

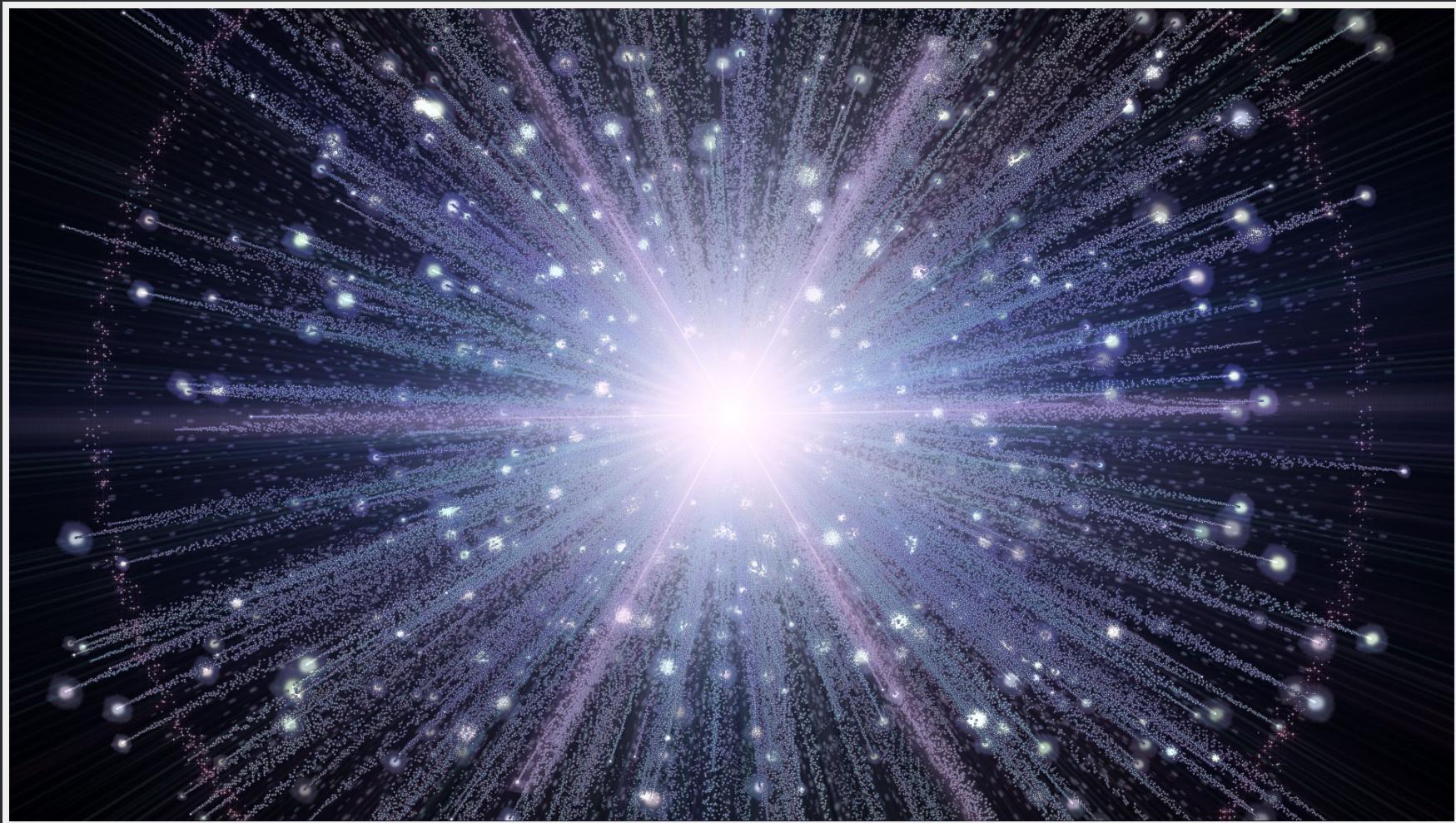
# ANGULARJS

## SUPERHEROIC JAVASCRIPT MVW FRAMEWORK

Created by [Drew Tayman](#) / [dretay@gmail.com](mailto:dretay@gmail.com)

To properly explain AngularJS its necessary to understand the environment in which it was developed.

