

Ξ Angular. Primera parte

sábado, 9 de septiembre de 2017 13:22

- Introducción a Angular
 - *Frameworks en JS*
 - Presentación de AngularJS y Angular
- Tecnologías implicadas
 - Entorno de trabajo
 - Instalación e inicio. Angular cli
 - ES6
 - *TypeScript*

Navegadores.
Editores de código
Gestión de versiones. GIT
NodeJS y npm. Empaquetado

Formas de creación de proyectos
Arquitectura del proyecto (*Scaffolding*). Angular-cli

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/>



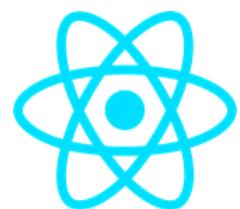
Ember.js

<http://emberjs.com/>



AngularJS

<https://angularjs.org>



React.js

<http://emberjs.com/>

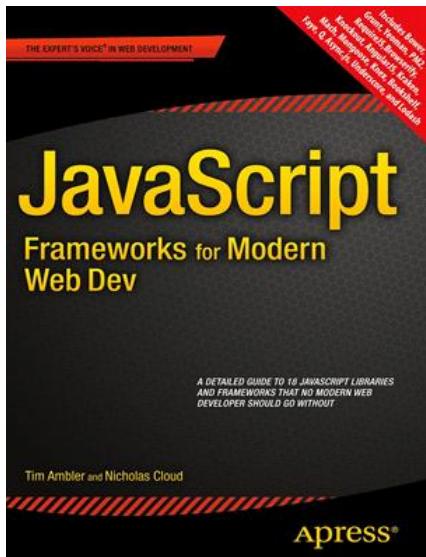
La elección de uno de ellos depende de los objetivos de cada proyecto



<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



Angular

Introducción a Angular

sábado, 9 de septiembre de 2017 16:48

The screenshot shows the AngularJS website. At the top is the AngularJS logo with the text "ANGULARJS by Google". Below it is the tagline "HTML enhanced for web apps!". There are two main calls-to-action: "Download AngularJS 1" (with a link to "1.6.9rc2 / 1.5.8 / 1.2.37") and "Try the new Angular 2" (with a link to "1.6.9rc2 / 1.5.8 / 1.2.37"). Below these are buttons for "View on GitHub" and "Design Docs & Notes". Social media links for GitHub, Facebook, and Twitter are present, along with a "Follow @AngularJS" button. A "Learn Angular in your browser for free!" button is also visible.

<https://angularjs.org/>

The screenshot shows the Angular website. The header includes the Angular logo and navigation links for FEATURES, DOCS, EVENTS, and NEWS. A large red "A" logo is centered on a blue background with the text "One framework. Mobile & desktop." Below it is a "GET STARTED" button. A banner at the bottom promotes "Google Developer Day Beijing & Shanghai 12/2016" with a "REGISTER NOW" button.

<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 website. The header includes the MEAN logo and links for Home, Documentation, Packages, Release Notes, Support, Blog, and Contact. The main content features the tagline "The Friendly & Fun Javascript Fullstack for your next web application". It describes MEAN as an opinionated fullstack javascript framework that simplifies and accelerates web application development. Below this is a section titled "Get MEAN by running..." with the command "\$ sudo npm install -g mean-cli" and "\$ mean init yourNewApp". At the bottom, there are links for "LATEST RELEASE: v0.6.5", "LATEST COMMIT: Nov 23, 2015", and "FORKS: 2296".

Ejemplos de uso

madewithANGULAR

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

Communication



Education



SEE ALL



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*))
 - está 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*, *Karma* y *Protractor*
 - 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*

Aplicaciones cada vez más ambiciosas

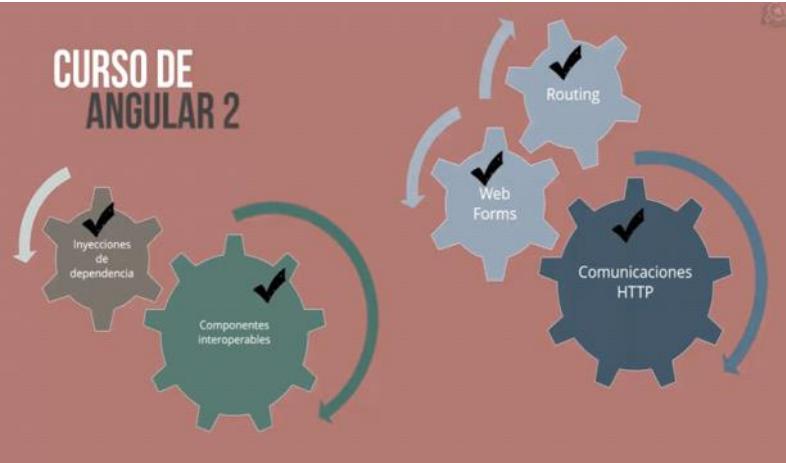


Angular 2.x
Angular 4.x
Angular 5.x

- amplia el modelo de **extender** las funcionalidades **HTML** empleando **componentes**
 - Árboles de componentes Web
 - Interconexiones entre ellos
 - Cada uno su propia interface I/O
 - Inyección de dependencias totalmente renovado
- con ello se modifica la forma de emplear el patrón arquitectónico **MVC** (Modelo, Vista, Controlador) o similares (estrictamente sería MV* o MVW (*Model-View-Whatever*))
- sigue estando especialmente orientado a la creación de aplicaciones **SPA** (*Single-Page Applications*).



CURSO DE ANGULAR 2



- Angular 2 Versión final: Septiembre 2016
Angular 4 : Abril 2017
- Está implementado desde cero no como una evolución de AngularJS
- Angular 2 no es compatible con AngularJS: no existe el `$scope`
- La documentación de AngularJS no sirve para Angular

Tabula Rasa



Funcionalidades en Angular

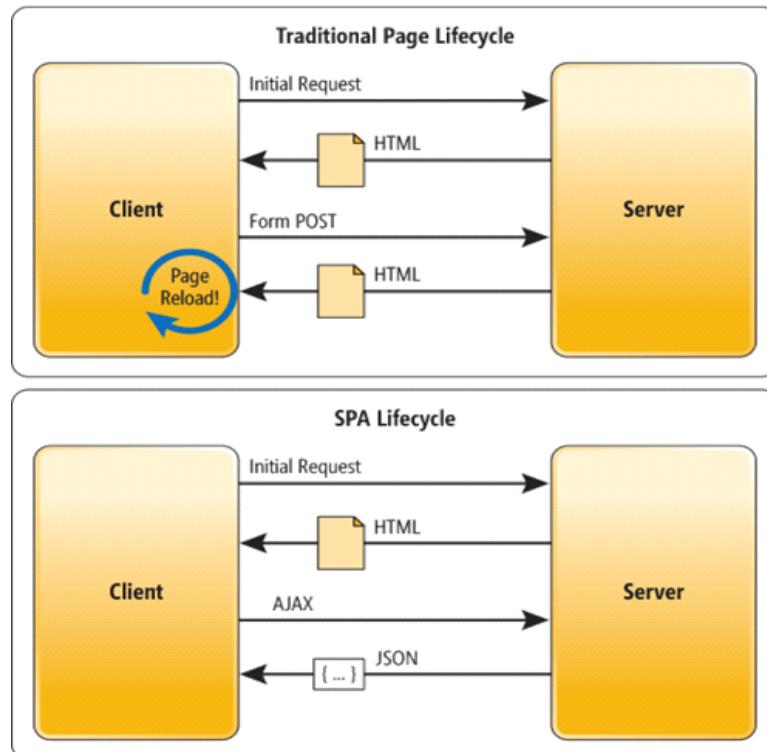
- Inyección de dependencias
- Servicios
- Cliente http (APIs REST)
- Navegación por la app (*Router*)
- Animaciones
- Internacionalización
- Soporte para tests unitarios y e2e
- Librerías de componentes
- Renderizado en el servidor (Angular Universal)



SPA: *Single-Page Applications*

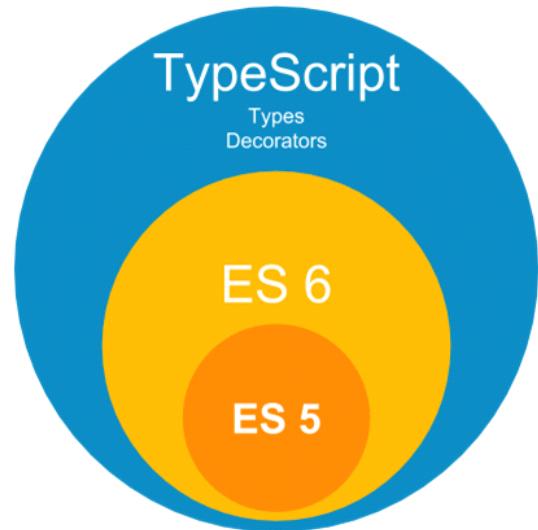
sábado, 9 de septiembre de 2017 17:13

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



Programación en ES6 + Typescript

- ES6
 - let (variables con ámbito)
 - clases (class)
 - módulos (import y export)
 - funciones arrow
- TypeScript
 - tipos
 - anotaciones



Transpilación



resultados en ES5 / ES6
(compatibilidad)



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

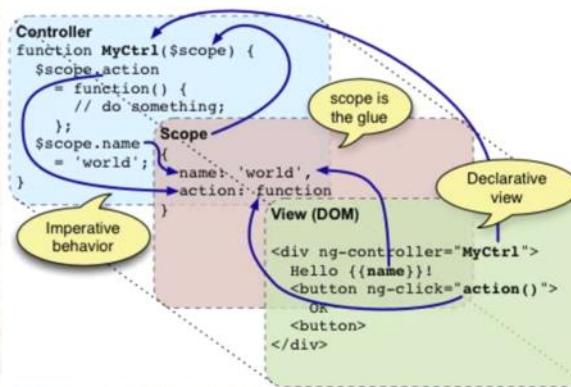


Model – View - Whatever



- **Vistas:** Será la representación de los datos o la información, es decir, el código del interfaz de usuario, básicamente el DOM/HTML/CSS .
- **Controladores:** Se encargarán de la lógica de la aplicación, incluyendo "Factorías" y "Servicios" para mover datos contra servidores o memoria local en HTML5.
- **Modelo o Modelo de la vista,** según se emplee la variante del patrón MVC o MVVC.

El modelo 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.



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 está disponible un *polyfile* para que puedan usarse en otros.

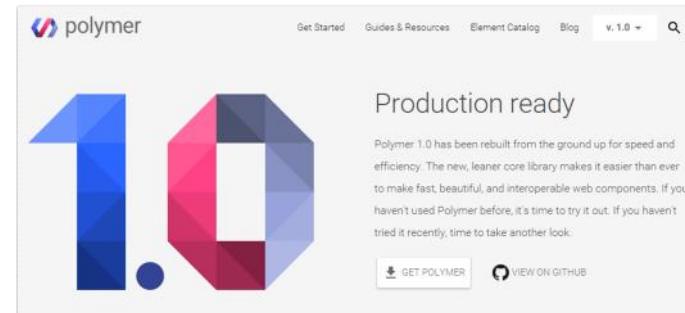
Constan de cuatro especificaciones:



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



<http://webcomponents.org/>



<https://www.polymer-project.org/1.0/>

Web Components 1

domingo, 10 de diciembre de 2017

21:07

Adopta la sintaxis de ES6

Custom elements

```
class AppSample extends HTMLElement {  
    constructor () {  
        super()  
        console.log("Creado el componente")  
    }  
  
    customElements.define('app-sample', AppSample)
```

Shadow DOM

```
class AppSample extends HTMLElement {  
    constructor () {  
        super()  
        console.log("Creado el componente")  
        this.attachShadow({mode: 'open'}).innerHTML  
            = "<h1>Componente Web</h1>"  
    }  
}  
customElements.define('app-sample', AppSample)
```

Templates

```
<template id="temp1">  
    <style>  
        h1 { color: orange; }  
    </style>  
    <h1>Hola Mundo</h1>  
    <p>  
        Lorem ipsum dolor sit,  
        amet consectetur adipisicing elit.  
        Reiciendis, facilis magnam. Beatae, maxime itaque?  
        Id unde corrupti vero, eligendi obcaecati ullam placeat  
        ducimus sint, iste cumque neque non iure consequatur.  
    </p>  
</template>
```

HTML Imports

```
<link rel="import" href=".//templete.html">
```

Resultado final

```
class AppSample extends HTMLElement {  
    constructor () {  
        super()  
  
        const oImport =  
            document.querySelector('link[rel="import"]').import;  
        const oElem = oImport.querySelector('#temp1')  
  
        this.attachShadow({mode: 'open'}).innerHTML = oElem.innerHTML  
    }  
}
```

Referencia al *template* importado para almacenarlo como un objeto de tipo elemento del DOM

```
}
```

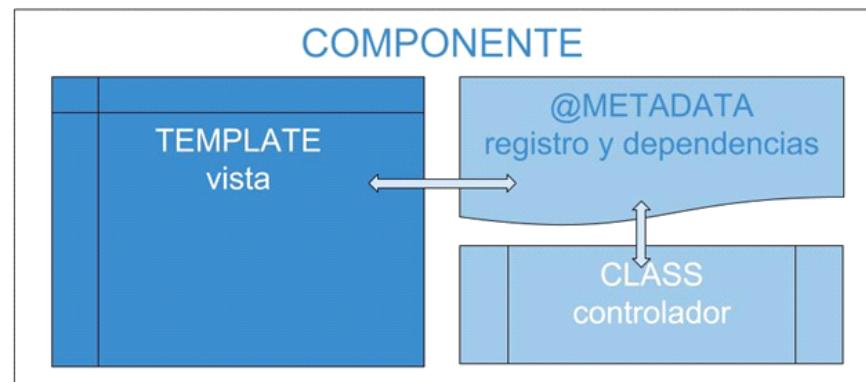
```
customElements.define('app-sample', AppSample)
```

Componentes en Angular

domingo, 1 de octubre de 2017 11:55

Componentes en Angular

- **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 componentes definidos en AngularJS 1.5



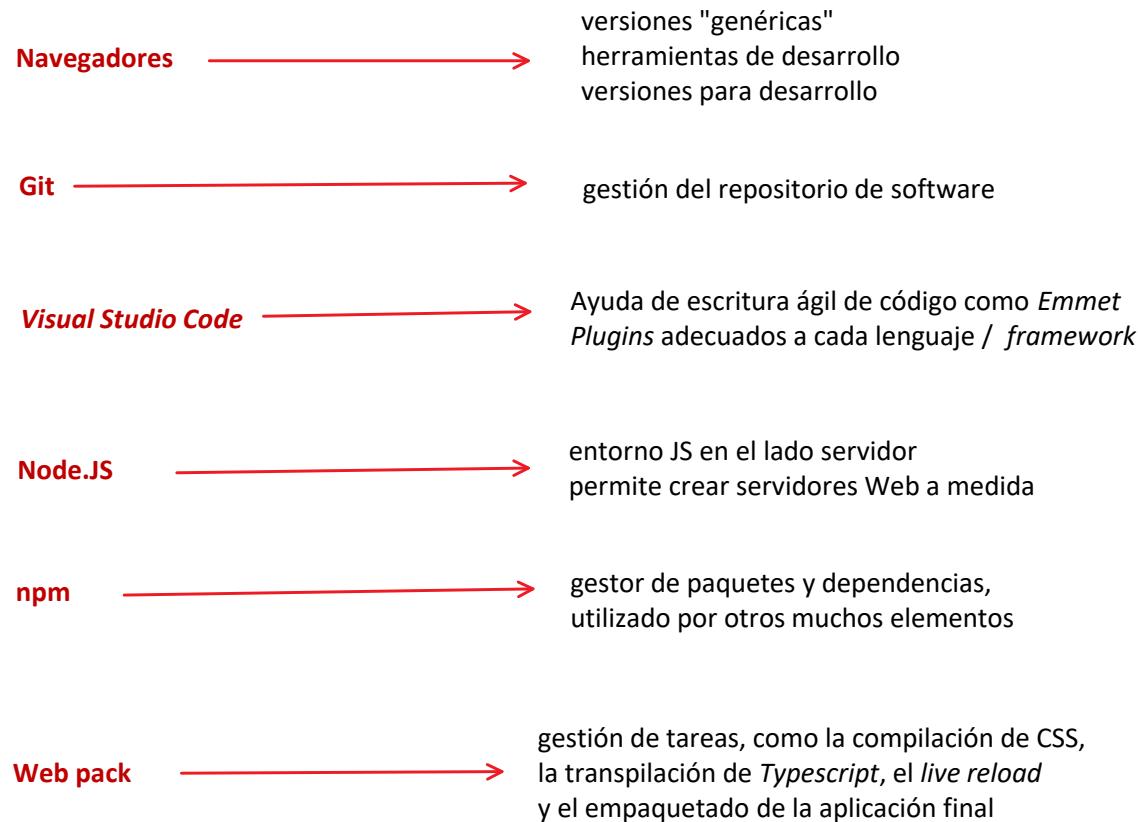
Árbol de componentes



Entorno de trabajo

sábado, 9 de septiembre de 2017 13:33

Navegadores.
Editores de código
Gestión de versiones. GIT
NodeJS y npm. Empaquetado



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



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



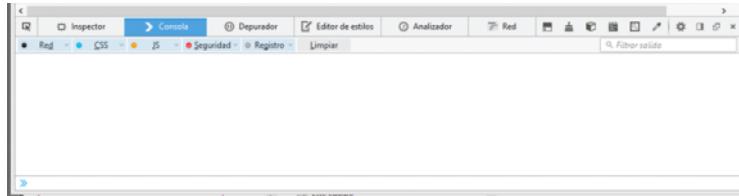
Herramienta de desarrollador en Edge, el navegador incorporado desde Windows 10



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 landing page. It features the title "Get on the bleeding edge of the web" and a subtext about being designed for developers and early adopters. A prominent yellow button labeled "Download Chrome Canary" is visible, along with download links for Windows 10/8.1/7 64-bit and links for Mac OS X and Android. Below the download section, there's an image showing three devices (laptop, tablet, and smartphone) all displaying the Chrome logo.

