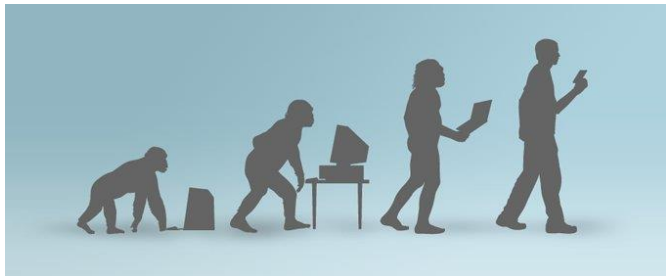
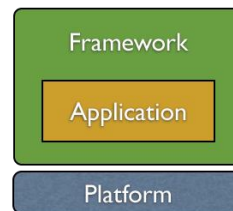


# Temas de Hoy



**Evoluciones**

**Frameworks**





PREVIOUSLY...

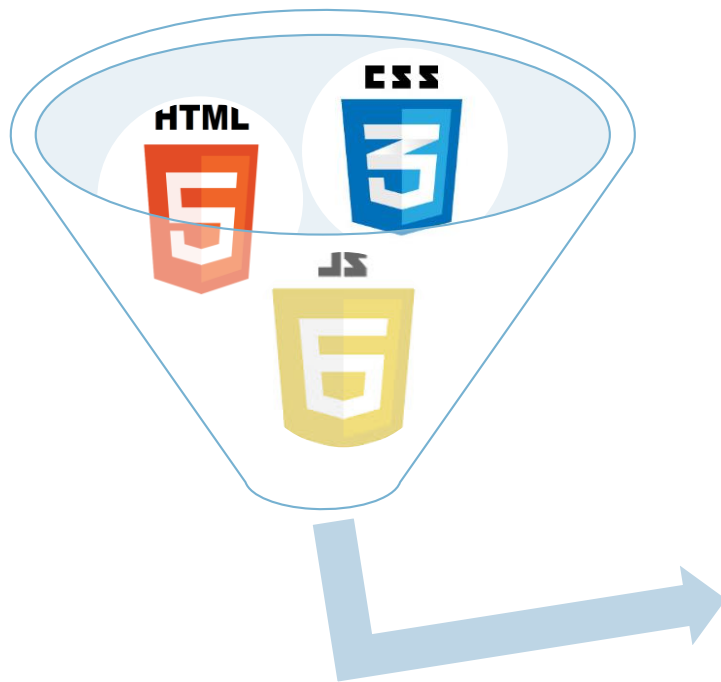
**Retomando...**



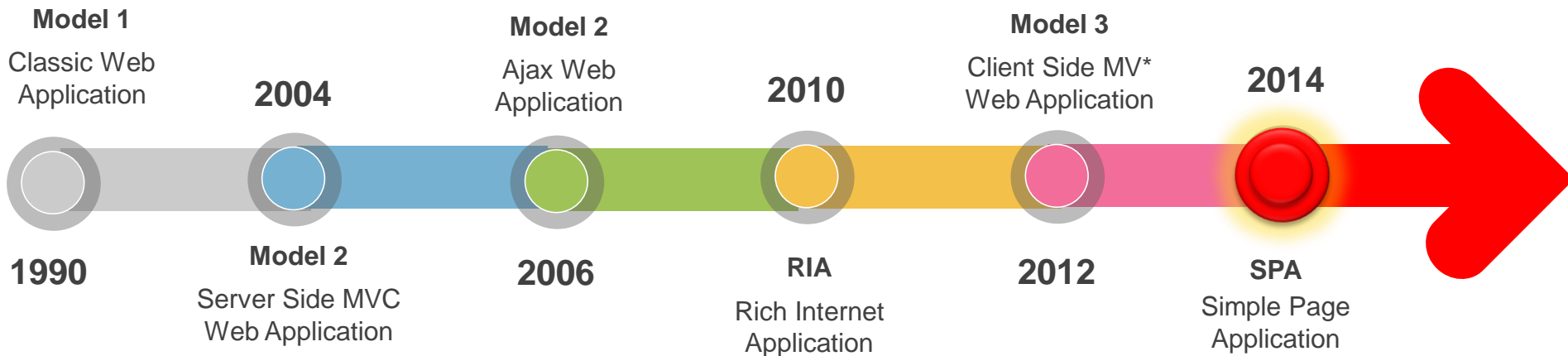
DEPARTAMENTO  
DE COMPUTACION

Facultad de Ciencias Exactas y Naturales - UBA

# Y ahora...¿listo?



# Evolución de las Aplicaciones Web



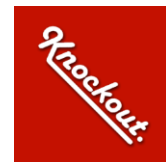
# Evolución

- La unión hace la fuerza...

- El fin de una época...



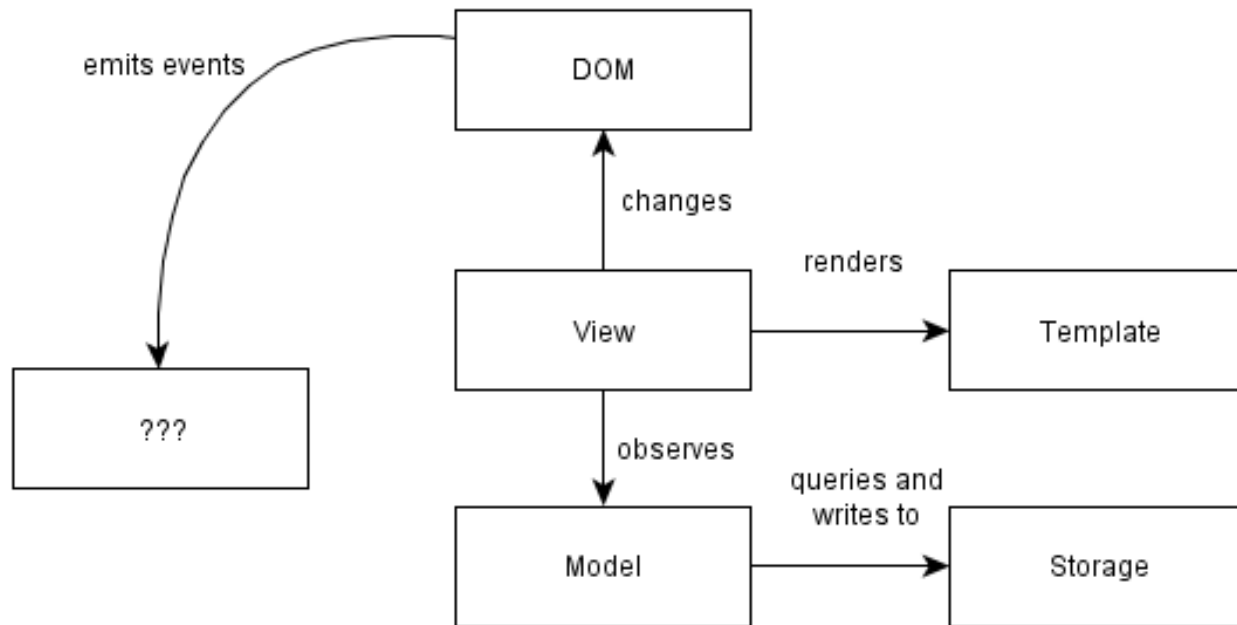
- Y el nacimiento de otros frameworks



# Single Page Applications

- Aplicaciones web que se ejecutan en una única página, logrando así una experiencia de usuario más cercana a una aplicación de escritorio.

# ¿Cómo describir estas aplicaciones?



- Y todo esto sin contar las llamadas al servidor...

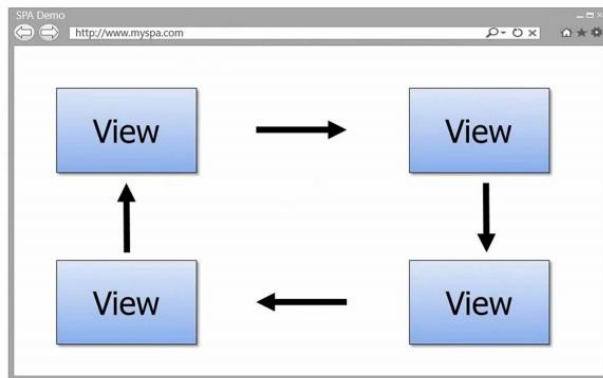
# Single Page Applications

- Aplicaciones web que se ejecutan en una única página, logrando así una experiencia de usuario más cercana a una aplicación de escritorio.



# Single Page Applications

- En una SPA el usuario no navega por un sistema de enlaces tradicionales si no que en su lugar, mediante el uso de JavaScript, Ajax, HTML5 o una combinación de las anteriores, se actualiza lo que el usuario ve siempre desde la misma página (sin cambiar de URL ni refrescar el contenido entero).



# Desafíos SPA

## Problemas a resolver

- Manipulación DOM
- Historia
- Module loading
- Routing
- Caching
- Diseño Orientado a Objetos
- Data Binding
- Ajax
- View loading

# Javascript++

- A medida que javascript obtuvo mayor popularidad y aceptación, surgen de él distintas librerías y frameworks.



# AngularJS



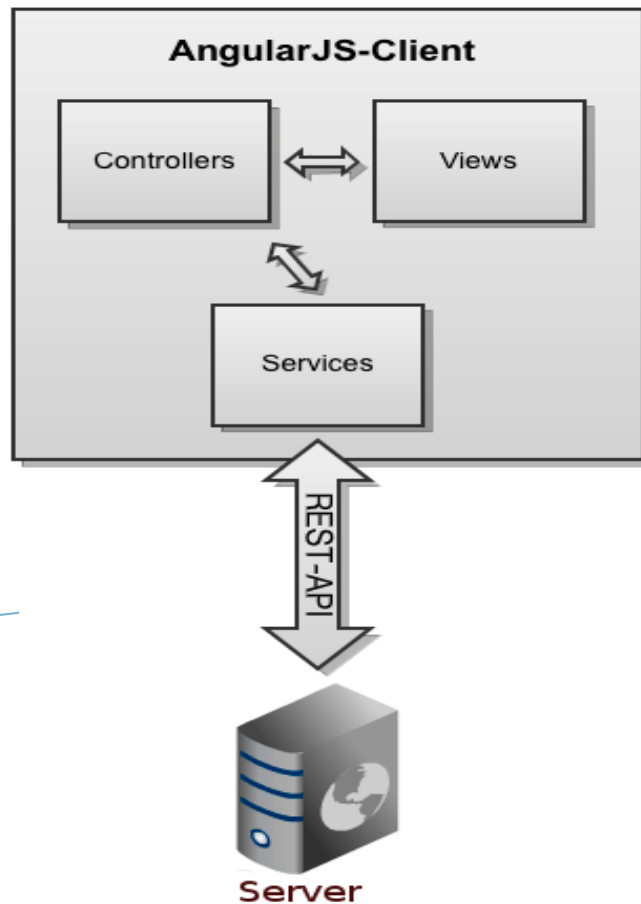
# ¿Por qué Angular?