# IN THE BEGINNING THERE WAS HTML...



HTML was originally intended to enable physicists to share and use documents; it contained 18 tags that could be used to indicate things like paragraphs, tables, and lists within a page.

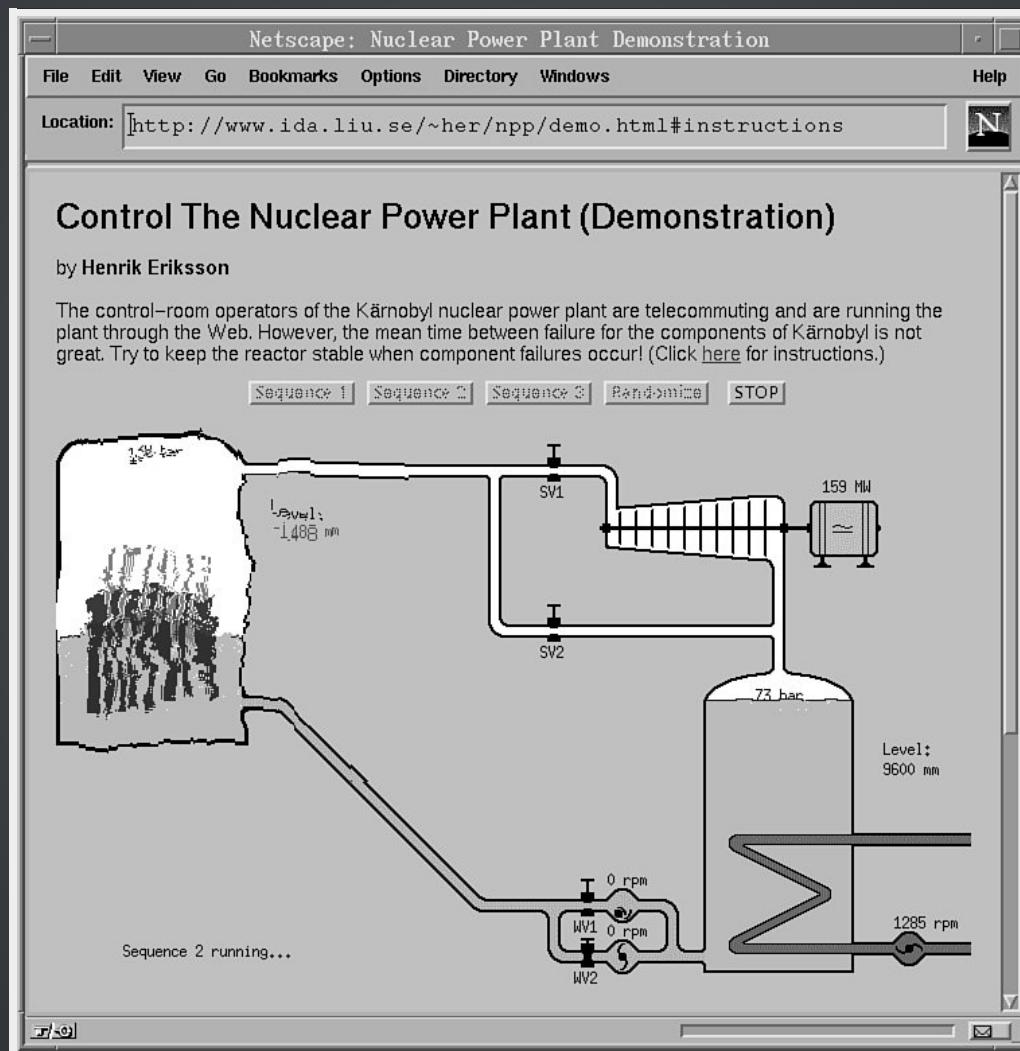
```
<p>
    My favorite elements
    <ol>
        <li>Helium</li>
        <li>Oxygen</li>
        <li>Nitrogen</li>
    </ol>
</p>
```

*My favorite elements*

1. Helium
2. Oxygen
3. Nitrogen

As the web grew in popularity developers hit a snag: users wanted more interactive applications but HTML could only deliver static content.

