**A. AngularJS căn bản (lý thuyết)**

**Học AngularJS 1 từ website freetuts**

* Bài 01: Tổng quan về AngularJS
* Bài 02: Download AngularJS và viết ứng dụng AngularJS đầu tiên
* Bài 03: Tìm hiểu Controller trong AngularJS
* Bài 04: Tìm hiểu Scope và rootScope trong AngularJS
* Bài 05: Tìm hiểu AngularJS Directives - Danh sách Directives
* Bài 06: Directive ng-model trong AngularJS
* [Bài 07: Directive ng-model-options trong AngularJS](http://freetuts.net/directive-ng-model-options-trong-angularjs-207.html)
* [Bài 08: Directives ng-bind - ng-bind-html - ng-bind-template trong AngularJS](http://freetuts.net/directives-ng-bind-ng-bind-html-ng-bind-template-trong-angularjs-208.html)
* [Bài 09: Directive Form trong AngularJS](http://freetuts.net/directive-form-trong-angularjs-209.html)
* [Bài 10: Directive ng-form trong AngularJS](http://freetuts.net/directive-ngform-trong-angularjs-212.html)
* [Bài 11: Directive input trong AngularJS](http://freetuts.net/directive-input-trong-angularjs-213.html)
* [Bài 12: Directive input checkbox trong AngularJS](http://freetuts.net/directive-input-checkbox-trong-angularjs-214.html)
* [Bài 13: Directive input date trong AngularJS](http://freetuts.net/directive-input-date-trong-angularjs-216.html)
* [Bài 14: Directive input datetime-local trong AngularJS](http://freetuts.net/directive-input-datetime-local-trong-angularjs-217.html)
* [Bài 15: Directive input time trong AngularJS](http://freetuts.net/directive-input-time-trong-angularjs-218.html)
* [Bài 16: Directive input month và input week trong AngularJS](http://freetuts.net/directive-input-month-va-input-week-trong-angularjs-219.html)
* [Bài 17: Directive ng-show và ng-hide trong AngularJS](http://freetuts.net/directive-ng-show-va-ng-hide-trong-angularjs-220.html)
* [Bài 18: Directive ng-repeat trong AngularJS](http://freetuts.net/directive-ngrepeat-trong-angularjs-221.html)
* [Bài 19: Directive ng-init trong AngularJS](http://freetuts.net/directive-ng-init-trong-angularjs-233.html)
* [Bài 20: Directive ng-include trong AngularJS](http://freetuts.net/directive-ng-include-trong-angularjs-234.html)
* [Bài 21: Services trong AngularJS và cách tạo Service mới](http://freetuts.net/services-trong-angularjs-va-cach-tao-service-moi-241.html)
* [Bài 22: Directive ng-class trong angularjs](http://freetuts.net/directive-ngclass-trong-angularjs-247.html)
* [Bài 23: Directive ng-click trong angularjs](http://freetuts.net/directive-ngclick-trong-angularjs-250.html)
* [Bài 24: Directive ng-class-odd và ng-class-even trong Angularjs](http://freetuts.net/directive-ng-class-odd-va-ng-class-even-trong-angularjs-251.html)

**Bài 01: Tổng quan về AngularJS**

Trong loạt bài serie này tôi và các bạn sẽ tìm hiểu các kiến thức căn bản nhất của Angular, mỗi bài sẽ đi tìm hiểu một chủ đề, trong mỗi bài sẽ có các ví dụ minh họa và demo trực tiếp trên live editor nên rất dễ để học. Hy chúng ta sẽ thành công khi xong loạt serie này.

**1. AngularJS là gì?**

Angular là một bộ Javascript Framework rất mạnh và thường được sử dụng để xây dựng project **Single Page Application** (SPA). Nó hoạt động dựa trên các thuộc tính mở rộng HTML (các atributes theo quy tắc của Angular). Đây là một Framework mã nguồn mở hoàn toàn miễn phí và được hàng ngàn các lập trình viên trên thế giới ưa chuộng và sử dụng. Framework này được thế hệ Web 2.0 phát triển khá mạnh ở nước ngoài, tuy nhiên ở Việt Nam thì vẫn chưa thông dụng lắm.

**2. Những ai cần nắm bắt AngularJS**

Về thành phần đối tượng học serie **AngularJS căn bản đến nâng cao** này là những bạn muốn đi theo nghề lập trình web nói chung và những bạn muốn sử dụng AngularJS để làm dự án nói riêng. Mình cam đoan rằng sau khi serie này hoàn thành bạn có thể tự mình làm ứng dụng **Angular căn bản**, hay thầm chí là ứng dụng **Angular nâng cao** nếu như bạn có tâm huyết tìm hiểu đến cùng.

Hy vọng rằng sau loạt serie này bạn sẽ tự mình nghiên cứu thêm nhé, vì bản chất học lập trình không thể nào trình bày full được mà chỉ ở mức basic, chỉ đường cho bạn đi thôi.

**3. Bạn cần chuẩn bị những kiến thức gì để học AngularJS**

Để học được Angular thì bạn phải có kiến thức cơ bản về javascript, biết cách xử lý object, string, … Hay thậm chí nếu như bạn biết chuyên sâu Javascript thì đó là một lợi thế rất nhiều. Vì bản chất Angular hoạt động dạng Single Page, sử dụng dạng API để lấy dữ liệu, vì vậy bạn phải biết các kỹ thuật xử lý DHTML, Ajax. Nếu như bạn chưa biết các công nghệ kỹ thuật đó thì hãy dừng serie này tại đây nhé, vì serie này bạn chưa đủ trình độ để tìm hiểu nó.

**4. Các đặc tính của AngularJS**

* AngularJS là một Framwork phát triển dựa trên Javascript để tạo các ứng dụng web phong phú
* AngularJS thường dùng để phát triển frontend (giao diện khách hàng) thông qua các API để gọi data, sử dụng mô hình MVC rất mạnh mẽ
* Mã nguồn AngularJS tự động fix với các trình duyệt khác nhau nên bạn không cần phải lo vấn đề tương thích trình duyệt
* Angular là mã nguồn mở, hoàn toàn miễn phí và được phát triển bởi hàng ngàn các lập trình viên trên thế giới.

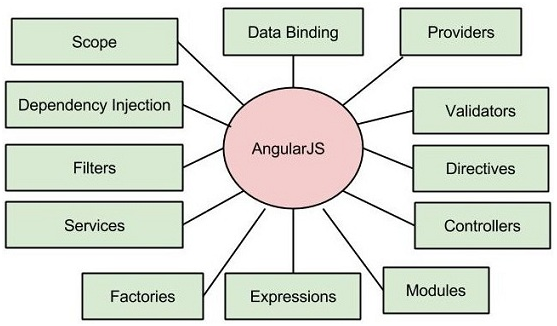
Chung quy lại có thể hiểu khi làm việc với AngularJS giống như là đang làm việc với Ajax, sử dụng cớ chế bind data, hoạt động theo mô hình MVC và sử dụng service để tương tác với dữ liệu từ server. Để rõ hơn thì chúng ta tìm hiểu các tính năng cố lõi của nó nhé.

**5. Các tính năng code lõi của AngularJS**

Sau đây là các tính năng cố lõi quan trọng trong AngularJS

* **Data-binding**: (liên kết dữ liệu) tự động đồng bộ dữ liệu giữa model và view
* **Scope**: (Phạm vi) Đây là những đối tượng kết nối giữa Controller và View
* **Controller**: Đây là những hàm javascript xử lý kết hợp với bộ điều khiển Scope
* **Service**: Như tôi đề cập ở trên, AngularJS sử dụng các API được xây dựng từ các web service (PHP, ASP) để thao tác với DB.
* **Filters**: Bộ lọc lọc ra các thành phẩn của một mảng và trả về mảng mới
* **Directives**:  đánh dấu vào các yếu tố của DOM, nghĩa là sẽ tạo ra các thẻ HTML tùy chỉnh
* **Templates**: hiển thị thông tin từ controller, đây là một thành phần của views
* **Routing**:  chuyển đổi giữa các action trong controller
* **MVC**: Mô hình chia thành phần riêng biệt thành Model, View, Controller. Đây là một mô hình khá hay nhưng trong Angular thì nó được chế biến lại một chút gần giốn với MVVM (Model View View Model)
* **Deep Linking:** Liên kết sâu, cho phép bạn mã hóa trạng thái của ứng dụng  trong các URL  để nó có thể đánh dấu được với công cụ tìm kiếm.
* **Dependency Injection:** Angular giúp các nhà phát triển tạo ứng dụng  dễ dàng hơn để phát triển, hiểu và thử nghiệm dễ dàng.

Sau đây là hình ảnh mô hình cac thành phần quan trọng trong AngularJS:

  
Các thành phần của AngularJS

**6. Ưu điểm và nhược điểm của AngularJS**

**Ưu điểm:**

* Angular cho phép tạo ra các ứng dụng một cách đơn giản, code sạch
* Angular sử dụng data bind giống .NET với tính năng liên kết với HTML nên giúp người dùng cảm thấy dễ chịu.
* Angular đang ở giai đoạn thử nghiệm
* Angular có thể chạy trên hầu hết các trình duyệt điện thoại thông minh.

**Nhược điểm:**

Mặc dù angular có nhiều lợi thế ưu điểm nhưng nó cũng có mặt trái ngược lại:

* **Không an toàn**: Được phát triển từ javascript nên nó không an toàn, phía máy chủ phải thường xuyên xác nhận quyền để hệ thống chạy trơn tru.
* **Phụ thuộc**: Nếu người dùng vô hiệu hóa javascript thì coi như đi tong con long bong.

**7. Các Components chính trong AngularJS**

Angular JS được chia làm ba thành phần chính sau đây:

* **ng-app**: định nghĩa này chỉ thị một kết nối ứng dụng Angular JS tới HTML
* **ng-model**: chỉ thị này liên kết với dữ liệu của ứng dụng Angular
* **ng-bind**: chỉ thị này dùng đưa dữ liệu vào HTML tags
* .....

Còn rất nhiều nhưng ta sẽ tìm hiểu dần qua các bài sau nhé.

**8. Lời kết**

Qua loạt lý thuyết này bạn sẽ thấy AngularJS có rât nhiều thứ ta cần phải học và rất cực nhọc để master được nó đấy nhé. Hy vọng chúng ta sẽ thành công đưa nó vào lòng bàn tay để serie được hoàn thành một cách ... ngon lành nhé

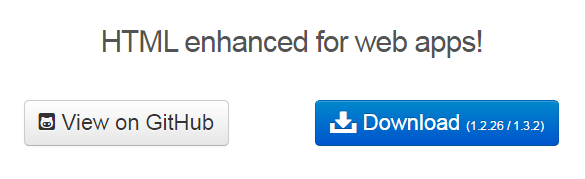
**Bài 02: Download AngularJS và viết ứng dụng AngularJS đầu tiên**

Bài trước chúng ta đã tìm hiểu khái niệm về AngularJS và các ứng dụng, ưu và nhược điểm của Angular, tuy nhiên đó là về mặt lý thuyết ngắn gọn nên mình không thể trình bày hết nội dung của nó được. Và để các bạn rõ ràng hơn thì trong bài này chúng ta sẽ viết một ứng dụng bằng AngularJS đầu tiên với tên là "**Hello World**".

Như ta biết, AngularJS là một Framwork được viết bằng Javascript nên đương nhiên trước khi sử dụng nó đòi hỏi trình duyệt của bạn phải bật JS nhé, và cái thứ hai là phải tích hợp bộ thư viện của nó vào là phải download bộ Angular từ [trang chủ](http://angularjs.org/) của nó đã nhé.

**1. Download AngualarJS**

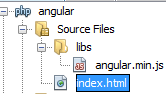
Bạn vào trang chủ của AngularJS với đường link [angularjs.org](https://angularjs.org/):



Tại đây có hai sự lựa chọn cho bạn, nếu bạn click chọn **View on GitHub** thì bạn sẽ lên trang Github để download nhé, còn nếu bạn chọn **Download** thì nó sẽ hiển thị một popup cho bạn download. Sau khi download xong bạn sẽ có một file với tên　**angular.min.js.**Đây chính là bộ thư viện AngularJS.

**2. Viết chương trình AngularJS đầu tiên**

Để hiểu angular là gì thì ta sẽ viết ứng dụng đầi tiên nhé. Bước đầu tiên bạn phải tạo một project với tên là **angular**nằm trong webserver của bạn, sau đó bạn tạo một folder tên libs và copy thư viện AngularJS vào, sau đó bạn tạo một file index.html để viết ứng dụng. Cấu trúc như sau (ảnh chụp từ netbeans):



Bây giờ bạn mở file **index.html** lên và import thư viện Angular vào như sau:

|  |
| --- |
| <!DOCTYPE html>  <html>      <head>          <title></title>          <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">          <script language="javascript" src="libs/angular.min.js"></script>      </head>      <body>          <div>TODO write content</div>      </body>  </html> |

Bây giờ ta sẽ viết ứng dụng in ra một câu chào trên màn hình bằng cách sử dụng AngularJS. Bạn sửa lại file **index.html** với nội dung như sau:

|  |
| --- |
| <!DOCTYPE html>  <html>      <head>          <title></title>          <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">          <script language="javascript" src="libs/angular.min.js"></script>      </head>      <body>          <div ng-app="myapp">              <div ng-controller="HeaderController">                  <h2>Xin chào {{data.title}} đến với website {{data.website}}!</h2>              </div>              <script>                  angular.module("myapp", []).controller("HeaderController", function($scope) {                      $scope.data = {                          title : 'các bạn',                          website : 'freetuts.net'                      };                  });              </script>          </div>      </body>  </html> |

Trong đó các bạn cần chú ý những đoạn code sau:

* <div **ng-app="myapp"**>

trong đó ng-app="myapp" là đoạn code khai báo bắt đầu chương trình Angular, lúc này AngularJS sẽ nhận biết và xử lý những đoạn code nằm bên trong  thẻ div này. myapp là tên của ứng dụng nên ta có thể đặt tên tùy ý, tuy nhiên ng-app là thuộc tính bắt buộc nhé, bạn không thể đổi tên nó được.

* <div **ng-controller="HeaderController"**>

trong　đó ng-controller="HeaderController" là khai báo một controller với tên là HeaderController, còn key ng-controller là bắt buộc vì đó là thuộc tính của AngularJS.

* Đoạn code:

|  |
| --- |
| angular.module("myapp", []).controller("HeaderController", function($scope) {      $scope.data = {          title : 'các bạn',          website : 'freetuts.net'      };  }); |

là đoạn khai báo viết code cho ứng dung, trong đó:

* + **angular.module("myapp", [])**là bắt đầu viết code cho ứng dụng có tên là myapp
  + **controller("HeaderController",function($scope)**là xử lý cho controller có tên là **HeaderController**.
  + Bên trong là những đoạn code khai báo giá trị trả về. Bạn cần lưu ý rằng biến **$scope** là bắt buộc nhé (*không thể đổi thành tên khác*), nó là thành phần trao đổi thông tin gữa controller và view, nghĩa là nếu bạn gán bất kì biến nào trong biến $scope thì trong view sẽ nhận được biến đó **với key tương ứng**. Như ở ví dụ trên thì key ở view tương ứng sẽ là **{{data.title}**} và **{{data.website}}** vì trong **$scope** mình khai báo một object có hai key như vậy

Chạy code lên bạn sẽ thấy trên màn hình xuất hiện như sau:



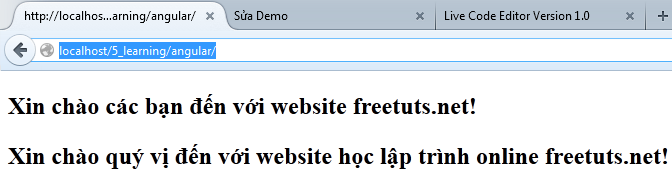
**3. Viết chương trình AngularJS thứ hai**

Để các bạn hiểu rõ hơn nữa thì mình sẽ viết thêm một ví dụ tương tự như ví dụ ở trên. Trong ví dụ này mình sẽ thêm một controller nữa bên trong **myapp** và xử lý nó. Bạn sửa lại file index.html với nội dung như sau:

|  |
| --- |
| <!DOCTYPE html>  <html>      <head>          <title></title>          <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">          <script language="javascript" src="libs/angular.min.js"></script>      </head>      <body>          <div ng-app="myapp">              <div ng-controller="HeaderController">                  <h2>Xin chào {{data.title}} đến với website {{data.website}}!</h2>              </div>              <div ng-controller="ContentController">                  <h2>Xin chào {{data.title}} đến với website {{data.website}}!</h2>              </div>              <script>                  angular.module("myapp", []).controller("HeaderController", function($scope) {                      $scope.data = {                          title : 'các bạn',                          website : 'freetuts.net'                      };                  }).controller("ContentController", function($scope){                      $scope.data = {                          title : 'quý vị',                          website : 'học lập trình online freetuts.net'                      };                  });              </script>          </div>      </body>  </html> |

Sự khác biệt so với ví dụ 1 chỉ là thêm một controller ở View (HTML), xử lý thêm một controller (JS). Trong đó bạn cần chú ý:

* Trong 2 controller mình đặt 2 tên khác nhau và trong JS mình trả về biến **$scope** có các key giống nhau nhưng nó lại không bị nhầm lẫn khi render sang View
* Mỗi controller các giá trị trong **$scope** dù bị trùng key nhưng bản chất nó là 2 controller khác nhau nên nó không hề nhầm lẫn với nhau
* Khi một app có nhiều controller thì ta dùng dấu chấm **(.)** để nối giữa các đoạn code
* Và đây là giao diện khi chạy　lên



**4. Lời kêt**

Tới bài này bạn có thấy bắt đầu khó không nhỉ, thật sự là khó đấy nếu như bạn không rành cách xử lý object trong javascript. Vì vậy nếu bạn không hiểu thì hãy tìm hiểu JS thật kỹ nhé.

