

# AngularJS

Βασίλειος Καραβάσιλης  
Μονάδα Αριστείας ΕΛΛΑΚ | ΕΤΕΠΗ | 13/10/2014

# Ιστορία

- 2009, Google Feedback
- Αρχική υλοποίηση
  - 17000 γραμμές
  - 6 μήνες ανάπτυξη
- Ένα μέλος τις ομάδας είπε ότι μπορεί να το κάνει σε 2 βδομάδες με ένα open source εργαλείο που ανέπτυξε στον ελεύθερο χρόνο του
- Αποτέλεσμα:
  - 1500 γραμμές
  - 3 βδομάδες ανάπτυξη

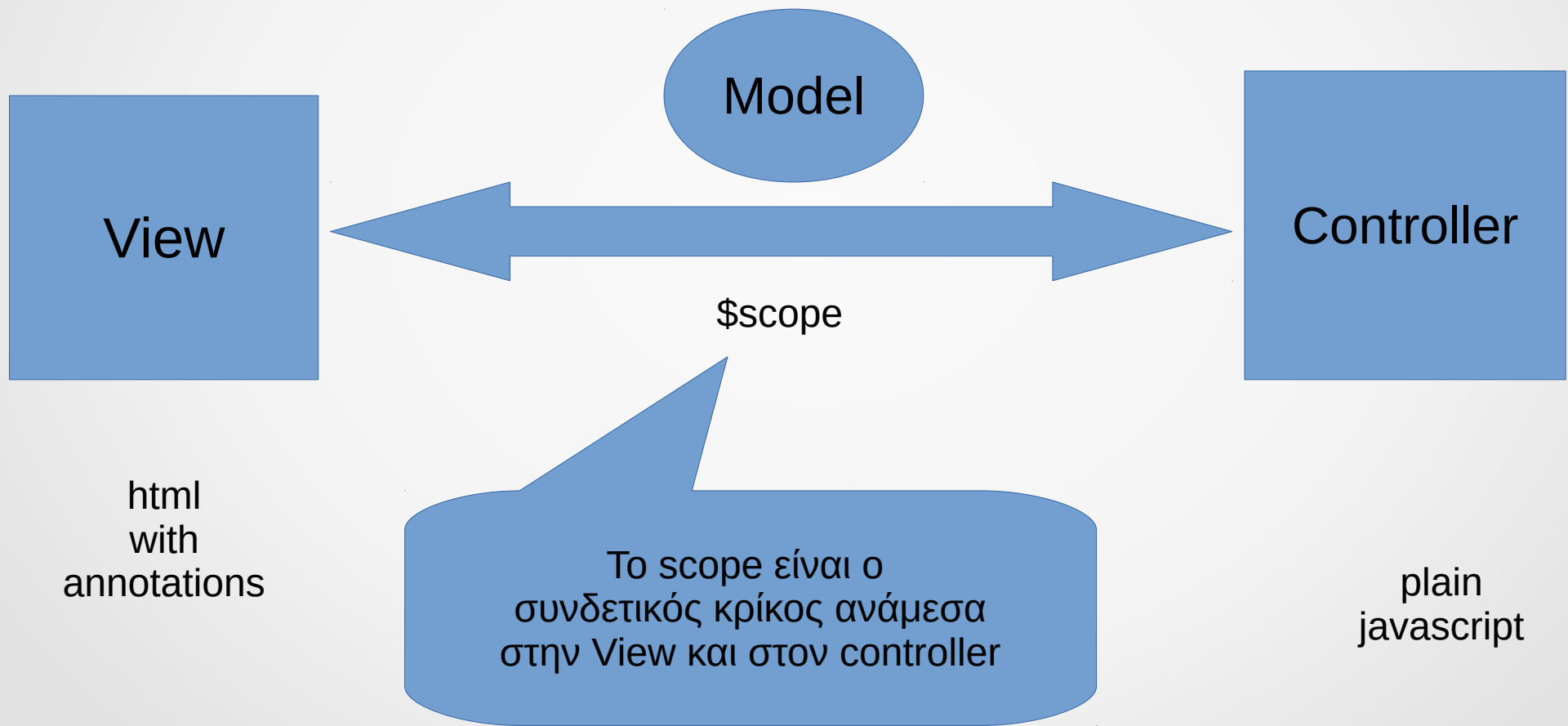
# Άδεια χρήσης

- MIT license
- Επιτρέπει να χρησιμοποιήσουμε την AngularJS χωρίς να δώσουμε το source code (θεωρητικά).
- Στο software που θα δώσουμε πρέπει να πούμε ότι χρησιμοποιεί AngularJS και να δώσουμε ένα αντίγραφο από το MIT license.
- Κάποιος άλλος μπορεί να πάρει τον πηγαίο κώδικα (όχι το εκτελέσιμο), να τον τροποποιήσει (αν θέλει), και να τον αναδιανείμει (είτε επί πληρωμή είτε όχι).
  - Θα πρέπει όμως να πει ότι χρησιμοποιεί το software που πήρε από εμάς.

# Εισαγωγή

- Είδος: single page apps
  - Οι διάφορες λειτουργίες επηρεάζουν μόνο ένα μέρος της σελίδας
  - Δεν φορτώνουμε μια νέα σελίδα κάθε φορά που επιλέγουμε κάτι
  - Παράδειγμα: Gmail
- Βασική ιδέα: Η παρουσίαση των περιεχομένων (html, css) διαχωρίζεται από τα δεδομένα
  - Από τον server αποστέλλονται ξεχωριστά:
    - html σελίδες
    - Τα δεδομένα (ίσως μέσω web services)
  - Η προσθήκη των δεδομένων στις σελίδες γίνεται στον browser του χρήστη
- Αρχιτεκτονική: Model – View – Controller (MVC)

# Model – View - Controller



# Χρήση

## Download AngularJS

Branch

1.2.x (legacy)

1.3.x (latest)

Build

Minified

Uncompressed

Zip

CDN

https://ajax.googleapis.com/ajax/libs/angularjs/1.2.26/angular.min.js

Bower

bower install angular#1.2.26

Extras

Browse additional modules

Previous Versions

 Download

Μπορούμε να επιλέξουμε  
να κατεβάσουμε

[illegible]

ένα αρχείο με μέγεθος  
περίπου 100KB

Ή να το χρησιμοποιήσουμε κατευθείαν:

```
<script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.26/angular.min.js">
</script>
```

# Παράδειγμα

Αρχείο: index.html

- ```
<html ng-app>
