Creando directivas en Angular

Anxo Fole
afole@plainconcepts.com

Para qué sirven las directivas

- Extienden el lenguaje de las vistas (HTML)
- Añaden elementos o atributos que encapsulan plantillas y/o comportamiento
  - De forma declarativa
- Son reutilizables ☺

En vez de esto:

```
<div class="alert alert-warning alert-dismissable">
    <button type="button" class="close" data-dismiss="alert">&times;</button>
    <strong>Warning!</strong> This is the Bootstrap alert
</div>
```

Podemos escribir esto:

```
<alert type="warning">
    <strong>Warning!</strong> This is the AngularJs version
</alert>
```

Separación de responsabilidades

- Único componente de AngularJS en el que "*está permitido*" interactuar con el DOM

- No hacerlo en:
  - Controladores
  - Servicios

Registro de directivas

```
angular.module(‘moduleName’).directive('directiveName', function factory(injectables) {
    var directiveDefinitionObject = {
        ...
    };
    return directiveDefinitionObject;
 });
```

Normalización de nombres

Nombre normalizado de la directiva

```
app.directive('ngBind', function () {
    ...
});
```

Puede apararecer en el HTML como

- ng-bind
- ng:bind
- ng_bind
- data-ng-bind
- x-ng-bind

Usar prefijos:

- Para no crear elementos o atributos que puedan colisionar con el estándar HTML en el futuro
- No usar ***ng*** para evitar colisionar con futuras directivas de angular

Directive Definition Object

```
var directiveDefinitionObject = {
    priority: 0,
    terminal: true,
    template: '<div></div>',
    transclude: false,
    restrict: 'A',
    templateNamespace: 'html',
    scope: false,
    controller: function($scope, $element, $attrs, $transclude, otherInjectables) { },
    controllerAs: 'stringAlias',
    require: 'siblingDirectiveName',
    compile: function compile(tElement, tAttrs, transclude) {
        return {
            pre: function preLink(scope, iElement, iAttrs, controller) { },
            post: function postLink(scope, iElement, iAttrs, controller) { }
    },
    // or
    // link: function postLink(scope, iElement, iAttrs, controller) { ... }
};
```

Mi directiva utiliza una plantilla

template: string | function(tElements, tAttributes)

templateUrl: string | function(tElements, tAttributes)

templateNamespace: string [html | svg | math]

Funciones

- Se ejecuta antes de cualquier otra función del *DDO*
- **tElements** incluye nodos HTML (jQLite) con atributos sin normalizar
- **tAttributes** incluye atributos normalizados
- Valores de atributos no interpolados (no hay scope todavía)

Quiero restringir cómo se usa

restrict: [EACM]

- E (nombres) `<my-directive></my-directive>`
- A (atributos) `<div my-directive="exp"></div>`
- C (nombres de clases) `<div class="my-directive: exp;"></div>`
- M (comentarios) `<!-- directive: my-directive exp -->`

Se pueden combinar
Por defecto: ***EA (la directiva reconoce nombres o atributos)***

¿Elemento o atributo?

Sobre un elemento solo puede haber una directiva que defina una plantilla

La recomendación oficial:

- Usa ***E*** cuando la directiva defina una template
- Usa ***A*** cuando la directiva no defina template

It’s up to you... Pero piensa cómo quieres que se use

¿Cual se ejecuta primero?

priority: int

terminal: bool

- Varias directivas sobre el mismo elemento
- Ejecución por orden de prioridad descendente
  - Si no se define = 0
- Si *terminal === true* no se procesan directivas de menor prioridad en el elemento

Necesito comunicarme con el exterior

scope: bool | object

Si es true:

- Crea un nuevo scope desde su scope padre

Si es object:

- Isolated scope desde rootScope

  @ -> bindea al valor del atributo (string)

