Desarrollo Web Front-End con AngularJS

Características de Angular

Data Binding | MVC | Routing | Testing

jqLite | Templates | History | Factories

AngularJS is a full-featured SPA framework

ViewModel | Controllers | Views | Directives

Controllers | Dependency Injection | Validation

Scope

- Es un objeto con las propiedades y metodos disponibles
- Es el vínculo entre el HTML (View) y el Javascript (Controller)
- Está disponible tanto en la vista como en el controlador

```
function foo() {
   var name = "John";

   function hello() {
      var name = "Jack";
      return "Hello, " + name;
   }

   function goodbye() {
      return "Good bye, " + name;
   }

   hello();    //returns "Hello, Jack"
   goodbye(); //returns "Good bye, John";
}
```

Directivas

```
<!DOCTYPE html>
<html ng-app>
<head>
    <title></title>
</head>
<body>
    <div class="container">
        Name: <input type="text" ng-model="name" /> {{ name }}
    </div>

    <script src="Scripts/angular.js"></script>
</body>
</html>
```

Directive

Directive

Data Binding Expression

Ng-Repeat

```
<html data-ng-app="">
...

   <div class="container"
      data-ng-init="names=['Dave','Napur','Heedy','Shriva']">

       <h3>Looping with the ng-repeat Directive</h3>
       <ul>
           <li data-ng-repeat="name in names">{{ name }}</li>
       </ul>
   </div>

...
</html>
```

Iterate through names

Filters

```
<ul>
    <li data-ng-repeat="cust in customers | orderBy:'name'">
        {{ cust.name | uppercase }}
    </li>
</ul>
```

Order customers by name property

Filter customers by model value

```
<input type="text" data-ng-model="nameText" />
<ul>
    <li data-ng-repeat="cust in customers | filter:nameText | orderBy:'name'">
      {{ cust.name }} - {{ cust.city }}</li>
</ul>
```