  <head>
    <script src="angular.js"></script>
    <script src="controllers.js"></script>
  </head>
  <body>
    <div ng-
      controller='HelloController'>
      <p>
        {{greeting.text}}, World
      </p>
    </div>
  </body>
</html>
```

Το ng-app ορίζει ποια στοιχεία της σελίδας ελέγχει το angularjs. Εδώ ελέγχει όλη την σελίδα.

Τα άγκιστρα {{}} ορίζουν ότι το συγκεκριμένο μέρος με κάτι θα αντικατασταθεί

Το ng-controller ορίζει τον controller που θα χρησιμοποιηθεί στο συγκεκριμένο div.

Αρχείο: controllers.js

- ```
function HelloController($scope) {
  $scope.greeting = { text: 'Hello' };
}
```

Στο αρχείο controllers.js ορίζουμε μια συνάρτηση που έχει το ίδιο όνομα με το όνομα στο ng-controller.

Το \$scope είναι ένα αντικείμενο που δημιουργείται αυτόματα από το angularjs. Μέσα στο \$scope ορίζουμε ένα νέο αντικείμενο που το λέμε greeting.

Το greeting είναι ένα λεξικό (hash) που περιέχει την αντιστοίχιση: text => 'Hello'

# Παράδειγμα

Αρχείο: index.html

- ```
<html ng-app>
  <head>
    <script src="angular.js"></script>
    <script src="controllers.js"></script>
  </head>
  <body>
    <div ng-
      controller='HelloController'>
      <p>
        {{greeting.text}}, World
      </p>
    </div>
  </body>
</html>
```

Αρχείο: controllers.js

- ```
function HelloController($scope) {
  $scope.greeting = { text: 'Hello' };
}
```

Αποτέλεσμα:  
Hello, World

# Μειονέκτημα

- Αλλαγές στην html
  - Τα επιπλέον στοιχεία που εισάγουμε λέγονται directives.
- Πρέπει να μάθουμε ακόμη ένα framework

# Πλεονεκτήματα

- Δεν χρησιμοποιούμε ID στα στοιχεία της HTML.
  - Μπορούμε να γράψουμε unit tests για τους controller.
- Απλώς γράψαμε έναν controller με το ίδιο όνομα και στα δύο αρχεία.
- Ο controller είναι κανονική javascript.
- Το \$scope δεν το δημιουργήσαμε εμείς, ούτε το συνδέσαμε με το view και τον controller.
- Ο controller καλέστηκε αυτόματα όταν χρειάστηκε ώστε να παραχθεί το greeting.text.
- Τα html αρχεία κατεβαίνουν μόνο μια φορά από τον server.