# ONE IDEA WAS TO USE BROWSER PLUGINS LIKE APPLETS AND FLASH TO BRIDGE THE GAP



# THERE WERE A LOT OF PROBLEMS WITH THIS SOLUTION

- Required that users install and maintain runtimes independent of their browser
- Pages took much longer to load
- Look and feel often differed from the rest of the page
- Resulted in pages had an "island of interactivity" surrounded by static content

Another idea was to use server-side frameworks like Coldfusion, JSP, or ASP to dynamically generate pages as they were requested.

These frameworks tended to be slow and very complicated.



They could also result in inappropriate touching between the view and controller or spaghetti code - ultimately leading to maintainability issues.

```
<html>

<%
    double num = Math.random();
    if (num > 0.95) {
%>
    <h2>Welcome to the Healthcare.gov!</h2>
<%
    } else {
%>
    <h2>Sorry the site is busy, please try again later.</h2>
<%
    }
%>
<a href="<%=" request.getRequestURI() %>"><h3>Try Again</h3></a>

<html>
```

Then AJAX is invented (by IE!) and suddenly web development takes off.



Suddenly developers have the ability to deliver quick local updates to the page (like input validation) and reach back to a server for data (like table content).

There is an explosion of Javascript frameworks.

Start  
Javascript Frameworks  
Taken from Wikipedia  
Crawler by Tim Pietrus



[Edit this Pen](#) [Report Abuse](#)

They share a similar goal of abstracting / normalizing how you interact with the browser. This allows you to write generic code that will execute across all major browsers.

Some frameworks went further and created "widget" frameworks that allow developers to be even more insulated from web development. ExtJS for example lets users build things that look eerily similar to Java Swing.

```
Ext.create('widget.window', {
    title: 'Layout Window',
    closable: true,
    width: 600,
    height: 350,
    layout: 'border',
    items: [{{
        region: 'west',
        title: 'Navigation',
        width: 200,
        split: true,
        collapsible: true,
        floatable: false
    } , {
        region: 'center',
        xtype: 'tabpanel',
        items: [{{
            title: 'Closable Tab',
            html: 'Hello world',
            closable: true
        }}]
    } ]
});
```



**COOL STORY, BRO**

So if Javascript has all the tools needed to build rich internet applications, why are so many frameworks being invented?

Are developers bored?

Are they looking for job security?

I think its because the display language available (HTML) lacks the ability to sufficiently express an application. We're no longer just creating lists of stuff.

```
<p>
    My favorite elements
    <ol>
        <li>Helium</li>
        <li>Oxygen</li>
        <li>Nitrogen</li>
    </ol>
</p>
```

