

Introducción al Front mediante "Holas Mundos"



Juan Manuel Rodríguez Pérez

singular



@juan_manuel_rp



jmrp81



commit

Contenidos

1. Introducción	4
2. Descripción Hardware & Software.....	5
3. Pequeña Descripción de los Frameworks / Librerías	6
3.1 Angular	6
3.2 Aurelia	6
3.3 React	6
3.4 Vue.....	6
3.5 Ionic	6
3.6 NativeScript	7
3.7 ReactNative	7
4. Datos Comparativos	8
4.1 Encuesta Twitter	8
4.2 StackOverflow Etiquetas.....	8
4.3 Cursos Udemy.....	9
4.4 Github REPOSITARIOS.....	9
4.5 Github Commits	10
4.6 Github Issues.....	10
5. Recursos.....	11
5.1 Angular	11
5.2 Aurelia	11
5.3 React	11
5.4 Vue.....	11
5.5 Ionic	11
5.6 NativeScript	11
5.7 ReactNative	11
6. Instalación.....	12
6.1 General.....	12
6.1.1 Instalación de Visual Studio Code	12
6.1.2 Instalación de Node	12
6.1.3 Uso de NPM	12
6.1.4 Instalación de TypeScript.....	12
6.2 Angular (Material)	13
6.2.1 Elementos que instalar	13

6.2.2	Creando nuestra aplicación.....	13
6.2.3	Alguna configuración necesaria.....	14
6.2.4	Ejemplo de las principales clases.....	16
6.2.5	Arquitectura Creada.....	20
6.3	Aurelia	21
6.3.1	Instalación de Aurelia-Cli	21
6.3.2	Creación del Proyecto	21
6.3.3	Arquitectura creada	23
6.3.4	Ejecutar Aurelia	24
6.4	React	26
6.4.1	Instalar React	26
-	npm install -g create-react-app	26
6.4.2	Crear proyecto.....	26
6.4.3	Estructura del proyecto	27
6.4.4	Ejecución de la aplicación	28
6.5	Vue.....	29
6.5.1	Instalación de Vue	29
6.5.2	Instalación de vue-cli.....	29
6.5.3	Herramientas para el navegador	29
6.5.4	Creando nuestro proyecto.	29
6.5.5	Ya creado el proyecto	32
6.5.6	Ejemplo de archivos	33
6.5.7	Ejecutando la app	36
6.6	Ionic	37
6.6.1	Preparación Entornos desarrollo Android e IOS.....	37
6.6.2	Instalación de Ionic	37
6.6.3	Creando nuestra aplicación.....	37
6.6.4	Ejecutar la aplicación en navegador:	39
6.6.5	Añadir platform Android al proyecto:	40
6.6.6	Añadir platform iOS al proyecto:	41
6.6.7	Ejecutar la aplicación en emulador:.....	41
6.6.8	Ejecutar o generar la aplicación para producción:	42
6.6.9	Arquitectura Creada.....	42
6.7	NativeScript	43
6.7.1	Preparación Entornos desarrollo Android e IOS.....	43

6.7.2	Instalación de NativeScript	44
6.7.3	Creando nuestra Aplicación	45
6.7.4	Preparar las platform:	46
6.7.5	Ejecución de la aplicación:	47
6.7.6	Dispositivos móviles disponibles:	48
6.7.7	Algunos errores al ejecutar en emulador:.....	48
6.7.8	Arquitectura creada	48
6.8	ReactNative	49
6.8.1	Preparación Entornos desarrollo Android e los	49
6.8.2	Instalación de ReactNative	49
6.8.3	Creación de un proyecto.....	49
6.8.4	Ejecución del proyecto	50
6.8.5	Comandos Utiles.....	0
6.8.6	Arquitectura Creada.....	1
7.	Preparación Entorno de Desarrollo con Android	2
7.1.1	Introducción	2
7.1.2	Preparación de Android con Chocolatey	3
7.1.2.1	Para instalar Chocolatey pasos descritos en su web:	3
7.1.2.2	Preparación AVDs (Dispositivos virtuales):	5
7.1.3	Preparación de Android con Android Studio	7
7.1.4	Emulador de Windows	7
8.	Bibliografía	8
9.	Control de versiones	9

1. Introducción

Este documento es la versión extendida de la presentación usada para la charla en la CommitConf 2018 en Madrid:

“Introducción al Front mediante ‘Holas Mundos’”

Lo que se pretende con este documento como con la presentación en la charla, **NO** es hacer una comparativa sobre los diferentes frameworks y/o librerías existentes para saber cuál es mejor o cual usar.

La intención es acercar la situación actual del ecosistema “Front End” como de las diferentes herramientas existentes para afrontar las diferentes necesidades en el desarrollo o a la hora de afrontar nuevos proyectos.

La temática surge a raíz de tener un perfil Full Stack e intentar orientarme más hacía el lado Front End y de que [Azahara Fernández Guizán](#) estuviera haciendo pequeños proyectos con el Framework ‘Aurelia’.

Al investigar sobre los diferentes frameworks e intentar trabajar por primera vez con ellos a parte de la curva de aprendizaje de cada uno y hablando con gente que empieza en el sector me di cuenta de que a veces lo que más cuesta es hacer el primero y conseguir ejecutarlo.

Por ello es simplemente la realización de proyectos “Holas Mundos”, por lo que queda claro que tampoco se profundiza lo suficiente en cada framework y/o librería a la hora del desarrollo.

2. Descripción Hardware & Software

Hardware	Software
<ul style="list-style-type: none"> ❖ Marca y Modelo: MSI – GE72 2QL(Apache)-252XES ❖ Procesador: Intel Core i7-5700HQ 2.70 GHz ❖ RAM: 8 GB DDR3L SODIMM ❖ Controlador Gráfico: nVidia Geforce GTX950M, 2GB GDDR5 ❖ Pantalla: 17.3" LED Full HD (1920x1080) 16:9 Anti-Glare ❖ Disco Duro: Samsung 840 Evo 250 GB 	<ul style="list-style-type: none"> ❖ Windows 10 Pro ❖ Visual Studio Code versión 1.29.1 ❖ Node 8.12.0 LTS ❖ JDK 8 191 ❖ chocolatey v0.10.11

3. Pequeña Descripción de los Frameworks / Librerías

3.1 Angular

Angular es un Framework multiplataforma desarrollado en TypeScript para usar el patrón MVC o MVVM en aplicaciones web basadas en SPA (Single Page Application).

Tiene licencia MIT y es mantenido por Google su última versión estable es la 7 y está fechada del 11 de octubre de 2018.

3.2 Aurelia

Aurelia es un Framework escrito con EcmaScript para usar en aplicaciones web, móvil o de escritorio usando los patrones MVC o MVVM.

Esta bajo licencia MIT y el equipo de desarrollo lo lidera Rob Eisenberg.

3.3 React

React es una librería multiplataforma desarrollado en JavaScript, uno de sus objetivos es ayudar a crear páginas web SPA manejando únicamente la Interfaz de usuario. Es decir, React sería la View en el modelo MVC o MVVM.

Se puede usar junto React-based usado para las partes no-UI.

Es mantenido por Facebook bajo licencia MIT y su última versión estable es la 16.4.2 fechada del 1 de agosto de 2018.

3.4 Vue

Vue o Vue.js es un framework multiplataforma desarrollado en JavaScript para la construcción de interfaces de usuario y páginas web basada en SPA.

Su autor original fue Evan You, tiene licencia MIT y su última versión es la 2.5.17 con fecha del 1 de agosto del 2018.

3.5 Ionic

Ionic es un framework para desarrollar aplicaciones híbridas para Android e IOS utilizando tecnología web como HTML, CSS, SASS, TypeScript; en las últimas versiones está construido sobre Angular.

Está mantenido por la empresa Drift Co (creado por Max Lynch, Ben Sperry and Adam Bradley) bajo licencia MIT su última versión estable es la 3.9.2 con fecha 8 de noviembre de 2017.

3.6 NativeScript

NativeScript es un framework multiplataforma construido sobre JavaScript y TypeScript para construir aplicaciones para Android e IOS, soporta el uso de Angular y Vue.js.

Es mantenido por la empresa Telerik bajo una licencia Apache su última versión la 4.2.3 data del 28 de agosto de 2018.

3.7 ReactNative

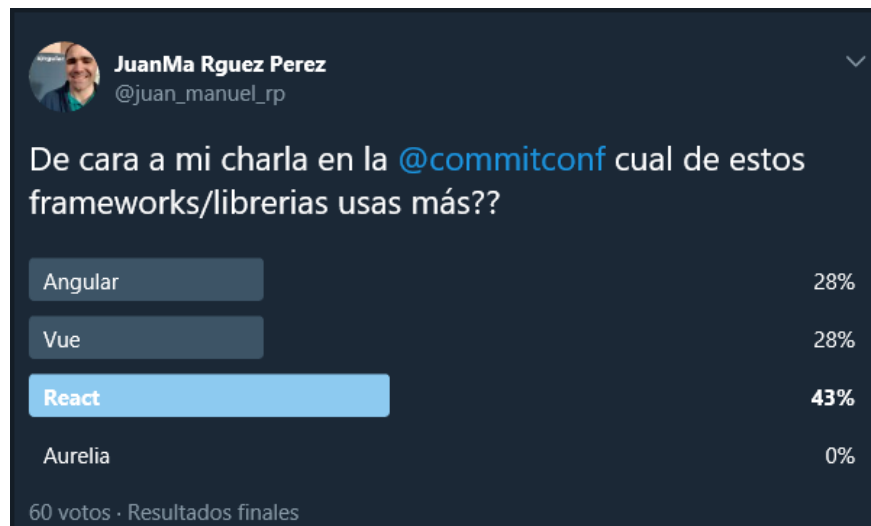
ReactNative es un framework desarrollado a partir de React para desarrollar aplicaciones con arquitectura nativa de Android, IOS y UWP usando JavaScript y React, aunque te permite el uso de código nativo.

4. Datos Comparativos

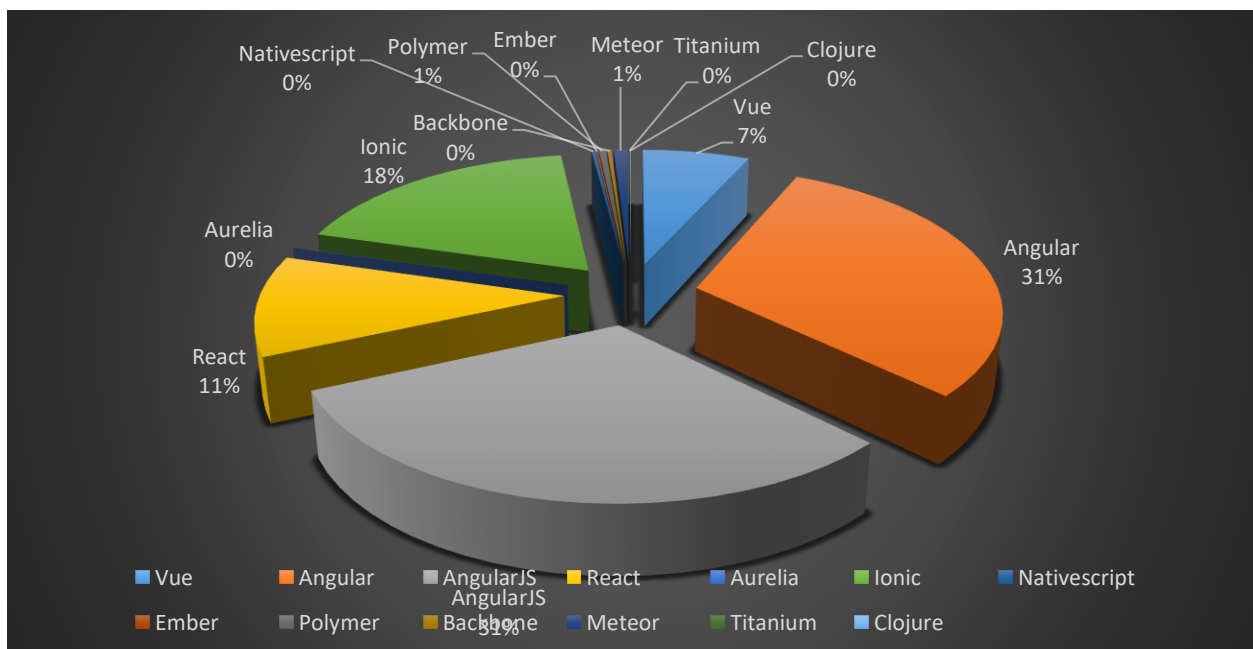
Como se comenta en la introducción **NO** se pretende hacer una comparación para elegir uno u otro framework sino para hacernos una idea de sus usos por diferentes personas.

Los datos son obtenidos del mes de octubre de 2018.

