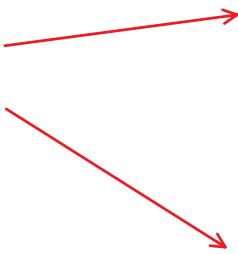
# Ξ AngularJS. Fundamentos

sábado, 16 de septiembre de 2017 19:36

- Fundamentos de AngularJS
  - *Frameworks en JS*
  - Presentación de AngularJS



- Tecnologías implicadas
  - Entorno de trabajo
  - ECMAScript 6 (ES6)



- Versiones de AngularJS. AngularJS 1.5. Angular 2.x - 4.x
- Líneas maestras. MVC SPA
- Elementos de AngularJS
  - Vista y controlador. Doble *binding*
  - Módulos y estilos
- Evolución: Angular 1.5. Componentes

- Navegadores y Servidor Web
- Editores de código
- Gestión del proyecto
  - Instalaciones: Node, npm
  - Versiones : Git, GitHub
  - Despliegue

- Funciones Arrow
- Clases
- Promises

# Frameworks en JS

sábado, 9 de septiembre de 2017 13:32

## Bibliotecas o frameworks

- facilitan el desarrollo
- automatizan procesos
- aumentan la eficacia
- mejoran el producto

*jQuery  
Underscore.js  
MooTools  
Prototype  
Google Web Toolkit (de Java a JS)  
YUI*  
  
*Ext JS  
Vue.js  
SAP - OpenUI5*  
  
**AngularJS / Angular**  
*BackboneJS  
Ember.js  
React.js*

*AccDC  
Ample SDK  
Atoms.js  
DHTMLX  
Dojo  
Echo3  
Enyo  
Handlebars  
Kendo UI  
Knockout..js  
D3.js - Kinetic.js*

*Meteor  
PhoneJS  
Pyjamas  
qooxdoo  
Rialto  
SmartClient & SmartGWT  
Socket.IO  
SproutCore  
Wakanda  
ZK  
Webix*



### BackboneJS

<http://backbonejs.org/>



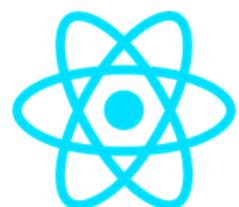
### Ember.js

<http://emberjs.com/>



### AngularJS

<https://angularjs.org>



### React.js

<http://emberjs.com/>

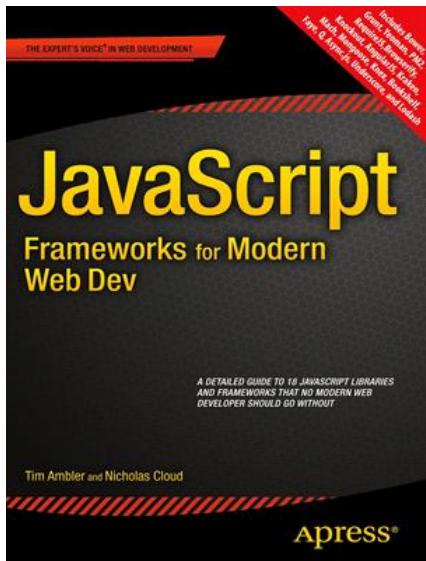
La elección de uno de ellos depende de los objetivos de cada proyecto



<http://todomvc.com/>



## Frameworks interrelacionados



**JavaScript Frameworks for Modern Web Dev**  
Tim Ambler & Nicholas Cloud  
Apress, 2015

## Contents at a Glance

About the Authors.....	xix
About the Technical Reviewer.....	xxi
Acknowledgments .....	xxii
Introduction .....	xxv
■ Chapter 1: Bower .....	1
■ Chapter 2: Grunt .....	11
■ Chapter 3: Yeoman .....	37
■ Chapter 4: PM2 .....	53
■ Chapter 5: RequireJS .....	73
■ Chapter 6: Browserify.....	101
■ Chapter 7: Knockout.....	121
■ Chapter 8: AngularJS .....	155
■ Chapter 9: Kraken.....	191
■ Chapter 10: Mach .....	251
■ Chapter 11: Mongoose.....	297
■ Chapter 12: Knex and Bookshelf .....	345
■ Chapter 13: Faye.....	381



AngularJS

# Introducción a Angular

sábado, 9 de septiembre de 2017 16:43

The screenshot shows the AngularJS.org homepage. At the top, there's a navigation bar with links for Home, Learn, Develop, Discuss, and a search bar. The main header features the AngularJS logo with the text "ANGULARJS by Google". Below the header, a tagline says "HTML enhanced for web apps!". There are two prominent buttons: "Download AngularJS 1" (version 1.6.0-rc.2 / 1.5.9 / 1.2.32) and "Try the new Angular 2". Below these are links to "View on GitHub" and "Design Docs & Notes". Social media links for Facebook, Twitter, and GitHub are also present. A button at the bottom says "Learn Angular in your browser for free!".

<https://angularjs.org/>

The screenshot shows the Angular.io homepage. It features a large Angular logo at the top. Below it, a banner with a city skyline background contains the text "One framework. Mobile & desktop." and a "GET STARTED" button. A callout box in the bottom right corner promotes "Google Developer Day Beijing & Shanghai 12/2016" with a "REGISTER NOW" link.

<https://angular.io/>

## Orígenes y desarrollo

Misko Hevery

Adam Abrons



- proyecto de **código abierto**, realizado íntegramente en JavaScript
- creado en **2009**, por Misko Hevery de *Brat Tech LLC* y Adam Abrons
- está mantenido por **Google** y junto con una amplia y creciente **comunidad**.
- puede coexistir con **otros frameworks**  
(e.g JQuery, Bootstrap, Material Design)
- se ha hecho muy popular desde finales de 2012 hasta ahora,
- especialmente en asociación con otras tecnologías, dando lugar a **MEAN**

MongoDB  
ExpressJS  
AngularJS  
NodeJS



<http://mean.io/#!/>

Se habla de una nueva *technology fullstack*  
como antes era *xAMP* (Apache + MySQL + PHP)

Se traduce a aplicaciones **JavaScript de principio a fin** (*End-to-End*)

The screenshot shows the MEAN.io homepage. The header includes a navigation bar with links for Home, Documentation, Packages, Release Notes, Support, Blog, and Contact. The main title is "The Friendly & Fun Javascript Fullstack for your next web application". Below the title, a subtitle reads "MEAN is an opinionated fullstack javascript framework - which simplifies and accelerates web application development.". A section titled "Get MEAN by running..." provides the command "\$ sudo npm install -g mean-cli" and "\$ mean init yourNewApp". At the bottom, there are links for "LATEST RELEASE: v0.5.8", "LATEST COMMIT: Nov 23, 2015", and "FORKS: 2396".

## Ejemplos de uso

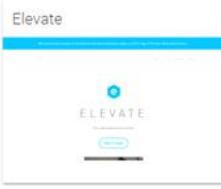
made with NGULAR

<https://www.madewithangular.com/#/>

### Communication



### Education



[SEE ALL](#)

## Instalación básica

domingo, 17 de septiembre de 2017 0:08

The screenshot shows the AngularJS website's download section. At the top, there's a header with the AngularJS logo and the text "HTML enhanced for web apps!". Below it, there are links to "View on GitHub", "Download (1.5.0-beta.2) [2.0MB]", and "Deploy Direct & Notes". A large blue arrow points from the left towards the download options. On the right, there's a sidebar with the title "Descarga" and "Vinculación a una CDN". The main content area has sections for "Download AngularJS", "Branch" (set to 1.5.x (beta)), "Build" (Minified), "CDN" (https://ajax.googleapis.com/ajax/libs/angularjs/1.5.0-beta.2/angular.min.js), "Bower" (bower install angular#1.5.0-beta.2), "npm" (npm install angular@1.5.0-beta.2), and "Extras" (Browse additional modules). At the bottom, there's a "Previous Versions" link and a prominent blue "Download" button.

The screenshot shows the AngularJS website's module download section. The background features a yellow and orange abstract design. At the top, there's a large teal header "Módulos de Angular" with a red Angular logo to its right. Below it, there's a "Download AngularJS" section with a "Previous Versions" link and a "Download" button. The main content area has sections for "Branch" (1.5.x (beta)), "Build" (Minified), "CDN" (https://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js), "Bower" (bower install angular#1.4.8), "npm" (npm install angular@1.4.8), and "Extras" (Browse additional modules). A large blue arrow points from the left towards the module list. To the right of the module list, there's a bulleted list of available modules: angular-animate, angular-aria, angular-cookies, angular-loader, angular-message / message-format, angular-resource, angular-route, angular-sanitize, and angular-touch. At the bottom, there's an "Index of /1.4.8/" link.

## *Editores de código*

**Editores de texto  
específicos para  
código**

Sublime Text



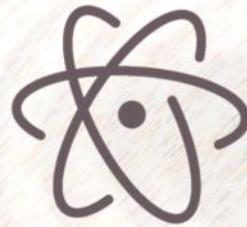
<http://www.sublimetext.com/>

Brackets



<http://brackets.io/>

Atom



<https://atom.io/>

# Visual Studio Code

The screenshot shows the official Visual Studio Code website. At the top, there's a navigation bar with links to 'Visual Studio Code', 'Docs', 'Updates', 'Blog', 'Extensions', and 'FAQ'. A green 'Download' button is also present. Below the navigation, a message says 'Version 1.7 is now available! Read about the new features and fixes in October.' On the left, there's a large banner with the text 'Code editing. Redefined.' and 'Free. Open source. Runs everywhere.' followed by a 'Download for Windows' button. A note below it says 'Available on other platforms and insiders build' and includes a link to the license and privacy statement. In the center, a large blue 'VS Code' logo is displayed. To the right, a screenshot of the VS Code interface shows a code editor with several tabs ('app.ts', 'www.ts', 'package.json', 'README.md') and a sidebar with extension icons like 'C/C++', 'Python', 'Debugger for Chrome', etc. A URL box at the bottom contains '<https://code.visualstudio.com/>'.

## AngularJS 1.x

- extiende directamente las funcionalidades HTML
- utiliza como **patrón arquitectónico MVC** (Modelo, Vista, Controlador) o similares (estrictamente sería MV\* o MVW (Model-View-Whatever))
- esta especialmente orientado a la creación de **aplicaciones SPA** (Single-Page Applications).
- es muy eficiente: promueve el **uso patrones** de diseño de software
- permite crear **tests** unitarios y End-to-End de forma sencilla empleando Jasmine y Karma
- al estar exclusivamente **orientado a la lógica**, es un framework muy liviano: no incluye elementos gráficos ni CSS.  
Se complementa muy bien con Bootstrap o Material Design



# Características: MVC

**Patrón arquitectónico** (según otros autores **patrón de diseño**)  
separación del código de los programas dependiendo de su responsabilidad

Se basa en las ideas claves  
en el desarrollo de la  
ingeniería del software

- **reutilización de código**  
(*code reuse*, [Douglas McIlroy](#), 1968)
- **separación de conceptos**  
(*separation of concerns*,  
[Edsger W. Dijkstra](#), 1974 )

Fue introducido por el científico  
noruego [Trygve Reenskaug](#)  
cuando trabajaba con Smalltalk-76 en el  
*Xerox Palo Alto Research Center* (PARC)

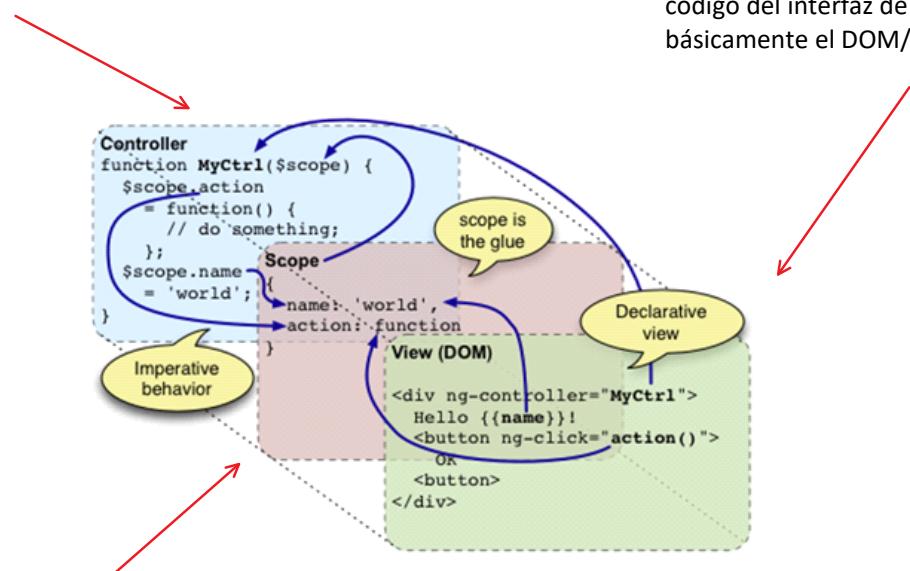
Más tarde fue re implementado  
en Smalltalk-80



## Model - View - Whatever

**Controladores:** Se encargarán de la lógica de la aplicación, incluyendo junto al "controller", "Factorías" y "Servicios" para mover datos contra servidores o memoria local en HTML5.

**Vistas:** Será la representación de los datos o la información, es decir, el código del interfaz de usuario, básicamente el DOM/HTML/CSS .



**Modelo o Modelo de la vista,**  
según se emplee la variante del  
patrón MVC o MVVC, es la

estructura lógica que subyace a los datos.

Asociado a él se define el ***scope***, responsable de detectar los cambios en el modelo y proporciona el contexto a las plantillas.

