Yago González Rozas

De angularJS a angular

Índice

¿Qué es angular?

¿Qué es angular?

- **Idea**: Crear un framework simple para web
- **Resultado**: Framework para web dinámicas
- **Versiones**: Oficialmente cada 6 meses
  - ***1 o angularJS*** - Versión inicial, sin componentización
  - ***2*** - Primer paso a la componentización, problemas con router
  - ***4*** - Estado actual

¿Cómo funciona angular?

¿Cómo funciona angular?

- **Base**: HTML + Javascript
- **Pasos**:
  1. Creación de páginas web HTML
  2. Lectura de los tags customizados definidos en dichas páginas
  3. Interpretación de los tags por angular
  4. Traducción directa a Javascript
- **Distribución**:
  - ***1 o angularJS*** - Módulos, directivas, factorías, servicios, etc.
  - ***2*** - Tipo especial de directivas, los componentes
  - ***4*** - Nuevo paradigma, sólo módulos, componentes y servicios

¿Qué necesita angular?

angular 1 vs angular 4

- **Común**
  - Instalación de un entorno gráfico, como Visual Studio Code
  - Instalación de Node
  - Instalación del gestor de dependencias, yarn o npm

- **angular 1 (angularJS)**
  - La creación de programas se hace a través de la creación por npm

- **angular 4**
  - Se tiene que instalar angular-cli a través del npm:

    npm -g install @angular/cli

Práctica 0: Crear una aplicación

¿Qué queremos?

- **Objetivo:** Crear una aplicación básica
- **Ejemplo:** Crear un ejemplo
- **Práctica:** Probar a crear una aplicación

¿Qué necesitamos?

- **Webpack**
  - Cargador javascript similar a gulp, grunt o requirejs
- **angular**
  - Diferente instalación, según sea versión 1 o 4

angular 1 - Pasos

1. Crear el directorio para nuestra aplicación
2. Entrar en el directorio
3. Ejecutar la inicialización del npm o yarn
4. Instalar las dependencias para desarrollador
   - webpack
   - webpack-dev-server o similar
5. Instalar las dependencias para ejecución
   - angular
6. Crear el directorio de carpetas
   - **Opción 1:** Todo junto -app-
   - **Opción2:** Source code -src- y distribution code -dist-

angular 1 - Pasos

7. Crear el archivo webpack.config.js en el directorio
   - Definirá los archivos js iniciales -al menos dos, el módulo inicial y el vendor-
   - Definirá los ficheros de output
   - Cargará los módulos necesarios
8. Añadir en el package.json los scripts necesarios
   - El script de build llamará al webpack con su fichero config para generar los output
   - El script de start llamará al server de webpack sobre la carpeta de distribución
9. Modificaciones en el directorio del source code
   - Crear el módulo inicial - Cargará el angular.module inicial, extensiones y template
   - Crear el fichero vendor.js, preferiblemente en core - Importará librerías necesarias
10. Modificaciones en el directorio del distribution code
    - Crear index.html, que contendrá la carga de los output y enlazará el html con el ng-app
11. Ejecutar el script de start

angular 4 - Pasos

1. Ejecutar ng new con el nombre de la aplicación a crear
   - Tiene múltiples opciones, como —prefix (-p) o —skip-tests (-st)
2. Ejecutar el comando start con el gestor de dependencias

Práctica 1: Componentes

¿Qué queremos?

- **Objetivo:** Crear una aplicación por componentes
- **Ejemplo:** Crear el cuerpo humano
- **Práctica:**

1. Crear un coche - Motor + Habitáculo + Maletero
2. Crear una casa - Habitación + Electrodomésticos

¿Qué necesitamos?

- **No hay requerimientos especiales**

angular 1 - Pasos

1. Cambios en el directorio del source code
   - Crear directorio cabeza para el módulo y componente cabeza
   - Crear directorio tronco para el módulo y componente tronco
   - Crear directorio extremidades para su módulo y componentes
   - Crear componente inicial
2. Cambios en el directorio cabeza
   - Módulo - Creará un angular module e importará los componentes
   - Component - Utilizará los subcomponentes y se exportará
   - Crear directorios ojo, boca y nariz, que contendrán un componente exportable

angular 1 - Pasos

3. Cambios en el directorio tronco
   - Módulo - Creará un angular module e importará el componente
   - Componente - Contendrá un componente exportable
4. Cambios en el directorio extremidades
   - Módulo - Creará un angular module e importará los componentes
   - Crear directorios brazo y pierna, que contendrán un componente exportable
5. Cambios en el módulo inicial para enlazar los submódulos
6. Cambios en el html inicial para usar los componentes

angular 4 - Pasos

1. Ejecutar ng g m para crear los módulos

   ng g m cabeza

2. Ejecutar ng g c con export para crear componentes principales

   ng g c cabeza/cabeza —flat —export

   ng g c extremidades/pierna —export

3. Ejecutar ng g c para crear componentes privados

   ng g c cabeza/ojo

4. Cambios en el módulo principal para importar los submódulos

5. Cambios en los html de los módulos para enlazar los componentes

Práctica 2: Enrutador

¿Qué queremos?