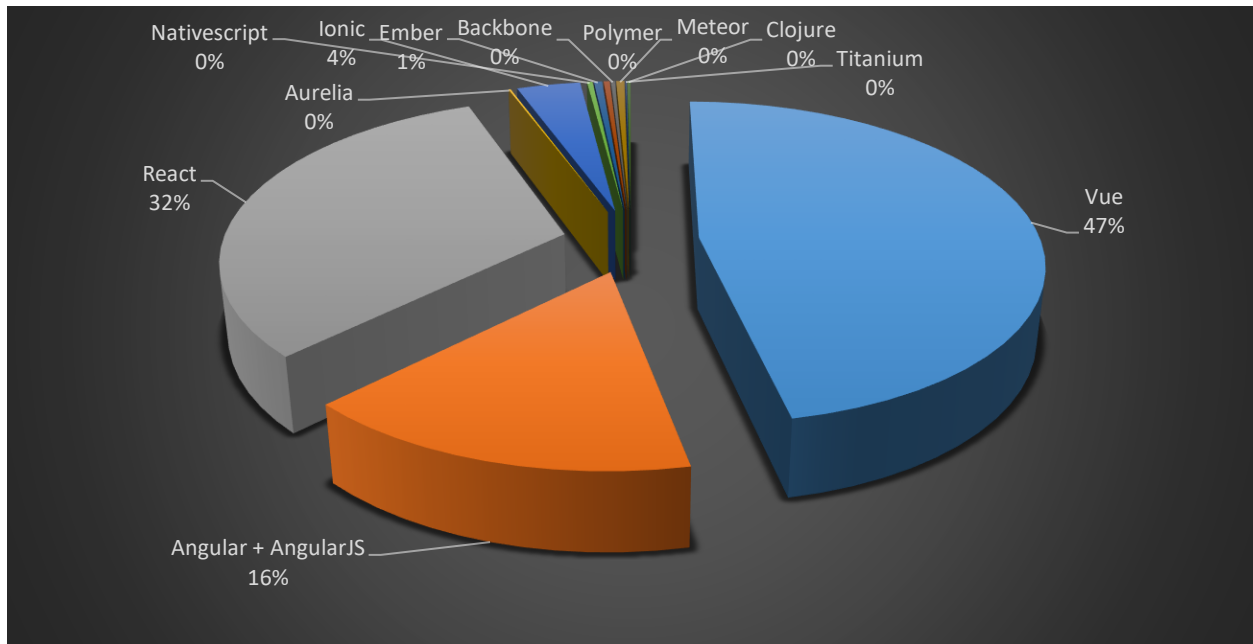
4.1 Encuesta Twitter



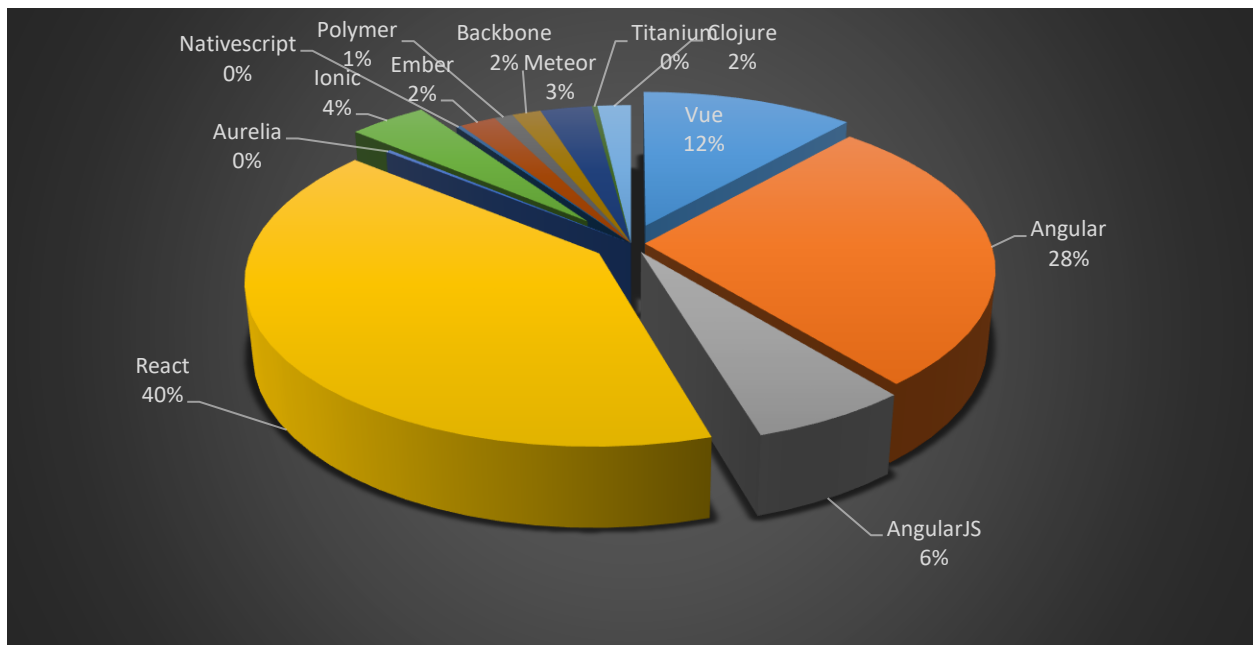
4.2 StackOverflow Etiquetas



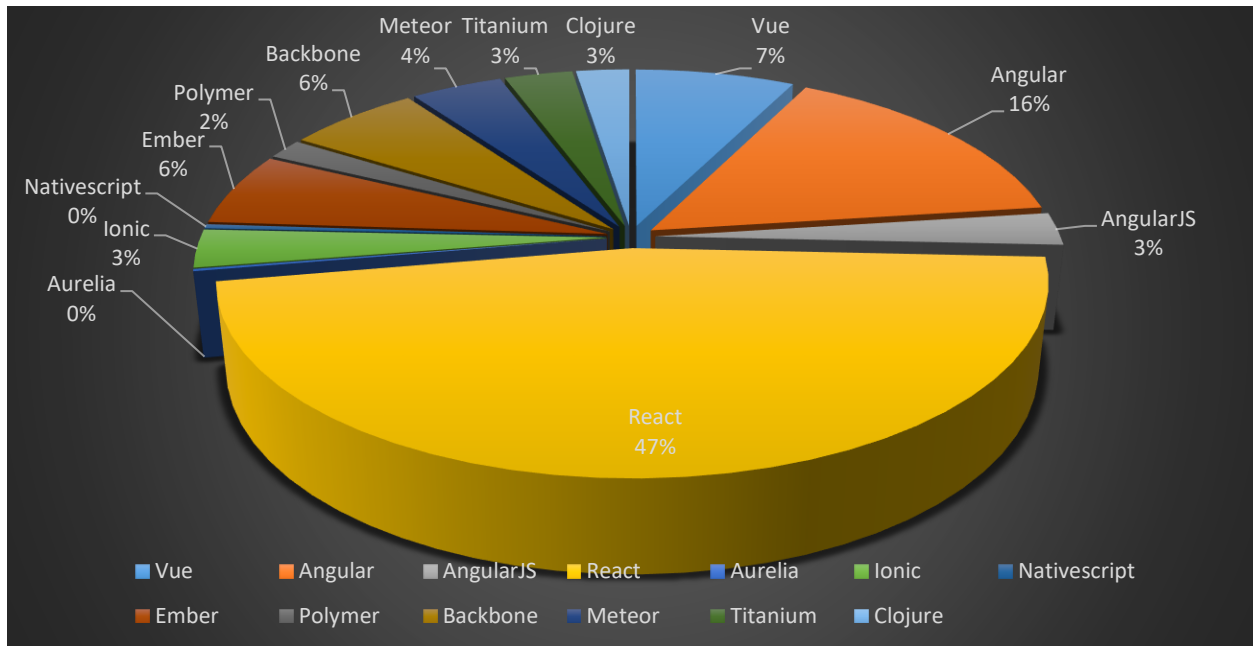
4.3 Cursos Udemy



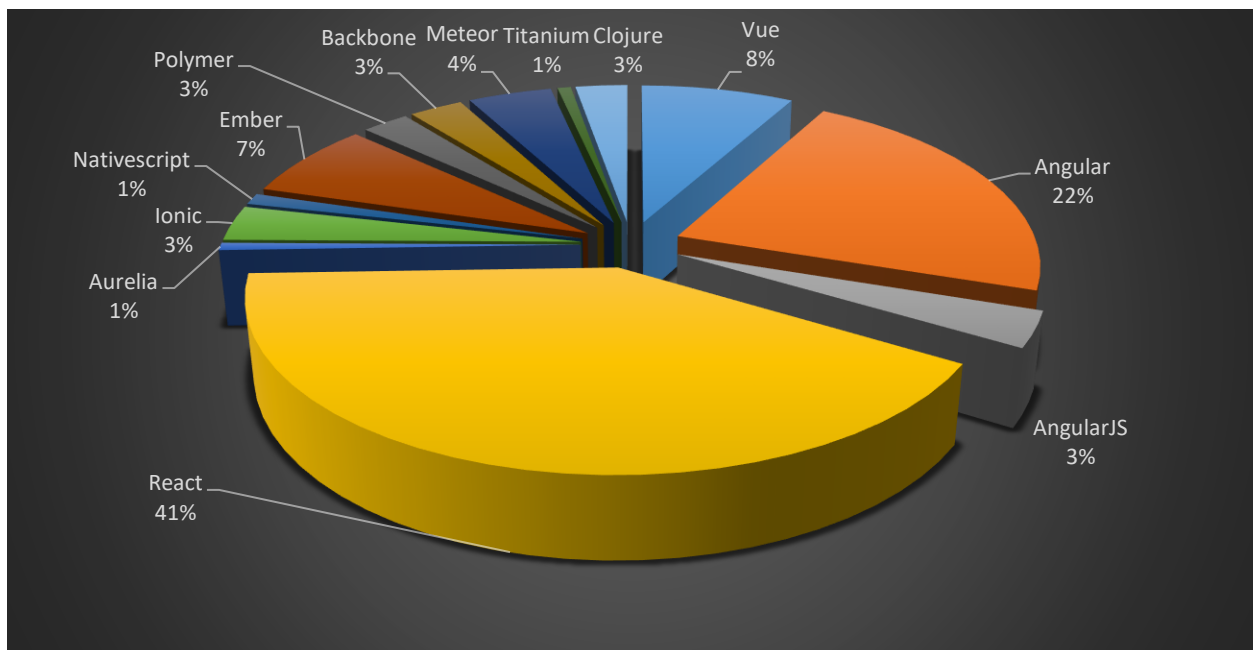
4.4 Github REPOSITORIOS



4.5 Github Commits



4.6 Github Issues



5. Recursos

5.1 Angular

- <https://angular.io/>
- <https://github.com/angular/angular>
- <https://material.angularjs.org/>
- A quien seguir en Twitter: **@carlosazaustre**

5.2 Aurelia

- <https://aurelia.io>
- <https://github.com/aurelia>
- <https://www.uno-de-piera.com/aurelia-un-gran-framework-javascript>
- A quien seguir en Twitter: **@azahara_fergui**

5.3 React

- <https://reactjs.org>
- <https://github.com/facebook/react>
- <https://www.udemy.com/aprendiendo-react>
- A quien seguir en Twitter: **@midudev**

5.4 Vue

- <https://github.com/jdonsan/estudiando-vue-js>
- <https://carlosazaustre.es/que-es-lo-que-me-gusta-de-vue-js/>
- <https://vuejs.org/>
- A quien seguir en Twitter: **@jdonsan**

5.5 Ionic

- <https://ionicframework.com/>
- <https://github.com/ionic-team/ionic>

5.6 NativeScript

- <https://docs.nativescript.org/tutorial/chapter-0>
- <https://www.progress.com/nativescript>
- <https://nativescript-vue.org/en/docs/introduction/>

5.7 ReactNative

- <https://facebook.github.io/react-native/>
- <http://www.reactnative.com/>

6. Instalación

En este punto por cada framework/librería explicare los pasos para hacer la instalación del entorno y crear en cada uno de ellos un 'Hola Mundo'.

6.1 General

En este apartado y sus subapartados describiré las instalaciones comunes a todos los frameworks/librerías.

6.1.1 Instalación de Visual Studio Code

Para el desarrollo he usado el editor de texto Visual Studio Code de Microsoft, lo podéis descargar desde el enlace:

- <https://code.visualstudio.com/download>

6.1.2 Instalación de Node

El primer paso de todo es instalar en nuestro equipo Node, para ello podéis acceder a su zona de descargas (<https://nodejs.org/es/download/>).

6.1.3 Uso de NPM

Una vez instalado Node podemos buscar su consola de comandos para seguir instalando diferentes componentes usando NPM (<https://www.npmjs.com/>).

6.1.4 Instalación de TypeScript

En todos los proyectos he usado el lenguaje TypeScript, que no deja de ser un superconjunto de JavaScript que básicamente añade tipado y objetos basados en clases. Su código será traducido a JavaScript a través de un compilador.

Para su instalación usando la línea de comandos de Node y NPM ejecutamos:

- `npm install -g typescript`

6.2 Angular (Material)

6.2.1 Elementos que instalar

Desde la consola de Node

- a. Angular: **npm install -g angular**

```
[.....] \ extract:angular: verb lock using C:\Users\jmp8\AppData\Roaming\npm-cache\_locks\staging-42e5b4b
```

- b. Angular/cli: **npm install -g @angular/cli**

```
[.....] \ fetchMetadata: sill resolveWithNewModule micromatch@3.1.10 checking installable status
```

- c. El uso de Bootstrap es opcional → **npm install bootstrap --save**

```
[.....] \ extract:bootstrap: verb lock using C:\Users\jmp8\AppData\Roaming\npm-cache\_locks\staging-bd94f
```

, en el documento style de nuestro proyecto ponemos:

```
@import url('../node_modules/bootstrap/dist/css/bootstrap.min.css');
```

6.2.2 Creando nuestra aplicación

Antes de empezar podemos ver más información sobre material en su web oficial (<https://material.angular.io/>) también tienen una guía de inicio (<https://material.angular.io/guide/getting-started>)

- a. Creamos proyecto de Angular desde la consola de comandos y situados en la carpeta raíz que queramos usar escribimos el comando:

```
ng new HelloW-AngularAndMaterial
```

Cuidado con el nombre, por ejemplo, el guion bajo no lo admite y te avisa después de haber realizado alguno de los siguientes pasos.

Nos preguntara si queremos añadir AngularRouting:

```
PS G:\CommitCom_2018> ng new HelloW-AngularAndMaterial
? Would you like to add Angular routing? (y/N) [ ]
```

Nos preguntara que tipo de formato de hojas de estilos quieres usar:

```
? Which stylesheet format would you like to use?
CSS
> SCSS [ http://sass-lang.com ]
SASS [ http://sass-lang.com ]
LESS [ http://lesscss.org ]
Stylus [ http://stylus-lang.com ]
```

```

CREATE Hello-AngularAndMaterial/src/app/app.component.spec.ts (1152 bytes)
CREATE Hello-AngularAndMaterial/src/app/app.component.ts (230 bytes)
CREATE Hello-AngularAndMaterial/src/app/app.component.scss (0 bytes)
CREATE Hello-AngularAndMaterial/e2e/protractor.conf.js (752 bytes)
CREATE Hello-AngularAndMaterial/e2e/tsconfig.e2e.json (213 bytes)
CREATE Hello-AngularAndMaterial/e2e/src/app.e2e-spec.ts (317 bytes)
CREATE Hello-AngularAndMaterial/e2e/src/app.po.ts (204 bytes)
npm WARN deprecated circular-json@0.5.9: CircularJSON is in maintenance only, flattened is its successor.
[ ] ..... ] - extract:path-dirname: sill extract map-age-cleaner@0.1.3

```

b. Instalamos Material en el proyecto anteriormente creado usando el siguiente comando en la terminal de comandos (situarse en la carpeta del proyecto antes):

npm install --save @angular/material @angular/cdk

```

[ ] ..... ] / loadExtraneous: sill resolveWithNewModule parse5@5.1.0 checking installable status

```

c. Además de lo anterior si quisiéramos tener animaciones y algunos componentes como (mat-slide-toggle, mat-slider, matTooltip) de HammerJS deberíamos instalarlos usando para animaciones el comando:

npm install --save @angular/animations

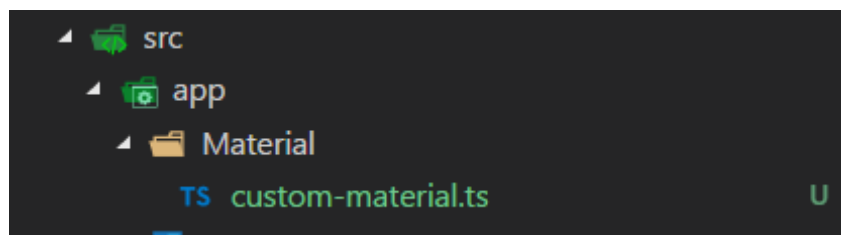
Y para HammerJS el comando:

npm install -g --save hammerjs

(Debemos tener en cuenta no confundir los comandos npm y ng, mientras uno es de angular (ng) el otro es de NPM por lo cual las 'g' tienen diferente significado. Mientras en el entorno NPM significa global, por lo que si la indicamos estaremos haciendo la instalación global para todos los proyectos que realicemos, en Angular es la abreviatura de generate.)

6.2.3 Alguna configuración necesaria

Para el uso de Material recomiendo crear en nuestro proyecto un módulo que en mi caso llamo CustomMaterial donde tendremos todo el contenido a usar de material.