View, Controller, Scope

$scope is the "glue" (ViewModel) between a controller and a view

Creando la Vista y el Controlador

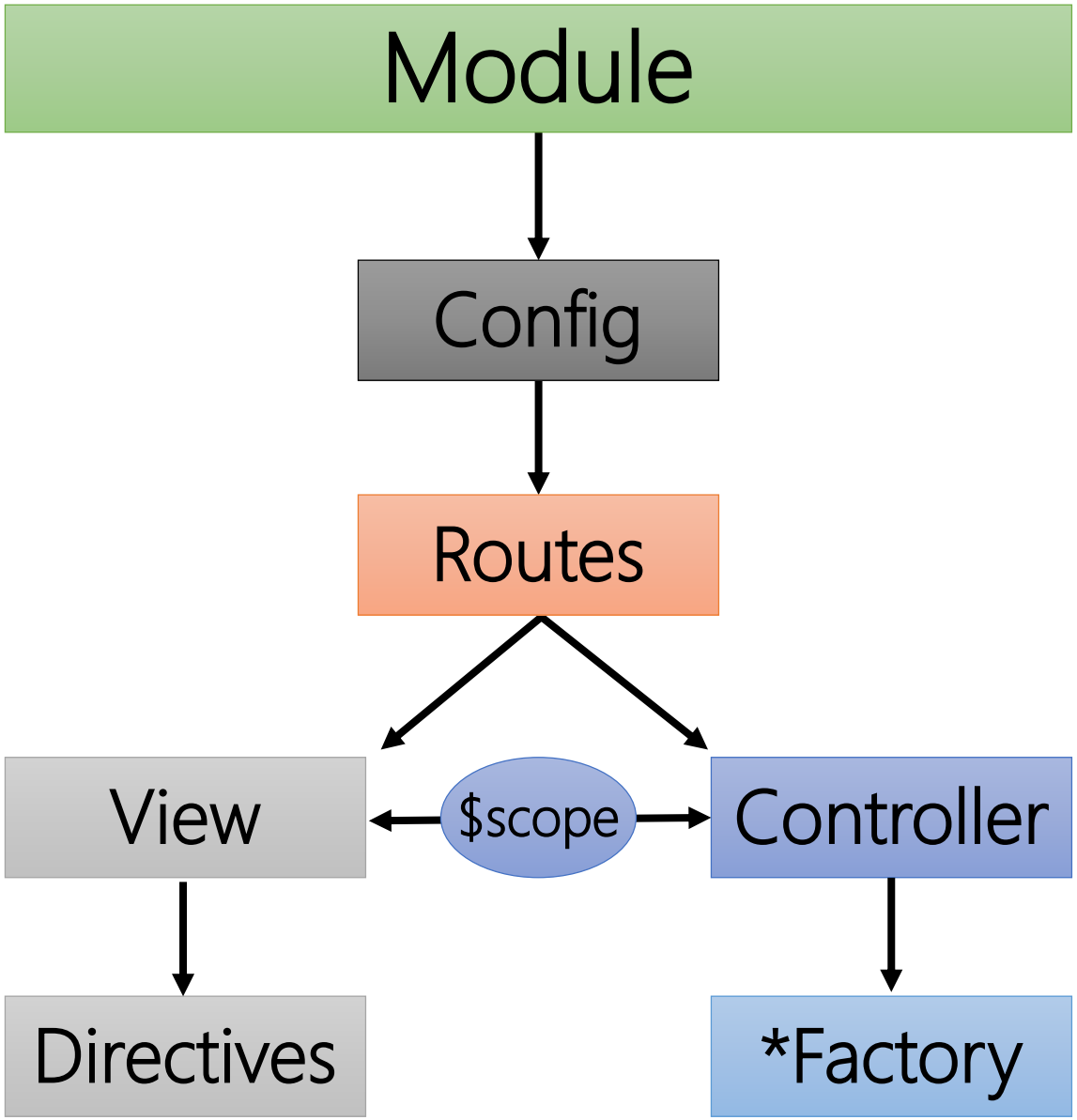
```
<div class="container" data-ng-controller="SimpleController">
    <h3>Adding a Simple Controller</h3>
    <ul>
        <li data-ng-repeat="cust in customers">
            {{ cust.name }} - {{ cust.city }}
        </li>
    </ul>
</div>

<script>
    function SimpleController($scope) {
        $scope.customers = [
            { name: 'Dave Jones', city: 'Phoenix' },
            { name: 'Jamie Riley', city: 'Atlanta' },
            { name: 'Heedy Wahlin', city: 'Chandler' },
            { name: 'Thomas Winter', city: 'Seattle' }
        ];
    }
</script>
```

