Eventos y Routing

Anxo Fole
afole@plainconcepts.com

Notificaciones

Suscripción de eventos

- Suscripción a eventos:
  - $scope.$on('eventName', function(event, args…){})
    - event.preventDefault para cancelar el comportamiento por defecto del evento
    - event.stopPropagation para cancelar la propagación
- El resultado de suscribirse a un evento es una función para dessuscribirse
  - Atención a las fugas de memoria
- Se puede utilizar el evento **destroy** de un controlador para dessuscribirse a todos los eventos suscritos en el controlador

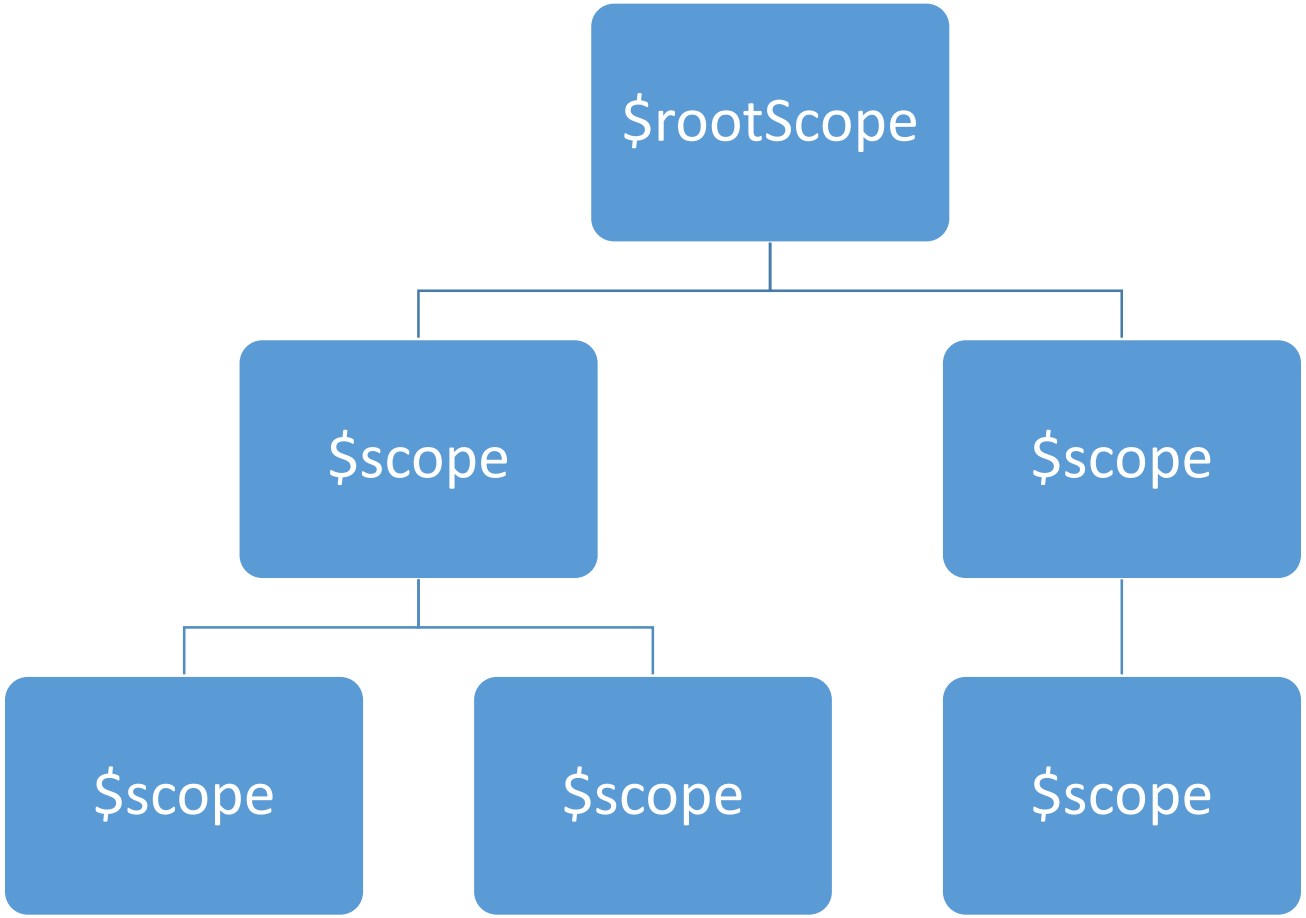
Generar eventos

- Se realizan a nivel de $scope

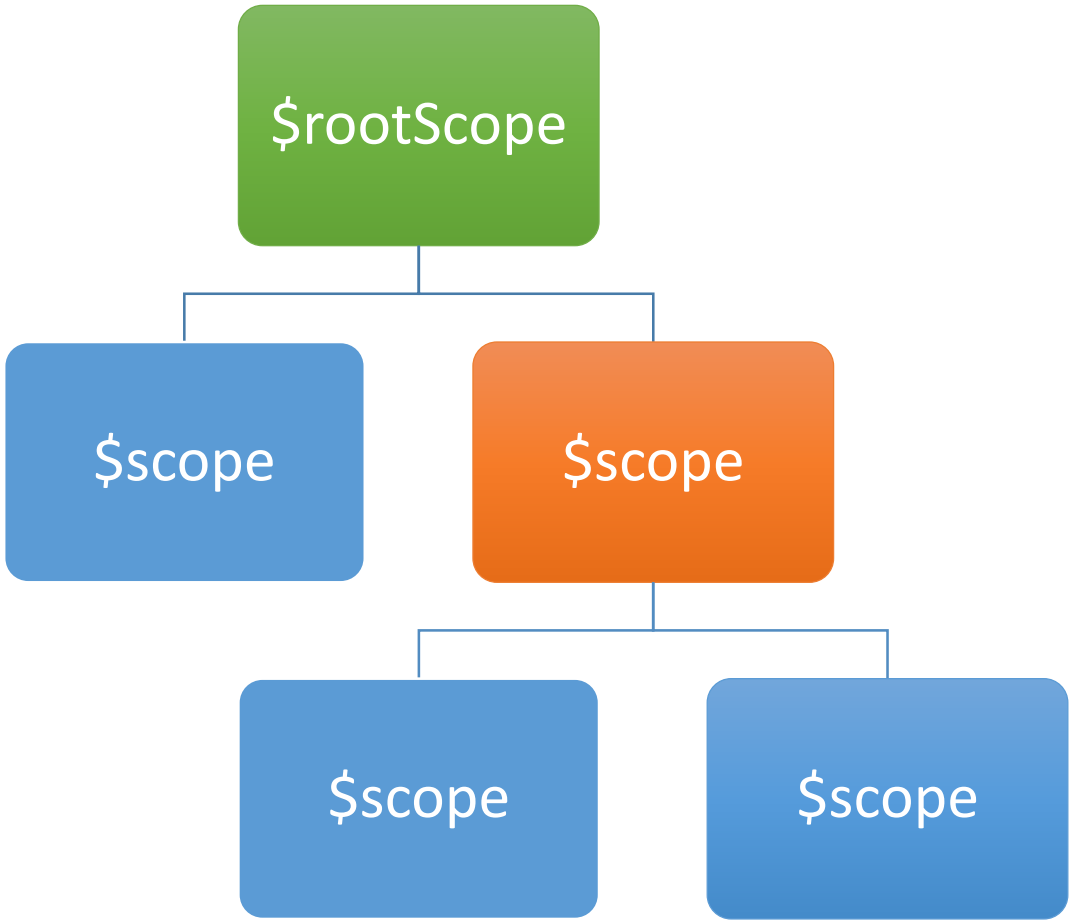
- $scope.$emit
  - Notificación desde el $scope actual hacia arriba. **Bubbling**.
- $scope.$broadcast
  - Notificación desde el $scope actual hacia abajo. **Tunneling**.