- Extensión del HTML mediante directivas
- Generar las vistas de la una aplicación web
- Controles reutilizables
- Bindeo de datos
- Filtros incorporados (Fechas, Monedas)
- Modularización del código
- Manejo RESTfull
- Uso de AJAX
- Aliviar la carga del servidor

# Y el servidor?

- Angular está hecho para que se ejecute exclusivamente en el cliente, por lo que, es totalmente independiente del código del lado del backend. El servidor puede estar hecho con cualquier otra tecnología, ya sea Java, .Net, Rails, etc...
  - En particular, Angular utiliza objetos JSON para la obtención de datos a través de un servicio.
  - Idealmente el servidor deberá definir todos los servicios Rest que la aplicación Angular vaya a necesitar

# Estructura



# Hello World

```
<!DOCTYPE html >
<html>
  <head>
    <script type="text/javascript" src="js/angular.js"></script>
    <script type="text/javascript"
      src="js/helloangular.js"></script>
    <title>Hello world example</title>
  </head>
  <body ng-app="helloangular">
    <helloworld></helloworld>
  </body>
</html>
```



**Código Javascript de "helloangular.js"**

```
(function() {
  var app = angular.module("helloangular", []);

  app.directive("helloworld", function() {
    return { restrict: 'E', template: "<h1>
Hello world!</h1>" };
  });
})();
```



# Hello World

```
<!DOCTYPE html >
<html>
  <head>
    <script type="text/javascript" src="js/angular.js"></script>
    <script type="text/javascript"
      src="js/helloangular.js"></script>
    <title>Hello world example</title>
  </head>
  <body ng-app="helloangular">
    <helloworld></helloworld>
  </body>
</html>
```

*Código Javascript de "helloangular.js"*

```
(function(){
var app = angular.module("helloangular", []);

app.directive("helloworld", function() {
  return { restrict: 'E', template: "<h1>
Hello world!</h1>" };
});
})();
```

# Hello World

```
(function() {  
  var app = angular.module("helloangular", []);  
  
  app.directive("helloworld", function() {  
    return { restrict: 'E', template: "<h1> Hello world!</h1>" };  
  });  
})();
```

- Se define el nombre de la aplicación, asignándola a una variable, el corchete indica las dependencias a *inyectarse*, en este caso ninguna.
- Se define una directiva personalizada de la aplicación: *helloworld*.
  - Para definir que la directiva será un nuevo elemento HTML se la define como restrict: 'E', análogamente, un atributo, se lo define como restrict:'A'
  - Se describe el código HTML al que corresponderá el uso de la directiva (también se podría estar en un archivo aparte, pudiendo referenciarlo de la forma: templateUrl: "nombre")

# Elementos de AngularJS

- Directivas
- Expresiones
- Controllers
- Scope
- Filtros
- Servicios

# Directivas

- Las directivas permiten darle nuevos comportamientos al HTML

```
<!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

# Directivas

- Son el corazón del framework y definen gran parte de la funcionalidad de Angular.
- Las directivas propias se emplean como atributos de elementos html, de la siguiente forma

```
<tag ng-directiva="expresión"></tag>
```

- Principales directivas:
  - Ng-app: Define el nombre de la aplicación
  - Ng-repeat: Permite repetir una porción de código HTML
  - Ng-controller: Define un controlador
  - Ng-show: Muestra el elemento HTML en función de un booleano
  - Ng-src: Reemplaza al src estandar
  - Ng-route: Funciona de nexo entre directivas y servicios
  - Ng-view: Actualiza los html dados por ng route
  - Ng-model: Bindea los campos de un input a un controlador
  - Ng-bind: Bindea manualmente un elemento a una propiedad

# Views, Controllers & Scopes



- \$scope es la forma de comunicar/unir una view con su controllers.
- Similar al concepto de ViewModel

# Expresiones

- Angular permite la inserción de código Angular directamente en el HTML, utilizando `{{código}}` que luego es renderizado por el navegador.

```
<p>{{contador}}</p>
```

Imprimirá el valor de la variable *contador* del controller asociado a la porción de HTML.

# Ng-model

- Relaciona un elemento input, select, textarea (or custom form control) con una propiedad del scope asociado.

```
<script>
  angular.module('inputExample', [])
    .controller('ExampleController', ['$scope', function($scope) {

      $scope.val = '1'; }]);
</script>
```

Update input to see transitions when valid/invalid. Integer is a valid value.

```
<form name="testForm" ng-controller="ExampleController">
  <input ng-model="val" ng-pattern="/^\d+$/" name="anim"/>
</form>
```



# Ng-bind

- El atributo ngBind reemplazará el contenido de texto del elemento HTML especificado con el valor de una expresión dada.
- También actualizará el contenido cuando el valor de la expresión cambie.

```
<script>
  angular.module('bindExample', []) .controller('ExampleController', ['$scope', function($scope) {
    $scope.name = 'Whirled'; }]);
</script>
<div ng-controller="ExampleController"> Enter name:
  <input type="text" ng-model="name"><br> Hello
  <span ng-bind="name"></span>!
</div>
```

# Ng-model vs Ng-bind

- **ng-bind** es unidireccional:
  - \$scope --> view
  - Tiene un shortcut: {{ val }} inserta el valor de \$scope.val en el html.
- **ng-model** se utiliza dentro de formularios y en su caso es bi-direccional:
  - \$scope --> view & view --> \$scope

# Ng-repeat

- Instancia una plantilla una vez por cada elemento de una colección.
- Cada instancia de plantilla posee su propio scope.

# Ng-repeat

```
<div
```

```
  ng-init="friends = [ {name:'John', age:25, gender:'boy'}, {name:'Jessie', age:30, gender:'girl'}, {name:'Jo
  hanna', age:28, gender:'girl'}, {name:'Joy', age:15, gender:'girl'}, {name:'Mary', age:28, gender:'girl'}, {name:'Pet
  er', age:95, gender:'boy'}, {name:'Sebastian', age:50, gender:'boy'}, {name:'Erika', age:27, gender:'girl'}, {name:'P
  atrick', age:40, gender:'boy'}, {name:'Samantha', age:60, gender:'girl'} ]">
```

I have {{friends.length}} friends. They are:

```
<input type="search" ng-model="q" placeholder="filter friends..." />
```

```
  <ul class="example-animate-container">
```

```
    <li class="animate-repeat"
```

```
      ng-repeat="friend in friends | filter:q as results"> [{{$index + 1}}]
      {{friend.name}} who is {{friend.age}} years old. </li>
```

```
    <li class="animate-repeat" ng-if="results.length == 0">
      strong>No results found...</strong>
```

```
    </li>
```

```
  </ul>
```

```
</div>
```

# Ng-repeat

Variable	Type	Details
<code>\$index</code>	number	iterator offset of the repeated element (0..length-1)
<code>\$first</code>	boolean	true if the repeated element is first in the iterator.
<code>\$middle</code>	boolean	true if the repeated element is between the first and last in the iterator.
<code>\$last</code>	boolean	true if the repeated element is last in the iterator.
<code>\$even</code>	boolean	true if the iterator position <code>\$index</code> is even (otherwise false).
<code>\$odd</code>	boolean	true if the iterator position <code>\$index</code> is odd (otherwise false).

# Además

```
<div
```

```
  ng-init="friends = [ {name:'John', age:25, gender:'boy'}, {name:'Jessie', age:30, gender:'girl'}, {name:'Johanna', age:28, gender:'girl'}, {name:'Joy', age:15, gender:'girl'}, {name:'Mary', age:28, gender:'girl'}, {name:'Peter', age:95, gender:'boy'}, {name:'Sebastian', age:50, gender:'boy'}, {name:'Erika', age:27, gender:'girl'}, {name:'Patrick', age:40, gender:'boy'}, {name:'Samantha', age:60, gender:'girl'} ]">
```

I have {{friends.length}} friends. They are:

```
<input type="search" ng-model="q" placeholder="filter friends..." />
```

```
  <ul class="example-animate-container">
```

```
    <li class="animate-repeat"
```

```
      ng-repeat="friend in friends | filter:q as results" > [{{$index + 1}}]  
      {{friend.name}} who is {{friend.age}} years old. </li>
```

```
    <li class="animate-repeat" ng-if="results.length == 0">
```

```
      <strong>No results found...</strong>
```

```
    </li>
```

```
  </ul>
```

```
</div>
```

# Ng-if

- Elimina o inserta una porción del árbol DOM en función de la evaluación de una expresión.

Click me:

```
<input type="checkbox" ng-model="checked" ng-init="checked=true" />
```

```
<br/> Show when checked:
```

```
<span ng-if="checked" class="animate-if">
```

I'm removed when the checkbox is unchecked.

```
</span>
```

# Ng-show

- Muestra o esconde una porción del árbol DOM en función de la evaluación de una expresión.