Define the controller to use

Access $scope

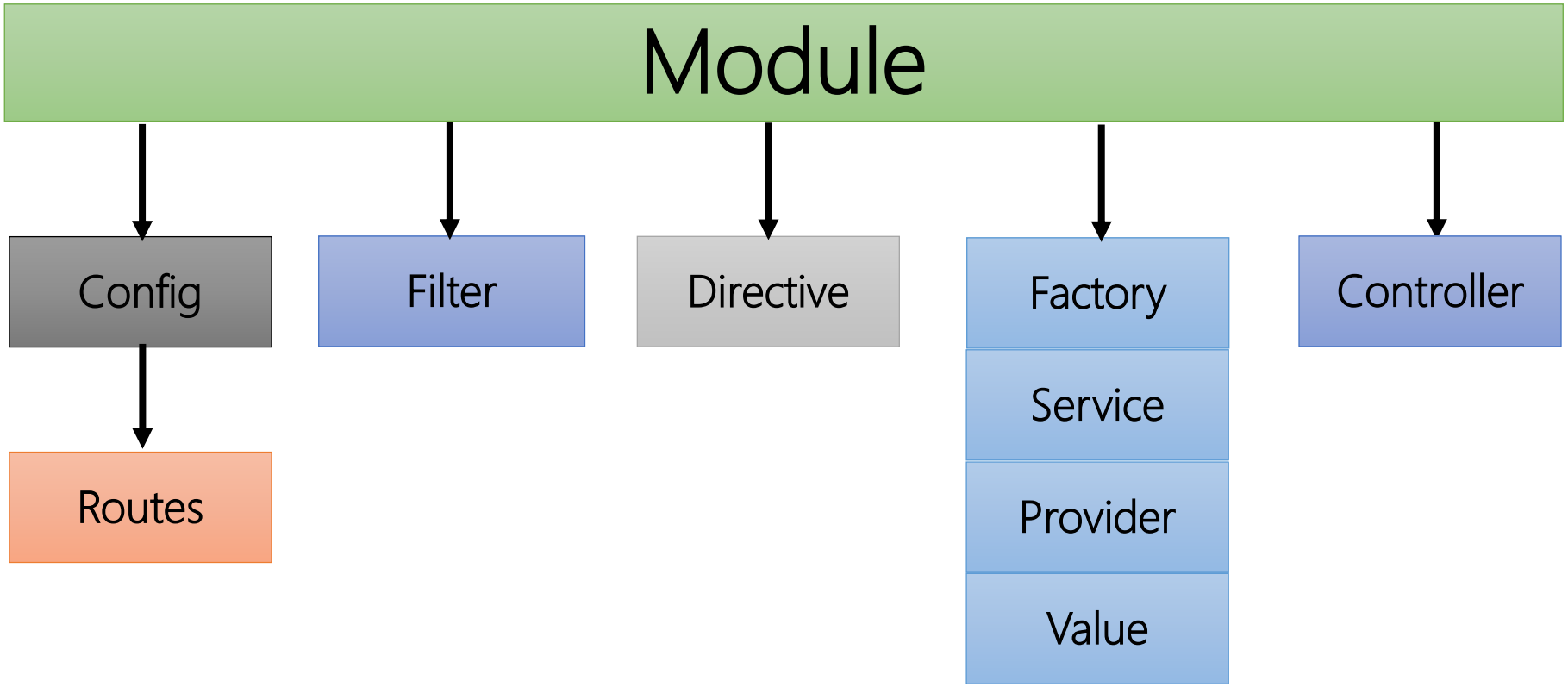
$scope injected dynamically

Basic controller


Module
Config
Routes
View
$scope
Controller
Directives
*Factory

Modules are Containers

```
<html ng-app="moduleName">
```



Creating a Module

What's the Array for?

```
var demoApp = angular.module('demoApp', []);
```

```
var demoApp = angular.module('demoApp',
  ['helperModule']);
```