Trong bài này ta chỉ tìm hiểu cách download AngularJS và chạy chương trình AngularJS đầu tiên với mục đích giúp các bạn đở thắc mắc hơn.

**Bài 03: Tìm hiểu Controller trong AngularJS**

Ở bài trước chúng ta đã thực hành hai ví dụ đơn giản trong AngualrJS nhưng có những cái mà có lẽ bạn thắc mắc không hiểu tại sao phải code như vậy, những thứ đó chúng ta sẽ tìm hiểu dần dần qua từng bài, từng ví dụ cụ thể nên nếu bạn không tiếp thu được thì hãy bỏ qua đoạn đó nhé. Tuy nhiên theo tôi thì nó đơn giản nên đa số bạn nào cũng hiểu được cả.

Trong các ví dụ ứng dụng Angular đầu tiên tôi có sử dụng controller để xử lý, đây là một thành phần trong mô hình MVC nên trong bài này chúng ta sẽ tìm hiểu nó trước, sau đó ở các bài tiếp theo chúng ta tìm hiểu các thành phần khác như template, services, scope.

**1. Controller trong Angular**

Trong Angular, Controller là một hàm khởi tạo có nhiệm vụ thực hiện xử lý dữ liệu cho đối tượng $scope, từ đây bên Views sẽ sử dụng các dữ liệu trong $scope để hiển thị ra tương ứng.

Khi một controller được gán vào các DOM trong HTML bằng thuộc tính chỉ thị ng-controller  thì Angular sẽ nhận biết và tạo một đối tượng Controller mới sử dụng như là một hàm khởi tạo, nghĩa là sẽ được chạy khi website được load xong. Lúc này Angular cũng tạo luôn một đối tượng $scope mới dành riêng cho controller đó. Như ở ví dụ 2 bài trước các bạn thấy hai $scope ở hai controller khác nhau có cùng giá trị nhưng khi hiển thị ra thì nó không liên quan với nhau.

**Nên sử dụng Controller trong Angular khi:**

* Controller trong Angular dùng để thiết lâp giá trị, trạng thái ban đầu cho đối tượng $scope
* Thêm các hàm, hành vi vào đối tượng $scope

**Không nên sử dụng Controller trong Angular khi:**

* Thao tác với DOM, nghĩa là dùng Controller để thay đổi giá trị của DOM (nội dung, thuộc tính). Trong trường hợp này thì có cơ chế **data-binding** rồi, bởi vậy không nên sử dụng Controller.
* Validate form, nghĩa là không dùng controller để kiểm tra định dạng dữ liệu input của form. Trong trường hợp này sử dụng Angular Form Controls để thay thế.
* Lọc dữ liệu, nghĩa là chuyển đổi định dạng dữ liệu, trong trường hợp này thì dùng đối tượng Filter trong Angular.
* Chia sẽ dữ liệu, trường hợp này dùng Angualar Service
* Quản lý thời gian sống của các component

**2. Khai báo controller trong Angular**

**Trường hợp 1:** Mỗi controller trong Angular dùng để thực thi một nhiệm vụ nhất định và nó thuộc phạm vi của một ứng dụng (**ng-app**).

**Ví dụ**:

|  |
| --- |
| <div ng-app="myapp">      <div ng-controller="HeaderController">          <h2>Xin chào {{data.title}} đến với website {{data.website}}!</h2>      </div>  </div> |

Trong đó mình đã khai báo một app là ng-app="myapp" và tạo một controller nằm bên trong với tên là ng-controller="HeaderController", như vậy Controller (*HeaderController*) đã nằm gọn bên trong App (*myapp*). Trong trường hợp này bắt buộc ta phải dùng javascript để khai báo với Angular là ta có sử dụng app nào và controller nào:

|  |
| --- |
| angular.module("myapp", []).controller("HeaderController", function($scope) {      $scope.data = {          title : 'các bạn',          website : 'freetuts.net'      };  }); |

**Trường hợp 2**: Controller thuộc một ứng dụng rỗng (**ng-app=""**)

**Ví dụ**:

|  |
| --- |
| <div ng-app="">      <div ng-controller="HeaderController">          <h2>Xin chào {{data.title}} đến với website {{data.website}}!</h2>      </div>  </div> |

Trong trường hợp này ta sẽ xây dựng một hàm trong javascript có tên gọi giống như tên của ta khai báo trong thuộc tính app-controller.

|  |
| --- |
| function HeaderController($scope){      $scope.data = {          title : 'các bạn',          website : 'freetuts.net'      };  } |

**Lưu ý**: là tên tham số truyền vào hàm HeaderController bắt buộc phải tên là $scope.

**3. Ví dụ Controller trong Angular**

Sau đây là ví dụ hiển thị thông tin cá nhân của website freetuts.net bằng cách sử dụng AngularJS để hiển thị. Trong ví dụ này chúng ta sẽ tạo một đoạn chữ hiển thị thông tin cá nhân, mỗi thông tin ta sẽ đưa tham số **{{ten}}** vào để **Angular binding** lặp dữ liệu.

|  |
| --- |
| <!DOCTYPE html>  <html>      <head>          <title></title>          <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">          <script src="<http://ajax.googleapis.com/ajax/libs/angularjs/1.2.15/angular.min.js>">  </script>      </head>      <body ng-app="myapp">          <div ng-controller="WebsiteInfomation">              <p>                  Chủ sở hữu: {{data.fullname}} <br/>                  Email: {{data.email}} <br/>                  Điện Thoại: {{data.phone}} <br/>                  Chủ Đề: {{data.topic}} <br/>                  Website: {{data.website}}              </p>          </div>          <script>              angular.module('myapp', []).controller('WebsiteInfomation', function($scope){                  $scope.data = {                      fullname : 'Nguyễn Văn Cường',                      email : 'Thehalfheart@gmail.com',                      phone : '0979 306 603',                      topic : 'Học lập trình online miễn phí',                      website : "freetuts.net"                  };              });          </script>      </body>  </html> |

|  |
| --- |
|  |

**4. Lời kết**

Qua bài này các bạn đã biết một số cách sử dụng Controller trong Angular và những trường hợp nào nên sử dụng Controller và trường hợp nào thì không nên sử dụng Controller

**Bài 04: Tìm hiểu Scope và rootScope trong AngularJS**

Các ví dụ ở bài trước chúng ta đã sử dụng đối tượng $scope rất nhiều rồi và tôi cũng đa có giải thích tác dụng và cách dùng của nó, nhưng để các bạn dễ hình dung hơn thì trong bài này chúng ta sẽ tìm hiểu kỹ hơn đối tượng $scope này. Nhưng trước khi bắt đầu bạn nên ôn kỹ các bài trước đã nhé.

**1. Scope là gì?**

**$scope là một object** (đối tượng) có nhiệm vụ giao tiếp dữ liệu giữa controller và view của ứng dụng. Nó sẽ thực hiện dưới dạng biểu thức, nghĩa là ở model sẽ được khai báo đúng với quy cách thì đối tượng scope sẽ truyền hành động (function) hoặc dữ liệu tương ứng và ta có thể truyền các sự kiện thông qua đối tượng này.

Scopes cung cấp các biểu thức giống như các template engine hiện nay, ví dụ để hiển thị username thì ta sẽ khai báo là **{{username}}** và ở controller chúng ta chỉ việc gán **$scope.username = 'something'** thì đối tượng này sẽ lấy key có tên là username gán vào view **{{username}}**.

**$scope là cầu nối giữa controller và view**

Điều này mình thấy cũng đúng thật, nghĩa là bạn thêm dữ liệu cho $scope ở controller thì nếu bên view có khai báo theo đúng quy tắc thì nó sẽ tự đống gán thông tin vào đúng vị trí đó.

**Ví dụ**:

|  |
| --- |
| <!DOCTYPE html>  <html>      <head>          <title></title>          <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">          <script src="<http://ajax.googleapis.com/ajax/libs/angularjs/1.2.15/angular.min.js>">  </script>      </head>      <body ng-app="myapp">          <div ng-controller="MyController">              Nhập tên của bạn:              <input type="text" value="" ng-model="username">              <button ng-click='sayHello()'>In thông báo</button>              <hr>              {{greeting}}          </div>          <script>              angular.module('myapp', [])              .controller('MyController', ['$scope', function($scope)              {                  $scope.sayHello = function() {                    $scope.greeting = 'Xin chào ' + $scope.username + '!';                  };              }]);          </script>      </body>  </html> |

Trong ví dụ này tôi khai báo một action ng-click ="sayHello()" có ý nghĩa rằng khi click vào input đó thì nó sẽ gọi đến hàm sayHello() mà ta định nghĩa trong **$scope ở controller**.

|  |
| --- |
| $scope.sayHello = function() {    $scope.greeting = 'Xin chào ' + $scope.username + '!';  }; |

Trong hàm sayHello() có nhiệm vụ là đổi giá trị cho gretting, điều này có nghĩa là nó đang gán lại giá trị cho view ở biểu thức {{greeting}}.

Ngoài ra tôi còn khai báo thêm input có thuộc tính ng-model="username"

|  |
| --- |
| <input type="text" value="" ng-model="username"> |

Điều này có nghĩa rằng ta đang khai báo đây là một model với key của nó là **username**, muốn lấy dữ liệu của nó chỉ cần sử dụng **$scope** như sau "$scope.username", và muốn gán thì ta sẽ dùng cú pháp $scope.username = 'giá trị'

**2. Phạm vi ảnh hưởng của $scope**

Trong một ứng dụng AngularJS thì ta có thể có nhiều Controller, nhiều $scope khác nhau. Các bạn xem ví dụ dưới đây:

|  |
| --- |
| <!DOCTYPE html>  <html>      <head>          <title></title>          <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">          <script src="<http://ajax.googleapis.com/ajax/libs/angularjs/1.2.15/angular.min.js>">  </script>          <style>              .show-scope-demo.ng-scope,              .show-scope-demo .ng-scope  {                  border: 1px solid red;                  margin: 3px;              }              .show-scope-demo .ng-scope .ng-scope{                  border:solid 1px blue;              }          </style>      </head>      <body ng-app="myapp">          <div class="show-scope-demo">              <div ng-controller="GreetController">                  Hello {{name}}!              </div>              <div ng-controller="ListController">                  Hello {{name}}!              </div>          </div>          <script>              angular.module("myapp", [])                      .controller('GreetController', function ($scope){                          $scope.name = 'Nguyễn Văn Cường';                      })                      .controller('ListController', function ($scope){                          $scope.name = 'Học Lập Trình Online';                      });          </script>      </body>  </html> |

Trong đó đoạn code dưới đây là khai báo view thứ nhất:

|  |
| --- |
| <div ng-controller="GreetController">      Hello {{name}}!  </div> |

Tương ứng với controller khai báo trong view là đoạn JS:

|  |
| --- |
| .controller('GreetController', function ($scope){      $scope.name = 'Nguyễn Văn Cường';  }) |

Đoạn code sau khai bái view thứ hai:

|  |
| --- |
| <div ng-controller="ListController">      Hello {{name}}!  </div> |

Tương ứng với controller khai báo trong view là đoạn JS:

|  |
| --- |
| .controller('ListController', function ($scope){      $scope.name = 'Học Lập Trình Online';  }); |

Như vậy rõ ràng trong 2 view có hai giá trị {{name}} thì làm sao $scope nhận biết mà truyền dữ liệu được? Đó chính là do ta khai báo **ng-controller** ở div bên ngoài nên trong đoạn js nó hiểu là giá trị {{name}} nằm trong controller hiện tại. Chạy lên giao diện như sau:

phạm vi hoạt động của scope

**3. Tìm Hiểu $rootScope**

Tiếp tục ví dụ ở phần 2, bây giờ ta sẽ đổi một chút cú pháp của đoạn mã AngularJS như sau:

|  |
| --- |
| <script>      angular.module("myapp", [])              .controller('GreetController', function ($scope, $rootScope){                  $rootScope.name = 'Học Lập';              })              .controller('ListController', function ($scope){              });  </script> |

Trong đoạn ví dụ này sự khác biệt chính là ở đoạn mã JS Controller thứ nhất có thêm tham số $rootScope, và ở đoạn Controller thứ hai thì không xử lý gì cả. Chạy lên các bạn thấy giao diện như sau:

scope root

Như vậy rõ ràng đoạn code trên mình không truyền giá trị cho $scope ở cả 2 controller mà bên view vẫn có? Đó là vì biến $rootScope. Điều này có nghĩa là khi ứng dụng được chạy thì sẽ có một $rootScope được tự động tạo, **$rootScope** là bậc cao nhất nên sẽ bao quát hết các $scope bên trong nó, điêu này không giống với $scope là chỉ ảnh hưởng trong phạm vi của controller.

**4. $scope lồng nhau**

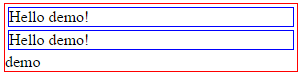
Bây giờ ta sẽ thêm một controller ngoài cùng bao cả hai controller như ở ví dụ trên, mã HTML lại như sau:

|  |
| --- |
| <div class="show-scope-demo">      <div ng-controller="TopController">          <div ng-controller="GreetController">              Hello {{name}}!          </div>          <div ng-controller="ListController">              Hello {{name}}!          </div>          {{name}}      </div>  </div> |

Và đoạn JS sửa lại chút:

|  |
| --- |
| angular.module("myapp", [])          .controller('TopController', function ($scope){              $scope.name = 'demo';          })          .controller('GreetController', function ($scope){          })          .controller('ListController', function ($scope){          }); |

Chạy lên giao diện như sau:

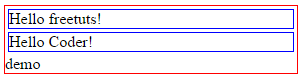


Các bạn thấy trong 2 controller bên trong mình không code gì cả mà tại sao view vẫn nhận được giá trị? Đó chính là do controller ngoài cùng mình khai báo $scope.name = 'demo' nên nó sẽ nhận cả bên trong.

Bây giờ bạn sửa đoạn code JS lại một xíu:

|  |
| --- |
| angular.module("myapp", [])          .controller('TopController', function ($scope){              $scope.name = 'demo';          })          .controller('GreetController', function ($scope){              $scope.name = 'freetuts';          })          .controller('ListController', function ($scope){              $scope.name = 'Coder';          }); |

Chạy lên giao diện sẽ như sau:



Thú vị phải không nào, theo nguyên tắc chạy thì controller thứ nhất ngoài cùng sẽ chạy trước và tất cả các {{name}} đều có giá trị là demo, nhưng các controller còn lại nó tự gán lại giá trị cho các {{name}}  của nó nên giá trị demosẽ bị ghi đè thành giá trị mới.

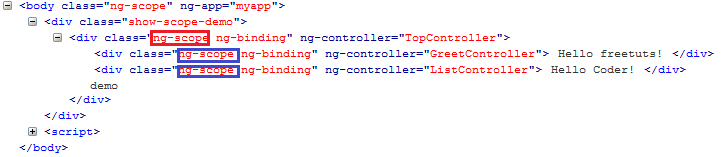
Khi nói tới **$rootScope** và **$scope lồng nhau** thì rất nhiều ví dụ nhưng mình chỉ đưa ra được bấy nhiêu đây thôi, còn lại các bạn tự test để rút ra bài học nhé vì mình không thể trình bày full ở đây được, mong các bạn thông cảm.

**5. Bật Firebug để xem các class tự tạo trong angularjs**

Tới đây không biết bạn có thắc mắc tại sao các kết quả trên có đường viền màu đỏ và màu xanh không? Đấy là do mình khai báo trong đoạn CSS:

|  |
| --- |
| .show-scope-demo .ng-scope  {      border: 1px solid red;      margin: 3px;  }  .show-scope-demo .ng-scope .ng-scope{      border:solid 1px blue;  } |

Đoạn CSS này báo rằng class **.ng-scope** thứ nhất sẽ là border màu đỏ, class **.ng-scope** thứ hai sẽ là border màu xanh. Nhưng đoạn mã HTML mình không có tạo class này mà tạo sao kết quả vẫn có? Đó là tại vì AngularJS tự động thêm vào đấy các bạn. Ứng dụng có bao nhiêu $scope (controller) thì nó sẽ tạo bấy nhiêu cái, nếu lồng nhau thì class đó sẽ tạo lồng nhau. Không tin các bạn xem hình dưới đây nhé:



Ô màu đỏ tượng trưng cho border màu đỏ, ô màu xanh tượng trưng cho border màu xanh.

**6. Lời kết**

Nói về $scope thật sự còn rất nhiều nhưng vì phạm vi bài quá dài nên mình sẽ tạm dừng ở đây, nếu chúng ta tiếp tục tìm hiểu **$scope nâng cao** thì bản thân mình thấy chưa cần thiết, vì chúng ta phải thực hành nhiều và nhiều hơn nữa, nếu đi quá nhiều thì bạn sẽ không nhớ bao nhiêu. Nên vấn đề này chúng ta sẽ tìm hiểu ở một bài nào đó tiếp theo trong serie này nhé.