# Μεγάλο πλεονέκτημα

- Η σύνδεση του controller με το greetings.text γίνεται αυτόματα (data binding)
  - Αν αλλάξει ο controller κάποια στιγμή το greetings.text, η αλλαγή θα γίνει αυτόματα ορατή στην σελίδα, χωρίς να κάνουμε τίποτα
  - Αν το greetings.text άλλαζε στην σελίδα, τότε θα μπορούσε να καλείτε αυτόματα μια λειτουργία του controller
    - (στο συγκεκριμένο παράδειγμα δεν γίνεται, αλλά σε άλλες περιπτώσεις γίνεται)

# Δυναμικό παράδειγμα

- ```
<html ng-app>
  <head>
    <script src="angular.js"></script>
    <script src="controllers.js"></script>
  </head>
  <body>
    <div ng-controller='HelloController'>
      <input ng-model='greeting.text'>
      <p>
        {{greeting.text}}, World
      </p>
    </div>
  </body>
</html>
```

Δεν χρειάζεται να  
πειράζουμε τον  
controller

Το ng-model συνδέει  
το input με  
κάποια περιεχόμενα του  
\$scope

Αποτέλεσμα:

Hello, World

# Πολλαπλά στοιχεία

```
• <html ng-app>
  <head>
    <script src="angular.js"></script>
    <script>
      function MyController($scope) {
        $scope.items = [
          {title: 'Apple', price: 10},
          {title: 'Orange', price: 12.95}
        ];
      }
    </script>
  </head>
  <body ng-controller='MyController'>
    <div ng-repeat='item in items'>
      <span>{{item.title}} - {{item.price | currency}} </span>
    </div>
  </body>
</html>
```

Τον controller τον έχουμε εδώ.

Θα φτιάξει 2 <span>

Θα εφαρμόσει ένα  
φίλτρο  
(θα βάλει το \$ μπροστά)

Αποτέλεσμα:

Apple - \$10.00  
Orange - \$12.95

# Διαγραφή

```
• <html ng-app>
  <head>
    <script src="angular.js"></script>
    <script>
      function MyController($scope) {
        $scope.items = [
          {title: 'Apple', price: 10},
          {title: 'Orange', price: 12.95}
        ];
        $scope.remove = function(index) {
          $scope.items.splice(index, 1);
        }
      }
    </script>
  </head>
  <body ng-controller='MyController'>
    <div ng-repeat='item in items'>
      <span>{{item.title}} - {{item.price | currency}} </span>
      <button ng-click="remove($index)">Remove</button>
    </div>
  </body>
</html>
```

Ορίζουμε μια συνάρτηση μέσα στο \$scope.

Το ng-click θα καλέσει την συνάρτηση remove του controller. Το \$index θα το βρει αυτόματα, γιατί είμαστε μέσα σε ng-repeat.

Αποτέλεσμα:

Apple - \$10.00 Remove

# Εμφάνιση κειμένου

- `<p> {{myText}} </p>`
- `<p ng-bind="myText"> </p>`
- Ανάμεσα στο `<p> ... </p>` θα εμφανιστεί το περιεχόμενο του `$scope.myText`. Αν το περιεχόμενο του `$scope.myText` αλλάξει, τότε θα αλλάξει αυτόματα και το περιεχόμενο ανάμεσα στα `<p> ... </p>`.

# Είσοδος πληροφορίας από <input>

- `<input type="checkbox" ng-model="myModel" ng-change="myFunction()"/>`
- Αν αλλάξει η τιμή του `$scope.myModel` στον controller, τότε θα αλλάξει και η εμφάνιση του checkbox.
- Αν ο χρήστης αλλάξει το checkbox, τότε θα αλλάξει και η τιμή του `$scope.myModel`. Επίσης, θα κληθεί η συνάρτηση `myFunction` του controller.