○ Importamos dicho modulo en el root **app.module.ts**:

```
import { CustomMaterial } from '../Material/custom-material';
```

```
import { BrowserAnimationsModule } from '@angular/platform-  
browser/animations';
```

- Ejemplo de la anterior clase custom citada para albergar material:

```
import { NgModule } from '@angular/core';  
  
// imports material  
import {  
  MatToolbarModule  
} from '@angular/material';  
  
@NgModule({  
  imports: [  
    MatToolbarModule  
  ],  
  exports: [  
    MatToolbarModule  
  ]  
})  
export class CustomMaterial {}
```

En ella iremos importando los módulos de Material a usar y añadiéndolos a import y export del módulo.

- Importación de un estilo predeterminado de Material, para ello usaremos la hoja general de estilos style.css del proyecto:

```
@import '~@angular/material/prebuilt-themes/purple-green.css';
```


6.2.4 Ejemplo de las principales clases.

a. App.module.ts

```
import { BrowserModule } from '@angular/platform-browser';
import { NgModule } from '@angular/core';
import { HttpClientModule } from '@angular/common/http';
import { CustomMaterial } from './Material/custom-material';
import { BrowserAnimationsModule } from '@angular/platform-
browser/animations';

import { AppComponent } from './app.component';

@NgModule({
  declarations: [
    AppComponent
  ],
  imports: [
    BrowserModule,
    HttpClientModule,
    CustomMaterial,
    BrowserAnimationsModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

b. App.component.ts

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  templateUrl: './app.component.html',
  styleUrls: ['./app.component.css']
})
export class AppComponent {
  title = 'Angular Material';
}
```

c. App.component.html

```

<section>
  <header>
    <mat-toolbar color="primary">
      <mat-toolbar-row>
        <mat-icon svgIcon="helloIcon"></mat-icon>
        <span>Welcome to {{ title }}!</span>
      </mat-toolbar-row>
    </mat-toolbar>
  </header>

  <article>
    <mat-grid-list cols="2" color="primary">
      <mat-grid-tile>
        <mat-list role="list">
          <mat-list-item role="listitem">
             Some links to help you:
            </mat-list-item>
            <mat-divider></mat-divider>
            <mat-list-item role="listitem">
              <a class="urlToOtherWeb" target="_blank" rel="noopener"
href="https://angular.io/tutorial">
                <button mat-raised-button>
                  <mat-icon svgIcon="earthIcon"></mat-icon>
                  Tour of Heroes
                </button>
              </a>
            </mat-list-item>
            <mat-list-item role="listitem">
              <a class="urlToOtherWeb" target="_blank" rel="noopener"
href="https://github.com/angular/angular-cli/wiki">
                <button mat-raised-button>
                  <mat-icon svgIcon="earthIcon"></mat-icon>
                  CLI Documentation
                </button>
              </a>
            </mat-list-item>
            <mat-list-item role="listitem">
              <a class="urlToOtherWeb" target="_blank" rel="noopener"
href="https://blog.angular.io/">

```

```

        <button mat-raised-button>
          <mat-icon svgIcon="earthIcon"></mat-icon>
          Angular blog
        </button>
      </a>
    </mat-list-item>
  </mat-list>
</mat-grid-tile>
<mat-grid-tile>
  <mat-list role="list">
    <mat-list-item role="listitem">
      <h2>Material samples</h2>
    </mat-list-item>
    <mat-divider></mat-divider>
    <mat-list-item role="listitem">
      <a class="urlToOtherWeb" target="_blank" rel="noopener"
href="https://material.angular.io/">
        <button mat-raised-button>
          <mat-icon svgIcon="earthIcon"></mat-icon>
          Angular Material
        </button>
      </a>
    </mat-list-item>
    <mat-list-item role="listitem">
      <a class="urlToOtherWeb" target="_blank" rel="noopener"
href="https://material.angular.io/components/categories">
        <button mat-raised-button>
          <mat-icon svgIcon="earthIcon"></mat-icon>
          Components
        </button>
      </a>
    </mat-list-item>
    <mat-list-item role="listitem">
      <a class="urlToOtherWeb" target="_blank" rel="noopener"
href="https://github.com/angular/material2">
        <button mat-raised-button>
          <mat-icon svgIcon="earthIcon"></mat-icon>
          Github
        </button>
      </a>
    </mat-list-item>
  </mat-list>
</mat-grid-tile>
</mat-grid-list>
</article>
</section>

```

d. Custom-material.ts

```
import { NgModule } from '@angular/core';
import { DomSanitizer } from '@angular/platform-browser';

// imports material
import {
  MatToolbarModule, MatSidenavModule, MatGridListModule, MatIconRegistry
} from '@angular/material';
import { MatIconModule } from '@angular/material/icon';
import { MatDividerModule } from '@angular/material/divider';
import { MatListModule } from '@angular/material/list';

@NgModule({
  imports: [
    MatToolbarModule,
    MatIconModule,
    MatSidenavModule,
    MatGridListModule,
    MatDividerModule,
    MatListModule
  ],
  exports: [
    MatToolbarModule,
    MatIconModule,
    MatSidenavModule,
    MatGridListModule,
    MatDividerModule,
    MatListModule
  ]
})
export class CustomMaterial {
  constructor(iconRegistry: MatIconRegistry, sanitizer: DomSanitizer) {
    iconRegistry.addSvgIcon(
      'earthIcon',

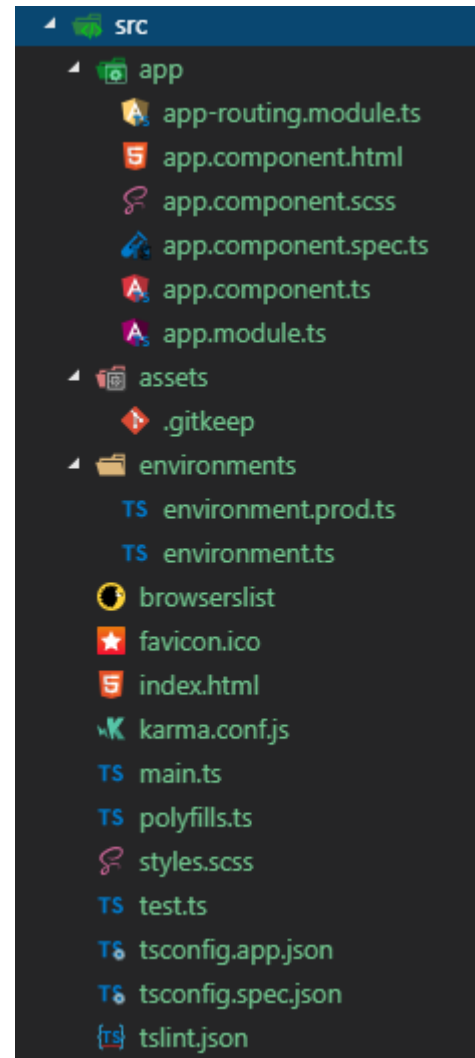
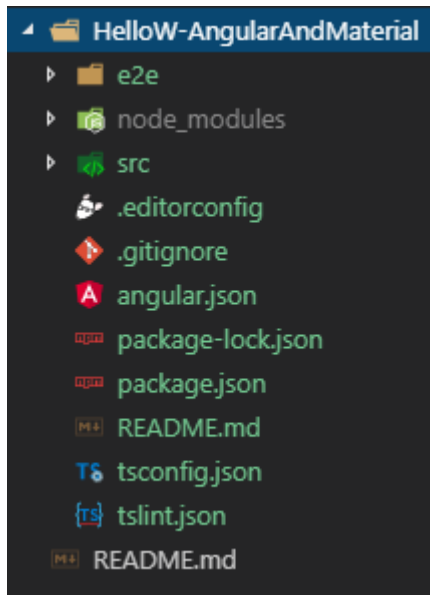
sanitizer.bypassSecurityTrustResourceUrl('../assets/images/earth.svg'));
    iconRegistry.addSvgIcon(
      'helloIcon',

sanitizer.bypassSecurityTrustResourceUrl('../assets/images/hello.svg'));
  }
}
```

e. Styles.css

```
@import '~@angular/material/prebuilt-themes/purple-green.css';
```

6.2.5 Arquitectura Creada



6.3 Aurelia

6.3.1 Instalación de Aurelia-Cli

Instalamos Aurelia-Cli desde línea de comandos usando el comando:

- `npm install -g aurelia-cli`

```
PS G:\CommitConf_2018> npm install -g aurelia-cli  
[.....] \ fetchMetadata: sill resolveWithNewModule stream-exhaust@1.0.2 checking installable status
```

6.3.2 Creación del Proyecto

Para crear el proyecto desde línea de comandos usamos el comando:

- `au new`

```
PS G:\CommitConf_2018> au new  
No Aurelia project found.  
  
  _____  _____  
 /  _  _  _  /  _  _  _  
/  _  _  _  /  _  _  _  
 \  _  _  _  \  _  _  _  
  \  _  _  _  \  _  _  _  
  
Please enter a name for your new project below.  
  
[aurelia-app]> HelloW_Aurelia
```

Esto nos mostrara un asistente de instalación donde mediante preguntas nos dejara elegir ciertas características.

```
Would you like to use the default setup or customize your choices?  
  
1. Default ESNext (Default)  
   A basic web-oriented setup with Babel and Webpack for modern JavaScript development.  
2. Default TypeScript  
   A basic web-oriented setup with TypeScript and Webpack for modern JavaScript development.  
3. Custom  
   Select bundlers (built-in or webpack), loaders (requirejs or systemjs), transpilers, CSS pre-processors  
   and more.  
  
[Default ESNext]>
```

```
[Default ESNEXT]> Default TypeScript
```

Project Configuration

```
Name: HelloW_Aurelia
Platform: Web
Bundler: Webpack
Loader: None
Transpiler: TypeScript
Markup Processor: Minimal Minification
CSS Processor: None
Unit Test Runner: Jest
Unit Test Runner: Karma
Integration Test Runner: None
Editor: Visual Studio Code
```

Would you like to create this project?

1. Yes (Default)
Creates the project structure based on your selections.
2. Restart
Restarts the wizard, allowing you to make different selections.
3. Abort
Aborts the new project wizard.

```
[Yes]> 
```

Would you like to install the project dependencies?

1. Yes (Default)
Installs all server, client and tooling dependencies needed to build the project.
2. No
Completes the new project wizard without installing dependencies.

```
[Yes]> 
```

Installing project dependencies.
added 1452 packages from 1284 contributors and audited 63365 packages in 160.917s

found 0 vulnerabilities

Congratulations

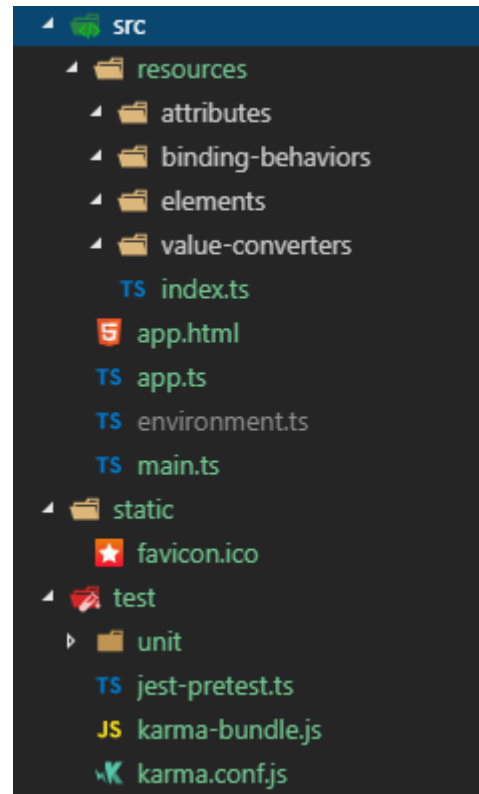
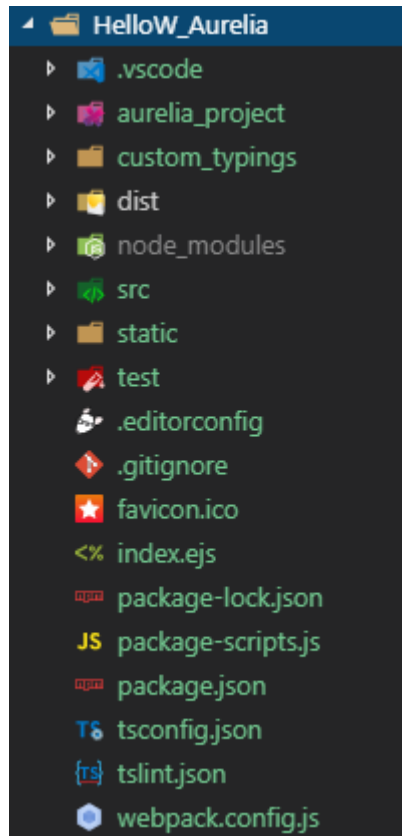
Congratulations! Your Project "HelloW_Aurelia" Has Been Created!

Getting started

Now it's time for you to get started. It's easy. First, change directory into your new project's folder. You can use `cd HelloW_Aurelia` to get there. Once in your project folder, simply run your new app with `au run`. Your app will run fully bundled. If you would like to have it auto-refresh whenever you make changes to your HTML, JavaScript or CSS, simply use the `--watch` flag. If you want to build your app for production, run `au build --env prod`. That's just about all there is to it. If you need help, simply run `au help`.

Happy Coding!

6.3.3 Arquitectura creada



6.3.4 Ejecutar Aurelia

Usamos el comando:

- **npm start**

```

    vendor.bluebird.6e35bcd0404fd04b567.chunk.js 102 KiB
    vendor.bluebird [emitted]
    Entrypoint app [big] = vendor.bluebird.6e35bcd0404fd04b567.chunk.js vendor.bluebird.6e35bcd0404fd04b567.bundle.map vendor.aurelia-binding.6e35bcd0404fd04b567.chunk.js vendor.aurelia-binding.6e35bcd0404fd04b567.bundle.map vendor.aurelia-templating.6e35bcd0404fd04b567.chunk.js vendor.aurelia-templating.6e35bcd0404fd04b567.bundle.map vendor.aurelia.6e35bcd0404fd04b567.chunk.js vendor.aurelia.6e35bcd0404fd04b567.bundle.map vendor.6e35bcd0404fd04b567.chunk.js vendor.6e35bcd0404fd04b567.bundle.map app.6e35bcd0404fd04b567.bundle.js app.6e35bcd0404fd04b567.bundle.map
    [0] multi aurelia-webpack-plugin/runtime/empty-entry aurelia-webpack-plugin/runtime/pal-loader-entry (webpack)-dev-server/client?http://localhost:8081 aurelia-bootstrapper 64 bytes {app} [built]
    [./node_modules/aurelia-bootstrapper/dist/native-modules/aurelia-bootstrapper.js] 5.17 KiB {vendor.aurelia} [built]
    [./node_modules/aurelia-loader-webpack/dist/native-modules/aurelia-loader-webpack.js] 15.2 KiB {vendor.aurelia} [built]
    [./node_modules/aurelia-pal/dist/native-modules/aurelia-pal.js] 2.18 KiB {vendor.aurelia} [built]
    [./node_modules/aurelia-polyfills/dist/native-modules/aurelia-polyfills.js] 24.4 KiB {vendor.aurelia} [built]
    [./node_modules/aurelia-webpack-plugin/runtime/empty-entry.js] 585 bytes {vendor.aurelia} [built]
    [./node_modules/aurelia-webpack-plugin/runtime/pal-loader-entry.js] 1.56 KiB {vendor.aurelia} [built]
    [./node_modules/bluebird/js/browser/bluebird.js-exposed] 65 bytes {vendor.bluebird} [built]
    [./node_modules/loglevel/lib/loglevel.js] 7.68 KiB {vendor} [built]
    [./node_modules/process/browser.js] 5.29 KiB {vendor} [built]
    [./node_modules/strip-ansi/index.js] 161 bytes {vendor} [built]
    [./node_modules/url/url.js] 22.8 KiB {vendor} [built]
    [./node_modules/webpack-dev-server/client/index.js?http://localhost:8081] (webpack)-dev-server/client?http://localhost:8081 7.78 KiB {vendor} [built]
    [./node_modules/webpack-dev-server/client/overlay.js] (webpack)-dev-server/client/overlay.js 3.58 KiB {vendor} [built]
    [./node_modules/webpack-dev-server/client/socket.js] (webpack)-dev-server/client/socket.js 1.05 KiB {vendor} [built]
    + 87 hidden modules
  Child html-webpack-plugin for "index.html":
    1 asset
    Entrypoint undefined = index.html
    [./node_modules/html-webpack-plugin/lib/loader.js!./index.ejs] 890 bytes {0} [built]
    [./node_modules/lodash/lodash.js] 527 KiB {0} [built]
    [./node_modules/webpack/buildin/global.js] (webpack)/buildin/global.js 489 bytes {0} [built]
    [./node_modules/webpack/buildin/module.js] (webpack)/buildin/module.js 497 bytes {0} [built]
  i [wdm]: Compiled successfully.