**Bài 05: Tìm hiểu AngularJS Directives - Danh sách Directives**

Bài trước chúng ta tìm hiểu về Scope trong angularJS nhưng chưa bàn luận về các kiến thức sâu hơn như mô hình cấu trúc DOM, chu kỳ sống của Scope. Nhưng tôi quyết định bỏ qua và sẽ trình bày nó trong một bài gần nhất, lý do đơn giản là nếu tiếp tục tìm hiểu mà không có viết ứng dụng kết hợp với các kiến thức khác thì bạn sẽ khó hiểu bài, thay vì tiếp tục thì ta sẽ tìm hiểu một số vấn đề liên quan rồi sau đó quay ngược lại, lúc này bạn sẽ vỡ òa "sao mà nó đơn giản quá vậy"

Trong bài này chúng ta sẽ tìm hiểu khái niệm **Directives trong AngularJS**, bạn sẽ thấy có nhiều bất ngờ khi tìm hiểu nó đấy, lý do là chúng ta đã sử dụng nó trong các ví dụ ở bài trước mà không hề hay biết nó là Directives crying. Không lôi thôi dài dòng nữa, ta bắt đầu thôi.

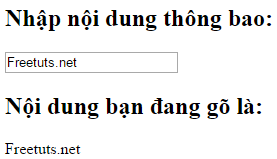
**1. AngularJS Directives là gì?**

Directives là một thành phần mở rộng HTML, hay nói cách khác là các thuộc tính (properties) của các thẻ HTML mà Angular nó định nghĩa thêm, vì nó của riêng của Angular nên phải tuân thủ theo nguyên tắc của nó là chữ bắt đầu luôn luôn là ký tự mà tôi bôi đỏ ng-prefix, trong đó tiền tố prefix là tên của derective mà chúng ta sử dụng. Như ở các ví dụ trước, để khai báo là một **Directive Controller** thì chúng ta khai báo ng-controller, quá đơn giản phải không nào.

**Ví dụ dưới đây chúng ta sử dụng ba Directives (ng-controller, ng-model, ng-bind)**:

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script language="javascript">                  angular.module('myapp', [])                  .controller('MyController', function($scope) {                      $scope.name = 'Chào mừng các bạn đến với website freetuts.net';                  });          </script>      </head>      <body ng-app="myapp">          <div ng-controller="MyController">              <h2>Nhập nội dung thông bao:</h2>              <input ng-model="name"> <br/>              <h2>Nội dung bạn đang gõ là:</h2>              <span ng-bind="name"></span>        </div>      </body>  </html> |

**Giao diện**:



Trong ví dụ này mục đích là khi nhập dữ liệu vào ô input thì dòng ở dưới sẽ thay đổi theo. Nếu chúng ta sử dụng javascript để thực hiện thì ta sẽ bắt sự kiện onkeypress cho input và đổi giá trị cho thẻ span. Nhưng với AngularJS thì rất đơn giản, chúng ta sẽ sử dụng **Directive ng-model** để khai báo model và dùng **Directive ng-bind** để lấy data từ model. Nhưng để hai Directives ng-model và ng-bind "*biết đến nhau*"  thì giá trị của nó phải giống nhau nhé (ng-model="name", ng-bind="name").

Các bạn để ý rằng trong đoạn JS tôi chỉ khai báo module, controller nhưng không xử lý gì bên trong. Vậy bạn có thắc mắc là tại sao không code xử lý gì mà lại khai báo chi cho mệt? Lưu ý với bạn đấy là điều bắt buộc nhé, nghĩa là ta phải khai báo để AngularJS hiểu, nếu không nó sẽ báo lỗi đấy.

**2. Danh sách AngularJS Directives**

Nói đến danh sách **Directives trong AngularJS** thì mình không thể kể ra hết được, để xem được hết danh sách bạn lên trang chủ [Angular API](https://docs.angularjs.org/api) để đọc nhé. Trong bài này mình chỉ nêu ra một vài ví dụ đơn giản cho các bạn dễ hiểu hơn thôi.

**ng-app**

Khi chúng ta khai báo một Directive ng-app thì AngularJS sẽ hiểu là bắt đầu một ứng dụng Angular, nó sẽ xác định đây là thẻ gốc (element root) và tiếp theo sẽ khởi tạo các thông số cấu hình bên trong mà ta gọi là bootstraps. Ngoài ra ng-app còn được sử dụng để mô tả các module khác nhau trong ứng dụng. Về ví dụ thì chúng ta demo rất nhiều rồi nên tôi không trình bày thêm về nó nũa.

**ng-init**

Trong lập trình hướng đối tượng thì chúng ta có khái niệm hàm khởi tạo (constructor), thì ng-init cũng vậy, nó có ý nghĩa là khai báo dữ liệu khởi tạo khi ứng dụng vừa được chạy. Các dữ liệu này sẽ được dùng cho toàn bộ phạm vi của controller mà nó thuộc về. Ví dụ:

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script language="javascript">                  angular.module('myapp', [])                  .controller('MyController', ['$scope', function($scope) {}]);          </script>      </head>      <body ng-app="myapp">         <div ng-controller="MyController">              <div my-customer ng-init="demo='Nội dung'">{{demo}}</div>              {{demo}}          </div>          {{demo}}      </body>  </html> |

**Giao diện**:

ng-init angularjs

Các bạn thấy rõ ràng mình khai báo ba vị trí khác nhau với cùng key là demo ({{demo}}) nhưng chỉ có hai vị trí nhận được dữ liệu, lý do đơn giản là vị trí cuối cùng năm ngoài Controller của data-init nên không thể nhận được.

**ng-...**

Còn rất nhiều nữa nhưng bài này mình chỉ mang tính chất giới thiệu nên sẽ không nói thêm về vụ này nữa, chúng ta sẽ lần lượt tìm hiểu từng Directives cụ thể trong các bài sau nhé.

**3. Tự định nghĩa Directives**

Cái này thú vị nhất nè, lúc trước mình cứ nghĩ là **danh sách AngularJS Directives**là cố đinh, điều này là một sai lầm nhé. Nghĩa là ngoài các Directives mặc định có sẵn trong Angular thì chúng ta có thể tự tạo được.

**Ví dụ 1**: Xây dựng một Directive xuất hiện lời chào với nội dung "*Chào mừng các bạn đến với website freetuts.net*".

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script language="javascript">                  angular.module('myapp', [])                  .controller('MyController', function($scope) {                      //                  })                  .directive('myDirective', function() {                      return {                          template: '<h1>Chào mừng các bạn đến với website freetuts.net</h1>'                      };                  });          </script>      </head>      <body ng-app="myapp">         <div ng-controller="MyController">              <div my-directive></div>          </div>      </body>  </html> |

**Giao diện**:



Trong ví dụ này mình đã xây dựng một div và khai báo nó là một Directive <div my-directive></div>. Trong đoạn code JS mình sử dụng hàm khai báo một directive là:

|  |
| --- |
| .directive('myDirective', function() {      return {          template: '<h1>Chào mừng các bạn đến với website freetuts.net</h1>'      };  }); |

Trong đó **myDirective** tương ứng với **my-directive**, nghĩa là khi bạn tạo tên directive trong đoạn code JS thì tiền tố đầu tiên và tiền tố sau được nối lại và tiền tố sau viết chữ in hoa. Bên trong mình xử lý return về một object có key là **template,**key này chính là nội dung HTML bên trong của directive này, không tin bạn đổi giá trị return này và xem lại giao diện nhé.

**Ví dụ 2**: Xây dựng form login bằng cách định nghĩa Directive tên my-formregister.

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script language="javascript">                  angular.module('myapp', [])                  .controller('MyController', function($scope) {                      //                  })                  .directive('myFormregister', function() {                      var html = '<table>';                          html += '<tr>';                              html += '<td>Username:</td>';                              html += '<td><input type="text"/></td>';                          html += '</tr>';                          html += '<tr>';                              html += '<td>Password:</td>';                              html += '<td><input type="password"/></td>';                          html += '</tr>';                          html += '<tr>';                              html += '<td></td>';                              html += '<td><input type="Button" value="Login"/></td>';                          html += '</tr>';                      html += '</table>';                        return {                          template: html                      };                  });          </script>      </head>      <body ng-app="myapp">         <div ng-controller="MyController">              <div my-formregister></div>          </div>      </body>  </html> |

**Giao diện**:

:

Rất rõ ràng phải không nào, sự khác biệt so với **ví dụ 1** là đoạn code trong javascript return về mã html form login.

**4. Lời kết**

Tương tự như phần Scope thì chúng ta sẽ ngưng tại đây nhé, không nên đọc nhiều nội dung quá trong một bài vì bạn sẽ khó nhớ đến nó, hãy học thật chắc, nhớ thật dai và đi theo từng bước, hãy học đi rồi học chạy nhé.

**Bài 06: Directive ng-model trong AngularJS**

**1. Ng-model trong AngularJS là gì ?**

ng-model hay còn gọi là **ngModel**là một Directive dùng để liên kết dữ liệu với client, nghĩa là nó thường được dùng để cho người dùng nhập liệu nên ta hay sử dụng trong FORM html. Ở bài Scope trong Angularjs chúng ta cũng đã làm một ví dụ về ng-model mà có lẽ bạn vẫn còn nhớ, ở bài đó chúng ta đã kết hợp ng-model với ng-bind để xây dựng một ứng dụng nho nhỏ. Về định nghĩa ta nói chính xác hơn thì ng-model (**ngModel**) sẽ liên kết với một thuộc tính của Scope sử dụng **ngModelController** (được khởi tạo và  bởi ng-model). Còn về ng-bind tạm thời chúng ta có thể hiểu đơn giản dùng để liên kết dữ liệu với ng-model, còn nâng cao hơn chúng ta sẽ tìm hiểu ở một bài khác nhé.

**2. Ng-model trong AngularJS có nhiệm vụ gì?**

**ng\_model có nhiệm vụ :**

* Liên kết View trong model và một số directives khác như input, textarea hoặc select.
* Cung cấp các thao thác validate dữ liệu như kiểm tra kiểu dữ liệu có phải là số, là email, ...
* Kiểm soát thông tin từ client nhập vào có hợp lệ hay không và xuất thông báo lỗi
* Thiết lập các css class trong thẻ HTML (ng-valid, ng-invalid, ng-dirty, ng-pristine, ng-touched, ng-untouched) và các hiệu ứng thông báo
* ng-model sẽ cố gắng liên kết với các giá trị được khai báo (ví dụ {{name}}), nếu không tồn tại thì nó sẽ được **tạo ngầm và lưu vào Scope**. Điều này có nghĩa rằng  nếu ta khai báo ng-model="somename" thì khi ứng dụng chạy lên trong $scope sẽ tồn tại một giá trị $scope.somename, nên trong controller ta muốn xử lý gán gái trị cho model thì chỉ cần dùng  $scope để thay đổi. Ở ví dụ cuối cùng của bài này chúng ta có đề cập đến vấn đề này đấy.

**3. Danh sách các CSS được ng-model thêm vào**

Sau đây là danh sách các class được ng-model tự thêm vào cho từng trường hợp cụ thể

* ng-valid: model is valid
* ng-invalid: model is invalid
* ng-valid-[key]: for each valid key added by $setValidity
* ng-invalid-[key]: for each invalid key added by $setValidity
* ng-pristine: the control hasn't been interacted with yet
* ng-dirty: the control has been interacted with
* ng-touched: the control has been blurred
* ng-untouched: the control hasn't been blurred
* ng-pending: any $asyncValidators are unfulfilled

Thật sự tôi không biết dịch sao cho các class này nữa nên thay vì cố gắng rặn từng chữ thì chúng ta sẽ tìm hiểu dần, từ đó bạn sẽ hiểu được ý nghĩa của từng class. Như vậy sẽ tốt hơn là ngồi giải thích một loạt mà không hề có một ví dụ cụ thể nào. Trong các ví dụ dưới này chúng ta có tìm hiểu một số class phía trên đấy.

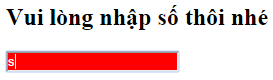
**4. Các ví dụ về ng-model trong angularjs**

**ng-pattern**

**Ví dụ**: Xây dựng FORM và kiểm tra dữ liệu nhập vào input có phải là các chữ số hay không

|  |
| --- |
| <html>      <head>          <title>Angular JS Model</title>          <script src="<http://ajax.googleapis.com/ajax/libs/angularjs/1.2.15/angular.min.js>">  </script>          <script>              angular.module('MyForm', [])                      .controller('ExampleController', ['$scope', function($scope) {              }]);          </script>          <style>              .my-input {                  -webkit-transition:all linear 0.5s;                  transition:all linear 0.5s;                  background: transparent;              }              .my-input.ng-invalid {                  color:white;                  background: red;              }          </style>      </head>      <body ng-app="MyForm">          <form name="testForm" ng-controller="ExampleController">              <input ng-model="val" ng-pattern="/^\d+$/" name="anim" class="my-input" />          </form>      </body>  </html> |

**Giao diện**:



Các bạn thấy mình đã nhập dữ liệu là chữ s nên không đúng với yêu cầu, còn yêu cầu như thế nào thì ta sẽ mổ xẻ ví dụ đó nhé.

Trong đoạn code CSS dưới đây mình định nghĩa lại hiệu ứng CSS cho class ng-invalid cho background màu đỏ:

|  |
| --- |
| <style>      .my-input {          -webkit-transition:all linear 0.5s;          transition:all linear 0.5s;          background: transparent;      }      .my-input.ng-invalid {          color:white;          background: red;      }  </style> |

Còn trong form HTML mình  tạo một ng-model = "val", một pattern ng-pattern="/^\d+$/". khi chạy lên thì n**gModel** thấy có khai báo pattern nên dữ liệu nhập vào sẽ được validate bởi pattern này. Nếu bạn chưa biết về Pattern thì có thể tham khảo serie [Regular Expression](http://freetuts.net/tag/regular-expression) này nhé. Bây giờ bật firebug lên thì bạn sẽ thấy các class css mà ng-model tự thêm vào như sau:

tìm hiểu ng-partern

**Trong đó**:

* Class màu đà (ng-dirty) là do ta từng nhập dữ liệu rồi nên nó được thêm vào
* Class màu xanh (ng-invalid) là do trạng thái của input không đúng nên nó được thêm vào.
* Class màu đỏ (ng-invalid-pattern) là do pattern bị sai nên nó được thêm vào

**Lưu ý:**Trạng thái class của các input là cho từng input, còn của form là được quyết định bởi toàn bộ các input, nghĩa là nếu toàn bộ các input trong form đúng thì form đó mới đúng, không tin bạn thử thêm một input như vậy nữa và nhập vào một cái đúng một cái sai và soi debug thử nhé.

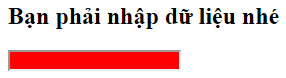
**Required**

Nếu bạn đã học qua HTML5 thì required là một thuộc tính của HTML5 dùng để yêu cầu bắt buộc nhập dữ liệu vào. Trong AngularJS nếu bạn khai báo nó trong các thẻ input thì nó cũng được coi là một **Directive của AngularJS**.

**Ví dụ**: Kiểm tra bắt buộc nhập dữ liệu vào thẻ input.

|  |
| --- |
| <html>      <head>          <title>Angular JS Model</title>          <meta charset='utf8'/>          <script src="<http://ajax.googleapis.com/ajax/libs/angularjs/1.2.15/angular.min.js>">  </script>          <script>              angular.module('MyForm', [])                      .controller('ExampleController', ['$scope', function($scope) {              }]);          </script>          <style>              .my-input {                  -webkit-transition:all linear 0.5s;                  transition:all linear 0.5s;                  background: transparent;              }              .my-input.ng-invalid {                  color:white;                  background: red;              }          </style>      </head>      <body ng-app="MyForm">          <form name="testForm" ng-controller="ExampleController">              <h2>Vui lòng nhập dữ liệu nhé</h2>              <input ng-model="val" required name="anim" class="my-input" />          </form>      </body>  </html> |

**Giao diện**:



Lý do ta không nhập gì mà vẫn bị màu đỏ là do thuộc tính required đấy. Bây giờ bạn bật firebug lên và xem các class của nó thay đổi như thế nào nhé, tôi không làm thao tác này nữa vì mệt quá :D.

**Lưu ý**: Ngoài thuộc tính required trên ta có thể thay bằng ng-required, ví dụ <input ng-model="val" ng-required='true' name="anim" class="my-input" />

**Các Directives validate khác**

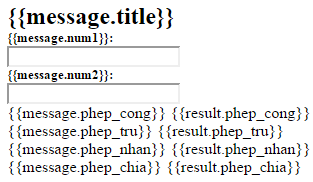
Còn rất nhiều nữa nhưng có lẽ mình sẽ trình bày trong bài tới, trong bài này chúng ta chỉ tìm hiểu 2 directive này cho bạn hiểu thôi nhé. Bây giờ ta sẽ làm một ví dụ nho nhỏ là làm chương trình máy tính tính tổng, hiệu, thương và tích của hai số nhé.

**Ví dụ**: Tạo 2 input và cho người dùng nhâp vào 2 số, validate bắt buộc nhập vào phải là số đồng thời xuất ra màn hình kết quả cộng trừ nhân chia của hai số đó.

