Curso sobre Angular

Tabla de contenido

[Documentación 1](#_Toc496704412)

[Angular 1](#_Toc496704413)

[Javascript 1](#_Toc496704414)

[Typescript 1](#_Toc496704415)

[Instalación de entornos y herramientas de desarrollo 2](#_Toc496704416)

[Utilidades 3](#_Toc496704417)

[Instalar los módulos que son dependencias indicadas en el fichero “package.json” 3](#_Toc496704418)

[Arrancar el servidor a partir del script personalizado que se ha modificado en el fichero “package.json” 3](#_Toc496704419)

Ng new componentes –si (crea un nuevo proyectos sin nodemodules, si=skip instalation)

Seleccionar el origen del vínculo

Soltar como –> vínculo dinámico

Ng g c nombre (crea componente)

Ng g m nombre (crea módulo)

Ng g d nombre (crea directiva propia)

Ng g p nombre (crea pipe propio)

ngModel en lugar del atributo value para usar el [()] (banana in a box)

Profesor: Alejandro Cerezo

Email: [alce65@hotmail.com](mailto:alce65@hotmail.com)

Repositorio github: https://github.com/alce65

Pantalla compartida: screentask

# Documentación

Comparador de frameworks: todomvc.com

Angular.io: web de angular

Ejemplos de angular: [www.madewithangular.com](http://www.madewithangular.com)

Webcomponents.org (del w3c)

# Angular

* MVC en cliente
* Prácticamente todo es javascript
* Con el servidor solo intercambia datos mediante ajax, generalmente json y no html (jsp, jsf, ..) como se hace en entornos basados en 3 capas, como jsf, jsp, struts, … El cliente es quien ya hace todo con esos datos en json y maneja el dom/html/javascript
* Decoradores más importantes: module, component
* Component/template es el equivalente a controlador/vista
* Un módulo es una agrupación de elementos.
* Cada aplicación debe tener al menos un módulo y cada módulo al menos un componente
* Por convenio, el árbol de módulos debería coincidir con el árbol de carpetas, aunque no es obligatorio.
* En los import de los módulos nunca se pone la extensión .ts, porque si no dará problemas una vez compilado, ya que se convierte en .js

# Javascript

* Todo lo que va entre {}, las que sean (if, bucles, …) definen un ámbito de memoria independiente, por lo que las variables que están dentro de ese ámbito no existen fuera.

# Typescript

* Clases, interfaces, objetos, herencia, …
* Es tipado, aunque no es obligatorio indicarlo en la creación de propiedades, pero si se indica se obliga a mantener ese tipo.
  + Existe el tipo “any”, que es como en javascript, cualquier cosa, por lo que acepta number, boolean, string, …
  + Existen tipos unión. Ej: autor: number | string ;
* Por defecto las propiedades son public, pero se pueden definir prívate o protected.
* Decoradores. Es una etiqueta que aplicamos a una clase y que nos permite pasarle unos metadatos.

# Instalación de entornos y herramientas de desarrollo

* Visual studio code (entorno de desarrollo)
  + Instalar extensiones:
    - Html boilerplate
    - Angular 4 and typescript/html …
    - Angular 5 snippets
    - Angular Language Service
    - Angular 5 typescript snipptes
    - Angular2 – inline
    - Auto import
    - Debugger for chrome
    - Path intellisense
    - Tslint
* Hay versiones de chrome y firefox específicas para desarrolladores
  + Chrome canary
  + Firefox developer
* Plugin de chrome para angular: Augury
* Git
  + Github
    - Repositorio del profesor: <https://github.com/alce65>
    - Mi repositorio: <https://github.com/abetor1> (registrado con [atorres@rae.es](mailto:atorres@rae.es) y con usuario abetor1)
  + Cómo clonar un repositorio desde github a git
    - Desde github pulsar en la opción de “clonar”
    - Desde la ventana de comandos, situado en el directorio donde queremos clonar, ejecutar el comando: git clone ctrl+v (lo clonado de github)
  + En el fichero “.gitignore” se indica qué no queremos subir al repositorio, por ejemplo lo que hay en “node-modules”. Esos “node-modules” se podrán reinstalar con “npm install”, ya que estaría en pacakage.json
  + Comandos git
    - https://git-scm.com/docs
    - https://www.hostinger.es/tutoriales/comandos-de-git
* Node.js
  + Lo usamos como servidor web integrado con visual studio code donde correrán nuestros desarrollos para pruebas
  + Además de node.js también instalamos npm (instalador de paquetes)
  + <http://localhost:4200> (puerto por defecto del node interno que sirve integrado en visual studio code)
  + “Npm –s” guarda en el package.json la configuración de lo que vamos instalando, para no subirlo a repositorio y que posteriormente con “npm –install” se reinstale.
* Angular-cli (herramientas de ayuda a angular, como compilador de typescript, empaquetador, generador de versiones javascript/html minimas para producción, …)
  + https://cli.angular.io
  + Incluye Webpack (<https://webpack.github.io>)
  + Lo instalamos mediante npm (npm install –g @angular/cli)
  + Ng –v (versión de angular-cli)
  + Ng new (genera un proyecto nuevo, creando toda la estructura y todos los ficheros/configuración básicos y necesarios)
  + Ng serve (compila, empaqueta y sirve)
    - ng serve --preserve-symlinks (no muestra los warnings de symlinks)
  + Ng build (genera una versión para llevar a otro servidor web)
  + Ng built –t production (genera una versión mínima para llevar a otro servidor web)

# Utilidades

## Instalar los módulos que son dependencias indicadas en el fichero “package.json”

Desde el directorio donde esté el “package.json” ejecutar “npm install” y se descarga todos los node-modules de los que tira.

## Arrancar el servidor a partir del script personalizado que se ha modificado en el fichero “package.json”

Desde el directorio donde esté el “package.json” ejecutar “Npm start”