# Click σε button

- `<button ng-click="myFunction1()"> press me1 </button>`
- `<button ng-dbclick="myFunction2()"> press me2 </button>`
- Όταν πατήσουμε το κουμπί press me1 θα κληθεί η συνάρτηση myFunction1.
- Όταν πατήσουμε διπλό κλικ στο κουμπί press me2 θα κληθεί η συνάρτηση myFunction2.

# Υποβολή form

- `<form ng-submit="myProcess()">`
- Όταν ο χρήστης υποβάλει την φόρμα (πατώντας πιθανόν κάποιο κουμπί), θα κληθεί και η συνάρτηση `myProcess`.

# Επανάληψη σε λίστες

- `<li ng-repeat="item in items">`  
    `<a href="/item/{{item.id}}"> {{item.name}} </a>`  
    `</li>`
- Έστω ότι έχουμε μια λίστα `items`.
- Το `ng-repeat` θα δημιουργήσει πολλαπλά αντίγραφα του `<li> ... </li>`. Το κάθε αντίγραφο θα έχει ένα διαφορετικό στοιχείο της λίστας `items`.
- Μέσα στο `element` του `ng-repeat` υπάρχουν διαθέσιμα:
  - `$index`: η θέση του `item` μέσα στην λίστα `items`. Ξεκινά από το 0.
  - `$first`, `$middle`, `$last`: `boolean` που είναι `true` αν το συγκεκριμένο `item` είναι πρώτο, μεσαίο ή τελευταίο.

# Εμφάνιση – Απόκρυψη

- `<a href="..." ng-show="myFlag"> ... </a>`
- Το στοιχείο `<a> ... </a>` θα εμφανιστεί αν το `myFlag` είναι `true`. Αν είναι `false`, δεν θα εμφανιστεί. Αν αλλάξει τιμή το `myFlag`, θα αλλάξει εμφάνιση και το στοιχείο.
- Υπάρχει και το `ng-hide` που είναι το αντίθετο από το `ng-show`.

# Εμφάνιση – Απόκρυψη μέσω css

- `.hidden {  
 display:none;  
}`
- `<a href="..." class="{{myClass}}"> ... </a>`
- Αν το myClass είναι hidden, τότε δεν θα εμφανιστεί το `<a> ... </a>`.
- Εναλλακτικά:  
`<a href="..." style="display:{{myDisplay}}"> ... </a>`

# src, href

- Αντί για src, href χρησιμοποιούμε ng-src, ng-href.
  - Ο browser μπορεί να το φορτώνει παράλληλα με την σελίδα, οπότε να μην το βρει.
- ``
- `<a ng-href="/pages/{{myPage}}">`

# Συνάρτηση \$watch

- Ορίζουμε μια συνάρτηση που θα καλείται όταν αλλάζει ένα μέρος του μοντέλου.
- `$watch(watchFn, watchAction, deepWatch)`
  - `watchFn`: Το όνομα ενός αντικειμένου ή μια συνάρτηση που επιστρέφει την τιμή κάποιο τμήματος του μοντέλου. Αν είναι συνάρτηση δεν πρέπει να αλλάζει το μοντέλο.
  - `watchAction`: το όνομα μιας συνάρτησης που καλείται αν η τιμή του `watchFn` αλλάζει. Η μορφή της είναι:
    - `function(newValue, oldValue, scope)`
  - `deepWatch`: αν είναι `true`, τότε ελέγχει κάθε μέλος του `watchFn`, αν αυτό είναι μια περίπλοκη δομή, πχ πίνακας.
  - Επιστρέφει μια συνάρτηση που μπορούμε να την καλέσουμε για να σταματήσουμε την παρακολούθηση.