# We're creating dynamic user interfaces

```
<html class="x-strict x-viewport">
  <head></head>
  <body class="x-body x-webkit x-chrome x-mac x-border-layout-ct x-border-box x-container x-container-default" id="ext-gen1018">
    <div id="msg-div"></div>
    <div class="x-container x-border-item x-box-item x-container-default x-box-layout-ct" style="height: 52px; right: auto; left: 0px; top: 0px; margin: 0px; width: 1526px;" id="app-header">...</div>
    <div class="x-panel x-autowidth-table x-grid-no-row-lines x-grid-header-hidden x-border-item x-box-item x-panel-default x-tree-panel x-tree-arrows x-grid" style="width: 250px; height: 923px; right: auto; left: 0px; top: 52px; margin: 0px;" id="navigation-1011">...</div>
    <div class="x-splitter x-border-item x-box-item x-splitter-default x-splitter-vertical x-unselectable" style="width: 8px; height: 923px; right: auto; left: 250px; top: 52px; margin: 0px;" id="navigation-1011-splitter">...</div>
    <div class="x-panel x-border-item x-box-item x-panel-default" id="content-panel" style="right: auto; left: 250px; top: 52px; margin: 0px; width: 600px;">...
      <div class="x-panel x-border-item x-header x-header-horizontal x-docked x-unselectable x-panel-header-default x-horizontal-noborder x-panel-header-horizontal x-panel-header-default x-top x-panel-header-top x-panel-header-default-top x-horizontal-noborder x-panel-header-horizontal-noborder x-panel-header-default-horizontal x-docked-top x-panel-header-docked-top x-panel-header-default-docked-top x-noborder-trl" id="content-panel_header" style="right: auto; left: 0px; top: 0px; width: 600px;">...
        <div id="content-panel-body" class="x-panel-body x-panel-body-default x-panel-body-default x-noborder-trl" style="overflow: auto; width: 660px; height: 887px; left: 0px; top: 36px;">...
          <span id="content-panel-outer" style="display: table; width: 100%; height: 100%;">...
            <div id="content-panel-inner" style="display: inline-block; vertical-align: top; width: 100%; height: 100%; border: 1px solid #ccc; padding: 5px;...">...
              <div class="x-panel x-border-item x-header x-header-horizontal x-docked x-unselectable x-panel-header-default x-horizontal-noborder x-panel-header-horizontal-noborder x-panel-header-default x-docked-top x-panel-header-docked-top x-noborder-trl" id="multi-sort-grid-1033_header" style="right: auto; left: 0px; top: 0px; width: 600px; height: 350px; right: auto; left: 30px; top: 269px;" id="multi-sort-grid-1033">...
                <div class="x-panel x-border-item x-header x-header-horizontal x-docked x-unselectable x-panel-header-default x-horizontal-noborder x-panel-header-horizontal-noborder x-panel-header-default x-docked-top x-panel-header-docked-top x-noborder-trl" id="multi-sort-grid-1033_header" style="right: auto; left: 0px; top: 0px; width: 600px; height: 350px; right: auto; left: 30px; top: 269px;">...
                  <div class="x-toolbar x-docked x-toolbar-docked-top x-toolbar-default-docked-top x-box-layout-ct x-noborder-trl" id="toolbar-1039" style="right: auto; left: 0px; top: 36px; width: 600px;">...
                    <div class="x-grid-header-ct x-docked x-grid-header-ct-dft-docked-top x-grid-header-ct-dft-docked-top x-grid-header-ct-dft-docked-top x-grid-header-ct-dft-docked-top x-grid-header-ct-dft-docked-top x-grid-header-ct-dft-docked-top x-grid-header-ct-dft-docked-top x-noborder-trl" id="headercontainer-1034" style="right: auto; left: 0px; top: 72px; width: 600px;">...
                      <div id="multi-sort-grid-1033-body" class="x-panel-body x-grid-body x-panel-body-default x-layout-fit x-panel-body-default x-noborder-trl" style="width: 600px; height: 248px; left: 0px; top: 102px;">...
                        <table role="presentation" id="gridview-1038" tabindex="1" style="overflow: auto; margin: 0px; width: 600px; height: 247px;">...
                          <caption>...</caption>
                          <thead>...
                            <tr>...
                              <th>...</th>
                            </tr>
                          </thead>
                          <tbody id="gridview-1038-body">...