```

Abrimos el navegador en : <http://localhost:8081/>

A web browser address bar showing navigation icons (back, forward, refresh, home) and the address "localhost:8081/".

Hello World!

6.4 React

6.4.1 Instalar React

Instalar React :

- `npm install -g create-react-app`

6.4.2 Crear proyecto

Para crear el proyecto usamos el comando desde la consola de comandos:

- `create-react-app [nombreApp]`

```
PS G:\CommitConf_2018> create-react-app HelloW_React
Could not create a project called "HelloW_React" because of npm naming restrictions:
* name can no longer contain capital letters
```

No se pueden usar letras mayúsculas.

```
PS G:\CommitConf_2018> create-react-app hellow_react

Creating a new React app in G:\CommitConf_2018\hellow_react.

Installing packages. This might take a couple of minutes.
Installing react, react-dom, and react-scripts...

[ ] ..... | extract:babel-helpers: sill tarball trying serve-index@1.7.2 by hash: sha1-03aNabHn2C5c4
```

```
Success! Created hellow_react at G:\CommitConf_2018\hellow_react
Inside that directory, you can run several commands:
```

```
npm start
```

```
Starts the development server.
```

```
npm run build
```

```
Bundles the app into static files for production.
```

```
npm test
```

```
Starts the test runner.
```

```
npm run eject
```

```
Removes this tool and copies build dependencies, configuration files
and scripts into the app directory. If you do this, you can't go back!
```

```
We suggest that you begin by typing:
```

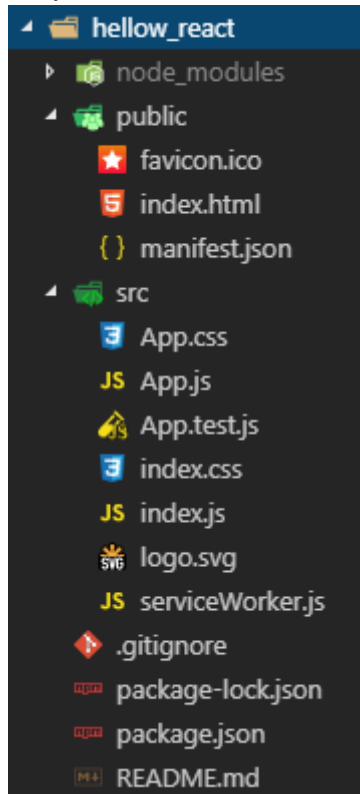
```
cd hellow_react
```

```
npm start
```

```
Happy hacking!
```

6.4.3 Estructura del proyecto

Al crear el proyecto podemos ver como se nos genera la estructura de la izquierda.



node_modules: Es la carpeta donde se genera las diferentes librerías de node y react necesarias para el desarrollo del proyecto.

public: En esta vemos como se generan los archivos de carácter público, como puede ser el **favicon** a usar por la aplicación como el archivo html **index.html** que será el usado por los navegadores para mostrar nuestra página web desarrollada.

El archivo **manifest.json** será un archivo json donde se especificará diferente información sobre la aplicación por ejemplo el nombre, nombre corto, iconos a usar, página de inicio, color de tema, color del fondo, etc.

src: En esta carpeta será donde realizaremos gran parte del desarrollo.

De inicio podemos ver un bloque formado por diferentes archivos con la palabra **App**, donde **App.js** sería el archivo JavaScript que contendrá nuestro componente principal, **App.css** es la hoja de estilos que contendrá los estilos generales orientados a los componentes y **App.test.js** será el fichero JavaScript de test.

Podemos observar el archivo **index.js** que será un JavaScript que contendrá la información para renderizar nuestra aplicación, desde él se llamará al componente principal; en este caso existente en **App.js** para cargarlo en el fichero **index.html** en el tag:

```
<div id="root"></div>
```

También tenemos el **index.css** hoja de estilos que contendrá los estilos generales a nivel de la aplicación general, en mi opinión mientras en **App.css** podemos tener los estilos generales para los componentes en este se podrían tener los relativos a los elementos generales de HTML como el body, footer, header, etc.

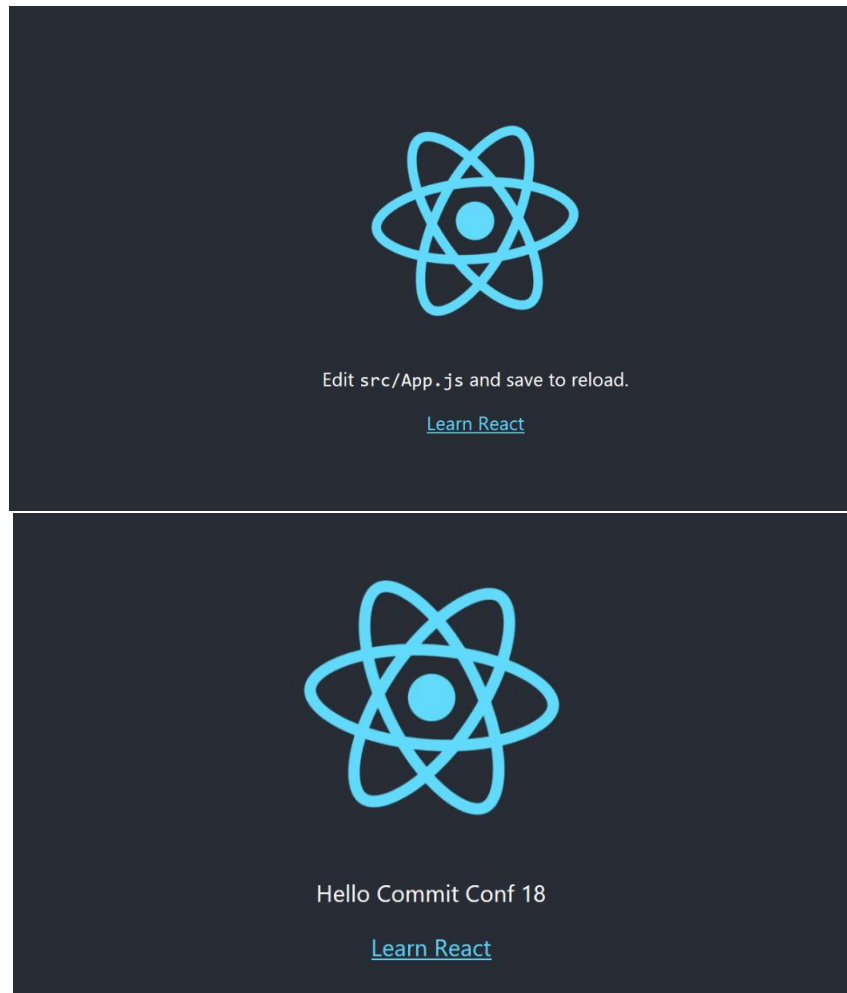
Una buena recomendación es generar una buena estructura de carpetas dentro de **src** para tener más ordenado y limpio nuestro proyecto.

6.4.4 Ejecución de la aplicación

Una buena recomendación es generar una buena estructura de carpetas dentro de src para tener más ordenado y limpio nuestro proyecto.

Para ejecutar la aplicación podemos usar el comando:

npm start



Para facilitar el teste podéis usar la extensión para los navegadores como Chrome, firefox:

- **react developer tools**

Otros comandos:

Construir la aplicación: `npm run build`

Iniciar Test: `npm test`

6.5 Vue

6.5.1 Instalación de Vue

Desde la línea de comandos de Node instalamos Vue mediante el comando:

- **npm -g install vue**

6.5.2 Instalación de vue-cli

Desde la línea de comandos de Node instalamos vue-cli mediante el comando:

- **npm -g install vue-cli**

```
npm WARN deprecated coffee-script@1.12.7: CoffeeScript on NPM has moved to "coffeescript" (no hyphen)
[.....] / loadDep:isarray: still resolveWithNewModule buffer@3.6.0 checking installable status
```

6.5.3 Herramientas para el navegador

Una buena recomendación es usar la extensión vue-devtools en tu navegador.

6.5.4 Creando nuestro proyecto.

Para crear nuestra aplicación una vez lo tengamos todo instalado es muy sencillo sirve con ejecutar el siguiente comando:

- **vue init webpack HelloWorld_Vue**

Una vez ejecutado comenzara la instalación, con diferencia con otros frameworks como puede ser Angular durante la instalación te preguntara ciertas opciones de instalación.

Nombre del proyecto:

```
? Project name (HelloW_Vue) 
```

Si no introducimos un nombre por defecto dejara el marcado entre paréntesis que será el que coincida con el indicado en el comando usado para crear la app con vue-cli.

Por ejemplo, en mi caso, debo introducir otro nombre al usar mayúsculas, como bien podemos observar en la notificación que obtenemos por consola.

```
? Project name (HelloW_Vue) 
>> Sorry, name can no longer contain capital letters.
```

Descripción del proyecto.

Si en el paso anterior va todo correcto nos pedirá que le demos una descripción al proyecto.

```
? Project name hellow_vue
? Project description Proyecto Hola Mundo CommitConf 2018
```

Indicar el nombre del autor:

```
? Project description
? Author Juanma
```

Opción de compilación:

En este punto nos dará a elegir entre dos opciones a elegir con las flechas del teclado.

```
? Vue build (Use arrow keys)
> Runtime + Compiler: recommended for most users
  Runtime-only: about 6KB lighter min+gzip, but templates (or any Vue-specific HTML) are ONLY allowed in .vue files - render functions are required elsewhere
```

Yo he elegido la que recomienda para la mayoría de los usuarios.

Opción de instalar router:

En esta opción le podemos indicar que 'si (y)' queremos que nos instale el gestor de routing de vue.

```
? Install vue-router? (Y/n) y
```

Opción instalar ESLint:

ESLint, para quien no lo sepa es un helper de EcmaScript para ayudar que nuestro código sea mucho más limpio ayudándonos con notificaciones mientras desarrollamos.

```
? Use ESLint to lint your code? (Y/n) y
```

Elección del tipo de ESLint a usar:

En esta pregunta podemos elegir entre varias opciones de configuración del ESLint.

```
? Pick an ESLint preset (Use arrow keys)
> Standard (https://github.com/standard/standard)
  Airbnb (https://github.com/airbnb/javascript)
  none (configure it yourself)
```

Preparación de Unit Test

Nos da la opción de preparar la app para el uso de Unit Test

```
? Set up unit tests (Y/n) y
```

Selección de herramienta de ejecución de los test.

```
? Pick a test runner (Use arrow keys)
> Jest
  Karma and Mocha
  none (configure it yourself)
```

Preparar test e2e

```
? Setup e2e tests with Nightwatch? (Y/n) y
```

Opción de ejecutar automáticamente npm install

En este paso nos hace una pregunta interesante, ya que nos pregunta si después de la instalación queremos que nos ejecute automáticamente el comando npm install de node por el cual instalara todas las librerías necesarias teniendo en cuenta también las opciones elegidas anteriormente.

```
? Should we run `npm install` for you after the project has been created? (recommended) (Use arrow keys)
> Yes, use NPM
  Yes, use Yarn
  No, I will handle that myself
```

Resumen opciones elegidas:

```
? Project name hellow_vue
? Project description Proyecto Hola Mundo CommitConf 2018
? Author Juanma
? Vue build standalone
? Install vue-router? Yes
? Use ESLint to lint your code? Yes
? Pick an ESLint preset Standard
? Set up unit tests Yes
? Pick a test runner karma
? Setup e2e tests with Nightwatch? Yes
? Should we run `npm install` for you after the project has been created? (recommended) (Use arrow keys)
> Yes, use NPM
  Yes, use Yarn
  No, I will handle that myself
```

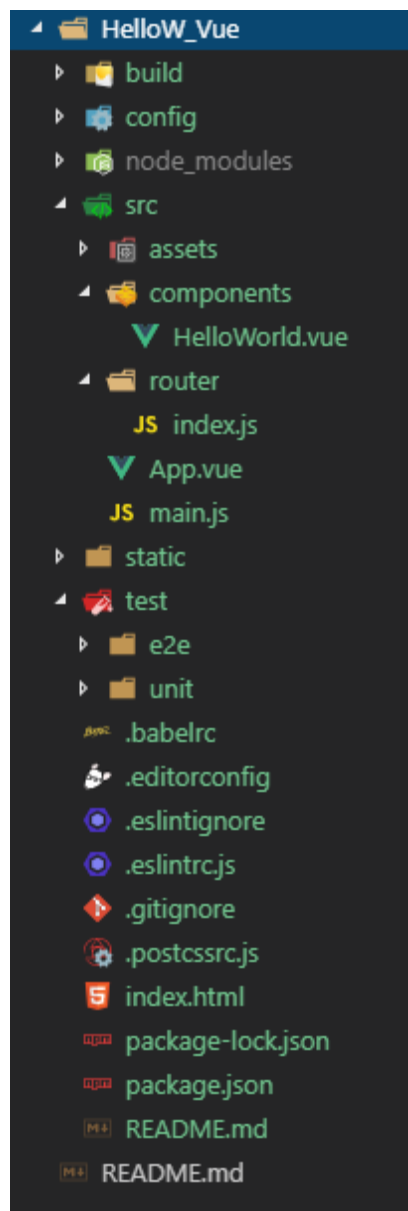

Una vez esto iniciara la creación del proyecto:

```
vue-cli · Generated "Hello-World-Vue".

# Installing project dependencies ...
# =====
[.....] | fetchMetadata: sill pacote range manifest for eslint-friendly-formatter@^3.0.0 fetched in 84ms
```

6.5.5 Ya creado el proyecto

Una vez finalizado los pasos anteriores, vue cli nos habrá creado el proyecto con la siguiente estructura (puede variar según las opciones seleccionadas).



En la carpeta src será donde vayamos realizando nuestro desarrollo de la aplicación, en ella podemos destacar las siguientes carpetas y archivos.

Assets: Será la carpeta para alojar recursos como imágenes, etc.

Components: Carpeta para ir albergando los diferentes componentes vue desarrollados.