- **Objetivo:** Crear un ejemplo de enrutado
- **Ejemplo:** Crear un catálogo
- **Práctica:**
  1. Crear una librería
  2. Crear un directorio Europa / América / Asia

¿Qué necesitamos?

- **Enrutador**
  - Para angular 1, angular-ui-router + oclazyload (opcional)
  - Para angular 4, por defecto con —routing

angular 1 - Pasos

1. Instalar las dependencias para ejecución
   - angular-ui-router
   - oclazyload (opcional)
2. Crear el fichero config, que contendrá las rutas
   - Definir el state provider por defecto a inicio
   - Definir el state provider para enlazar a inicio, catálogo y sobre
3. Cambios en el directorio routes
   - Crear directorio inicio con su módulo y componente
   - Crear directorio catálogo con su módulo y componente
   - Crear directorio sobre con su módulo y componente
4. Enlazar el fichero de inicio, enlazándolo con los módulos y el config

angular 4 - Pasos

1. Crear directorio layout, que contendrá la estructura

ng g m core/layout —routing
ng g c core/layout/shell --export —flat
ng g c core/layout/top-bar
ng g c core/layout/main-content

2. Cambiar el routing de forChild a forRoot

3. Crear el módulo inicio

ng g m routes/inicio
ng g c routes/inicio/inicio —flat —export

4. Enlazar el módulo inicio
   - Cambiar el top-bar para crear la nav bar
   - Importar el componente para cambiar el routing del layout
   - Importar el módulo en el layout

angular 4 - Pasos

5. Crear el módulo catálogo por lazy routing

   ng g m routes/catalogo —routing

   ng g c routes/catalogo/catalogo --flat

6. Enlazar el catálogo
   - Cambiar el top-bar para añadir la ruta al catálogo
   - Cambiar el routing del layout para añadir la ruta al módulo
   - Cambiar su routing para añadir la ruta base
   - Crear el nav bar y el router outlet en su template

7. Crear el submódulo lista y nuevo y enlazarlo en el router de su padre

   ng g c routes/catalogo/nuevo

angular 4 - Pasos

enrutado lazy

Crear un modulo, el cual gestionara sus propias rutas. con un archivo ...routing.js

--routing: genera routing
--flat: no crea la carpeta interna, crea todo en mismo nivel
--export: exporta modulos,componentes..-.

8. Crear el módulo sobre

ng g m routes/sobre —routing

En routing poner la ruta, con path y el componenet que se va a mostrar

ng g c routes/sobre/sobre —flat

9. Enlazar el sobre
   - Cambiar el top-bar para añadir la ruta al catálogo `.html`
   - Importar el módulo en el layout

10. Cambiar el main para cargar el router outlet

Práctica 3: Bindings

¿Qué queremos?

- **Objetivo:** Crear enlaces de datos entre componentes
- **Ejemplo:** Crear un buscador y acerca de en el catálogo
- **Práctica:**

1. Extender la librería para crear un buscador
2. Extender el directorio para incluir introducción

¿Qué necesitamos?

- **No hay requerimientos especiales**

angular 1 - Pasos

- **Comentario**
  - = 2-Way binding
  - < 1-Way binding
  - @ 1-Way binding para texto
  - & Method binding

1. Crear el directorio empresa en el directorio sobre
2. Crear el componente para empresa y su vista, y solicitarlo en el módulo
3. Definir los bindings en el componente empresa para título y empresa
4. Definir en el módulo sobre un controlador que inicie esos valores

angular 1 - Pasos

5. Cambiar la vista de sobre para que la rodee controlador
6. Cambiar la vista de sobre para crear la empresa y pasarle los valores
7. Crear el directorio lista en el directorio catálogo
8. Crear el componente para lista y su vista, y solicitarla en el módulo
9. Definir los bindings en el componente para la función y la búsqueda
10. Definir en el módulo sobre un controlador que inicie esos valores
11. Cambiar la vista de catálogo para que la rodee el controlador
12. Cambiar la vista de catálogo para crear la lista y pasarle los valores

angular 4 - Pasos

- **Comentario**
  - **[()]** 2-Way binding -> Banana in a box
  - **[]** 1-Way binding
  - **()** Method binding

1. Crear el componente empresa

   ng g c routes/sobre/empresa

2. Crear la clase que compartirán padre e hijo

   ng g class routes/sobre/_data/empresa.model

3. Crear en el componente sobre las variables para título y empresa

angular 4 - Pasos

4. Cambiar la vista para pasar las variables con []
5. Recoger las variables en el componente empresa con @Input
6. Imprimir los datos en la vista
7. Crear el componente lista

   ng g c routes/catalogo/lista

8. Crear la clase que compartirán padre e hijo

   ng g class routes/catalogo/_data/busqueda.model

9. Crear en el componente catalogo la variables búsqueda y su callback

angular 4 - Pasos

10. Cambiar la vista para pasar la variable con [()] y el callback con ()
11. Recoger la variable y el callback en el componente lista
    - Variable con @Input
    - Callback con @Output y EventEmitter
12. Crear elementos auxiliares
    - Output que emita el cambio de búsqueda
    - Método que ejecute dicho output
    - Método que ejecute el callback
13. Crear la vista de lista recogiendo la variable en [ngModel] y emitiendo con
    - Variable con [(ngModel)]
    - Método de output de variable con (ngModelChange)
    - Método callback con (click)
14. Importar en el módulo el FormsModule

Práctica 4: Formulario

¿Qué queremos?