```
<!--when $scope.myValue is truthy (element is visible)-->  
<div ng-show="myValue"></div>
```

```
<!--when $scope.myValue is falsy (element is hidden)-->  
<div ng-show="myValue" class="ng-hide"></div>
```



# @Override

- AngularJS modifica el comportamiento de muchos tags estandares:

a	input[radio]
form	input[text]
input	input[time]
input[checkbox]	input[url]
input[dateTimeLocal]	input[week]
input[date]	script
input[email]	select
input[number]	textarea

# Controladores

- Los controladores, son los módulos de una aplicación que definen un comportamiento específico para una porción del HTML.
- Todos los controladores se definen mediante la directiva `ng-controller="nombre"`. El archivo.js en el que esté definido puede o bien ser el mismo en el que se define la aplicación o bien ser un módulo distinto.

# Controladores

```

<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>

```

Basic controller

# Controladores

```

<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>

```

\$scope injected dynamically

# Controladores

- El \$scope esta siempre disponible dentro del fragmento donde esta definido (en este caso: desde el inicio del DIV hasta su cierre).

```
<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>
```



# Controladores

- El \$scope esta siempre disponible dentro del fragmento donde esta definido (en este caso: desde el inicio del DIV hasta su cierre).

```
<ul>
  <li data-ng-repeat="cust in customers">
    {{ cust.name }} - {{ cust.city }}
  </li>
</ul>
...
```

Access \$scope

```
<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>
```

\$scope injected  
dynamically

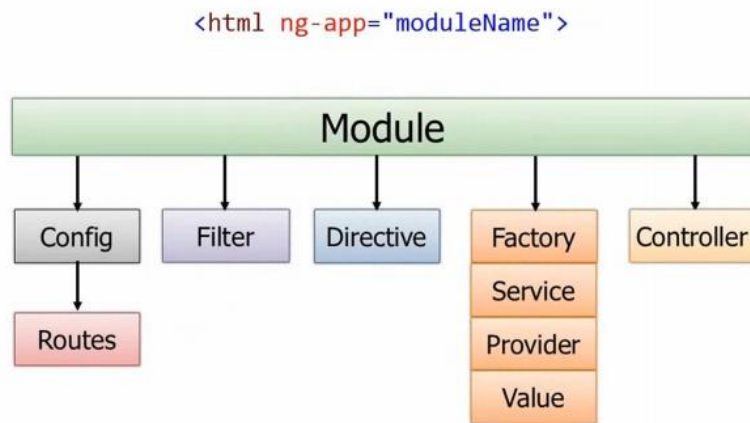
```
<script>
  function SimpleController($scope) {

    $scope.customers = [
      { name: 'Dave Jones', city: 'Phoenix' },
      { name: 'Jamie Riley', city: 'Atlanta' },
      { name: 'Heedy Wahlin', city: 'Chandler' },
      { name: 'Thomas Winter', city: 'Seattle' }
    ];

  }
</script>
```

# Módulos

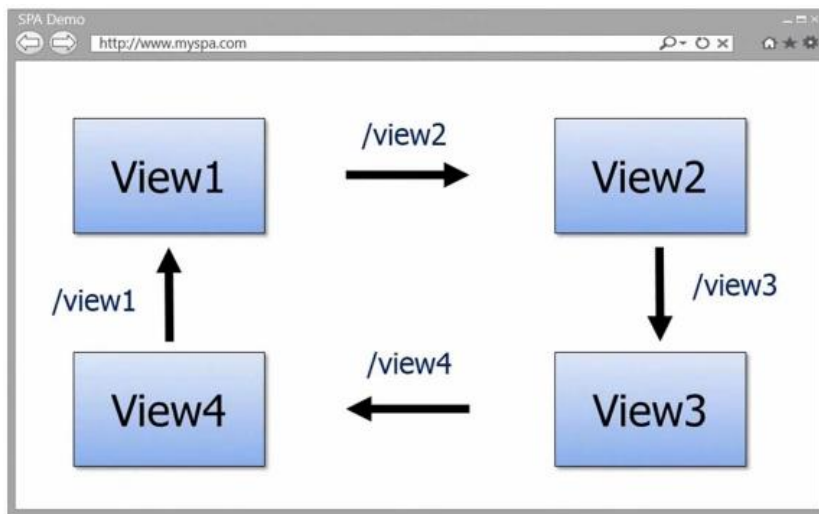
- Un módulo pueden pensarse como un contenedor para las diferentes partes de su aplicación - controladores, servicios, filtros, directivas, etc





# Routes

- Dado un modulo y un controller, necesitaremos alguna manera de diferenciar diferentes vistas de una misma pagina.





# Routes

- Qué hacer cuando el path es "/"
- Qué hacer cuando el path es "/partial2"

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

demoApp.config(function ($routeProvider) {
  $routeProvider
    .when('/',
      {
        controller: 'SimpleController',
        templateUrl: 'View1.html'
      })
    .when('/partial2',
      {
        controller: 'SimpleController',
        templateUrl: 'View2.html'
      })
    .otherwise({ redirectTo: '/' });
});
```

Define Module  
Routes

# Routes & ng-view

- En el modulo demoApp
- Definimos un placeholder para los fragmentos HTML

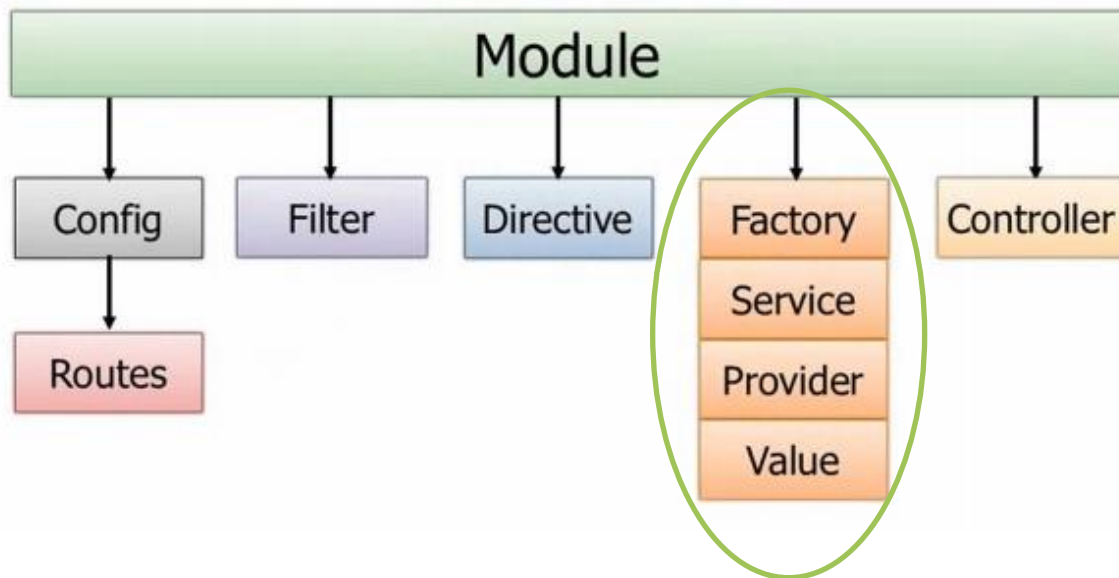
```
<div>
  <!-- Placeholder for views -->
  <div data-ng-view=""></div>
</div>
<script src="Scripts/angular.min.js"></script>

<script>
  var demoApp = angular.module('demoApp', []);

  demoApp.config(function ($routeProvider) {
    $routeProvider
      .when('/',
        {
          controller: 'SimpleController',
          templateUrl: 'Partials/View1.html'
        }
      )
  })
```

# Services & Providers & Factories

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



# Services & Providers & Factories

- Cada aplicación web esta compuesta por objetos que colaboran entre sí.
- Estos objetos necesitan ser instanciados y propagados a través de la aplicación.
- Angular provee un motor de inyeccion de dependencias que propaga estas dependencias de manera automaticamente.

# Services & Providers & Factories

- En resumen
  - Sirven para encapsular y reutilizar código
  - Se inyectan en los controladores
  - Son funcionalidades de la aplicación no relacionadas con el interfaz gráfico.
  - Nunca acceden a la página (DOM)
  - Son singleton

# \$http

- El servicio de \$http es un servicio que facilita la comunicación con los servidores HTTP remotos a través objeto XMLHttpRequest del navegador o mediante JSONP.

```
$http({method: 'GET', url: '/someUrl'}). success(function(data, status, headers, config) {  
    // this callback will be called asynchronously  
    // when the response is available  
}).  
error(function(data, status, headers, config) {  
    // called asynchronously if an error occurs  
    // or server returns response with an error status.  
});
```

# \$location

- El servicio \$location analiza la URL en la barra de direcciones del navegador (basado en la window.location) y deja la URL a disposición de su aplicación.
- Los cambios en la dirección URL en la barra de direcciones se reflejan en el servicio de localización y viceversa.

```
$location.path(expression)
```

# \$exceptionHandler

- Cualquier excepción no detectada en las expresiones angulares se delega a este servicio.
- La implementación por defecto simplemente delega en el \$ log.error que registra en la consola del navegador.



# \$exceptionHandler

```
var myApp = angular.module('myApp', ['ng']).provider({

    $exceptionHandler: function(){
        var handler = function(exception, cause) {
            alert(exception);
            //I need rootScope here
        };

        this.$get = function() {
            return handler;
        };
    }
});

myApp.controller('MyCtrl', function($scope, $exceptionHandler) {
    console.log($exceptionHandler);
    throw "Fatal error";
});
```

# Services Core

\$anchorScroll	\$location
\$animate	\$log
\$cacheFactory	\$parse
\$compile	\$q
\$controller	\$rootElement
\$document	\$rootScope
\$exceptionHandler	\$sce
\$filter	\$sceDelegate
\$http	\$templateCache
\$httpBackend	emplateRequest
\$interpolate	\$t
\$interval	\$timeout
\$locale	\$window

# Services & Providers & Factories

- **Services**

- Sintaxis : `module.service( 'serviceName', function );`  
Devuelve una instancia de la función (como alto orden)

- **Factories**

- Sintaxis : `module.factory( 'factoryName', function );`  
Devuelve el resultado (valor) despues de invocar la función.