- No está soportada la comunicación directa entre hermanos (siblings)

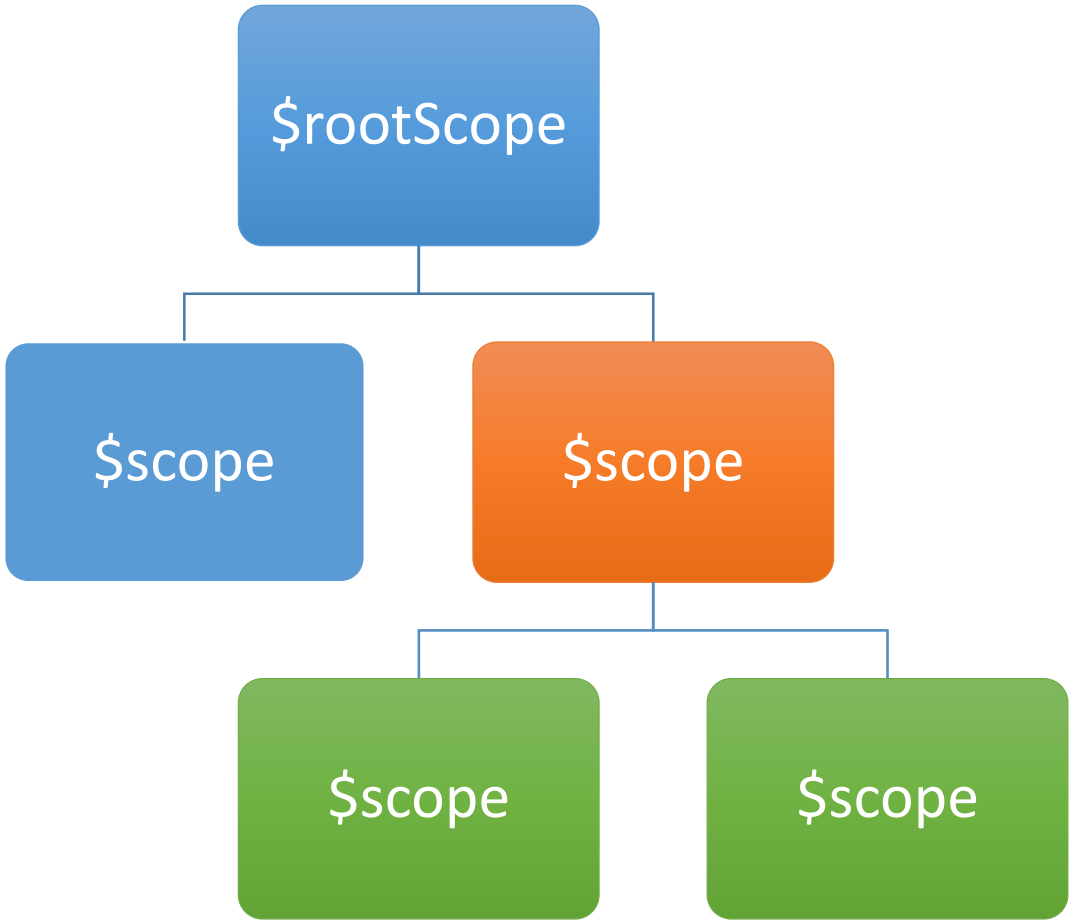
Grafo de $scope en el DOM



$scope.$emit



$scope.$broadcast



Routing y navegación

Anxo Fole
anxofole@plainconcepts.com

Navegación

Necesidades de navegación de una SPA

- Deep-Linking: Una página de la SPA debe poder tener una URL única que se pueda enviar por correo o guardar y que nos lleve a la misma página aunque no sea la primera página que sirve el servidor
- Integración con los botones de adelante y atrás del navegador (HTML5 History API)
- Soporte homogéneo de estas características entre diferentes navegadores

URL Fragment

- Es la parte de la URL que se encuentra detrás de la #
- En cualquier navegador se puede modificar esta parte de la URL sin forzar una recarga de la página
- Se añaden también entradas a la pila del histórico del navegador

- Por ejemplo
  - http://myhost.com/#/admin/users/list
  - http://myhost.com/#/admin/users/new
  - http://myhost.com/#/admin/users/[userId]

Hashbang

- Para hacer más amigable las aplicaciones de Angular a los buscadores se utiliza el #! En vez de sólo # desde la versión 1.6
- Si queremos quitar el prefijo debemos configurar el servicio $location para definir otro prefijo
  - mydomain.com/#/a/b/c => mydomain.com/#!/a/b/c

```
angular.module('location', [])
.config(function ($locationProvider) {
    $locationProvider.hashPrefix('');
})
```

$location

- Servicio que parsea la URL del navegador (basado en window.location) y la hace accesible a la aplicación
- Proporciona un API para acceder a las diferentes partes de una URL
  - Permite manipular la URL del navegador y refleja los cambios de la URL
- Nos permite observar cambios en diferentes componentes de la URL
- Nos abstrae del uso de URL basadas en # o en HTML5

