Directivas propias

Fundamentos

- Como en los controladores, las directivas se registran en los módulos

Fundamentos

- Para registrar una directiva se utiliza la API **module.directive**

```
myModule.directive('greeting', function() {...});
```

Fundamentos

- Una directiva es un nuevo markup al que se le asocia un comportamiento
- Está asociado a un elemento del DOM
- Puede llevar asociado un template Angular

Fundamentos

```
<body ng-app="myApp">
    <div ng-controller="greetingCtrl">
        <greeting></greeting>
    </div>
</body>
```

Fundamentos

```
var myApp = angular.module('myApp', []);
myApp.controller('greetingCtrl', function ($scope){
    $scope.greeting = function(){ $scope.greeting =
'Hola!!! Waaasssup?';}
});
```

Fundamentos

```
myApp.directive('greeting', function() {
    return {
        restrict: 'E',
        template:  '<button type="submit" class="btn
        btn-default" ng-click="greetMe()"> Greet Me
        </button> {{ greeting }}'
    };
});
```

Restrict

- Cuando se crea una directiva, esta se limita a atributos y elementos por defecto
- Para modificar esto, se debe utilizar la opción ‘**restrict**’

Restrict

- La opción ‘**restrict**’ se establece en:
  - '**A**' - sólo coincide con el nombre de atributo
  - '**E**' - sólo coincide con nombre de elemento
  - '**C**' - sólo coincide con el nombre de clase

Restrict

- Estas restricciones se pueden combinar según sea necesario
- '**AEC**' - coincide con cualquiera de atributo o elemento o nombre de la clase

Ejercicios

- Utilizando "gna-como-directiva.html" crea una app Angular que esté compuesta de una directiva **gna** creada por ti, con restrict '**A**', que agrupe el código html que tenías hasta ahora

Scope

- El scope de una directiva funciona exáctamente igual que como hasta ahora
- Hereda el scope de la directiva o controlador “padres” a los que pertenece

Scope

```
<div ng-controller="gnaCtrl"><gna-dir></gna-dir></div>

gna.controller('gnaCtrl', function ($scope) {  $scope.
  randomNumber = generateRandomNumber(); });

gna.directive('gnaDir', function() { return { restrict: 'E',
  template:  '{{randomNumber}}' }; });
```

Scope

- ¿Y si queremos que nuestra directiva tenga su propio scope y no que lo herede de nadie?

Isolated Scope

Isolated scope

- El ámbito aislado de la directiva aísla todo excepto los modelos que se añaden explícitamente al **scope: {}**

Isolated scope

```
gna.directive('gnaDir', function() {
    return {
        scope: {},
        restrict: 'E',
        template:  '{{randomNumber}}'
    };
});
```

Isolated scope

- Úsalo para crear componentes reutilizables
- Evita que un componente cambie el estado de su modelo, salvo los modelos que se especifican de forma explícita

Isolated scope

- La directiva está ahora aislada por completo del ámbito de aplicación de los padres
- Cómo le pasa datos a la directiva?

Isolated scope

- Utilizamos los caracteres @, =, y **&**
- **@:** Se usa para acceder a valores definidos fuera de la directiva de tipo string

Isolated scope (@)

```
<div ng-controller="gnaCtrl">
    <gna-dir current="{{randomNumer}}"></gna-dir>
</div>

gna.directive('gnaDir', function() { return {
        scope: { randomNumber: '@current'},
        restrict: 'E', template:  '{{randomNumber}}'
    };
});
```

Isolated scope (@)

- “**One way data binding**”
- Si el controlador cambia, la directiva también
- Pero si la directiva cambia el controlador no

Isolated scope (=)

- “**2 way data binding**”
- Unión entre el scope exterior y el scope aislado de la directiva

Isolated scope (=)

```
<div ng-controller="gnaCtrl">
    <gna-dir current="{{randomNumer}}"></gna-dir>
</div>

gna.directive('gnaDir', function() { return {
        scope: { randomNumber: '=current'},
        restrict: 'E', template:  '{{randomNumber}}'
    };
});
```

Isolated scope (&)

