
DECOR
HOGAR
Vestimos tu hogar

INDICE

1. Análisis

El proceso de desarrollo de aplicaciones web se puede dividir en diferentes etapas del ciclo de vida. Esto puede ayudar a administrar el equipo de trabajo con eficacia, de manera que podamos adaptar procedimientos para lograr la máxima calidad.

Una vez que un cliente provee sus requisitos, el equipo se involucra hacia el análisis de requisitos preliminares. El análisis debe abarcar todos los aspectos, especialmente en la forma en que la aplicacion va a unirse al sistema existente. La primer cosa importante es encontrar el público objetivo, entonces, todo el hardware actual, software, personas y datos deben ser considerados durante el tiempo de análisis.

En mi caso me voy a centrar en una pequeña tienda llamada Decorhogar que lleva más de 40 años en su sector y se quiere dar a conocer a través de la web por primera vez. Al ser un cliente de avanzada edad no sabe exactamente lo que quiere, el solo busca darse a conocer y ya que no está relacionado con la tecnología nosotros lo ayudaremos.

En este caso vamos a ver que nuestro cliente solo desea una página informativa dónde pueda mostrar sus trabajos a gente dispuesta a pagar por ellos ofreciendo una gran calidad a la hora de realizar dicha tarea ya que todo lo que realiza es a medida y con un trato muy cercano a su clientela.

Para él cada cliente es único y diferente a los demás.

1.1 Tecnologías

En cuanto al Frontend mi idea es ir a por AngularJs 1.X ya que es una de las tecnologías que mas domino ahora mismo ya que entre el tiempo cursado y las FCT he tocado Angular varios meses el porblema que se me ha planteado es que en un proyecto con tan poca ambición y dónde el cliente me ha cambiado los esquemas de la página varias veces complica las cosas porque por lo que he aprendido lo más difícil de este trabajo es poder satisfacer al cliente por completo ya que el en varias visitas y charlas que he tenido con él cada vez quería una cosa distinta.

Bueno dejando redundancias a un lado vamos a hablar de Angular.

Angular JS es el nombre que recibe el framework de JS creado por empleados de Google, lanzado en Octubre de 2010, el framework se basa en el patrón MVC (Model, View, Controller), con el objetivo de separar las capas de presentación, lógica y componentes de una aplicación.

Se concibió con los siguientes objetivos en mente:

- Desacoplar la manipulación del DOM de la lógica de aplicación
- Desacoplar el lado cliente del lado servidor en una aplicación
- Proveer estructura para el desarrollo de una aplicación

¿Para que se usa AngularJS?

Su principal uso es la creación de SPA's (Single Page Apps), aplicaciones pensadas con el objetivo de lograr la mayor fluidez posible en UX, esto se logra haciendo que la comunicación entre cliente y servidor se realice de forma transparente al usuario, por lo que se logra que este tenga la sensación de no abandonar nunca la pagina principal de la aplicación.

Ahora que sabemos un poco más sobre Angular vamos a ver que ventajas y desventajas nos aporta este framework desarollado por Google

Ventajas

Two way data binding:

A diferencia de la mayoría de sistemas de 'templates' y/o frameworks Angular usa un sistema en el que vista y modelo están en relación constante, se considera el modelo como 'Single-Source-of-Truth'. Gracias a esto, se logra que todo cambio visual, se actualice a tiempo real en el modelo y viceversa, evitando que sea el desarrollador el encargado de lograr la sincronía entre modelo y vista, como es el caso de otros frameworks.

Directivas:

Consisten en marcadores en un elemento de DOM que indican al compilador de Angular que dicho elemento tiene un comportamiento especifico, gracias a esto, se puede trabajar fácilmente a nivel de componentes, siendo estos componentes reutilizables en toda la aplicación.

Supongamos que tenemos una aplicación con usuarios y que estos pueden añadir una foto a su perfil, que se mostrará en toda la aplicación, en caso de no tener dicha foto puesta, habría que cargar una por defecto. Todo ese comportamiento se podría englobar en una directiva a la que se le indicase que usuario se quiere mostrar y esta ya se encargaría de pedir la foto de dicho usuario o poner la predeterminada en caso que el usuario no tuviese ninguna puesta.

