Angular 2 hace un par de meses que ya tiene una versión final. Todo el mundo esperaba desde un principio un éxito arrollador. Sin embargo en mas de una ocasión me he encontrado con respuestas un poco tibias. De hecho hay cada día mas gente mirando a Facebook y a su framework React como una alternativa a Angular 2. ¿Porque se produce esto? , ¿No era Angular el framework preferido por todos?. ¿Qué es lo que esta haciendo a la gente dudar? . Vamos a hablar un poco del tema.

Angular 2

¿Qué es lo que ha cambiado con Angular?. La respuesta es : Muchas cosas y puede que en un primer punto de partida te parezcan demasiadas.

TypeScript: El primer cambio importante es que hay una apuesta por TypeScript como lenguaje de programación , aunque se puede seguir desarrollando en JavaScript plano si quieres. Esto nos guste o no es una barrera, tenemos que aprender un nuevo lenguaje. Algo que llevara tiempo y genera dudas.

Hola Mundo: El ejemplo de hola mundo hay que bajarselo de GitHub y no es precisamente un único fichero , sino que es un proyecto grande .

RxJS: El manejo de promesas para gestionar peticiones asíncronas ya tiene su complejidad. Usar RxJS nos permite avanzar más pero también es algo más que aprender.

Componentes vs Controladores: Se diseña el framework orientado a componentes y no a controladores lo que implica cambios en la forma de pensar por parte del desarrollador.

Angular 2 y 2.2: En dos meses hemos pasado a la versión 2.2 lo lo parece que hay muchas cosas que había que afinar.

Podríamos seguir hablando de cosas , pero esa es la realidad, hay bastantes cosas que aprender y es un framework nuevo.

¿Google porqué todo esto?

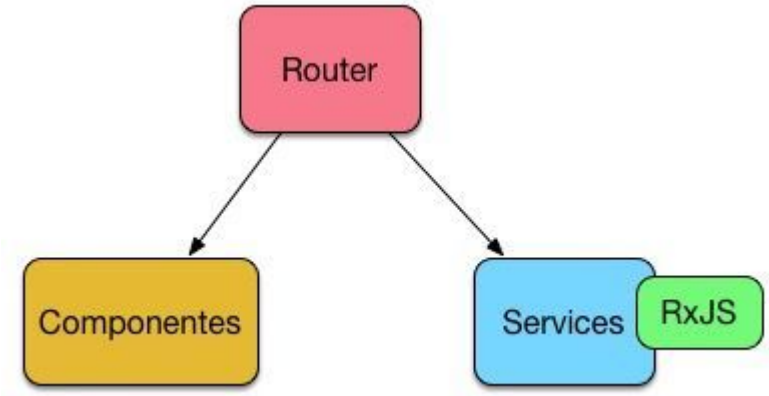
Es una buena pregunta , ¿Por qué tantos cambios? . Angular 1.x funcionaba tan bien que tengo mas de un amigo que me dice .. con lo que yo controlaba la versión 1 … era una máquina con ella. Ahora a volver a empezar de cero.
A mi siempre me gusta reflexionar un poco más a fondo sobre estos temas. ¿Se ha equivocado Google o ha acertado?. Para mí ha acertado y voy a intentar explicar el porqué lo veo así. ¿Qué problemas tiene JavaScript importantes para convertirse en referencia?

Mis Reflexiones

La modularización : JavaScript no soporta módulos de forma “natural” hasta la versión ES6. No solo eso sino que los navegadores ni siquiera esta muy claro como cargar de forma asíncrona los módulos y sus dependencias. ¿Angular 2 soluciona esto? . Sí a través del sistema de módulos de TypeScript que es parecido a ES6 y usando System.js como cargador de módulos. Esto ayudará a reducir el caos.

Compilación : Es verdad que para esto hay gustos . Hay gente que le gustan los lenguajes compilados y a otros los no compilados como Javascript por su flexibilidad. Pero si miramos a las grandes plataformas Java y .NET son compilados. Así que apostar por TypeScript un lenguaje de Microsoft ,me parece bastante correcto.

Arquitectura: Podemos decir que hay muchos cambios, pero la arquitectura MVC se mantiene. Disponemos de Componentes para la vista , Enrutador para la capa de control y servicios para la capa de backend.



El framework aporta una arquitectura y separación de responsabilidades clara. Algo que en JavaScript siempre hace falta.

Metadata y Anotaciones : Han apostado por añadir anotaciones y metadatos como hacen las otras dos grandes plataformas para añadir funcionalidad transversal. . Era algo que se echaba mucho en falta en JavaScript. Si miramos el ejemplo de hola mundo de Angular lo podemos ver.

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

@Component({
selector: 'my-app',
template: `

<h1>Hello {{name}}</h1>


`
})
export class AppComponent { name = 'Angular'; }
```

@Component es una anotación que define de que tipo de componente es nuestra clase.

Inyección de dependencias: Soportan de forma muy natural las inyecciones de dependencias apoyándose en TypeScript. Esta es una de las cosas que a mi me parecen fundamentales. Es verdad que era algo ya previamente soportado , pero recordemos que con un inyector ya no somos nosotros los encargados de crear objetos sino que lo hace el framework. Se puede añadir funcionalidad transversal de forma transparente utilizando proxies.

Angular 2 y el futuro

Todo esto esta muy bien , es cierto que Angular 2 es un avance .. ¿Pero merece la pena? . Esa no es una pregunta importante , NO es la pregunta correcta. La pregunta correcta es : ¿Es Angular un framework de cliente? . ¿Podría usar todo lo que trae del lado del servidor?. La respuesta es Sí , existe otro proyecto, Angular Universal que nos permite ejecutar Angular 2 desde el Servidor .

El salto de Google ha sido tan fuerte porque su objetivo no es quedarse con la plataforma de cliente , que ya la tienen muy orientada sino que su objetivo es abrir las puertas a la plataforma servidor en donde Java y .NET mandan.

Es donde Node.js cojea porque no tiene ningún framework de integración sólido. ¿Puede ser Angular 2 un competidor de Spring framework? esa era una pregunta que antes no estaba en la mesa y puede que a futuro lo esté. Google ha sido valiente y ha arriesgado asumiendo un salto tecnológico importante para poder ganar mucho más en el futuro. Angular 2 es muy joven y hay que esperar, pero puede ser una de las plataformas con más futuro **y**

tendremos que estar muy atentos.

Otros artículos relacionados: Angular.js vs Node.js , Angular.js inyección de dependencia y COC ,Arquitecturas Web y su evolución