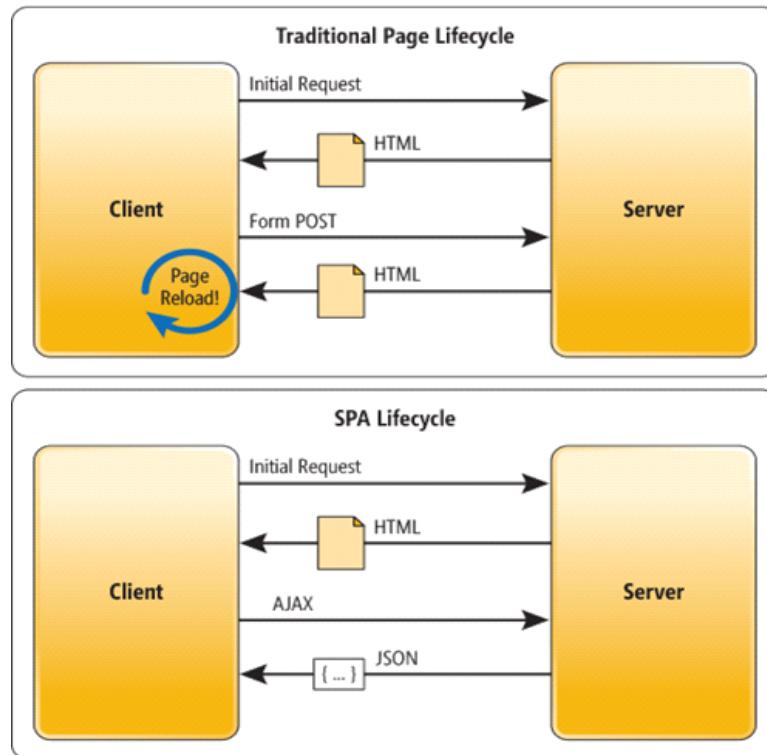
**Ampliación:** MVW en Angular

- MVC – Model-View-Controller
- MVVM – Model-View-View Model

## SPA: Single-Page Applications

sábado, 9 de septiembre de 2017 17:13

- carga completa en **una sola página**
- **Asincronicidad**
- limitada dependencia del **servidor**
- aproximación a las **aplicaciones de escritorio**

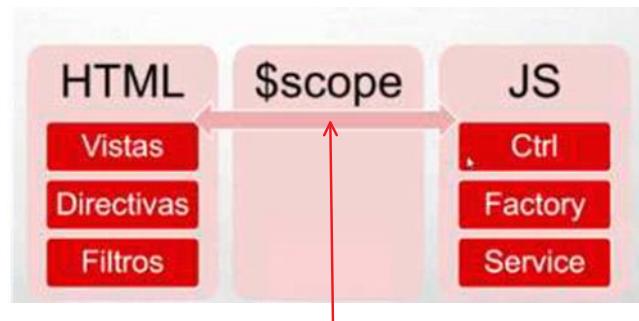


# Elementos de Angular JS

sábado, 9 de septiembre de 2017 17:35

En términos prácticos, se distingue

- HTML = programación declarativa
- JS = programación imperativa



El *binding* entre ambos es responsabilidad del *scope*

**objeto normal de JavaScript** que almacena los datos de un modelo.



Recibe su nombre porque sustituye a *window* como ámbito global a nivel de la vista, por lo que actuaría como un *view model*

## Elemento: HTML

### (Vistas)



AngularJS  
extiende el vocabulario  
del código HTML

proporcionarnos la introducción  
de lógica en la representación  
de nuestra información



#### Directivas

- Son atributos HTML **ng-**...
- Suponen la posibilidad de modificar los elementos del DOM
- Pueden ser predefinidas o propias

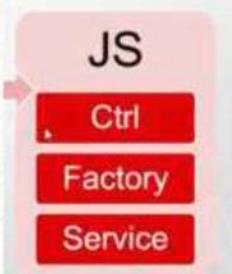
#### Expresiones {{}}

- permite el uso de los elementos del modelo a nivel de las vistas
- También permite expresiones, e.g. operaciones matemáticas
- Lenguaje de plantillas : Similar a otros motores, como *Mustache*, *Smarty* o *Jade*



# Elemento: JavaScript

(Controlador / Modelo)



## Controlador (controller)

Es el código con la lógica que:

- define el modelo
- lo comunica con la vista mediante el *scope*

## Modelos

Son la representación de los datos de la aplicación.

Se pueden definir de dos maneras

- en el código JS del **controlador**
- directamente en el HTML (la vista)

ng-init: tareas de inicialización de la aplicación, también permite definir el modelo directamente en el HTML

**NADA RECOMENDABLE** (anti-patrón)

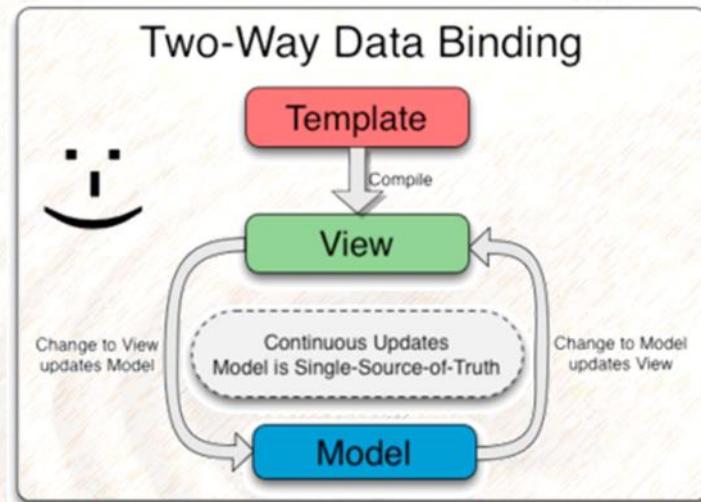


## Elemento: Doble binding

**Doble binding (Two-way data binding)**  
entre el interface (vista) y el modelo

la vista se actualiza  
automáticamente  
cuando cambia el  
modelo, y viceversa

Técnica frecuente en  
Flex / ActionScript





# Desarrollo declarativo

desarrollo declarativo → **expansión** de las características del **HTML**, añadiéndole **funcionalidades** sin necesidad de escribir código JavaScript

Se puede ver como una forma de **agregan valor semántico** al HTML.

**Directivas** atributos **ng-** específicos de angularJS que se pueden asignar a cualquier etiqueta HTML.

**Expresiones** lenguaje de **plantillas** mediante **{{}}**

En todas ellas es posible añadir el prefijo **data-** para que las directivas no supongan problema de validación



## Directivas básicas

- **ng-app**      inicializa la aplicación;  
                      define el elemento en el que se **auto ejecuta Angular**;
  - ✓ lo más común es ponerlo al principio del documento, en la etiqueta HTML o BODY
  - ✓ también se puede colocar en un área más restringida dentro del documento en otra de las etiquetas de tu página (e.g. SECTION, DIV)
  
- **ng-init**      tareas de **inicialización** de la aplicación.  
                      Permite definir el modelo, creando propiedades en el **scope**, directamente en el HTML (anti-patrón)
  - ✓ El único caso apropiado donde se debería de usar ngInit es en enlace de propiedades especiales de **ngRepeat**.

## Hola Mundo

Ejemplo de código AngularJS **puramente declarativo** con una funcionalidad muy básica:  
la vista incorpora implícitamente el modelo y el controlador,

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<title>Hola Mundo</title>
<meta name="viewport" content="width=device-width, initial-scale=1">
<script src="../lib/angular/angular.js"></script>
</head>
<body ng-app ng-init="name='Pepe';apellido='Perez'">
<form>
<p>
<label for="name">Nombre</label>
<input id="name" type="text" ng-model="name">
</p>
<p>
<label for="apellido">Apellido</label>
<input id="apellido" type="text" ng-model="apellido">
</p>
<p>
<button ng-click="name=''; apellido=''">Borrar</button>
</p>
</form>
<p>Hola {{name.toUpperCase()}} {{apellido.toUpperCase()}}</p>
</body>
</html>
```

## Hola Mundo MVC

Ejemplo de código AngularJS con la misma funcionalidad básica repartida entre elementos **declarativos** (la vista HTML) y elementos **imperativos** (el modelo y el controlador definidos en el fichero JS)

### Vista HTML

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<title>Hola Mundo</title>
<meta name="viewport" content="width=device-width, initial-scale=1">
<script src="../lib/angular.js"></script>
<script src="./app.js"></script>
</head>
<body ng-app="appMain" ng-controller="AppController">
<form>
<p>
<label for="name">Nombre</label>
<input id="name" type="text" ng-model="user.name">
</p>
<p>
<label for="apellido">Apellido</label>
<input id="apellido" type="text" ng-model="user.apellido">
</p>
<p>
<button ng-click="borrar()">Borrar</button>
</p>
</form>
<p>Hola {{user.name.toUpperCase()}} {{user.apellido.toUpperCase()}}</p>
</body>
</html>
```

- **ng-app** : define el **módulo** de Angular que se instanciará como inicio de la aplicación
- **ng-controller** : define cual será la **función constructora** encargada de instanciar el \$scope
- **ng-model** : define aquellos elementos dinámicos (entradas de datos) asociados a **propiedades del modelo**, i.e. de \$scope
- **ng-click** : permite declarar un método del \$scope que actuara como un determinado **manejador de evento**
- las expresiones {{}} permiten hacer uso de los valores del modelo como parte de la salida de datos en cualquier punto de la vista

### Controlador y modelo (JS)

```
Instanciación del módulo principal → angular.module('appMain', [])

Propiedades del modelo declaradas en el scope → .controller('AppController', ['$scope', function ($scope)
{
    $scope.user = {
        name : 'Pepe',
        apellido : 'Perez'
    }
    $scope.borrar = function () {
        $scope.user.name='';
        $scope.user.apellido=''
    }
}]]>
```

controller principal 2 argumentos:  
- su nombre  
- array de inyección de dependencias  
- nombre de los argumentos  
- función anónima con dichos argumentos

# Elementos: Modularización



## Máxima modularización

- el código del controlador es el mínimo
- se distribuye en **módulos (module)**
- se transfieren funciones a los **servicios (service)**
- la creación de objetos se canaliza en **factorías (factory)**

## Inyección de dependencias

- Modularidad → escalabilidad
- Acceso a servicios únicamente cuando son necesarios

# Guía de estilo



John Papa

↓  
Modularización  
extrema

[GitHub](#) This repository Search Explore Features Enterprise Pricing Sign up Sign in

johnpapa / angular-styleguide Watch 1,096 Star 15,272 Fork 2,349

Code Issues 34 Pull requests 8 Pulse Graphs

Angular Style Guide: A starting point for Angular development teams to provide consistency through good practices. <http://johnpapa.net>

1,017 commits 3 branches 0 releases 131 contributors

Branch: master New pull request New file Find file HTTPS https://github.com/johnpapa/ Download ZIP

johnpapa Merge pull request #634 from kel-sakal/bk/turkish	Latest commit 58599ad4 4 days ago
assets description changed from "Angular controller" to "Angular directive"	2 months ago
assets Turkish translation	7 days ago
LICENSE Update license year to 2016	5 days ago
README.md Update license year to 2016	5 days ago
README.md	

Angular Style Guide  
Angular Team Endorsed

[https://github.com/johnpapa/  
angular-styleguide](https://github.com/johnpapa/angular-styleguide)

Special thanks to Igor Minar, lead on the Angular team, for reviewing, contributing feedback, and entrusting me to shepherd this guide.

- Uso del formato "*controller as*" unido al empleo de una variable dentro del controlador,  
`var vm = this`, para evitar el uso de `this`
- Referencia a los módulos sin mapearlos nunca como variables
- *Closures* con el patrón de auto ejecución "*Immediately Invoked Function Expression*" (IIFE),  
 con objeto de eliminar las variables del ámbito global.
- Empleo continuado de funciones con nombre en lugar de funciones anónimas
- Uso de `$inject` para definir los nombres de los elementos que se inyectan,  
 de cara a evitar los problemas en el caos de la minimificación
- Extrema modularización

Arquitectura de un *controller* tal como lo proporciona el *snippet* de la extensión de **John Papa** para VSC

```

Closures con el patrón
de auto ejecución IIFE
  ↗
(function() {
  'use strict';
  angular
    .module('Module')
    .controller('ControllerController', ControllerController);

Referencia a los
módulos sin mapearlos
  ↗
ControllerController.$inject = ['dependency1'];
function ControllerController(dependency1) {
  var vm = this;
  activate();
  ///////////////
  function activate() { }

  })(());
}

Función con
nombre
  ↗
variable dentro del controlador que
referencia a this, como complemento
al uso del formato "controller as" en la
vista (HTML)
  ↗
Uso de $inject
  ↗

```

## Más Hola Mundos

sábado, 16 de septiembre de 2017 20:31

### Hola Mundo "as"

Modo "*controller as*" que define el acceso al *scope* mediante una variable asignada en la vista, tal como recomienda la guía de estilo "oficial" creada por **John Papa**.

#### Vista HTML

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Hola Mundo</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <script src="../lib/angular.js"></script>
    <script src="./app.js"></script>
</head>
<body ng-app="appMain" ng-controller="AppController as vm">
    <form>
        <p>
            <label for="name">Nombre</label>
            <input id="name" type="text" ng-model="vm.user.name">
        </p>
        <p>
            <label for="apellido">Apellido</label>
            <input id="apellido" type="text" ng-model="vm.user.apellido">
        </p>
        <p>
            <button ng-click="vm.borrar()">Borrar</button>
        </p>
    </form>
    <p>Hola {{vm.user.name.toUpperCase()}} {{vm.user.apellido.toUpperCase()}}</p>
</body>
</html>
```

Acceso al *scope* mediante una variable definida en la vista

Todos los accesos al *scope* (modelo y funciones del controlador) se realizan mediante la variable definida

#### Controlador y modelo (JS)

```
Propiedades del modelo → angular.module('appMain', [])
                           .controller('AppController', function () {
                               this.user = {
                                   name : 'Pepe',
                                   apellido : 'Perez'
                               }
                               this.borrar = function () {
                                   this.user.name='';
                                   this.user.apellido='';
                               }
                           })
Métodos del controlador →
```

Los elementos del *scope* se declaran directamente como propiedades de la función constructora que constituye el *controller*

Siguiendo las recomendaciones de la guía de estilo de **John Papa**, se pueden utilizar funciones con nombre en lugar de anónimas.

```
angular.module('appMain', [])
.controller('AppController', AppController)
function AppController () {
    this.user = {
        name : 'Pepe',
        apellido : 'Perez'
    }
    this.borrar = function () {
        this.user.name='';
        this.user.apellido='';
    }
}
```

Controller declarado como una función con nombre

La propiedad del *hoisting* (alzamiento) permite declarar la función después de que haya sido utilizada.

Siguiendo estrictamente dicha guía el código quedaría como sigue

```
(function() {
    'use strict';
    angular
        .module('appMain', [])
        .controller('AppController', AppController);

    ControllerController.$inject = ['dependency1'];
    function AppController(dependency1) {
```

```
var vm = this;  
vm.user = {  
    name : 'Pepe',  
    apellido : 'Perez'  
}  
vm.borrar = function () {  
    vm.user.name='';  
    vm.user.apellido='';  
}  
}();
```

## Elementos de AngularJS 1.5



### Componentes

- template + controller
- controllerAs
- (por defecto \$ctrl)
- uso de clases
- constructores:
- inyección de dependencias
- ciclo de vida del componente  
(onInit)
- comunicación entre componentes:
  - parent
  - binding