Comunidad:

La comunidad de desarrolladores han dado gran soporte a este framework, por lo que hay gran cantidad de módulos ya creados. Esto permite facilitar y agilizar el desarrollo de aplicaciones, pues el desarrollador se puede focalizar en las partes más complejas del desarrollo.

Desventajas

Performance:

Como se ha comentado anteriormente, Angular tiene el denominado ‘two-way-data-binding‘, que permite que las variables del modelo y la vista estén relacionadas y se actualicen en ‘real time’, para lograr esto, pero, hay que evaluar cada variable de las usadas por Angular en cada ‘digest cycle‘ de la aplicación. Esto supone un problema potencial en la ‘performance’ de la aplicación si el desarrollador no lo ha tenido en cuenta y hay estructuras de datos complejas a evaluar.

Inyección de dependencias:

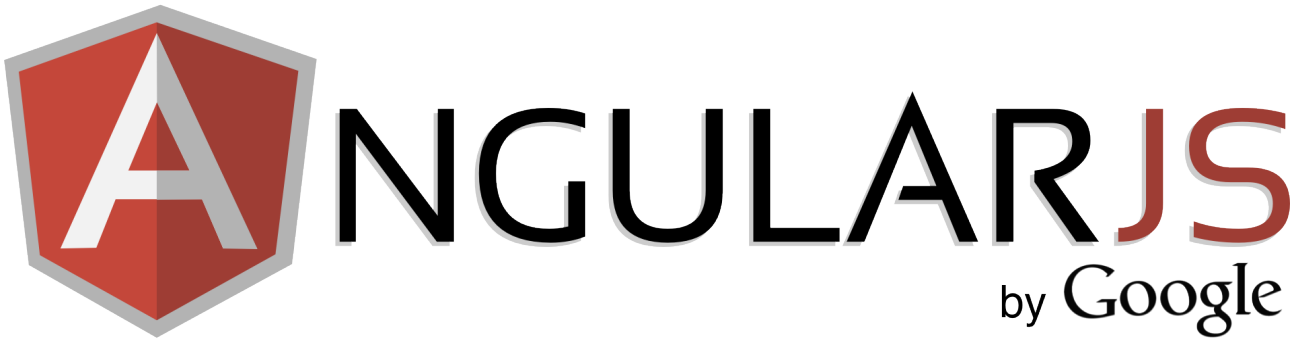
El sistema de inyección de dependencias de Angular carga todos los módulos necesarios al cargar la aplicación, esto supone que incluso si el usuario no accede a ciertas partes de esta, se cargan los módulos necesarios para que dichas partes funcionen. Esto implica un volumen de trafico mayor del necesario, frente a otros modelos, como el implementado por RequireJS, que carga los módulos únicamente cuando estos son necesarios y se van a usar.

Otra cuestión a tratar sobre la inyección de dependencias en Angular, es la anotación que usa, permitiendo tres formas diferentes de anotar las dependencias a un módulo y considerando que una de estas posibles formas, no funcionará si se planea minificar el código de la aplicación

Shared state

Angular usa los llamados ‘scopes‘ como el modelo de la aplicación. Estos se organizan en un árbol con el denominado ‘root scope‘ en la raíz que permite que por defecto cada ‘scope’ hijo herede las propiedades de su ‘scope’ padre y por tanto permite el acceso del hijo hacia el padre. Además todos los ‘scopes’ pueden acceder directamente a ‘root scope‘, lo que implica que por defecto Angular se encuentra bajo estado compartido, eso no es bueno desde el momento en que se pretendan reusar componentes en la aplicación, pues el estado de esta podría influenciar en la creación o aparición de dichos componentes, por eso interesa que la aplicación sea ‘stateless‘.