                            <tr role="row" id="gridview-1038-record-ext-record-720" data-boundview="gridview-1038" data-recordid="ext-record-720" data-recordindex="0" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-737" data-boundview="gridview-1038" data-recordid="ext-record-737" data-recordindex="1" class="x-grid-row x-grid-row-alt x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-717" data-boundview="gridview-1038" data-recordid="ext-record-717" data-recordindex="2" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-722" data-boundview="gridview-1038" data-recordid="ext-record-722" data-recordindex="3" class="x-grid-row x-grid-row-alt x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-728" data-boundview="gridview-1038" data-recordid="ext-record-728" data-recordindex="4" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-726" data-boundview="gridview-1038" data-recordid="ext-record-726" data-recordindex="5" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-719" data-boundview="gridview-1038" data-recordid="ext-record-719" data-recordindex="6" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-727" data-boundview="gridview-1038" data-recordid="ext-record-727" data-recordindex="7" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-732" data-boundview="gridview-1038" data-recordid="ext-record-732" data-recordindex="8" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-718" data-boundview="gridview-1038" data-recordid="ext-record-718" data-recordindex="9" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-723" data-boundview="gridview-1038" data-recordid="ext-record-723" data-recordindex="10" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-724" data-boundview="gridview-1038" data-recordid="ext-record-724" data-recordindex="11" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-728" data-boundview="gridview-1038" data-recordid="ext-record-728" data-recordindex="12" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-716" data-boundview="gridview-1038" data-recordid="ext-record-716" data-recordindex="13" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-731" data-boundview="gridview-1038" data-recordid="ext-record-731" data-recordindex="14" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-721" data-boundview="gridview-1038" data-recordid="ext-record-721" data-recordindex="15" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-733" data-boundview="gridview-1038" data-recordid="ext-record-733" data-recordindex="16" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-735" data-boundview="gridview-1038" data-recordid="ext-record-735" data-recordindex="17" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-736" data-boundview="gridview-1038" data-recordid="ext-record-736" data-recordindex="18" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-715" data-boundview="gridview-1038" data-recordid="ext-record-715" data-recordindex="19" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-725" data-boundview="gridview-1038" data-recordid="ext-record-725" data-recordindex="20" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-729" data-boundview="gridview-1038" data-recordid="ext-record-729" data-recordindex="21" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-730" data-boundview="gridview-1038" data-recordid="ext-record-730" data-recordindex="22" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-713" data-boundview="gridview-1038" data-recordid="ext-record-713" data-recordindex="23" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                            <tr role="row" id="gridview-1038-record-ext-record-713" data-boundview="gridview-1038" data-recordid="ext-record-713" data-recordindex="24" class="x-grid-row x-grid-data-row" tabindex="-1">...
                              <td>...</td>
                            </tr>
                          </tbody>
                        </table>
                      </div>
                    </div>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
  <div class="x-splitter x-border-item x-box-item x-splitter-default x-splitter-vertical x-unselectable" style="width: 8px; height: 923px; right: auto; left: 918px; top: 52px; margin: 0px;" id="east-region-splitter">...</div>
  <div class="x-panel x-border-item x-box-item x-panel-default" style="width: 600px; height: 923px; right: auto; left: 926px; top: 52px; margin: 0px;" id="east-region">...</div>
  <div class="x-tip x-layer x-tip-default x-border-box" id="ext-quicktips-tip" style="display: none;">...</div>
</body>
</html>
```