```
1 // ...
2 class SampleController {
3
4   /** ...
5    * @ngInject;
6    */
7   constructor($scope) {
8     'ngInject';
9     this.$scope = $scope;
10   }
11
12   /** ...
13    * @ngInject;
14    */
15   $onInit () {
16     this.name = "Sample"
17     this.value = parent.value
18   }
19   // Fin del $onInit
20
21 } // Fin del controller SampleController
22
23 angular.module('moduleName')
24
25 /**
26 */
27 .component("sample", {
28   require: {parent: '^appMain'},
29   templateUrl: 'components/sample.html',
30   // usa controller as por defecto
31   controller: SampleController,
32   //controllerAs: '$ctrl', valor por defecto
33   bindings: {}
34
35 })
36 //Fin del componente y del objeto que lo define
37
38
39
40
41
42
43
44
45
46
47
48
49 })
```

# Guía de estilo 1.5



Personal Open source Business Explore Pricing Blog Support This repository Search Sign in Sign up

Watch 345 Star 5,073 Fork 596

Issues 4 Pull requests 0 Projects 0 Pulse Graphs

leguide for teams <https://ultimateangular.com>

181 commits 3 branches 0 releases 40 contributors

laskoviy mishka committed with **toddmotto** feat(ts): Initial typescript version of guide. (#127) 8 days ago

i18n Adding Italian translation (#138) 11 days ago

typescript feat(ts): Initial typescript version of guide. (#127) 8 days ago

README.md Update README.md (#137) 21 days ago

<https://github.com/toddmotto/angular-styleguide>

## Angular 1.x styleguide (ES2015)

Architecture, file structure, components, one-way dataflow and best practices

Want an example structure as reference? Check out my component based architecture 1.5 app.

## **Futuro de HTML: Web Components**

### **Web Components**

conjunto de **estándares** que, permiten crear y utilizar elementos HTML personalizados.

Se puede así ampliar el “vocabulario” de HTML con elementos propios

En ellos está trabajando la W3C. Ya se soportan en algunos navegadores (Chrome) y esta disponible un *polyfile* para que puedan usarse en otros.

Constan de cuatro especificaciones:

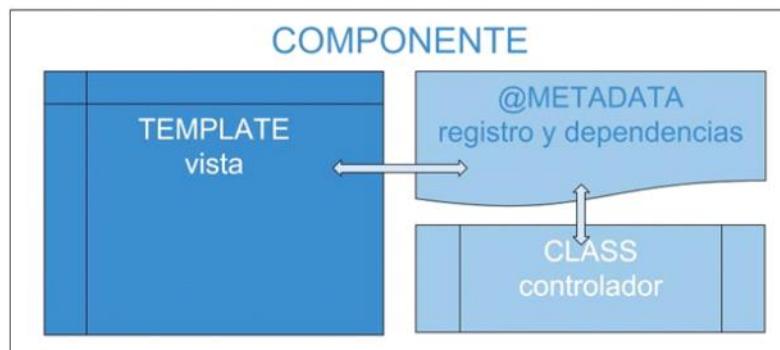
- **Custom elements:** Nos permite definir nuevos elementos HTML.
- **Templates:** Sistema de plantillas nativas en el navegador.
- **Shadow DOM:** DOM scope.
- **HTML Imports:** Carga de documentos HTML.

# Continuando con Web Components

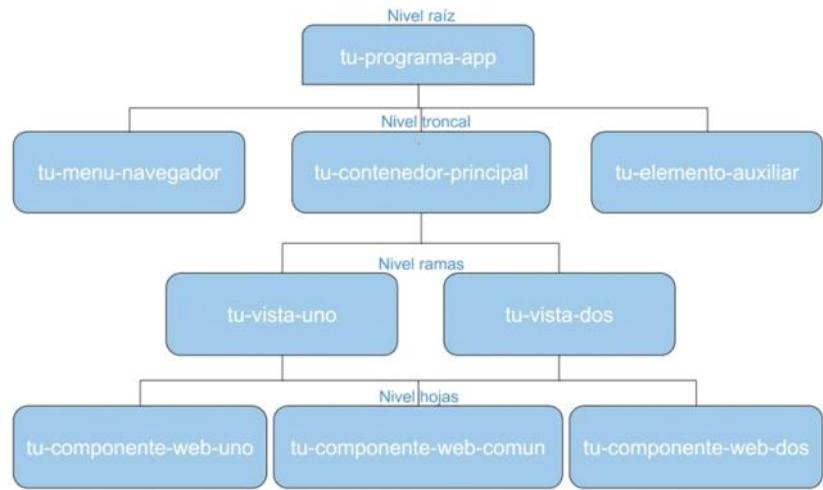


## Componentes en Angular 2

- **Elemento personalizado:** Nos permite definir nuevos elementos HTML.
- Cada uno de ellos con su **template**: Sistema de plantillas nativas en el navegador.
- **Shadow DOM**: contenedor no visible
- **ciclo de vida** bien definido
- evolución de los componente definidos en Angular 1.5



## Árbol de componentes



# Últimos Hola Mundos

sábado, 16 de septiembre de 2017 21:25

## Hola Mundo ES6

### Vista HTML

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Hola Mundo</title>
    <meta name="description" content="">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <script src="../lib/angular.js"></script>
    <script src=".appModule.js"></script>
    <script src=".appController.js"></script>
</head>
<body ng-app="appMain" ng-controller="AppController as $ctrl">
    <form>
        <p>
            <label for="name">Nombre</label>
            <input id="name" type="text" ng-model="$ctrl.user.name">
        </p>
        <p>
            <label for="apellido">Apellido</label>
            <input id="apellido" type="text" ng-model="$ctrl.user.apellido">
        </p>
        <p>
            <button ng-click="$ctrl.borrar()">Borrar</button>
        </p>
    </form>
    <p>Hola {{ctrl.user.name.toUpperCase()}} {{ctrl.user.apellido.toUpperCase()}}</p>
</body>
</html>
```

### Controlador y modelo (JS)

```
class AppController {
    constructor () {} ← Sólo se usa para inyección de dependencias
    $onInit () {
        this.user = {
            name : 'Pepe',
            apellido : 'Perez'
        }
    } ← Método ligado automáticamente al momento inicial del ciclo de vida de la clase
    borrar () {
        this.user.name = '';
        this.user.apellido = ''; ← Método manejador de un evento click
    }
}
angular.module('appMain')
.controller('AppController', AppController)
```

## Hola Mundo ES6 por componentes

### Vista HTML

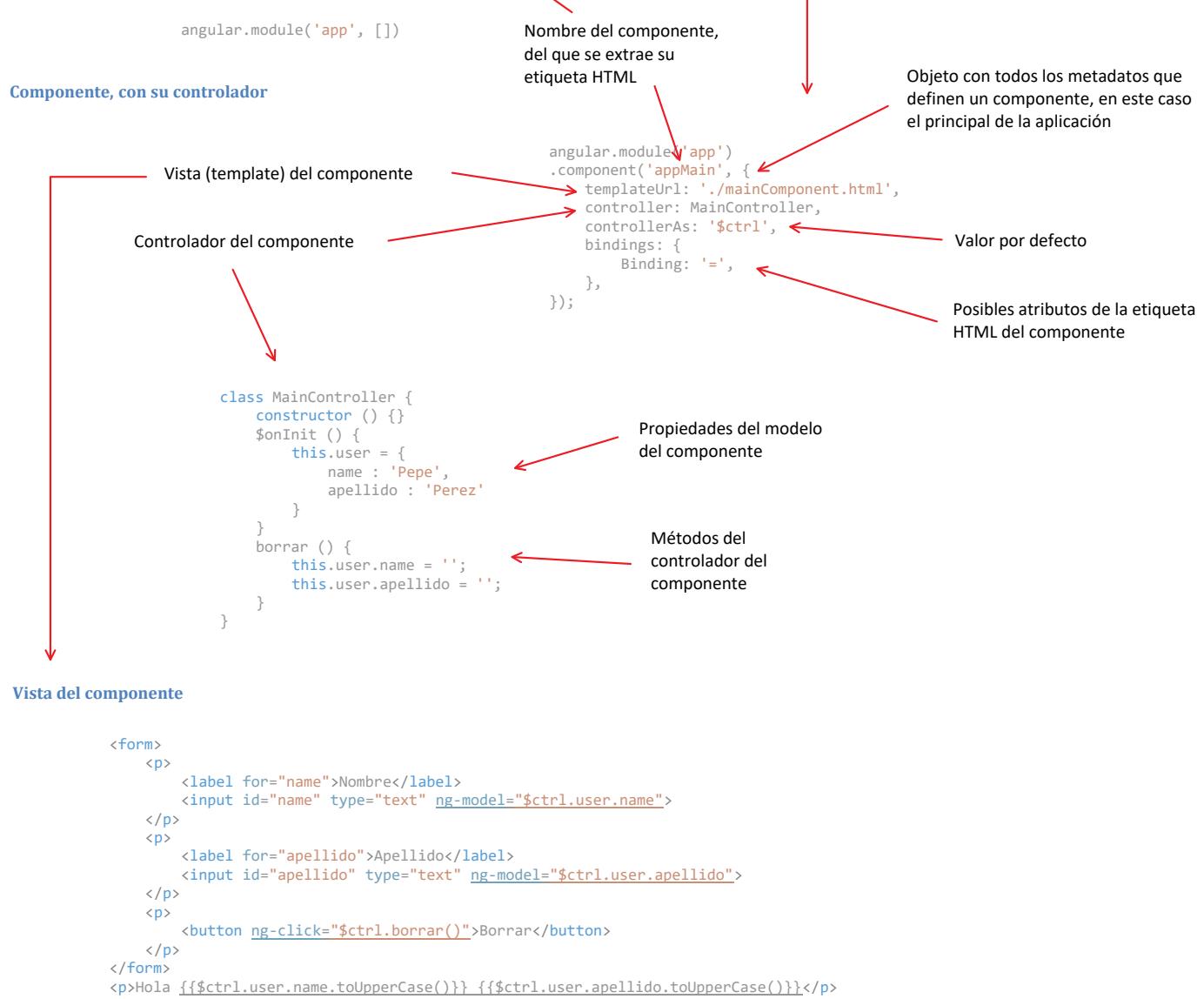
```
<!DOCTYPE html>
<html lang="es">
    <head>
        <meta charset="utf-8">
        <meta http-equiv="X-UA-Compatible" content="IE=edge">
        <title>Hola Mundo</title>
        <meta name="description" content="">
        <meta name="viewport" content="width=device-width, initial-scale=1">
        <script src="../lib/angular.js"></script>
        <script src=".appModule.js"></script>
        <script src=".mainComponent.js"></script>
    </head>
    <body ng-app="app">
        <app-main></app-main>
    </body>
</html>
```

Módulo principal

```
angular.module('app', [])
```

Nombre del componente, del que se extrae su etiqueta HTML

Objeto con todos los metadatos que



# Tecnologías y Entorno de trabajo

sábado, 9 de septiembre de 2017 13:33

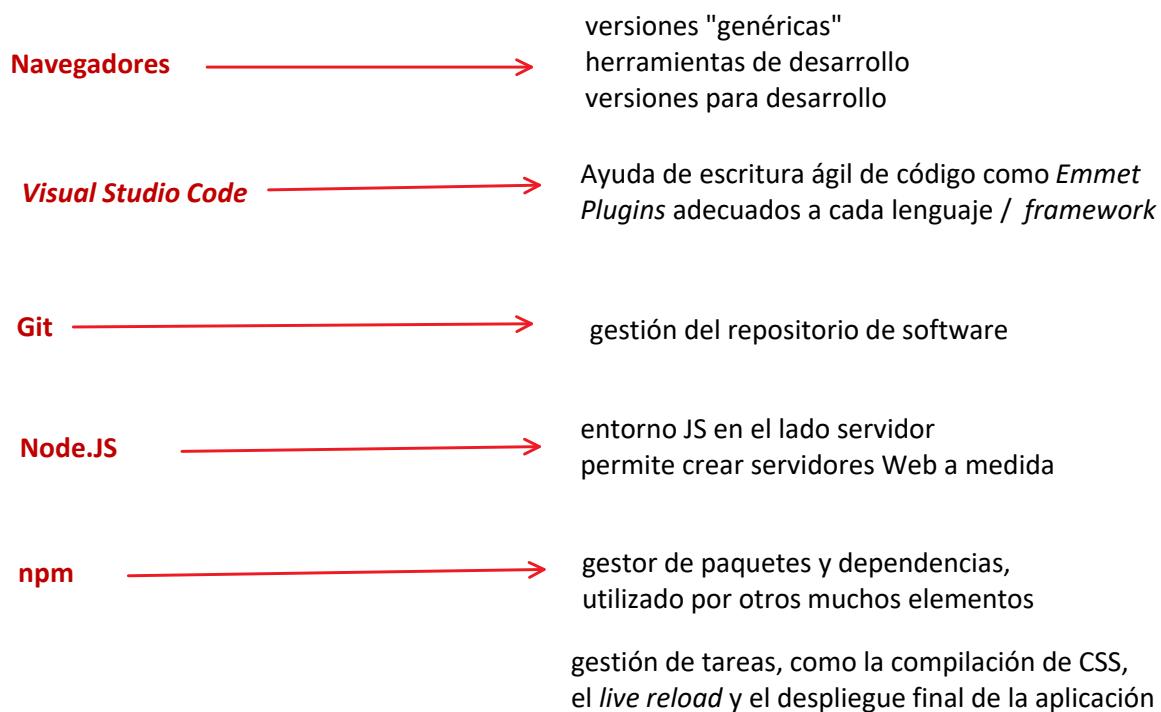
## Entorno de trabajo

- Navegadores y servidor Web
- Editores de código
- Gestión de versiones. GIT
- *NodeJS* y npm. Despliegue

## Tecnologías

- ES6

## Entorno de trabajo



# Navegadores

sábado, 9 de septiembre de 2017 18:20

Herramienta de desarrollador en las últimas versiones de Chrome, en este caso la 60.0.3



Llévate Chrome a todas partes

Inicia sesión con tu cuenta de Google para acceder a los marcadores, las contraseñas, el historial y otros ajustes desde todos tus dispositivos.

Guarda página como... Ctrl+S  
Añade al escritorio...  
Borrar datos de navegación... Ctrl+Mayús+S  
Extensiones  
Administrador de tareas Mayús+E  
Herramientas para desarrolladores Ctrl+Mayús+I

Editor Cortar Copiar Pegar  
Configuración Ayuda Salir

HTML ( 1 item hidden by filters )

```
html { direction: ltr; }
html {
    -google-red-100: #4c7c31;
    -google-red-300: #e67e22;
    -google-red-500: #d94d4d;
    -google-red-700: #d94d4d;
}
```

Herramienta de desarrollador en las últimas versiones de Firefox, en este caso la 55.0.3



Tu Firefox está al día.  
Ahora ponte en marcha.

Envía Firefox a tu teléfono y da rienda suelta a tu Internet.

Escribe tu email

Navegador web Mozilla Firefox | Página de inicio de Mozilla | +

html body

Inspector Console Depurador Editor de estilos Memoria Red Almacenamiento Buscar en HTML

```
<!DOCTYPE html>
<html class="windows x86_js_w7up_64_loaded" dir="ltr" data-latest-firefox="55.0.3" data-esr-versions="52.3.0" data-optimize-project="140882122" data-gtm-container-id="GTM-M2R8V" data-gtm-page-id=""/><!--firefox-whatsnew--> data-stub-attribution-rate="1.0" lang="es-ES" data-gtm-page-id=""/>
```

Herramienta de desarrollador en Edge, el navegador incorporado desde Windows 10



Sugerencias de Microsoft Edge

Tu progreso 3% completado

Sigue siendo productivo

Recibe notificaciones de tus sitios web

Transmite tu contenido

Las opciones "Inspeccionar elemento" y "Ver origen" ahora se mostrarán en el menú contextual.

F12 Explorador DOM Consola 1 Depurador Red Rendimiento Memoria Emulación Experimentos

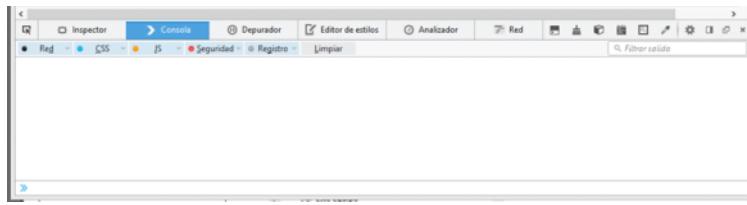
background-color: #fff;

body { margin: 0; overflow-y: scroll; }

## Consola JavaScript

El ambiente en el que se ejecutan los scripts (navegador) proporciona un objeto console, que corresponde a la consola JS que podemos hacer visible en la parte inferior del navegador.

Los métodos console.log y console.dir son otra alternativa para presentar texto en pantalla desde un script. Algunos autores la prefieren a alert(), por considerarla menos intrusiva.



## Versiones "especiales" para desarrolladores

The screenshot shows the Google Chrome Canary download page. At the top, it says 'Get on the bleeding edge of the web'. Below that, it says 'Google Chrome Canary has the newest of the new Chrome features. Be forewarned: it's designed for developers and early adopters, and can sometimes break down completely.' A yellow 'Download Chrome Canary' button is prominent. Below the button, it says 'For Windows 10/8.1/8/7 64-bit' and 'You can also download Chrome for Windows 32-bit, OSX, and Android.' An image shows a desktop monitor, a laptop, and two mobile devices (a smartphone and a tablet) all displaying the Chrome logo.