**Bước 1**: Xây dựng FORM

|  |
| --- |
| <body ng-app="MyForm">      <form name="calForm" ng-controller="ExampleController">          <h2>{{message.title}}</h2>          <h5>{{message.num1}}:</h5>          <input ng-model="so\_thu\_nhat" ng-required='true' ng-pattern="/^[0-9]+$/" class="my-input" ng-keyup="show\_result()" />          <h5>{{message.num2}}:</h5>          <input ng-model="so\_thu\_hai" ng-required='true' ng-pattern="/^[0-9]+$/" class="my-input" ng-keyup="show\_result()" />          <div style='{{styleresult}}'>              {{message.phep\_cong}} {{result.phep\_cong}}<br/>              {{message.phep\_tru}}  {{result.phep\_tru}}<br/>              {{message.phep\_nhan}} {{result.phep\_nhan}}<br/>              {{message.phep\_chia}} {{result.phep\_chia}}          </div>      </form>  </body> |

Giao diện lúc này như sau:



Lý do nó hiển thị y nguyên HTML ban đầu như vậy là ta chưa khai báo Angular bằng JS

**Bước 2**: Xây dựng giá trị ban đầu cho FORM

|  |
| --- |
| angular.module('MyForm', [])  .controller('ExampleController', ['$scope', function($scope)  {      // Khởi tạo giá trị ban đầu      $scope.message = {          title : 'Trò Chơi Tính Toán',          num1 : 'Số thứ nhất',          num2: 'Số thứ Hai',          phep\_cong : "Cộng hai số:",          phep\_tru : "Trừ hai số:",          phep\_nhan : "Nhân hai số:",          phep\_chia : "Chia hai số:"      };      // vì ban đầu chưa nhập gì nên ẩn khung kết quả      $scope.styleresult = 'display:none';    }]); |

Giao diện lúc này như sau:



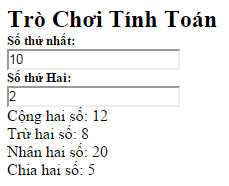
**Bước 3**: Viết code cho sự kiện ng-keyup (**giống onkeyup trong Javascript**) cho 2 input. Bạn tiếp tục thêm đoạn mã Javascript sau bên dưới đoạn code xây dựng giá trị ban đầu cho FORM

|  |
| --- |
| // Khi nhập các số vào các input thì gọi sự kiện này  $scope.show\_result = function()  {      // Nếu validate form đúng      if ($scope.calForm.$valid){          $scope.styleresult = 'display:block';          $scope.result = {              phep\_cong : parseInt($scope.so\_thu\_nhat) + parseInt($scope.so\_thu\_hai),              phep\_tru : parseInt($scope.so\_thu\_nhat) - parseInt($scope.so\_thu\_hai),              phep\_nhan : parseInt($scope.so\_thu\_nhat) \* parseInt($scope.so\_thu\_hai),              phep\_chia : parseInt($scope.so\_thu\_nhat) / parseInt($scope.so\_thu\_hai)          };      }      // nếu validate form sai thì ẩn result      else {          $scope.styleresult = 'display:none';      }  }; |

**Tại bước này bạn cần chú ý**:

* Tôi sử dụng tên của form để kiểm tra trạng thái của form có tên là calForm (**$scope.calForm.$valid**)
* Vì tôi khai báo 2 ng-model ở hai input nên khi chạy ứng dụng 2 input này sẽ được tự thêm vào $scope, vì thế muốn sử dụng nó chỉ cần dùng cú pháp $scope.name ($scope.so\_thu\_nhat, $scope.so\_thu\_hai)

Chạy lên bạn nhập vào hai số và kết quả sẽ là:



**Toàn bộ code cho ví dụ này**:

|  |
| --- |
| <html>      <head>          <title>Angular JS Model</title>          <meta charset='utf8'/>          <script src="<http://ajax.googleapis.com/ajax/libs/angularjs/1.2.15/angular.min.js>">  </script>          <style>              \*{margin:0;}              body{padding: 20px}              .my-input {                  -webkit-transition:all linear 0.5s;                  transition:all linear 0.5s;                  background: transparent;              }              .my-input.ng-invalid {                  color:white;                  background: red;              }          </style>          <script>              angular.module('MyForm', [])                      .controller('ExampleController', ['$scope', function($scope)                  {                      // Khởi tạo giá trị ban đầu                      $scope.message = {                          title: 'Trò Chơi Tính Toán',                          num1: 'Số thứ nhất',                          num2: 'Số thứ Hai',                          phep\_cong: "Cộng hai số:",                          phep\_tru: "Trừ hai số:",                          phep\_nhan: "Nhân hai số:",                          phep\_chia: "Chia hai số:"                      };                      $scope.styleresult = 'display:none';                        // Khi nhập các số vào các input thì gọi sự kiện này                      $scope.show\_result = function()                      {                          // Nếu validate form đúng                          if ($scope.calForm.$valid) {                              $scope.styleresult = 'display:block';                              $scope.result = {                              phep\_cong: parseInt($scope.so\_thu\_nhat) + parseInt($scope.so\_thu\_hai),                              phep\_tru: parseInt($scope.so\_thu\_nhat) - parseInt($scope.so\_thu\_hai),                              phep\_nhan: parseInt($scope.so\_thu\_nhat) \* parseInt($scope.so\_thu\_hai),                              phep\_chia: parseInt($scope.so\_thu\_nhat) / parseInt($scope.so\_thu\_hai)                              };                          } else { // nếu validate form sai thì ẩn result                              $scope.styleresult = 'display:none';                          }                      };                  }]);          </script>      </head>      <body ng-app="MyForm">          <form name="calForm" ng-controller="ExampleController">              <h2>{{message.title}}</h2>              <h5>{{message.num1}}:</h5>              <input ng-model="so\_thu\_nhat" ng-required='true' ng-pattern="/^[0-9]+$/" class="my-input" ng-keyup="show\_result()" />              <h5>{{message.num2}}:</h5>              <input ng-model="so\_thu\_hai" ng-required='true' ng-pattern="/^[0-9]+$/" class="my-input" ng-keyup="show\_result()" />              <div style='{{styleresult}}'>                  {{message.phep\_cong}} {{result.phep\_cong}}<br/>                  {{message.phep\_tru}}  {{result.phep\_tru}}<br/>                  {{message.phep\_nhan}} {{result.phep\_nhan}}<br/>                  {{message.phep\_chia}} {{result.phep\_chia}}              </div>          </form>      </body>  </html> |

**5. Lời kết**

Tới đây thật sự mình quá mỏi tay rồi không gõ thêm được chữ nào nữa nên có lẽ ta sẽ ngưng ở đây, bài tuy ngắn nhưng rất khó đấy nhé, nhất là ở ví dụ cuối cùng nó hơi khó cho những bạn mới tìm hiểu AngularJS. Hy vọng qua bài này bạn hiểu được **Directive ng-model trong angularJS**  là gì và cách sử dụng nó trong một số ví dụ ở trên

**Bài 07: Directive ng-model-options trong AngularJS**

Bài trước chúng ta đã tìm hiểu một Directive rất quan trọng đó là [ng-model trong AngularJS](http://freetuts.net/directive-ng-model-trong-angularjs-206.html), và chúng ta cũng làm được khá nhiều ví dụ cho chủ đề ng-model này. Vậy để tiếp tục theo luồng serie thì trong bài này chúng ta sẽ tìm hiểu một Directive khác có liên quan mật thiết đến ng-model đó là **ng-model-options trong AngularJS**.

Tôi cũng xin nhắc lại nếu bạn là người mới tìm hiểu thì nên đọc kỹ những nội dung mà tôi viết chứ đừng coi ví dụ không thôi, như vậy bạn sẽ không hiểu được ý đồ của mình đâu.

**1. ng-model-options là gì?**

Dịch cái tên ra cũng đủ hiểu ý nghĩa của nó rồi, chúng ta thường sử dụng [Directive](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html) **ngModelOptions** dùng để cấu hình một số thông số liên quan tới ng-model. Khi ứng dụng được chạy lên thì Directive này sẽ thực thi trước và sau đó **ng-model mới được tạo**.

Những thông số cấu hình này sẽ quyết định đến giá trị của model có được thay đổi và lưu vào $scope hay không? Nếu như lưu thì khi nào lưu? và cách lưu như thế nào? Tất cả những câu hỏi này chính là những ngModelOptions trong angular đấy;

**2. Tìm hiểu $rollbackViewValue**

Như các bạn thấy mỗi khi dữ liệu thay đổi thì sẽ được lưu vào trong $scope nên những thuộc tính nào đang được liên kết (ng-bind) với nó sẽ thay đổi theo, nhưng có những trường hợp ta muốn người dùng nhập nhưng không thay đổi nó trong $scope và ta cần lấy giá trị nguyên thủy này thì phải làm thế nào? Rất đơn giản, $roolbackViewValue sẽ giải quyết vấn đề này cho bạn. Vậy $rollbackViewValue chính là giá trị hiện tại của các thẻ input (model) được lưu trong mô hình của chúng ta.

Để gọi tới một view (giá trị thẻ input có khai báo model) nào đó trong form ta sẽ dùng cú pháp như sau trong controller:

|  |
| --- |
| $scope.formName.inputName.$rollbackViewValue; |

Tại sao lại thông qua $scope? Câu trả lời mình cũng đã nhắc ở bài trước nay mình nhắc lại luôn. Lý do là khi ứng dụng được chạy lên thì các model (form, input, ...) trong Controller hiện tại sẽ được khởi tạo và lưu vào object [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) với key tương ứng name của nó. Nếu input nằm trong Form thì ta có thể dùng cú pháp trên để gọi tới nó.

**3. Các tham số của ng-model-options trong AngularJS**

Các thông số options dưới đây sẽ được gắn vào [Model](http://freetuts.net/directive-ng-model-trong-angularjs-206.html) hiện tại:

* **updateOn**: Quyết định khi nào thì thao tác update dữ liệu vào trong $scope, thường là blur, default, kepress ....
* **debounce**: Giá trị của nó kiểu INT và nó quyết định sau khoảng bao lâu thì thao tác update dữ liệu vào $scope được thực hiện
* **getterSetter**: Chúng ta sẽ tìm hiểu thông qua phần 4 nhé, đơn giản nó là những phương thức get, set giống trong C++, C# hay PHP.
* Còn hai options nữa là **allowInvalid, timezone**nhưng chúng ta sẽ tìm hiểu nó sau nhé.

**4. Các ví dụ về ng-model-options trong AngularJS**

Sau đây chúng ta sẽ tìm hiểu một vài ví dụ liên quan đến **Directive ng-model-options** nhé, hãy chắc chắn rằng bạn đã đọc qua lý thuyết ở trên kia trước khi vào các ví dụ này.

**Cách sử dụng ng-model-options (updateOn)**

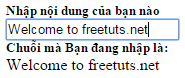
updateOn dùng để thiết lập khi nào thì thay đổi giá trị cho model

Trong ví dụ này ta sẽ thực hiện một chương trình như sau, chúng ta sẽ có một thẻ input và một thẻ span dùng để hiển thị giá trị của nó lúc nhập liệu. Chúng ta sẽ chia ra làm 2 bài, 1 bài có sử dụng options updateOn và một bài không có sử dụng.

**Bài 1: Không sử dụng ng-model-options updateOn**

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <style>\*{margin:0}body{padding:20px}</style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script language="javascript">              angular.module('myapp', [])                      .controller('optionsController', ['$scope', function($scope) {                      }]);          </script>      </head>      <body ng-app="myapp">          <div ng-controller="optionsController">              <form name="userForm">                  <h5>Nhập nội dung của bạn nào</h5>                  <input type="text" name="userName" ng-model="user.name"/><br />                  <h5>Chuỗi mà Bạn đang nhập là:</h5>                  <span ng-bind="user.name"></span>              </form>          </div>      </body>  </html> |

**Giao diện**:

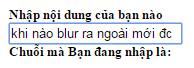


Khi bạn nhập dữ liệu thì bên dưới sẽ thay đổi theo bởi vì mình sử dụng cơ chế liên kết data binding (ng-bind).

**Bài 2: Có sử dụng ng-model-options updateOn**

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <style>\*{margin:0}body{padding:20px}</style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script language="javascript">              angular.module('myapp', [])                      .controller('optionsController', ['$scope', function($scope) {                      }]);          </script>      </head>      <body ng-app="myapp">          <div ng-controller="optionsController">              <form name="userForm">                  <h5>Nhập nội dung của bạn nào</h5>                  <input type="text" name="userName" ng-model-options="{updateOn:'blur'}" ng-model="user.name"/><br />                  <h5>Chuỗi mà Bạn đang nhập là:</h5>                  <span ng-bind="user.name"></span>              </form>          </div>      </body>  </html> |

**Giao diện**:



Các bạn thấy ở ví dụ này mình có thêm thuộc tính ng-model-options="{updateOn:'blur'}" có nghĩa rằng khi ta nhập dữ liệu và con trỏ chuột thoát khỏi ô input đó thì sự kiện update lưu vào trong $scope mới thực hiện chính vì vậy nó không thay đổi khi ta đang nhập.

**Cách sử dụng ng-model-options (debounce)**

debounce dùng để khai báo một khoảng thời gian nào đó thì sự kiện update giá trị vào $sope mới thực hiện.

**Ví dụ**: Trong ví dụ này ta sẽ xây dựng một ứng dụng tương tự như hai ví dụ trên, chỉ khác ở chỗ khi ta ngưng gõ khoảng thời gian  1 giây thì sự kiện update giá trị vào $scope mới thực hiện.

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <style>\*{margin:0}body{padding:20px}</style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js">  </script>          <script language="javascript">              angular.module('myapp', [])                      .controller('optionsController', ['$scope', function($scope) {                      }]);          </script>      </head>      <body ng-app="myapp">          <div ng-controller="optionsController">              <form name="userForm">                  <h5>Nhập nội dung của bạn nào</h5>                  <input type="text" name="userName"  ng-model-options="{debounce:1000}" ng-model="user.name"　/>  <br />                  <h5>Chuỗi mà Bạn đang nhập là:</h5>                  <span ng-bind="user.name"></span>              </form>          </div>      </body>  </html> |

Chạy lên bạn gõ vào và ngưng gõ khoảng 1 giây thì dữ liệu sẽ được lưu vào $scope, nên ng-bind sẽ lấy trong $scope và in ra ngoài màn hinh ngay tại thẻ span (thẻ ta khai báo ng-bind).

**Kết hợp debouce vaf updateOn**

Chuyện gì sẽ xảy ra nếu ta kết hợp cả hai options này lại với nhau? Rất đơn giản nếu ta sử dụng nhiều options với nhau thì nó sẽ chạy từng options cho tới khi xong hết thì mới cập nhật $scope. Ví dụ dưới đây ta sẽ nhập vào và khi con trỏ chuột ra khỏi input thì khoảng 1 giây sau thì $scope được cập nhật, còn nếu ta không đưa con trỏ ra ngoài thì hành động cập nhật $scope này không được xử lý.

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <style>\*{margin:0}body{padding:20px}</style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script language="javascript">              angular.module('myapp', [])                      .controller('optionsController', ['$scope', function($scope) {                      }]);          </script>      </head>      <body ng-app="myapp">          <div ng-controller="optionsController">              <form name="userForm">                  <h5>Nhập nội dung của bạn nào</h5>                  <input type="text" name="userName" ng-model-options="{updateOn:'blur', debounce:1000}" ng-model="user.name"/><br />                  <h5>Chuỗi mà Bạn đang nhập là:</h5>                  <span ng-bind="user.name"></span>              </form>          </div>      </body>  </html> |

**ng-model-options getterSetter**

Bạn có từng nghe qua phương thức get, set trong C#, C++ hay thậm chí là [PHP](http://freetuts.net/php) không nhỉ? Chắc là có rồi nên mình không nói nhiều nữa về vấn đề này nhé. Ví dụ dưới đây có sử dụng getterSetter, các bạn để ý kỹ nhé.

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <style>\*{margin:0}body{padding:20px}</style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script language="javascript">              angular.module('myapp', [])              .controller('optionsController', ['$scope', function($scope) {                  var \_name = 'Freetuts.net';                  $scope.user = {                      name: function(newName) {                              return angular.isDefined(newName) ? (\_name = newName) : \_name;                          }                      };                  }]);          </script>      </head>      <body ng-app="myapp">          <div ng-controller="optionsController">              <form name="userForm">                  <h5>Nhập nội dung của bạn nào</h5>                  <input type="text" name="userName" ng-model-options="{ getterSetter: true }"  ng-model="user.name"/><br />                  <h5>Chuỗi mà Bạn đang nhập là:</h5>                  <span ng-bind="user.name()"></span>              </form>          </div>      </body>  </html> |

Trong ví dụ này khi chạy lên lúc ta chưa nhập gì thì input có giá trị là **freetuts.net**, tại sao? Lý do chính là ta sử dụng thuộc tính **getterSetter**trong ng-model-options, giá trị của nó sẽ là true hoặc false, nếu true thì cho phép sử dụng getterSetter và ngược lại nếu false thì không có sử dụng getterSetter.

Các bạn để ý thêm đoạn mã JS dưới đây:

|  |
| --- |
| var \_name = 'Freetuts.net';  $scope.user = {      name: function(newName) {          return angular.isDefined(newName) ? (\_name = newName) : \_name;      }  }; |

trong đoạn này ta sử dụng hàm **angular.isDefined** để kiểm tra biến đó có được định nghĩa hay không, nếu có thì gán giá trị mới đó vào trong biến \_name, còn không thì không gán gì và cuối cùng là trả \_name về (*biến \_name được khởi tạo ban đầu bằng 'freetuts.net'*).

Như vậy ở đoạn mã html <span ng-bind="user.name()"></span> sẽ nhận được giá trị đó nên sẽ in ra ngoài trình duyệt.

**5 Lời kết**

Cũng hơi dài rồi nên mình sẽ ngưng bài viết tại đây, các bạn chú ý then chốt của bài này là hướng dẫn các bạn sử dụng **Directive ngModelOptions** để kết hợp với ngModel, mục đích cuối cùng là giúp bạn có cái nhìn tổng quát nên vẫn không thật sự đầy đủ cho lắm, các bạn lên mạng vào các trang nước ngoài và tìm hiểu thêm nhé

**Bài 08: Directives ng-bind - ng-bind-html - ng-bind-template trong AngularJS**

Ở các bài trước mình đã sử dụng ng-bind rất nhiều ở các ví dụ nên có lẽ bạn cũng không thấy xa lạ gì nữa. Nhưng mình vẫn muốn tạo một bài viết riêng như thế này mục đích giúp các bạn có thể tìm kiếm dễ dàng hơn và cũng như tìm hiểu kỹ về **Directive ng-bind** hơn. Trong bài này không những tìm hiểu về ngBind mà chúng ta tìm hiểu luôn ngBindHtml (ng-bind-html), một directive khác có công dụng cũng giống như ngBind.