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

The screenshot shows the Mozilla Firefox Developer Edition landing page. It has a dark blue header with the Firefox logo and the text "Herramientas modernas para la Web Abierta". Below the header, there's a subtext about creating and refining web experiences with open-source tools. Three main sections are highlighted: "Modernizar" (Create interfaces de usuario rápidos, fáciles e interactivos con las herramientas integradas de React y Redux), "Personalizar" (Haz tuyas tus propias herramientas de desarrollo gracias al código abierto y la personalización), and "Optimizar" (Crea sitios adaptables y compatibles que funcionan para todos, en todos partes). A "Firefox Developer Edition" download button is at the bottom.

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

Plugin en Chrome

The screenshot shows the Augury extension page on the Chrome Web Store. The top navigation bar includes "OVERVIEW", "REVIEWS", "SUPPORT", "RELATED", and a "G+1" button. The "OVERVIEW" tab is selected, showing a component tree for an Angular 2 application. The tree shows components like "Component Tree", "Router Tree", "TodoList", "TodoInput", "TodoApp", and "TodoList". To the right of the tree, there's a sidebar with sections for "Properties", "Injector Graph", and "Additional Information". The "Properties" section shows "TodoList (View Source)" and "TodoService Object". The "Injector Graph" section shows "TodoInput (a-id = 0 13 0)", "TodoService", and "FormatService". The "Additional Information" section provides details about the extension: Version 1.2.5, Updated November 18, 2016, Size 858KB, and Language English. A "Report Abuse" button is also present.

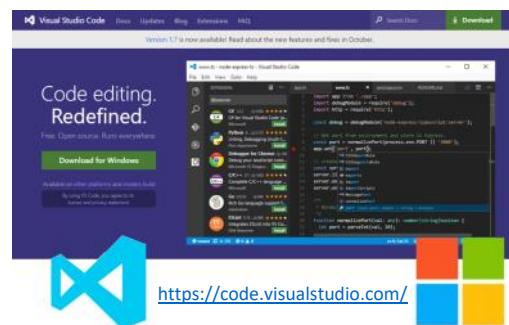
<https://chrome.google.com/webstore/detail/augury/elgalmkoelkbchkhacckoklkejnhcd>

Editores de código

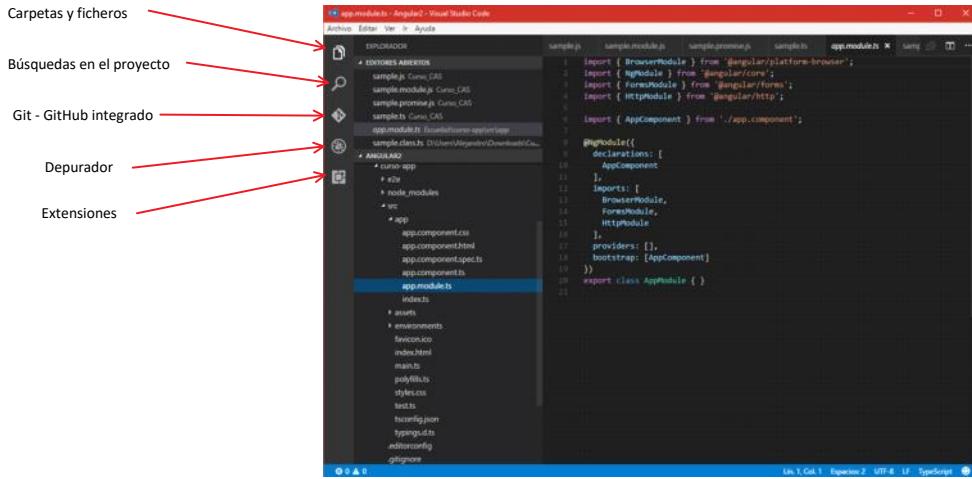
sábado, 9 de septiembre de 2017 18:21



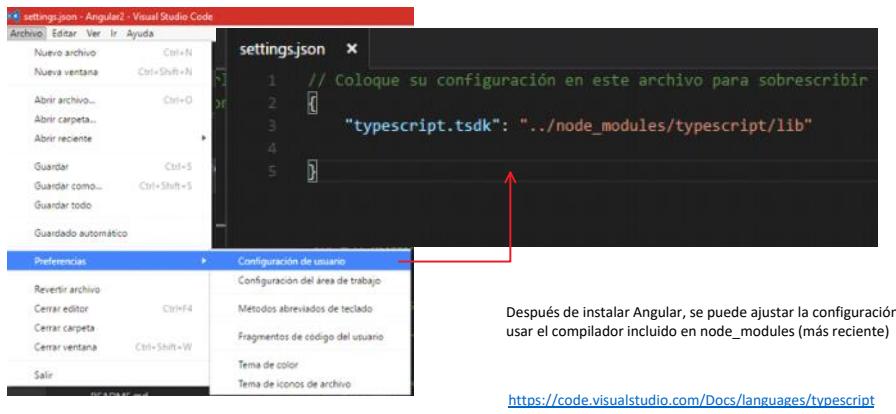
Visual Studio Code



Visual Studio Code



Configuración VSC

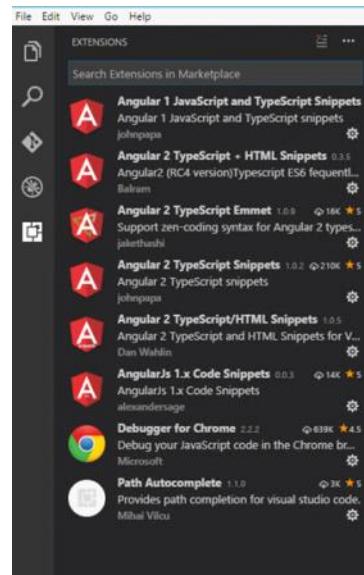


Extensiones de VSC

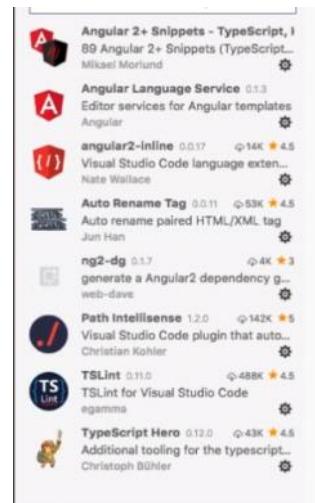
- Angular 2 Typescript
- Angular 4 Typescript Emmet
- Angular 4 and TypeScript/ HTML VS Code Snippets (Dan Wahlin)
- **Angular 4 TypeScript Snippets **** (John Papa)**
- Angular Lannguaje Service
- angular2-inline (Nate Wallace)

- **TSLint ***** (egamma)**
- Auto Import (steoates) / TypeScript Hero (Christoph Bühler)
- AutoRenameTag

- Debugger para Chrome ***** (Microsoft)
- npm *** (egamma)
- **Path Intellisense *** (Christian Kohler)**



A.Basalo, Angular 4



<https://marketplace.visualstudio.com/items?itemName=johnpapa.angular-essentials>

Pack con las extensiones de Angular, según recomendación de John Papa

- Angular v5 Snippets
- Angular Language Service
- Angular Inline
- Path Intellisense
- tslint
- Chrome Debugger

- Editor Config
- PrettierVS Code
- Winter is Coming theme



Git es un control de versiones distribuido diseñado 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

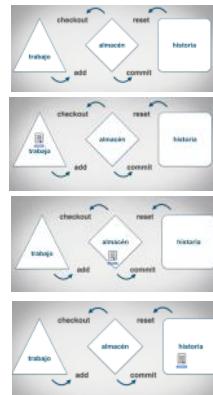
Comandos

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 -> conectarse a un repositorio remoto.

git branch -> crear una rama.
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 -> obtener los cambios de un repositorio remoto y fusionarlos en 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

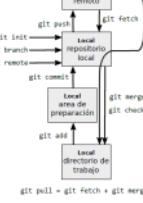


El repositorio creado tiene tres partes:
directorio de trabajo: el directorio en lo que se manejan los cambios.
almacén: el directorio donde se guarda la historia del repositorio.
historia: git que ha sido creado por el comando git init.

.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.

Este 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).



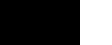
Resultado



Comprobación

```
git init
git add .
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/username/repo.git
git push -u origin main
```

Comprobación

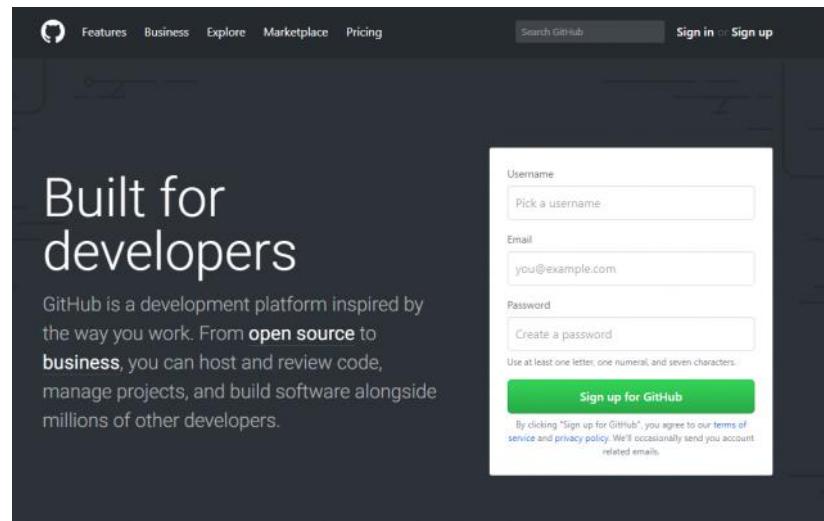


Resultados

Git en la Nube. GitHub

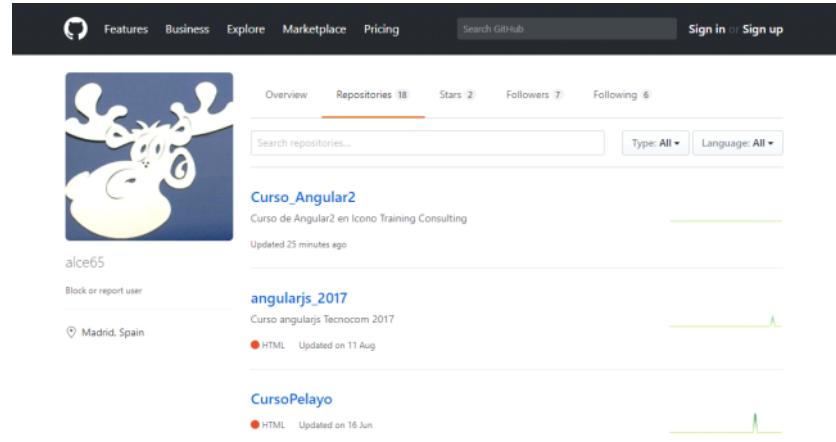
domingo, 10 de septiembre de 2017 12:28

- **GitHub** →
- GitLab
- Bitbucket



The screenshot shows the GitHub sign-up interface. At the top, there are navigation links for Features, Business, Explore, Marketplace, and Pricing. On the right, there are buttons for 'Search GitHub', 'Sign in', and 'Sign up'. The main area features a large 'Built for developers' heading. Below it, a paragraph describes GitHub as a development platform inspired by the way you work, from open source to business. It mentions hosting and reviewing code, managing projects, and building software alongside millions of other developers. To the right, there is a form for creating a new account, including fields for Username, Email, Password, and a 'Sign up for GitHub' button. A small note at the bottom of the form states: 'By clicking "Sign up for GitHub", you agree to our terms of service and privacy policy. We'll occasionally send you account-related emails.'

<https://github.com/alce65>

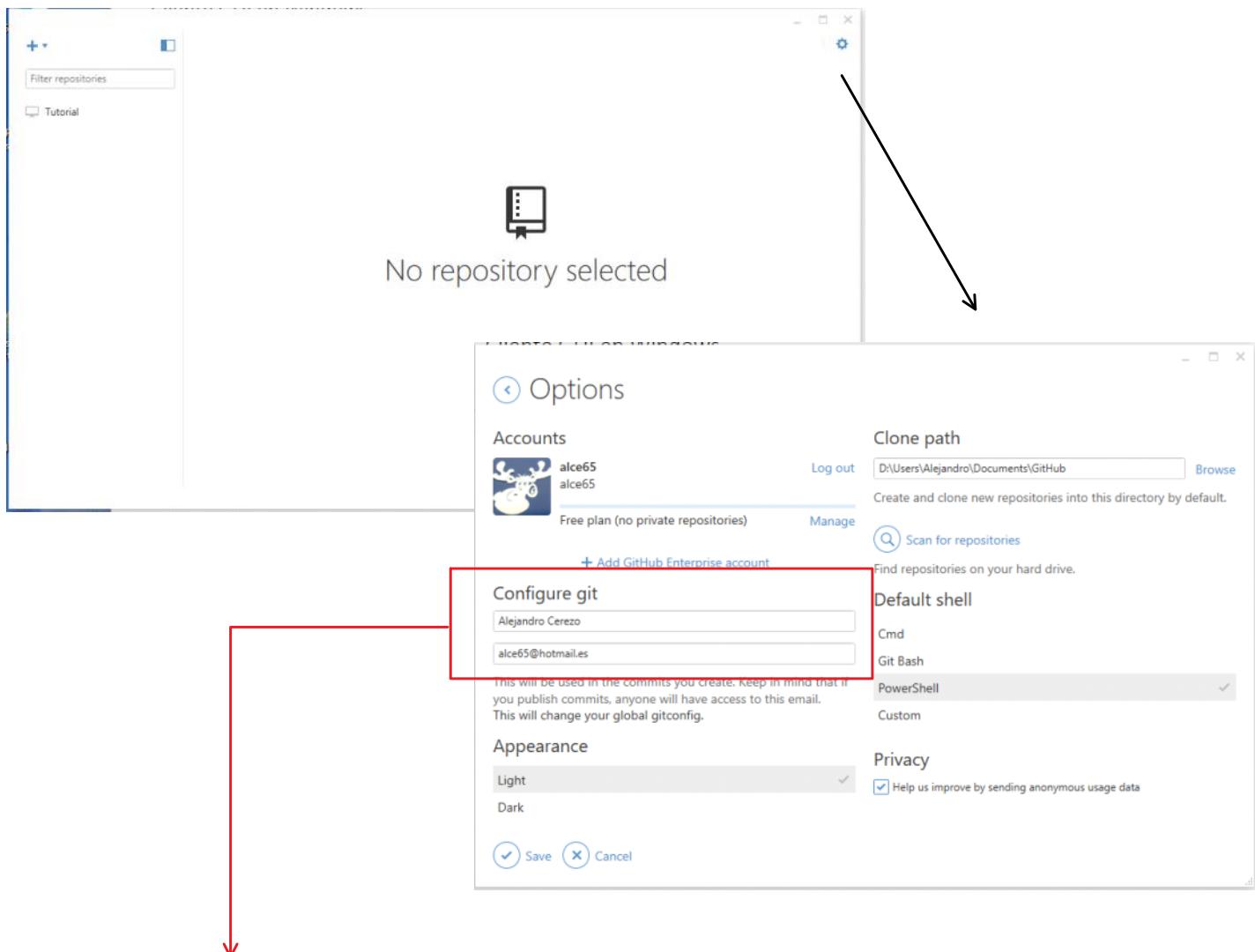
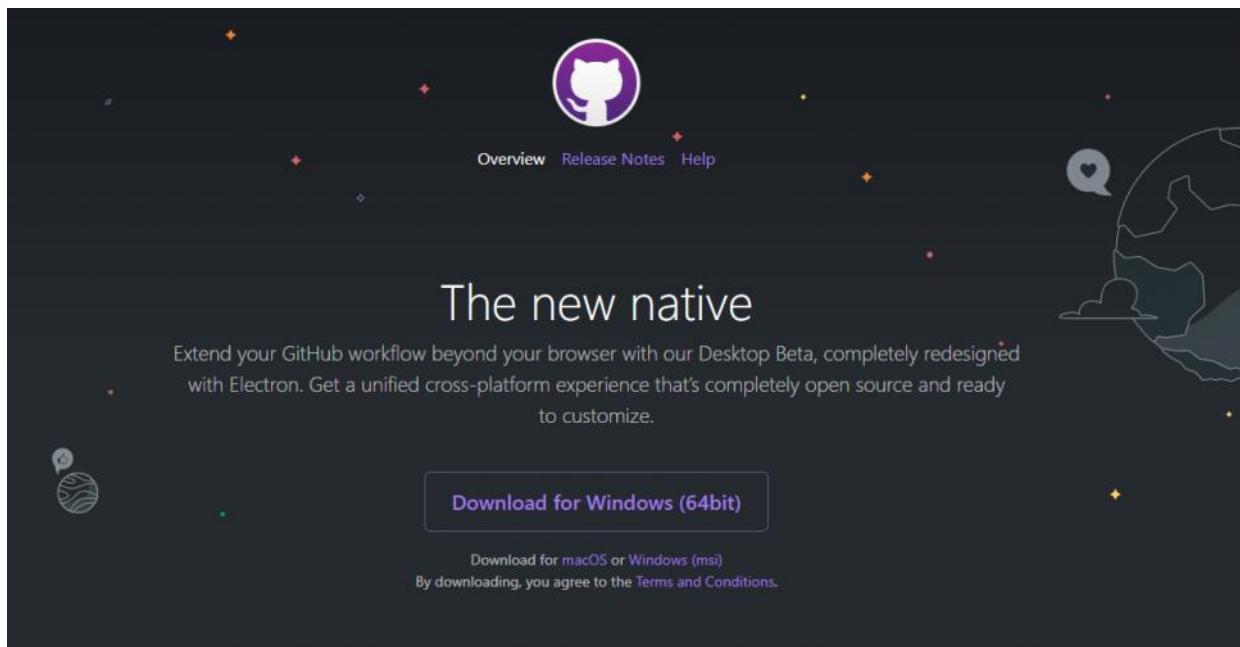