<https://www.google.es/chrome/browser/canary.html>

The screenshot shows the Mozilla Firefox Developer Edition landing page. It features the Firefox logo and the text 'Herramientas modernas para la Web Abierta'. Below that, it says 'Crea y depura experiencias web con poderosas herramientas de código abierto.' A blue 'Firefox Developer Edition' button is at the top. To the left, there's a 'Privacidad de Firefox' section. To the right, there are three columns: 'Modernizar' (Create interfaces de usuario rápidas, flexibles e interactivas con las herramientas integradas de React y Redux), 'Personalizar' (Haz tu propia herramienta de desarrollo gracias al código abierto y la personalización), and 'Optimizar' (Crea sitios adaptables y compatibles que funcionan para todos, en todos partes).

<https://www.mozilla.org/es-ES/firefox/developer/>

# Plugin para Chrome

A screenshot of the AngularJS Batarang extension page on the Chrome Web Store. The page title is "AngularJS Batarang" with a subtitle "offered by AngularJS". It shows a 4-star rating with 1148 reviews, 293,492 users, and tabs for "OVERVIEW", "REVIEWS", and "RELATED". A large image in the center shows the Batarang interface integrated into the Chrome DevTools, displaying AngularJS code and data structures. A callout box points to this image with the text: "Batarang: ayuda en la depuración mediante Chrome de proyectos de AngularJS". To the right of the main content, there's a sidebar with developer information: "Compatible with your device", "Extends the Developer Tools, adding tools for debugging and profiling AngularJS applications.", and details about the extension: "Website", "Report Abuse", "Version: 0.10.1", "Updated: October 1, 2015", "Size: 390KIB", and "Language: English". A black bat silhouette is positioned to the right of the sidebar.

Batarang: ayuda en la depuración mediante Chrome de proyectos de AngularJS

AngularJS Batarang  
offered by AngularJS

★★★☆☆ (1148) | Developer Tools | 293,492 users

OVERVIEW REVIEWS RELATED

Compatible with your device

Extends the Developer Tools, adding tools for debugging and profiling AngularJS applications.

Website Report Abuse

Version: 0.10.1 Updated: October 1, 2015

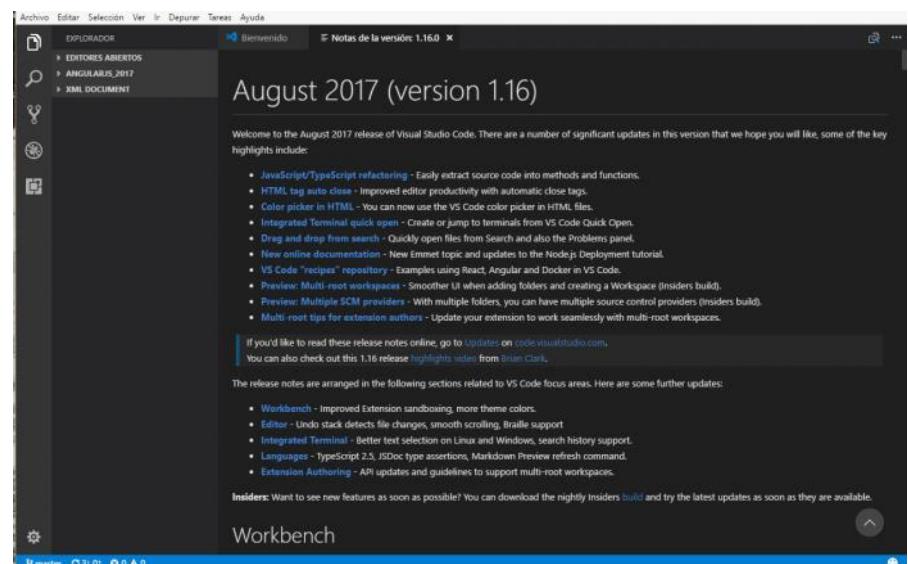
Size: 390KIB Language: English

# Servidor Web

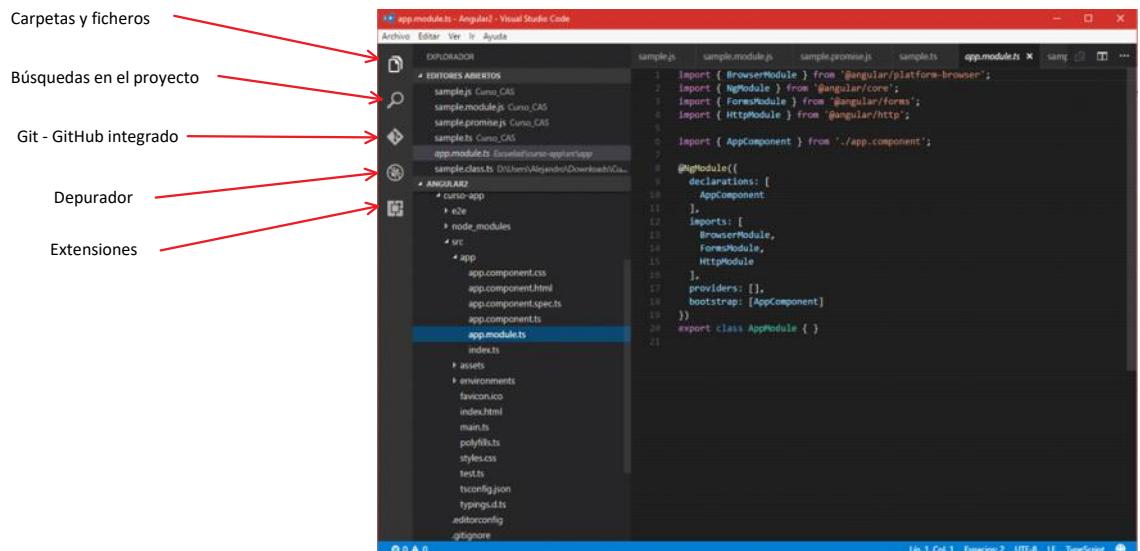
domingo, 17 de septiembre de 2017 9:44

# Editores de código

sábado, 9 de septiembre de 2017 18:21

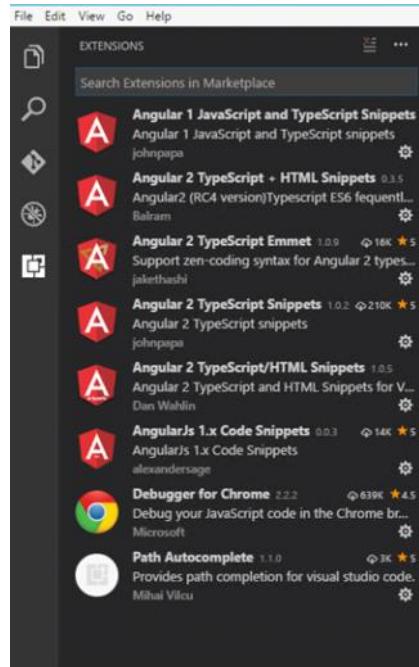


## Visual Studio Code



## Extensiones de VSC

- Angular 1 and TypeScript/  
HTML VS Code Snippets  
(Dan Wahlin)
- **Angular 1 JavaScript and Typescript  
Snippets \*\*\*\*\*  
(John Papa)**
- **Angular Language Service  
angular2-inline  
(Nate Wallace)**
- Debugger para Chrome \*\*\*\*\*  
(Microsoft)
- npm \*\*\*  
(egamma)
- **Path Intellisense \*\*\*  
(Christian Kohler)**





Git es un control de versiones distribuido para la gestión eficiente de flujos de trabajo distribuidos no lineales. Git fue diseñado y desarrollado inicialmente por Linus Torvalds en 2005 para el desarrollo del kernel de Linux. La licencia de Git es libre y hay distribuciones oficiales para los sistemas operativos:

- Mac OS X.
- Windows.
- Linux.
- Solaris.

La distribución de Git incluye herramientas de línea de comando y de escritorio.

Además, hay disponibles herramientas proporcionadas por terceros que permiten una mayor integración con el escritorio o con entornos de desarrollo.



Estados del archivo

Modificado      Preparado      Confirmado

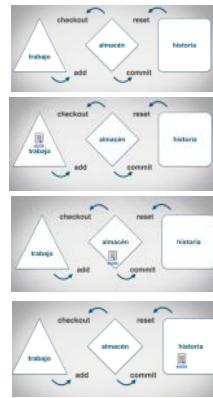
## Comandos

git init -> crear un repositorio local en un directorio.  
git clone -> crear un repositorio local haciendo una copia de otro (local o remoto).  
git add -> registra los ficheros del directorio de trabajo cuyos cambios se quieren.  
git commit -> confirma los cambios de los ficheros registrados  
git remote -> comando para administrar repositorios remotos

git branch -> crear y borrar ramas.  
git checkout -> permite cambiar de rama en el directorio de trabajo (Por defecto se trabaja en la rama denominada master)  
git push -> enviar los cambios a un repositorio remoto en la rama indicada  
git pull -> comando que combina git fetch y git merge para obtener cambios de un repositorio local

Este comando es exactamente un encadenamiento de dos comandos:  
git fetch - obtiene los cambios de una rama remota  
git merge - fusiona si es posible estos cambios con una rama local.

## Ciclo de operaciones



El repositorio creado tiene tres partes principales: el directorio en lo que se llama directorio de trabajo, la carpeta que es la carpeta .git que ha sido creada por el comando git init, y la carpeta .git que se encuentra en la zona de preparación, que actúa como una zona intermedia, y el historial de cambios.

.Git permite el uso de ramas (branches).

El comando git pull es un ejemplo de la orientación a caja de herramientas que caracteriza el diseño de Git.

Git es implementado como un conjunto de programas y scripts de shell que son fácilmente encadenables para formar nuevos procesos. Los scripts tienen mecanismos para llamar scripts de usuario cuando suceden ciertos eventos en el flujo de trabajo (denominados puntos de ejecución hooks).

git pull = git fetch + git merge



Resultado



## Comprobación

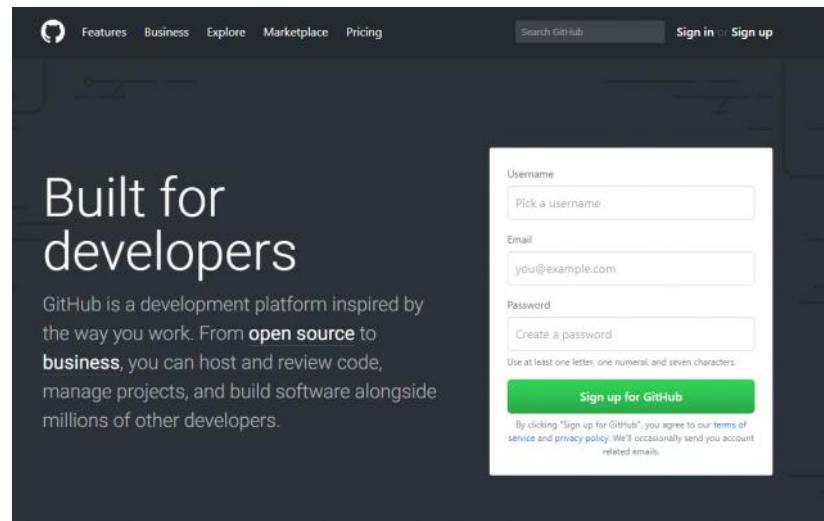
```
git init
git add .
git commit -m "primera comprobacion"
```



# Git en la Nube. GitHub

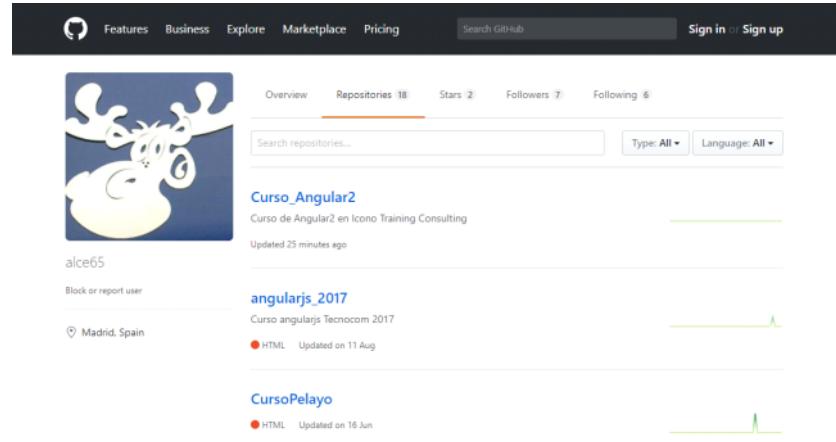
domingo, 10 de septiembre de 2017 12:28

- **GitHub** →
- GitLab
- Bitbucket



The screenshot shows the GitHub sign-up interface. At the top, there's a navigation bar with links for Features, Business, Explore, Marketplace, and Pricing. On the right side of the header, there are buttons for 'Search GitHub', 'Sign in', and 'Sign up'. The main content area features a large heading 'Built for developers' and a descriptive paragraph about GitHub's mission to support developers. Below this, there are input fields for 'Username' (with placeholder 'Pick a username'), 'Email' (with placeholder 'you@example.com'), and 'Password' (with placeholder 'Create a password'). A note below the password field specifies that the password must be at least one letter, one numeral, and seven characters. A prominent green 'Sign up for GitHub' button is located at the bottom right of the form area.