**1. Directive ng-bind trong angularjs**

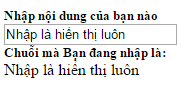
**Ng-bind** sẽ được biên dịch ở cấp 0, nghĩa là cấp đầu tiên của ứng dung. Khi một thẻ HTML nào đó có sử dụng ng-bind thì AngularJS sẽ tim và thay thế nội dung bên trong thẻ HTML đó bởi một giá trị nào đó mà có cùng tên định nghĩa (*tôi gọi là biểu thức A*), khi biểu thức A này thay đổi thì trong thẻ HTML vẫn sẽ thay đổi theo (cơ chế liên kết binding).

Trong thực tế chúng ta không nên sử dụng cơ chế **ng-bind** một cách trực tiếp, thay vào đó sử dụng quy tắc mở hai cặp dấu ngoặc nhọn {{key}}. Tốt hơn hết là **nên sử dụng ng-bind**trong trường hợp ta kết hợp với ng-model và ng-model-options, nếu bạn chưa rõ thì quay lại hai bài này và đọc các ví dụ nhé.

**Ví dụ sử dụng ng-bind:**

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <style>\*{margin:0}body{padding:20px}</style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script language="javascript">              angular.module('myapp', [])              .controller('bindingController', ['$scope', function($scope) {              }]);          </script>      </head>      <body ng-app="myapp">          <div ng-controller="bindingController">              <form name="userForm">                  <h5>Nhập nội dung của bạn nào</h5>                  <input type="text" name="userName" ng-model="name"/><br />                  <h5>Chuỗi mà Bạn đang nhập là:</h5>                  <span ng-bind="name"></span>              </form>          </div>      </body>  </html> |

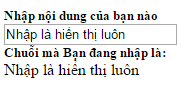
**Giao diện**:



**Ví dụ không sử dụng ng-bind**

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <style>\*{margin:0}body{padding:20px}</style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script language="javascript">              angular.module('myapp', [])              .controller('bindingController', ['$scope', function($scope) {              }]);          </script>      </head>      <body ng-app="myapp">          <div ng-controller="bindingController">              <form name="userForm">                  <h5>Nhập nội dung của bạn nào</h5>                  <input type="text" name="userName" ng-model="name"/><br />                  <h5>Chuỗi mà Bạn đang nhập là:</h5>                  <span>{{name}}</span>              </form>          </div>      </body>  </html> |

**Giao diện**:



**Kết luận**

Cả hai ví dụ trên đều có kết quả giống nhau, nhưng theo mình nghĩ thì với bài toán này chúng ta nên sử dụng ng-bind nếu như muốn quản lý được quá trình lưu model vào [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) (xem lại bài [ng-model-options](http://freetuts.net/directive-ng-model-options-trong-angularjs-207.html)), còn không thf sử dụng biểu thức {{name}} sẽ tốt hơn.

**2. Directive ng-bind-html trong AngularJS**

Quay lại ví dụ trên bạn thử nhập nội dung là thẻ HTML xem kết quả như thế nào? Nó sẽ in dạng text ra luôn chứ không hiểu chúng ta đang nhập thẻ HTML. Vậy giờ ta muốn nhập nội dung nếu là thẻ HTML thì khi in ra trình duyệt sẽ ở dạng HTML chứ không phải dạng text như trên? Bài toán này được giải quyết bởi [**Directive**](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html)**ngBindHtml trong AngularJS**.

**Lưu ý**:

* Directive ng-bind-html sử dụng thêm một thư viện phụ có tên là sanitize nên bắt buộc a phải thêm nó vào thì chương trình mới chạy.  
  <script　src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/　angular-sanitize.js"></script>
* Phải khai báo cho ứng dụng biết có sử dụng thư viện phụ này bằng cách thêm tên của nó (ngSanitize) vào trong danh sách các object có sử dụng ở hàm **angular.module**

(cú pháp là angular.module('tenModule', [danh\_sach\_object]))

Để rõ ràng hơn thì ta xem ví dụ dưới đây.

**Ví dụ**: Nhập nội dung là một thẻ a link tới trang chủ của freetuts.net

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <style>\*{margin:0}body{padding:20px}</style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular-sanitize.js">  </script>          <script language="javascript">              angular.module('myapp', ['ngSanitize'])                      .controller('ExampleController', ['$scope', function($scope) {              }]);          </script>      </head>      <body ng-app="myapp">          <div ng-controller="ExampleController">              <input type="text" ng-model="myHTML"/>              <p ng-bind-html="myHTML"></p>          </div>      </body>  </html> |

**Giao diện**:

ng-bind-html trong angularjs

Khi các bạn nhập nội dung trong thẻ INPUT là HTML thì bên dưới sẽ hiển thị dạng HTML cho bạn.

**3. Directive ng-bind-template trong AngularJS**

Nhắc đến template có lẽ bạn cũng hình dung đến những template engine mà ta thường hay sử dụng rồi nhỉ :D, đừng hiểu nhầm nó, trong này tuy chức năng tương tự nhưng hình thức load không giống nhau. Directive **ngBindTemplate** sẽ xác định nội dung cần replace thông qua cú pháp biểu thức trong angular {{tenkey}}, không giống như ngBind, ngBindTemplate có thể chứa nhiều biểu thức trong nó, ví dụ như ng-bind-template="{{header}}{{foooter}}". Ng-bind-template này rất cần thiết với các thẻ mà nội dung bên trong nó không thẻ chứa các thẻ HTML như (*title, option, meta*).

Khi ứng dụng chạy thì Directive này được thực thi luôn nên cấp độ thực thi của nó = 0 (đầu tiên).

**Ví dụ**:

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive</title>          <style>\*{margin:0}body{padding:20px}</style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script>              angular.module('myapp', [])                      .controller('ExampleController', ['$scope', function($scope) {              }]);          </script>      </head>      <body ng-app="myapp">          <div ng-controller="ExampleController">              Name <input type="text" ng-model="name"><br>              Website: <input type="text" ng-model="website"><br>              <pre ng-bind-template="{{name}} {{website}}!"></pre>          </div>      </body>  </html> |

Các bạn để ý từng dòng code [controller](http://freetuts.net/tim-hieu-controller-trong-angularjs-202.html) ở trên nhé, sẽ có sự khác biệt so với **ngBind** đấy.

**4. Lời kết**

Cũng không có gì đặc biệt cho bài này, vì vậy nó chỉ mang tính chất tham khảo giúp bạn dễ dàng tìm kiếm với từ khóa Directive ng-model trong angularJS mà thôi. Nội dung trọng tâm của nó là so sánh sự khác biệt của 3 Directives là ngBind, ngBindHtml và ngBindTemplate

**Bài 09: Directive Form trong AngularJS**

Như ta biết FORM dùng để thao tác lấy thông tin của người dùng, nghĩa là sẽ hiển thị một form lấy thông tin trên website, người dùng nhập dữ liệu vào và submit form. Lúc này Server sẽ nhận các thông tin đó và xử lý. Bởi vậy trong các ứng dụng web FORM đóng vai trò rất quan trọng, nên trong AngularJS cũng không thể thiếu đối tượng này.

**1. Tìm hiểu Directive Form trong AngularJS**

**Directive Form trong AngularJS** có thể lồng lẫn nhau theo nguyên tắc con làm cha chịu, nghĩa là khi tất cả các ràng buộc bên trong OK thì FORM mới có thể submit được. Ví dụ như bạn làm một form đăng ký thì khi tất cả các thông tin người dùng nhập vào hợp lệ thì lúc này FORM mới có giá trị là hợp lệ, còn nếu chỉ tồn tại duy nhất một lỗi nhỏ nao đó thì FORM lúc này không phải là FORM hợp lệ.

Chúng ta đã được tìm hiểu một số class được thêm vào để thể hiện trạng thái của các thẻ Input thì trong Form cũng tương tự, Angular sẽ tự động thêm những Class tùy vào trạng thái dưới đây :

* ng-valid nếu Form hợp lệ.
* ng-invalid nếu Form không hợp lệ.
* ng-pristine nếu form chưa có thao tác nào (chưa thay đổi nội dung trong FORM).
* ng-dirty nếu Form đã có thao tác (có thay đổi nội dung trong FORM).
* ng-submitted nếu Form đã submit.

**2. Thao tác Submit Form trong AngularJS**

Để submit một Form trong Angular thì thông thường chúng ta có hai cách đó là:

* Sử dụng [**Directive**](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html)**ngSubmit** nằm trong thẻ Form (nghiên cưu sau)
* Sử dụng **Directive ngClick** nằm trong thẻ input có type là submit đầu tiên (nghiên cứu sau)

Lưu ý với các bạn rằng:

* Nếu Form chỉ có một thẻ Input duy nhất thì thao tác submit khi chúng ta nhấn Enter vào thẻ input đó
* Nếu Form có hai thẻ Input trở lên và không có button submit thì (input[type="submit"]) thì khi chúng ta Enter thì Form sẽ không submit
* Nếu Form có một hoặc nhiều Fields và có hơn 1 input submit thì khi chúng ta Enter vào một field bất kỳ  thì sự kiện Click vào Input Submit đầu tiên được kích hoạt, lúc này Directive ngSubmit được kích hoạt nên Form sẽ được Submit

**3. Chỉnh sửa CSS các class của Form**

Để làm ví dụ này thì trước tiên chúng ta sẽ định nghĩa lại CSS cho các class mà AngularJS tự động thêm vào để dễ nhận diện hơn, nội dung CSS như sau:

|  |
| --- |
| .my-form {    transition:0.5s linear all;    background: white;  }  .my-form.ng-invalid {    background: red;    color:white;  } |

Như vậy trong nội dung trên mình đã định nghĩa style cho class ng-invalid là background màu đỏ.

**4. Ví dụ Directive Form trong AngularJS**

Bây giờ chúng ta sẽ thực hành một ví dụ và xem firebug nhé (Không có sử dụng [ng-model](http://freetuts.net/directive-ng-model-trong-angularjs-206.html)). Các bạn tạo một file bất kỳ với nội dung như sau:

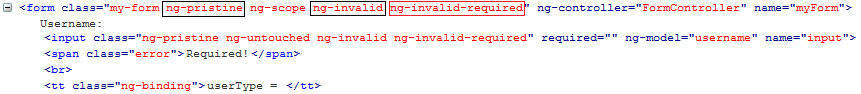
|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive Form</title>          <style>              \*{margin:0}body{padding:20px}              .my-form {                  -webkit-transition:all linear 0.5s;                  transition:all linear 0.5s;                  background: transparent;              }              .my-form.ng-invalid {                  background: red;              }          </style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script>                  angular.module('myapp', [])                      .controller('FormController', ['$scope', function($scope) {                    }]);          </script>      </head>      <body ng-app="myapp">          <form name="myForm" ng-controller="FormController" class="my-form">              Username: <input name="input" ng-model="username" required>              <span class="error">Required!</span><br>              <tt>userType = {{username}}</tt>          </form>      </body>  </html> |

**Gao diện**:

ng-form trong angularjs.

Luc mới chạy lên vì chương trình yêu cầu username không được để trống  nên nó bị báo lỗi. Bây giờ bạn thử nhập thì ngay lập tức background này biến mất. Tại sao? Lý do rất đơn giản là tại vì mình có định nghĩa CSS cho class ng-invalid là backgroud màu đỏ mà form này chưa đúng nên nó có add thêm class ng-invalid nên đẫn đến background bị màu đỏ.

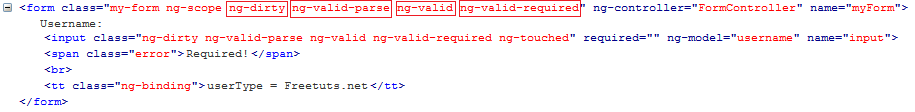
Bật Firebug lên các bạn sẽ thấy hình như sau:



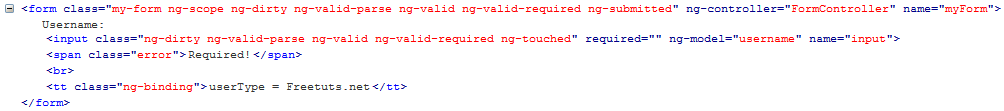
Các bạn thấy những class nằm trong Form mà tôi kẻ ô cho nó là những class sau khi chạy sẽ được tự thêm vào, **background màu đỏ**là do class ng-invalid gây ra đấy. Bây giờ ta nhập thông tin vào ô input và xem kết quả sẽ như thế nào nhé.

ng-form-trong-angular-js-2

Như vậy là background là mất, ta check Firebug và xem kết quả nhé.



Có gì khác biệt không gì? Có đấy, nó xóa đi một số class và thêm một số class giống như phần số 1 (**Tìm Hiểu Directive Form Trong AngularJS**). Bây giờ ta thử nhấn Enter và xem Firebug có gì thay đổi không nhé.



Nhìn kỹ xem nào, có một class tên là ng-submitted được thêm vào.

**5. Các các kiểm tra giá trị của form**

Mình gọi là giá trị của Form vì không biết gọi nó tên là gì nữa hi hi, nhưng chung quy lại thì trong phần này mình sẽ tìm hiểu các đối tượng liên quan đến Form đó là:

* Xem danh sách lỗi của Form và kiểm tra Form có Valid hay không
* Kiểm tra một Input nào đó trong Form có valid hay không và cách lấy danh   sách lỗi

Để rõ hơn ta bắt đầu làm một ví dụ nhé, trong ví dụ này chúng ta không có sử dụng [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) và [controller](http://freetuts.net/tim-hieu-controller-trong-angularjs-202.html) không có khai báo gì nhé các bạn..

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive Form</title>          <style>              .my-form {                  -webkit-transition:all linear 0.5s;                  transition:all linear 0.5s;                  background: transparent;              }              .my-form.ng-invalid {                  background: red;              }          </style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script>              angular.module('myapp', [])                  .controller('FormController', ['$scope', function($scope) {                  }]);          </script>      </head>      <body ng-app="myapp">          <form name="myForm" ng-controller="FormController" class="my-form">              Thông báo: <input name="input" ng-model="userType" required><br/>              <tt>myForm.input.$valid = {{myForm.input.$valid}}</tt><br>              <tt>myForm.input.$error = {{myForm.input.$error}}</tt><br>              <tt>myForm.$valid = {{myForm.$valid}}</tt>          </form>      </body>  </html> |

Chạy lên bạn sẽ thấy background màu đỏ là do lỗi chưa nhập dữ liệu, lúc này:

* **myForm.input.$valid = false** => input trong form chưa nhập dữ liệu nên có gias trị False
* **myForm.input.$error = {"required":true}** => input trong form chưa nhập dữ liệu nên **validate required** sẽ có giá trị là true (true nghĩa là à có lỗi ), nếu trong ứng dụng bạn có thêm các validate khác thì nó sẽ thêm key đó vào trong object này.
* **myForm.$valid = false** => vì input myForm.input.$valid = false  nên form sẽ mang giá trị false

Bây giờ bạn nhập dữ liệu vào, lúc này:

* **myForm.input.$valid = true** => input này thỏa mọi điều kiện validate
* **myForm.input.$error = {}** => input này không có lỗi nào
* **myForm.$valid = true**=> vì input không có lỗi nên suy ra Form cũng không có lỗi

**6. Lời kết**

Mình không biết viết gì thêm cho phần Form nà nữa, tạm thời các bạn biết tới đây là OK rồi, sau này chúng ta làm ví dụ sẽ có thêm nhiều cái để nói hơn về Directive Form này

**Bài 10: Directive ng-form trong AngularJS**

Ở bài trước chúng ta đã tìm hiểu [Form](http://freetuts.net/directive-form-trong-angularjs-209.html) rồi nhưng chúng ta chưa xử lý sự kiện submit của Form là tại vì đê rsubmit form thì cần có Directive ng-form nên mình dành một bài riêng viết về chủ đề này luôn. Thời lượng bài này có lẽ không dài như các bài trước nên rất dễ để tìm hiểu nó.

**1. Directive ng-form trong AngularJS**

ng-form hay còn gọi là ngForm trong AngularJS có nhiệm vụ liên kết dữ liệu khi Form có giá trị là đúng (Valid). Trong Javascript bình thường thì để xử lý sự kiện submit ta sẽ dùng sụ kiện có sẵn của Form đó là onsubmit="function", trong AngularJS cũng vậy nhưng cú pháp nó hơi khác đó là ta sẽ dùng ng-submit="function()"

Như bạn biết khi làm việc với AngularJS thì mọi thứ được lưu trong đối tượng [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) nên function()bên trong ng-submit đó cũng phải là một hàm của $scope nhé các bạn.

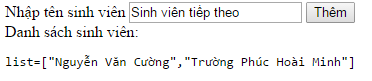
Lưu ý với bạn rằng để subimit Form thì thông thường chúng ta có hai cách chính đó là click vào input[type=submit] hoặc là nhấn Enter, nhưng không phải lúc nào cũng đúng, để rõ hơn thì bạn quay lại bài [Directie Form trong AngularjS](http://freetuts.net/directive-form-trong-angularjs-209.html) nhé. Và đương nhiên là Form không có lỗi mới submit được.

