Buenas Prácticas de AngularJS Directives

Limit 1 per file

Crear una directiva por archivo. Nombrar el archivo con el nombre de la directiva.

Por qué?: Si bien es fácil declarar todas nuestras directivas en un solo archivo, su uso se complica a future si queremos agregar algunas directivas específicas en módulos u otros componentes particulares.

Por qué?: Una directiva por archivo se más fácil de mantener.

```
/* avoid */
/* directives.js */

angular
    .module('app.widgets')

    /* order directive that is specific to the order module */
    .directive('orderCalendarRange', orderCalendarRange)

    /* sales directive that can be used anywhere across the sales app */
    .directive('salesCustomerInfo', salesCustomerInfo)

    /* spinner directive that can be used anywhere across apps */
    .directive('sharedSpinner', sharedSpinner);


function orderCalendarRange() {
    /* implementation details */
}

function salesCustomerInfo() {
    /* implementation details */
}

function sharedSpinner() {
    /* implementation details */
}
```

```
/* recommended */
/* calendarRange.directive.js */

/**
* @desc order directive that is specific to the order module at a company named Acme
* @example <div acme-order-calendar-range></div>
*/
angular
    .module('sales.order')
    .directive('acmeOrderCalendarRange', orderCalendarRange);

function orderCalendarRange() {
    /* implementation details */
}


/* recommended */
/* customerInfo.directive.js */

/**
* @desc spinner directive that can be used anywhere across the sales app at a company named Acme
* @example <div acme-sales-customer-info></div>
*/
angular
    .module('sales.widgets')
    .directive('acmeSalesCustomerInfo', salesCustomerInfo);

function salesCustomerInfo() {
    /* implementation details */
}
```

```
/* recommended */
/* spinner.directive.js */

/**
* @desc spinner directive that can be used anywhere across apps at a company named Acme
* @example <div acme-shared-spinner></div>
*/
angular
    .module('shared.widgets')
    .directive('acmeSharedSpinner', sharedSpinner);

function sharedSpinner() {
    /* implementation details */
}
```

Nota: Hay varias formas de nombrar a nuestras directivas, debido a que pueden ser usadas en aplicaciones de distintos tamaños y arquitecturas. Seleccione el nombre que haga a su directiva fácil de ser distinguida y que sea lo más claro posible.

Manipulate DOM in a Directive

Cuando quiera manipular elementos del DOM directamente, use una directiva. Si tiene distintas alternativas tales como estilos CSS, servicios de animaciones, templates de Angular, ngHide o ngShow, uselos. Por ejemplo si nuestra directiva simplemente se oculta y se muestra, use ngHide/ngShow.

Por qué?: Las manipulaciones del DOM pueden ser difíciles de testear, debuggear y siempre hay mejores formas (CSS, animaciones, templates).

Provide a Unique Directive Prefix

Utilice para su directiva prefijos cortos, únicos y descriptivos tales como acmeSalesCustomerInfo que será declarado en el HTML como acme-sales-customer-info.

Por qué?: Prefijos cortos y únicos identifican el origen y contexto de nuestra directiva. Por ejemplo un prefijo cc- puede indicar que la directiva es parte de la aplicación CodeCamper mientras que acme- puede indicar que la directiva es de la compañía Acme.

Nota: Evite usar el prefijo ng- ya que el mismo es reservado para directivas nativas de AngularJS. Deberá hacer una pequeña búsqueda o investigación del Mercado para no utilizer prefijos usados por distintos frameworks que puedan traer conflictos como por ejemplo el prefijo ion- utilizado por Ionic.

Restrict to Elements and Attributes

Cuando cree directivas que tengan sendito viviendo por sí solas como un element aislado, utilice restrict E, si va a estar extendiendo una funcionalidad existente, cree su directiva como un atributo que podrá ser aplicado a cualquier element HTML para lo cual deberá usar restrict A.

Por qué?: Tiene mucho sentido.

Por qué?: Si bien podemos declarar nuestra directiva como una clase, si la misma está realmente actuando como un elemento tiene más sentido declararla como elemento o al menos como atributo.

Nota: EA es el comportamiento por defecto para AngularJS 1.3 +

```
<!-- avoid -->
<div class="my-calendar-range"></div>
/* avoid */
angular
    .module('app.widgets')
    .directive('myCalendarRange', myCalendarRange);

function myCalendarRange() {
    var directive = {
        link: link,
        templateUrl: '/template/is/located/here.html',
        restrict: 'C'
    };
    return directive;

    function link(scope, element, attrs) {
      /* */
    }
}
```

```
<!-- recommended -->
<my-calendar-range></my-calendar-range>
<div my-calendar-range></div>
/* recommended */
angular
    .module('app.widgets')
    .directive('myCalendarRange', myCalendarRange);

function myCalendarRange() {
    var directive = {
        link: link,
        templateUrl: '/template/is/located/here.html',
        restrict: 'EA'
    };
    return directive;

    function link(scope, element, attrs) {
      /* */
    }
}
```

Directives and ControllerAs

Use la sintaxis controller as con sus directivas para ser consitente con la misma sintaxis utilizada en controladores y vistas.

Por qué?: Tiene más sentido y no resulta complicado.

Nota: La directiva que se muestra a continuación demuestra alguna de las formas de utilizar scope dentro de la función link en combinacion con controladores usando la sintaxis controller as, como así también el template inline para tener todo en un sólo lugar.

```
<div my-example max="77"></div>
angular
    .module('app')
    .directive('myExample', myExample);

function myExample() {
    var directive = {
        restrict: 'EA',
        templateUrl: 'app/feature/example.directive.html',
        scope: {
            max: '='
        },
        link: linkFunc,
        controller : ExampleController,
        controllerAs: 'vm'
    };

    return directive;

    function linkFunc(scope, el, attr, ctrl) {
        console.log('LINK: scope.max = %i', scope.max);
        console.log('LINK: scope.vm.min = %i', scope.vm.min);
        console.log('LINK: scope.vm.max = %i', scope.vm.max);
    }
}
```

```
ExampleController.$inject = ['$scope'];

function ExampleController($scope) {
    // Injecting $scope just for comparison
    var vm = this;

    vm.min = 3;
    vm.max = $scope.max;
    console.log('CTRL: $scope.max = %i', $scope.max);
    console.log('CTRL: vm.min = %i', vm.min);
    console.log('CTRL: vm.max = %i', vm.max);
}
/* example.directive.html */
<div>hello world</div>
<div>max={{vm.max}}<input ng-model="vm.max"/></div>
<div>min={{vm.min}}<input ng-model="vm.min"/></div>
```