<https://github.com/alce65>

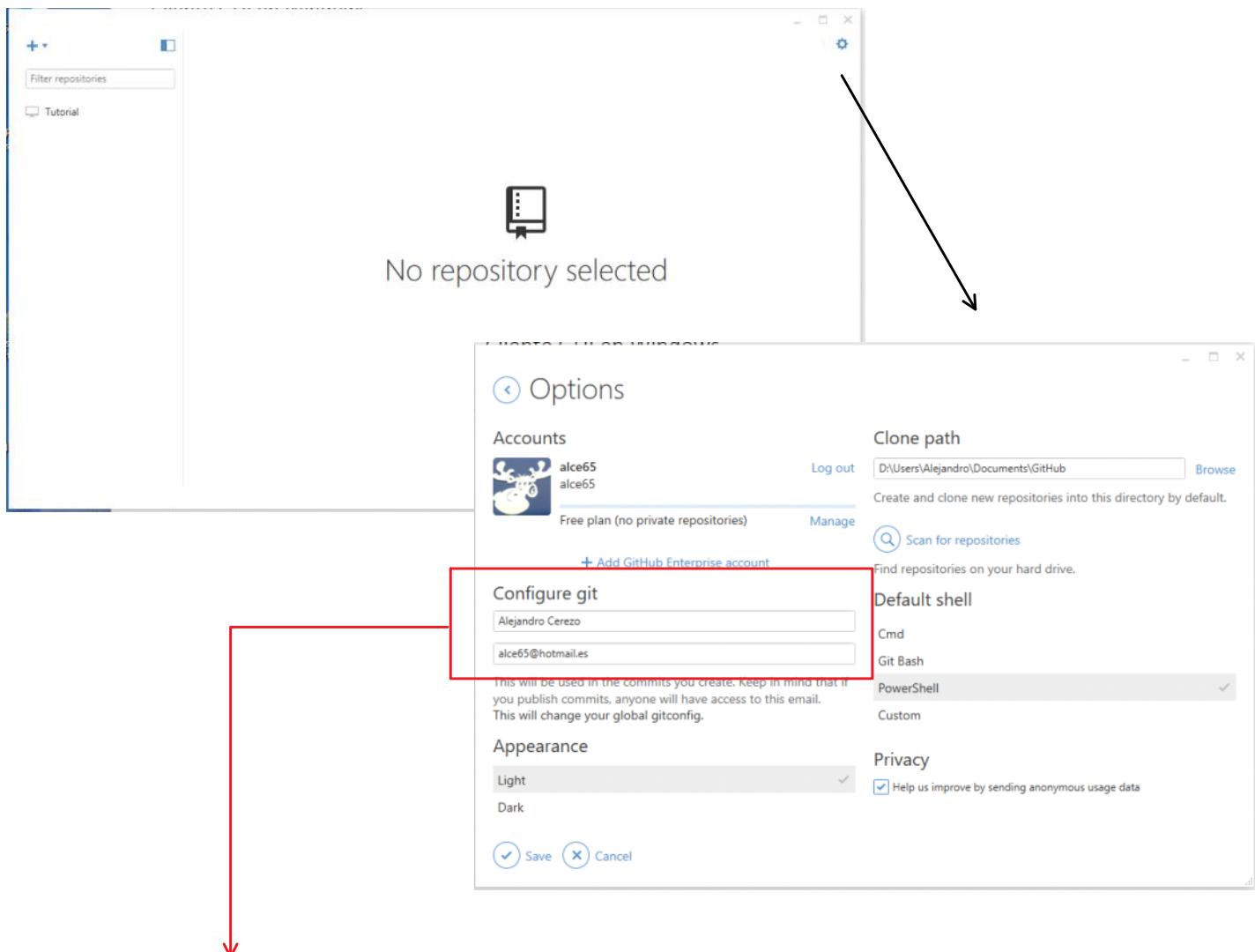
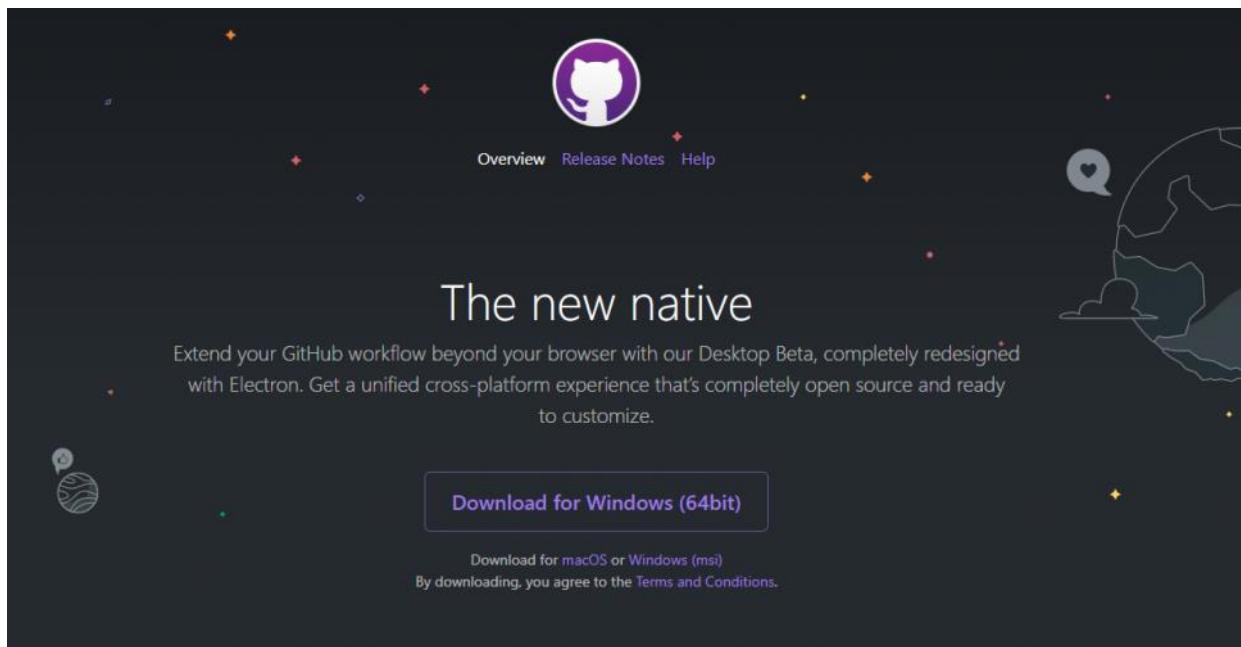


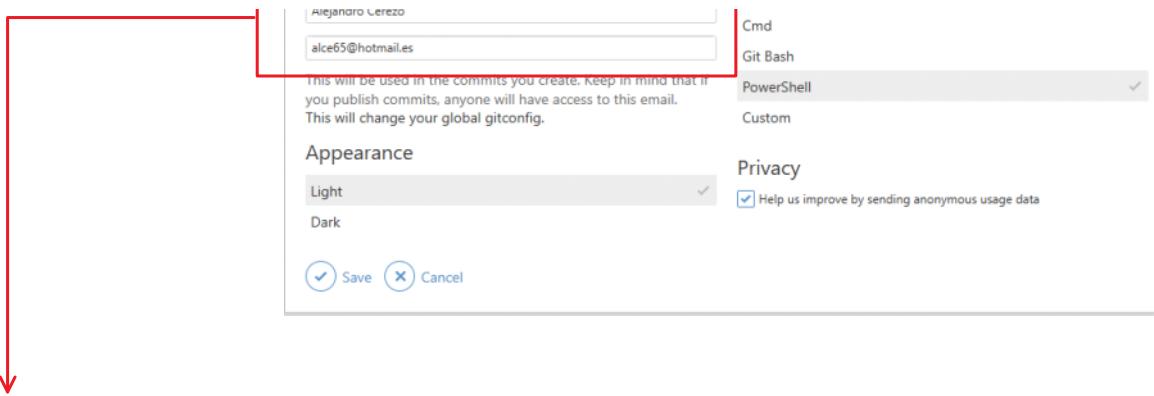
The screenshot shows the GitHub profile page for the user 'alce65'. The top navigation bar is identical to the sign-up page. The profile page includes a large profile picture of a reindeer, the user's name 'alce65', and a link to their repositories. Below this, there are three repository cards: 'Curso\_Angular2' (last updated 25 minutes ago), 'angularjs\_2017' (last updated on 11 Aug), and 'CursoPelayo' (last updated on 16 Jun). Each card shows the repository name, description, language (HTML), and update date.

# GUI para GitHub

domingo, 10 de septiembre de 2017 12:52

<https://desktop.github.com/>





Corresponde a los comandos de configuración de *git*

```
$ git config --global user.name "Pepe Perez"  
$ git config --global user.email pperez@example.com
```

# Repositorio en GitHub

domingo, 10 de septiembre de 2017 12:25

## Nuevo repositorio en GitHub

The screenshot shows the GitHub interface for creating a new repository. At the top, the search bar contains 'Search GitHub'. Below it, there are links for 'Pull requests', 'Issues', 'Marketplace', and 'Explore'. On the right side of the header are icons for notifications, a plus sign, and a user profile.

The main area is titled 'Create a new repository' with the sub-instruction 'A repository contains all the files for your project, including the revision history.' Below this, there are fields for 'Owner' (set to 'alce65') and 'Repository name' (set to 'Curso\_Angular2'). A note says 'Great repository names are short and memorable. Need inspiration? How about probable-spork.' Below the name field is a 'Description (optional)' field containing 'Curso de Angular2 en Icono Training Consulting'.

There are two radio button options for visibility: 'Public' (selected) and 'Private'. A note under 'Public' says 'Anyone can see this repository. You choose who can commit.' A note under 'Private' says 'You choose who can see and commit to this repository.'

A checked checkbox labeled 'Initialize this repository with a README' has a note below it saying 'This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.' Below this are buttons for 'Add .gitignore: Node' and 'Add a license: MIT License'.

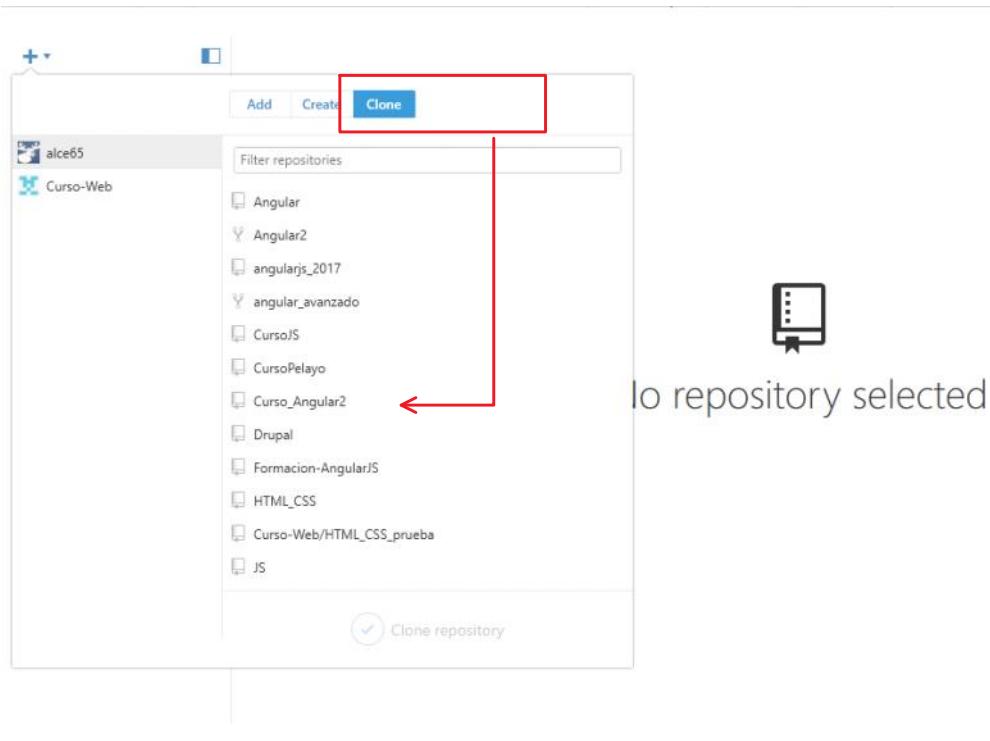
A large green 'Create repository' button is at the bottom of the form. A red arrow points downwards from this button to the repository details page below.

The repository details page shows the repository 'alce65 / Curso\_Angular2'. It includes sections for 'Code', 'Issues: 0', 'Pull requests: 0', 'Projects: 0', 'Wiki', 'Settings', and 'Insights'. The 'Code' section shows a single commit by 'alce65' with files '.gitignore', 'LICENSE', and 'README.md'. The 'About' section shows the repository name 'Curso de Angular2 en Icono Training Consulting' and the latest commit message 'alce65 initial commit'.

## Clonación local del repositorio

# Clonar un repositorio

domingo, 10 de septiembre de 2017 17:22



the repository selected

# Node y npm

sábado, 9 de septiembre de 2017 18:21

## JS en el servidor (SSJS): Node.js

<https://nodejs.org/>



Node.js® es un entorno de ejecución para JavaScript construido con el motor de JavaScript V8 de Chrome. Node.js usa un modelo de operaciones E/S sin bloqueo y orientado a eventos, que lo hace liviano y eficiente. El ecosistema de paquetes de Node.js, npm, es el ecosistema más grande de librerías de código abierto en el mundo.

Important security releases, please update now!

Descargar para Windows (x64)

v6.11.3 LTS

Recomendado para la mayoría

v8.4.0 Actual

Últimas características

Otras Descargas | Cambios | Documentación del API

Otras Descargas | Cambios | Documentación del API

Node.js® es un **entorno de ejecución para JavaScript** (una plataforma de software)

- Se emplea para construir aplicaciones de red escalables (especialmente servidores).
- Está construido con el motor de JavaScript V8 de Chrome
- Utiliza un modelo de operaciones que lo hace liviano y eficiente, gracias a
  - o **operaciones E/S sin bloqueo** y
  - o orientado a eventos, con un **bucle de eventos de una sola hebra**

Además, el **ecosistema de paquetes de Node.js, npm**, es el ecosistema más grande de librerías de código abierto en el mundo.

Su funcionalidad es especialmente adecuada en

- operaciones en tiempo real (e.g. chats)
- Bases de datos *NoSQL* / No relacionales

# Node.js: ampliación de JS



Al *core* de JS no le acompañan las APIs habituales en cualquier lenguaje de programación

→ Node.js puede entenderse como la ampliación de JS para llegar a ser un lenguaje "completo", independiente de un entorno huésped

## v8 (JavaScript)

- Una gramática que define la sintaxis del lenguaje
- Un intérprete/compilador que lo sabe interpretar y ejecutar
- Mecanismos para interactuar con el mundo exterior (llamadas al sistema)
- Librería estándar (consola, ficheros, red, etc...)
- Utilidades (intérprete interactivo, depurador, paquetes)

Node.js

También puede verse como un entorno huésped para JS en el servidor

# Node.js: orígenes



Tiene su origen en un proyecto de **Ryan Dahl** y sus colaboradores en la empresa *Joyent*, que fue presentado en una conferencia en la *JSConf* de 2009.

<https://youtu.be/ztspvPYybIY>

Escrito en C/C++ y JS

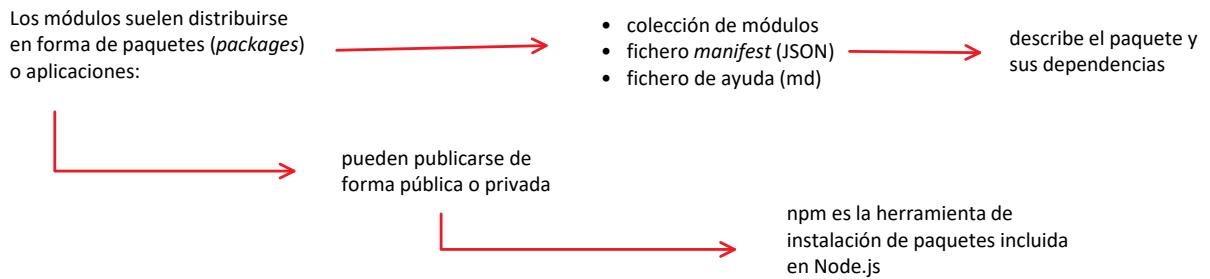
Objetivo del proyecto

→ Escribir aplicaciones muy eficientes en E/S con el lenguaje dinámico más rápido (v8) para soportar miles de conexiones simultáneas

Planteado sin complicaciones innecesarias

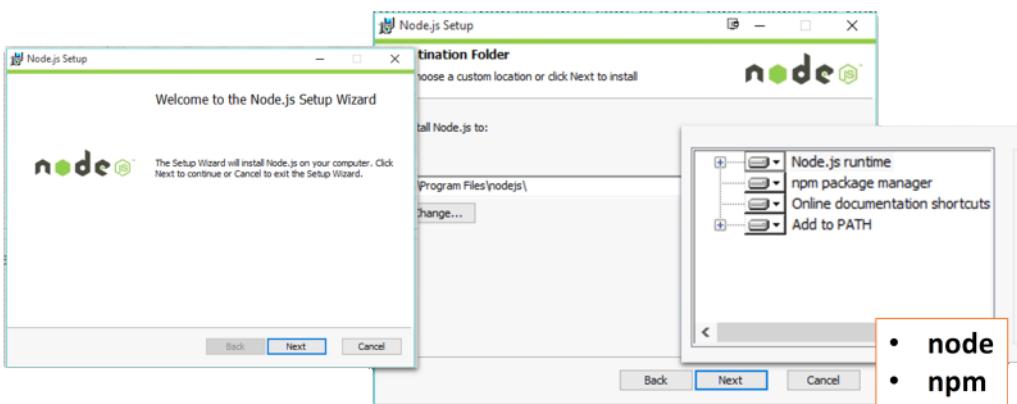
- Concurrencia sin paralelismo
- Lenguaje sencillo y muy extendido: JS
- API muy pequeña y muy consistente
- Apoyándose en Eventos y *Callbacks*

## Paquetes: npm



# Instalación de Node.js & npm

sábado, 9 de septiembre de 2017 20:35



v8.4.0 Current  
Latest Features

node-v8.4.0-x64.msi

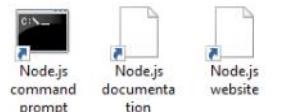
Welcome to the Node.js Setup Wizard

Installation Folder

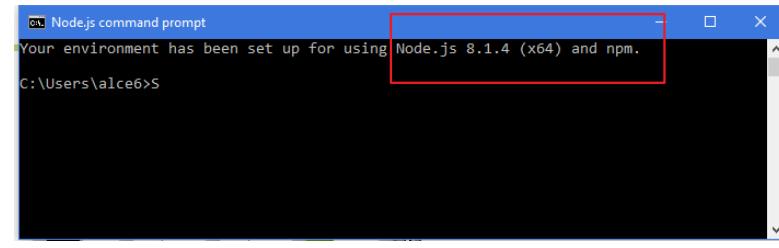
Select Node.js to:

- Program Files\nodejs
- Change...