Module that demoApp depends on

Creating a Controller in a Module

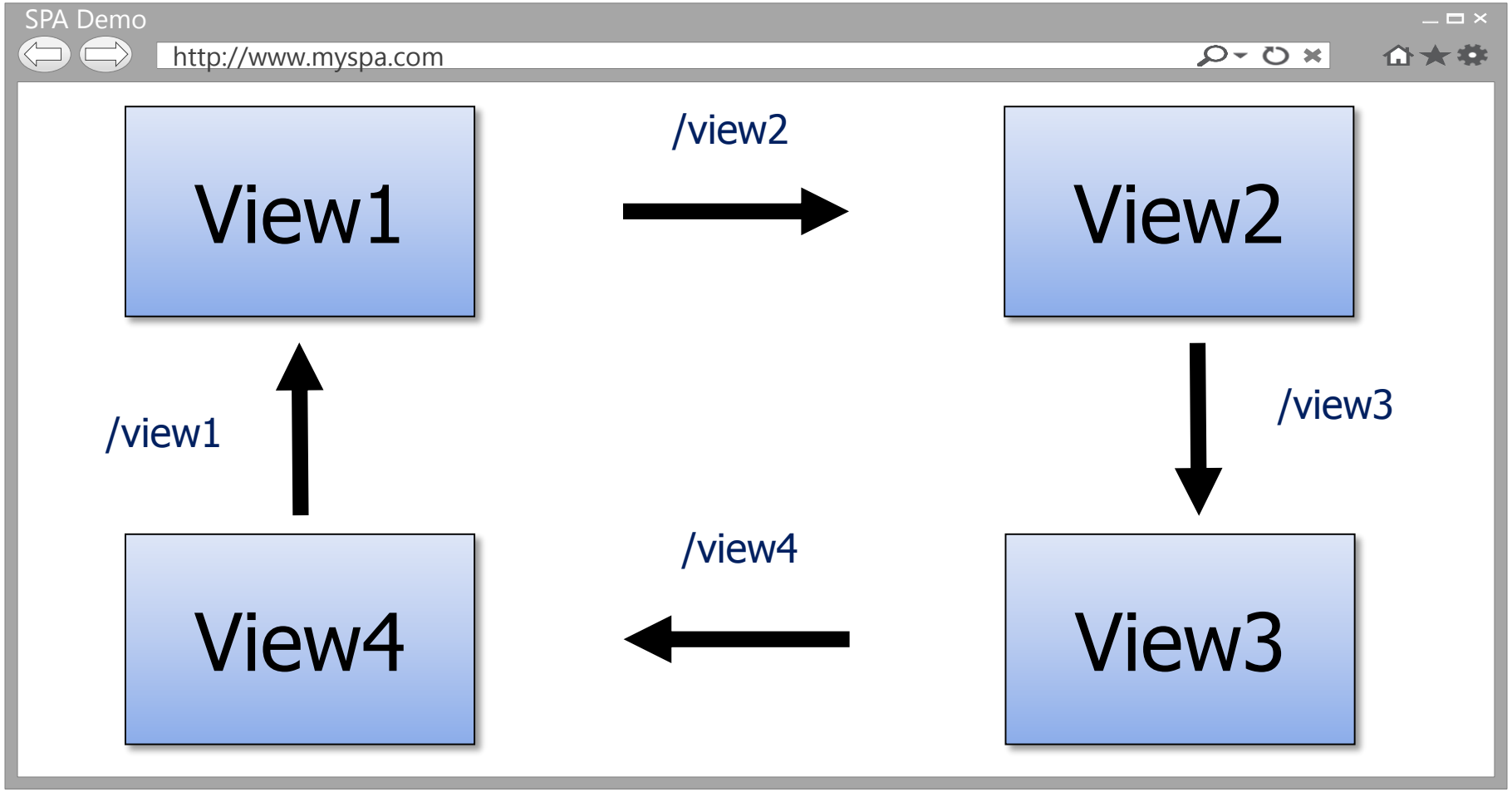
```
var demoApp = angular.module('demoApp', []);
```

Define a Module

Define a Controller

```
demoApp.controller('SimpleController', function ($scope) {
    $scope.customers = [
        { name: 'Dave Jones', city: 'Phoenix' },
        { name: 'Jamie Riley', city: 'Atlanta' },
        { name: 'Heedy Wahlin', city: 'Chandler' },
        { name: 'Thomas Winter', city: 'Seattle' }
    ];

});
```

The Role of Routes



Defining Routes

```
var demoApp = angular.module('demoApp', ['ngRoute']);

demoApp.config(function ($routeProvider) {
    $routeProvider
        .when('/',
            {
                controller: 'SimpleController',
                templateUrl:'View1.html'
            })
        .when('/view2',
            {
                controller: 'SimpleController',
                templateUrl:'View2.html'
            })
        .otherwise({ redirectTo: '/' });
});
```

Define Module Routes

Where do Views Go in a Page?