# Παράδειγμα \$watch

- `<div ng-controller="MyController">  
 <span> {{value}} </span>  
 <input ng-model="inValue"/>  
</div>`
- ```
function MyController($scope){  
  function myFun(newValue, oldValue, scope) {  
    $scope.value = newValue * 10;  
  }  
  $scope.$watch($scope.inValue, myFun);  
}
```

# Κλήση συνάρτησης ανά τακτά χρονικά διαστήματα

- ```
<html ng-app>
<head>
  <script src="angular.js"></script>
  <script>
    function MyController($scope, $timeout) {
      $scope.counter = 0;
      $scope.onTimeout = function(){
        $scope.counter++;
        mytimeout = $timeout($scope.onTimeout,1000);
      }
      var mytimeout = $timeout($scope.onTimeout,1000);
      $scope.stop = function(){
        $timeout.cancel(mytimeout);
      }
    }
  </script>
</head>
<body>
  <div ng-controller="MyController">
    {{counter}}
    <button ng-click="stop()">Stop</button>
  </div>
</body>
</html>
```

Κάθε φορά που καλείται,  
την βάζουμε να  
ξανακαλεί τον εαυτό της.

Καλείται την πρώτη φορά.

Αν πατήσουμε το stop  
τότε σταματάμε  
την κλήση.

# Modules

- Βοηθούν να τακτοποιήσουμε τον κωδικά
- `var appMod = angular.module('app', []);`
  - ορίζουμε ένα module με όνομα app
- `var appMod2 = angular.module('app2', ['app', 'app1']);`
  - Ορίζουμε το module με όνομα app2 που εξαρτάται από τα module app και app1. Αυτό σημαίνει ότι θα το angular θα κάνει include και τα app και app1 αν εμείς χρησιμοποιήσουμε το appMod2.
- Χρήση στην HTML:
  - `<html ng-app='app2'>`

# Modules και controllers

- `var appMod = angular.module('app', []);`
- `appMod.controller('MyController', function ($scope) {  
...  
});`
- `'H`
- `var controllers = {};  
controllers.MyController = function ($scope) {  
...  
};  
appMod.controller(controllers);`

# Φίλτρα

- Μπορούμε να τα χρησιμοποιήσουμε για να αλλάξουμε την μορφή ενός αριθμού, να επιλέξουμε μόνο τα αποτελέσματα που μας ενδιαφέρουν ή να ταξινομήσουμε μια λίστα. Μπορούμε να φτιάξουμε εμείς καινούρια φίλτρα.
- `{{13 | currency}}`
  - Βάζει ένα \$ μπροστά από τον αριθμό
- `<li ng-repeat="item in items | filter:search">`
  - Εμφανίζει μόνο τα αντικείμενα που περιέχουν στο όνομα με το search
- `<li ng-repeat="item in items | orderBy:name">`
  - Ταξινομεί τα αντικείμενα με βάση το name.
- `<li ng-repeat="item in items | filter:search | orderBy:name">`
  - Επιλέγει κάποια αντικείμενα και αυτά τα ταξινομεί

# Routes

- Ουσιαστικά, κάνουμε μια σελίδα που είναι single page application να αλλάζει μέρος των περιεχομένων και ουσιαστικά να δίνει την αίσθηση ότι μεταφερόμαστε σε πολλές σελίδες.
- Τα routes μπορούν να χρησιμοποιηθούν ώστε ανάλογα με το URL που έχει ο browser να δείχνουμε διαφορετικές πληροφορίες.
- Χρησιμοποιούμε το `$routeProvider` (είναι έτοιμο από το AngularJS).

# Παράδειγμα Routes - index.html

- ```
<html ng-app='app'>
  <head>
    <script src='angular.js'> </script>
    <script src='angular-route.min.js'> </script>
    <script src='controllers.js'> </script>
  </head>
  <body>
    <div ng-view>
    </div>
  </body>
</html>
```

# Παράδειγμα Routes - list.html

- ```
<table>
  <tr>
    <td> name </td>
    <td> details </td>
  </tr>
  <tr ng-repeat='item in items'>
    <td> {{item.name}} </td>
    <td> <a href='#/view/{{item.id}}'> GO </a> </td>
  </tr>
</table>
```

# Παράδειγμα Routes - detail.html

- `<div> {{item.name}} </div>`  
`<div> {{item.id}} </div>`  
`<a href='#/'> Back </a>`

# Παράδειγμα Routes – controllers.js

- ```
var module = angular.module("sampleApp", ['ngRoute']);
module.config(['$routeProvider',
function($routeProvider) {
  $routeProvider.
    when('/', {
      templateUrl: 'list.html',
      controller: 'ListController'
    }).
    when('/view/:id', {
      templateUrl: 'detail.html',
      controller: 'DetailController'
    }).
    otherwise({
      redirectTo: '/'
    });
}]);
```