• node  
• npm



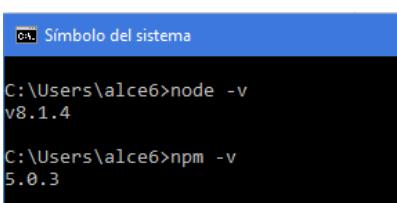
- Node.js command prompt
- Node.js documentation
- Node.js website
- Node.js
- Uninstall Node.js



Node.js command prompt

Your environment has been set up for using Node.js 8.1.4 (x64) and npm.

C:\Users\alce6>



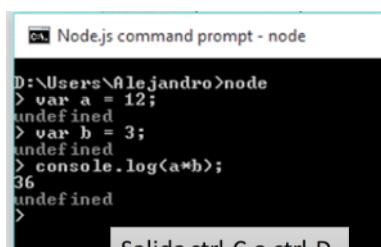
Símbolo del sistema

C:\Users\alce6>node -v

v8.1.4

C:\Users\alce6>npm -v

5.0.3



Node.js command prompt - node

```
D:\Users\Alejandro>node
> var a = 12;
undefined
> var b = 3;
undefined
> console.log(a+b);
36
undefined
>
```

Salida ctrl-C o ctrl-D

Con el comando "node" entramos en la consola de Node, un entorno REPL (Read-Eval-Print-Loop) similar a la consola de JS en los navegadores

El mismo comando node <fichero.js>, permite la ejecución de un fichero

# Configuración de proxies

lunes, 7 de agosto de 2017 21:38

## Configuración git

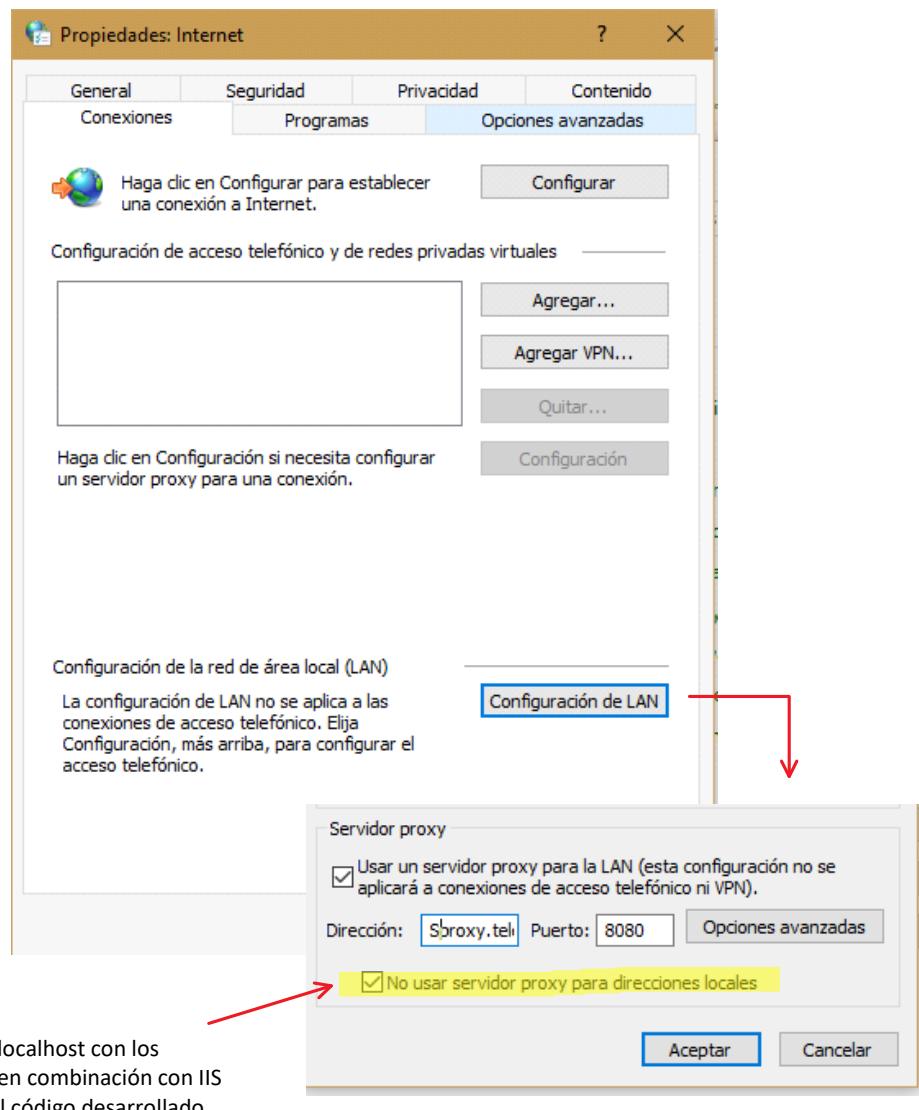
```
git config --global http.proxy http://user:passw@proxy.empresas.es:8080
```

## Configuración npm

```
$>npm config set proxy http://user:passw@proxy.empresas.es:8080
```

```
$>npm config set https-proxy http://user:passw@proxy.empresas.es:8080
```

## Propiedades de internet



Permite usar localhost con los navegadores en combinación con IIS para probar el código desarrollado

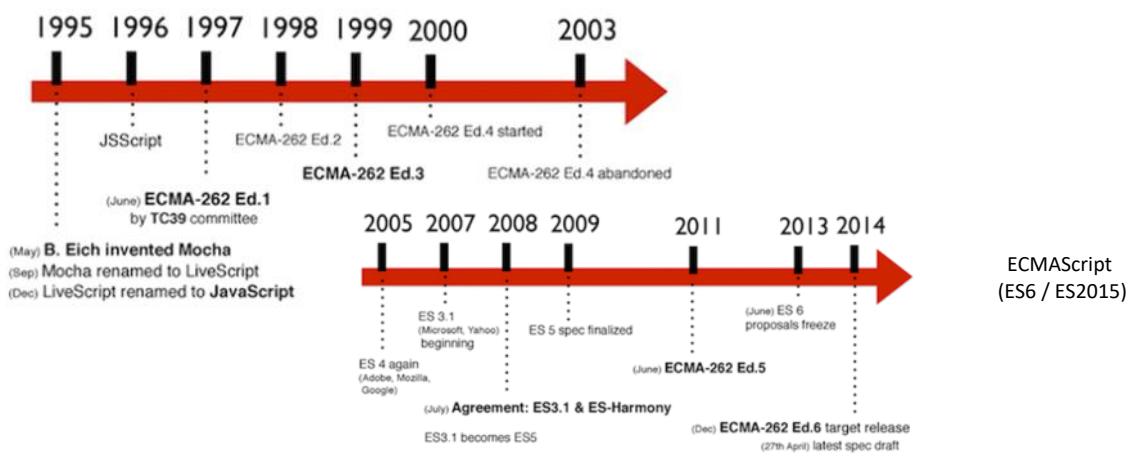
## ECMAScript 6 (ES6 / ES2015)

sábado, 9 de septiembre de 2017 13:33

Brendan Eich  
Netscape



European Computer Manufacturers' Association



Diciembre de 2014

- Constates (`const`). Variables con ámbito (`let`)
- Función Arrow. This "semántico"
- *Template Strings*: interpolación de variables
- Valores por defecto
- Clases (`class`)
- Módulos (`export / import`)
- Promesas (`promise`)
- *Destructuring* ...

# Más información

JS

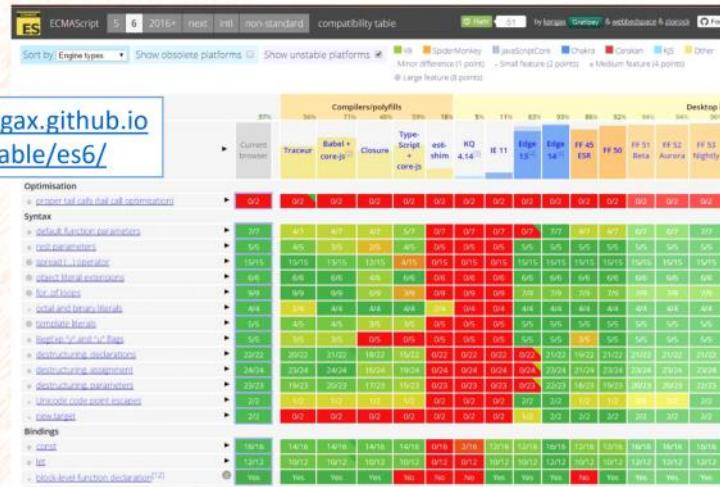
## ECMAScript 6 — New Features: Overview & Comparison

<http://es6-features.org/>

<http://kangax.github.io/compat-table/es6/>



TRAINING



## Nuevos elementos de código

domingo, 30 de julio de 2017 22:19

- **Constates (const).** Variables con ámbito (let)
- **Función Arrow.** This "semántico"
- **Template Strings:** interpolación de variables
- Valores por defecto

El uso de var sigue siendo identico a versiones anteriores, usandose en este caso para declarar un array

const y let

Salida por consola utilizando "template strings" en los que se conservan los saltos de línea

```
// Ejemplo de código en ES6
var data = [{precio: 12}, {precio: 34}, {precio: 19}];
data.forEach( elem => {
  if (true) {
    const iva = 1.16
    let precioFinal = elem.precio * iva
    console.log(`Oferta:
      El precio final es ${precioFinal}`);
  }
})
// console.log (iva)
```

la función callback del método forEach, propio de ES5 se define con el nuevo formato "**Arrow function**" con elem como único argumento

línea que daría error por hacer referencia a una variable en un ámbito en el que no existe

# Clases

sábado, 29 de julio de 2017 15:30

```
// Ejemplo de código en ES6
Clase "padre"      → class Libro {}

Clase que hereda de → class LibroTecnico extends Libro {
la anterior         

Constructor           → constructor(tematica, paginas) {
                           super(tematica, paginas);
                           this.capitulos = [];
                           this.precio = "";
                           // ...
                         }

Método que define   → metodo(pValor = "foo") {
valores por defecto    // ...
                         }
```

NO EXISTEN  
Propiedades definidas  
fuera de los métodos

Azúcar sintáctico.

En JS NO EXISTEN CLASES

Sólo hay PROTOTYPES

La nueva forma de escribir en ES6  
hace más sencillo el uso de los  
prototipos al asimilarlos a la forma  
habitual de trabajar con clases

# Módulos

sábado, 29 de julio de 2017 15:30

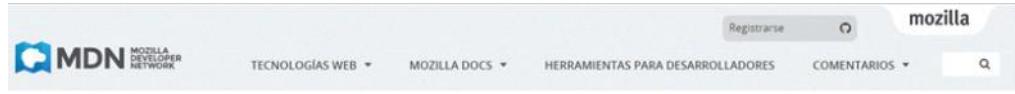
Creación de un módulo en el que se exporta una función, escrita en el nuevo formato "arrow function"

```
//File: lib/sample.js
definición de → module "sample" {
    un módulo
función →     export hello = (nombre) => {
    exportada      } return "Hola " + nombre;
}
}
```

Uso del módulo anteriormente creado

```
importación de → //File: app.js
una función → import { hello } from "sample";
objeto en el que se usa la → var app = {
función importada →     saludo : () => {
                           hello("Carlos");
                           }
}
app.saludo()
objeto exportado → export app;
```

NO DISPONIBLE EN  
LOS NAVEGADORES



<https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Sentencias>

La sentencia `import` se usa para importar funciones que han sido exportadas desde un módulo externo, otro script, etc.

 **Nota:** Esta característica aún no es implementada en ningún navegador por el momento. Esto es implementado en muchos transpiladores, tales como [Traceur Compiler](#) y [ES6 Module Transpiler](#).

La declaración `export` es usada para exportar funciones, objetos o tipos de dato primitivos a partir de un archivo (o módulo).

 **Note:** Esta característica no ha sido implementada de forma nativa todavía. Está implementada en algunos transpiladores, como [Traceur Compiler](#), [Babel](#) o [Rollup](#).

Una promesa representa el resultado eventual de una operación.  
Se utiliza para especificar que se hará cuando esa eventual operación de un resultado de éxito o fracaso.