Router: Tendrá el archivo js con el enrutamiento que queramos para nuestra app.

App.vue: Será el componente principal de la aplicación y el padre del resto de componentes será el que renderice vue y será el llamado desde el index.html

Main.js: Archivo js principal de configuración de la aplicación donde por ejemplo declararemos los componentes que vayamos creando entre otros.

Index.html: Archivo index de la aplicación punto de entrada de la aplicación y archivo principal usado por los navegadores.

6.5.6 Ejemplo de archivos

Index.html:

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8">
5      <meta name="viewport" content="width=device-width,initial-scale=1.0">
6      <title>hellow_vue</title>
7    </head>
8    <body>
9      <div id="app"></div>
10     <!-- built files will be auto injected -->
11   </body>
12 </html>
13
```

Es un archivo html común, donde nos debemos de fijar en el div con el identificador id="app" que será usado por el archivo main.js para montar el componente principal.

App.vue:

Componente principal de la app, es un archivo vue, donde podemos observar y ver que llama la atención de como Vue nos aconseja que queda componente este en un único fichero conteniendo tanto el código html, javascript y css.

El código html como veis se encierra entre las etiquetas de <template>

```
<template>
  <div id="app">
    
    <router-view/>
  </div>
</template>

<script>
export default {
  name: 'App'
}
</script>

<style>
#app {
  font-family: 'Avenir', Helvetica, Arial, sans-serif;
  -webkit-font-smoothing: antialiased;
  -moz-osx-font-smoothing: grayscale;
  text-align: center;
  color: #2c3e50;
  margin-top: 60px;
}
</style>
```

Al ser el componente principal tendrá la etiqueta:

<router-view/>

Donde se irán cargando el resto de los componentes según naveguemos por la app.

Componente secundario, **HelloWorld.vue**:

Este componente será el que contenga el contenido, en este caso el saludo típico de Hola Mundo.

Archivo **index.js** del router:

Este archivo contendrá nuestras rutas de la aplicación y en nuestro caso será el que indique que inicialmente dentro de la etiqueta router-view que está en el App.vue se cargue el anterior componente al indicarlo en la especificación como lo vemos en la siguiente imagen.

```
✓ HelloWorld.vue x
1  <template>
2    <div class="hello">
3      <h1>{{ msg }}</h1>
4      <h2>Essential Links</h2>
5      <ul>...
47    </ul>
48    <h2>Ecosystem</h2>
49    <ul>...
82    </ul>
83  </div>
84 </template>
85
86 <script>
87 export default {
88   name: 'HelloWorld',
89   data () {
90     return {
91       msg: 'Hello World Commit Conf 18'
92     }
93   }
94 }
95 </script>
96
97 <!-- Add "scoped" attribute to limit CSS to this component only -->
98 <style scoped>
99   h1, h2 {
100     font-weight: normal;
101   }
102   ul {
103     list-style-type: none;
104     padding: 0;
105   }
106   li {
107     display: inline-block;
108     margin: 0 10px;
109   }
110   a {
111     color: #42b983;
112   }
113 </style>
114
```

6.5.7 Ejecutando la app

Para ejecutar la aplicación desde la misma carpeta usamos el comando:

- **npm run dev**

```
> hello-world-vue@1.0.0 dev [redacted] Hello-World-Vue
> webpack-dev-server --inline --progress --config build/webpack.dev.conf.js

95% emitting

DONE Compiled successfully in 4405ms

I Your application is running here: http://localhost:8080
```

Abrimos el navegador y escribimos la url que nos indica en la consola de comandos.



Hello World Commit Conf 18

Essential Links

[Core Docs](#) [Forum](#) [Community Chat](#) [Twitter](#)
[Docs for This Template](#)

Ecosystem

[vue-router](#) [vuex](#) [vue-loader](#) [awesome-vue](#)

Y podemos observar cómo nos carga nuestra app.

6.6 Ionic

6.6.1 Preparación Entornos desarrollo Android e IOS

Posiblemente este punto puede ser el que dé más “problemas” si además nunca los has preparado anteriormente.

Desde la página de Ionic Framework te recomiendan seguir las siguientes guías:

- <https://cordova.apache.org/docs/en/7.x/guide/platforms/android/>
- <https://cordova.apache.org/docs/en/7.x/guide/platforms/ios/>

Ver apartado preparación 7. Preparación Entorno de Desarrollo con Android

6.6.2 Instalación de Ionic

Desde la consola de Node usamos el comando:

- **npm install -g cordova ionic**

```
[.....] - extract:is-wsl: still tarball no local data for is-git-url@1.0.0. Extracting by manifest.
```

6.6.3 Creando nuestra aplicación

En este punto tenemos diferentes opciones:

- ionic start myApp blank → la genera vacía
- ionic start myApp tabs → genera el proyecto con el tema de tabs
- ionic start HelloW_Ionic sidemenu → genera el proyecto con una barra de menú

Durante la generación de la creación de la App te preguntara si quieres probar Ionic 4 (en mí caso le he dicho que sí, hay que ir a la última):

```
PS G:\CommitConf_2018> ionic start HelloW_Ionic sidemenu
[INFO] You are about to create an Ionic 3 app. Would you like to try Ionic 4
(beta)?

Ionic 4 uses the power of the modern Web and embraces the Angular CLI and
Angular Router to bring you the best version of Ionic ever. See our blog
announcement[1] and documentation[2] for more information. We'd love to
hear your feedback on our latest version!

[1]: https://blog.ionicframework.com/announcing-ionic-4-beta/
[2]: https://beta.ionicframework.com/docs/

? Try Ionic 4? (y/N)
```

Luego instalara **IONIC DEVAPP** con ello podremos testear las aplicaciones sin tener los SDKs nativos de Android e iOS.

```
[INFO] Existing git project found (G:/CommitConf_2018). Git operations are
disabled.
✓ Preparing directory .\HelloW_Ionic - done!
✓ Downloading and extracting sidemenu starter - done!

Installing dependencies may take several minutes.

  *   IONIC  DEVAPP   *

Speed up development with the Ionic DevApp, our fast, on-device testing mobile app

-   Test on iOS and Android without Native SDKs
-   LiveReload for instant style and JS updates

-->   Install DevApp: https://bit.ly/ionic-dev-app   <--

> npm i
npm WARN deprecated circular-json@0.5.9: CircularJSON is in maintenance only, flattened is its successor.
[.....] | extract:strip-ansi: sill tarball trying dom-serialize@^2.2.0 by hash: sha1-ViromZ9Ev1
```

Después de estar un tiempo con la instalación te preguntara si quieres probar el SDK pro de Ionic, yo le he dicho que no.

```
  *   IONIC  PRO   *

Supercharge your Ionic development with the Ionic Pro SDK

-   Track runtime errors in real-time, back to your original TypeScript
-   Push remote updates and skip the app store queue

Learn more about Ionic Pro: https://ionicframework.com/pro

? Install the free Ionic Pro SDK and connect your app? (Y/n) |
```

6.6.4 Ejecutar la aplicación en navegador:

Usamos el comando.

- **ionic serve** en la terminal que estemos usando.

```
PS G:\CommitConf_2018\HelloW_Ionic> ionic serve
> ng run app:serve --host=0.0.0.0 --port=8100

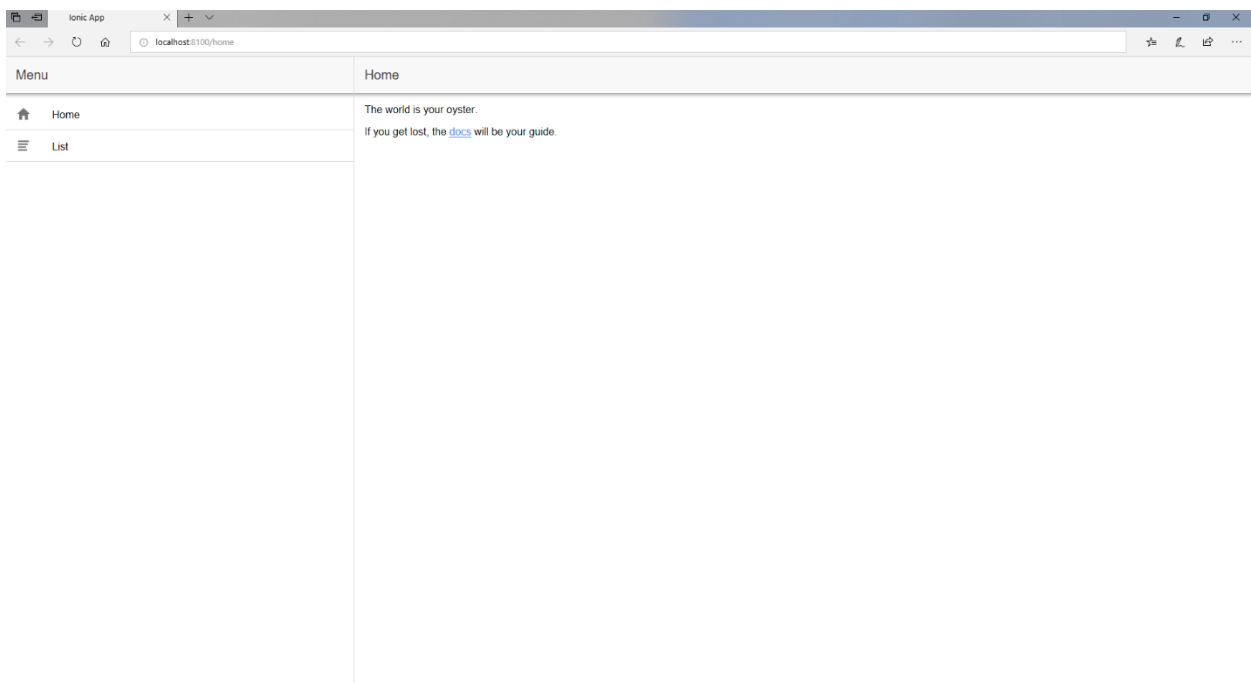
[INFO] Development server running!

Local: http://localhost:8100
External: http://192.168.1.89:8100
DevApp: HelloW_Ionic@8100 on DESKTOP-CFKKAT4

Use Ctrl+C to quit this process

[INFO] Browser window opened to http://localhost:8100!

[ng] i [wdm]: wait until bundle finished: /
```



6.6.5 Añadir platform Android al proyecto:

Usamos el comando:

- **ionic cordova platform add android --save**

```
CREATE resources\ios\icon\icon-76@2x.png
CREATE resources\ios\icon\icon-83.5@2x.png
CREATE resources\ios\icon\icon-small.png
CREATE resources\ios\icon\icon-small@2x.png
CREATE resources\ios\icon\icon-small@3x.png
CREATE resources\ios\icon\icon.png
CREATE resources\ios\icon\icon@2x.png
CREATE resources\ios\splash\Default-568h@2x~iphone.png
CREATE resources\ios\splash\Default-667h.png
CREATE resources\ios\splash\Default-736h.png
CREATE resources\ios\splash\Default-Landscape-736h.png
CREATE resources\ios\splash\Default-Landscape@2x~ipad.png
CREATE resources\ios\splash\Default-Landscape@~ipadpro.png
CREATE resources\ios\splash\Default-Landscape~ipad.png
CREATE resources\ios\splash\Default-Portrait@2x~ipad.png
CREATE resources\ios\splash\Default-Portrait@~ipadpro.png
CREATE resources\ios\splash\Default-Portrait~ipad.png
CREATE resources\ios\splash\Default@2x~iphone.png
CREATE resources\ios\splash\Default@2x~universal~anyany.png
CREATE resources\ios\splash\Default~iphone.png
[OK] Integration cordova added!
✓ Creating .\www directory for you - done!
> cordova platform add android --save
Using cordova-fetch for cordova-android@~7.1.1
```

.....

```
Installing "cordova-plugin-splashscreen" for android
Adding cordova-plugin-splashscreen to package.json
Saved plugin info for "cordova-plugin-splashscreen" to config.xml
Discovered plugin "cordova-plugin-ionic-webview" in config.xml. Adding it to the project
Installing "cordova-plugin-ionic-webview" for android
Subproject Path: CordovaLib
Subproject Path: app
Adding cordova-plugin-ionic-webview to package.json
Saved plugin info for "cordova-plugin-ionic-webview" to config.xml
Discovered plugin "cordova-plugin-ionic-keyboard" in config.xml. Adding it to the project
Installing "cordova-plugin-ionic-keyboard" for android
Adding cordova-plugin-ionic-keyboard to package.json
Saved plugin info for "cordova-plugin-ionic-keyboard" to config.xml
--save flag or autosave detected
Saving android@~7.1.3 into config.xml file ...
```

6.6.6 Añadir platform iOS al proyecto:

Usamos el comando:

- **ionic cordova platform add ios --save**

6.6.7 Ejecutar la aplicación en emulador:

Usamos el comando:

- **ionic cordova run Android --device** desde el terminal.

ERROR:

```
PS G:\CommitConf_2018\HelloW_Ionic> ionic cordova run Android --device
> cordova platform add Android --save
Using cordova-fetch for Android@7.1.3
Failed to fetch platform Android@7.1.3
Probably this is either a connection problem, or platform spec is incorrect.
Check your connection and platform name/version/URL.
Error: cmd: Command failed with exit code 1 Error output:
npm ERR! code E404
npm ERR! 404 Not Found: Android@7.1.3

npm ERR! A complete log of this run can be found in:
npm ERR! C:\Users\jmp8\AppData\Roaming\npm-cache\_logs\2018-11-22T17_01_54_660Z-debug.log
[ERROR] An error occurred while running subprocess cordova.

cordova platform add Android --save exited with exit code 1.