# Παράδειγμα Routes – controllers.js

- ```
items = [  
  {name:'A', id: 1},  
  {name:'X', id: 2}  
]
```

```
module.controller("ListController", function($scope) {  
  $scope.items = items;  
});
```

```
module.controller("DetailController", function($scope,  
$routeParams) {  
  $scope.item = items[$routeParams.id-1];  
});
```

# Επικοινωνία με server

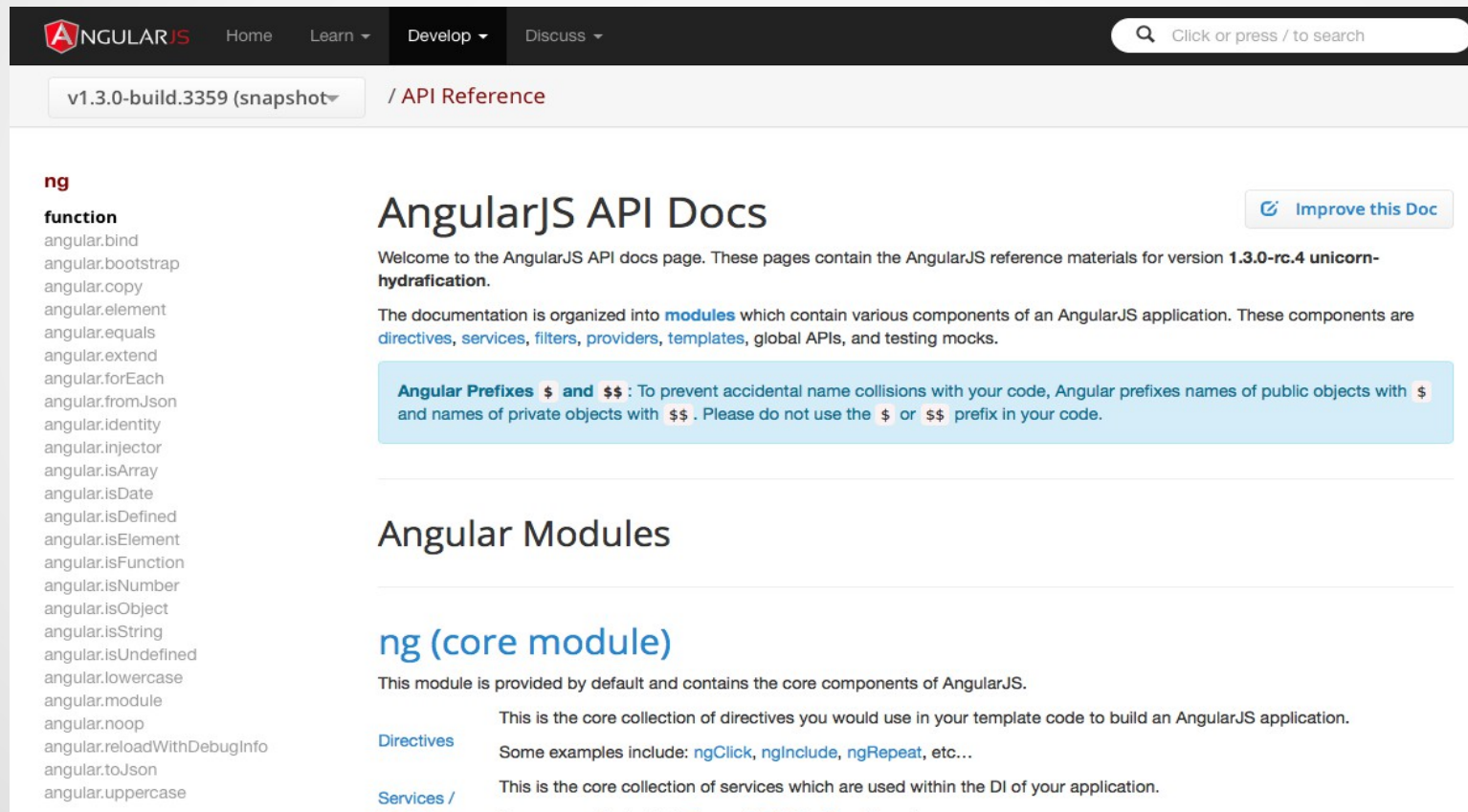
- Τα αντικείμενα τα δίνει ο server σε μορφή json:
  - [  
  {name:'a',  
  id: 1},  
  {name:'b',  
  id: 2},  
  ...  
]
- Χρησιμοποιούμε το \$http που είναι έτοιμο από το AngularJS.

