AureliaJS

Jesse Javier Cogollo Alvarez

Developer by passion

email: cogollo87@gmail.com

MedellinJS

August 14, 2015

Contenido

Que es Aurelia?

Aurelia is a next generation JavaScript client framework that leverages simple conventions to empower your creativity.
http://aurelia.io/

Rob Eisenberg
robeisenberg.com

Caracteristicas

1. **Forward-thinking**

Escrito con ES6 y ES7. se integra con web components. no tiene dependencias externas (excepto polyfills).

Caracteristicas

1. **Forward-thinking**
2. **Modern Architecture**

Aurelia esta compuesto por pequenos modulos. Utiliados juntos como un completo framework. Pero tambien permite personalizarlo con diferentes modulos.

Caracteristicas

1. **Forward-thinking**
2. **Modern Architecture**
3. **Two-Way Databinding**

Aurelia tiene una poderosa two-way binding. el cual utiliza tecnicas adaptativas para utilizar el mecanismo mas eficiente para observar cada propiedad. Object.observer

Caracteristicas

1. **Forward-thinking**
2. **Modern Architecture**
3. **Two-Way Databinding**
4. **Extensible HTML**

La extensibilidad HTML de Aurelia permite crear elementos HTML personalizados, anadir nuevos comportamientos para elementos ya existentes.

Caracteristicas

1. **Forward-thinking**
2. **Modern Architecture**
3. **Two-Way Databinding**
4. **Extensible HTML**
5. **Routing and UI Composition**

Router con acoplamiento activo, patrones de ruta dinamica, activacion de pantallas asincronicamente, etc.

Caracteristicas

1. **Forward-thinking**
2. **Modern Architecture**
3. **Two-Way Databinding**
4. **Extensible HTML**
5. **Routing and UI Composition**
6. **MV* with Conventions**

Quieres invertir tiempo en escribir codigo de configuracion? Aurelia maneja una convecion simple.

Caracteristicas

1. **Forward-thinking**
2. **Modern Architecture**
3. **Two-Way Databinding**
4. **Extensible HTML**
5. **Routing and UI Composition**
6. **MV* with Conventions**
7. **Broad Language Support**

Utiliza ES5, ES6, TypeScript, AtScript, CoffeeScript.

Caracteristicas

1. **Forward-thinking**
2. **Modern Architecture**
3. **Two-Way Databinding**
4. **Extensible HTML**
5. **Routing and UI Composition**
6. **MV* with Conventions**
7. **Broad Language Support**
8. **Testable**

combinando modulos ES6 con contenedor inyeccion de dependencias. se hace mas facil crear codigo altamente cohesivo y bajo acoplamiento. haciendo las pruebas unitarias algo facil.

Como empezar - Estructura

Descargamos la ultima version del skeleton de aurelia.

https://github.com/aurelia/skeleton-navigation/releases/tag/0.15.1

descomprimimos y ya tenemos una estrutura lista para utilizar:

- build (configuracion gulp).
- doc (documentacion).
- src (codificacion).
- styles (estilos CSS).
- test (pruebas).
- README.md (instrucciones para utilizar AureliaJS).

Modulos

- **logging**

Esta libreria es parte de la plataforma aurelia y contiene un "Appender" para el logging.

uso: welcome.js

Modulos

- **Plugins**

Aurelia permite crear e implementar plugins que se adaptan a Aurelia. Estos plugins se configuran en: implementacin: index.html uso: animation-main.js

Vistas y vistas modelos

- **DI inyeccion de dependencias**

Los AureliaElements son creados como clases los cuales son instaciados por el framework usando contenedor de inyeccion de dependencias.

uso: flickr.js

Vistas y vistas modelos

- **Plantillas**

El motor de plantillas de aurelia es el encargado de cargar las vistas y sus recursos, compilando su HTML para un optimo desempeno y renderizando el UI en la pantalla.

¡template¿ ¡/template¿. Todo lo que este adentro de esas etiquetas sera administrado por aurelia.

uso: welcome.html

Databiding

- **Que es**

El Databinding nos permite enlazar el estado y comportamiento en un objeto(js) y una vista(html).

Cualquier cambio es enlazado y/o sincronizado en una o varias direcciones. Se puede diferenciar en la vista por que el elemento va seguido de un punto en el atributo con el valor (bind, one-way, two-way o one-time).

implementacin: welcome.html

Bindings

- **delegate, trigger and call**

Los bindings no solamente conectan elementos o atributos. Tambien son utilizados para lanzar comportamientos. Se recomienda utilizar el delegate por defecto por que es el mas eficiente al utilizar la memoria y la cpu.

Implementation: TODO

Bindings

- **string interpolation**

Sintaxis: $nameVariable. es un binding de un solo camino, la cual la salida se convierte en un string.

Implementacion: welcome.html

Bindings

- **ref**

Es especial para conectar elementos personalizados, utilizando esta tecnica puedes conectar diferentes componentes.

Implementacion: TODO

Bindings

- **select elements**

"value.bind" es un HTMLSelectElement que tiene un comportamiento especial para soportar seleccion simple y seleccion multiple de valores. normalmente se combina con el elemento repeat. Pero existe otro elemento especial que permite trabajar con objetos "model.bind".

