

Controladores y Servicios

Controladores

- En Angular, un controlador es una **función constructor JavaScript**
- Se utiliza para **aumentar el scope**
- Cuando un controlador se asocia al DOM a través de la directiva `'ngController'`, Angular crea una instancia de un nuevo objeto controlador

Controladores

- Un subscope se crea como parámetro inyectable a la función constructor del controlador como `$scope`
- Utilizamos controladores Angular para:
 - Configurar el **estado inicial** del objeto **`$scope`**
 - Añadir **comportamiento** al objeto **`$scope`**

Controladores

- No utilizamos controladores Angular para:
 - Manipular DOM - Controladores deben contener sólo la lógica de negocio
 - Formatear inputs de formularios
 - Aplicar filtros
 - Compartir código entre controladores (Para eso servicios)

Controladores

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

Ejercicios

- Utilizando "gna-uilogic-mio.html", crea una app Angular (usando module) y añádele un controlador que genere números aleatorios

Scope:Propiedades

- A menudo, al crear una app, necesitas inicializar el estado inicial del \$scope de Angular
- \$scope **es simplemente un objeto**
- Inicializas el estado añadiendo propiedades a este objeto

Scope:Propiedades

- Las propiedades contienen el modelo de vista (el modelo que será presentado por la vista)
- Todas las propiedades del \$scope estarán **a disposición del template en el DOM**, donde se ha registrado en el controlador

Scope:Propiedades

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

Scope:Propiedades

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

Scope:Comportamiento

- Con el fin de reaccionar a eventos o ejecutar cálculos relacionados con la vista debemos proporcionar un comportamiento al scope
- Le damos comportamiento al scope **añadiendo métodos** al `$scope`

Scope:Comportamiento

- Estos métodos están disponibles para ser llamados desde la vista

```
<body ng-app="myApp">
  <div ng-controller="greetingCtrl">
    <button type="submit" class="btn btn-default" ng-click="greetMe()">
      Greet Me </button> {{ greeting }}
    </div>
  </body>
```

Scope: Comportamiento

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

Scope:Comportamiento

- **ngClick:** La directiva ngClick permite llamar una función específica del \$scope cuando se hace click sobre el elemento

```
<button type="submit" class="btn btn-default"  
ng-click="generateRandomNumber()">Generar Numero Aleatorio</button>
```

Ejercicios

- Añade a tu controlador un método que ejecute la función que calcula números aleatorios cuando este sea convocado. Crea un botón y utiliza la directiva ngClick para poder llamar a este método desde la UI

Servicios

- Son **componentes Angular** disponibles por **Inyección de dependencia**
- **Instanciación perezosa**: Angular solo instancia un servicio cuando un componente depende de el

Servicios

- Son **singletons**
- Angular ofrece muchos servicios útiles (como \$http, \$location etc)
- Al igual que otros identificadores básicos de Angular, los servicios integrados siempre comienzan con \$ (por ejemplo, \$source)

Servicios

- Para utilizar un servicio angular, se agrega como una dependencia para el componente (controlador, servicio, filtro o directiva) que depende del servicio
- La inyección de dependencias de angular se encarga del resto

Servicios

```
var myApp = angular.module('myApp', []);  
myApp.controller('secondsCtrl', function ($scope, $interval){  
    $scope.seconds = 0;  
    $scope.startCount = function(){  
        $scope.seconds ++;  
    }  
});
```

Servicios

- **\$interval** `$interval(fn, delay, [count], [invokeApply]);`
 - Envoltorio Angular `window.setInterval`
 - La función `fn` es ejecutada cada `delay` en milisegundos

Servicios

```
$scope.startCount = function(){  
    $interval(function() {  
        $scope.seconds ++;  
    }, 1000);  
};
```

Ejercicios

- Mejora tu generador de números aleatorios de tal manera que al hacer click, se genere un número aleatorio cada segundo. Puedes usar `$interval` inyectando el servicio en tu controlador