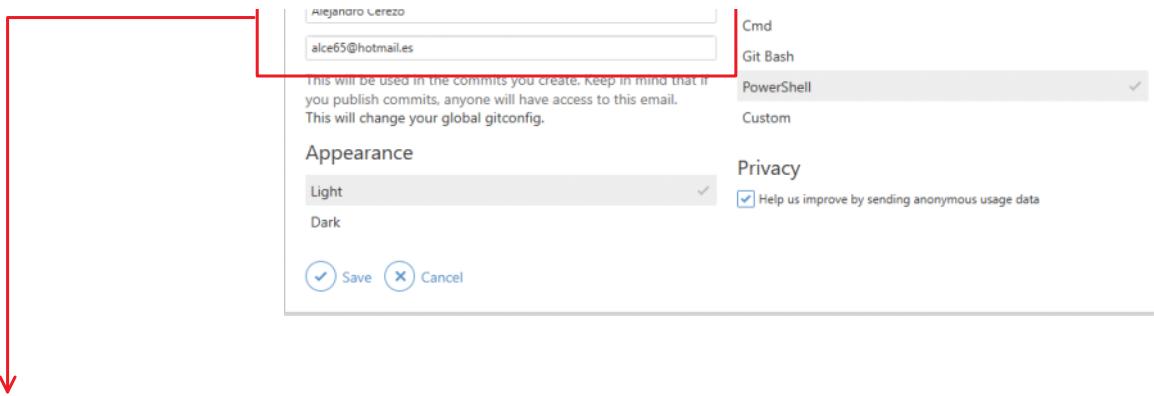
The screenshot shows the GitHub profile page for the user 'alce65'. At the top, there are navigation links for Features, Business, Explore, Marketplace, and Pricing. On the right, there are buttons for 'Search GitHub', 'Sign in', and 'Sign up'. The main area shows the user's profile picture (a reindeer), name 'alce65', and location 'Madrid, Spain'. Below the profile, there are three repository cards: 'Curso_Angular2' (18 stars, updated 23 minutes ago), 'angularjs_2017' (2 stars, updated on 11 Aug), and 'CursoPelayo' (0 stars, updated on 16 Jun). There are also tabs for Overview, Repositories, Stars, Followers, and Following.

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

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner / Repository name

Great repository names are short and memorable. Need inspiration? How about probable-spork.

Description (optional)
Curso de Angular2 en Icono Training Consulting

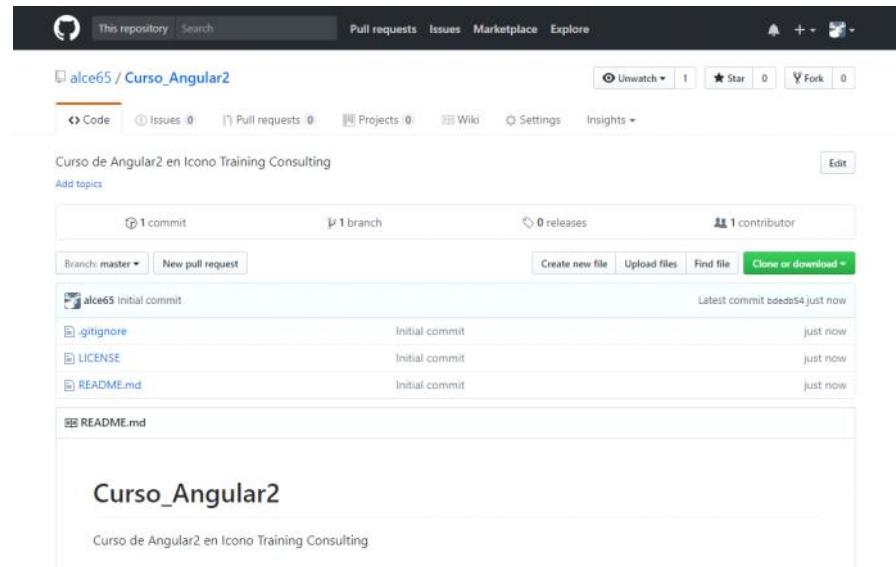
Public Anyone can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Initialize this repository with a README This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: Node | Add a license: MIT License | ⓘ

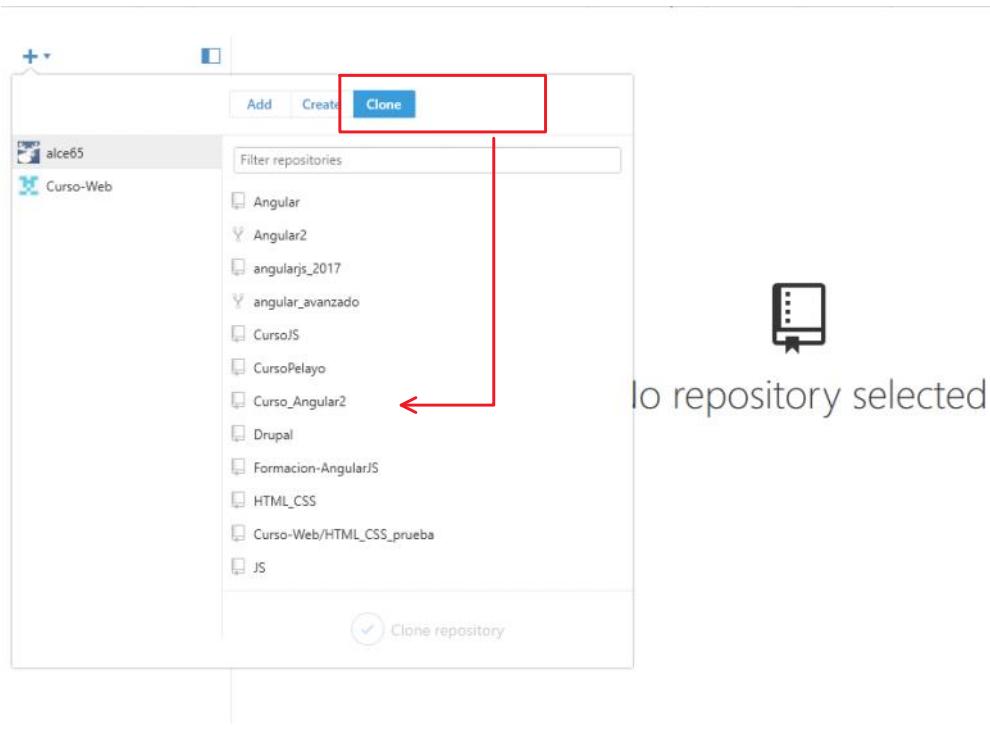
Create repository



Clonación local del repositorio

Clonar un repositorio

domingo, 10 de septiembre de 2017 17:22



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)

Versión LTS →

v6.11.3 LTS

Recomendado para la mayoría

Otras Descargas | Cambios | Documentación del API

v8.4.0 Actual

Últimas características

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

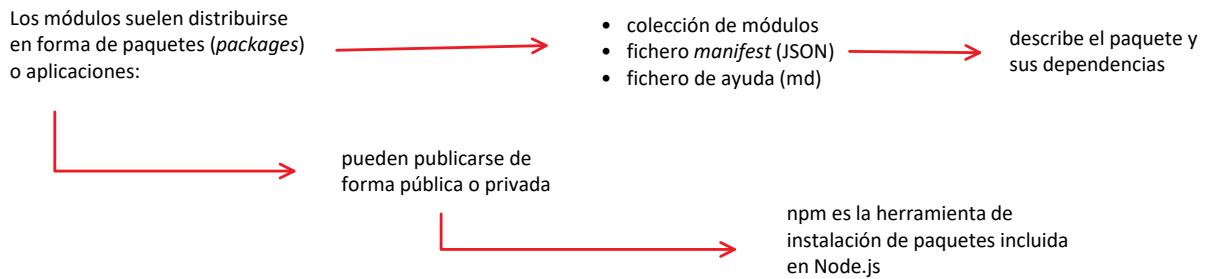
Objetivo del proyecto

→ 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



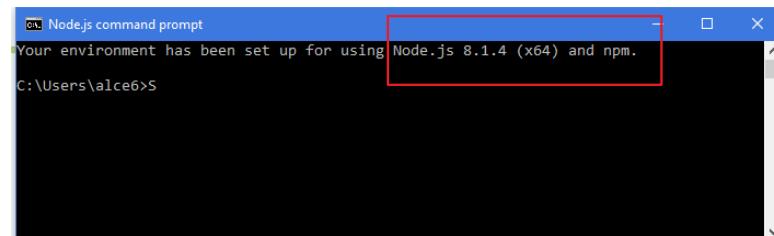
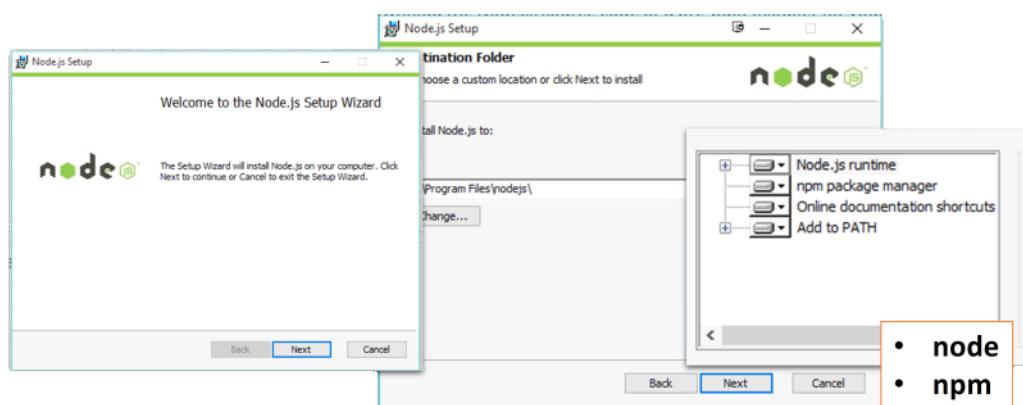
Instalación de Node.js & npm

sábado, 9 de septiembre de 2017 20:35

v8.4.0 Current

Latest Features

node-v8.4.0-x64.msi



- Accesible desde cualquier CLI (*cmd*, *PowerShell...*)
- Incluido en el *path*
- Se puede comprobar consultando la versión de *Node* y de *npm*

```
C:\ Símbolo del sistema
C:\Users\alce6>node -v
v8.1.4

C:\Users\alce6>npm -v
5.0.3
```

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

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

Salida ctrl-C o ctrl-D

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

Construcción de proyectos / empaquetado

herramientas para procesar los fuentes de la aplicación

- Reducción del tiempo de descarga
- Preprocesadores CSS
- Optimización del código, CSS, HTML
- Cumplimiento de estilos y
- Generación de JavaScript (transpilación)



<http://gruntjs.com/>



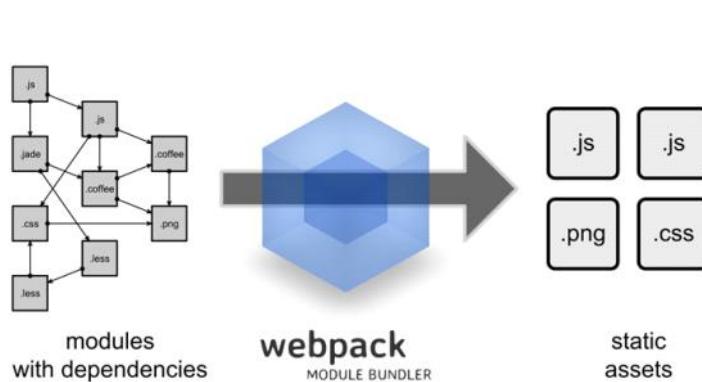
<http://gulpjs.com/>



<http://broccolijs.com/>



<https://webpack.github.io/>



finalmente incluido en [Angular-cli](#)

Configuración de proxies

lunes, 7 de agosto de 2017 21:38

Configuración git

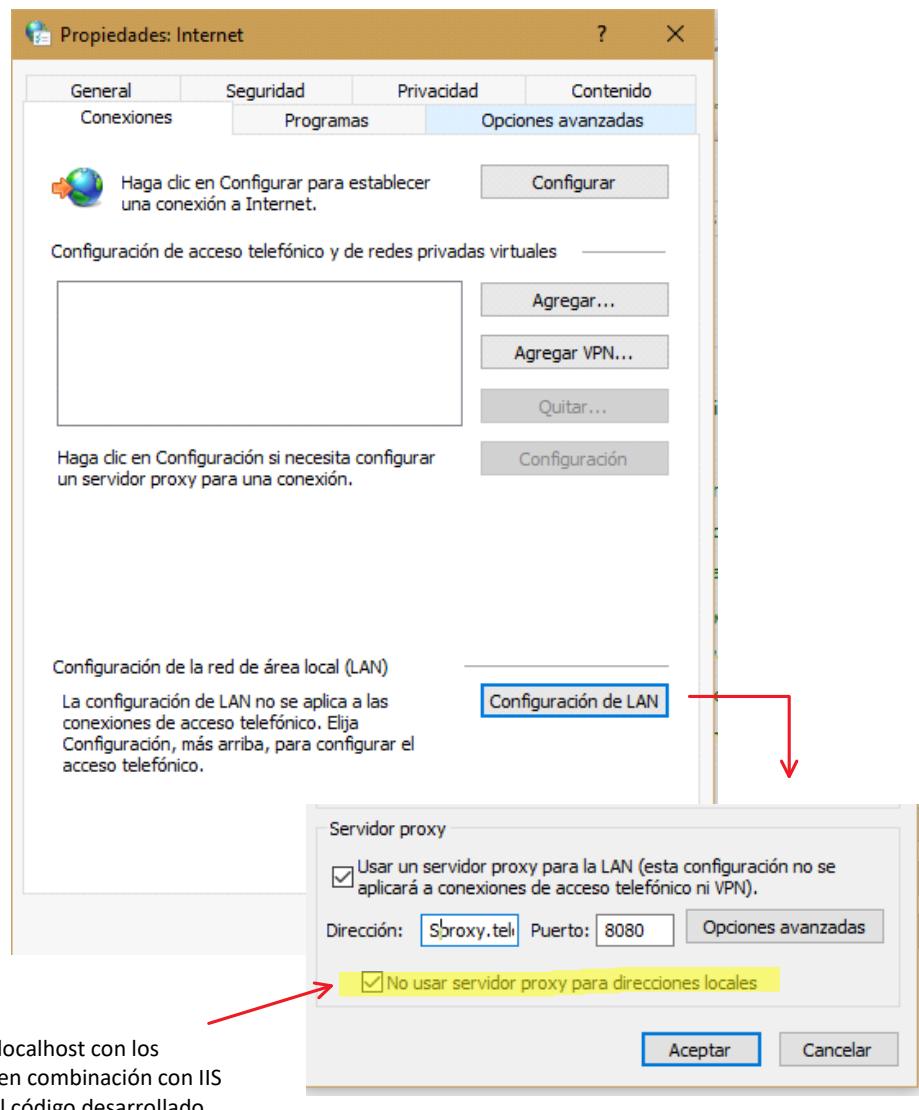
```
git config --global http.proxy http://user:passw@proxy.empresas.es:8080
```

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



Instalación e inicio

sábado, 9 de septiembre de 2017 13:33

Arquitectura del proyecto (*Scaffolding*).

Formas de creación de proyectos

No se suele crear un proyecto desde cero porque hay muchos ficheros y carpetas que son muy parecidos en todos los proyectos



Arquitectura o esqueleto (*scaffolding*)

Enfoques para conseguir el esqueleto inicial (*scaffolding*) de una web SPA (AngularJS / Angular)



- Generadores de plantillas ([Yeoman](#))
- Proyectos semilla ([seed](#)) disponibles en github
- En el caso de Angular
Herramienta oficial de gestión de proyectos : [angular-cli](#)

The screenshot shows the official Yeoman website. At the top center is the word "Yeoman" in a large, bold, teal font. To the right is a circular icon of the Yeoman mascot, a man with a large brown mustache, wearing a black top hat with a red band and a red scarf. Below the title is a brief description in Spanish: "Ecosistema de generadores de código y de proyectos de distintos lenguajes y plataforma, incluyendo uno específico para Angular.JS". Underneath this is a blue button with the URL "<http://yeoman.io/>". The main content area features a large illustration of the Yeoman mascot on the left, waving his hand. To the right of the illustration, the text "THE WEB'S SCAFFOLDING TOOL FOR MODERN WEBAPPS" is displayed in white capital letters against a teal background. Below this, there's a cartoon illustration of three people working on a large white rocket or satellite. A woman in a lab coat stands on the left, a man in a suit stands on the right, and another man in a top hat sits on a ladder. In the bottom left corner of the main content area, there's a small text box containing installation instructions: "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

Ejecutamos

npm -g install yo grunt-cli bower

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

Generadores de código Angular 2 no oficiales basados en Yeoman

- <https://www.npmjs.com/package/generator-modern-web-dev>
- <https://www.npmjs.com/package/generator-angular2>
- <https://www.npmjs.com/package/generator-gulp-angular2>
- <https://github.com/joshuacaron/generator-angular2-gulp-webpack>
- <https://www.npmjs.com/package/slush-angular2>



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

```
Welcome to Yeoman,
ladies and gentlemen!
-----(o)-----
( - 'U' - )
\ / A \
| ~ |
-----y-----

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



Yeoman: instalación (3)

```
Done, without errors.

