

AureliaJS

Jesse Javier Cogollo Alvarez

Developer by passion

email: cogollo87@gmail.com

MedellinJS

August 11, 2015

Contenido

AureliaJS

DEMO

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

Características

1. Forward-thinking

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

Características

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.

Características

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

Características

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

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

Características

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 conveccion simple.

Características

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.

Características

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>

descomprimos y ya tenemos una estructura 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: `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

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.

- **Plantillas**

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

uso: `welcome.html`

Databinding

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

- **Que es**

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 `textContent` 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**

Implementacion: TODO
Important

HTML Extensions

- **if**

Implementacion: TODO
Important

HTML Extensions

- **repeat**

Implementacion: TODO
Important

HTML Extensions

- **\$parent**

Implementacion: TODO
Important

HTML Extensions

- **\$index**

Implementacion: TODO
Important

HTML Extensions

- **\$first**

Implementacion: TODO
Important

HTML Extensions

- **\$last**

Implementacion: TODO
Important

HTML Extensions

- **\$even**

Implementacion: TODO
Important

HTML Extensions

- **\$odd**

Implementacion: TODO
Important

HTML Extensions

- **compose**

Implementacion: TODO
Important

HTML Extensions

- **\$global behavior**

Implementacion: TODO
Important

Routing

- **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

[/jessecogollo](#)

3. **Facebook**

<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-dojos/>

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

Preguntas

Empecemos...

Gracias !!! =)