**2. Ví dụ sử dụng Directive ng-form trong AngularJS**

Bây giờ chúng ta sẽ làm một ví dụ nho nhỏ đó là xây dựng chương trình nhập danh sách sinh viên và lưu vào trong một mảng rồi in ra ngoài trình duyệt. Các bạn tạo một file với nội dung như sau:

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-Form</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script>              angular.module('myapp', [])                      .controller('ExampleController', ['$scope', function($scope) {                      // Khởi tạo giá trị ban đầu cho list = [];                      $scope.list = [];                        // Khi submit form thì chạy hàm này                      $scope.submit = function()                      {                          // nếu người dùng có nhập nội dung thì lưu nó vào list                          if ($scope.text)                          {                              $scope.list.push(this.text);                                // đồng thời xóa dữ liệu trong thẻ input                              $scope.text = '';                          }                      };                  }]);          </script>      </head>      <body ng-app="myapp">          <form ng-submit="submit()" ng-controller="ExampleController">              Nhập tên sinh viên              <input type="text" ng-model="text" name="text" />              <input type="submit" id="submit" value="Thêm" /> <br/>              Danh sách sinh viên: <pre>list={{list}}</pre>          </form>      </body>  </html> |

**Giao diện**:



Các bạn nhập nội dung và nhấn Enter hoặc click vào button Thêm thì lập tức dữ liệu được lưu vào danh sách list, lý do tại sao thì các bạn có thể mò trong đoạn javascript [controller](http://freetuts.net/tim-hieu-controller-trong-angularjs-202.html) nhé.

Lứu ý với các bạn rằng các key của $scope tương đương với các key mình định nghĩa trong View nhé ($scope.text  <=> ng-model="text"), tương tự cho các key khác.

**Tại sao submit form mà không reload trang**

Có lẽ bạn cũng thắc mắc vấn đề này nhỉ :D mình xin nhắc lại là khi làm việc với AngularJS thì bạn sẽ thao tác với API rất nhiều, chính vì thế trong sự kiện submit form kia thay vì reload trang gửi lên server  thì bạn sẽ gửi một request bằng ajax chẳng hạn.

**3. Lời kết**

Bài học chỉ tới đây thôi nhưng rất thú vị, giải thích rõ ràng cho [Directive](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html) Form trong AngularJS rồi, hy vọng các bạn hiểu được bài

**Bài 11: Directive input trong AngularJS**

Nói về input thì có nhiều loại lắm, ví dụ như checkbox, textbox, select, radio, textarea, .. vậy phải chăng hôm nay chúng ta đang tìm hiểu món này? Không phải nhé các bạn, hôm nay chúng ta chỉ tìm hiểu chung chung thôi không thuộc thể loại trên. Nhưng vì là input không khai báo type nên mặc định trình duyệt hiểu đó là textbox đấy :D.

**1. Tìm hiểu Directive input trong AngularJS**

Khi chúng ta sử dụng các thẻ HTML input kết hợp với **ng-model** thì no sẽ cung cấp các liên kết giữa model với nơi được binding, và nó sẽ sinh ra một key trong [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) để lưu trữ giá trị này nên thông thường bạn có thể dụng [ng-bind](http://freetuts.net/directives-ng-bind-ng-bind-html-ng-bind-template-trong-angularjs-208.html) hoặc là biểu thức {{key}} đều được.

Lưu ý: Input[type="file"] không support trong Directive [ng-model](http://freetuts.net/directive-ng-model-trong-angularjs-206.html).

**2. Danh sách các Directives thường đi kèm với input**

Sau đây là danh sách các Directives thường hay sử dụng chung với input, mỗi [directive](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html) sẽ có các cách sử dụng khác nhau và chúng ta sẽ nghiên cứu từng phần ở các bài sau nhé

| **Tên** | **Giá trị** | **Chi tiết** |
| --- | --- | --- |
| **ngModel** | String | Khao bái key lưu vào $scope và sẽ được liên kết tới biểu thức tương ứng ({{bieuthuc}}) |
| **name** | String | Tên của input |
| **required** | String | Thiết lập lỗi khi người dùng không nhập liệu |
| **ngRequired** | Boolean | Thiết lập thuộc tính required nếu giá trị truyền vào là true |
| **ngMinlength** | Number | Thiết lập chiều dài tối thiểu của giá trị nhập vào |
| **ngMaxlength** | Number | Thiết lập chiều dài tối đa của giá trị nhập vào |
| **ngPattern** | String | Biểu thức so khớp với giá trị nhập vào |
| **ngChange** | String | Gọi đến hàm truyền vào khi có sự kiện change |
| **ngTrim** | Boolean | Nếu thiết lập true thì khi lấy dữ liệu nó sẽ được trim khoảng trắng đầu và cuối |

**3. Ví dụ sử dụng Directive input trong AngularJS**

Bây giờ chúng ta sẽ làm một ví dụ nho nhỏ đó là xây dựng chức năng đăng nhập với hai thông tin nhập vào là tên đăng nhập và mật khẩu. Các rule như sau:

* Nếu tên đăng nhập là admin và mật khẩu là 123456 thì thông báo là đăng nhập thành công, ngược lại thông báo đăng nhập thất bại.
* Với tên đăng nhập chỉ chấp nhận các ký tự chữ cái và số, bắt buộc nhập và sử dung ng-trim để xóa khoảng trắng và dài từ 3 đến 10 ký tự
* Với mật khẩu dài từ 3 đến 10 ký tự và dùng ng-trim để xóa khoảng trắng
* Nếu những quy tắc trên không thỏa thì các chữ label sẽ có màu đỏ (dùng css style cho class ng-invalid có **color:red**)

Nội dung như sau:

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-Form</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              .ng-invalid{                  color:red;              }          </style>          <script>              angular.module('myapp', [])                      .controller('ExampleController', ['$scope', function($scope) {                      // Khi submit form thì chạy hàm này                      $scope.login = function() {                          // nếu người dùng có nhập nội dung thì lưu nó vào list                          if ($scope.username == 'admin' && $scope.password == '123456'){                              alert('Đăng Nhập Thành Công');                          } else {                              alert('Đăng Nhập Thất Bại');                          }                      };                  }]);          </script>      </head>      <body ng-app="myapp">          <h1>Đăng Nhập</h1>          <form ng-controller="ExampleController" name="myForm" ng-submit="login()">              User name: <br/>              <input type="text" name="userName" ng-pattern="/^([a-zA-Z0-9]+)$/"  ng-model="username"  ng-trim="true" required ng-minlength="3" ng-maxlength="10" /> <br/>              Password: <br/>              <input type="text" name="userPass" ng-model="password" required ng-minlength="3"  ng-maxlength="10" ng-trim="true"> <br/><br/>              <input type="submit" value='Login'/>            </form>      </body>  </html> |

**Giao diện**:



Chạy lên và thưởng thức nhé các bạn, mỗi đoạn code đều đáp ứng yêu cầu của đề bài.

**3. lời kết**

Công nhận mấy bài này ngắn thật, mới đọc một chút xíu đã hết rồi :D. Trong bài này các bạn chỉ cần hiểu rằng khi làm việc với **input trong angularjs** thì nó có hỗ trợ chúng ta những directive giúp validate thông tin của form như chiều dài tối thiểu và tối đa, dựa vào những  directive này thì ta hoàn toàn kiểm soát được dữ liệu người dùng nhập vào

**Bài 12: Directive input checkbox trong AngularJS**

Checkbox dùng để lấy thông tin của người dùng ở dạng multiselect, nghĩa là người dùng có thể chọn 1 hoặc nhiều checkbox. Trong AngularJS thì checkbox chức năng cũng không khác gì mấy, mục đích cũng là lấy thông tin của người dùng mà thôi.

Để khai báo một [input](http://freetuts.net/directive-input-trong-angularjs-213.html) là checkbox thì ta sẽ sử dụng thuộc tính **type="checkbox"**, như vậy là ta đã có một checkbox rồi đấy, và thông thường khi ta thiết lập giá trị cho checkbox thì ta dùng thuộc tính **value="giá trị".**Nhưng AngularJS thì có cách khác, chi tiết mời bạn đọc phần dưới đây.

**1. Danh sách các Directive, tham số của checkbox**

| **Tên** | **Giá trị** | **Mô tả** |
| --- | --- | --- |
| [ngModel](http://freetuts.net/directive-ng-model-trong-angularjs-206.html) | String | Tên liên kết với [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) |
| name | String | Tên của checkbox |
| ngTrueValue | Biểu thức | Biểu thức được thiết lập khi checked |
| ngFalseValue | Biểu thức | Biểu thức được thiết lập khi không checked |
| ngChange | String | Hàm sẽ gọi tới khi  input checked hoặc hủy bỏ checked |

**2. Ví dụ cách sử dụng checkbox trong AngularJS**

Bây giờ chúng ta thực hiện ví dụ sau đây, tạo hai checkbox và thiết lập checkbox thứ nhất khi checked là 'Có' và khi không checked là 'KHÔNG', tương tự cho checked thứ hai.

|  |
| --- |
| <!doctype html>  <html lang="en">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-Form</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              .ng-invalid{                  color:red;              }          </style>          <script>              angular.module('myapp', [])                      .controller('ExampleController', ['$scope', function($scope) {                      $scope.game = 'KHÔNG';                      $scope.tour = 'KHÔNG';                  }]);          </script>      </head>      <body ng-app="myapp">          <H1>Sở thích bạn là gì?</H1>          <form name="myForm" ng-controller="ExampleController">              Chơi Game: <input type="checkbox" ng-model="game"                                ng-true-value="'CÓ'" ng-false-value="'KHÔNG'"/> <br/>              Du Lịch: <input type="checkbox" ng-model="tour"                              ng-true-value="'CÓ'" ng-false-value="'KHÔNG'"/> <br/>              <h4>Bạn đang chọn</h4>              <tt>Chơi Game = {{game}}</tt><br/>              <tt>Du Lịch = {{tour}}</tt><br/>          </form>      </body>  </html> |

**Giao diện**:

'

Có một lưu ý trong ví dụ này là  giá trị bên trong ng-true-value và

ng-false-value phải được bao quanh bởi dấu nháy đơn nhé.

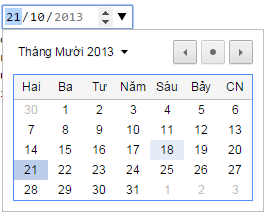
Bây giờ các bạn tự thực hành bằng cách thêm một [Directive](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html) ng-change xem có được không nhé.

**3. Lời kết**

Bài này tới đây là hết rồi, nhớ là nắm vững checkbox này rồi

**Bài 13: Directive input date trong AngularJS**

Thông thường để xây dựng một chức năng hiển thị ngày tháng năm ra ngoài thì ta sử dụng [plugin jquery](http://freetuts.net/tag/jquery-plugin) UI, nhưng điều này không còn nữa nếu ta sử dụng HTML5 cho thẻ input với type = "date", kết hợp với AngularJS nữa thì việc xử lý trên object này hoàn toàn dễ dàng. Đây là giao diện khi chúng ta khai báo type="date" trong các thẻ input.



Khi bạn sử dụng input[date] thì nó cũng có các [Directives](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html) dùng để validate thông tin như những bài trước, tuy nhiên có sự khác biệt xíu là các directive sẽ sử dụng không giống nhau do giá trị của date là ngày / tháng / năm không giống với các thẻ input thông thường. Để rõ hơn ta sẽ tìm hiểu các phần dưới đây.

**1. Directive input[type="date"] trong AngularJS**

Như ở trên mình có nói thuộc tính date này chỉ hỗ trợ trên HTML5 vì đây là một thẻ mới không hỗ trợ trong các trình duyệt cũ, chính vì vậy nếu bạn khai báo kiểu date mà trình duyệt không hỗ trợ HTML5 thì lúc này input này sẽ được chuyển sang loại mặc định (textbox).Trong trường hợp này dữ liệu nhập vào phải đúng với quy cách mà coder chọn, ví dụ như ngày\tháng\năm hoặc

ngày-tháng-năm, ...

Với loại này thì dữ liệu nhập vào luôn luôn là một đối tượng Date đúng định dạng, nếu không sẽ bị thông báo lỗi không hợp lệ, lúc này dữ liệu sẽ được tính giống như là một chuỗi rỗng.

Các múi giờ (Timezone)  được xác định thông qua Directive [ng-model-options](http://freetuts.net/directive-ng-model-options-trong-angularjs-207.html), chi tiết thế nào thì chúng ta sẽ tìm hiểu thông qua các ví dụ về **date trong angularjs** nhé.

**2. Danh sách Directive dùng trong input[type="date"]**

| **Directive** | **Giá trị** | **Chi tiết** |
| --- | --- | --- |
| [ngModel](http://freetuts.net/directive-ng-model-trong-angularjs-206.html) | String | Tên liên kết với [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) |
| name | String | Tên của input |
| min | String | Thiết lập ngày nhỏ nhất, tuân theo định dạng yyyy-MM-dd |
| max | String | Thiết lập ngày lớn nhất, tuân theo định dạng yyyy-MM-dd |
| required | String | Bắt buộc nhập cho input |
| ngRequired | String | Nếu giá trị bằng true thì Angular sẽ tự thêm thuộc tính Required |
| ngChange | String | Xay ra khi ta đổi giá trị của input |

**3. Ví dụ Directive input[type="date"] trong AngularJS**

**Lấy ngày hiện tại**

Trong ví dụ dưới đây chúng ta sẽ lấy ngày tháng năm hiện tại để gán vào thẻ input, ta sẽ sử dụng hàm new Date() để lấy ngày hiện tại và truyền vào trong [controller](http://freetuts.net/tim-hieu-controller-trong-angularjs-202.html).

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-Form</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script>              angular.module('myapp', [])                  .controller('DateController', ['$scope', function($scope) {                      $scope.value = new Date();                  }]);          </script>      </head>      <body ng-app="myapp">          <form name="myForm" ng-controller="DateController">              <input type="date" name="input" ng-model="value" />          </form>      </body>  </html> |

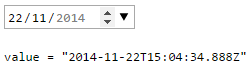
**Hiển thị giá trị ra trình duyệt**

Trong ví dụ này chúng ta xây dựng ứng dụng khi chọn ngày thì lập tức sẽ hiển thị nó ra, quy trình và các rules như sau:

* Sử dụng hai thuộc tính min và max để xác định khoảng năm có thể chọn
* Sử dụng biểu thức {{value}} để hiển thị giá trị khi nhập liệu

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-Form</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script>              angular.module('myapp', [])                  .controller('DateController', ['$scope', function($scope) {                      $scope.value = new Date();                  }]);          </script>      </head>      <body ng-app="myapp">          <form name="myForm" ng-controller="DateController">              <input type="date" name="input" ng-model="value" min="2014-01-01" max="2014-12-31" required /> <br/><br/>              <tt>value = {{value}}</tt><br/>          </form>      </body>  </html> |

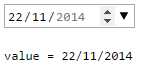
**Giao diện**:



Bạn thấy giá trị khi lấy lên nó hiển thị luôn cả timezone, muốn chỉ hiển thị ngày tháng năm thôi thì chúng ta thay đổi biểu thức hiển thị một chút xíu đó là từ {{value}} thành {{value | date : "dd/MM/yyyy"}}. Trong đó value là tên key, date là dạng dữ liệu là date, chuỗi dd/MM/yyyy là định dạng hiển thị ngày tháng năm.

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-Form</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script>              angular.module('myapp', [])                  .controller('DateController', ['$scope', function($scope) {                      $scope.value = new Date();                  }]);          </script>      </head>      <body ng-app="myapp">          <form name="myForm" ng-controller="DateController">              <input type="date" name="input" ng-model="value" min="2014-01-01" max="2014-12-31" required /> <br/><br/>              <tt>value = {{value | date : "dd/MM/yyyy"}}</tt><br/>          </form>      </body>  </html> |

**Giao diện**:



Các bạn có thấy năm nó bị mờ đi và ta không chọn được không? Lý do chính là hai giá trị min và max nằm tron năm 2014 nên ta không thể chọn năm khác được.

**4. Lời kết**

Tới đây là quá đủ cho Directive date này rồi, các bạn chỉ cần đọc qua, xem và làm các ví dụ và hiểu được là quá ok rồi. Tới đây có lẽ các bạn đã biết được định nghĩa Directive là gì và cách sử dụng nó rồi nhỉ

**Bài 14: Directive input datetime-local trong AngularJS**

Chúng ta đã tìm hiểu Directive [date](http://freetuts.net/directive-input-date-trong-angularjs-216.html) rồi mà tại sao lại xuất hiện thêm một directive tên là **datetime-local** nữa? có lý do đấy các bạn ạ, để rõ ràng thì chúng ta sẽ tìm hiểu và so sánh với kiểu date cho minh bạch nhé, mời các bạn theo dõi phần dưới đây.

**1. Directive datetime-local trong AngularJS**

Với Directive Date thì thường  chúng ta phải sử dụng định dạng HTML5 mới hỗ trợ kiểu Date, trong trường hợp Browser không hỗ trợ thì Datetime-local là một giải pháp để chúng ta sử dụng. Trong trường hợp này bắt buộc chúng ta phải khai báo đúng định dạng ngày tháng yyyy-MM-ddTHH:mm:ss. Nhưng các bạn đừng lo, trong tương lai HTML5 sẽ phổ biến và các trình duyệt lỗi thời sẽ bị loại bỏ dần thôi.

Tương tự, khi khai báo như vậy thì AngularJS sẽ chuyển nó thành một Object Date nên nếu ta khai báo không đúng định dạng thì sẽ bị lỗi, trường hợp này dữ liệu truyền vào sẽ bị hiểu là khoảng trống. Timezone mặc định sẽ sử dụng của trình duyệt, nếu bạn muốn thay đổi thì phải sử dụng Directive [ngModelOptions](http://freetuts.net/directive-ng-model-options-trong-angularjs-207.html).