- **Providers**

- Sintaxis: `module.provider( 'providerName', function );`  
Devuelve la implementación de la función `$get`

# Provider

## Provider Function



```
angular.module('myApp', [])
  .provider('myPro', function() {
    console.log('myApp => Create provider => retrun object with $get');
    return {
      isTrue: false,
      $get: function b($http) {
        var self = this;
        console.log('myApp => my Provider $get => retrun func');
        return function c(msg) {
          console.log(msg + " isTrue: " + self.isTrue);
        };
      }
    };
  });

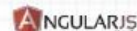
app.config(function (myProProvider) {
  myProProvider.isTrue = true;
  console.log('myApp --> config');
});
```

**Step 1:** Invoke the function before the config stage. No args.

**Step 2:** This object will be available in config stage as injectable service. The name is "myProProvider".

**Step 3:** \$get func is a factory func for the service, invoke only if needed and only once. Available after the config stage.

**Step 4:** The injectable service.



# Validaciones

- Angular trae por defecto varias validaciones que son útiles a la hora de implementar formularios.
- Como contrapartida de esto, es fuertemente recomendado desactivar (`<form novalidate/>`) la validación por defecto de HTML5 para que Angular pueda trabajar sin problemas.

# Validaciones

- Validan el contenido de los `<input>`, no del modelo en el `$scope`
- Se compone de directivas
- Clases CSS
- JavaScript Formulario
- JavaScript `<input>`

# Directivas en <input>

- ❑ **min**: Valor mínimo de un número
- ❑ **max**: Valor máximo de un número
- ❑ **ngRequired**: Si es requerido
- ❑ **ngMinlength**: Tamaño mínimo de un String
- ❑ **ngMaxlength**: Tamaño máximo de un String
- ❑ **ngPattern**: Expresión regular que debe cumplir

# Clases CSS

- AngularJS establece estas clases en el tag `<input>`
  - **ng-valid**: Cuando es válido
  - **ng-invalid**: Cuando es inválido
  - **ng-pristine**: Cuando NO se ha modificado su valor
  - **ng-dirty**: Cuando ha sido modificado su valor



# Javascript Formulario

- **nombreFormulario.\$pristine**: Vale 'true' si no se ha modificado ningún valor.
- **nombreFormulario.\$dirty**: Valor 'true' si se ha modificado algún valor.
- **nombreFormulario.\$valid**: Vale 'true' si todos los campos del formulario son válidos.
- **nombreFormulario.\$invalid**: Vale 'true' si algún campo del formulario es inválido.

# Javascript Formulario

- `nombreFormulario.nombreInput.$error.nombreError` :  
Vale 'true' si ese error ha fallado.

Siendo `nombreError`:

- min
- max
- required
- minlength
- Etc.

# Filtros

- Angular provee un set de filtros útiles para presentar información de la forma que queramos.
- Estos filtros se utilizan de la forma:  

```
{ expresion | filtro }
```
- Angular también permite la definición de filtros customizados.

# Extensiones de Angular

- Existe una gran comunidad que apoya y trata de extender Angular para hacerlo aún más versátil.
- <http://ngmodules.org/>

The screenshot shows the Angular Modules website interface. At the top, there's a dark header with the text "ANGULAR MODULES" and "FIND MODULES FOR ANGULARJS". A search bar and a "Sign in with Github" button are on the right. Below the header, there are tabs for "Modules", "Submit a Module", and "Tags". On the right side of the main content area, it says "841 modules".

On the left, there are three sorting options: "Popular", "Newest", and "A-Z". The main content area displays three module cards:

- abourget-angular** — abourget: Re-usable AngularJS tools and directives, mostly \$watch fighters for now. 1 person uses it. Includes a "watch" button.
- ac-fancy-input** — alexcppns: Enhanced search input with 'placeholder animation' and categorized suggestions for AngularJS. There is no jQuery dependency, just pure angularJS. 1 person uses it. Includes tags like "typeahead", "input", "component", "fancyinput", "animated text", "placeholder animation", "nested suggestions", and "input directive".
- activerecord** — bfanger: A Backbone.Model inspired model layer for AngularJS. 2 people use it. Includes tags like "backbone", "REST", "model", "activerecord", and "backbone.model".

On the right side, there's a green "Add a Module" button and a "Popular Tags" section with a grid of tags and their counts:

- directive (361)
- service (78)
- api (44)
- bootstrap (43)
- angular (41)
- ui (33)
- filter (27)
- input (24)
- mobile (22)
- AngularJS (21)
- i18n (21)
- form (21)
- upload (19)
- table (18)
- jQuery (16)
- tree (16)
- validation (16)
- forms (16)
- select (16)
- http (16)
- module (16)
- model (15)
- scroll (15)
- directives (14)
- grid (14)

# Extensiones

- ¿Qué significa incorporar una extensión?

```
<script src="vendor/angular-growl/angular-growl.js"></script>
```

```
/* Main module of the application.
```

```
*/
```

```
(function() {
```

```
    var app = angular.module('intelliTrackApp', [ 'ngResource', 'ngRoute', 'ngSanitize', 'ngAnimate', 'ngAria', 'ngMessages', 'ngTable', 'http-auth-interceptor', 'loginModule', 'contentModule', 'localStorageModule', 'ngResource', 'angular-growl', 'ui.bootstrap', 'pascalprecht.translate' ])
```

```
    app.config([ 'growlProvider', function(growlProvider) {
```

```
    } ]);
```

```
        app.config([ 'growlProvider', function(growlProvider) {
```

```
            growlProvider.globalTimeToLive(5000);
```

```
        'angularFileUpload', 'ngTable', 'http-auth-interceptor',
```

```
        'loginModule', 'contentModule',
```

```
        'localStorageModule', 'ngResource',
```

```
        'angular-growl', 'ui.bootstrap',
```

```
        'pascalprecht.translate' ])
```

```
    app.config([ 'growlProvider', function(growlProvider) {
```

```
        'angularFileUpload', 'ngTable', 'http-auth-interceptor',
```

```
        'loginModule', 'contentModule',
```

```
        'localStorageModule', 'ngResource',
```

```
        'angular-growl', 'ui.bootstrap',
```

```
        'pascalprecht.translate' ])
```

```
    app.config([ 'growlProvider', function(growlProvider) {
```

```
        'angularFileUpload', 'ngTable', 'http-auth-interceptor',
```

```
        'loginModule', 'contentModule',
```

```
        'localStorageModule', 'ngResource',
```

```
        'angular-growl', 'ui.bootstrap',
```

```
        'pascalprecht.translate' ])
```

```
    app.config([ 'growlProvider', function(growlProvider) {
```

```
    .module('intelliTrackApp')
```

```
    .controller('SectorCtrl',
```

```
        function($scope, $location, $http, ngTableParams, $rootScope,
```

```
            growl, dialogs) {
```

```
                $scope.paginaActual = 1;
```

```
                $scope.paginaActual = 1;
```

```
                $scope.registrosTotales = 0;
```

```
        if (!codigoOk(response)) {
```

```
            var descripcion = response['desc'];
```

```
            growl.error(descripcion);
```

```
        }
```

## Angular Table directive

part of Bazalt CMS

## Project GitHub

## Who is using ngTable?

Forks 302

If you have interesting example of using `ng-table`, just create it on <http://plnkr.co/> and email me [esvit666@gmail.com](mailto:esvit666@gmail.com), and add your example on this page.

- ## #16: Editable demo



## AJAX Data Loading

Code

Edit

Preview

Reload

Clear sorting

Name

Age

2

Enos false

34

Enos false

34

Enos false

34

Enos false

34

Jacob false

27

Jacob false

27

Jacob false

27

Jacob false

27

Moroni false

50

Nephi false

29

10	25	50	100
----	----	----	-----

$\kappa$	1	2	$\infty$
----------	---	---	----------

# angularFileUpload

## Angular file upload Demo

Visit [angular-file-upload](#) on github

http method: ☒ post ☐ put

How to upload:

- ☒ Multipart/form-data upload using \$upload.upload() service cross browser
- ☐ File binary content with the file type as Content-Type header using \$upload.http() service

The second option could be used to upload files to [CouchDB](#), [imgur](#), etc... for HTML5 FileReader browsers.

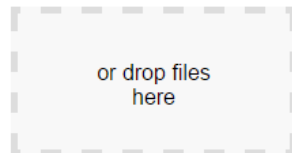
myModel:  model object to be sent with the file.

choose a single file:  Ningún archivo seleccionado

or multiple files:  Ningún archivo seleccionado

or only images:  Ningún archivo seleccionado

You can have any element as an upload button: file



☒ Upload right away

Progress:

# checklist-model

## ☑ Checklist-model

AngularJS directive for list of checkboxes

### Why this is needed?

In Angular one checkbox `<input type="checkbox" ng-model="...">` is linked with one model.

But in practice we usually want one model to store array of checked values from several checkboxes.

**Checklist-model** solves that task without additional code in controller.

You should play with attributes of `<input type="checkbox">` tag:

1. set `checklist-model` instead of `ng-model`
2. set `checklist-value` - what should be picked as array item

Please, try out demos below:

### Array of primitives

demo

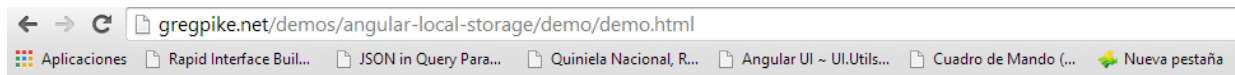
- ☒ guest
- ☒ user
- ☒ customer
- ☐ admin

user.roles

```
[
  "user",
  "guest",
  "customer"
]
```



# angularLocalStorage

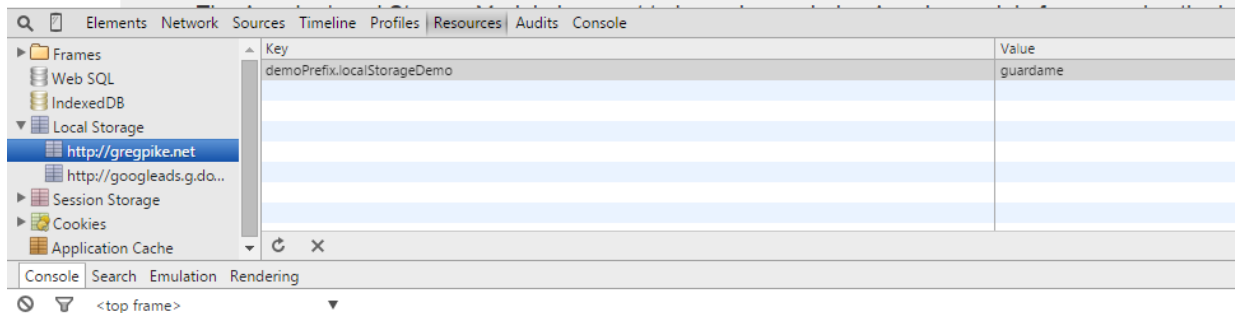


## Angular Local Storage

Give it a try

guardame

— Local storage value



# dialogs.main

codepen.io/m-e-conroy/pen/ALsdF

Aplicaciones Rapid Interface Buil... JSON in Query Para... Quiniela Nacional, R... Angular UI ~ UI.Util... Cuadro de Mando (... Nueva pestaña jax rs - JAX-RS, RestE... java - logout user vi...