$location [Lectura y escritura]

http://myhost.com/myapp#/admin/users/list?active=true#bottom
http://myhost.com/myapp#!/admin/users/list?active=true#bottom

$location.url(): /admin/users/list?active=true#bottom
$location.path(): /admin/users/list
$location.search(): {active: true}
$location.hash(): Bottom

$location [sólo lectura]

http://myhost.com/myapp#/admin/users/list?active=true#bottom

http://myhost.com/myapp/admin/users/list?active=true#bottom

$location.protocol(): http://

$location.host(): myhost.com

$location.port(): N/A

$location.absUrl(): La URL completa

Eventos de navegación

- $locationChangeStart
  - Notifica el inicio del cambio en la URL
  - Se puede cancelar llamando a preventDefault del evento

- $locationChangeSuccess
  - Notifica cuando se ha navegado a una URL

Navegación desde el controlador

- Dentro de la SPA
  - $location.path(‘route’)
- Fuera de la SPA
  - Necesitamos navegar mediante el servicio $window
  - Abstracción del objeto window del navegador
  - $window.location.href

Routing con uiRouter

ui-router

- Módulo independiente no desarrollado por el equipo de Angular
- https://github.com/angular-ui/ui-router

- Añadir dependencias al modulo **ui.router**

- Proporciona la posibilidad de definir vistas anidadas y relacionadas
  - Facilidad para desarrollar interfaces complejas

Navegación por estados

- Cada estado define el contenido de la/s vistas para una URL determinada
- Pueden estar anidados
  - home
  - contacts
  - contacts.list
  - contacts.detail
  - contacts.detail.item

- Cada estado puede establecer el contenido de múltiples vistas definidas en la página

Definición de las rutas $stateProvider

```
$stateProvider
    .state("home", {
        url: "/",
        template: '<\p>This template is shown in parent unnamed ui-view</p>'

    });
```

```
$stateProvider
    .state("home", {
        url: "/detail/{todoId}",
        component: ‘component’

    });
```

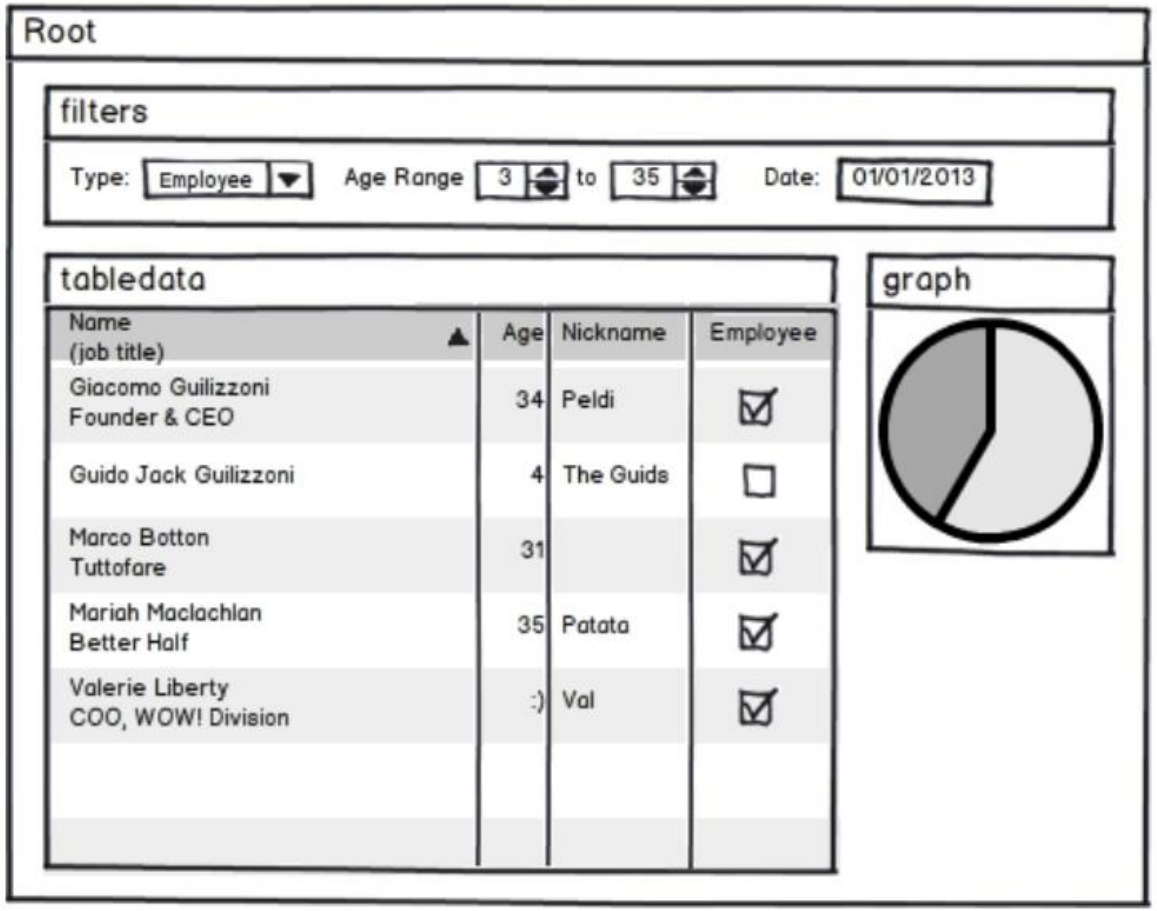
```
$stateProvider
    .state("home", {
        url: "/",
        template: ‘<component></component>’

    });
```