## Promesas

JS

Un objeto promesa representa un valor que todavía no está disponible pero que lo estará en algún momento en el futuro

Permiten escribir código asíncrono de forma más similar a como se escribe el código síncrono:

- La función asíncrona retorna inmediatamente y → ese retorno se trata como un proxy cuyo valor se obtendrá en el futuro

El API de las promesas en Angular corresponde al servicio **\$q**

la biblioteca Q desarrollada por **Kris Kowal**

<https://github.com/kriskowal/q>



## Promesas: \$q

JS

```
function getPromise()
```

```
    var deferred=$q.defer();
```

crea una promesa

```
    deferred.resolve()  
    deferred.reject()
```

resuelve la promesa en un sentido  
u otro al cabo del tiempo

```
    return deferred.promise
```

devuelve la promesa

```
var promise = getPromise();
```

```
promise.then(successCallback,failureCallback,notifyCallback);  
promise.catch(errorCallback)  
promise.finally(callback)
```

promise.finally(callback)



# Promesas: ES6

JS

Implementación: new Promise

el objeto promesa recibe como parámetros dos funciones:

- La función "resolve": se ejecuta cuando queremos finalizar la promesa con éxito.
- La función "reject": se ejecuta cuando queremos finalizar una promesa informando de un caso de fracaso.

```
function hacerAlgoPromesa() {  
    return new Promise( function(resolve, reject) {  
        console.log('hacer algo que ocupa un tiempo...');  
        setTimeout(resolve, 1000);  
    })  
}
```



# Promesas: ES6

JS

## Utilización

A la función que retorna el objeto promesa se le encadenan dos:

- `.then` : la función que se ejecutará cuando la promesa haya finalizado con éxito.
- `.catch` : la función que se ejecutara cuando la promesa haya finalizado informando de un caso de fracaso.

```
hacerAlgoPromesa()
  .then( function() { console.log('la promesa terminó.');
  })
  .catch( function() { console.log('la promesa fracasó.');}
```



## Ejemplo de promesas en ES6

```
function msgAfterTimeout (msg, who, timeout) {
  return new Promise((resolve, reject) => {
    setTimeout(
      () => resolve(` ${msg} Hello ${who}!`),
      timeout)
  })
}

msgAfterTimeout("", "Foo", 100)
  .then((msg) =>
    msgAfterTimeout(msg, "Bar", 200))
  .then((msg) => {
    console.log(`done after 300ms:${msg}`)
```

Función que crea y devuelve un **objeto promesa**

En este caso, la promesa siempre se resuelve correctamente, creando un mensaje de saludo a un usuario

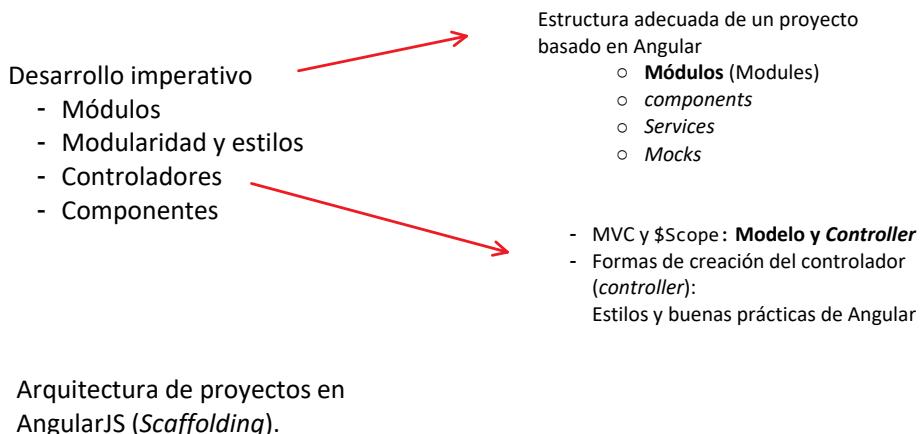
Utilización de las promesas, encadenando las llamadas a ellas

```
PS D:\Desarrollo\Angular2\Curso_Angular2\tecnologias\ES6> node .\sample.promise.js
done after 300ms: Hello Foo! Hello Bar!
PS D:\Desarrollo\Angular2\Curso_Angular2\tecnologias\ES6>
```

[https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos\\_globales/Promise](https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/Promise)

# Ξ Arquitectura de aplicaciones

sábado, 16 de septiembre de 2017 20:30



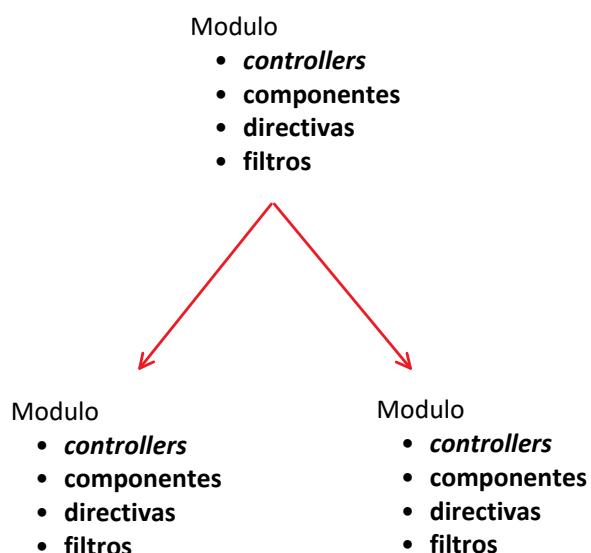
## Desarrollo imperativo: JavaScript

La forma de definir las funcionalidades mediante lenguajes de programación "tradicionales", en este caso **JS**, es lo que se denomina desarrollo imperativo o por procedimientos.

Todos los elementos de JS deben incluirse en un bloque <script>, que podría incluir el código en el propio fichero HTM, pero que en la práctica será siempre una referencia a un fichero externo.

Con independencia de su distribución en uno o varios ficheros, el código JS solo es accesible para Angular si pertenece a un **módulo** referenciado en la estructura **jerárquica de módulos** de la aplicación

Por tanto los **módulos** van a ser el elemento utilizado en la organización de todo el código JS. En ellos tendrán cabida cada uno de los demás elementos que componen la aplicación, comenzando por **controllers** y **componentes**, como piezas centrales del desarrollo imperativo es el :



# Módulos

sábado, 16 de septiembre de 2017 23:50

contenedor donde se sitúa el código de los controladores, componentes, directivas, etc., de forma que queda aislado, evitando colisiones con nombres repetidos en otras partes del código:

- aportan un espacio de nombres
- permiten que el código sea más reutilizable.

## Evolución

En JS, la implementación inicial de módulos fue definida en *CommonJS*. Posteriormente ha sido reinterpretada en entornos como Node.js o AngularJS. Finalmente gracias a los métodos *export / import* se ha incorporado al estándar de ES6, aunque aún no ha sido implementado por los navegadores.

## Creación de módulos en AngularJS

Utilizamos el método *module()* del objeto global angular, creado al cargar AngularJS mediante la directiva *ngApp*. Permite instanciar el objeto correspondiente al módulo.

### HTML

```
<body ng-app="appMain"> ← index.html carga angular con una referencia a cuál  
será el módulo principal de la aplicación
```

### Código JS

```
var oModulo = angular.module('miAplicacion',  
    [ ... ],  
    function(...){ ... }) ← 3 argumentos:  
    ↑  
    • nombre de la aplicación  
    • [ inyección de dependencias, e.g. otros módulos ]  
    • función anónima opcional que configura el módulo)
```

En lugar de "cachearlo" como una variable, el acceso posterior al módulo se puede realizar con el mismo método que lo crea pasándole como único argumento el nombre del módulo.

### Instanciación

```
angular.module('miAplicacion', [ ... ], function(...){ ... }) ← El método module utiliza un patrón  
factoría (factory), y nos devuelve  
una instancia de la "clase" module  
(Objeto module)
```

### Posterior uso

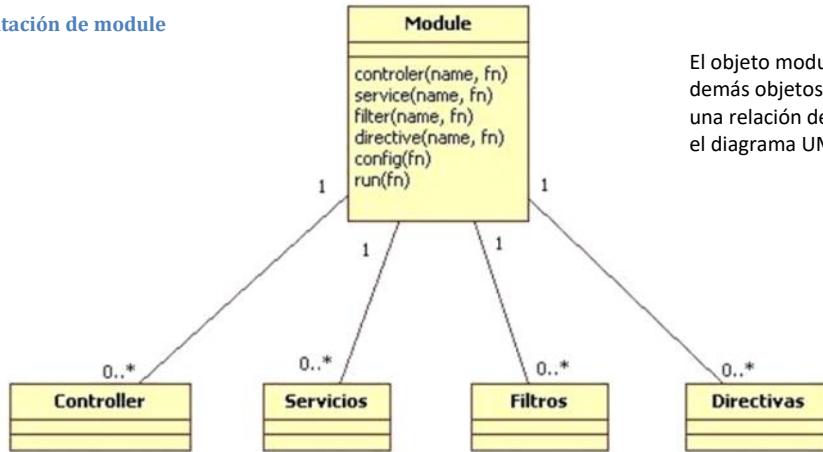
```
angular.module('miAplicacion').  
metodo(...).  
metodo() ← La sintaxis denominada estilo "Fluent"  
concatena tras la instanciación los  
métodos invocados del objeto
```

El objeto module tiene una serie de métodos que permiten controlar la lógica de la aplicación.

Estos métodos devuelven al propio objeto que los invoca, con lo que pueden encadenarse sucesivamente.

- *config*
- *run*
- *controller*
- *component*
- *directive*
- *constant*
- *value*
- *service*
- *factory*
- *provider*

### Representación de module



El objeto module instancia como atributos todos los demás objetos de la aplicación, creando con ellos una relación de asociación, tal como se muestra en el diagrama UML

### Módulos y ficheros

Esto permite distribuir el código JS de un módulo en tantos ficheros como sea necesario

Fichero del módulo:

- nombre.module.js

```
angular.module('miAplicacion', [ ... ], function(...){ ... })
```

Otros ficheros:

- nombre.controller.js
- nombre.componente.js
- nombre.servicio.js ...

```
angular.module('miAplicacion')
```



## Ficheros y módulos

En cualquiera de las distribuciones, hay que diferenciar 2 procesos

- distribuir el código en módulos
- distribuir el código en ficheros

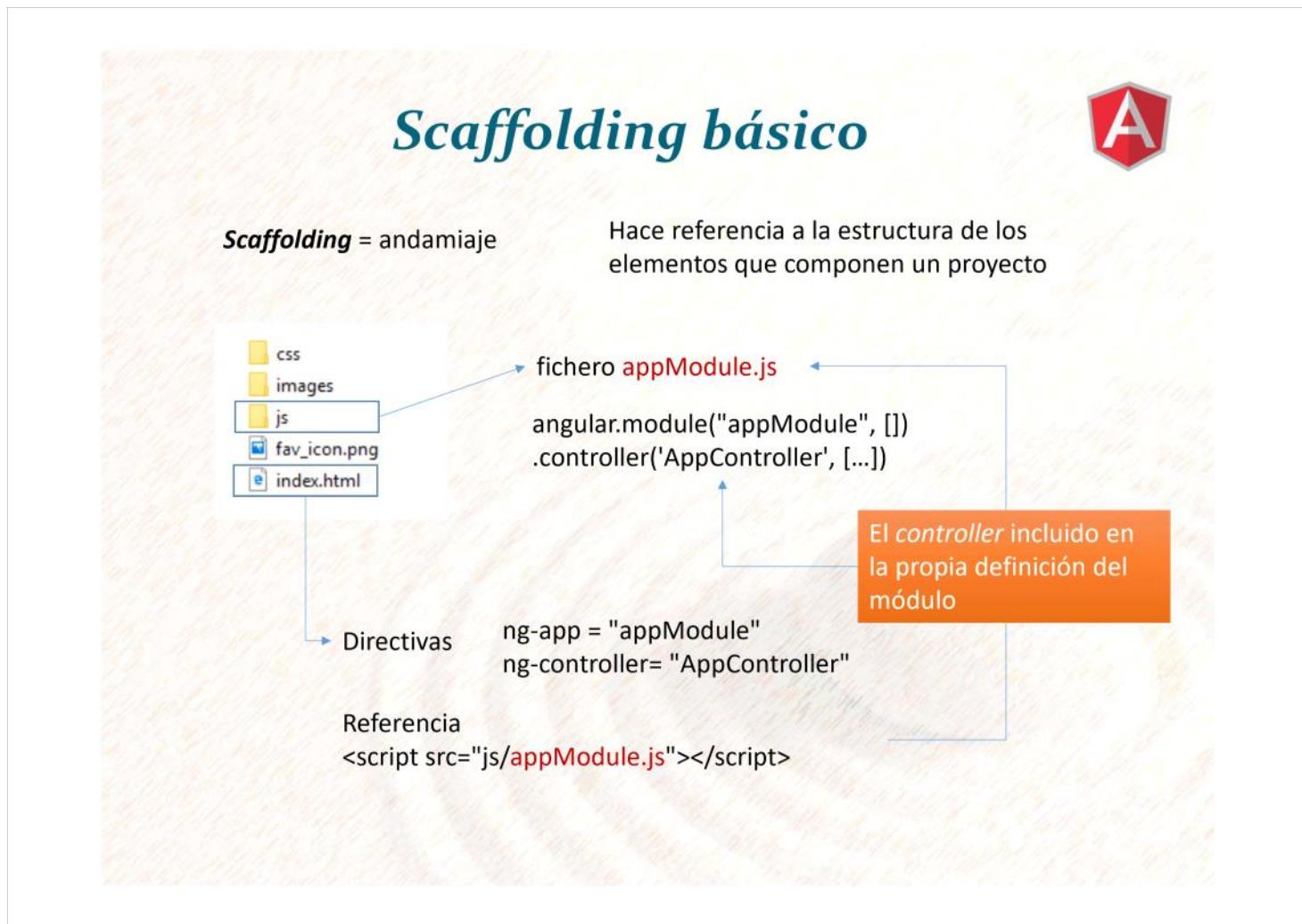
index.html → ng-app='mainApp'

app.js → 

```
angular.module('mainApp',  
['loginModule','commentModule']);
```

loginModule.js → 

```
angular.module('loginModule',  
['mainApp.login.controllers',  
'mainApp.login.directives']);
```



## Ejemplo

**Acumulador**

**Control de operación:**

Incremento

+
-

**Totales:**

En el acumulador llevamos 10

```
Acumulador_Modulos
  ▲ css
  # default.css
  ▲ js
    JS acumulador.controller.js
    JS app.js
  fav_icon.png
  index.html
```

# Modularidad



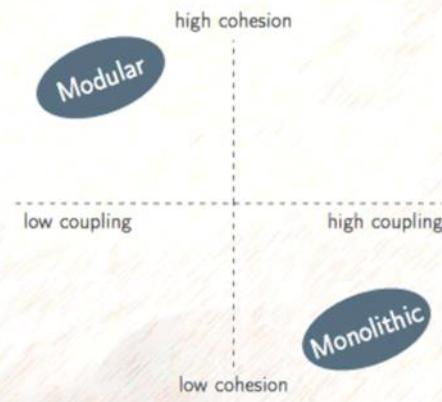
**cohesión** es la medida de lo que los módulos hacen

**acoplamiento** es la medida de la dependencia entre los distintos módulos

En Angular

```
angular.module("appModule", [...])
```

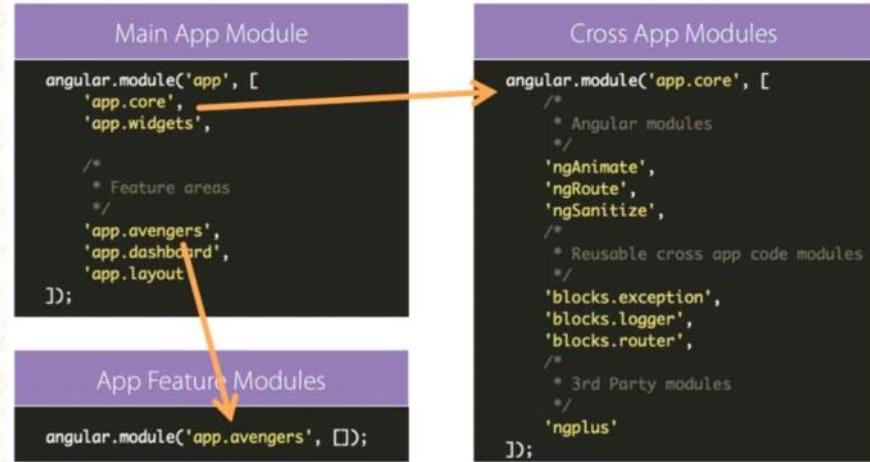
Reflejo en la guía de estilo de John Papa



Modularity and AngularJS -  
João Figueiredo

<https://medium.com/@lucalanca/modularity-and-angularjs-454e457c6df#.3ksndvsmx>

# Módulos según John Papa



Módulo APP [Style Y161]

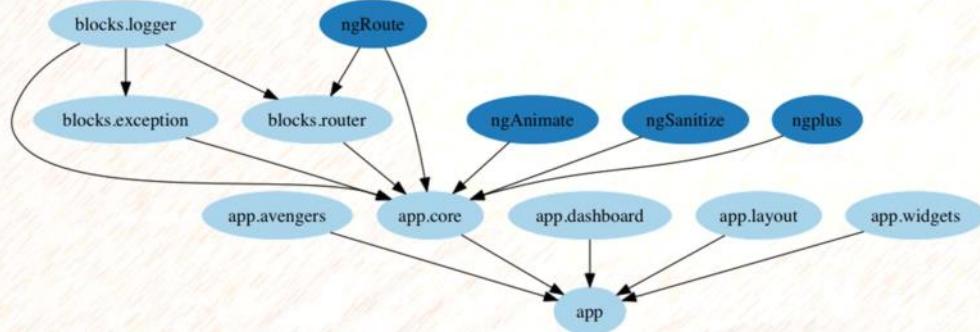
Módulos de "características" [Style Y163]

Módulos reutilizables [Style Y164]

Nomenclatura

*Scaffolding elegido*

# Diagramas de módulos



A screenshot of a GitHub repository page for 'lucalanca/grunt-angular-architecture-graph'. The repository has 93 commits, 2 branches, 6 releases, and 5 contributors. It was last updated on April 10, 2015. The page shows options to 'New pull request', 'Find file', and 'Download ZIP'. The URL of the repository is <https://github.com/lucalanca/grunt-angular-architecture-graph/>.