- En los casos en que es necesario pasar como parámetro una función
- Vía de unión entre el scope exterior y el scope aislado de la directiva

Isolated scope (&)

- **Atributo función** generateRandomNumber()

```
<div ng-controller="gnaCtrl">
    <gna-dir mod="{{mod}}"
             generate="generateRandomNumber()"></gna-dir>
</div>
```

Isolated scope (&)

```
gna.directive('gnaDir', function() { return {
        scope: { mod: '=', action: '&generate' },
        restrict: 'E',
        template: '<button ng-click="action()">Generate ' +
                  '</button> {{randomNumber}} '
    };
});
```

Compile, controller & link

- Son 3 tipos de funciones que se pueden declarar en una directiva

Compile, controller & link

- Cada una se ejecuta en un momento determinado, en un orden determinado y sirven para diferentes propósitos

Directivas propias: compile

- Es la que se ejecuta primero de las 3
- Se llama sólo una vez por cada declaración de la directiva

Directivas propias: compile

- En la fase de compilación Angular devuelve el template
- Para manipular el DOM original, antes de que Angular cree una instancia de él y del scope

Directivas propias: controller

- Los controladores pueden ser compartidos/instanciados por otras directivas a través de la palabra “**require**”

Directivas propias: controller

- Dicen los de Angular:
  - Buena práctica: utiliza “**controller**” para exponer una **API** para otras directivas. De lo contrario usar “**link**”.

Directivas propias: link

- Utiliza link para manipular el **DOM**
- function `link(scope, element, attrs){ ... }` donde:
  - `scope` es un objeto $scope Angular
  - `element` es el elemento ‘jqLite-incrustado’

Directivas propias: link

- function `link(scope, element, attrs){ ... }` donde:
  - `attrs` es un objeto hash con pares clave-valor de los nombres de los atributos y sus correspondientes valores

Directivas propias: link

```
<body ng-app="myApp">
    <greeting></greeting>
</body>
```

Directivas propias: link

```
myApp.directive('greeting', function() {
    return {
        restrict: 'E',
        template: '<button type="submit" class="btn btn-default"' +
        'ng-click="greetMe()"> Greet Me </button> {{ greeting }}',
        link: function (scope, element, attrs) {
            scope.greetMe = function(){ scope.greeting = 'Hola!!!
             +  Waaasssup?'; }
        }
}});
```

Ejercicios

- Utiliza la función link, para poder asociar la lógica de generar números aleatorios a la directiva, sin necesidad del controlador

Función link: attrs

- Desde la directiva podemos acceder a los **atributos del markup** que hemos definido

```
<body ng-app="myApp">
    <greeting name="Dani"></greeting>
</body>
```

Función link: attrs

```
myApp.directive('greeting', function() {
    return {
        restrict: 'E',
        template:  '<button type="submit" class="btn btn-default"
        ng-click="greetMe()"> Greet Me </button> {{ greeting }}',
        link: function (scope, element, attrs) {
            scope.greetMe = function(){ scope.greeting = 'Hola!!!'
        + attrs.name + 'Waaasssup?'; }
        }
}});
```

Ejercicios

- Pásale a tu gna a través de un atributo, el intervalo de tiempo en el que quieres que se genere un nuevo número

Función link: element

- **jqLite** es un subconjunto compatible con la **API de jQuery** que provee Angular para manipular el DOM de una manera compatible con todos los navegadores

Función link: element

- jqLite implementa sólo las funciones más utilizadas.
- https://docs.angularjs.org/api/ng/function/angular.element

Función link: element

- Envuelve ("**wrappes**") un elemento **DOM** o un string **HTML** como elemento jQuery.
- Si jQuery está disponible, angular.element es un alias para la función jQuery.

Función link: element

- Si no, angular.element delega al subconjunto integrado de jQuery de Angular "lite jQuery" o "**jqLite**."

Ejercicios

- Utiliza JQLite para cambiar lo que te de la gana del DOM cuando el número aleatorio generado sea par.

Ejercicios

- Utiliza lo que sabes para poder parar/pausar la generación de números.