  = -> bindeo bidireccional entre propiedad del scope y la del padre

  & -> ejecuta una expresión en el contexto del scope del padre

Contenido personalizable en la template

transclude: bool

- Permite definir contenido en la vista que será inyectado ***dentro*** de la plantilla, tiene acceso al scope de fuera

alert.html

```
<div class="alert">
    <button type="button" class="close">&times;</button>
    <div ng-transclude></div>
</div>
```

directiva

```
app.directive('alert', function () {
        return {
            restrict: 'EA',
            templateUrl: 'alert.html',
            transclude: true,
            replace: true,
        };
    });
```

uso de la directiva

```
<alert type="warning">
    <strong>Warning!</strong> This is the AngularJs version
</alert>
```

Manipulación del DOM

link: function(scope, iElements, iAttrs, ctrl, transclude) | {pre,post}

Podemos manipular el DOM y suscribirnos a eventos
Los valores de los atributos está ya interpolados

- scope.$apply
  - Fuerza la ejecución de un ciclo de $digest
- scope.$watch
  - Nos permite reaccionar ante cambios en el scope
- attrs.$observe
  - Observa las modificaciones de un atributo interpolado

¡Necesito un controlador!

controller: string | function(scope, elements, attributes, transclude, …)

Instanciado antes de la fase de pre-link

Dos razones:

- Función **link** muy compleja que queremos refactorizar y hacer más fácilmente testable
- Definir un API para comunicación entre directivas

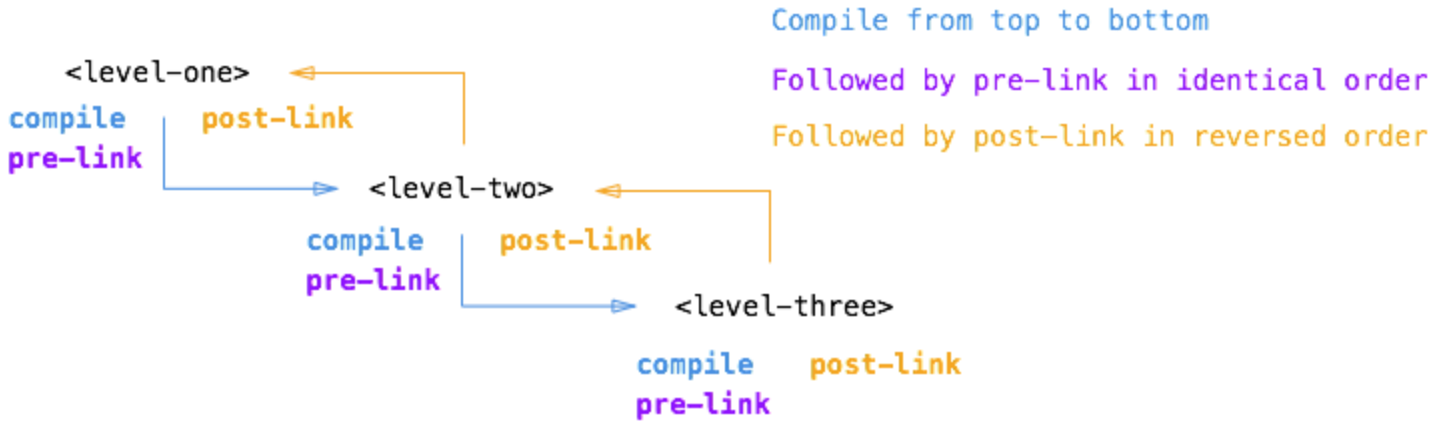
require: string [(?^)controlerName(s)]

compile

compile: function(tElement, tAttributes, transclude)

Realiza modificaciones en el DOM de la plantilla antes de clonarla

Poco usada porque pocas directivas la necesitan (**ngRepeat**)



¿Compilar HTML?

- Proceso de identificar directivas y aplicar sus comportamientos y transformaciones
- Por cada directiva se ejecutan dos fases
  - **compile**: Genera la plantilla para el nodo y devuelve una función de link
  - **link**: Aplica el scope a la plantilla