Execution Time (2015-11-28 19:33:40 UTC)
loading tasks      534ms [██████████] 50%
loading grunt-wiredep 14ms [██] 15%
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

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 web apps or sites.

Karma
Spectacular Test Runner for JavaScript.

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 the framework most suited to your application development.

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>

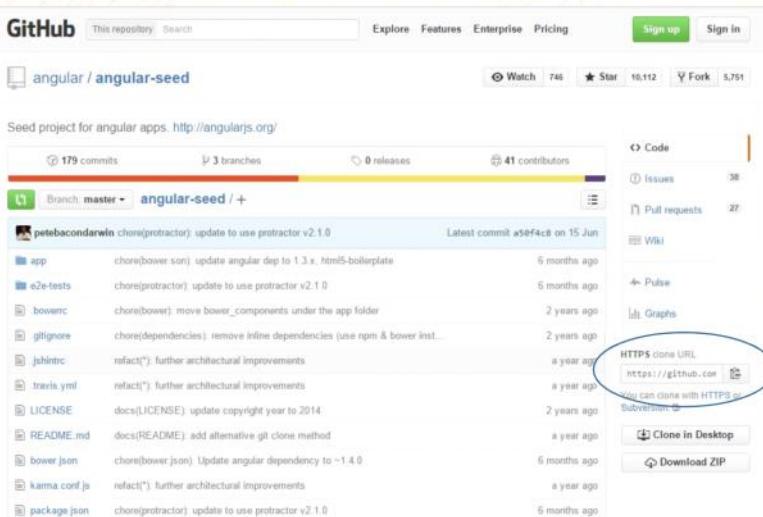
Crearía la correspondiente vista nueva en la carpeta adecuada

Angular seed 1.x



en Angular 1.x esqueleto de una aplicación desarrollada por el propio equipo de AngularJS, con el respaldo de Google, como punto de partida para proyectos

<https://github.com/angular/angular-seed>



Angular seed 2



Proyectos semilla (seed) disponibles en github

- <http://mgechev.github.io/angular2-seed/>
- <https://github.com/ghpabs/angular2-seed-project>
- <https://github.com/cureon/angular2-sass-gulp-boilerplate>
- <https://angularclass.github.io/angular2-webpack-starter/>
- <https://github.com/LuxDie/angular2-seed-jade>
- <https://github.com/justindujardin/angular2-seed>

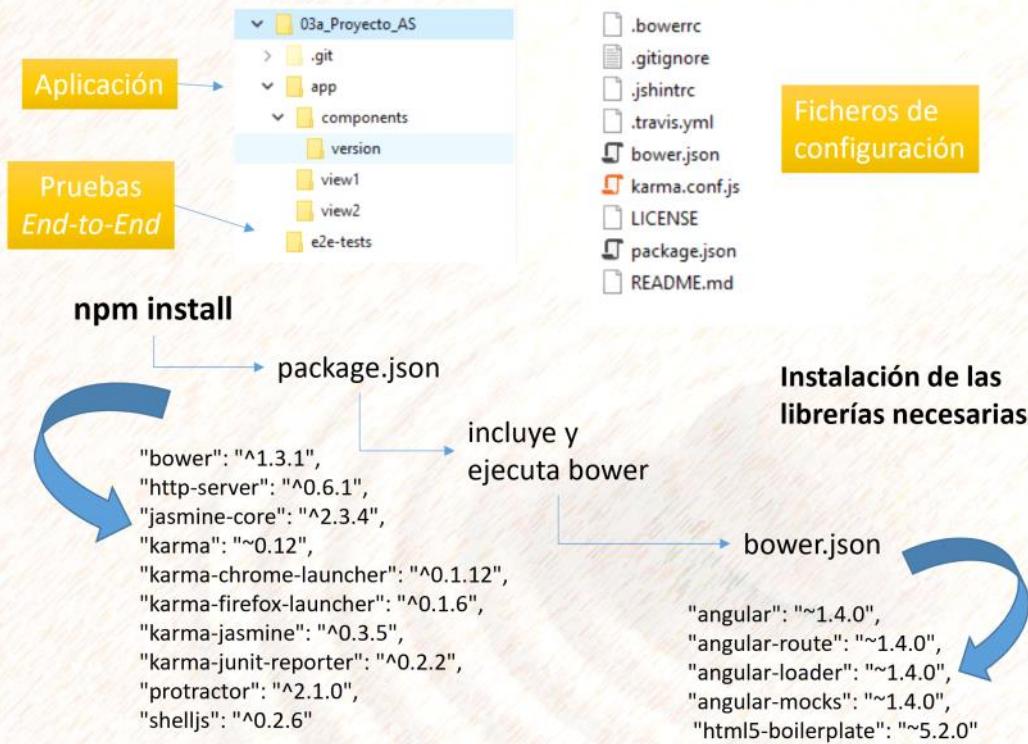
git clone origen destino

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

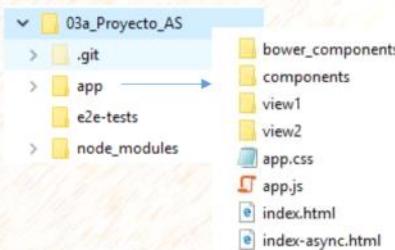
```
D:\Users\Alejandro\Mi nube\OneDrive\Desarrollo\Angular>git clone https://github.com/angular/angular-seed.git 03a_Proyecto_AS
Cloning into '03a_Proyecto_AS'...
remote: Counting objects: 2590, done.
Receiving objects: 100% (2590/2590), 12.21 MiB | 4.16 MiB/s, done.
Resolving deltas:  1% (14/1370) 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

No incluye aún ninguna librería

Angular seed: Instalación



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.

Angular-cli

sábado, 9 de septiembre de 2017 18:51

Angular command line interface
Herramienta oficial de gestión de proyectos

<https://cli.angular.io>

Ofrece comandos para todo el **ciclo de desarrollo**:

- Generación (*bootstrapping*) del proyecto inicial
- Herramientas de generación de código
- Modo desarrollo con
 - compilado automático de *TypeScript*
 - arranque de un servidor Web
 - y actualización del navegador (*Live Reloading.*)
- *Testing*
- Construcción del proyecto para distribución (*build*)



Herramientas de terceros incluidas en Angular-cli



webpack
MODULE BUNDLER

<https://webpack.github.io/>

Construcción
del proyecto

 **KARMA**
<https://karma-runner.github.io>

 **Jasmine**
<http://jasmine.github.io/>

 **Protractor**
end to end testing for AngularJS
<http://www.protractortest.org/>

Herramientas de testing

Instalación y uso

sábado, 14 de octubre de 2017 20:11

Instalación

Desde una ventana de terminal "como administrador"

```
npm install -g @angular/cli
```

instala globalmente la herramienta angular cli

```
npm update -g @angular/cli
```

actualiza la instalación global de angular cli

Destino de la instalación

```
"<user>\AppData\Roaming\npm\node_modules"
```

Resultado de la instalación

The screenshot shows a Windows command prompt window titled 'C:\WINDOWS\system32\cmd.exe'. The command 'D:\>ng version' is entered, and the output shows the following information:

```
D:\>ng version
angular-cli: 1.0.0-beta.19-3
node: 8.4.0
os: win32 x64
@angular/cli: 1.4.1
node: 8.4.0
os: win32 x64
C:\Users\alce6>
```

Generación de un proyecto

Desde la carpeta donde queremos que resida nuestro proyecto

```
ng new my-app
```

crea una carpeta en la que descarga hasta 250MB. configura y organiza el conjunto de herramientas de una forma prefijada

```
cd my-app
```

desde el directorio del proyecto, recién creado se levanta un servidor Node que mostrará la aplicación en localhost:4200

```
ng serve
```

- transpilará Typescript
- recargara automáticamente al guardar un fichero fuente

En nuestro caso, el comando de creación del proyecto será:

```
ng new curso-app -p acc --routing true
```

<nombre
proyecto>

<prefijo>

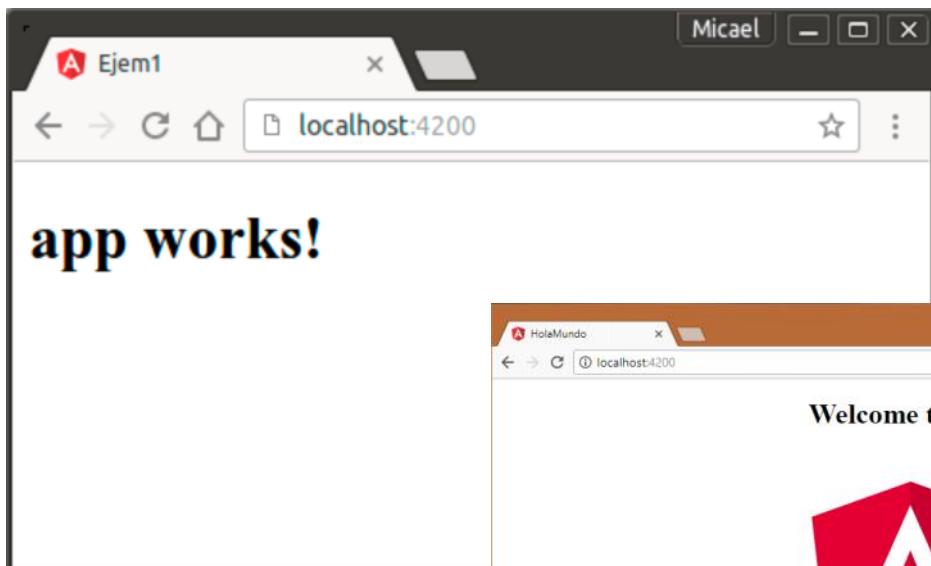
Ejecución: ng serve

domingo, 4 de febrero de 2018 21:21

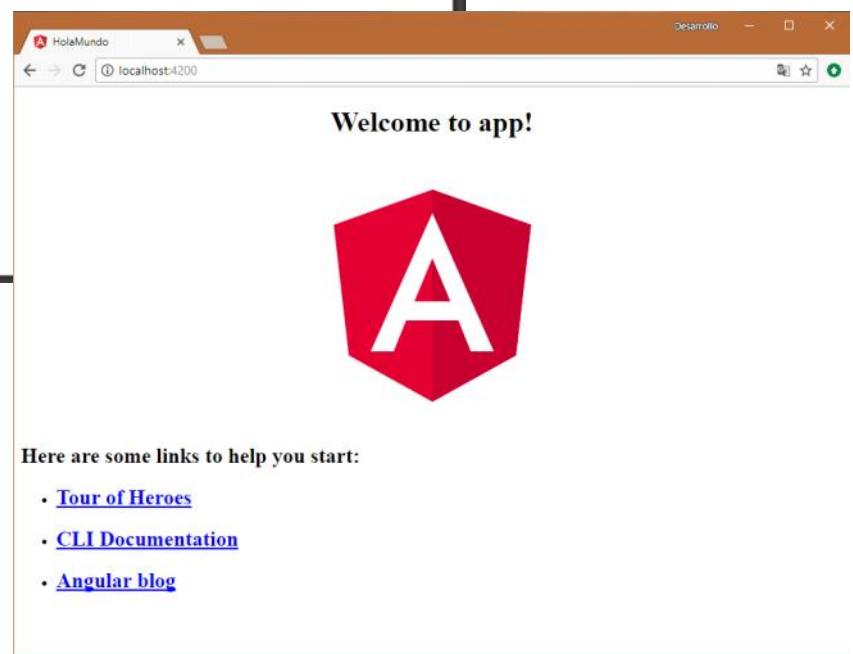
```
D:\Desarrollo\Angular2\Curso_CAS\curso-app>ng serve
** NG Live Development Server is running on http://localhost:4200. **
43871ms building modules
47ms sealing
0ms optimizing
0ms basic module optimization
180ms module optimization
4ms advanced module optimization
18ms basic chunk optimization
15ms chunk optimization
0ms advanced chunk optimization
16ms module and chunk tree optimization
266ms module reviving
15ms module order optimization
16ms module id optimization
16ms chunk reviving
0ms chunk order optimization
31ms chunk id optimization
109ms hashing
16ms module assets processing
250ms chunk assets processing
15ms additional chunk assets processing
0ms recording
0ms additional asset processing
4069ms chunk asset optimization
2923ms asset optimization
78ms emitting
Hash: 541eb735658c9449ad60
Version: webpack 2.1.0-beta.25
Time: 52002ms
    Asset      Size  Chunks   Chunk Names
main.bundle.js  2.71 MB  0, 2  [emitted]  main
styles.bundle.js 10.2 kB  1, 2  [emitted]  styles
    inline.js  5.53 kB  2  [emitted]  inline
    main.map  2.82 MB  0, 2  [emitted]  main
    styles.map 14.2 kB  1, 2  [emitted]  styles
    inline.map  5.59 kB  2  [emitted]  inline
index.html 475 bytes          [emitted]
Child html-webpack-plugin for "index.html":
    Asset      Size  Chunks   Chunk Names
index.html  2.81 kB  0
webpack: bundle is now VALID.
```

Ventana de la consola en la que webpack

- levanta un servidor Web basado en *Node* en la máquina local y el puerto 4200
- con la aplicación empaquetada de forma temporal en modo adecuado para el desarrollo,
- capaz de actualizarse automáticamente ante los cambios en los ficheros fuente



Primera aplicación en Angular2 generada por angular-cli en sus versiones definitivas



En Angular 5, las mejoras del proceso de compilación aot lo hacen utilizable incluso en

fase de desarrollo por lo que se recomienda añadirlo al comando serve

```
ng serve --aot -o
```

Lo habitual es guardar el comando con sus modificadores como script npm



```
"scripts": {  
  "ng": "ng",  
  "start": "ng serve --aot -o",  
  "build": "ng build --prod",  
  "test": "ng test",  
  "lint": "ng lint",  
  "e2e": "ng e2e"  
},
```

Al ser un comando estándar npm se ejecuta directamente (sin anteponer run)

```
npm start
```

Hola Mundo

sábado, 9 de septiembre de 2017 23:57

Instalación de VSC y sus extensiones

Instalación de *git*
configuración del proxy

```
$>git config --global http.proxy http://user:passw@proxy.empresa.es:8080
```

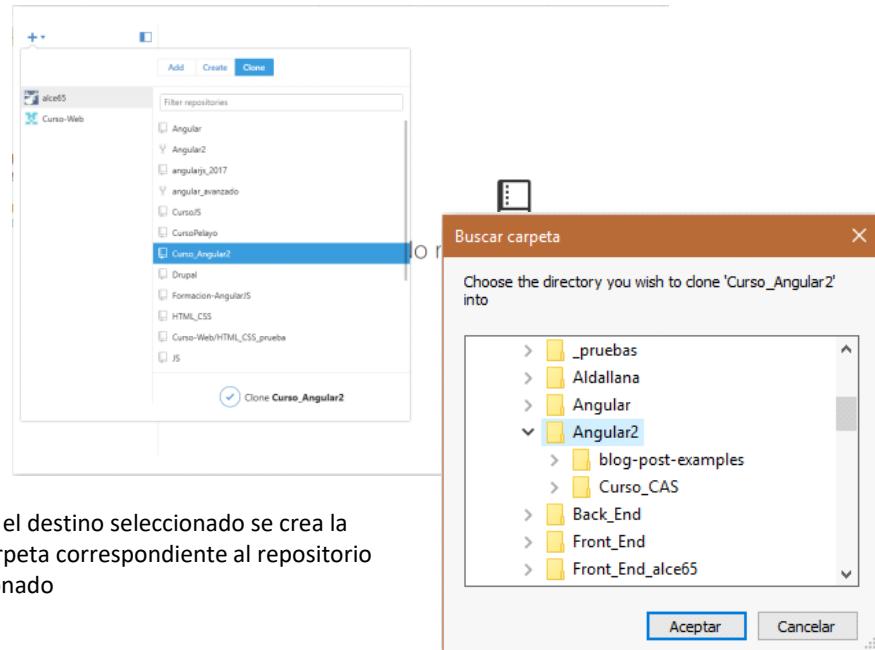
Instalación de *node/npm*
configuración del proxy

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

Instalación de Angular-cli:

```
$>npm install -g @angular/cli
```

Creación de la carpeta para el curso,
como repositorio *git* vinculado a GitHub



En el destino seleccionado se crea la
carpeta correspondiente al repositorio
clonado

Creación del proyecto: ng new hola-mundo

```
C:\ Administrador: C:\Windows\system32\cmd.exe
D:\Desarrollo\Angular2\Curso_Angular2> ng new hola-mundo

C:\ Administrador: C:\Windows\system32\cmd.exe
D:\Desarrollo\Angular2\Curso_Angular2> ng new hola-mundo
Unable to find "@angular/cli" in devDependencies.