Implementacion: TODO

Bindings

- **radios**

"checked.bind" es un HTMLInputElement tiene un comportamiento especial para soportar binding no booleano, como strings y objetos.

Implementacion: TODO

Bindings

- **checkboxes**

Implementacion: TODO

Bindings

- **InnerHTML**

podemos hacer binding de los elementos innerHTMl con el atributo innerhtml.

Implementacion: TODO

Bindings

- **textContent**

Podemos hacer binding de los elementos textContect con el atributo textcontent.

Implementacion: TODO

Bindings

- **style**

Podemos hacer binding a los string css u objetos en un elemento style.

Implementacion: TODO

Bindings

- **adaptive binding**

Implementacion: TODO
Important

HTML Extensions

- **show**

Nos permite condicionar si un elemento o elementos seran visibles o no en el DOM
Implementacion: TODO

HTML Extensions

- **if**

Tiene el mismo comportamiento que el "show". lo que lo hace diferente es que esta etiqueta si remueve el componente del DOM.

Implementacion: TODO
Important

HTML Extensions

- **repeat**

Este atributo permite renderizar un template multiples veces y trabaja en conjunto con el ".for"

Implementacion: TODO
Important

HTML Extensions

- **repeat $parent**

permite acceder a las propiedades de la viewModel.

Implementacion: TODO
Important

HTML Extensions

- **repeat $index**

Es el index del item en el array

Implementacion: TODO
Important

HTML Extensions

- **repeat $first**

es verdadero si es el primer item del array

Implementacion: TODO
Important

HTML Extensions

- **repeat $last**

es verdadero si es el ultimo item del array.

Implementacion: TODO
Important

HTML Extensions

- **repeat $even**

es verdadero si el item es un numero par

Implementacion: TODO
Important

HTML Extensions

- **repeat $odd**

es verdadero si el item es un numero impar

Implementacion: TODO
Important

Routing

- **descripcion**

Cuando implementas enrutadores, tienes varias posibilidades de implementarlo, como: navegacion de la App, dashboards y MDI interfaces.

El Routing de Aurelia, el Router que vive en los viewModels y los router-view que viven en las vistas.

Routing

- **que podemos hacer**

rutas estaticas: /contactUs

rutas parametriables:
/user/:userId/dasboard

rutas wilcard: file*path

Routing

- **Ciclo de vida**

Existen 4 estados:

canActivate: permite validar si se puede navegar en el viewModel.

activate: permite implementar logica antes del que el viewModel sea cargado.

canDeactivate: permite controlar si deseo que en ese momento se pueda abandonar el viewModel.

deactivate: permite implementar logica una vez el viewModel ya fue cambiado.

Routing

- **Convencion de enrutadores**

Implementacion: TODO
Important

Routing

- **Personalizando el enrutador**

Aurelia nos permite agregar pasos en el proceso del Router, los que nos permite agregar validaciones de permisos.

Implementacion: TODO Important!!!

Routing

- **pushState**

si asi lo prefieres, puedes remover el hash de la url.

Implementacion: TODO Important!!!

Routing

- **reutilizando un viewModel existente**

si alguna vez usted quiere implementar la misma vista para diferentes rutas.

Implementacion: TODO
Important

Routing

- **renderizando multiples viewPorts**

si alguna vez necesitas renderizar contenido en mas de un area con la misma ruta.

Implementacion: TODO
Important !!!

Routing

- **Rutas personalizadas**

Aurelia nos permite generar Routes personalizados.

Implementacion: TODO Important!!!

Extendiendo HTML

- **TODO**

Implementacion: TODO
Important

Redes sociales

1. **Meetup**

/MongoDB-Medellin

`http://goo.gl/fw5Gyh`

Redes sociales

1. Meetup
2. **Twitter**

@jessecogollo

http://goo.gl/gdCAjF

Redes sociales

1. Meetup
2. Twitter
3. **Facebook**

/jessecogollo

http://goo.gl/Q1JnXQ

Redes sociales

1. Meetup
2. Twitter
3. Facebook
4. Google Plus
5. **Lista de correo**

+ correo

`http://goo.gl/FJvrjT`

Redes sociales

1. Meetup
2. Twitter
3. Facebook
4. Google Plus
5. Lista de correo
6. **Grupo de estudio**

Formulario grupo de estudio

`http://goo.gl/7ALdst`

Donde aprender

1. **Organizar un coding dojo**

http://johannesbrodwall.com/2011/12/18/how-to-start-a-coding-dojo/

Donde aprender

1. Organizar un coding dojo
2. **Cyber dojo**

http://cyber-dojo.org/

Donde aprender

1. Organizar un coding dojo
2. Cyber dojo
3. **Codingdojo**

http://www.codingdojo.org/

Donde aprender

1. Organizar un coding dojo
2. Cyber dojo
3. Codingdojo
4. **Ejercicios**

http://rosettacode.org/wiki/Category:Programming_Tasks

Donde aprender

1. Organizar un coding dojo
2. Cyber dojo
3. Codingdojo
4. Ejercicios
5. **Mas ejercicios**

http://brendan.enrick.com/post/Coding-Katas-and-Exercises

Preguntas

Empecemos...

Gracias !!! =)