# Επικοινωνία με server

- Controller:
- ```
function loadContents($scope, $http) {  
  $http.get('/items').success(  
    function(data, status, headers, config) {  
      $scope.items = data;  
    }  
  );  
}
```
- HTML:
- ```
<li ng-repeat='item in items'>  
  {{item.name}} – {{item.id}}  
</li>
```

# Περισσότερες Πληροφορίες

- <https://docs.angularjs.org/api>



The screenshot shows the AngularJS API documentation page for version 1.3.0-build.3359 (snapshot). The page has a dark header with the AngularJS logo and navigation links: Home, Learn, Develop, and Discuss. A search bar is located on the right. Below the header, the version and the current section, "/ API Reference", are displayed. The main content area is titled "AngularJS API Docs" and includes a welcome message and a brief overview of the documentation structure. A blue box contains a note about Angular prefixes. The left sidebar lists various AngularJS modules, with "ng" selected. The main content area also lists "Angular Modules" and "ng (core module)".

**AngularJS API Docs**

Welcome to the AngularJS API docs page. These pages contain the AngularJS reference materials for version **1.3.0-rc.4 unicorn-hydrification**.

The documentation is organized into **modules** which contain various components of an AngularJS application. These components are **directives**, **services**, **filters**, **providers**, **templates**, global APIs, and testing mocks.

**Angular Prefixes** **\$** and **\$\$** : To prevent accidental name collisions with your code, Angular prefixes names of public objects with **\$** and names of private objects with **\$\$**. Please do not use the **\$** or **\$\$** prefix in your code.

## Angular Modules

### ng (core module)

This module is provided by default and contains the core components of AngularJS.

**Directives**

This is the core collection of directives you would use in your template code to build an AngularJS application. Some examples include: **ngClick**, **ngInclude**, **ngRepeat**, etc...

**Services /**

This is the core collection of services which are used within the DI of your application.

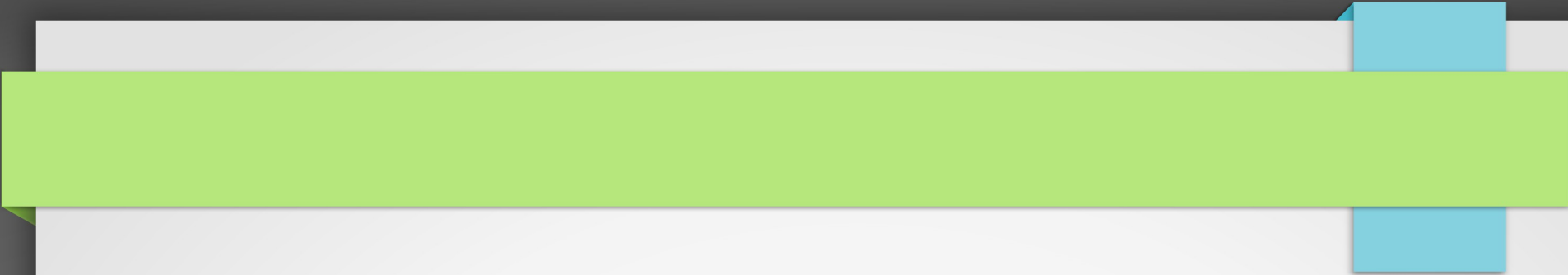
**ng**

**function**

- angular.bind
- angular.bootstrap
- angular.copy
- angular.element
- angular.equals
- angular.extend
- angular.forEach
- angular.fromJson
- angular.identity
- angular.injector
- angular.isArray
- angular.isDate
- angular.isDefined
- angular.isElement
- angular.isFunction
- angular.isNumber
- angular.isObject
- angular.isString
- angular.isUndefined
- angular.lowercase
- angular.module
- angular.noop
- angular.reloadWithDebugInfo
- angular.toJson
- angular.uppercase

# Demo

- Chat application
- Ο κάθε χρήστης θα μπορεί να γράψει κάτι.
- Ό,τι γράφει θα εμφανίζεται σε όλους τους χρήστες αυτόματα.



# Ερωτήσεις