Please take the following steps to avoid issues:
"npm install --save-dev @angular/cli@latest"

  create  hola-mundo/e2e/app.e2e-spec.ts (292 bytes)
  create  hola-mundo/e2e/app.po.ts (208 bytes)
  create  hola-mundo/e2e/tsconfig.e2e.json (235 bytes)
  create  hola-mundo/karma.conf.js (923 bytes)
  create  hola-mundo/package.json (1315 bytes)
  create  hola-mundo/protractor.conf.js (722 bytes)
  create  hola-mundo/README.md (1100 bytes)
  create  hola-mundo/tsconfig.json (363 bytes)
  create  hola-mundo/tslint.json (3040 bytes)
  create  hola-mundo/.angular-cli.json (1128 bytes)
  create  hola-mundo/.editorconfig (245 bytes)
  create  hola-mundo/.gitignore (516 bytes)
  create  hola-mundo/src/assets/.gitkeep (0 bytes)
  create  hola-mundo/src/environments/environment.prod.ts (51 bytes)
  create  hola-mundo/src/environments/environment.ts (387 bytes)
  create  hola-mundo/src/favicon.ico (5430 bytes)
  create  hola-mundo/src/index.html (296 bytes)
  create  hola-mundo/src/main.ts (370 bytes)
  create  hola-mundo/src/polyfills.ts (7489 bytes)
```

```
Administrator: C:\Windows\system32\cmd.exe
D:\Desarrollo\Angular2\Curso_Angular2> ng new hola-mundo
Unable to find "@angular/cli" in devDependencies.

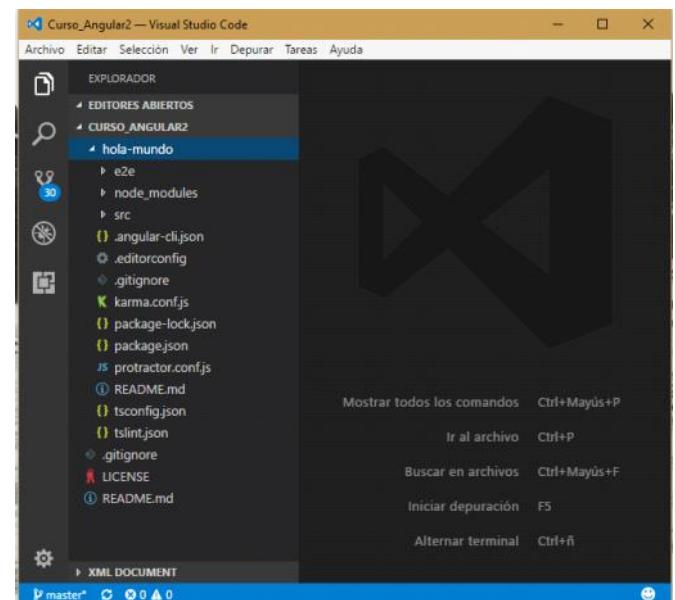
Please take the following steps to avoid issues:
"npm install --save-dev @angular/cli@latest"

  create  hola-mundo/e2e/app.e2e-spec.ts (292 bytes)
  create  hola-mundo/e2e/app.po.ts (208 bytes)
  create  hola-mundo/e2e/tsconfig.e2e.json (235 bytes)
  create  hola-mundo/karma.conf.js (923 bytes)
  create  hola-mundo/package.json (1315 bytes)
  create  hola-mundo/protractor.conf.js (722 bytes)
  create  hola-mundo/README.md (1100 bytes)
  create  hola-mundo/tsconfig.json (363 bytes)
  create  hola-mundo/tslint.json (3040 bytes)
  create  hola-mundo/.angular-cli.json (1128 bytes)
  create  hola-mundo/.editorconfig (245 bytes)
  create  hola-mundo/.gitignore (516 bytes)
  create  hola-mundo/src/assets/.gitkeep (0 bytes)
  create  hola-mundo/src/environments/environment.prod.ts (51 bytes)
  create  hola-mundo/src/environments/environment.ts (387 bytes)
  create  hola-mundo/src/favicon.ico (5430 bytes)
  create  hola-mundo/src/index.html (296 bytes)
  create  hola-mundo/src/main.ts (370 bytes)
  create  hola-mundo/src/polyfills.ts (2480 bytes)
  create  hola-mundo/src/styles.css (80 bytes)
  create  hola-mundo/src/test.ts (1085 bytes)
  create  hola-mundo/src/tsconfig.app.json (211 bytes)
  create  hola-mundo/src/tsconfig.spec.json (304 bytes)
  create  hola-mundo/src/typings.d.ts (104 bytes)
  create  hola-mundo/src/app/app.module.ts (314 bytes)
  create  hola-mundo/src/app/app.component.html (1075 bytes)
  create  hola-mundo/src/app/app.component.spec.ts (986 bytes)
  create  hola-mundo/src/app/app.components.ts (207 bytes)
  create  hola-mundo/src/app/app.component.css (0 bytes)

Installing packages for tooling via npm.
Installed packages for tooling via npm.
Directory is already under version control. Skipping initialization of git.
Project 'hola-mundo' successfully created.

D:\Desarrollo\Angular2\Curso_Angular2>
```

Accedemos al proyecto desde VSC



Resultado

domingo, 10 de septiembre de 2017 18:51

The screenshot illustrates the workflow for developing an Angular 2 application:

- Visual Studio Code:** The top window shows the code editor with the file `app-component.html`. A red arrow points from the code editor down to the terminal.
- Terminal:** The middle window shows the command line interface running on a Windows PowerShell. It displays the command `ng serve` being typed. Another red arrow points from the terminal down to the browser.
- Browser:** The bottom window shows a web browser window titled "HolaMundo" at the URL `localhost:4200`. The page content includes the Angular logo and the text "Welcome to app!". A third red arrow points from the browser back up to the code editor.

Code Editor (Visual Studio Code):

```
only a placeholder and can be replaced with text
<h2>
  <a href="https://angular.io/tutorial">Tour of Heroes</a>
</h2>
<h2>
  <a href="https://github.com/angular/angular-cli/wiki">CLI Dev Guide</a>
</h2>
<h2>
  <a href="https://blog.angular.io/">Angular blog</a>
</h2>
```

Terminal:

```
PS D:\Desarrollo\Angular2\Curso_Angular2> cd ./hola-mundo
PS D:\Desarrollo\Angular2\Curso_Angular2\hola-mundo> ng serve
```

Browser:

Welcome to app!

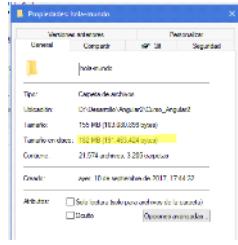


Here are some links to help you start:

- [Tour of Heroes](#)
- [CLI Documentation](#)
- [Angular blog](#)

```
** NG Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ ***
Date: 2017-09-10T16:55:54.342Z
Hash: d48ae1a31f62b6eb16bc
Time: 8899ms
chunk {inline} inline.bundle.js, inline.bundle.js.map (inline) 5.83 kB [entry] [rendered]
chunk {main} main.bundle.js, main.bundle.js.map (main) 8.64 kB [vendor] [initial] [rendered]
chunk {polyfills} polyfills.bundle.js, polyfills.bundle.js.map (polyfills) 209 kB [inline] [initial] [rendered]
chunk {styles} styles.bundle.js, styles.bundle.js.map (styles) 11.3 kB [inline] [initial] [rendered]
chunk {vendor} vendor.bundle.js, vendor.bundle.js.map (vendor) 2.28 MB [initial] [rendered]
webpack: Compiled successfully.
```

Resultado: estructura



ficheros de infraestructura

- Infraestructura para TypeScript
- Infraestructura de pruebas
- Infraestructura de despliegue

- .editconfig
- .gitignore: git

- package.json: librerías (dependencias)
- package_lock.json
- angular-cli.json: angular-cli
- angular-cli-build.js: angular-cli
- tsconfig.json: typescript
- tslint.json: tipos
- karma.conf.js
- protractor.conf.js

Ficheros de configuración del editor y de git

package.json

```
{
  "name": "curso-app",
  "version": "0.0.0",
  "license": "MIT",
  "angular-cli": {},
  "scripts": {
    "start": "ng serve",
    "lint": "tslint \"src/**/*.ts\"",
    "test": "ng test",
    "pree2e": "webdriver-manager update",
    "e2e": "protractor"
  },
  "private": true,
  "dependencies": {
    "@angular/common": "~2.1.0",
    "@angular/compiler": "~2.1.0",
    "@angular/core": "~2.1.0",
    "@angular/forms": "~2.1.0",
    "@angular/http": "~2.1.0",
    "@angular/platform-browser": "~2.1.0",
    "@angular/platform-browser-dynamic": "~2.1.0",
    "@angular/router": "~2.1.0",
    "core-js": "~2.4.3",
    "rxjs": "5.0.0-beta.12",
    "ts-helpers": "~1.1.1",
    "zone.js": "~0.6.23"
  },
  "devDependencies": {
    "@types/jasmine": "~2.2.30",
    "@types/node": "~6.0.42",
    "angular-cli": "1.0.0-beta.19-3",
    "codeyever": "1.0.0-beta.1",
    "jasmine-core": "2.4.1",
    "jasmine-spec-reporter": "2.5.0",
    "karma": "1.2.0",
    "karma-chrome-launcher": "~2.0.0",
    "karma-dll": "~1.0.1",
    "karma-jasmine": "~1.0.2",
    "karma-remap-istanbul": "~0.2.1",
    "protractor": "4.0.9",
    "ts-node": "1.2.1",
    "tslint": "3.13.0",
    "typescript": "~2.0.3",
    "webdriver-manager": "10.2.5"
  }
}
```

Scripts

Dependencias

Dependencias de desarrollo

- **e2e:** Testing end to end
- **dist:** Recursos que hay que publicar en el servidor web
- **node_modules:** Librerías y herramientas descargadas
- **src:** Fuentes de la aplicación

Angular 2.0

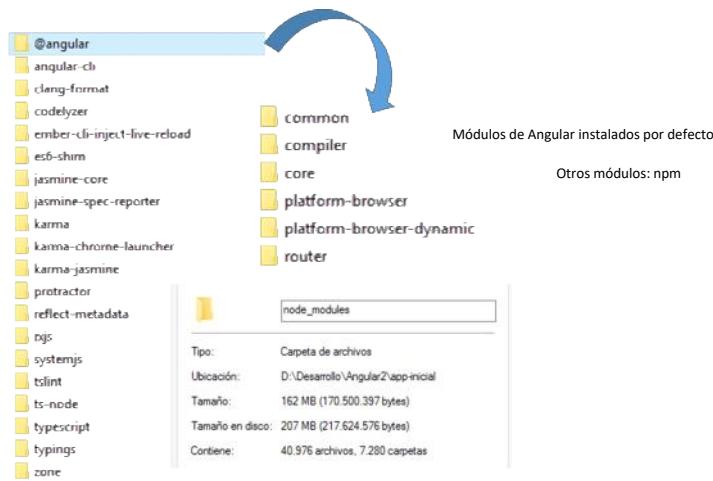
- **config:** configuración de environments y tests
- **E2E:** Testing end to end
- **dist:** Recursos que hay que publicar en el servidor web
- **node_modules:** Librerías y herramientas descargadas
- **public:**
- **src:** Fuentes de la aplicación
- **tmp:** temporal
- **typings:** tipos predefinidos
- **.clang-format**
- **.editorconfig**
- **.gitignore**
- **angular-cli.json**
- **angular-cli-build.js**
- **package.json**
- **tslint.json**
- **typings.json**

- karma.conf.js: tests con karma
- protractor.conf.js: tests con protractor

- config
- dist
- e2e
- node_modules
- public
- src
- tmp
- typings
- .clang-format
- .editorconfig
- .gitignore
- angular-cli.json
- angular-cli-build.js
- package.json
- tslint.json
- typings.json

- karma.conf.js: tests con karma
- protractor.conf.js: tests con protractor
- package.json: librerías (dependencias)
- angular-cli.json: angular-cli
- angular-cli-build.js: angular-cli
- tsconfig.json: typescript
- tslint.json: tipos

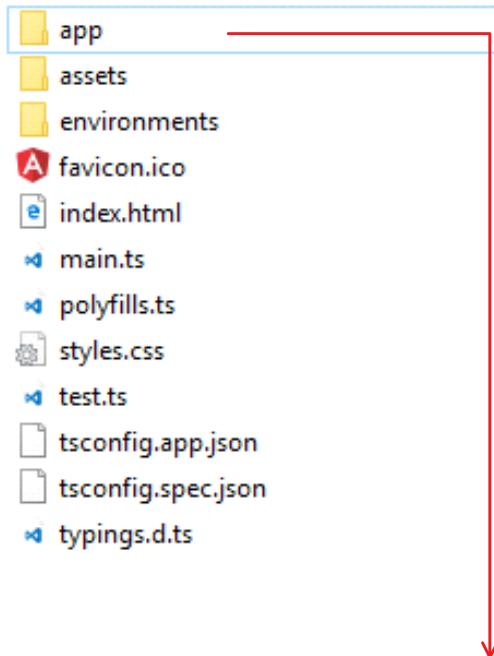
Resultado: node_modules



- No se incluye en la copia de los proyectos distribuida en repositorios *git* / *Github*
- Puede volver a generarse en cualquier momento, de acuerdo con lo indicado en *package.json*: *npm install*
- Desde el punto de vista de *npm/Node*, la carpeta *node_modules* puede reubicarse en niveles superiores para compartirla en diversos proyectos.

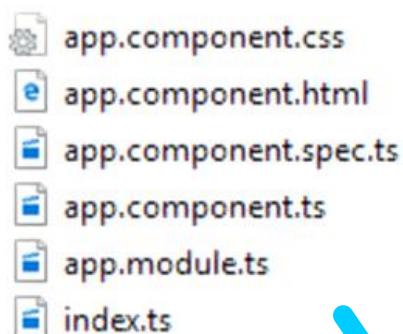
-
- *npm* busca la carpeta *node_modules* en la carpeta actual o en los niveles superiores
 - Puede reubicarse en niveles superiores para compartirlo en diversos proyectos
 - Puede ser necesario ajustar la configuración del editor de código, para indicarle donde se ubica el compilador de *typescript*

Sin embargo angular/cli necesita que se mantenga en su ubicación original



- **index.html:** Página principal.
Se editará para incluir CSS globales en la web
- favicon.ico: Icono de la aplicación
- **main.ts:** Fichero principal de la aplicación.
No es necesario modificarlo
- **tsconfig.app.json**
- **tsconfig.spec.json:** Configuración del compilador TS

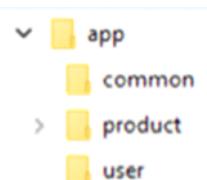
- style.css
- polyfills.ts
- test.ts
- typings.d.ts



carpeta que contiene los ficheros fuente principales de la aplicación.

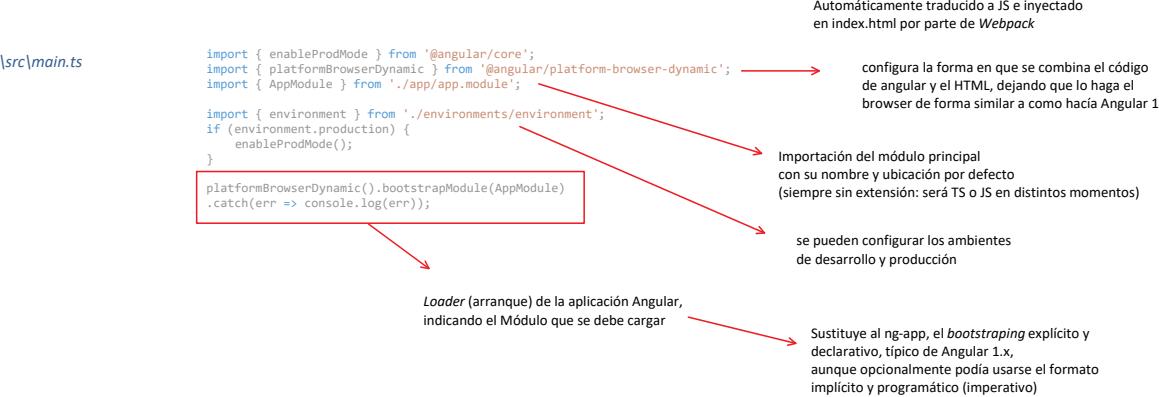
Distribución por características

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

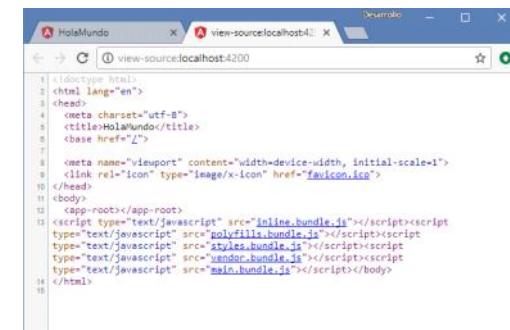
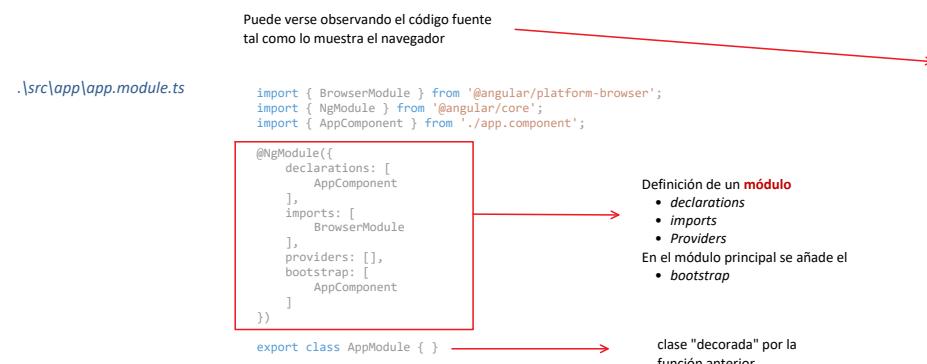


Ficheros principales

domingo, 10 de septiembre de 2017 18:24



Se echan en falta llamadas a ficheros externos (JS, CSS)
La función de Webpack es "empaquetar" toda esa información en bundles e injectar las correspondientes llamadas a ellos



.\src\app\app.component.html