Re-running this command with the --verbose flag may provide more information.
```

El error puede ser por la API instalada en este caso yo solo tengo la 28, si miramos el cuadro ofrecido por Cordova, vemos que la última API compatible es la 27.

cordova-android Version	Supported Android API-Levels	Equivalent Android Version
7.X.X	19 - 27	4.4 - 8.1
6.X.X	16 - 26	4.1 - 8.0.0
5.X.X	14 - 23	4.0 - 6.0.1
4.1X	14 - 22	4.0 - 5.1
4.0.X	10 - 22	2.3.3 - 5.1
3.7X	10 - 21	2.3.3 - 5.0.2

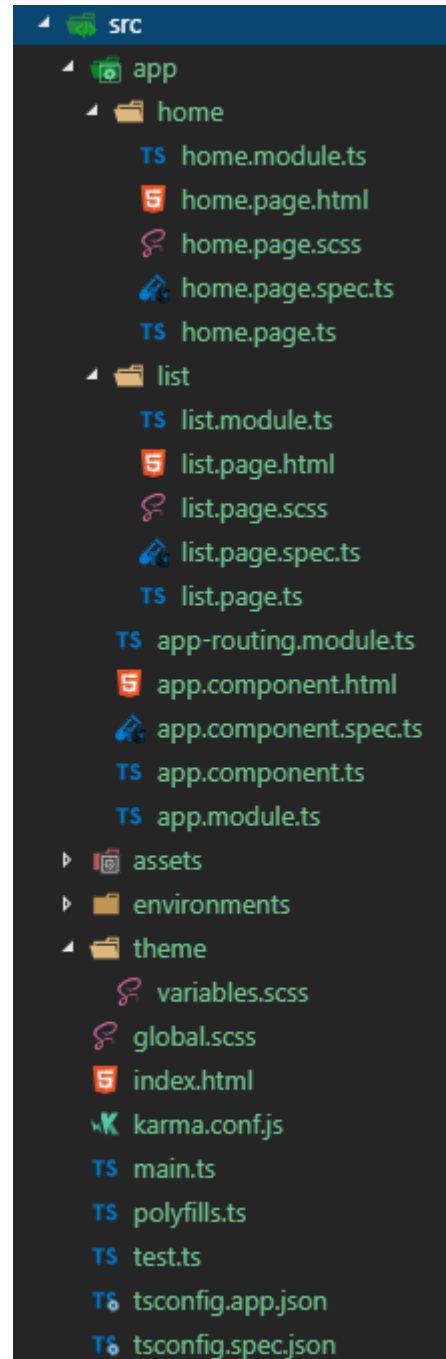
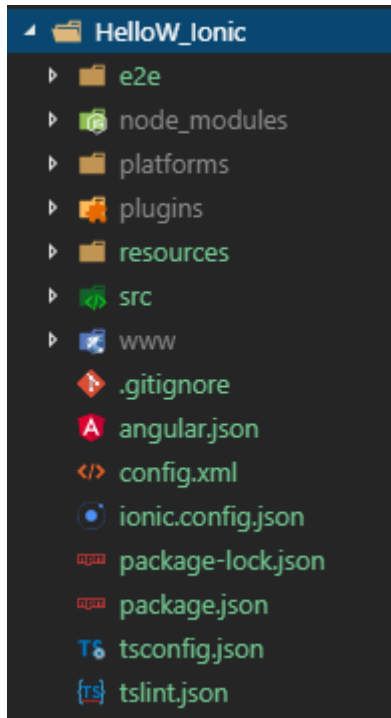
Siguiendo la preparación [7. Preparación Entorno de Desarrollo con Android](#) instalo la API 27.

6.6.8 Ejecutar o generar la aplicación para producción:

Usamos los comandos:

- **ionic cordova run android --prod --release**
- **ionic cordova build android --prod --release**

6.6.9 Arquitectura Creada



6.7 NativeScript

6.7.1 Preparación Entornos desarrollo Android e IOS

Igual que en el caso de Ionic, al ser un framework para desarrollar apps móviles se deben preparar sus entornos.

Desde la página de NativeScript nos recomiendan descargar tanto para Android como para IOS desde la App Store y Google Play las siguientes aplicaciones para que se usen durante el desarrollo de la aplicación.

- **NativeScript Playground**
- **NativeScript Preview**

(Personalmente no he probado ninguna de las dos)

Ver apartado preparación [7. Preparación Entorno de Desarrollo con Android](#)

6.7.2 Instalación de NativeScript

Desde la línea de comandos de Node usaremos el comando:

- **npm install -g nativescript**

Resumen instalación:

```
> nativescript@5.0.1 preuninstall C:\Users\jmrp8\AppData\Roaming\npm\node_modules\nativescript
> node preuninstall.js

C:\Users\jmrp8\AppData\Roaming\npm\nativescript -> C:\Users\jmrp8\AppData\Roaming\npm\node_modules\nativescript\bin\tns
C:\Users\jmrp8\AppData\Roaming\npm\tns -> C:\Users\jmrp8\AppData\Roaming\npm\node_modules\nativescript\bin\tns

> nativescript@5.0.1 postinstall C:\Users\jmrp8\AppData\Roaming\npm\node_modules\nativescript
> node postinstall.js

If you are using bash or zsh, you can enable command-line completion.
? Do you want to enable it now? No

Installation successful. You are good to go. Connect with us on http://twitter.com/NativeScript.

You have successfully installed the NativeScript CLI!
To create a new project, you use:
tns create <app name>

To build your project locally you use:
tns build <platform>

NOTE: Local builds require additional setup of your environment. You can find more information here: https://docs.nativescript.org/start/quick-setup

To build your project in the cloud you can use:
tns cloud build <platform>

NOTE: Cloud builds require Telerik account. You can find more information here: https://docs.nativescript.org/sidekick/intro/requirements

If you want to experiment with NativeScript in your browser, try the Playground: https://play.nativescript.org

If you have any questions, check Stack Overflow: https://stackoverflow.com/questions/tagged/nativescript and our public Slack channel: https://nativescriptcommunity.slack.com/

npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.4 (node_modules\nativescript\node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.4: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

+ nativescript@5.0.1
updated 1 package in 22.909s
```

Podemos comprobar el estado de la instalación mediante el comando:

- **tns doctor**

```
PS G:\CommitConf_2018> tns doctor
✓ Getting environment information

No issues were detected.
✓ Your ANDROID_HOME environment variable is set and points to correct directory.
✓ Your adb from the Android SDK is correctly installed.
✓ The Android SDK is installed.
✓ A compatible Android SDK for compilation is found.
✓ Javac is installed and is configured properly.
✓ The Java Development Kit (JDK) is installed and is configured properly.
✓ Local builds for iOS can be executed only on a macOS system. To build for iOS on a different operating system, you can use the NativeScript cloud infrastructure.
✓ Getting NativeScript components versions information...
✓ Component nativescript has 5.0.1 version and is up to date.
PS G:\CommitConf_2018>
```

6.7.3 Creando nuestra Aplicación

Desde la consola de comandos y situados en la raíz donde queramos crear el nuevo proyecto escribimos:

- **tns create HelloWorld --ng** (opción con template de angular)

Después de realizar la extracción, nos muestra la siguiente información:

```
> node postinstall.js

Adding 'es6' lib to tsconfig.json...
Adding 'dom' lib to tsconfig.json...
Adding 'es2015.iterable' lib to tsconfig.json...
Adding tns-core-modules path mappings lib to tsconfig.json...
Installing TypeScript...
+ typescript@3.1.6
updated 1 package and audited 16956 packages in 8.693s
found 0 vulnerabilities

npm notice created a lockfile as package-lock.json. You should commit this file.
npm WARN extract-text-webpack-plugin@3.0.2 requires a peer of webpack@^3.1.0 but none is installed. You must install peer dependencies yourself.

> nativescript-dev-webpack@0.18.0 postinstall G:\CommitConf_2018\HelloW_NativeScript\node_modules\nativescript-dev-webpack
> node postinstall.js

Creating file: G:\CommitConf_2018\HelloW_NativeScript\webpack.config.js
Creating file: G:\CommitConf_2018\HelloW_NativeScript\tsconfig.tns.json
Adding dev dependency: @angular/compiler-cli@~7.0.0
Adding dev dependency: @ngtools/webpack@~7.0.0

NativeScript Webpack plugin was successfully added.
You can now bundle your project by passing --bundle flag to NativeScript CLI commands:
  - tns build android --bundle
  - tns build ios --bundle
  - tns run android --bundle
  - tns run ios --bundle

You can also pass the "--env.uglify" flag to use UglifyJS for minification.
For more information check out https://docs.nativescript.org/tooling/bundling-with-webpack#bundling.

A few new dependencies were added. Run "npm install" before building your project.

npm WARN extract-text-webpack-plugin@3.0.2 requires a peer of webpack@^3.1.0 but none is installed. You must install peer dependencies yourself.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.4 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.4: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

added 620 packages from 424 contributors and audited 16955 packages in 135.675s
found 0 vulnerabilities

Project HelloWorld_NativeScript was successfully created.

Now you can navigate to your project with $ cd HelloWorld_NativeScript

After that you can preview it on device by executing $ tns preview

PS G:\CommitConf_2018>
```

Otras opciones:

- **tns create MyApp**: Crea la aplicación con el tema básico.
- **tns create MyApp --tsc**: Crea la aplicación usando el template de TypeScript
- **tns create MyApp --template nativescript-template-tutorial**: Crea la aplicación con el template del tutorial oficial.

Una vez creado abrimos la carpeta desde nuestro IDE, desde este momento las instrucciones serán referentes al uso del VSCode que nos incorpora la terminal en el propio IDE.

6.7.4 Preparar las platform:

Usamos los comandos desde la consola:

- **tns prepare android** (tns prepare ios).

```
PS G:\CommitConf_2018\HelloW_NativeScript> tns prepare android
[.....] - extract:@angular/compiler-cli: sill extract @angular/compiler-cli@7.0.0 extracted to G:\CommitConf_2
```

Suele emplear su tiempo hasta que termina así que paciencia.

```
dependencies yourself.
npm WARN optional SKIPPING OPTIONAL DEPENDENCY: fsevents@1.2.4 (node_modules\fsevents):
npm WARN notsup SKIPPING OPTIONAL DEPENDENCY: Unsupported platform for fsevents@1.2.4: wanted {"os":"darwin","arch":"any"} (current: {"os":"win32","arch":"x64"})

added 83 packages from 91 contributors and audited 20292 packages in 75.613s
found 0 vulnerabilities

Copying template files...
Platform android successfully added. v5.0.0
Executing before-shouldPrepare hook from G:\CommitConf_2018\HelloW_NativeScript\hooks\before-shouldPrepare\nativescript-dev-
webpack.js
Executing before-prepare hook from G:\CommitConf_2018\HelloW_NativeScript\hooks\before-prepare\nativescript-dev-typescript.j
Found peer TypeScript 3.1.6
Preparing project...
Executing before-prepareJSApp hook from G:\CommitConf_2018\HelloW_NativeScript\hooks\before-prepareJSApp\nativescript-dev-we
bpack.js
Successfully prepared plugin nativescript-angular for android.
Successfully prepared plugin nativescript-theme-core for android.
Successfully prepared plugin tns-core-modules for android.
Successfully prepared plugin nativescript-intl for android.
Successfully prepared plugin tns-core-modules-widgets for android.
Project successfully prepared (android)
Executing after-prepare hook from G:\CommitConf_2018\HelloW_NativeScript\hooks\after-prepare\nativescript-dev-webpack.js
PS G:\CommitConf_2018\HelloW_NativeScript>
```

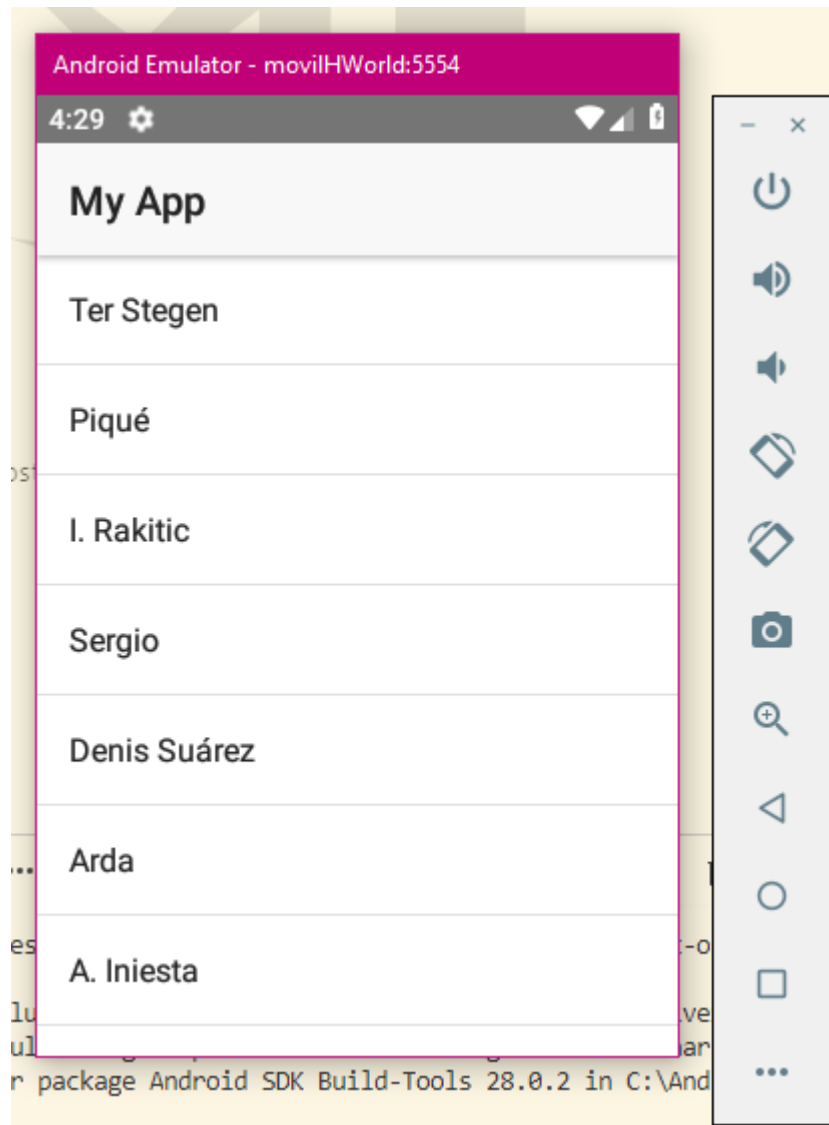
- **tns platform add <ios or Android>** (añadir platform)

6.7.5 Ejecución de la aplicación:

Ejecutar la aplicación, por ejemplo, escribimos en la terminal:

- **tns run Android**
- **tns run Android --emulator**

La primera vez suele tardar más de lo habitual (a mí unos 12 minutos, actualizo tools de Android),



- **tns run ios**

En el caso de Android si no se ejecuta y nos dice no encuentra el SDK suele ser problemas en el path.

6.7.6 Dispositivos móviles disponibles:

- **tns device <platform> --available-devices** donde <platform> será Android o iOS.
- **tns device android --watch**

6.7.7 Algunos errores al ejecutar en emulador:

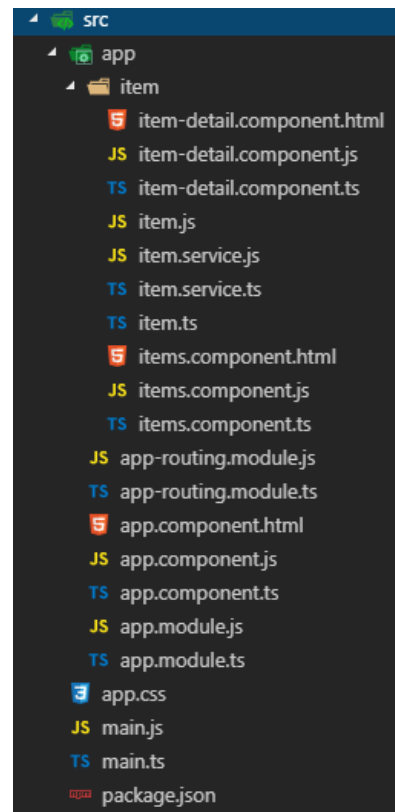
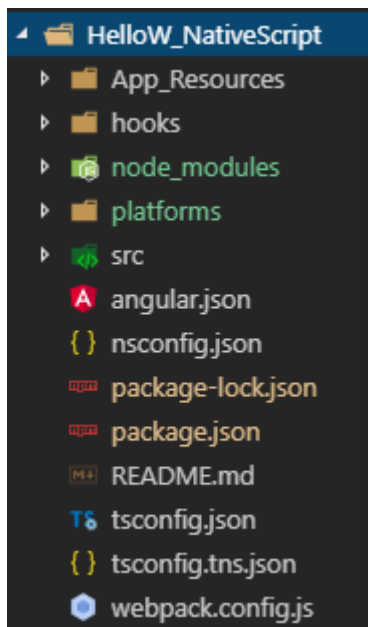
Para ciertos problemas no reconoce el device, en mi caso era porque no estaba bien instalado "Intel HAXM":

<https://software.intel.com/en-us/articles/intel-hardware-accelerated-execution-manager-intel-haxm>

Al hacer `tns run Android --emulator` he obtenido errores casi al final de la preparación de la aplicación para la instalación en el emulador.

Lo que he hecho es eliminar las platforms y ejecutar `tns run Android`. Y ha ido todo OK

6.7.8 Arquitectura creada



6.8 ReactNative

6.8.1 Preparación Entornos desarrollo Android e iOS

Igual que en el caso de Ionic y NativeScript al ser un framework para desarrollar apps móviles se deben preparar sus entornos.

Ver apartado preparación [7. Preparación Entorno de Desarrollo con Android](#)

6.8.2 Instalación de ReactNative

Desde la línea de comandos de Node usamos el comando que nos instalara el expo-cli usado por ReactNative:

- **npm install -g expo-cli**

6.8.3 Creación de un proyecto

Desde la línea de comandos usamos el comando:

- **expo init NameProject** ex: expo init HelloW_ReactNative

Nos mostrara la siguiente información para que escojamos entre el template **blank** o de **tabs**. Yo he elegido el de tabs.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

PS G:\CommitConf_2018> expo init HelloW_ReactNative
There is a new version of expo-cli available (2.4.0).
You are currently using expo-cli 2.2.5
Run 'npm install -g expo-cli' to get the latest version
? Choose a template: (Use arrow keys)
> blank
  The Blank project template includes the minimum dependencies to run and an empty root component.
  tabs
  The Tab Navigation project template includes several example screens.
```