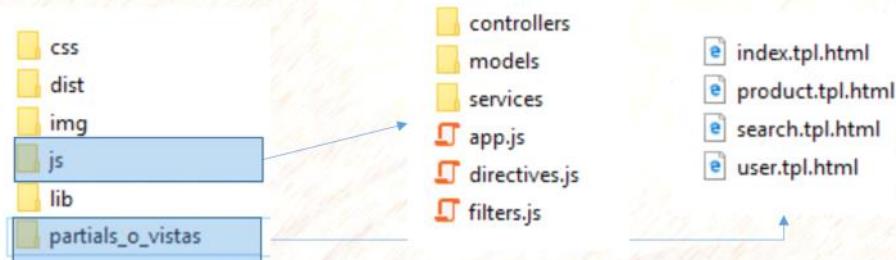
## Estructura de carpetas

domingo, 17 de septiembre de 2017 0:53



# Scaffolding por capas

## Distribución por capas



assets = css + img

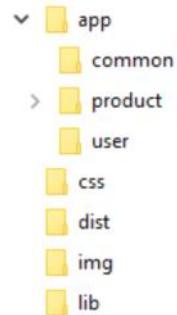
dist: imprescindible si hay procesos automatizados (grunt)

partials o views: se recomienda alguna nomenclatura para distinguir con precisión las vistas y la ruta a la que corresponden

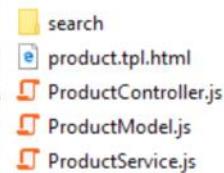
# Scaffolding por características



## Distribución por características



En proyectos más complejos se agrupan los elementos correspondientes a sus distintas características, e.g. las opciones del menú principal



Cada bloque incluye los archivos o incluso carpetas correspondientes a cada capa:

- *controller*
- *model*
- *partials*
- ...

# *Scaffolding* y plantillas

domingo, 17 de septiembre de 2017 0:58

## Generador es de plantillas

- ❑ Angular seed  
<https://github.com/angular/angular-seed>
- ❑ ngBoilerplate  
<https://github.com/ngbp/ngbp>
- ❑ Yeoman  
<http://yeoman.io/>
- ❑ CleverStack  
<http://cleverstack.io/>

The screenshot shows the GitHub repository page for 'angular / angular-seed'. The page features a large title 'Angular seed' with a red Angular logo to its right. Below the title, a subtitle reads: 'esqueleto de una aplicación desarrollada por el propio equipo de AngularJS, con el respaldo de Google, como punto de partida para proyectos'. A link 'https://github.com/angular/angular-seed' is provided. The GitHub interface includes a search bar, navigation links for Explore, Features, Enterprise, Pricing, and user buttons for Sign up and Sign in. The repository summary shows 179 commits, 3 branches, 0 releases, and 41 contributors. A detailed commit history is listed, with one commit by 'petebacondarwin' highlighted: 'chore(protractor): update to use protractor v2.1.0'. The commit was made on 15 Jun, 6 months ago. A circled 'HTTPS clone URL' link is shown at the bottom right of the commit list.

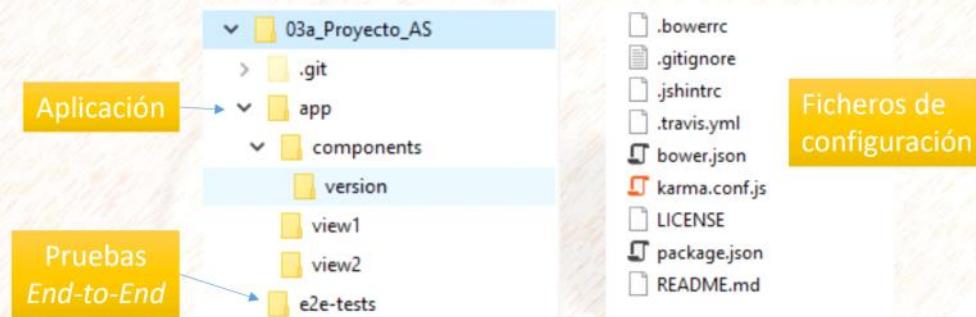
# Angular seed: Instalación (1)



**git clone origen destino**

```
git clone https://github.com/angular/angular-seed.git 03a_Proyecto_AS
```

```
D:\Users\Alejandro\Mi nube\OneDrive\Desarrollo\Angular>git clone https://github.com/angular/angular-seed.git 03a_Proyecto_AS
Cloning into '03a_Proyecto_AS'...
remote: Counting objects: 2590, done.
Receiving objects: 100% (2590/2590), 12.21 MiB | 4.16 MiB/s, done.
Resolving deltas: 1% (14/1370)   0 (delta 0), pack-reused 2590
Resolving deltas: 100% (1370/1370), done.
Checking connectivity... done.
```

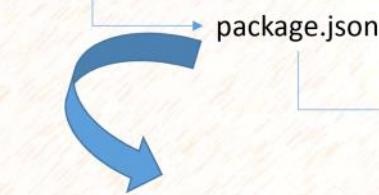


No incluye aún ninguna librería

## Angular seed: Instalación (2)



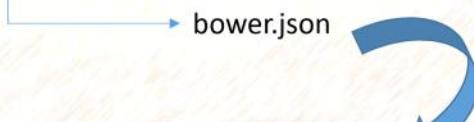
npm install



```
"bower": "^1.3.1",
"http-server": "^0.6.1",
"jasmine-core": "^2.3.4",
"karma": "~0.12",
"karma-chrome-launcher": "^0.1.12",
"karma-firefox-launcher": "^0.1.6",
"karma-jasmine": "^0.3.5",
"karma-junit-reporter": "^0.2.2",
"protractor": "^2.1.0",
"shelljs": "^0.2.6"
```

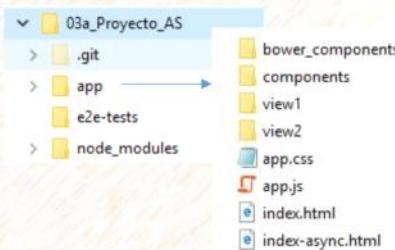
Instalación de las librerías necesarias

incluye y ejecuta bower



```
"angular": "~1.4.0",
"angular-route": "~1.4.0",
"angular-loader": "~1.4.0",
"angular-mocks": "~1.4.0",
"html5-boilerplate": "~5.2.0"
```

# Proyecto con Angular seed



**npm start**

```
> http-server -a localhost -p 8000 -c-1
Starting up http-server, serving ./ on port: 8000
Hit CTRL-C to stop the server
```

En el navegador

<http://localhost:8000/app>

Podemos modificar el contenido

Accedemos a  
index.html vía *localhost*

- usando IIS
- usando el servidor Node.js incluido y ya configurado

[ [view1](#) | [view2](#) ]

This is the partial for view 1.

Angular [ [view1](#) | [view2](#) ]

Este es el "parcial" para la vista 2.

The screenshot shows the GitHub repository page for `ngbp / ngbp`. The page title is **ngBoilerplate**, with a large Angular logo icon to the right. Below the title, it says "esqueleto de una aplicación desarrollada ...". A link to the repository is provided: <https://github.com/angular/angular-seed>.

The GitHub interface includes a search bar, navigation links for Explore, Features, Enterprise, and Pricing, and buttons for Sign up and Sign in.

The repository summary indicates 103 commits, 6 branches, 1 release, and 16 contributors. The branch dropdown is set to `v0.3.2-release`. The commit list shows recent activity, with a commit from 25 Jan highlighted. A callout box points to the HTTPS clone URL, which is <https://github.com/ngbp/ngbp>. Other options shown include Clone in Desktop and Download ZIP.

**Incluye**

- Bootstrap
- UI Bootstrap
- Angular UI
- Font Awesome
- LESS
- Grunt
- Angular Placeholders



# ngBoilerplate: Instalación

**git clone origen destino**

```
git clone https://github.com/ngbp/ngbp.git 03b_Proyecto_ngB
```

Es necesario tener instalado / instalar globalmente

Si ya tenemos los dos primeros, en vez de

- grunt
- bower
- karma

```
npm -g install grunt-cli karma bower
```

Ejecutamos      **npm -g install karma**

A nivel del proyecto, completamos todas las dependencias

**npm install** —————→ lee el fichero package.json

**bower install** —————→ lee el fichero bower.json



# ngBoilerplate: Uso

Como está basado en *Grunt*, es necesario ejecutar  
(sin cerrar la correspondiente consola)

## grunt watch

- supervisa constantemente las carpetas *src*, donde se debe hacer cualquier modificación
- actualiza el contenido de la carpeta *build*, donde se deben comprobar los resultados
- levanta un servidor web
- inicia Firefox con un acceso a karma, para las pruebas



- inicia LiveReload

# Proyecto con ngBoilerplate

The screenshot shows the GitHub repository for ngBoilerplate. The landing page title is "Non-Trivial AngularJS Made Easy". It features a "Good to Go!" badge, a "Complete Build System" badge, and a "Modularization" badge. A callout box highlights: "Comprobamos como se refleja en ella cualquier cambio en los ficheros .tpl.html de la carpeta src".

Accedemos a la carpeta *build* del proyecto

Comprobamos como se refleja en ella cualquier cambio en los ficheros .tpl.html de la carpeta src

# Yeoman



Ecosistema de generadores de código y de proyectos de distintos lenguajes y plataforma, incluyendo uno específico para Angular.JS

<http://yeoman.io/>

A screenshot of the Yeoman website homepage. The header features the Yeoman logo (a cartoon character wearing a top hat) and the word "YEOMAN". The main navigation links are "Using Yeoman", "Discovering generators", "Creating a generator", "Blog", and "Contributing". The main content area has a teal background with the text "THE WEB'S SCAFFOLDING TOOL FOR MODERN WEBAPPS". To the right is a cartoon illustration of three characters working on a large white machine with a red cross and a circular dial. Below the main content is a light gray sidebar with the text "Get started and then [find a generator](#) for your webapp. Generators are available for [Angular](#), [Backbone](#), [React](#), [Polymer](#) and over [1500+ other projects](#). One-line install using [npm](#): `npm install -g yo`".



# Yeoman: instalación (1)

Dependencias previas

- Git
  - Node.js
  - Ruby
  - Compass
- 

Pre-procesador  
SaSS

Si ya tenemos los *Grunt*, *Bower*, en vez de

```
npm -g install yo grunt-cli bower
```

Ejecutamos      **npm -g install yo**

A continuación se instalan los generadores requeridos, de los que existen para *Yeoman*; en este caso los de *Angular* y *Karma*

```
npm -g install generator-karma generator-angular
```



# Yeoman: instalación (2)

Creamos la carpeta del proyecto y en ella ejecutamos

**yo angular <nombre\_proyecto>**

.../03c\_Proyecto\_Yo>yo angular 03c\_Proyecto\_Yo

El interface permite seleccionar fácilmente las opciones presentadas

En un momento, la instalación parece quedar detenida ; hay que pulsar enter aunque no se indica

A screenshot of a terminal window showing the Yeoman configuration process. The window title is 'Welcome to Yeoman, ladies and gentlemen!'. It displays a list of recommended modules with checkboxes:

- ( ) angular-animate.js
- ( ) angular-aria.js
- (\*) angular-cookies.js
- (\*) angular-resource.js
- ( ) angular-messages.js
- (\*) angular-route.js
- (\*) angular-sanitize.js
- (\*) angular-touch.js

The user has selected 'angular-cookies.js', 'angular-resource.js', 'angular-route.js', and 'angular-sanitize.js'. The terminal also shows some configuration questions at the top:

```
But of the box I include Bootstrap and some AngularJS recommended modules.  
Would you like to use Gulp (experimental) instead of Grunt? No  
Would you like to use Sass (with Compass)? No  
Would you like to include Bootstrap? Yes  
Which modules would you like to include? (Press <space> to select)
```



## Yeoman: instalación (3)

```
Done, without errors.

Execution Time (2015-11-28 19:33:40 UTC)
loading tasks      534ms [██████████] 50%
loading grunt-wiredep 14ms [████] 1%
wiredep:app     472ms [██████████] 44%
wiredep:test     39ms [██] 4%
Total 1.1s
```

Una vez concluido ejecutamos **grunt**, para que realice todas las tareas definidas en Gruntfile.js

Finalmente **grunt serve** se encarga de levantar un servidor node.js en determinado puerto (e.g. 9000)

# Proyecto con Yeoman

Nuevamente disponemos en el proyecto de app (donde trabajamos) y *dist* (mantenida por *grunt*) y actualizada con los cambios que hagamos gracias a *LiveReload*.

Vemos en el ejemplo el modo responsive tras haber modificado la vista `home.html`

03cProyectoYo Home About Contact

'Allo, 'Allo!

03cProyectoYo

Always

Curso de Angular



Always a pleasure scaffolding your apps.

Splendid! ✓

HTML5 Boilerplate

HTML5 Boilerplate is a professional front-end template for building fast, robust, and adaptable web apps or sites.

Angular

AngularJS is a toolset for building

HTML5 Boilerplate

HTML5 Boilerplate is a professional front-end template for building fast, robust, and adaptable web apps or sites.

Karma

Spectacular Test Runner for Java

Angular

AngularJS is a toolset for building the framework most suited to your application development.

Angular

AngularJS is a toolset for building

Karma

Spectacular Test Runner for JavaScript.

♥ from the Yeoman team



# Yeoman: generadores

Yeoman dispone además de diversos generadores de código

<https://github.com/yeoman/generator-angular>

angular:controller  
angular:directive  
angular:filter  
angular:route  
angular:service  
angular:provider  
angular:factory  
angular:value  
angular:constant  
angular:decorator  
angular:view

Para ejecutarlos:

- detenemos el servidor web levantado por **grunt**
- ejecutamos yo y el generador que necesitamos

**yo angular:view <nombre>**

Crearía la correspondiente vista nueva en la carpeta adecuada

# Controllers

domingo, 17 de septiembre de 2017 0:57