Por suerte, nos permite crear lo que denomina ‘isolated scope‘ (mediante el uso de directivas), que permiten crear ‘scopes’ propios para esas directivas, de forma que ese este no compartirá estado con ningún otro, de este modo se logra crear un componente que se puede reusar en toda la aplicación.



Ahora nos toca hablar un poco de la segunda parte, el Backend y la tecnología por la que me he decidido, NodeJs.

¿Qué es y para que sirve NodeJS?

Node.js es una librería y entorno de ejecución de E/S dirigida por eventos y por lo tanto asíncrona que se ejecuta sobre el intérprete de JavaScript creado por Google V8. Lo cierto es que está muy de moda aunque no es algo nuevo puesto que existen librerías como Twisted que hacen exactamente lo mismo pero si es cierto que es la primera basada en JavaScript y que tiene un gran rendimiento.

Node.js es un entorno Javascript del lado del servidor, basado en eventos. Node ejecuta javascript utilizando el motor V8, desarrollado por Google para uso de su navegador Chrome. Aprovechando el motor V8 permite a Node proporciona un entorno de ejecución del lado del servidor que compila y ejecuta javascript a velocidades increíbles. El aumento de velocidad es importante debido a que V8 compila Javascript en código de máquina nativo, en lugar de interpretarlo o ejecutarlo como bytecode. Node es de código abierto, y se ejecuta en Mac OS X, Windows y Linux.

Pero ¿porque Javascript del lado del servidor?

Aunque Javascript tradicionalmente ha sido relegado a realizar tareas menores en el navegador, es actualmente un lenguaje de programación totalmente, tan capaz como cualquier otro lenguaje tradicional como C++, Ruby o Java. Ademas Javascript tiene la ventaja de poseer un excelente modelo de eventos, ideal para la programación asíncrona. Javascript también es un lenguaje omnipresente, conocido por millones de desarrolladores. Esto reduce la curva de aprendizaje de Node,js, ya que la mayoría de los desarrolladores no tendrán que aprender un nuevo lenguaje para empezar a construir aplicaciones usando Node.js.

Programación asíncrona, la manera más fácil

Además de la alta velocidad de ejecución de Javascript, la verdadera magia detrás de Node.js es algo que se llama Bucle de Eventos (Event Loop). Para escalar grandes volúmenes de clientes, todas las operaciones intensivas I/O en Node.js se llevan a cabo de forma asíncrona. El enfoque tradicional para generar código asíncrono es engorroso y crea un espacio en memoria no trivial para un gran número de clientes(cada cliente genera un hilo, y el uso de memoria de cada uno se suma). Para evitar esta ineficiencia,así como la dificultad conocida de las aplicaciones basadas en hilos, (programming threaded applications), Node.js mantiene un event loop que gestiona todas las operaciones asíncronas.

¿Qué problema resuelve Node?

¿Cuál es el problema con los programas de servidor actuales? Hagamos cuentas.En lenguajes como Java™ y PHP, cada conexión genera un nuevo hilo que potencialmente viene acompañado de 2 MB de memoria. En un sistema que tiene 8 GB de RAM, esto da un número máximo teórico de conexiones concurrentes de cerca de 4.000 usuarios. A medida que crece su base de clientes, si usted desea que su aplicación soporte más usuarios, necesitará agregar más y más servidores. Por todas estas razones, el cuello de botella en toda la arquitectura de aplicación Web (incluyendo el rendimiento del tráfico, la velocidad de procesador y la velocidad de memoria) era el número máximo de conexiones concurrentes que podía manejar un servidor.

Node resuelve este problema cambiando la forma en que se realiza una conexión con el servidor. En lugar de generar un nuevo hilo de OS para cada conexión (y de asignarle la memoria acompañante), cada conexión dispara una ejecución de evento dentro del proceso del motor de Node. Node también afirma que nunca se quedará en punto muerto, porque no se permiten bloqueos y porque no se bloquea directamente para llamados E/S.