**C O D E P E N** Fork Share Editor Log In

HTML

```

1 <html ng-app="modalTest">
2   <head>
3     <script
4       src="https://ajax.googleapis.com/ajax/libs/angularjs/1.1.5/angular.min.js"></script>
5     <script src="http://angular-ui.github.io/bootstrap/ui-bootstrap-tpls-0.6.0.js" type="text/javascript"></script>
6     <script src="http://m-e-conroy.github.io/angular-dialog-service/javascripts/dialogs.min.js" type="text/javascript"></script>
7   </head>
8   <body ng-controller="dialogServiceTest" class="nada">

```

CSS

```

1 /* Fix for Bootstrap 3 with Angular UI Bootstrap */
2
3 .modal {
4   display: block;
5 }
6
7 /* Custom dialog/modal headers */
8
9 .dialog-header-error { background-color: #d2322d; }
10 .dialog-header-wait { background-color: #428bca; }
11 .dialog-header-notify { background-color: #eeeeec; }

```

JS

```

1 angular.module('modalTest',
2   ['ui.bootstrap', 'dialogs'])
3 .controller('dialogServiceTest', function(
4   rootScope, $timeout, $dialogs){
5
6   $scope.confirmed = 'You have yet to be
7   confirmed!';
8   $scope.name = 'Your name here.';
9
10  $scope.launch = function(which){
11    var dlg = null;
12    switch(which){

```

## Bootstrap 3 & AngularJS Dialog/Modals

Error Dialog

Wait Dialog

Notify Dialog

Confirm Dialog

Custom Dialog

From Confirm Dialog: Shame on you for not thinking this is awesome!

# dialogs.main

Aplicaciones Rapid Interface Buil... JSON in Query Para... Quiniela Nacional, R... Angular UI ~ UI.Util... Cuadro de Mando (... Nueva pestaña jax rs - JAX-RS, RestE... java - logout user vi...

**CODPEN** Fork Share Editor Log In

**HTML**

```

1 <html ng-app="modalTest">
2 <head>
3   <script
4     src="https://ajax.googleapis.com/ajax/libs/angularjs/1.1.5/angular.min.js"></script>
5   <script src="http://angular-
6     ui.github.io/bootstrap/ui-bootstrap-tpls-0.6.0.js" type="text/javascript"></script>
7   <script src="http://m-e-
8     conroy.github.io/angular-dialog-
9     service/javascripts/dialogs.min.js"
10    type="text/javascript"></script>
11 </head>
12 <body ng-controller="dialogServiceTest"
13    class="nada">

```

**CSS**

```

1 /* Fix for Bootstrap 3 with Angular UI Bootstrap
2 */
3 .modal {
4   display: block;
5 }
6
7 /* Custom dialog/modal headers */
8
9 .dialog-header-error { background-color: #d2322d;
10 }
11 .dialog-header-wait { background-color: #428bca;
12 }
13 .dialog-header-notify { background-color:
14   #eeeeec; }

```

**JS**

```

1 angular.module('modalTest',
2   ['ui.bootstrap', 'dialogs'])
3 .controller('dialogServiceTest',function(
4   rootScope,$timeout,$dialogs){
5   $scope.confirmed = 'You have yet to be
6   confirmed!';
7   $scope.name = 'Your name here.';
8
9   $scope.launch = function(which){
10     var dlg = null;
11     switch(which){
12       // Error Dialog
13       case 'error':
14         dlg = $dialogs.error('This is m

```

**Bootstrap 3 & AngularJS D**

Error Dialog Wait Dialog Notify Dialog Cor

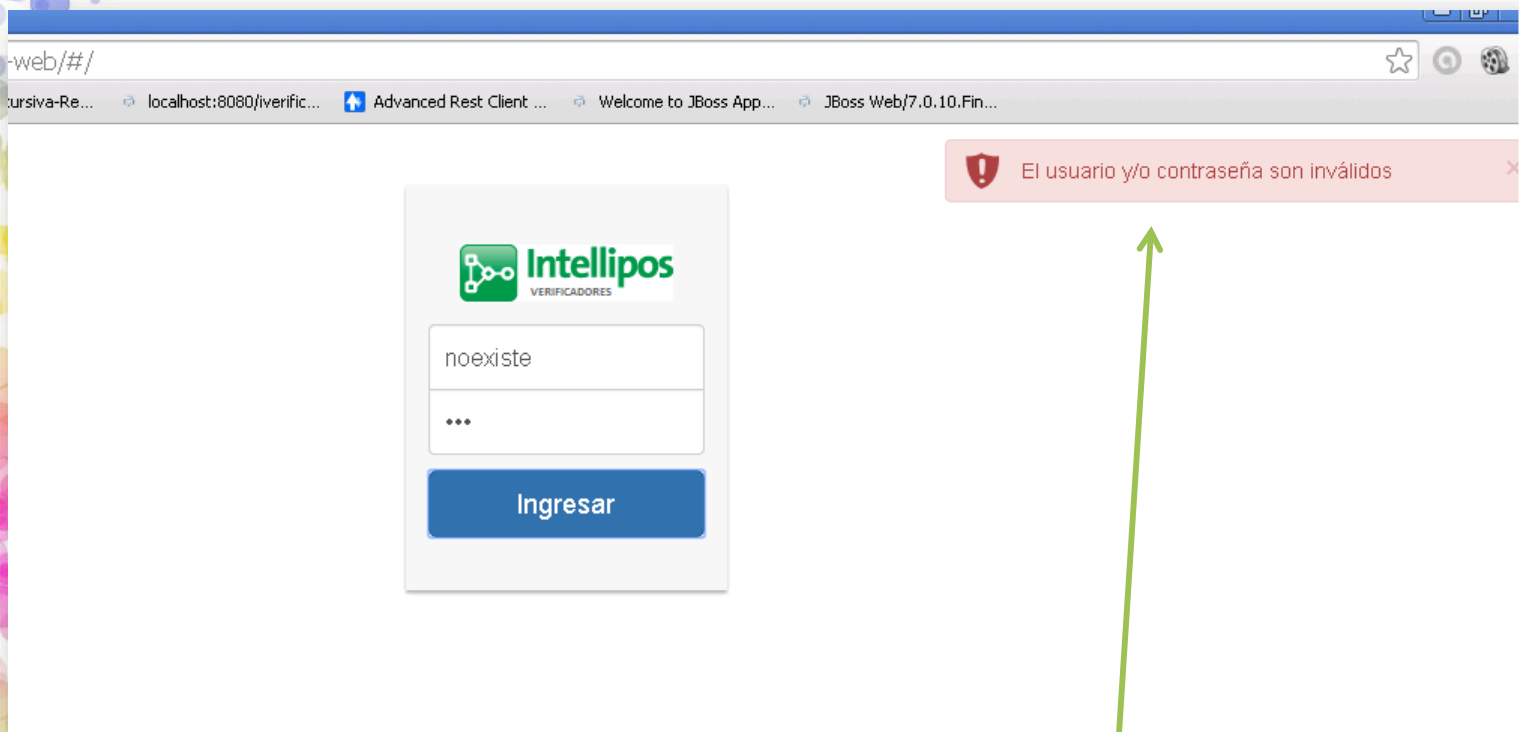
From Confirm Dialog: Shame on you for not thinking this

☒ Please Confirm

Is this awesome or what?

Yes No

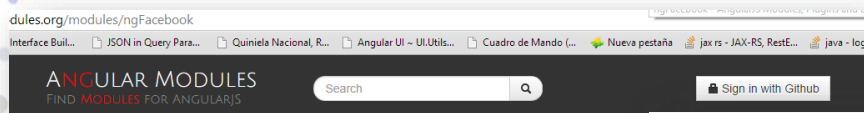
# angular-growl



```
sponse) {  
  
];  
velPermisos'];  
s'];  
  
ged', true);  
, token);  
ario', usuario);  
, nivelPermisos);  
, acciones);  
};
```

```
$location.path('/sincronizacion');  
$window.location.reload();  
} else {  
  growl.error(response['desc']);  
}
```

# Para todos los gustos



## ngFacebook

Angular facebook service

Homepage:

Author: GoDisco

Submitted by: AlmogBaku

## ng-nvd3

## ng-nvd3

A directive for nvd3 in AngularJS

[View the Project on GitHub](#)

## Angular Treeview

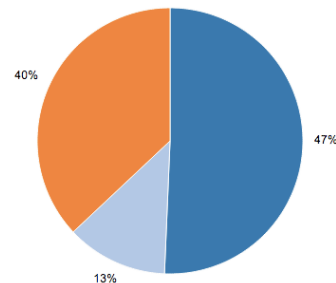
Pure [AngularJS](#) based tree menu directive.

- User
  - subUser1
  - subUser2
- Admin
  - subAdmin1
  - subAdmin2
    - subAdmin2-1
      - subAdmin2-1-1
      - subAdmin2-1-2
- Guest
  - subGuest1
  - subGuest2

## API

Pie Chart

Worse Same Better



How to use?