```
<!-The content below is only a placeholder and can be replaced.-->
<div style="text-align:center">
  <h1>
    Welcome to {{title}}!
  </h1>
    
    </div>  
</article>  
<footer>  
    <p>{{formador}} - {{fecha}}</p>  
    <p>{{empresa}}</p>  
</footer>
```

Interpolamos las variables del componente

Incorporamos el fichero



Reubicación de node_modules

sábado, 11 de noviembre de 2017 22:48

Es necesario crear un enlace simbólico que simule la ubicación original

manualmente, en una consola de Administrador:

```
mklink /D <nombre_del_enlace> <path_real>
```

mediante una extensión del *shell* del Explorador de Windows



Link Shell Extension
Hermann Schinagl

14,6 MB
15/07/2017

<http://schinagl.priv.at/nt/hardlinkshellex/linkshellextension.html>

Para que no se produzcan avisos (*warnings*) será necesario añadir en `ng serve` un modificador

```
ng serve --preserve-symlinks
```

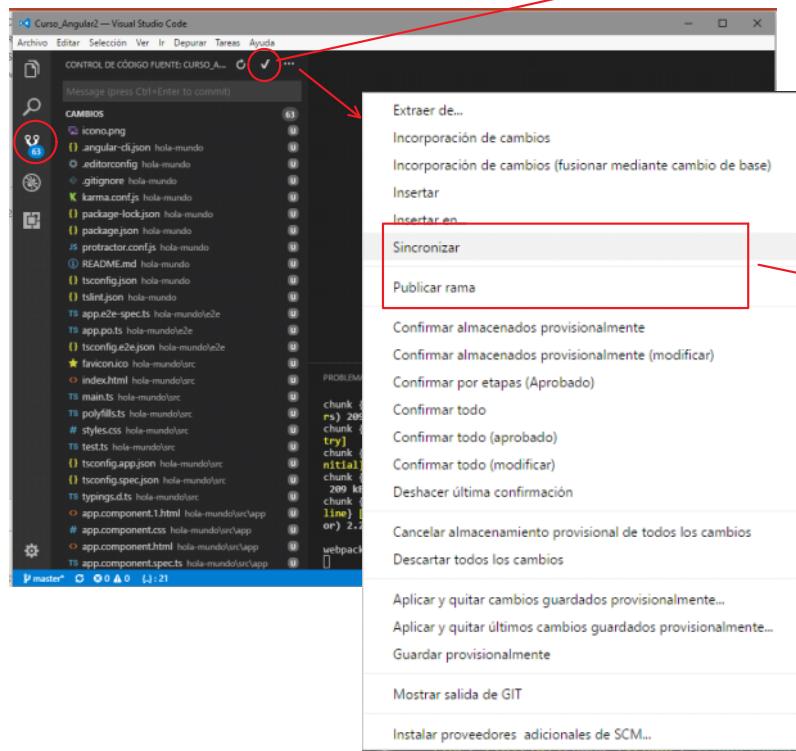
Es posible añadir este valor al comando `npm start` definido en `package.json`

Publicar en GitHub

domingo, 10 de septiembre de 2017 19:00

Desde el propio VSC podemos incorporar los proyectos al repositorio de **GitHub**

git commit



IMPORTANTE

.gitignore

Dependency directories
node_modules/
jspm_packages/

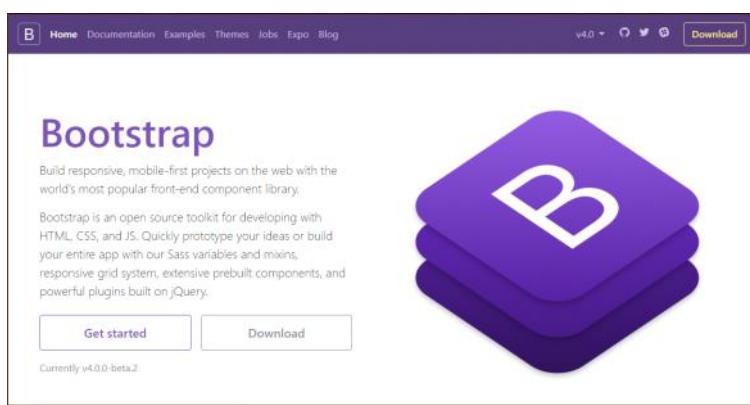
A screenshot of a GitHub repository page for 'alce65 / Curso_Angular2'. The page shows a commit history with 2 commits, 1 branch, 0 releases, and 1 contributor. The latest commit is '44efb97 26 seconds ago'. The commit details show files like 'hola-mundo', 'hola-mundo2', '.gitignore', 'LICENSE', 'README.md', and 'icono.png'. Below the commit history, there is a section titled 'Curso_Angular2' with a description 'Curso de Angular2 en Icono Training Consulting'.

CSS: Bootstrap

lunes, 6 de noviembre de 2017 20:37

Probablemente la más conocida entre las alternativas para construir el UI en las aplicaciones Angular

Actualmente en su versión 4.0



Instalación

```
npm install bootstrap@4.0.0-beta.2
```

- añade la carpeta Bootstrap en *node_modules*
- actualiza las dependencias en *package.json*

Más información

<https://medium.com/codingthesmartway-com-blog/using-bootstrap-with-angular-c83c3cee3f4a>

Utilización

Incorporamos el CSS en el fichero de configuración de angular cli: *.angular-cli.json*

```
"styles": [  
  "../node_modules/bootstrap/dist/css/bootstrap.min.css",  
  "styles.css"  
,
```

Existe un problema en angular/cli para acceder a los estilos cuando *node_modules* es un link simbólico. En ese caso hay que añadir Bootstrap desde *styles.css* empleando un import

```
styles.css:  
@import  "../node_modules/bootstrap/dist/css/bootstrap.min.css",
```

Componentes ng-bootstrap

Los componentes de Bootstrap han sido migrados a Angular como parte de <https://ng-bootstrap.github.io/#/home>

The screenshot shows the official GitHub repository page for 'ng-bootstrap'. At the top, there's a large blue header with a white 'B' logo. Below it, the text 'Bootstrap 4 components, powered by Angular' is displayed. A note says 'Currently at v1.0.0-beta.5'. There are two buttons: 'Demo' and 'Installation'. Below the header, there are two sections: 'Native' (represented by a CPU icon) and 'Widgets' (represented by a computer monitor icon). The 'Native' section describes it as 'Angular - specific widgets built from ground up using Bootstrap 4 CSS, APIs that makes sense in the Angular ecosystem. No dependencies on 3rd party JavaScript.' The 'Widgets' section describes it as 'All the Bootstrap widgets (ex. carousel, modal, popovers, tooltips, tabs, ...) and several additional goodies (datepicker, rating, timepicker, typeahead)'.

Instalación

Se instalan mediante npm

```
npm install @ng-bootstrap/ng-bootstrap
```

Utilización

Para utilizarlo, se importa en el módulo principal, ejecutando el método `forRoot()`:

```
import {NgbModule} from '@ng-bootstrap/ng-bootstrap';

@NgModule({
  ...
  imports: [NgbModule.forRoot(), ...],
```

Y se importa normalmente en otros módulos que tengan que utilizar los componentes

```
import {NgbModule} from '@ng-bootstrap/ng-bootstrap';
```

```
@NgModule({
  ...
  imports: [NgbModule, ...],
  ...
})
```

Ejemplo

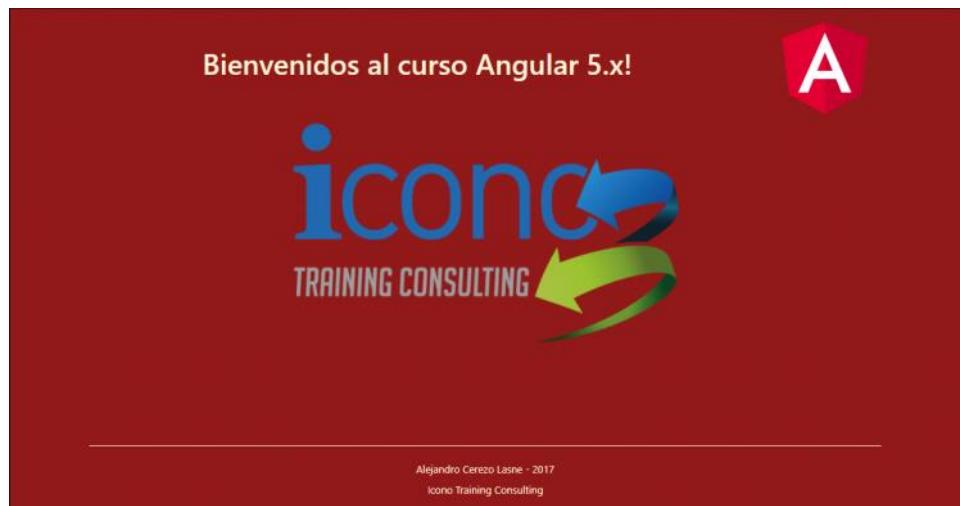
lunes, 6 de noviembre de 2017 22:00

```
<div class= "container">
  <header class="row align-items-center">
    <div class="col-10 text-center">
      <h1>Bienvenidos al curso {{curso}}!</h1>
    </div>
    <div class="col-2 logo">
      <img [src]="logoAngular" alt="logotipo de Angular">
    </div>
  </header>
  <article class="row">
    <div class="col-6 offset-3 logo">
      <img class="img-fluid" [src]="iconoLogo" alt="logotipo de Icono">
    </div>
  </article>
  <div class="row">
    <footer class="col-10 offset-1 text-center">
      <p>{{formador}} - {{fecha}}</p>
      <p>{{empresa}}</p>
    </footer>
  </div>
</div> <!--Fin del container-->
```

CSS

```
header {
  color : papayawhip;
}
footer {
  position: fixed;
  bottom : 0;
  left:0;
  border-top: 1px papayawhip solid;
  padding: 1em;
  font-size: 0.9em;
  color : papayawhip;
}
footer p {
  margin: 0.5em
}
```

```
body {
  background-color:#8D1919;
}
```



JS: MomentJS

domingo, 4 de febrero de 2018 22:32

<https://momentjs.com/>



1. Añadir la librería como script declarado explícitamente en la configuración de Angular CLI

```
"scripts": ["./node_modules/moment/min/moment.min.js"],
```

2. Importar Moment en los componentes que lo utilizan

```
import * as moment from 'moment';
```

Angular cli: despliegue

domingo, 10 de septiembre de 2017 10:40

Angular-cli incluye un comando para preparar el despliegue de la aplicación

Cuando queremos publicar la aplicación en **producción** tenemos que generar los archivos optimizados y publicarlos en un servidor web

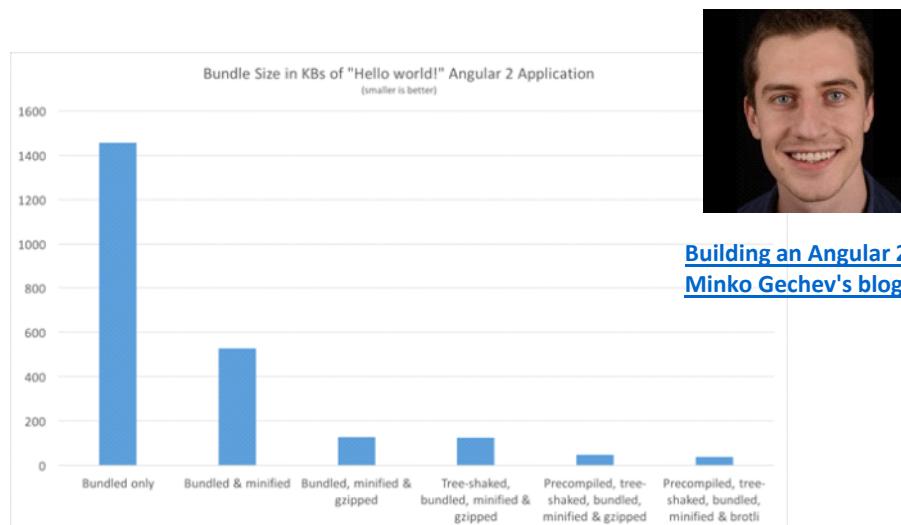
ng build

se generan en la carpeta *dist* los ficheros conocidos como *bundle*

Inicialmente no optimizaba el resultado
(main.bundle.js ocupa 2,5 megas),

Sucesivas versiones de angular-cli han ido ajustando la configuración de *webpack* hasta llegar a un *bundle* de 50Kb

Mejoras del bundle



<http://blog.rangle.io/optimize-your-angular2-application-with-tree-shaking/>

Generamos la aplicación Hola-mundo para comparar los *bundles* con los que se generan temporalmente en desarrollo

Se optimiza con la opción *ng build -t production*
equivale a *ng build --target development / production*
o sus alias *ng build -dev / -prod*

Flag	--dev	--prod
--aot	false	true
--environment	dev	prod
--output-hashing	media	all
--sourcemaps	true	false
--extract-css	false	true

--named-chunks	true	false
--build-optimizer	false	true (with AOT and Angular 5)

Compilación

domingo, 4 de febrero de 2018 20:49

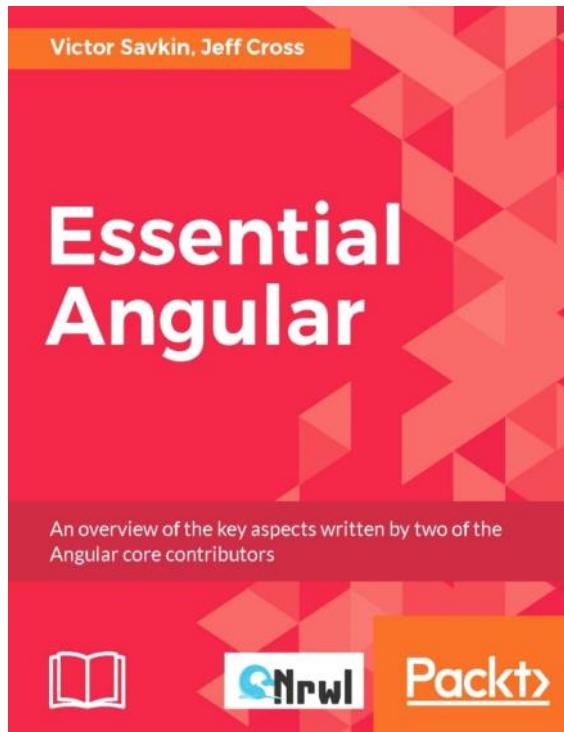
Angular incorpora un compilador de HTML (*HTML Compiler*) cuya función es

- recorrer el documento y localizar las directiva
- ejecutar los comportamientos asociados a esas directivas.

Este proceso puede tener lugar en 2 momentos diferentes

Just-in-time (JIT: durante la ejecución del código (*runtime*), cuando arranca (*bootstrapping*) la aplicación, como ocurre en AngularJS

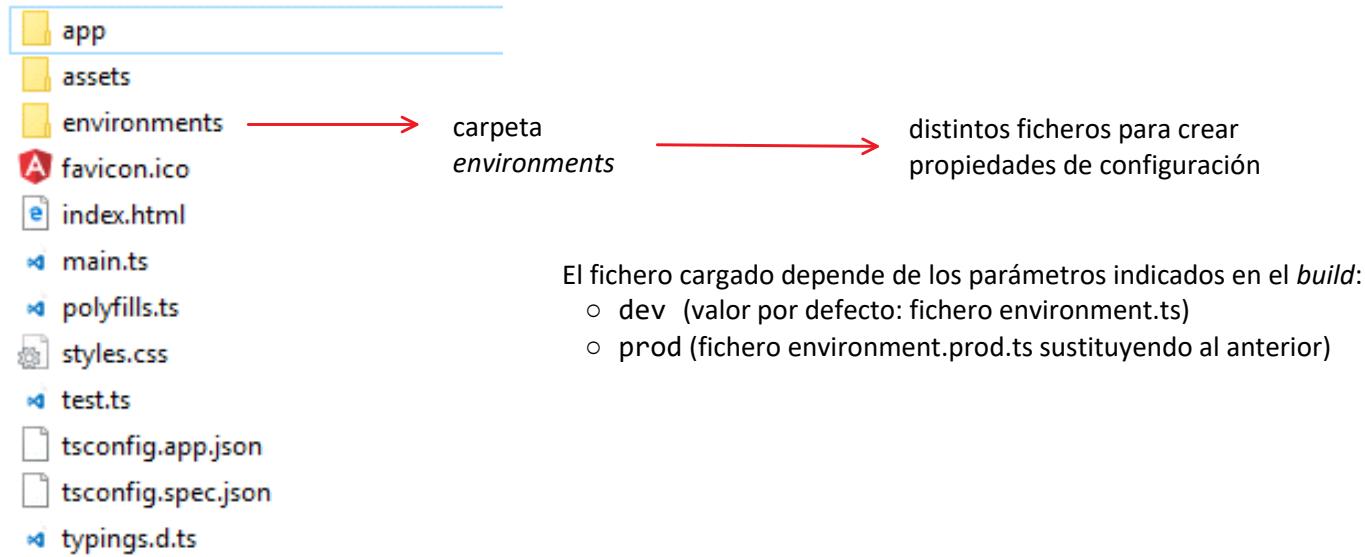
Ahead-of-time (AOT): durante el proceso de empaquetado y construcción (*build*) de la aplicación, con una considerable mejora del rendimiento



Essential Angular
Victor Savkin & Jeff Cross
Packt Publishing, 2017

Environments

domingo, 4 de febrero de 2018 21:09



ECMAScript 6 (ES6 / ES2015)

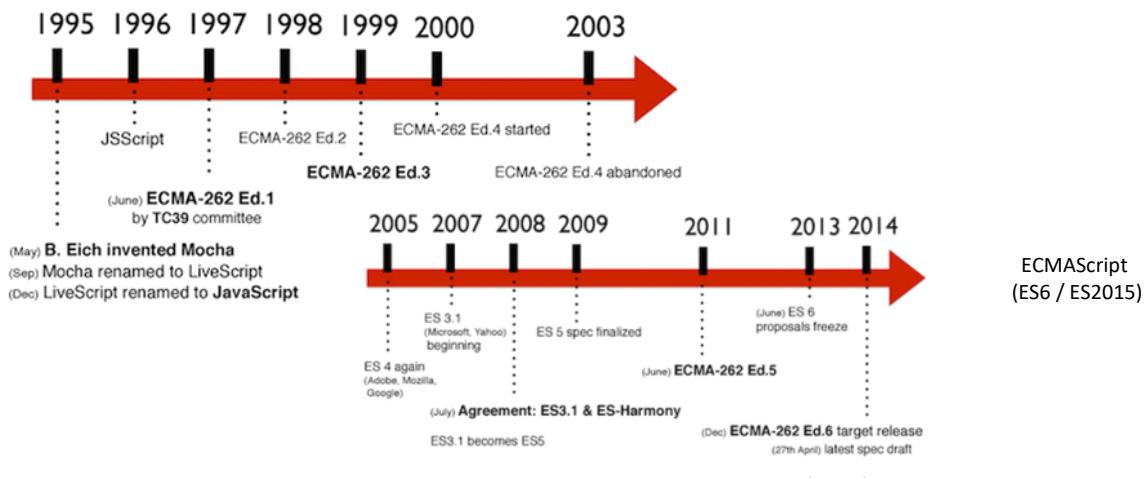
sábado, 9 de septiembre de 2017 13:33

Brendan Eich

Netscape



European Computer Manufacturers' Association



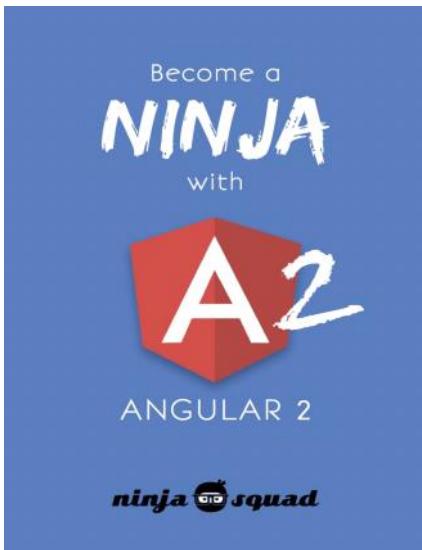
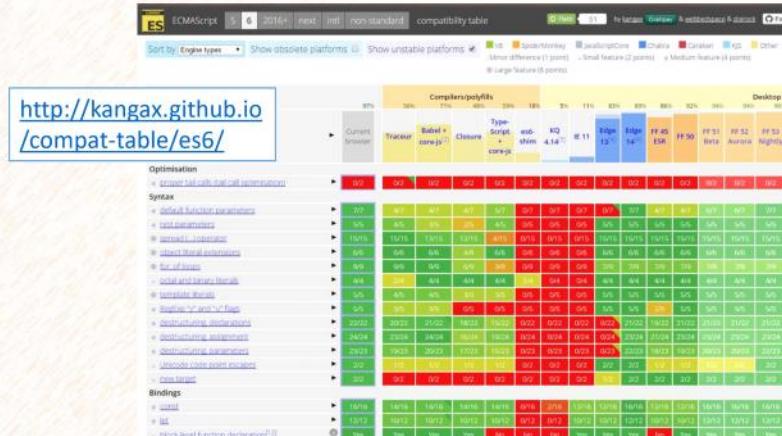
- **Constates (const).** Variables con ámbito (let)
- **Función Arrow.** This "semántico"
- **Template Strings:** interpolación de variables
- **Valores por defecto**
- **Clases (class)**
- **Módulos (export / import)**
- **Promesas (promise)**
- **Destructuring ...**

Más información

JS

ECMAScript 6 — New Features: Overview & Comparison

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



Become a ninja with Angular2

Cédric Exbrayat

Ninja Squad, 2016

En uno de sus primeros capítulos hace un resumen del nuevo estándar desde la perspectiva de Angular

Nuevos elementos de código

domingo, 30 de julio de 2017 22:19

- **Constates (const).** Variables con ámbito (let)
- **Función Arrow**
- **Template Strings:** interpolación de variables

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

Otros elementos:

- Nuevos objetos iterables Map y Set
- Bucle for ... of
- Valores por defecto en funciones y métodos

Nuevos objetos iterables de tipo Map, inicializados de 2 de las formas posibles