Desventajas de node.js

Como la mayoría de las nuevas tecnologías Node no es fácil de implementar en alojamientos existentes. En alojamientos compartidos habrá que ver si una aplicación de node.js puede funcionar; esto dependerá de la empresa de hosting. Si el alojamiento está en un VPS o servidor dedicado será más sencillo ya que se pueden ejecutar aplicaciones Nodejs sin problemas.Lo más fácil es usar un servicio escalable como Heroku, que es completamente gratuito para desarrollar una web y solo habrá que pagar cuando se necesiten más recursos.

Otras desventajas que se mencionan a menudo (pero que están solucionándose día a día son:

API Inestable: La API de Node tiene la mala costumbre de cambiar en formas que rompen la compatibilidad hacia atrás de versión en versión, lo que requiere que apliques cambios frecuentes en tu código para mantener todo funcionando en las versiones mas actuales.

Falta de una Librería Estándar: JavaScript es un lenguajes con un buen núcleo pero con una flaca librería estándar. Cosas que darías por hecho en otro lenguaje del lado del servidor simplemente no existen.

Falta de Librerías en General: ¿Necesitas una interfaz de bases de datos madura? ¿Un ORM? ¿Una librería de procesamiento de imágenes? ¿Un analizador XML? Como JavaScript no sido popular en el lado del servidor todo esto es muy reciente, o no está probado o está en camino.

Muchas Formas de Programar: La falta inherente de organización de código se puede considerar una gran desventaja. Se nota su efecto claramente cuando el equipo de desarrollo no está muy familiarizado con la programación asíncrona o los patrones de diseño estándar. Simplemente hay demasiadas formas de programar y de obtener código desparejo y difícil de mantener.

No está Probado lo Suficiente. Este punto puede ser susceptible a opiniones subjetivas debido a que es una cuestión bastante abierta. Mientras no tengamos grandes proyectos en producción por varios años, no podremos saber donde está el problema.

Lo más importante de node.js: La comunidad

Ademas de sus aplicaciones innatas, Node.js tiene una aplicaciones Node.js que ha escrito muchos módulos, excelente para agregar capacidades adicionales a las aplicaciones Node.js. Uno de los mas famosos es Socket.io, un módulo para gestionar las conexiones persistentes entre el cliente y el sevidor, permitiendo al servidor enviar actualizaciones en tiempo real a los clientes.(que es la magia que mantiene a nuestra plataforma de GPS y seguimiento en línea).

Como hemos podido comprobar la fuerza de estos dos frameworks es la gran comunidad que hay detras de cada uno y cada vez más y más grande. Este es uno de los principales motivos por los que elegí estas tecnologías.


node
JS

2. Frontend

Ahora vamos a profundizar un poco más en el Frontend, como he dicho anteriormente el frontend lo he hecho con Angular y como os he comentado hay una gran comunidad detrás de este frame por lo que incluso hay programadores especializados en Angular acreditados por Google que han creado su propia guía de estilo para programar como sería Todd Motto o John Papa, yo me he decidido a seguir la guía de estilo de John Papa.

Ahora que ya sabemos que pautas seguir ¿porque no conocemos un poco a este hombre?

John Papa

Es un experto desarrollador Google y Director Regional de Microsoft especializado en tecnologías web. A menudo suele dar charlas y formar en varios eventos alrededor del mundo en conferencias magistrales, talleres y sesiones para eventos tales como Build , ngConf , AngleBrackets , TechEd / Ignite, y VSLive .

Participo en un programa de televisión llamado Silverlight en el Canal 9 y sede de muchos eventos, incluyendo el mezclador y Open Source Fest en las principales conferencias.

Se puede contactar con él johnpapa.net o en Twitter en @john_papa . Además de estar enterado de las últimas noticias sobre tecnología.

Ahora que sabemos un poco quién es John Papa vamos a hablar sobre la guia de estilo que voy a seguir a la hora de realizar este proyecto.

En este enlace podréis encontrar la guia entera, yo voy a abordar los apartados más relevantes que me parezcan convenientes https://github.com/johnpapa/angular-styleguide/blob/master/a1/i18n/es-ES.md .

Un componente básico en el que se centra john papa es **la regla del 1 o reponsabilidad única**.