Angular-directive in html:  
`<nvd3-pie-chart ...></nvd3-pie-chart>`

Attributes:

# Un mundo feliz?

- Es muy fácil, a medida que se acomplejiza la vista, terminar con un código spaghetti, mezclándose javascripts, directivas, expresiones, html y demás yerbas, terminando por hacer más compleja la aplicación de lo
- que debería ser. Angular permite y promociona una modularización intensa, la cual debería ser aplicada.

# Un mundo feliz?

- Al igual que cualquier herramienta, a mayor poder y flexibilidad, mayor complejidad. Angular no es la excepción y su uso comprende una curva de aprendizaje superior a la mayoría de los restantes frameworks javascripts.

# Seguridad

- En este punto el problema es la naturaleza misma del lenguaje sobre el que está hecho Angular: Javascript.
- Es muy importante tener en cuenta que cualquier característica de seguridad en la aplicación debe ser repetida en el servidor.
- El código javascript que se ejecuta en el cliente puede ser perfectamente modificado por el cliente, por lo que se debe pensar en términos de seguridad en profundidad y siempre verificar permisos y cualquier aspecto de seguridad en el servidor.



# Seguridad: hay herramientas

- Extensión que intercepta todos los requerimientos HTTP para validar su origen, permisos y autenticación.

ANGULAR MODULES  
FIND MODULES FOR ANGULARJS

Sign in with Github

Modules
Submit a Module
Tags
Blog
Give Feedback

## http-auth-interceptor

Authentication management for AngularJS

Github
Download

Homepage: <http://witoldsz.github.com/angular-http-auth/>
69 people use it
I use it

Author: [witoldsz](#)
Submitted by: [jimrhoskins](#)

authentication

README
Discussion

### HTTP Auth Interceptor Module for AngularJS

This is the implementation of the concept described in [Authentication in AngularJS \(or similar\) based application](#).


Launch demo [here](#) or switch to [gh-pages](#) branch for source code of the demo.

# AngularJS Batarang

/chrome.google.com/webstore/detail/angularjs-batarang/ighndmenidhpncoggitloacoatjmptk

interface Buil... JSON in Query Para... Quiniela Nacional, R... Angular UI ~ UI.Util... Cuadro de Mando (... Nueva pestaña jax rs - JAX-RS, RestE... java - logo

Mostrar: Todo martin.urtasun@recursiva.com

 **AngularJS Batarang** + GRATIS

★★★★★ (300) [Herramientas para desarrolladores](#) [de AngularJS](#) 170.333 usuarios

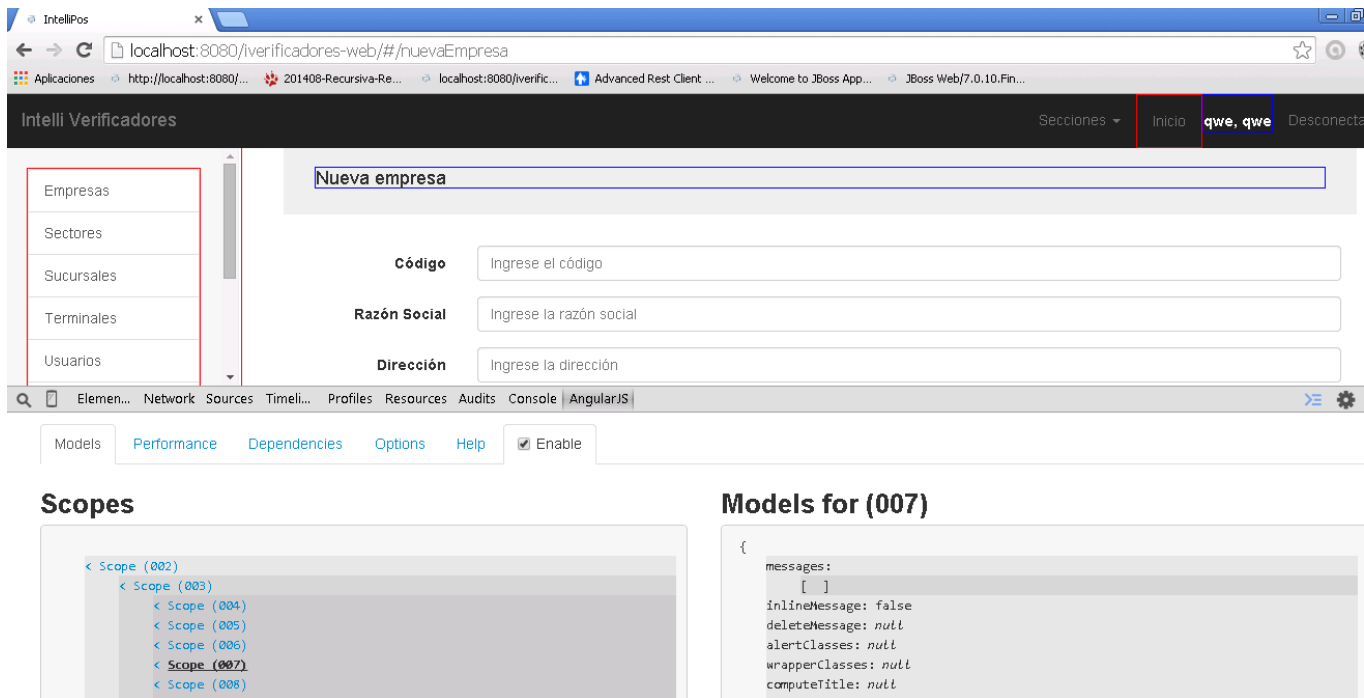
DESCRIPCIÓN GENERAL DETALLES OPINIONES RELACIONADOS +1 935

**AngularJS Batarang**

Extends the Developer Tools, adding tools for debugging and profiling AngularJS applications.

0:00 / 8:18 YouTube

# AngularJS Batarang



The screenshot displays a web browser window with the URL `localhost:8080/verificadores-web/#/nuevaEmpresa`. The application, titled 'Intelli Verificadores', features a sidebar menu with options: Empresas, Sectores, Sucursales, Terminales, and Usuarios. The 'Empresas' section is active, showing a form for 'Nueva empresa' with input fields for 'Código', 'Razón Social', and 'Dirección'. Below the browser window, the AngularJS Batarang extension interface is shown, including a 'Models' tab and a 'Scopes' tree. The 'Scopes' tree lists various scopes, with 'Scope (007)' selected. The 'Models for (007)' panel displays a JSON object representing the selected scope's data.

```

{
  messages:
    [ ]
  inlineMessage: false
  deleteMessage: null
  alertClasses: null
  wrapperClasses: null
  computeTitle: null
}

```



# Bootstrap

# ¿Qué es Bootstrap?



- **Twitter Bootstrap** es un framework o conjunto de herramientas de software libre para diseño de sitios y aplicaciones web. Contiene plantillas de diseño con tipografía, formularios, botones, cuadros, menús de navegación y otros elementos de diseño basado en HTML y CSS, así como, extensiones de JavaScript opcionales adicionales.
- En agosto del 2011, Twitter liberó a Bootstrap como código abierto.

# Responsive Web Design

- El objetivo es brindar una experiencia visual óptima para la diversa variedad de dispositivos actualmente existente.



# Elementos del RWD

- Fluid Grid
- Resizable Images
- Media Queries

# 960 Grid System

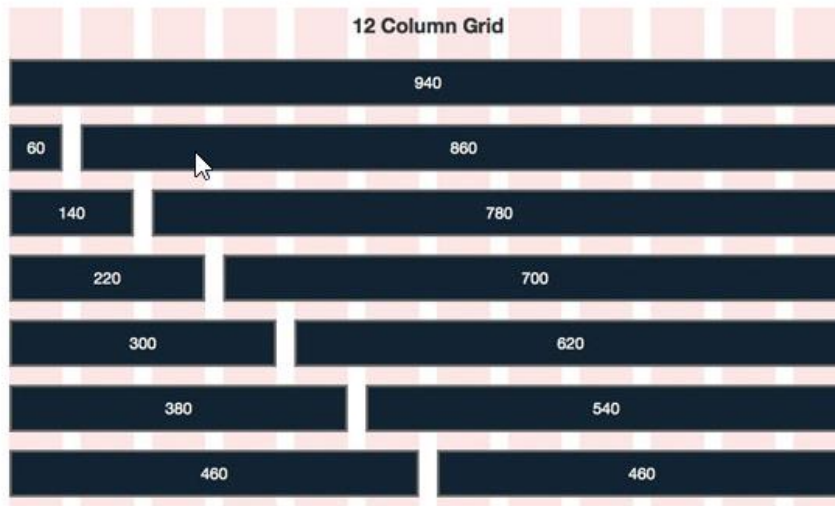


- El sistema de cuadrícula 960 es simplemente una manera de diseñar sitios web usando una rejilla de 960 píxeles de ancho. La razón es 960 píxeles de ancho se debe a que el número 960 simplifica las divisiones en columnas y los márgenes de la plantilla de diseño.
- El 960 GS viene en dos variantes principales: una cuadrícula de 12 columnas y una cuadrícula de 16 columnas. También incluye una versión de 24 columnas para diseños web en los que necesitemos incluir una mayor densidad de contenidos.



# 960 Grid System

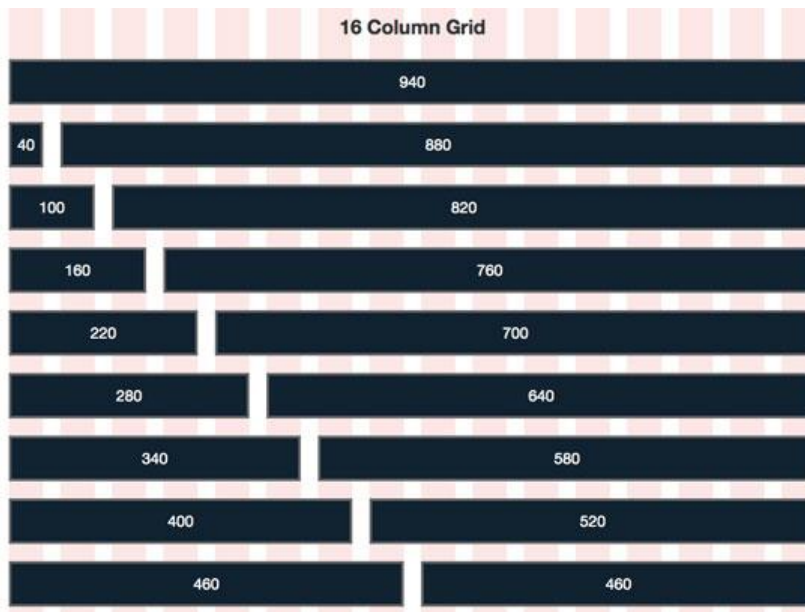
- En la versión de 12 columnas, la columna más estrecha es de 60 píxeles de ancho, incrementándose después en 80 píxeles. De esta forma, los anchos de las columnas disponibles son: 60, 140, 220, 300, 380, 460, 540, 620, 700, 780, 860 y 940.