**2. Danh sách các Directive hay sử dụng với Datetime-local**

Cũng chẳng có gì khác biệt so với Date, Datetime-local cũng có bấy nhiêu thông số đó thôi nhưng có sự khác đó là định dạng ngày tháng. Vì vậy các thông số min và max cũng phải khác đó là thêm giờ, phút và giây.

| **Directive** | **Giá trị** | **Chi tiết** |
| --- | --- | --- |
| [ngModel](http://freetuts.net/directive-ng-model-trong-angularjs-206.html) | String | Tên liên kết với [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) |
| name | String | Tên của input |
| min | String | Thiết lập ngày giờ nhỏ nhất, tuân theo định dạng yyyy-MM-ddTHH:mm:ss |
| max | String | Thiết lập ngày giờ lớn nhất, tuân theo định dạng yyyy-MM-ddTHH:mm:ss |
| required | String | Bắt buộc nhập cho input |
| ngRequired | String | Nếu giá trị bằng true thì Angular sẽ tự thêm thuộc tính Required |
| ngChange | String | Xay ra khi ta đổi giá trị của input |

**3. Ví dụ Directive Datetime-local trong AngularJS**

Sau đây chúng ta sẽ làm một ví dụ đơn giản để các bạn đọc sơ qua nhé, các bạn chỉ cần đọc và hiêu rồi lướt qua bài kế tiếp. Sau kh đọc xong lý thuyết thì chúng ta làm dự án, lúc này có sử dụng phần nào thì chúng ta quay lại xem là OK nhất, chứ đọc qua mà không thực hành thực tế nhiều thì cũng chả nhớ được bao nhiêu cả.

Để cho đơn giản thì mình sẽ lấy ví dụ trong Docs của Angular luôn nhé. Đề bài như sau:

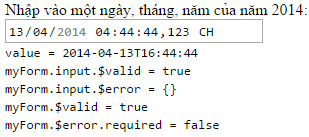
* YC1: Yêu cầu nhập một ngày trong năm 2014, Phải kiểm tra không cho phép chọn ngoài năm 2014
* YC2: Kiểm tra tính hợp lệ và bắt buộc nhập (Required), nếu không thì cho border ngoài cùng màu đỏ
* YC3: Thời gian mặc định gán vào là giờ và ngày hiện tại
* YC4: Xuất thông báo ngày nhập ra ngoài màn hình
* YC5: Xuất tình trạng của Form và input (valid , error) ra ngoài màn hinh

Quá đơn giản phải không nào, để giải quyết bài này thì chúng ta giải quyết như sau:

* YC1: Chúng ta sử dụng hai thông số là min và max để bắt người dùng nhập vào một ngày trong năm 2014
* YC2: Với tính hợp lệ thì vì đã khai báo type="datetime-local" nên angular sẽ tự kiểm tra, còn bắt buộc nhập thì chúng ta sử dụng Directive ngRequired, với border màu đỏ thì chúng ta sử dụng CSS style cho class .invalid.
* YC3: Muốn gán ngày hiện tại thì ta khai báo trong  $scope  ở [controller](http://freetuts.net/tim-hieu-controller-trong-angularjs-202.html), key sẽ tuong ứng với tên của ng-model trong input.
* YC4: Sử dụng key tương ứng là {{value}}, nhưng có một lưu ý là nếu ta chỉ xuất value như vậy thì kết quả nhìn không đẹp, chính vì vậy ta phải thêm một vài thông số vào là {{value | date: "yyyy-MM-ddTHH:mm:ss"}}
* YC5: Sử dụng cấu trúc formName.$valid, formName.$error để xem các lỗi của Form, sử dụng cấu trúc formName.input.$valid, formName.input.$error để xem các lỗi của thẻ input trong formName.

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive datetime-local</title>          <style>              form.invalid{color:red}          </style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script>              angular.module('dateExample', [])                      .controller('DateController', ['$scope', function($scope) {                      $scope.value = new Date();                  }]);          </script>      </head>      <body ng-app="dateExample">          <form name="myForm" ng-controller="DateController">              Nhập vào một ngày, tháng, năm của năm 2014: <br/>              <input type="datetime-local" name="input" ng-model="value"                     min="2014-01-01T00:00:00" max="2014-12-31T00:00:00" required /> <br/>              <tt>value = {{value | date: "yyyy-MM-ddTHH:mm:ss"}}</tt><br/>              <tt>myForm.input.$valid = {{myForm.input.$valid}}</tt><br/>              <tt>myForm.input.$error = {{myForm.input.$error}}</tt><br/>              <tt>myForm.$valid = {{myForm.$valid}}</tt><br/>              <tt>myForm.$error.required = {{!!myForm.$error.required}}</tt><br/>          </form>      </body>  </html> |

**Giao diện**:



Nếu bạn sử dụng trình duyệt có hỗ trợ HTML5 thì rất khó để chọn giá trị lỗi :D, chính vì vậy border màu đỏ không có thấy

**4. Lời kết**

Vậy là kết thúc bài rồi, chỉ bấy nhiêu thôi cũng đủ để mỏi tay rồi :D. Trong ví dụ trên mình làm từ trang chủ của AngularJS nên nếu bạn nghiên cứu được tiếng anh thì có thể lên trang chủ của nó để học thêm

**Bài 15: Directive input time trong AngularJS**

Bài này chúng ta tìm hiểu liên quan đến ngày tháng đó là **Directive Time,**mình cũng không giới thiệu nhiều về time là gì nữa mà sẽ đi thẳng vào vấn đề luôn nhé, vì tới bài này thì chắc hẳn bạn cũng đã hiểu phần nào về xử lý directive trong AngularJS rồi.

**1. Directive Time trong Angularjs**

Khi bạn sử dụng input[type="time"] thì textbox đó sẽ biến thành một ô nhập giờ, phút và giây nếu trình duyệt có hỗ trợ HTML5, nếu trình duyệt không hỗ trợ HTML5 thì chúng ta sẽ nhập vào dạng text theo định dạng (HH:mm;ss), ví dụ 12:56:30 - mười hai giờ 56 phút 30 giây.

Tương tự, Angular chỉ chấp nhận định dạng truyền vào kiểu Object Date nên nếu bạn truyền vào không phải thì sẽ lỗi nên lúc này nó sẽ hiểu giá trị của input time là rỗng.

Mặc định timezone sẽ lấy của trình duyệt nên nếu bạn muốn thay đổi timezone thì phải thông qu Directive [ng-model-options](http://freetuts.net/directive-ng-model-options-trong-angularjs-207.html).

**2. Các Directive thường dùng trong Directive Time**

Cũng chả khác là bao với hai kiểu [Date](http://freetuts.net/directive-input-date-trong-angularjs-216.html) và [Datetime-local](http://freetuts.net/directive-input-datetime-local-trong-angularjs-217.html), chỉ khác một điều là định dạng trong min, max phải là định dạng của giờ (time).

| **Directive** | **Giá trị** | **Chi tiết** |
| --- | --- | --- |
| [ngModel](http://freetuts.net/directive-ng-model-trong-angularjs-206.html) | String | Tên liên kết với [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) |
| name | String | Tên của input |
| min | String | Thiết lập giờ nhỏ nhất, tuân theo định dạng  HH:mm:ss |
| max | String | Thiết lập giờ lớn nhất, tuân theo định dạng  HH:mm:ss |
| required | String | Bắt buộc nhập cho input |
| ngRequired | String | Nếu giá trị bằng true thì Angular sẽ tự thêm thuộc tính Required |
| ngChange | String | Xay ra khi ta đổi giá trị của input |

**3 Ví dụ Directive Time trong AngularJS**

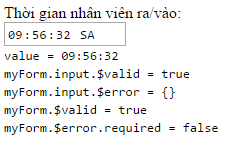
Hệ thống quản lý khách sạn yêu cầu giờ làm việc từ 8h sáng cho tới 5h chiều, chính vì vậy họ muốn hệ thống của họ phải nhập đúng định dạng là bắt đầu từ 8h và kết thúc là 5h. Mỗi nhân viên khi vào làm hay khi hết giờ đều phải ghi lại giờ vào và giờ ra trong khoảng đó. Mặc định sẽ lấy giờ hiện tại trên trình duyệt. Hãy xây dựng ứng dụng phù hợp với bài toán trên.

Thật ra thì bài này tui bịa ra đó , ta giải quyết nó nhé.

* Chúng ta sư dụng [Directives](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html)  ngRequired để yêu cầu nhập
* Sử dụng min và max để xác định khoảng thời gian 8h sáng tới 5h chiều
* Sử dụng {{keyname}} để in ra ngoài màn hình
* Sử dụng formName.$error và formName.$valid để kiểm tra trạng thái của form
* Sử dụng formName.input.$error và formName.input.$valid để kiểm tra trạng thái của input

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive datetime-local</title>          <style>              form.invalid{color:red}          </style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script>              angular.module('timeExample', [])                      .controller('DateController', ['$scope', function($scope) {                      var date = new Date();                      $scope.value = new Date(1970, 0, 1, date.getHours(), date.getMinutes(), date.getSeconds());                  }]);          </script>      </head>      <body ng-app="timeExample">          <form name="myForm" ng-controller="DateController">              Thời gian nhân viên ra/vào: <br/>              <input type="time" name="input" ng-model="value"                     min="08:00:00" max="17:00:00" required /> <br/>              <tt>value = {{value | date: "HH:mm:ss"}}</tt><br/>              <tt>myForm.input.$valid = {{myForm.input.$valid}}</tt><br/>              <tt>myForm.input.$error = {{myForm.input.$error}}</tt><br/>              <tt>myForm.$valid = {{myForm.$valid}}</tt><br/>              <tt>myForm.$error.required = {{!!myForm.$error.required}}</tt><br/>          </form>      </body>  </html> |

**Giao diện**:



Có một đoạn code có lẽ bạn sẽ thắc mắc đó là:

|  |
| --- |
| $scope.value = new Date(1970, 0, 1, date.getHours(), date.getMinutes(), date.getSeconds()); |

Ở đây vì ta chỉ lấy giờ cho nên ngày chúng ta không quan tâm lắm, chính vì vậy ta sẽ lấy thông số 1970,0,1.

**4. Lời kết**

Vẫn còn bài cuối cùng là bài input month và input week  liên quan đến ngày tháng trong angularjs

**Bài 16: Directive input month và input week trong AngularJS**

Hai đối tượng cuối cùng chúng ta tìm hiểu liên quan đến ngày tháng đó là input[type="month"] và input[type=""week], cả hai loại này đều là HTML5 nên nó cũng chẳng khác gì những Directive Date, Datetime-local và Time ở các bài trước. Trước tiên chúng ta sẽ tìm hiểu input Month trước nhé.

**1. Directive input[type="month"] trong AngularJS**

Input[type="month"] là thành phần mở rộng của HTML5 nên nó chỉ chạy trên trình duyệt hiện đại có hỗ trợ HTML5, trường hợp trình duyệt không hỗ trợ thì mặc định nó sẽ chuyển thành type = text và giá trị của nó AngularJS sẽ tự động biến thành giá trị rỗng. Trường hợp này gia trị truyền vào phải đúng với định dạng đó là yyyy-MM.

Mặc định timezone sẽ lấy ở trình duyệt nên nếu bạn muốn tha đổi timezone thì phải dùng [ngModelOptions](http://freetuts.net/directive-ng-model-options-trong-angularjs-207.html) để khai báo.

**Danh sách các Directive thường dùng với month.**

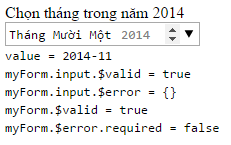
| **Directive** | **Giá trị** | **Chi tiết** |
| --- | --- | --- |
| [ngModel](http://freetuts.net/directive-ng-model-trong-angularjs-206.html) | String | Tên liên kết với [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) |
| name | String | Tên của input |
| min | String | Thiết lập giờ nhỏ nhất, tuân theo định dạng  yyyy/MM |
| max | String | Thiết lập giờ lớn nhất, tuân theo định dạng  yyy/MM |
| required | String | Bắt buộc nhập cho input |
| ngRequired | String | Nếu giá trị bằng true thì Angular sẽ tự thêm thuộc tính Required |
| ngChange | String | Xay ra khi ta đổi giá trị của input |

**Ví dụ Directive input[type="month"]**

Bây giờ ta làm một ví dụ tương tự như những bài trước đó là lấy ngày giờ hiện tại hiển thị lên input và khai báo khoảng thời gian cho phép là từ  tháng 1 năm 2014 đến tháng 12 năm 2014.

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive datetime-local</title>          <style>              form.invalid{color:red}          </style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script>              angular.module('timeExample', [])                      .controller('DateController', ['$scope', function($scope) {                      $scope.value = new Date();                  }]);          </script>      </head>      <body ng-app="timeExample">          <form name="myForm" ng-controller="DateController as dateCtrl">              Chọn tháng trong năm 2014 <br/>              <input type="month" name="input" ng-model="value"                     min="2014-01" max="2014-12" required />              <br/>              <tt>value = {{value | date: "yyyy-MM"}}</tt><br/>              <tt>myForm.input.$valid = {{myForm.input.$valid}}</tt><br/>              <tt>myForm.input.$error = {{myForm.input.$error}}</tt><br/>              <tt>myForm.$valid = {{myForm.$valid}}</tt><br/>              <tt>myForm.$error.required = {{!!myForm.$error.required}}</tt><br/>          </form>      </body>  </html> |

**Giao diện**:



**2. Directive input[type="week"] trong AngularJS**

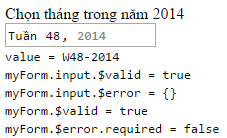
Tương tự như Month, week cũng là HTML5 nên cách xử lý cũng tương tự nên lý thuyết sẽ y chang như trên bạn nhé, , chỉ khác ở phần thông số cấu hình min và max là khác nhau.

| **Directive** | **Giá trị** | **Chi tiết** |
| --- | --- | --- |
| min | String | Thiết lập giờ nhỏ nhất, tuân theo định dạng  yyyy-W |
| max | String | Thiết lập giờ lớn nhất, tuân theo định dạng  yyyy-W |

**Ví dụ**:

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive datetime-local</title>          <style>              form.invalid{color:red}          </style>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <script>              angular.module('timeExample', [])                      .controller('DateController', ['$scope', function($scope) {                      $scope.value = new Date();                  }]);          </script>      </head>      <body ng-app="timeExample">          <form name="myForm" ng-controller="DateController as dateCtrl">              Chọn tháng trong năm 2014 <br/>              <input type="week" name="input" ng-model="value"                     min="2014-W32" max="2014-W52" required />              <br/>              <tt>value = {{value | date: "Www-yyyy"}}</tt><br/>              <tt>myForm.input.$valid = {{myForm.input.$valid}}</tt><br/>              <tt>myForm.input.$error = {{myForm.input.$error}}</tt><br/>              <tt>myForm.$valid = {{myForm.$valid}}</tt><br/>              <tt>myForm.$error.required = {{!!myForm.$error.required}}</tt><br/>          </form>      </body>  </html> |

**Giao diện**:



**3. Lời kết**

Hết bài này đúng là vui thật, vì các bài liên quan đến ngày tháng này làm lui làm tới thấy chán quá. Để các bạn không chán thì bài tiếp theo chúng ta sẽ tìm hiểu hai Directive khá hấp dẫn đó là ngShow và ngHide

# Bài 17: Directive ng-show và ng-hide trong AngularJS

Bài này chúng ta tìm hiểu hai đối tượng Directive khá là hay và rất giống với các hàm hide() và show() trong [jquery](http://freetuts.net/tag/jquery-toan-tap) đó là **ngHide và ngShow trong AngularJS**. Vì cả hai có mối liên hệ với nhau nên mình gộp chung một bài luôn và làm ví dụ chung cho dẽ hiểu và bài sẽ gọn hơn.

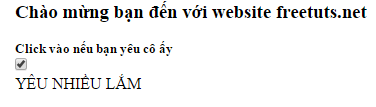
## 1. Directive ng-show trong angularjs

Show trong tiếng anh có nghĩa là 'Hiển Thị', thì ng-show cũng vậy thôi, tiền tố ng là của Angular và Show có nghĩa là hiển thị ra. Và ý nghĩa chung của Directive này là muốn xác định nó sẽ hiển thị khi nào? Rất đơn giản, khi biểu thức mà ta truyền vào nếu true thì sẽ hiển thị và ngược lại nó sẽ bị ẩn đi.

Để rõ hơn ta sẽ tìm hiểu một ví dụ dưới đây và bắt đầu soi mói nó.

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive datetime-local</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:50px}          </style>          <script>              angular.module('ngShowHide', [])                      .controller('myController', ['$scope', function($scope) {                    }]);          </script>      </head>      <body ng-app="ngShowHide">          <div ng-controller="myController">              <h3>Chào mừng bạn đến với website freetuts.net</h3> <br/>              <h5>Click vào nếu bạn yêu cô ấy</h5>              <input type="checkbox" ng-model="o\_checkbox" ><br/>              <div>                  <div ng-show="o\_checkbox">                      YÊU NHIỀU LẮM                  </div>              </div>          </div>      </body>  </html> |

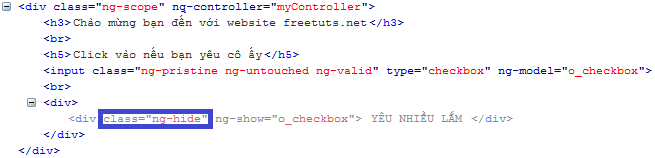
**Giao diện**:



Khi ta click vào ô checkbox thì chữ 'YÊU NHIỀU LẮM' sẽ hiển thị ra, lý do đơn giản là vì ta khai báo ng-show="o\_checkbox" mà o\_checkboxlà [model](http://freetuts.net/directive-ng-model-options-trong-angularjs-207.html) của checkbox nên khi ta check vào giá trị sẽ trả về TRUE => hiển thị.