AngularJS does a lot of things, however at its most basic it is a way of extending HTML's vocabulary so it can describe modern web applications.

For example, let's say you'd like to create a page of biographies  
for your company employees

## To page divisions - 1 coupling the model to

```
var printer = function(name,description){  
    return "<div>"+  
        "<div>"+employee.name+"</div>"+  
        "<div>"+employee.description+"</div>"+  
        "</div>";  
};  
$.ajax({  
    url: "/myCompany/employees"  
}).done(function(employees) {  
    for(var i=0;i<employees.length;i++){  
        var employee = employees[i];  
        $('body').append(  
            printEmployee(employee.name,employee.description)  
        );  
    }  
});
```

We go from talking about  
an object called employee

This code creates an maintenance issue: since we're working with javascript strings rather than HTML its very easy to make a typo and break the view.

From a GOF standpoint we're violating separation of concerns:  
the code that retrieves employees shouldn't care how employees  
are rendered on the screen.

As a consequence of the disconnect between the model and the view developers have to constantly juggle multiple

- Javascript objects
- Rendered HTML fragments
- "Modified" data from user input
- Etc...

representations of their data:

Wouldn't it make life easy if there was an "employee" HTML tag?

Even better, what if there was a "for loop" that we could just feed a list of employees to...

```
$scope.employees = $resource( '/myCompany/employees' );  
...  
<div ng:repeat="employee in employees">  
    <employee information="employee"></employee>  
</div>
```

- [HTML](#)
- 
- [CoffeeScript](#)
- [Result](#)

Edit on

```
<html ng-app="hrApp">
  <body ng-controller="employeeListCtrl">
    <div ng-repeat="employee in employees">
      <employee employee="employee"></employee>
    </div>
  </body>
</html>

angular.module('directives', []).directive 'employee', ->
  restrict: 'E'
  replace: false
  scope:
    employee: "="
    template: "<div>{{employee.name}}</div>"
angular.module('controllers',[]).controller 'employeeListCtrl', ['$scope', '$timeout', ($scope, $timeout) ->
  $scope.employees = [
    {name: "Placeholder"}
  ]
  $timeout ->
    $scope.employees = [
      {name: "Dwight"}]
```

These are known as

# DIRECTIVES

and they are Angular's way of dynamically redefining and extending HTML. They form one of the 3 core ideas Angular is built upon.

# THE THREE D'S IN ANGULARJS

- **DIRECTIVES**
  - markers on a DOM element that tell AngularJS to attach a specified behavior to that DOM element
- **DEPENDENCY INJECTION**
  - mechanism AngularJS uses to associate dependencies with the code that needs them
- **DATA BINDING**
  - automatic two-way synchronization of data between the model and view components

# THE THREE D'S IN ANGULARJS

- **DIRECTIVES**
  - markers on a DOM element that tell AngularJS to attach a specified behavior to that DOM element
- **DEPENDENCY INJECTION**
  - mechanism AngularJS uses to associate dependencies with the code that needs them
- **DATA BINDING**
  - automatic two-way synchronization of data between the model and view components

Directives are used everywhere in AngularJS. They can take the form of either an element, an attribute, css class, or comment.

```
$scope.employees = $resource( '/myCompany/empl
```

This an attribute  
adds some behav  
HTML c

...

```
<div ng-repeat="employee in employees">  
    <employee information="employee">  
</div>
```

This an element dire  
defines a new element  
be used anywhere on

# CSS AND COMMENT DIRECTIVES ARE RARELY USED:

## CSS Directive

```
<div class="employee: information;"></div>
```

## Comment Directive

```
<!-- directive: employee information -->
```

# THE THREE D'S IN ANGULARJS

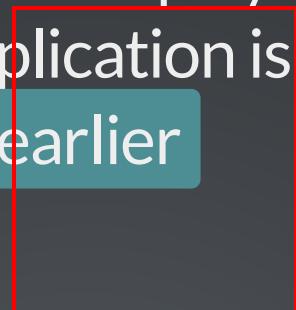
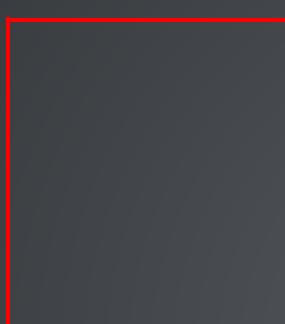
- **DIRECTIVES**
  - markers on a DOM element that tell AngularJS to attach a specified behavior to that DOM element
- **DEPENDENCY INJECTION**
  - mechanism AngularJS uses to associate dependencies with the code that needs them
- **DATA BINDING**
  - automatic two-way synchronization of data between the model and view components

AngularJS's Dependency Injection does not have an opinion on how resources are loaded onto the page. It is only concerned with helping a consumer choose what resources to consume.

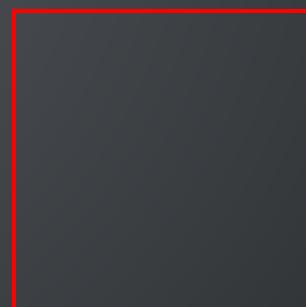
You will still need to use something like RequireJS or Browserify (or use script tags) to get your stuff on the page.