960 Grid System: disposición en 12 columnas

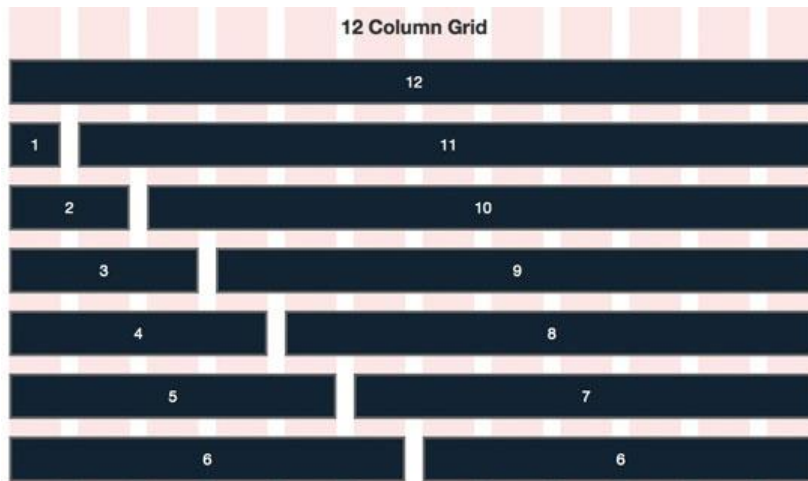
# 960 Grid System

- En la versión en 16 columnas, la columna más estrecha es de 40 píxeles de ancho y después se incrementa en 60 píxeles. Así que el ancho de las columnas disponibles son: 40, 100, 160, 220, 280, 340, 400, 460, 520, 580, 640, 700, 760, 820, 880 y 940.



# 960 Grid System

- Cada una de las barras horizontales podemos relacionarla con una clase CSS en el sistema 960 Grid, de forma que le asignaremos una capa (div) con el ancho adecuado.
- La nomenclatura de cada uno de los elementos del CSS será acorde con el ancho del elemento de la rejilla, de forma que la clase CSS `grid_1` representará a la columna más estrecha y la clase `grid_12` será la que aplicaremos a las columna más ancha para un sistema de 12 columnas.



# 960 Grid System

- Por ejemplo, si utilizamos el sistema de 12 columnas, simplemente tendremos que utilizar la clase `grid_4` para incluir uniformemente tres capas de texto que ocupen el diseño de lado a lado:



# Resizable Images

- Con CSS3 es posible redimensionar las imágenes dinámicamente.

```
img {  
    max-width: 100%;  
    height: auto;  
}
```

- Esto se complementa con:
  - Relative Font Size
  - Relative Margin
  - Etc...

# Media Queries

- Es un módulo CSS3 que permite adaptar la representación del contenido a características del dispositivo como la resolución de pantalla (por ejemplo, un smartphone frente a pantallas de alta definición) o la presencia de características de accesibilidad como el braille.

```
<!-- CSS media query on a link element -->
<link rel="stylesheet" media="(max-width: 800px)" href="example.css" />

<!-- CSS media query within a style sheet -->
<style> @media (max-width: 600px)
{
    .facet_sidebar { display: none; }
}
</style>
```

# Media Queries

- Media Types
  - braille
  - embossed
  - handheld
  - print
  - projection
  - screen
  - speech
  - tty
  - tv

## Atributos Media

La siguiente tabla contiene los atributos media recogidos de la última recomendación de la W3C para media queries

Atributo	Valor	Min/Max
color	integer	si
color-index	integer	si
device-aspect-ratio	integer/integer	si
device-height	length	si
device-width	length	si
grid	integer	no
height	length	si
monochrome	integer	si
resolution	resolution ("dpi" or "dpcm")	si
scan	"progressive" or "interlaced"	no
width	length	si



# Ejemplo



## Granada, España

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce a ligula massa. Donec consequat, risus nec viverra condimentum, elit vel dignissim du, id congue sapien leo tincidunt est. Mauris consectetur tempus lorem id aliquet. Proin eu faucibus massa. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In magna ligula, ornare sed posuere faucibus, consequat ac lacus. Fusce sodales fermentum nibh, a imperdiet nisi bibendum eget. Donec gravida iaculis sapien eu consectetur. Curabitur id augue augue. Nam mauris urna, suscipit eget faucibus sit amet, mollis vitae felis. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Mauris ipsum lectus, imperdiet id aliquam nec, vulputate vitae mauris. Integer gravida, neque eu placerat egestas, urna metus bibendum nisl, quis congue felis velit id diam. Quisque non tortor at turpis facilisis placerat quis sed felis. Ut eget ipsum dolor, id lacinia leo. Vivamus vitae blandit libero. Integer ultricies gravida leo quis lobortis. Morbi ultricies risus vulputate magna dignissim sed ultricies arcu tristique. Sed non facilisis sapien.



```
}  
body{  
  background:#f5f5f5;  
  font-family: Arial, sans-serif;  
  font-size:13px;  
  line-height:1.6em;  
  color:#444;  
}  
  
p{  
  margin:15px 0;  
}  
  
h2{  
  margin-top:20px;  
}  
  
#container{  
  background:#fff;  
  border-left:1px #ddd solid;  
  border-right:1px #ddd solid;  
  border-bottom:1px #ddd solid;  
  width:600px;  
  margin:0 auto;  
}  
  
header h1 a{  
  text-indent:-9999px;  
  display:block;  
  width:600px;  
  height:98px;  
  background:url(image-med.jpg) no-repeat 50% 0;  
}  
  
#content{  
  padding:0 15px;  
}
```



# Ejemplo



## Granada, España

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Fusce a ligula massa. Donec consequat, nisl nec viverra condimentum, elit vel dignissim du, id congue sapien leo tincidunt est. Mauris consectetur tempus lorem id aliquet. Proin eu faucibus massa. Lorem ipsum dolor sit amet, consectetur adipiscing elit. In magna ligula, ornare sed posuere faucibus, consequat ac lacus. Fusce sodales fermentum nibh, a imperdiet nisi bibendum eget. Donec gravida iaculis. Curabitur id augue augue. Nam massa urna, euismod eget faucibus sit

```
@media screen and (max-width:320px){
  img {
    max-width:200px;
  }

  #container{
    width:auto;
  }

  header h1 a{
    width:auto;
    height:52px;
    background:url(image-small.jpg) no-repeat 50% 0;
  }
}
```

```
body{
  background:#f5f5f5;
  font-family: Arial, sans-serif;
  font-size:13px;
  line-height:1.6em;
  color:#444;
}
```

```
p{
  margin:15px 0;
}
```

```
h2{
  margin-top:20px;
}
```

```
#container{
  background-color:#f5f5f5;
}
```

```
@media screen and (min-width:1200px){
  img {
    max-width:1000px;
  }

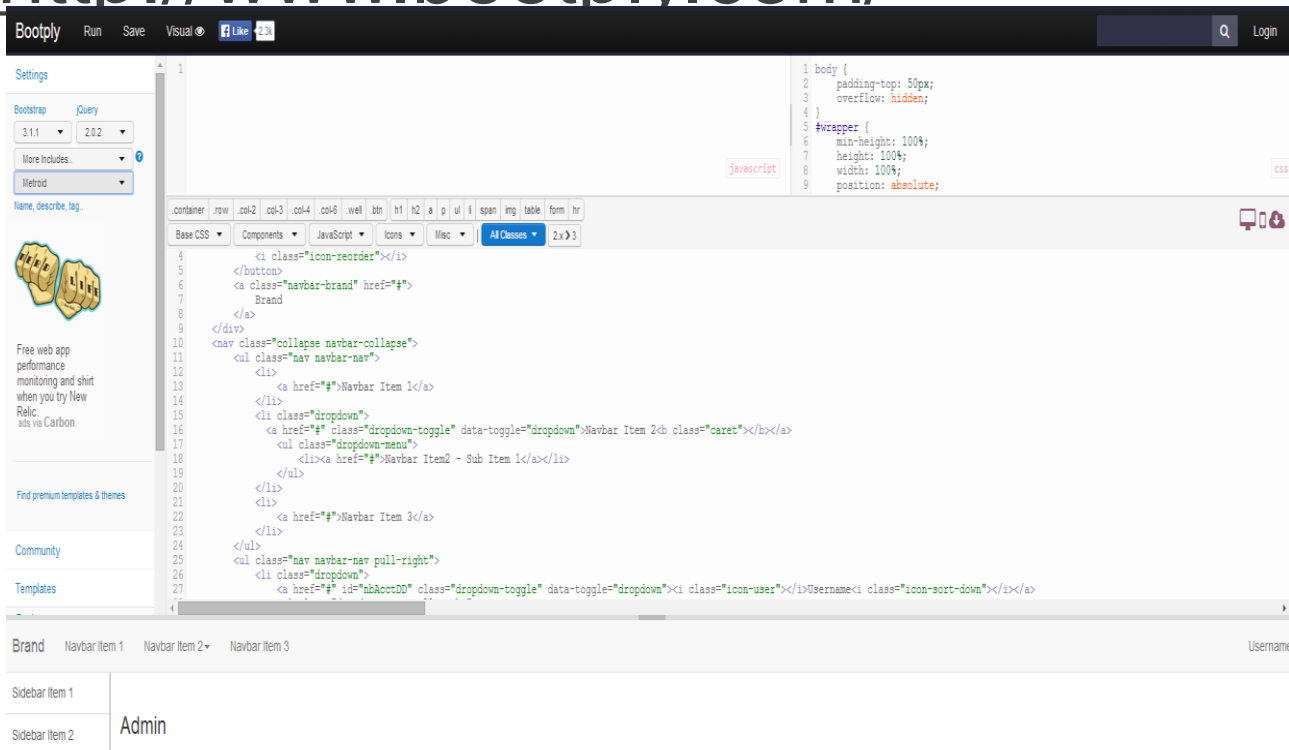
  #container{
    width:1100px;
  }

  header h1 a{
    width:1100px;
    height:180px;
    background:url(image.jpg) no-repeat 0 0;
  }
}
```

```
content{
  padding:0 15px;
}
```