```
let map = new Map()
.set("A",1)
.set("B",2)
.set("C",3);

let map2 = new Map([
  [ "A", 1 ],
  [ "B", 2 ],
  [ "C", 3 ]
]);
```

<https://hackernoon.com/what-you-should-know-about-es6-maps-dc66af6b9a1e>

bucles que iteran a través de los elementos de objetos iterables (incluyendo Array, Map, Set, el objeto arguments, etc.).

```
let aDatos = [10,20,30]
let nTotal1 = "";
for (let dato in aDatos) {
  nTotal1 += dato
  console.log(dato);
}
// "0"
// "1"
// "2"
console.log(`Total : ${nTotal1}`);
```

El bucle for...in, adecuado para Objetos, producía resultados incongruentes en el caso de los Arrays

```
let nTotal2 = 0;
for (let dato of aDatos) {
  nTotal2 += dato
  console.log(dato);
}
```

El bucle for...of se comporta igualmente en el caso de Objetos, añadiendo un comportamiento más coherente en caso de Arrays

```
// 10
// 20
// 30
console.log(`Total : ${nTotal2}`);
```

Variables y constantes

sábado, 30 de diciembre de 2017 12:47

la declaración con `var` siempre se "eleva" al inicio del código, independientemente de que acompañe o no a una inicialización

```
(function prueba_var () {
    console.log(x)
    var x = 20
})()
```

Equivale a

```
var x
(function prueba_var () {
    console.log(x)
    x = 20
})()
```

devuelve undefined

la declaración con `let` no se eleva

```
(function prueba_let () {
    console.log(x)
    let x = 20
})()
```

ReferenceError x is not defined

Además su ámbito de existencia está limitado al bloque en que se declara y los bloques contenidos en el

```
(function bloques () {
    let x = 0
    let y = 0
    {
        x = 20
        let y = "Modificada"
        console.log(x)
        console.log(y)
        let z = 25
    }
    console.log(x)
    console.log(y)
    console.log(z)
})();
```

20
Modificada
20
0
ReferenceError z is not defined

`const` declara como constantes los tipos elementales o las referencias a los objetos, pero nunca el contenido de los objetos. Para ello disponemos del método de ES5 `Object.freeze()`

```
(function constantes () {
    const MES = "Enero"
```

```
const DIAS = 31
const USER = {
  name : "",
  apellido : "",
  puesto : ""
}

USER.name = "Pepe"
USER.apellido = "Perez"
USER.edad = 25
delete USER.puesto
```

```
console.log(USER) → { name: 'Pepe', apellido: 'Perez', edad: 25 }

USER = {} →
```

En un objeto "constante" podemos modificar, añadir o eliminar propiedades.

No podemos reasignar el objeto
TypeError: Assignment to constant variable.

Operador de propagación

domingo, 28 de enero de 2018 23:51

El operador de propagación *spread operator* permite que una expresión sea expandida en situaciones donde se esperan múltiples argumentos (llamadas a funciones) o múltiples elementos (*arrays literales*).

Llamadas a funciones:
`f(...iterableObj);`

Arrays literales:
`[...iterableObj, 4, 5, 6]`

Desestructuración *destructuring*:
`[a, b, ...iterableObj] = [1, 2, 3, 4, 5];`

Se define una función que espera múltiples argumentos

```
function f(x, y, z) { }
```

La función es invocada pasándole un array con el operador de propagación, que será expandido a los múltiples argumentos que espera la función

```
var args = [0, 1, 2];
f(...args);
```

<http://www.etnassoft.com/2014/06/03/el-operador-de-propagacion-en-javascript-ecmascript-6-y-polyfill/>

Desestructuración

Lunes, 29 de enero de 2018 0:02

La desestructuración no es un concepto nuevo en programación. De hecho, eso es algo que se ha tenido muy en cuenta a la hora de fijar el estándar: incorporar de forma progresiva al lenguaje Javascript lo mejor de otros.

- en Python o en OCaml, tendríamos las tuplas
- en PHP las listas
- sus correspondientes en Perl y Clojure...

una expresión que permite asignar valores a nombres conforme a una estructura de tabla dada

Desestructuración de ARRAYS

```
const aNumbers = [1, 2, 3, 4]  
  
const [uno, dos, tres] = aNumbers  
const [uno, dos, tres] = [1, 2, 3]  
console.log(uno, dos, tres)
```

Desestructuración de un array declarando las variables

Desestructuración de un objeto

```
{  
  const oNumbers = {  
    uno: 1,  
    dos : 2,  
    tres : 3,  
    cuatro : 4  
  }  
  
  const {uno , cuatro, tres, dos} = oNumbers  
  
  console.log(uno, dos, tres, cuatro)  
}
```

Cada una de las variables recibe el valor de uno de los miembros del objeto

Ejemplos en el fichero basicos.4destructuring.js

<http://www.etnassoft.com/2016/07/04/desestructuracion-en-javascript-parte-1/>

Valores por defecto

Lunes, 29 de enero de 2018 0:11

ES6: parámetros por defecto y
desestructuración del paso de parámetros

```
function drawCircleES6( {radius = 30,  
                        coords = { x: 0, y: 0}  
                      } = {} ) {  
  console.log(radius, coords);  
}
```

Desestructuración:

```
{radius = 30,  
coords = { x: 0, y: 0}} = {}
```

valores por defecto
desestructurados:
- radius
- coords

parámetro real
recibido como objeto

Diversas llamadas a la función

```
drawCircleES6(); // radius: 30, coords.x: 0, coords.y: 0  
drawCircleES6({radius: 10}); // radius: 10, coords.x: 0, coords.y: 0  
drawCircleES6({coords: {y: 10, x: 30}, radius: 10}); // radius: 10, coords.x: 30, coords.y: 10 }
```

Ejemplos en el fichero basicos.4.default.js

Función Arrow. This "semántico"

sábado, 14 de octubre de 2017 11:38

```
let oPrueba = {
    precio: 12,
    iva : 1.16,
};

oPrueba.calculaIvaAsiync = function () {
    setTimeout (function () {
        let precioFinal = this.precio * this.iva
        console.log(`Usando una función clásica:  
El precio final es ${precioFinal}`);
    });
}, 1000)
}

oPrueba.calculaIvaAsiync()
```

// la función callback del método
setTimeout
// interpreta this como una
llamada al sistema,
// no como el objeto en el que se
ha definido

// Versión alternativa
usando una arrow function

```
oPrueba.calculaIvaAsiync_Arrow = function () {
    setTimeout (() => {
        let precioFinal = this.precio * this.iva;
        console.log(`Usando una arrow function:  
El precio final es ${precioFinal}`);
    });
}, 1000)
}

oPrueba.calculaIvaAsiync_Arrow();
```

// la función callback del método
setTimeout
// interpreta this semanticamente,
según donde se ha definido la
función que lo usa
// y no según donde se utiliza, que
supondría hacerlo como una llamada
al sistema.

Clases

sábado, 29 de julio de 2017 15:30

```
// Ejemplo de código en ES6
class Libro {}

class LibroTecnico extends Libro {
    constructor(tematica, paginas) {
        super(tematica, paginas);
        this.capitulos = [];
        this.precio = "";
        // ...
    }
    metodo(pValor = "foo") {
        // ...
    }
}
```

Clase "padre"

Clase que hereda de la anterior

Constructor

Método que define valores por defecto

NO EXISTEN

- Propiedades definidas fuera de los métodos
- Modificadores de acceso (*private*, *protected*, *public*)
- Interfaces

Estos elementos se añaden en la implementación de las clases propia de *TypeScript*

También existe el modificador **Static**

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

Ejemplo de Clases

sábado, 14 de octubre de 2017 17:33

Declaración de una clase

```
class Libro {  
    constructor(tematica, paginas) {  
        this.tematica = tematica  
        this.paginas = paginas  
    }  
}
```

constructor

Declaración de una clase que hereda de la anterior

```
class LibroTecnico extends Libro {  
    constructor(tematica, paginas, precio) {  
        super(tematica, paginas);  
        this.capitulos = [];  
        this.precio = precio;  
        // ...  
    }  
}
```

constructor que invoca el constructor de la clase padre

Instanciación de objetos

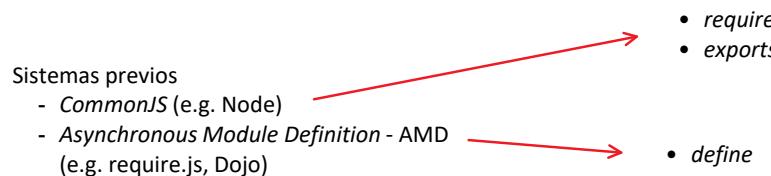
```
let libro1 = new LibroTecnico("Informatica", 250, 30)  
console.dir(libro1)  
console.dir(`Precio final: ${libro1.precioFinal()} €`)  
console.dir(`En Canarias : ${libro1.precioFinal(0)} €`)
```

ejemplo de método con parámetros con valor por defecto

```
LibroTecnico {  
    tematica: 'Informatica',  
    paginas: 250,  
    capitulos: [],  
    precio: 30 }  
'Precio final: 34.8'  
'En Canarias : 30'
```

Módulos

sábado, 29 de julio de 2017 15:30



Síncrono y asíncrono

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

definición de un módulo:
corresponde al fichero
que lo incluye

función
exportada

```
//File: modulo.js
export const hello = (nombre) => {
    return "Hola " + nombre;
}
```

Uso del módulo anteriormente creado

The diagram shows the import and use of a module. It starts with 'importación de una función' (Importing a function) which points to code in 'app.js'. This code imports the 'hello' function from 'modulo.js'. Below it, 'objeto en el que se usa la función importada' (Object where the imported function is used) points to another part of 'app.js' where the 'saludo' method uses the imported 'hello' function.

importación de una función → //File: app.js
import { hello } from "./modulo.js";

objeto en el que se usa la función importada → var app = {
 saludo : () => {console.log(hello("Carlos"));}
};
app.saludo()

Problema:

El estándar ES6 describe como se declaran los módulos,
pero no especifica cómo deben ser cargados

Hasta muy recientemente NO DISPONIBLE EN LOS NAVEGADORES

Tampoco es soportado en *NodeJS*, que continua usando su propia definición de módulos

Actualmente, se puede indicar al navegador que procese correctamente los archivos JavaScript que usan módulos utilizando el atributo `type="module"`

```
<script src="app.js" type="module"></script>
```

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.catch(callback)

promise.finally(callback)



Callbacks anidados

sábado, 14 de octubre de 2017 13:52

```
function msgAfterTimeout (msg, nombre, tiempo, cb) {
    setTimeout(function () {
        cb(msg, nombre);
    }, tiempo);
};

msgAfterTimeout("", "Pepe", 100,
    function (msg, nombre) {
        let saludo = (`${msg} Hola ${nombre}!`);

        msgAfterTimeout(saludo, "Juan", 200,
            function (msg, nombre) {
                let saludo = (`${msg} Hola ${nombre}!`)

                console.log(`Saludo después de 0,3 seg: ${saludo}`);
            } // Fin de la función callback
        ); // Fin de la llamada a msgAfterTimeout
    } // Fin de la función callback
); // Fin de la llamada a msgAfterTimeout
```

Se declara una función asincrónica

En ella se ejecuta la función recibida como callback dentro del setTimeout. Para poder pasárle parámetros hay que incluirla en una función anónima

Se invoca la función asincrónica

función enviada como callback

función enviada como callback

Se invoca nuevamente la función asincrónica

operación con el resultado acumulado de los dos callbacks

Promesas en ES6

jueves, 10 de agosto de 2017 20:35

Implementación new Promise

el objeto promesa recibe como parámetros dos funciones:

- La función "resolve": se ejecutará cuando queramos finalizar la promesa con éxito.
- La función "reject": se ejecutará cuando queramos 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);  
    })  
}
```

En este caso la promesa siempre se resuelve correctamente; la función admite como parámetro los datos que la promesa deba retornar

Utilización

a la función que retorna el objeto promesa se le encadenan el método then con dos funciones como parámetros,

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

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

Alternativamente puede utilizarse el método catch para declarar 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.'))
```

https://developer.mozilla.org/es/docs/Web/JavaScript/Referencia/Objetos_globales/Promise

Ejemplo de promesas en ES6

sábado, 14 de octubre de 2017 14:43

```
function msgAfterTimeout (msg, nombre, tiempo) {  
    return new Promise((resolve, reject) => {  
        setTimeout(  
            () => resolve(`${msg} Hola ${nombre}!`),  
            tiempo  
        )  
    })  
}
```

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 promesa, encadenando las llamadas a ellas

```
msg almacena el resultado del primer proceso asincrónico  
  
msg almacena los sucesivos resultados de los procesos asincrónicos  
  
msgAfterTimeout("", "Pepe", 100)  
.then((msg) =>  
    msgAfterTimeout(msg, "Juan", 200))  
.then((msg) => {  
    console.log(`Saludo después de 0,3 seg: ${msg}`)  
})
```

operamos finalmente con *msg*

```
MENSALES  
Saludo después de 0,3 seg: Hola Pepe! Hola Juan!  
PS D:\Desarrollo\Front_End_alce65\Angular\angular_4_2017\02_tecnologias\ES6>
```

Generadores

Lunes, 29 de enero de 2018 23:47

Procesar cada uno de los elementos en una colección es un tipo de operación muy común. JavaScript proporciona diversas formas de iterar a través de los elementos de una colección, desde simples bucles for hasta map(), y filter(). Los iteradores y los generadores acercan el concepto de iteración directamente al núcleo del lenguaje y proporcionan un mecanismo para personalizar el comportamiento de los bucles for...of.

https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Iterators_and_Generators

- Iteradores
- Iterables
- Generadores

Son **funciones especiales** (se declaran con function *) de las que se puede salir y entrar varias veces con resultados diferentes

- devuelven valores gracias a la palabra reservada yield
- continúan cuando se ejecuta el método next

<https://carlosazaustre.es/funciones-generadoras-en-ecmascript6/>

<https://speakerdeck.com/serabe/generadores-en-javascript>

Operaciones con arrays en JS 1.5

Js

- `map()`,
- `filter()`,
- `some()`,
- `every()`,
- `forEach()`,
- `reduce()`,
- `reduceRight()`,

En todos los nuevos métodos de utilizar una **función callback**, es decir una función que es pasada como parámetro para que el método la utilice de la forma en que tiene previamente definida.

Los parámetros de dicha función son (element, index, array)



Arrays en JS 1.5 (1)

Js

`map()` una proyección (como `select` en C#): el argumento es una función que transforma cada uno de los elementos.