The real value of DI comes into play during testing. Say your application is structured like the Employee directive from earlier

A service that provides necessary data-needed by the information system



A provider to make the remote connection and fetch the data



If you want to test this application you need to be able to simulate connecting to an external data source.

A traditional approach to testing this application would use something like Sinon.JS to stub any functions that make external calls

```
describe 'The employee directive', ->
  beforeEach ->
    sinon.stub $, 'ajax', (options)->
      deferred = $.Deferred()
      response = #some test input...
      deferred.resolve(JSON.parse(response))
      deferred.promise()
  afterEach ->
    $.ajax.restore()
```

# THIS APPROACH HAS SOME LIMITATIONS

1. The override is global - if you're running many tests in parallel there's a chance of collision
2. You must properly clean up stubs once your test is done or risk leaking one test into another
3. Code must be modular enough to accept stubbed methods (either relying on global variables or allowing the appropriate arguments to be passed into the constructor)
4. All possible requests must be handled
5. More difficult to share common resource code across multiple tests
6. Becomes more difficult as the more transport mechanisms are introduced (a WebSocket that falls back to Long Polling, which can fall back to interval polling)

With dependency injection many of these problems can be alleviated...

we can simply provide a mock provider

# WHEN THE APPLICATION IS INITIALIZED THE MOCK PROVIDER WILL BE INJECTED AUTOMATICALLY

```
describe 'The employee directive', ->
  beforeEach ->
    module ($provide)->
      $provide.provider 'provider', ->
        this.$get = ['$http', ($http)->
          #do test stuff...
        ]
```

# THIS APPROACH HAS SEVERAL ADVANTAGES

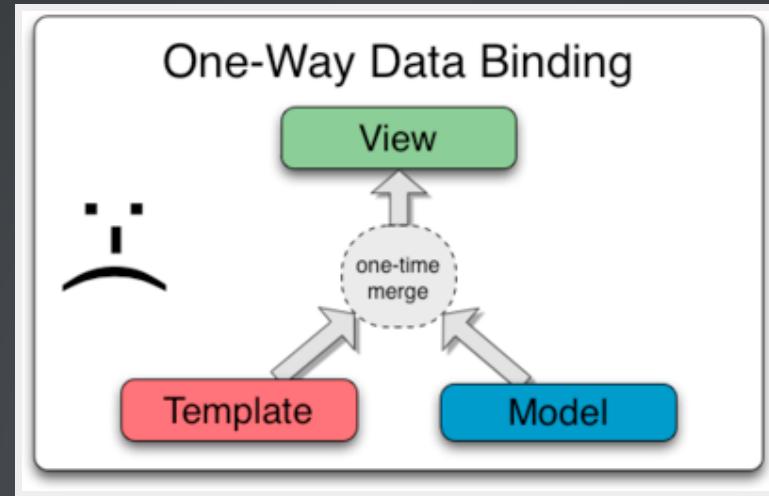
1. You are not responsible for how the application is initialized in the test suite
2. The mocked module can depend on any other module in your application regardless of whether or not it is mocked
  - A mocked module that makes an external request can use a non-mocked module to process the data
3. It makes it much easier to test larger applications with deep dependency graphs
4. Can yield much higher code coverage since you tend to mock less to get a working test
5. When combined with aspects of Angular JS's test suite can shorten the feedback loop if any module is failing to properly render