Después de un rato (paciencia) nos mostrara la siguiente info:

```
? Choose a template: tabs
[16:30:43] Extracting project files...
[16:34:22] Customizing project...

Your project is ready at G:\CommitConf_2018\HelloW_ReactNative
To get started, you can type:

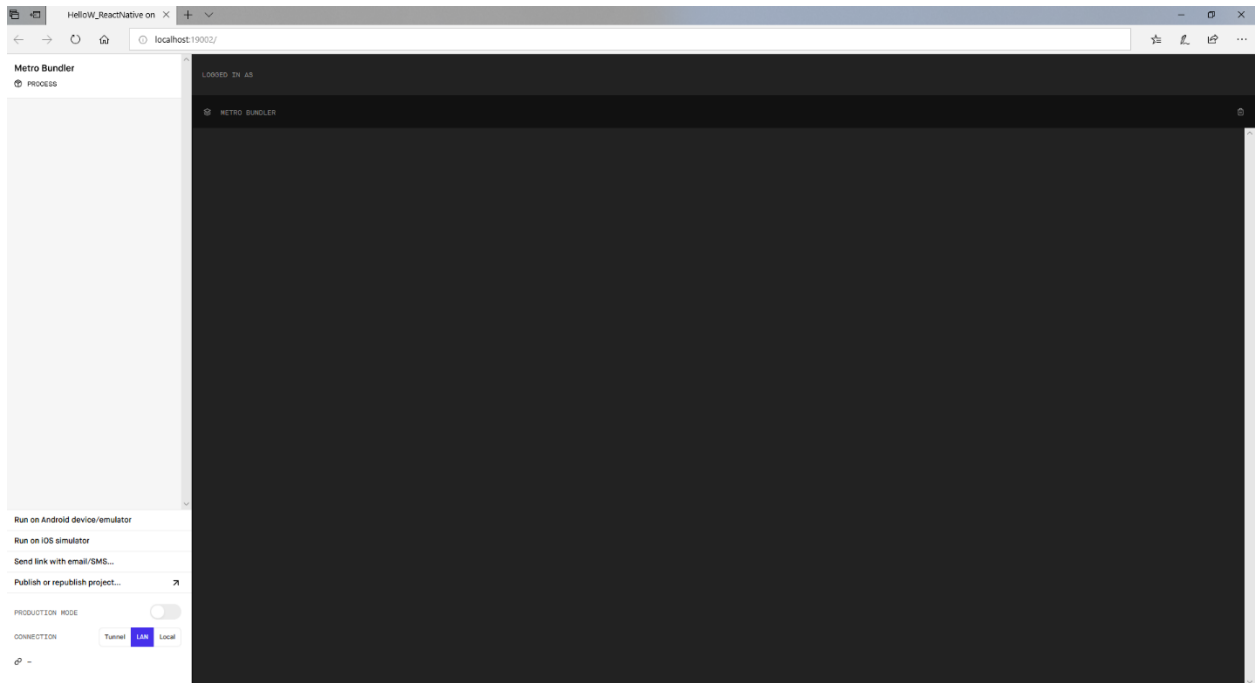
  cd HelloW_ReactNative
  expo start
```

6.8.4 Ejecución del proyecto

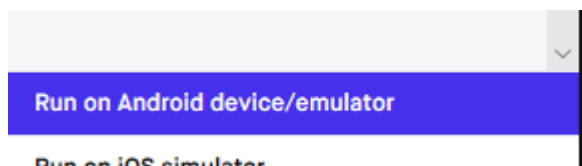
Desde la consola de comandos y la carpeta del proyecto usamos el comando:

- **npm start**

Esto nos abrirá en el navegador un asistente de expo para trabajar con la aplicación realizada. (Acordaros tener el emulador de Android ejecutado).

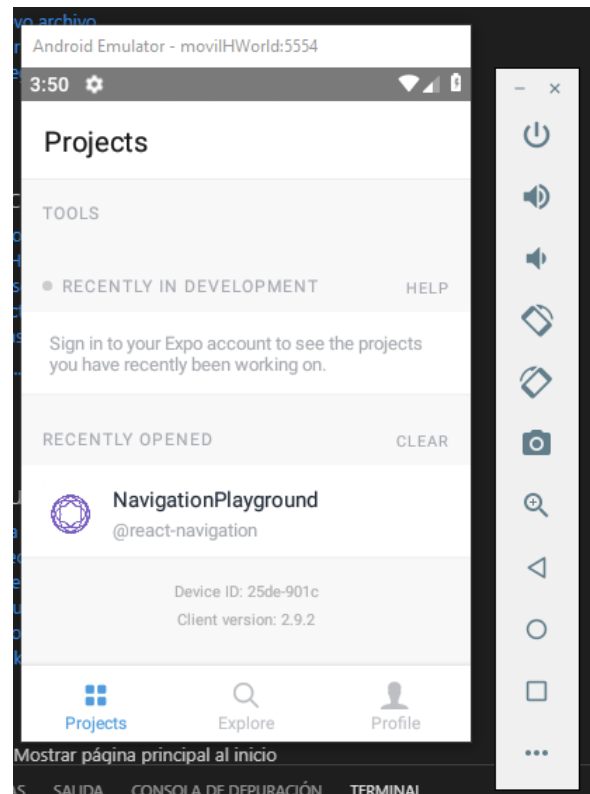
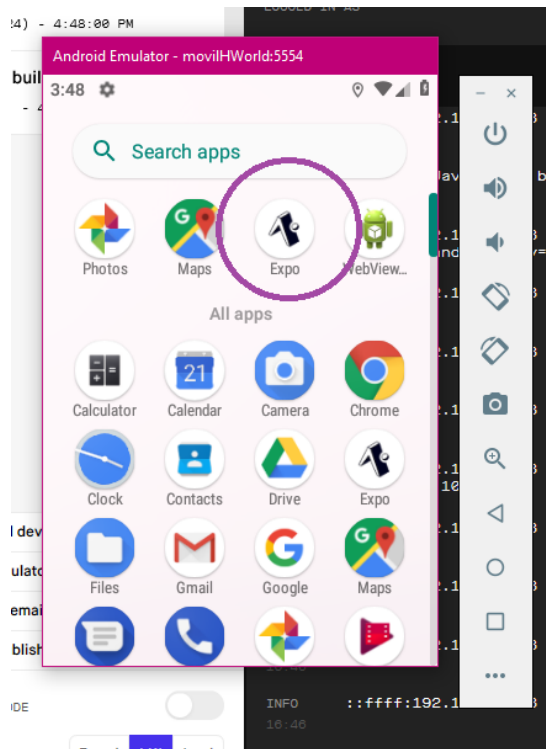


Pulsamos sobre la opción



Esto nos ejecuta la aplicación sobre el emulador de Android, que aparecerá en el dispositivo bajo el símbolo y nombre de expo.

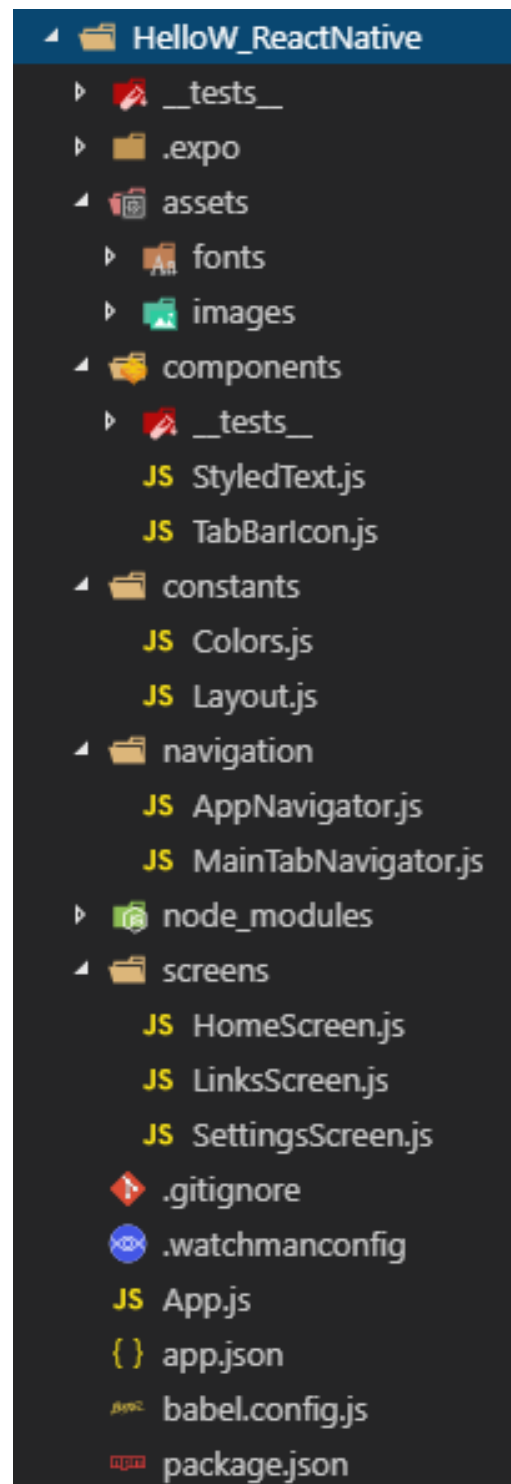
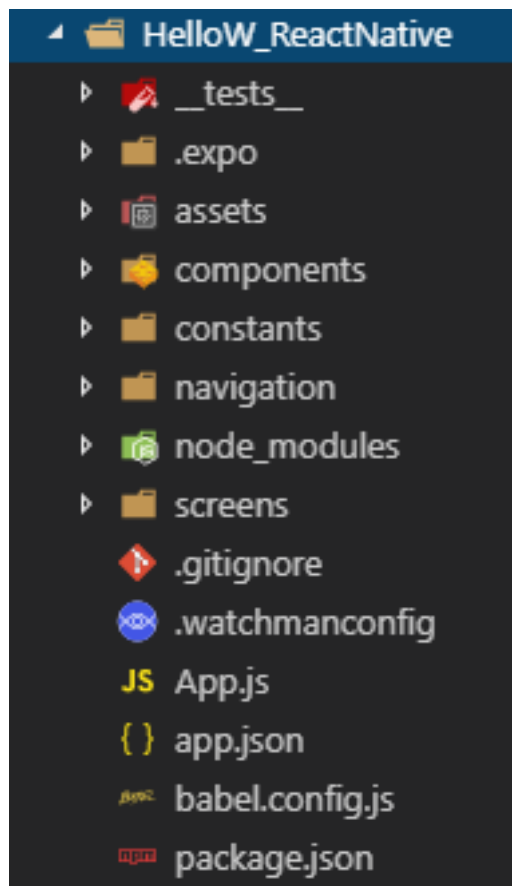
Al pulsar sobre el icono nos mostrara la aplicación con el contenido que genera por defecto el expo-cli:



6.8.5 Comandos Útiles

- **npm start** (inicia la web de soporte de expo)
- **npm android** (ejecutar android)
- **npm ios** (ejecutar ios)
- **npm test** (ejecutar test)

6.8.6 Arquitectura Creada



7. Preparación Entorno de Desarrollo con Android

7.1.1 Introducción

Como hemos visto para los frameworks dirigidos a aplicaciones móviles es recomendable preparar el entorno para poder usar simuladores o dispositivos virtuales.

Es cierto que muchos de estos frameworks al ejecutarlos tienen simuladores vía web, vía móvil a través de apps para los dispositivos móviles.

En mi caso el entorno que he preparado ha sido solo para Android ya que como muchos podréis saber para preparar un entorno de IOS debes tener dispositivos propios de Apple.

Llegado a este punto para el entorno de Android frameworks como Ionic o NativeScript te recomiendan la instalación del IDE Android Studio.

Esto conlleva instalar un IDE que posiblemente no se vaya a usar solo para poder crear dispositivos virtuales y gestionar los SDK de Android con las herramientas **AVDManager** y **SDKManager**.

Hace unos años desde la web oficial de Android te permitían descargar por separado estas herramientas.

En mi caso he usado dos opciones usar la vía de instalar **Android Studio** y la vía del uso de **Chocolatey** (manejador de paquetes para Windows).

Debemos tener en cuenta que deberemos tener Node instalado y en el caso de Android el JDK (Java Development Kit).

7.1.2 Preparación de Android con Chocolatey

7.1.2.1 Para instalar Chocolatey pasos descritos en su web:

Desde cmd.exe pegamos el siguiente comando:

```
@"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -InputFormat None -ExecutionPolicy Bypass -Command "iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))" && SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"
```

Desde PowerShell.exe pegamos el siguiente comando:

```
Set-ExecutionPolicy Bypass -Scope Process -Force; iex ((New-Object System.Net.WebClient).DownloadString('https://chocolatey.org/install.ps1'))
```

Una vez terminada la instalación es recomendable reiniciar la consola o terminal de comandos.