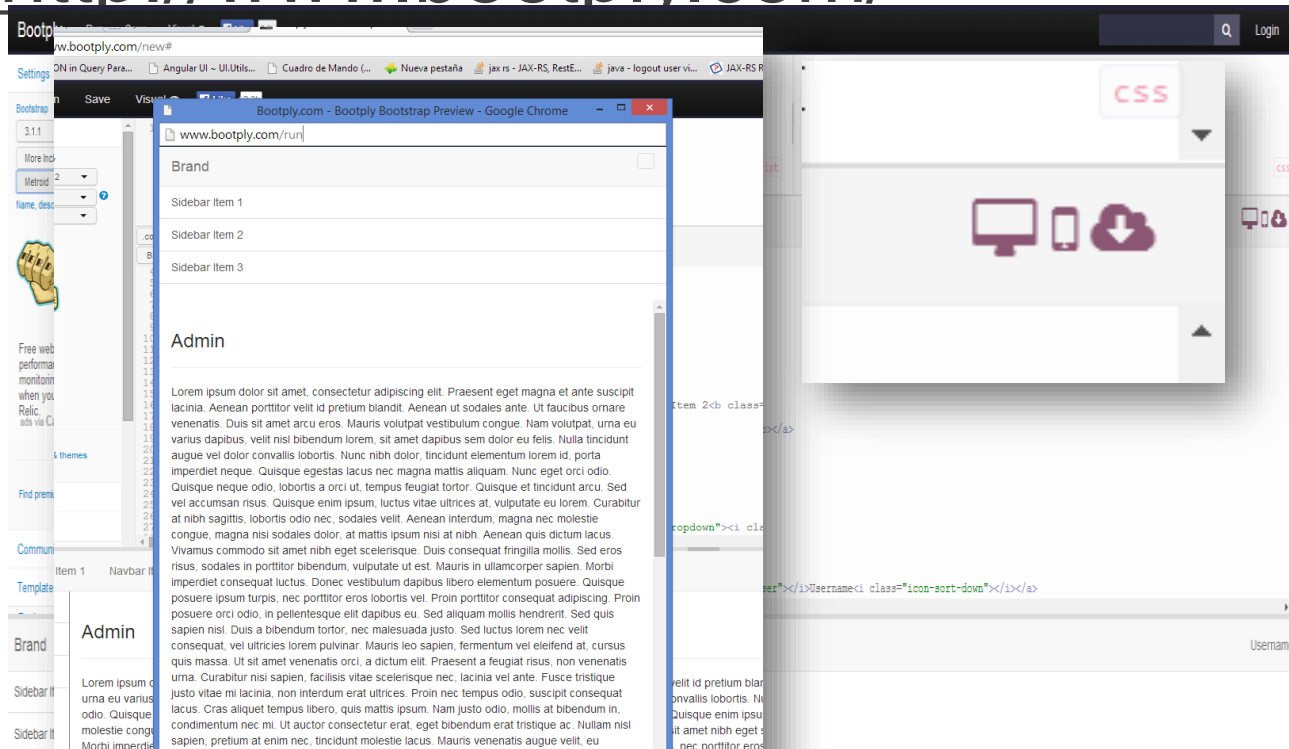
# Editores Online

- <http://www.bootply.com/>



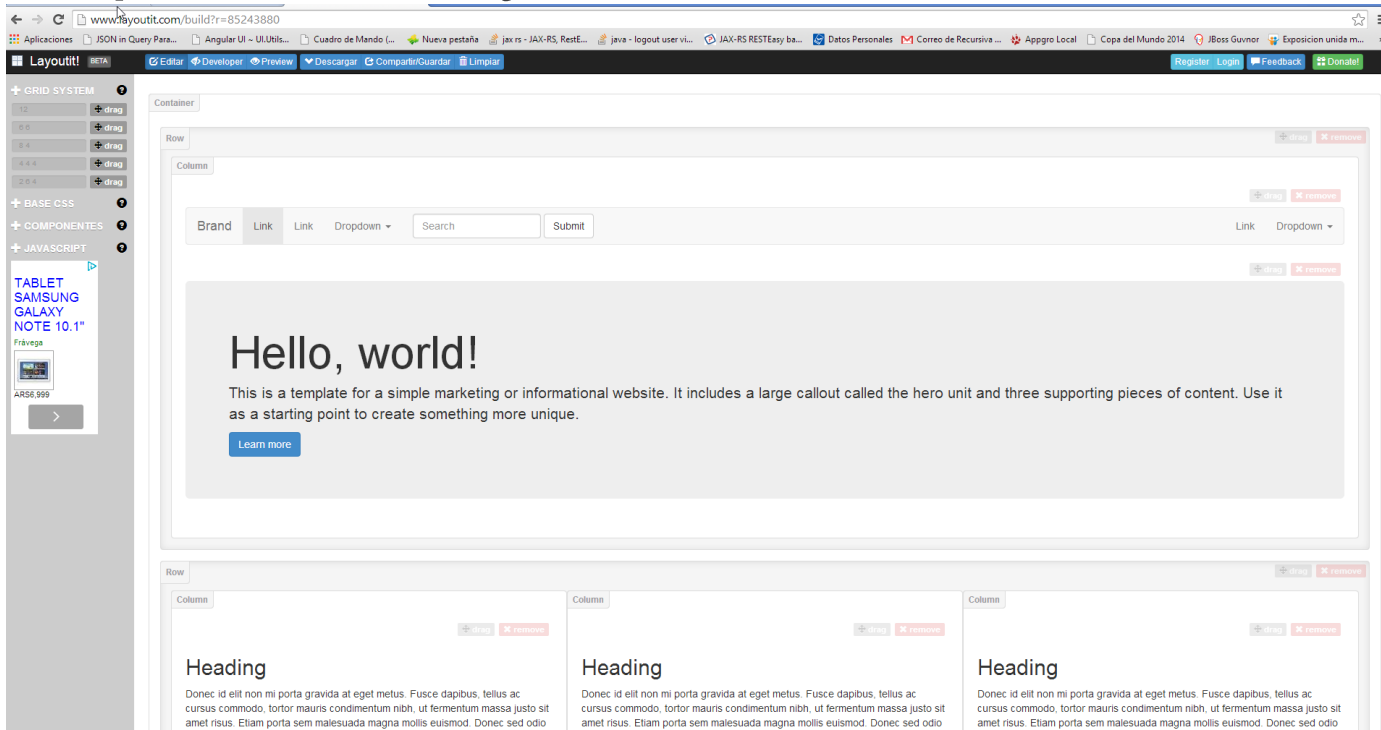
# Editores Online

- <http://www.bootply.com/>



# Editores Online

- <http://www.layoutit.com/build?r=85243880>



# CSS con Superpoderes

*Sass*

# Syntactically Awesome Stylesheets

- ¿Que pasaría si pudiéramos usar variables en las hojas de estilo?
- ¿Nunca han sentido que en lugar de estar repitiendo decenas de líneas sería mejor anidarlas?

# Sass

- Sass es un pre-procesador CSS.
- La idea es “compilar” los archivos Sass para generar archivos CSS.

# Características de Sass

- **Variables**

- Las variables nos permiten guardar datos para ser reutilizados

```
$color:red;
$size:10px;

p{
  color:$red;
  sfont-size:$size + 6;
}
```



# Características de Sass

- **Mixins**

- Los mixins nos permiten guardar un conglomerado de datos para ser reutilizados

```
@mixin caja{  
  display:inline-block;  
  margin:10px;  
  padding:10px;  
  vertical-align:top;  
}  
#cajaUno{  
  width:50%;  
  @include caja;  
}  
#cajaDos{  
  width:35%;  
  @include caja;  
}
```

# Características de Sass

- **Anidar elementos**

- Los mixins nos permiten guardar un conglomerado de datos para ser reutilizados

```
ul.menu{
  background:rgba(51,51,51,0.9);
  list-style:none;

  li{
    display:inline-block;

    a{
      color:#fff;
      text-decoration:none;
      padding:10px 20px;

      &:hover{
        color:white;
      }
    }
  }
}
```

# Compass

- Compass nos facilita un montón de mixins que se le agregan a Sass, como border-radius, box-shadow y gradientes



# Mixins en Compass

- Compass son un montón de mixins para reutilizar

## *Código*

```
box{
border:1px solid black;
height:100px;
width:100px;
@include border-radius(5px);
}
```



## CSS

```
box{
border:1px solid black;
height:100px;
width:100px;
border-radius:5px;
-webkit-border-radius:5px;
-moz-border-radius:5px;
}
```

# Mixins en Compass

- Compass son un montón de mixins para reutilizar

## Código

```
box{
width:100px;
height:100px;
@include border-radius(5px);
@include box-shadow($shadow-1);
}
```



## CSS

```
box{
border:1px solid black;
height:100px;
width:100px;
border-radius:5px;
-webkit-border-radius:5px;
-moz-border-radius:5px;
box-shadow:0px 1px 0px 1px black;
-webkit-box-shadow:0px 1px 0px 1px black;
-moz-box-shadow:0px 1px 0px 1px black;
}
```

# Desarrollando Aplicaciones

Server Side – Client Side  
Frontend – Backend – Fullstack  
Consideraciones....



¿Preguntas?



¡Hasta la próxima  
Semana!