Dynamically loaded views are injected into the shell page as a module loads:

SPA Demo

http://www.myspa.com

```
<div ng-view></div>
```

OR

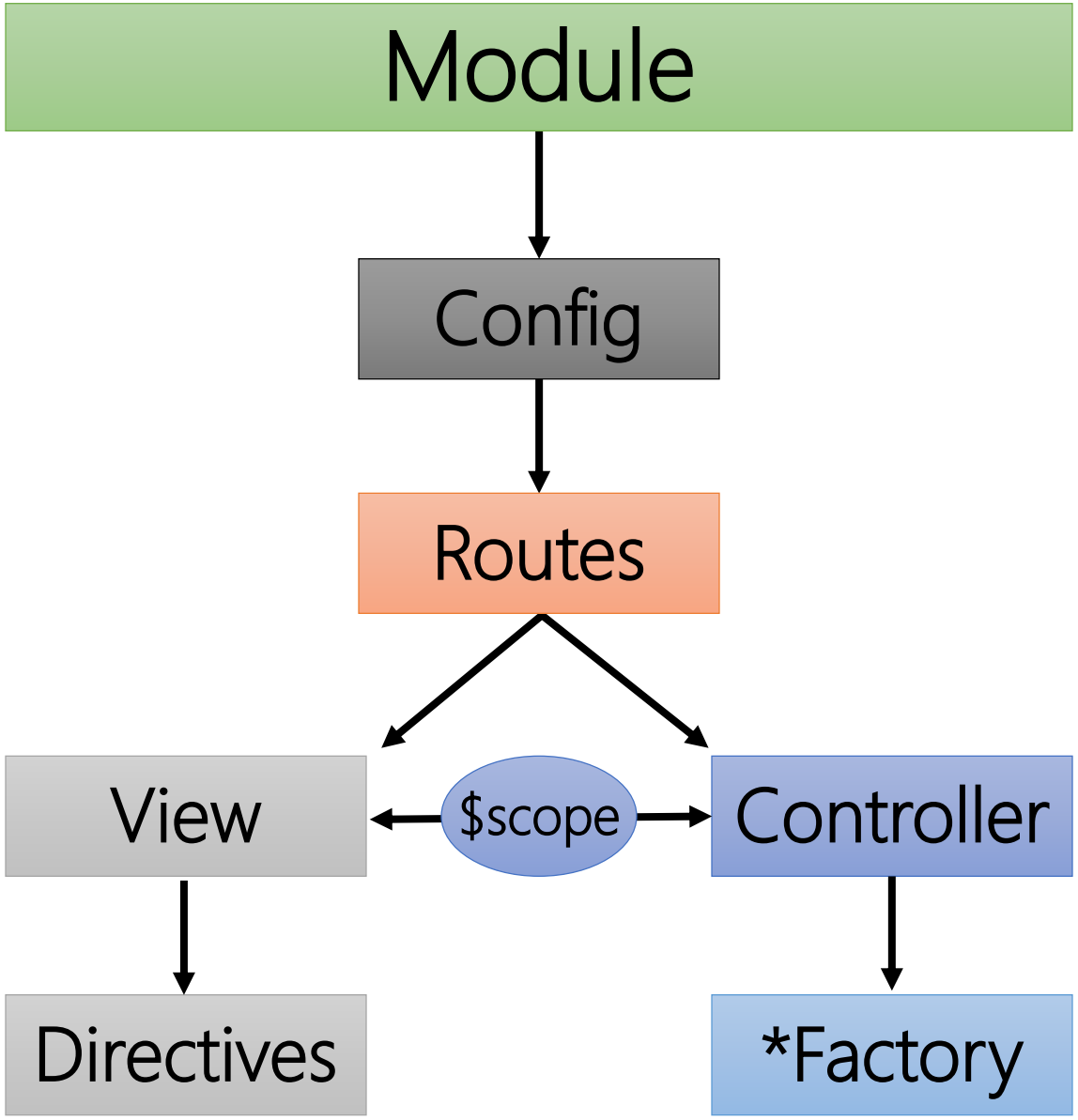
```
<ng-view></ng-view>
```

View1

The Role of Factories

```
var demoApp = angular.module('demoApp', [])
 .factory('simpleFactory', function () {
    var factory = {};
    var customers = [ ... ];
    factory.getCustomers = function () {
        return customers;
    };
    return factory;
 })
 .controller('SimpleController', function ($scope,
    simpleFactory) {
     $scope.customers = simpleFactory.getCustomers();
});
```

Factory injected into controller at runtime


Module
Config
Routes
View
$scope
Controller
Directives
*Factory

Dependency Injection

Cuando una función es dependiente de un dato o funcionalidad, esa función debe ser accessible.

- Option 1: acceso global

```
var a = 4, b = 2;

function divide() {
   return a / b;
}
```

- Option 2: Usando dependency injection

```
function divide(a, b) {
   return a / b;
}
```

Dependency Injection

- La inyección de dependencia require datos específicos por cada parámetro.
- No podemos utilizar la función divide como se muestra a continuación:

```
function divide(a, b) {
   return a / b;
}

divide('hello', { x: 2 });
```

Filters

- ¿Qué es un filtro?:
  - Un filtro formatea el valor de una expresión para mostrar al usuario
- Los filtros pueden ser utilizados en HTML mediante la notación por barra o en JavaScript inyectando el servicio $filter

Filters

- HTML Example

```
<div ng-app='myApp' ng-controller="myController">
   <p>{{name | uppercase}}</p>
   <p>{{uppercaseName()}}</p>
</div>
```

- JavaScript Example

```
var module = angular.module('myApp', []);

module.controller('myController', [
   '$scope', '$filter', function($scope, $filter) {

      $scope.name = 'John Smith';

      $scope.uppercaseName = function() {