Estados anidados

```
$stateProvider
    .state('contacts', {
        abstract: true,
        url: '/contacts',
        template: ‘<contacts><contacts>',
    })
    .state('contacts.list', {
        url: '',
        templateUrl: ‘<contacts-list><contacts-list>'
    })
```

Estado con plantillas para diferentes vistas



Estado con plantillas para diferentes vistas

```
.state('contacts.detail', {
    url: '/{contactId:[0-9]{1,4}}',
    views: {
        '': {
            templateUrl: 'app/report.filters.html',
            controller: ['$scope', '$stateParams', 'utils',
              function ($scope, $stateParams, utils) {
                  $scope.report = utils.findById($scope.reports, $stateParams.reportId);
              }]
        },
        'graph': {
            template: 'This is a graph data'
        },
        'filters': {
            templateProvider: ['$stateParams',
                               function ($stateParams) {
                                   return '<hr><small class="muted">Contact ID: ' +
                                       $stateParams.contactId + '</small>';
                               }]}
        }
})
```

Rutas con parámetros $stateParams

- Se pueden especificar parámetros en las rutas con :

```
.state('contacts.detail', {
        url: '/{contactId}',
        …
      })
```

- Los parámetros son accesibles en el controlador a través del servicio $stateParams

```
function ($stateParams) {
    return '<hr><small class="muted">Contact ID: ' +
        $stateParams.contactId + '</small>';
}
```

¿Dónde se muestran las vistas?

- ui-view

```
<div ui-view> </div>
```

- Puede haber más de una directiva ui-view
  - Sólamente una en cada template puede no tener nombre
    - Marca el element en el que se insertará la template
  - El resto serán todas con nombre

```
<p ui-view="hint"></p>
```

Eventos de navegación

- $stateChangeStart
  - Al inicio de la navegación
- $stateChangeSuccess
  - Cuando se ha navegado con éxito a un estado
- $stateChangeError
  - Cuando se ha producido algún error en la navegación a un estado
- $stateNotFound
  - Cuando se intent navegar a un estado que no está definido

Navegación

- A través de directivas:
  - ui-sref

```
<a ui-sref="contacts.detail({ id: contact.id })">{{ contact.name }}</a>
```

- A través del servicio $state
  - $state.go()

```
app.controller('ctrl', function ($scope, $state) {
    $scope.changeState = function () {
        $state.go('contact.detail');
    };
});
```

Marcado del estado activo

- A través de directivas
  - ui-sref-active

```
<ul>
  <li ui-sref-active="active" class="item">
    <a href ui-sref="app.user({user: 'bilbobaggins'})">@bilbobaggins</a>
  </li>
</ul>
```

- A través del servicio $state
  - $state.is()
  - $state.includes()

```
$state.$current.name = 'contacts.details.item';
$state.is('contact.details.item'); // true
$state.includes("contacts"); //  true
$state.includes("contacts.details"); //  true
```