# THE THREE D'S IN ANGULARJS

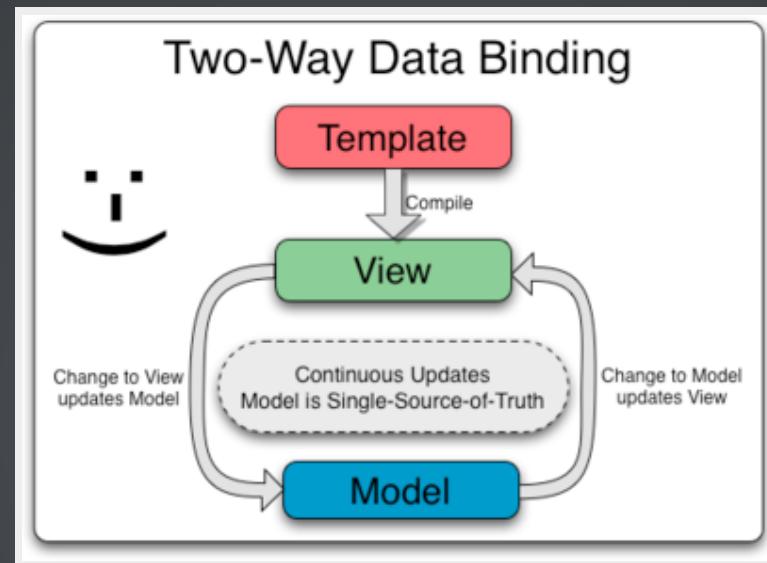
- **DIRECTIVES**
  - markers on a DOM element that tell AngularJS to attach a specified behavior to that DOM element
- **DEPENDENCY INJECTION**
  - mechanism AngularJS uses to associate dependencies with the code that needs them
- **DATA BINDING**
  - automatic two-way synchronization of data between the model and view components

Other frameworks such as Ember and Backbone are capable of something called "one-way" data binding. This means that they are able to update the view (HTML) automatically when data changes.

However once the data has been dumped on the page, you're responsible for keeping it in sync



This is where AngularJS really shines - its capable of "bi-directional" data binding - the framework is responsible for keeping your model and view in sync:



- [HTML](#)
- 
- [CoffeeScript](#)
- [Result](#)

Edit on

```
<html ng-app="hrApp">
  <body ng-controller="employeeListCtrl">
    <input ng-model="employee"></input><br/>
    You input: {{employee}}
  </div>
</body>
</html>

angular.module('controllers',[]).controller 'employeeListCtrl', ['$scope', '$timeout', ($scope, $timeout) ->
  $scope.employee = "any value you want"
]

angular.module 'hrApp', ['controllers']
```

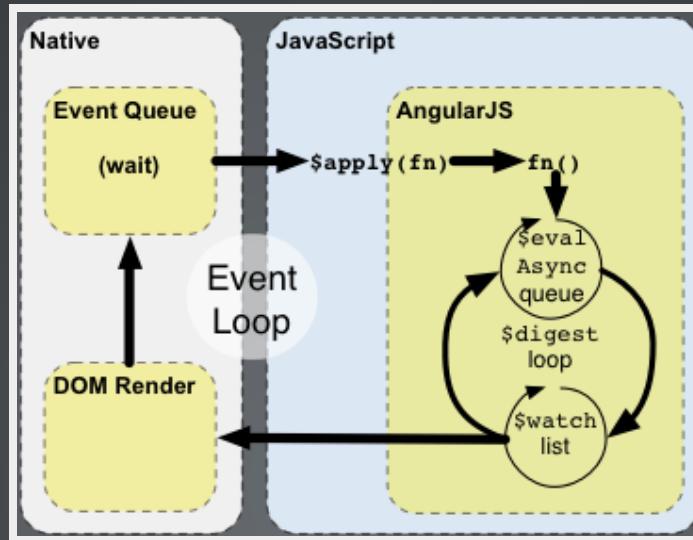
[View Compiled](#)

You input: any value you want

# WITH GREAT POWER COMES GREAT HEADACHES

While this may be the single greatest feature within the AngularJS framework, it is also likely to cause you the most ajada.

Angular keeps the model in sync using something called a "digest" cycle that is inserted into normal window DOM processing



Basically angular keeps an object reference to everything it sees  
and watches to see if any values changed when a dom event is  
fired

# THERE ARE ALL KINDS OF ISSUES THAT ARE LESS COMMON TO TRADITIONAL JS DEVELOPERS THAT CAN CROP UP DURING DIGEST, INCLUDING:

- variable scoping issues
- variable reference issues (using soft variable references)
- variable collision issues