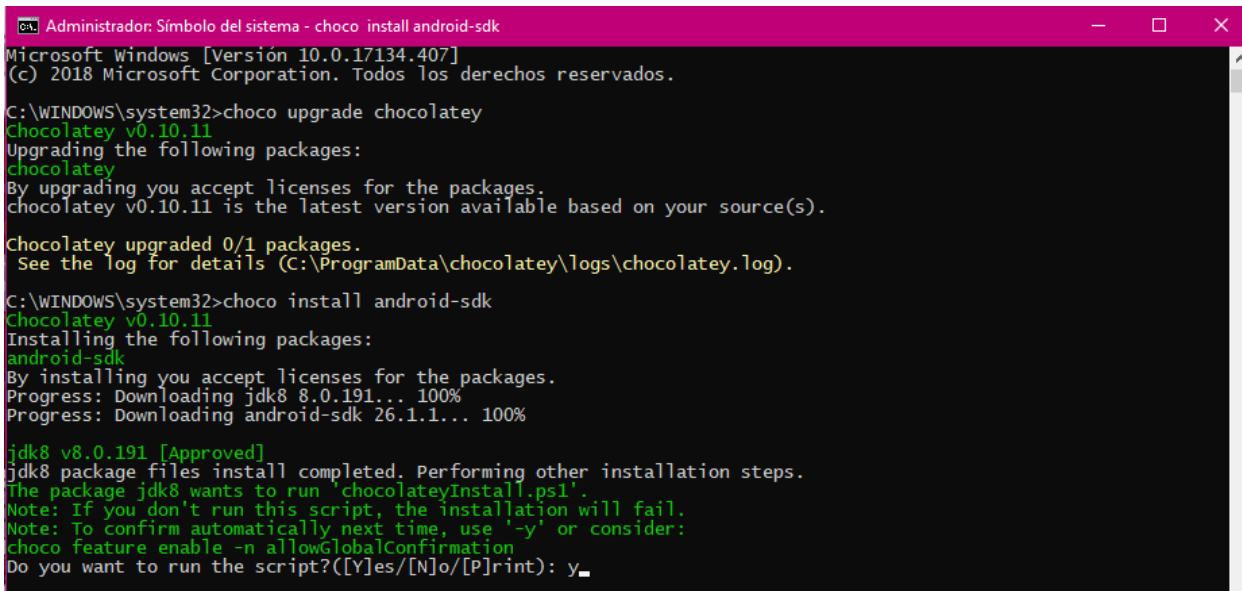
Para actualizar Chocolatey usamos el comando:

- choco upgrade chocolatey

Para instalar AndroidSDK:

Usamos el comando:

- choco install android-sdk



```
Administrador: Símbolo del sistema - choco install android-sdk
Microsoft Windows [Versión 10.0.17134.407]
(c) 2018 Microsoft Corporation. Todos los derechos reservados.

C:\WINDOWS\system32>choco upgrade chocolatey
Chocolatey v0.10.11
Upgrading the following packages:
chocolatey
By upgrading you accept licenses for the packages.
chocolatey v0.10.11 is the latest version available based on your source(s).
Chocolatey upgraded 0/1 packages.
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).

C:\WINDOWS\system32>choco install android-sdk
Chocolatey v0.10.11
Installing the following packages:
android-sdk
By installing you accept licenses for the packages.
Progress: Downloading jdk8 8.0.191... 100%
Progress: Downloading android-sdk 26.1.1... 100%

jdk8 v8.0.191 [Approved]
jdk8 package files install completed. Performing other installation steps.
The package jdk8 wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[N]o/[P]rint): y_
```

Introducimos la 'Y' pulsamos 'Intro'.
Comenzará verificando la instalación del JDK y luego seguirá con Android-SDK.

```
Administrador: Símbolo del sistema - choco install android-sdk
See the log for details (C:\ProgramData\chocolatey\logs\chocolatey.log).
C:\WINDOWS\system32>choco install android-sdk
Chocolatey v0.10.11
Installing the following packages:
android-sdk
By installing you accept licenses for the packages.
Progress: Downloading jdk8 8.0.191... 100%
Progress: Downloading android-sdk 26.1.1... 100%

jdk8 v8.0.191 [Approved]
jdk8 package files install completed. Performing other installation steps.
The package jdk8 wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[N]o/[P]rint): y

Downloading JDK from http://download.oracle.com/otn-pub/java/jdk/8u191-b12/2787e4a523244c269598db4e85c51e0c/jdk-8u191-windows-x64.exe
Installing jdk8...
jdk8 has been installed.
PATH environment variable does not have C:\Program Files\Java\jdk1.8.0_191\bin in it. Adding...
Environment Vars (like PATH) have changed. Close/reopen your shell to
see the changes (or in powershell/cmd.exe just type 'refreshenv').
The install of jdk8 was successful.
Software installed as 'exe', install location is likely default.

android-sdk v26.1.1 [Approved]
android-sdk package files install completed. Performing other installation steps.
The package android-sdk wants to run 'chocolateyInstall.ps1'.
Note: If you don't run this script, the installation will fail.
Note: To confirm automatically next time, use '-y' or consider:
choco feature enable -n allowGlobalConfirmation
Do you want to run the script?([Y]es/[N]o/[P]rint):
```

Volvemos a introducir la 'y' y pulsamos intro.

Para actualizar AndroidSDK usamos el comando:

- `choco upgrade android-sdk`

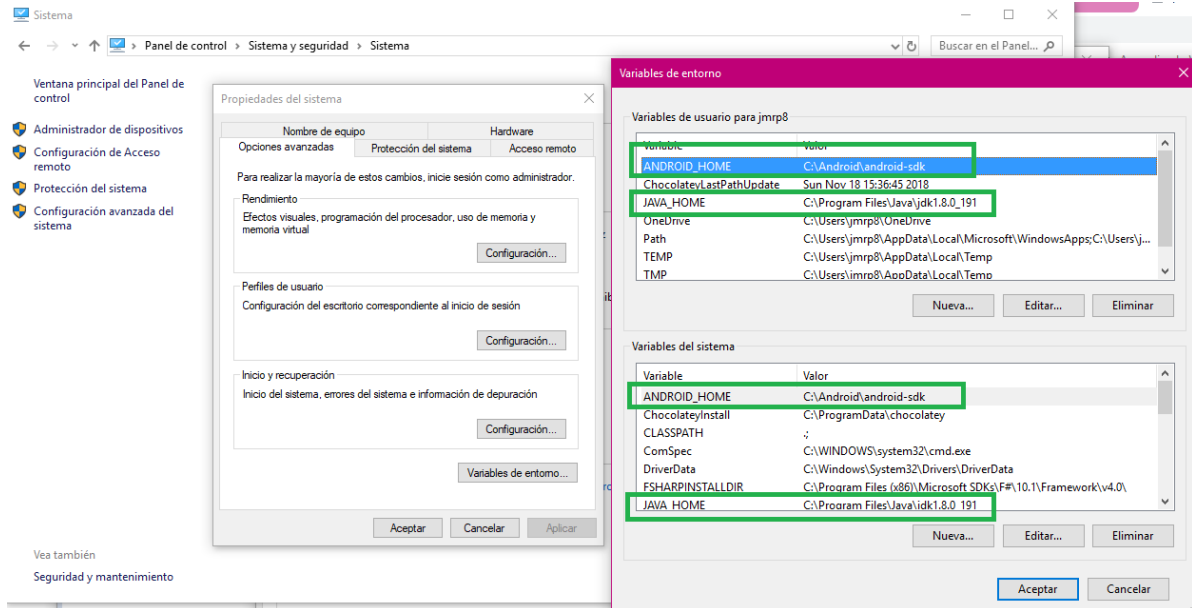
En el caso del proyecto de NativeScript, una vez seguido los anteriores pasos y habiendo reiniciado el terminal.

Instalación de Gradle usamos el comando:

- `choco install gradle`

7.1.2.2 Preparación AVDs (Dispositivos virtuales):

Comprobar las variables del entorno.



Como crear desde línea de comandos un dispositivo virtual:

Desde la carpeta de tools/bin en Android/Android-sdk

Preparamos el sdk de la versión 25 o 28 (La versión 25 la última vez me dio algún problema)

Ejecutamos:

- sdkmanager "system-images;android-25;google_apis;x86"
- sdkmanager "system-images;android-28;google_apis;x86"

```
c:\Android\android-sdk\tools\bin>sdkmanager "system-images;android-25;google_apis;x86"
[=====] 100% Unzipping... x86/encryptionkey.im
```

Si todo ha ido correctamente nos debería crear la carpeta emulador dentro de la carpeta Android-sdk.

Escribimos el comando:

- sdkmanager --licenses

```
c:\Android\android-sdk\tools\bin>sdkmanager --licenses
5 of 6 SDK package licenses not accepted. 100% Computing updates...
Review licenses that have not been accepted (y/N)?
```

Escibimos 'y'

Nos sale:

```

C:\Administrador: Símbolo del sistema - sdkmanager --licenses
INFRINGEMENT.

11. LIMITATION OF LIABILITY

11.1 YOU EXPRESSLY UNDERSTAND AND AGREE THAT GOOGLE, ITS SUBSIDIARIES AND AFFILIATES, AND ITS LICENSORS SHALL NOT BE LIA
BLE TO YOU UNDER ANY THEORY OF LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL CONSEQUENTIAL OR EXEMPLARY DAMAGE
S THAT MAY BE INCURRED BY YOU, INCLUDING ANY LOSS OF DATA, WHETHER OR NOT GOOGLE OR ITS REPRESENTATIVES HAVE BEEN ADVISE
D OF OR SHOULD HAVE BEEN AWARE OF THE POSSIBILITY OF ANY SUCH LOSSES ARISING.

12. Indemnification

12.1 To the maximum extent permitted by law, you agree to defend, indemnify and hold harmless Google, its affiliates and
their respective directors, officers, employees and agents from and against any and all claims, actions, suits or proce
edings, as well as any and all losses, liabilities, damages, costs and expenses (including reasonable attorneys fees) ar
ising out of or accruing from (a) your use of the Google TV Add-on, (b) any application you develop on the Google TV Add
-on that infringes any copyright, trademark, trade secret, trade dress, patent or other intellectual property right of a
ny person or defames any person or violates their rights of publicity or privacy, and (c) any non-compliance by you with
this License Agreement.

13. Changes to the License Agreement

13.1 Google may make changes to the License Agreement as it distributes new versions of the Google TV Add-on.

14. General Legal Terms

14.1 This License Agreement constitute the whole legal agreement between you and Google and govern your use of the Googl
e TV Add-on (excluding any services which Google may provide to you under a separate written agreement), and completely
replace any prior agreements between you and Google in relation to the Google TV Add-on.

14.2 You agree that if Google does not exercise or enforce any legal right or remedy which is contained in this License
Agreement (or which Google has the benefit of under any applicable law), this will not be taken to be a formal waiver of
Google's rights and that those rights or remedies will still be available to Google.

14.3 If any court of law, having the jurisdiction to decide on this matter, rules that any provision of this License Agr
eement is invalid, then that provision will be removed from this License Agreement without affecting the rest of this Li
cense Agreement. The remaining provisions of this License Agreement will continue to be valid and enforceable.

14.4 You acknowledge and agree that Google's API data licensors and each member of the group of companies of which Googl
e is the parent shall be third party beneficiaries to this License Agreement and that such other companies shall be enti
tled to directly enforce, and rely upon, any provision of this License Agreement that confers a benefit on (or rights in
favor of) them. Other than this, no other person or company shall be third party beneficiaries to this License Agreemen
t.

14.5 EXPORT RESTRICTIONS. THE GOOGLE TV ADD-ON IS SUBJECT TO UNITED STATES EXPORT LAWS AND REGULATIONS. YOU MUST COMPLY
WITH ALL DOMESTIC AND INTERNATIONAL EXPORT LAWS AND REGULATIONS THAT APPLY TO THE GOOGLE TV ADD-ON. THESE LAWS INCLUDE R
ESTRICTIONS ON DESTINATIONS, END USERS AND END USE.

14.6 The rights granted in this License Agreement may not be assigned or transferred by either you or Google without the
prior written approval of the other party. Neither you nor Google shall be permitted to delegate their responsibilities
or obligations under this License Agreement without the prior written approval of the other party.

14.7 This License Agreement, and your relationship with Google under this License Agreement, shall be governed by the la
ws of the State of California without regard to its conflict of laws provisions. You and Google agree to submit to the e
xclusive jurisdiction of the courts located within the county of Santa Clara, California to resolve any legal matter ari
sing from this License Agreement. Notwithstanding this, you agree that Google shall still be allowed to apply for injunc
tive remedies (or an equivalent type of urgent legal relief) in any jurisdiction.

August 15, 2011
-----
Accept? (y/N):

```

Y aceptamos escribiendo 'Y', y así se repite unas cinco veces.

avdmanager create avd -n name -k "sdk_id" [-c {path|size}] [-f] [-p path]

ex: avdmanager create avd -n moviilHWorld -k "system-images;android-28;google_apis;x86" -f

```

C:\Android\android-sdk\tools\bin>avdmanager create avd -n testHWorld -k "system-images;android-25;google_apis;x86"
Auto-selecting single ABI x86=====] 100% Fetch remote repository...
Do you wish to create a custom hardware profile? [no]

```

Desde la carpeta `android\android-sdk\emulator>`

listar dispositivos: `emulator -list-avds`

ejecutar dispositivo: `emulator @movilHWorld`

7.1.3 Preparación de Android con Android Studio

Esta forma puede ser la más sencilla, aunque no se use el IDE, siguiendo los pasos de la instalación desde la web:

- <https://developer.android.com/studio/>

Una vez instalado, sin necesidad de crear un proyecto desde la pantalla de inicio del IDE en la parte inferior podemos acceder a la configuración y desde ella a las opciones de SDK y tools para Android.

Pero para crear un dispositivo virtual yo he creado un proyecto vacío y una vez abierto he accedido al AVDManager para crearlo y lanzarlo.

También se puede hacer desde la terminal de Windows con los mismos comandos que los del apartado anterior sobre chocolatey.

7.1.4 Emulador de Windows

Como Para Windows 8 o superior se puede usar alternativamente su emulador de Android, (no lo he probado), para más info:

<https://www.visualstudio.com/vs/msft-android-emulator/>

8. Bibliografía

Enlaces interesantes para tener en cuenta o donde se ha obtenido información.

- Visual Studio Code: <https://code.visualstudio.com/>
- Node: <https://nodejs.org/es/>
- npm: <https://www.npmjs.com/>
- Typescript: <https://www.typescriptlang.org/>
- ESLint: <https://eslint.org/>
- TSLint: <https://palantir.github.io/tslint/>
- Angular: <https://angular.io/>
- Angular – Material: <https://material.angular.io/>
- Reactjs: <https://reactjs.org/>
- NPM: <https://www.npmjs.com/>
- React Developer tools extensión:
 - Chrome: <https://chrome.google.com/webstore/detail/react-developer-tools/fmkadmapgofadopljbjfkapdkoienihi>
 - Mozilla: <https://addons.mozilla.org/es/firefox/addon/react-devtools/>
- VueJS: <https://vuejs.org/>
- vue-devtools: <https://github.com/vuejs/vue-devtools#vue-devtools>
- Vue.JS school:
https://vueschool.io/?utm_source=Vuejs.org&utm_medium=Banner&utm_campaign=Sponsor&utm_content=V1
- Presentación de jdonsan “charla-aprendiendo-vuejs”:
<https://jdonsan.github.io/charla-aprendiendo-vuejs/slides/#slide=1>
- Aurelia: <https://aurelia.io/>
- Ionic: <https://ionicframework.com/>
- Nativescript: <https://www.nativescript.org/>
- ReactNative: <https://facebook.github.io/react-native/>
- Android Developer: <https://developer.android.com/studio/#downloads>
- Apple Developer: <https://developer.apple.com/develop/>
- Chocolatey Android SDK: <https://chocolatey.org/packages/android-sdk>
- Cordova: <https://cordova.apache.org/>
- Cordova Android Platforms:
<https://cordova.apache.org/docs/en/latest/guide/platforms/android/index.html>

9. Control de versiones

Versión	Descripción	Fecha
1.0	Versión Inicial	10/11/2018
1.5	Versión Pre-CommitConf18	23/11/2018