- **Objetivo:** Crear un formulario
- **Ejemplo:** Crear formulario de nuevo elemento
- **Práctica:**

1. Extender la librería para crear formulario de contacto
2. Extender el directorio para crear formulario de datos

¿Qué necesitamos?

- **No hay requerimientos especiales**

angular 1 - Pasos

1. Crear formulario en vista nuevo con input con nombre
   - Para título y autor añadir required
   - Para email email
   - Para año un validador de número positivo
2. Crear span en formulario que muestre errores si es inválido
3. Definir una directiva en el catálogo para la validación de número

angular 4 - Pasos

- **Comentario**: Reactive

1. Crear el módulo shared en core, que contendrá elementos comunes

   ng g m core/shared

2. Crear elementos útiles para formularios, como los validados

   ng g s core/shared/forms/form-tools —spec false

   ng g s core/shared/forms/validators —spec false

3. Crear método para control de errores en from-tools

4. Crear validador de números positivos en validators

5. Cambiar el módulo shared
   - Importar y exportar el ReactiveFormsModule
   - Proveer el form-tools y el validators

angular 4 - Pasos

6. Importar el módulo shared en el módulo catálogo
7. Crear el formulario en la vista de nuevo
   - Form con formGroup
   - Secciones nombradas
8. Crear el modelo de datos

   ng g class routes/catalogo/_data/elemento.model
9. Cambiar el componente para recoger y validar los datos
   - Crear el elemento
   - Crear el formGroup
   - Crear el control de elementos del formGroup

Práctica 5: Eventos

¿Qué queremos?

- **Objetivo:** Utilizar eventos de angular
- **Ejemplo:** Cambiar el formulario para añadir eventos
- **Práctica:**

1. Extender los formularios para añadir eventos

¿Qué necesitamos?

- **No hay requerimientos especiales**

angular 1 + angular 4 - Pasos

1. Cambios en el componente nuevo
   - Función de enviar formulario - ¿Cómo asociamos el elemento?
   - Función que muestre la ayuda
2. Cambios en la vista de nuevo
   - Asociar envió al formulario
   - Crear botón que muestre la ayuda en mouseover

Práctica 6: Aplicación

¿Qué queremos?

- **Objetivo:** Crear un ejemplo completo
- **Práctica:** Crear una aplicación con todo lo visto
- **Ideas**: Librería, taller, empleados, viajes, etc
- **Tiempo**: 1 semana

Semilla

¿Qué queremos?

- **Objetivo:** Crear una semilla de desarrollo
- **Ejemplo:** Semilla base

¿Qué necesitamos?

- **angular 1**
  - less + bootstrap + angular UI Bootstrap
  - webpack
  - webpack-dev-server o similar
  - angular
  - angular UI Router + OCLazyLoad o angular Route
  - lodash
- **angular 4**
  - webpack, angular y el router vienen instalados
  - No se recomienda usar less
  - bootstrap + ng-bootstrap

angular 1 - Pasos

1. Crear el directorio para nuestra aplicación
2. Entrar en el directorio
3. Ejecutar la inicialización del npm o yarn
4. Instalar las dependencias para desarrollador
   - webpack
   - webpack-dev-server o similar
   - css-loader, file-loader, imports-loader, less-loader, url-loader
5. Instalar las dependencias para ejecución
   - angular@1.6.4
   - less + bootstrap + angular-ui-bootstrap
   - angular-ui-router + oclazyload
   - lodash
6. Crear el directorio de carpetas
   - Crear directorio src para el source code
   - Crear directorio dist para html y ficheros de salida

angular 1 - Pasos

7. Crear el archivo webpack.config.js en el directorio
   - Definirá los archivos js iniciales -al menos dos, el módulo inicial y el vendor-
   - Definirá los ficheros de output
   - Cargará los módulos necesarios
8. Añadir en el package.json los scripts necesarios
   - El script de build llamará al webpack con su fichero config para generar los output
   - El script de start llamará al server de webpack sobre la carpeta de distribución
9. Modificaciones en el src
   - Crear el módulo inicial - Cargará el angular.module inicial, extensiones y template
   - Crear el fichero vendor.js, preferiblemente en core - Importará librerías necesarias
10. Modificaciones en el dist
   - Crear index.html, que contendrá la carga de los output y enlazará el html con el ng-app
11. Ejecutar el script de start

angular 4 - Pasos

1. Crear la aplicación
2. Instalar las dependencias para ejecución
   - bootstrap + ng-bootstrap (@ng-bootstrap/ng-bootstrap)
3. Importar los elementos necesarios
   - forRoot de ng-bootstrap en import del módulo principal
4. Modificar la vista inicial

Gracias por vuestra atención