```
array.map(function(i){return i.toUpperCase()});
```

convertiría cada elemento del array en mayúsculas

```
var numbers = [1, 4, 9];
var roots = numbers.map(Math.sqrt);
document.write("roots is : " + roots );
```

mostraría las raíces cuadradas de los elementos del array

Un ejemplo más complejo, podría transformar en objetos cada uno de los elementos del array

ECMAScript 5.1 (ECMA-262)

Arrays en JS 1.5 (2)

Js

`filter()` tiene como argumento una función lambda que evalúa cada elemento del array y devuelve un booleano. Se devuelve un **nuevo array** sólo con los elementos que hayan dado verdadero en la función callback

```
array.filter(function(i){return i[0]==="a"});
```

```
function isBigEnough(element, index, array) {  
    return (element >= 10);  
}  
var aDatos = [12, 5, 8, 130, 44]  
var aFiltrado = aDatos.filter(isBigEnough);  
console.log(aFiltrado);
```

Ejemplo que devuelve [12,130,44]

ECMAScript 5.1 (ECMA-262)

Arrays en JS 1.5 (3)

Js

`some()` y `every()` utilizan funciones del mismo tipo para evaluar si el array en su conjunto las cumple, y devolver en consecuencia verdadero a falso.

- `some()` devuelve verdadero si algún elemento del array lo devuelve
- `every()` devuelve verdadero si todos los elementos del array lo devuelven

```
function isBigEnough(element, index, array {  
    return (element >= 10);  
}  
var aDatos = [12, 5, 8, 130, 44]  
var isVal_some = aDatos.some(isBigEnough);  
var isVal_every = aDatos.every(isBigEnough);  
console.log(isVal_some);  
console.log(isVal_every);
```

true
false

ECMAScript 5.1 (ECMA-262)

Arrays en JS 1.5 (4)

Js

`forEach()` permite indicar cualquier función , booleana o no, que modifique o no los elementos, pero que en cualquier caso se aplica sobre cada uno de ellos.

```
function printBr(element, index, array) {  
    document.write("<br />[" + index + "] is " + element );  
}  
aDatos = [12, 5, 8, 130, 44];  
aDatos.forEach(printBr);
```

[0] is 12
[1] is 5
[2] is 8
[3] is 130
[4] is 44

```
function cuad(element, index, array) {  
    array[index] = element * element;  
}  
aDatos = [12, 5, 8, 130, 44];  
aDatos.forEach(cuad);  
console.log(aDatos);
```

[144, 25, 64, 16900, 1936]

ECMAScript 5.1 (ECMA-262)

Arrays en JS 1.5 (5)

Js

`reduce()`, aplica una función simultáneamente a pares de valores del array, desde la izquierda a la derecha, sucesivas veces, hasta reducir el array a un único valor

`reduceRight()`, realiza el mismo proceso desde la derecha a la izquierda, de nuevo sucesivas veces, hasta reducir el array a un único valor

```
var aDatos = [1, 2, 3, 4];
var nTotal = aDatos.reduce(function(a, b){ return a * b; });
console.log(aDatos);
console.log(nTotal);
```

[1, 2, 3, 4]

24

ECMAScript 5.1 (ECMA-262)

TypeScript

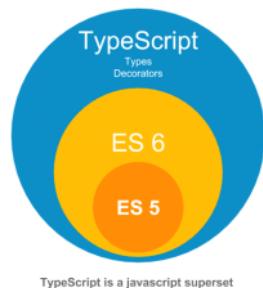
sábado, 9 de septiembre de 2017 13:34



<http://www.typescriptlang.org/>

Open source

Super Set de JavaScript ES6
(=> es JavaScript)



- Orientado a Objetos con clases
- Añade **tipos estáticos**
 - Inferencia de tipos
(no hay que declararlos en muchos sitios)
 - Tipos opcionales (si no quieres, no los usas)
- Anotaciones / Decoraciones
- Es **transpilado**: se genera código JavaScript ES5 (compatible con los navegadores web actuales)

Documentación on-line

The screenshot shows a GitBook page titled "TypeScript Deep Dive". The page has a header with "Pricing", "Explore", "About", "Blog", "Sign In", and "Sign Up". Below the header, it says "basarat > TypeScript Deep Dive" and "Updated 16 hours ago". There are sections for "ABOUT" and "44 DISCUSSIONS". At the bottom right, there are buttons for "Download PDF" and "Read".

<https://www.gitbook.com/book/basarat/typescript/details>

Basarat Ali Syed



Pruebas del código

<http://www.typescriptlang.org/play/>

The screenshot shows the TypeScript Playground interface. On the left, there is a code editor with the following TypeScript code:

```
function Greeter(greeting: string) {
  this.greeting = greeting;
}

Greeter.prototype.greet = function() {
  return "Hello, " + this.greeting;
}

let greeter = new Greeter("world");
let button = document.createElement('button');
button.textContent = "Say Hello";
button.onclick = function() {
  alert(greeter.greet());
};

document.body.appendChild(button);
```

On the right, there is a preview area showing the generated JavaScript code:

```
function Greeter(greeting) {
  this.greeting = greeting;
}

Greeter.prototype.greet = function () {
  return "Hello, " + this.greeting;
}

let greeter = new Greeter("world");
let button = document.createElement('button');
button.textContent = "Say Hello";
button.onclick = function () {
  alert(greeter.greet());
};

document.body.appendChild(button);
```

At the top, there are tabs for "TypeScript" (which is selected), "Share", and "Options". At the bottom, there are "Run" and "JavaScript" buttons.

Clases

sábado, 29 de julio de 2017 15:33

```
// ejemplo de clase en TypeScript
export class Empleado {
    private nombre: string;
    private salario: number;

    constructor(nombre: string, salario: number) {
        this.nombre = nombre;
        this.salario = salario;
    }

    getNombre() {
        return this.nombre;
    }

    toString() {
        return `Nombre: ${this.nombre} , Salario: ${this.salario}`;
    }
}
```

clase →

propiedades →

constructor →

métodos →

- Al estándar de ES6 se le añaden
- Propiedades definidas fuera de los métodos
 - Modificadores de acceso (*private*, *protected*, *public*)
 - Interfaces

Herencia

```
class Animal {
    constructor(public name: string) { }
    move(distanceInMeters: number = 0) {
        console.log(`"${this.name}" moved ${distanceInMeters}m.`);
    }
}

class Snake extends Animal {
    constructor(name: string) { super(name); }
    move(distanceInMeters = 5) {
        console.log("Slithering...");
        super.move(distanceInMeters);
    }
}

class Horse extends Animal {
    constructor(name: string) { super(name); }
    move(distanceInMeters = 45) {
        console.log("Galloping...");
        super.move(distanceInMeters);
    }
}
```

Herencia a partir de una clase padre →

Llamada al constructor de la clase padre →

Sobre escritura de los métodos de la clase padre →

Interfaces

Los interfaces son siempre públicos, por lo que no se utilizan modificadores de acceso

```
interface Usuario {
    id: number,
    name: string,
    dirección: {calle: string,
               num: number,
               zip: string}
    formación?: Array<string>
    saludar: Function;
    calcularPrecio(iva: number): void;
}
class Socio implements Usuario {
    id: number;
    name: string;
    dirección: {calle: string,
               num: number,
               zip: string}
    saludar() {
        console.log(`Hola, saludos de ${name}`);
    };
    calcularPrecio(iva) {
        ...
    }
}
```

elemento opcional, no tiene que estar en la implementación

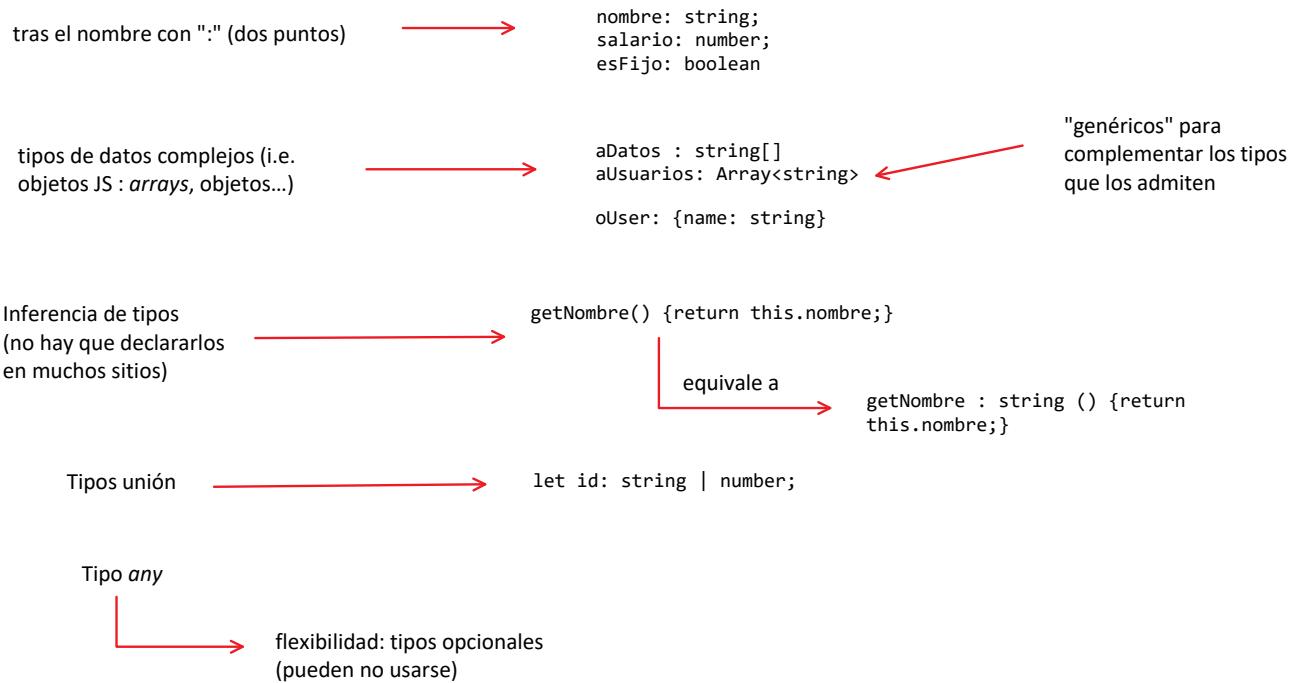
Método declarado en el interfaz y su correspondiente implementación

Los interfaces también se utilizan para definir tipos, como veremos a continuación

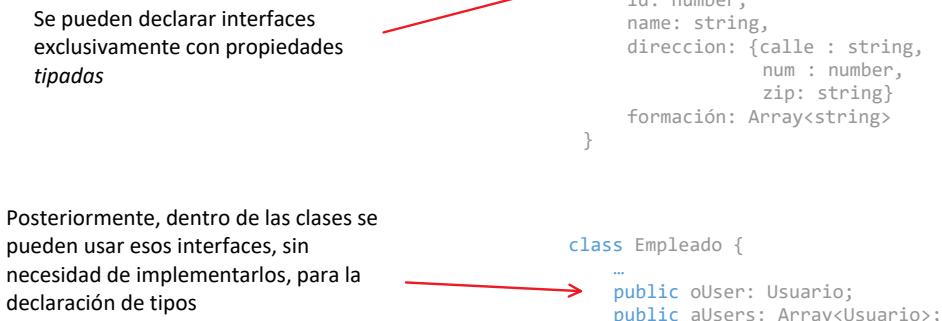
Tipos

domingo, 10 de septiembre de 2017 22:40

Tipos estáticos en tiempo de desarrollo;
evidentemente desaparecen tras la compilación (*traspilación*) a JS



Interfaces y tipos



A partir de ahí se pueden crear objetos literales "tipados" por interfaces

Alias

Crea un nuevo nombre para un tipo o conjunto de tipos

```
type Name = string;
type NameResolver = () => string;
type NameOrResolver = Name | NameResolver;
```

- aumenta el contenido semántico de los tipos
- agrupa conjuntos de tipos

<https://www.typescriptlang.org/docs/handbook/advanced-types.html>

Módulos (y elementos de ES6)

domingo, 10 de septiembre de 2017 22:49

empleado.ts
fichero en el que se crea y
exporta una clase

```
export class Empleado {  
    nombre : string;  
    salario: number;  
  
    constructor (pNombre, pSalario)  
    {  
        this.nombre = pNombre;  
        this.salario = pSalario;  
    }  
}
```

sample.ts
fichero en el que se importa y utiliza
la clase anterior

```
import { Empleado } from "./empleado";  
  
let emps = new Array<Empleado>();  
  
emps.push(new Empleado('Pepe', 500));  
emps.push(new Empleado('Juan', 200));  
emps.push({"Luis",400})  
  
for (let emp of emps) {  
    console.log(emp.getNombre());  
}  
  
emps.forEach(emp => {  
    console.log(emp);  
});
```

Importación desde otro módulo (fichero).
Se sobreentiende la extensión .ts

Tipo Array de objetos de la clase importada

código ES6 dentro de TypeScript

La ausencia de soporte del estándar ES6 en los navegadores o en *Node* hace necesaria la transpilación a ES5
cuando se utilizan módulos en TS

Desde TS se configura el uso de *Node/CommonJS* o sistemas de módulos alternativos.

Decoradores o anotaciones

sábado, 14 de octubre de 2017 19:13

Ejemplo de función que define un *decorator* sencillo, sin argumentos

```
function course(target) {  
    Object.defineProperty(  
        target.prototype,  
        'course',  
        {value: () => "Angular 2"}  
    )  
}
```

Uso del anterior *decorator* para modificar una clase

```
@course  
class Person {  
    firstName;  
    lastName;  
    constructor(firstName, lastName) {  
        this.firstName = firstName;  
        this.lastName = lastName;  
    }  
}
```

Instanciación de un objeto de esta clase

```
let oPersona = new Person("Pepe", "Pérez");  
console.log(oPersona.course()); // Angular 2
```

Ejemplo de función que define un *decorator* con argumentos

```
function Student(config) {  
    return function (target) {  
        Object.defineProperty(  
            target.prototype,  
            'course',  
            {value: () => config.course}  
        )  
    }  
}
```

Uso del anterior *decorator* para modificar una clase, pasándole un argumento en forma de objeto

```
@Student({  
    course: "Angular 2"  
})  
class Persona {  
    firstName;  
    lastName;  
    constructor(firstName, lastName) {  
        this.firstName = firstName;  
        this.lastName = lastName;  
    }  
}
```

Instanciación de un objeto de esta clase

```
let oEstudiante = new Persona("Pepe", "Pérez");  
console.log(oEstudiante.course()); // Angular 2
```

La función recibe como parámetro el objeto de configuración

al definir el decorador utiliza la clave correspondiente del objeto recibido



Decoradores en Angular

domingo, 10 de septiembre de 2017 23:04

Importación de una clase desde otro módulo (fichero)

decorador de la clase a la que acompaña

exportación de la clase "decorada"

```
import {Component} from 'angular2/core';

@Component({
  selector: 'app',
  templateUrl: 'app.component.html'
})
export class AppComponent {
  // código
  @input() nombre: string;
}
```

El decorador es un objeto JSON que da valor a una serie de METADATOS (propiedades) que esperan ser definidas y que pasarán a formar parte de la clase decorada