         return $filter('uppercase')($scope.name);
      };

   }]);
```

Filters with Parameters

- HTML Example

```
{{ expression | filterName : param1 : param2 }}
```

- JavaScript example

```
$filter('filterName')(expression, param1, param2);
```

Core Filters

- AngularJS tiene varios filtros incorporados:
  - currency
  - date
  - filter
  - json
  - limitTo
  - lowercase
  - number
  - orderby
  - uppercase

```
{{ '2015-03-19T19:00:00.000Z' | date : 'MMMM yyyy' }}
```

```
$filter('date')(new Date(), 'MMMM yyyy');
```

Routes

- $routeProvider – used for dealing with routes

Modified app.js

```
angular.module('F1FeederApp', [
  'F1FeederApp.services',
  'F1FeederApp.controllers',
  'ngRoute'
]).
config(['$routeProvider', function($routeProvider) {
  $routeProvider.
    when("/drivers", {templateUrl: "partials/drivers.html", controller: "driversController"}).
    when("/drivers/:id", {templateUrl: "partials/driver.html", controller: "driverController"}).
    otherwise({redirectTo: '/drivers'});
}]);
```

Service

- En AngularJS, un servicio es una función, o un objeto, que está disponible y limitado a nuestra aplicación AngularJS
- El servicio genera un singleton de un objeto instanciado

```
//define a service
myModule.service('person', [function() {
   this.first = 'John';

   this.last = 'Jones';

   this.name = function() {
      return this.first + ' ' + this.last;
   };
}]);

//inject the person service
myModule.controller('myController', ['$scope', 'person', function($scope, person) {
   $scope.name = person.name();
}]);
```

Service

- AngularJS tiene cerca de 30 servicios incorporados. Algunos de ellos:
  - $location.
  - $timeout
  - $http
  - $interval
- AngularJS supervisa constantemente tu aplicación, y para manejar los cambios y eventos de manera apropiada, AngularJS prefiere $location en lugar de window.location