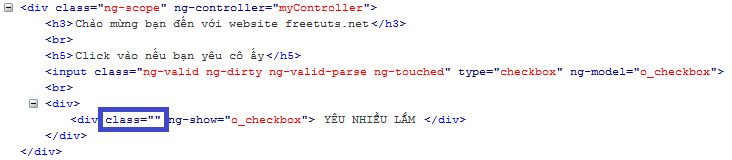
Bây giờ ta tìm hiểu lý do tại sao nó lại ẩn và hiển thị nhé, như ta biết muốn ẩn cái gì đó thì ta dùng CSS để style display:none, Angular cũng không có gì cao siêu đâu, nó cũng làm như vậy thôi. Không tin bạn bật Firebug lên và xem nhé.

**Hình Firebug khi chưa checked checkbox**



Trong hình vùng màu xanh chính là class ng-hide, khi một thẻ nào đó mà có class này thì nó sẽ bị ẩn đi. Bây giờ ta xem firebug.

**Hình Firebug khi checked checkbox**



Oh yeah! class ng-hide đã biến mất và div đó đã hiển thị ra.

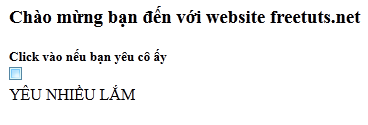
Vậy túm lại nếu một thẻ nào có class ng-hide thì sẽ bị ẩn đi và angular đã thêm class này nếu biểu thức truyền vào directive ng-show là đúng.

## 2. Directive ng-hide trong angularjs

Tương tự như ng-show, nếu biểu thức (giá trị) truyền vào ng-hide mang giá trị TRUE thì nó sẽ ẩn, còn FALSE thì nó sẽ hiển thị ra. Xem ví dụ dưới đây.

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive datetime-local</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:50px}          </style>          <script>              angular.module('ngShowHide', [])                      .controller('myController', ['$scope', function($scope) {                    }]);          </script>      </head>      <body ng-app="ngShowHide">          <div ng-controller="myController">              <h3>Chào mừng bạn đến với website freetuts.net</h3> <br/>              <h5>Click vào nếu bạn yêu cô ấy</h5>              <input type="checkbox" ng-model="o\_checkbox" ><br/>              <div>                  <div ng-hide="o\_checkbox">                      YÊU NHIỀU LẮM                  </div>              </div>          </div>      </body>  </html> |

**Giao diện**:



Khác với ví dụ ở ng-show là khi ta khi ta không click nó sẽ hiển thị, tại vì ng-show và ng-hide mang hai tính chất đối ngược nhau.

Bây giờ bạn check Firebug và xem kết quả nhé, bản chất nó vẫn dùng class ng-hide để xử lý thôi. Tôi sẽ không chụp hình và soi nữa vì mất thời gian quá :D.

## 3. Ví dụ sử dụng ng-show và ng-hide trong angularjs

Bây giờ ta làm một ví dụ tổng hợp kết hợp hay [Directive](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html)ng-hide và ng-show để xử lý.

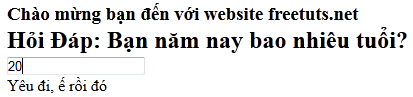
**Đề bài:** Viết chương trình kiểm tra tuổi của một người, nếu người đó từ 20 tuổi trở lên thì thông báo đủ tuổi yêu rồi, còn dưới 20 tuổi thì thông báo chưa đủ tuổi.

Để giải bài này ta sẽ làm như sau:

* Tạo một ô textbox để nhập thông tin
* Tạo một div để hiển thị thông báo đủ tuổi
* Tạo một div để hiển thị thông báo chưa đủ tuổi
* Viết một hàm trong [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) kiểm tra tuổi của ô textbox, nếu lớn hơn hoặc bằng 20 thì return true, ngược lại return false.
* Gán 2 **Directive ng-show và ng-hide** vào hai div tạo ở trên.

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40 | <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive datetime-local</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:50px}          </style>          <script>              angular.module('ngShowHide', [])                      .controller('myController', ['$scope', function($scope) {                      $scope.your\_age = 20;                      $scope.checkAge = function()                      {                          if ($scope.your\_age >= 20){                              return true;                          }                          return false;                      };                  }]);          </script>      </head>      <body ng-app="ngShowHide">          <div ng-controller="myController">              <h3>Chào mừng bạn đến với website freetuts.net</h3>              <h2>Hỏi Đáp: Bạn năm nay bao nhiêu tuổi?</h2>              <input type="text" ng-model="your\_age" ><br/>              <div>                  <div ng-hide="checkAge()">                      Bạn chưa đủ tuổi để yêu                  </div>                  <div ng-show="checkAge()">                      Yêu đi, ế rồi đó                  </div>              </div>          </div>      </body>  </html> |

**Giao diện**:



Quá đơn giản phải không nào? bạn chú ý ở hai **Directive ng-how ng-hide** mình truyền vào là hàm checkAge() ở trong [controlle](http://freetuts.net/tim-hieu-controller-trong-angularjs-202.html)r  nhé.

## 4. Lời kết

Thực tế ta áp dụng hai đối tượng này và CSS3 để tạo hiệu ứng nữa thì đẹp, nhưng vì chúng ta đang tìm hiểu từng đối tượng và bản chất của nó cho nên mình không làm những ví dụ quá cao siêu dẫ đến khó hiểu

**Bài 18: Directive ng-repeat trong AngularJS**

Repeat trong tiếng anh dịch ra có nghĩa là lặp lại, mà nhắc đến lặp thì hẳn các bạn cũng nhớ đề các vòng lặp trong PHP như  [vòng lặp for](http://freetuts.net/vong-lap-for-trong-php-7.html" \o "vòng lặp for), [vòng lặp for](http://freetuts.net/bai-04-vong-lap-trong-python-685.html)each hay while và do while. Trong javascript cũng có vòng lặp for, while và do while. Vậy AngularJS có không? Bản chất nó cũng là javascript nên Javascript có gì thì nó có nấy nhé các bạn.

Nhưng trong AngularJS thì có một cách lặp dữ liệu khác thông qua đối tượng **Directive ng-repeat**, với một cú pháp đơn giản là ta có thể lặp được rồi và nhìn sẽ sáng sủa hơn những cú pháp kia.

**1. Directive ng-repeat trong AngularJS**

Directive ngRepeat sẽ lặp một khối HTML nào đó, nơi mà ta khai báo repeat. Mỗi khối (item) như vậy sẽ có một [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) riêng tại mỗi lần lặp. Tại mỗi lần lặp ta có thể xem đang lặp phần tử thứ mấy, có phải là phần thứ đầu, giữa và cuối hay không? và có phải là phần tử lẻ và phần tử chẵn hay không. Các bạn xem bảng dưới đây nhé.

| **Tên biến** | **Giá trị** | **Chi tiết** |
| --- | --- | --- |
| $index | Number | Xác định phần tử đang lặp là thứ mấy, bắt đầu từ 0 cho tới n-1 |
| $first | Boolean | Trả về true nếu phần tử đang lặp là phần tử đầu tiên |
| $middle | Boolean | Trả về true nếu phần tử đang lặp nằm giữa phần tử đầu tiên và cuối cùng |
| $last | Boolean | Trả về true nếu phần tử đang lặp là phần tử cuối cùng |
| $even | Boolean | Trả về true nếu $index là số chẵn |
| $dd | Boolean | Trả về true nếu $index là số lẻ |

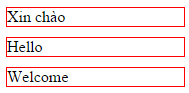
Angular sẽ lặp object hoặc mảng và mỗi lần lặp sẽ là một phần tử của mảng đó. Như bạn biết trong Javascript ta có thể tạo mảng bằng cú pháp  dùng  dấu [] và bên trong là danh sách các phần tử của mảng. Còn đối với Object thì ta sử dụng cặp dấu {}. Vậy tại sao tôi nhắc đến vấn đề này? Đó là vì cấu trúc dữ liệu lặp trong AngularJS sẽ tuân theo cú pháp này. Ví dụ:

|  |
| --- |
| [      {name:'Cương', age:25},      {name:'Tình', age:25},      {name:'Trang', age:25},      {name:'Phong', age:25}  ] |

Để rõ hơn ta sẽ làm một ví dụ nho nhỏ nhé, các bạn tạo một file mới với nội dung như sau:

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-repeat</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:50px}          </style>          <script>              angular.module('ngShowHide', [])                      .controller('myController', ['$scope', function($scope) {                      $scope.items = ["Xin chào", 'Hello', "Welcome"];              }]);          </script>      </head>      <body ng-app="ngShowHide">          <div ng-controller="myController">              <div ng-repeat="item in items" style="border:solid 1px red; margin: 10px">                  {{item}}              </div>          </div>      </body>  </html> |

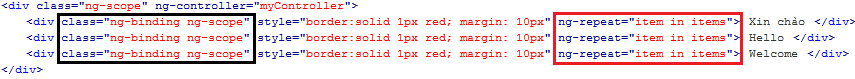
**Giao diện**:



Các bạn thấy, trong [controller](http://freetuts.net/tim-hieu-controller-trong-angularjs-202.html)  mình khai báo giá trị của items gồm 3 phần tử, như vậy trong view mình chỉ cần khai báo như sau là nó sẽ tự động lặp:

|  |
| --- |
| <div ng-repeat="item in items" style="border:solid 1px red; margin: 10px">      {{item}}  </div> |

Bây giờ bật Firebug và soi mói nhé.



Các bạn thấy nó sinh ra ba thẻ div giống y chang nhau.

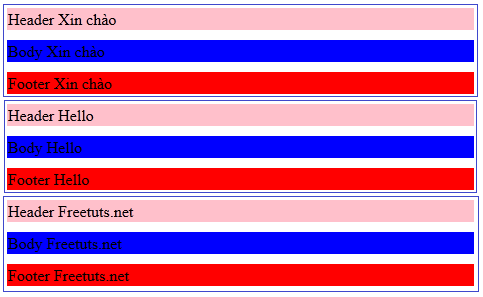
**2. ng-repeat-start và ng-repeat-end trong AngularJS**

Thay vì lặp như trên thì AngularJS có cung cấp cho chúng ta một cách lặp khác đó là sử dụng hai thuộc tính ng-repeat-start và ng-repeat-end bằng cách khai báo ở nơi ta muốn lặp và nơi dừng lặp.

**Ví dụ**:

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-repeat</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:50px}              footer, div.body,header{margin: 10px; }              footer{border:solid 1px red; background: red}              div.body{border:solid 1px blue; background: blue}              header{border:solid 1px pink; background: pink}          </style>          <script>              angular.module('ngShowHide', [])                      .controller('myController', ['$scope', function($scope) {                      $scope.items = ["Xin chào", 'Hello', "Freetuts.net"];              }]);          </script>      </head>      <body ng-app="ngShowHide">          <div ng-controller="myController">              <header ng-repeat-start="item in items">                  Header {{ item }}              </header>              <div class="body">                  Body {{ item }}              </div>              <footer ng-repeat-end>                  Footer {{ item }}              </footer>          </div>      </body>  </html> |

**Giao diện**:



Chói mắt quá :D, các bạn để ý nhé, trong đoạn mã HTML mình khai báo thẻ <header ng-repeat-start="item in items"> và thẻ kết thúc là <footer ng-repeat-end>. Như vậy mỗi lần lặp sẽ lặp từ thẻ bắt đầu cho tới thẻ kết thúc. Bây giờ các bạn hãy tự bật Firebug lên để xem cấu trúc có gì khác lạ không nhé.

**3. Ví dụ lặp ng-repeat trong AngularjS**

Bây giờ ta sẽ tìm hiểu một số cách lặp dữ liệu bằng **ng-repeat trong AngularJS** nhé.

**variable in expression**

Cách lặp này thường dùng trong trường hợp dữ liệu là một mảng đơn giản. Ví dụ:

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-repeat</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}          </style>          <script>              angular.module('ngShowHide', [])                      .controller('myController', ['$scope', function($scope) {                      $scope.items = ["Freetuts.net", 'TheHalfHeart'];              }]);          </script>      </head>      <body ng-app="ngShowHide">          <div ng-controller="myController">              <div ng-repeat="item in items">                  {{item}}              </div>          </div>      </body>  </html> |

**(key, value) in expression**

Trường hợp này nếu dữ liệu là một object trong javascript. Ví dụ

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-repeat</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}          </style>          <script>              angular.module('ngShowHide', [])                      .controller('myController', ['$scope', function($scope) {                      $scope.items = {                          name:"Văn Cường",                          website:"freetuts.net"                      };              }]);          </script>      </head>      <body ng-app="ngShowHide">          <div ng-controller="myController">              <div ng-repeat="(key,value) in items">                  {{key}} - {{value}}              </div>          </div>      </body>  </html> |

Các bạn thấy nó lặp giống như [foreach trong php](http://freetuts.net/vong-lap-foreach-trong-php-9.html) vậy đó, có key => value

**4. Lời kết**

Vẫn còn vài cách lặp nữa nhưng mình sẽ trình bày trong bài tiếp theo vì bày đầu tiên này nếu đưa vào thì sẽ hơi rối, và cái quan trọng nhất trong bài này chính là bạn phải biết được cách sử dụng căn bản của Directive ng-repeat để bài tiếp theo có thể áp dụng mà không có gì thắc mắc.

**Bài 19: Directive ng-init trong AngularJS**

**1. Tìm hiểu directive ng-init trong angularjs**

Tất cả các ví dụ từ trước tới nay đều sử dụng [Controller](http://freetuts.net/tim-hieu-controller-trong-angularjs-202.html) để thiết lập gái trị khởi tạo ban đầu, nhưng có một cách khác là sử dụng directive ng-init để thay thế. Chúng ta sẽ khai báo giá trị khởi tạo dưới dạng chuỗi, và cách dùng có thể dùng để gán giá trị hoặc khởi tạo giá trị ban đầu.

Như bạn biết tất cả các giá trị chúng ta sử dụng đều lưu trong đối tượng $scope nên ta có thể coi [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) là một biến toàn cục, vì ngInit dùng để khai báo giá trị khởi tạo hoặc gán biến gì đó và được chương trình ứng dụng nhận biết luôn nên bản chất là ngInit can thiệp vào $scope để thay đổi giá trị.

**2. ví dụ directive trong angularjs**

Để tiện cho việc hiểu bài thì chúng ta sẽ làm một số ví dụ đơn giản nhé, trong ví dụ này chúng ta có sử dụng directive [ng-repeat](http://freetuts.net/directive-ngrepeat-trong-angularjs-221.html) để lặp dữ liệu đấy nhé.

Cho một mảng danh sách sinh viên như sau:

|  |
| --- |
| 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'}  ]; |

Hãy sử dụng ng-init để gán giá trị ban đầu và sử dụng ng-repeat để in kết quả danh sách sinh viên ra ngoài trình duyệt.

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-init Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}          </style>          <script>              /\*               \* Author: thehalfheart@gmail.com               \* Website: freetuts.net               \*/              angular.module('ngInit', [])                      .controller('myController', ['$scope', function($scope) {                  }]);          </script>      </head>      <body ng-app="ngInit" ng-author="thehalfheart@gmail.com" ng-website="freetuts.net">          <div ng-controller="myController">              <ul ng-init="                                      students = [                                          {name: 'thehalfheart@gmail.com', age: 25, gender: 'boy'},                                          {name: 'freetuts.net', 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'}                                      ]                               ">                  <li ng-repeat="student in students">                      Tên: {{student.name}}, Tuổi {{student.age}}, Giới tính: {{student.gender}}                  </li>              </ul>          </div>      </body>  </html> |

**Giao diện**:



Các bạn thấy bên trong code mình đã sử dụng ng-init để khai báo danh sách sinh viên đúng không nào. Trong thực tế thay vì sử dụng như vậy thì bạn có thể khai báo nó trong controller giống như các ví dụ ở các bài trước. Và một điều nữa là cú pháp chúng ta sử dụng trong ng-init giống như code thật nhé các bạn, nếu sai cú pháp thì sẽ không chạy đâu.

**Lời kết**

Vậy là xong bài này rồi, đối tượng Directive ng-repeat này rất là hay, nó giúp chúng ta sử dụng linh hoạt không trong việc xây dựng ứng dụng angular. Khi học tới bài này thì bạn thấy thực sự angular rất mạnh đúng không nào? Nhất là cấu trúc [Directive](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html) tuyệt vời.

**Bài 20: Directive ng-include trong AngularJS**

Nếu bạn là dân lập trình PHP thì bạn cũng biết hàm include hay hàm required trong PHP dùng để import một file PHP khác vào file hiện tại. Đây là  phía Server, vconf phía client thì như thế nào? Thông thường chúng ta sẽ sử dụng ajax hoặc là iframe để xử lý vấn đề này. Tuy nhiên nếu sử dụng angular thì bạn sẽ có một cách đơn giản hơn đó là sử dụng đối tượng [**directive**](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html)**ng-include**. Chi tiết thế nào thì chúng ta sẽ tìm hiểu nhé.

**1. Directive ng-include trong Angularjs**

Để import một file (url web) nào đó vào trong một vị trí nào đó thì tại ví trí đó chúng ta sẽ sử dụng một đoạn code sau:

|  |
| --- |
| <div ng-include=" '[http://freetuts.net](http://freetuts.net/)' "></div> |

Trong đó đường dẫn **freetuts.net** chính là đường dẫn trỏ đến file cần import. Lưu ý với bạn là phải có dấu nháy đơn bao quanh URL lại nhé, nếu không sẽ bị báo lỗi đấy.

**Ví dụ**: Cho file template-a.html, template-b.html và index.html. Hãy sử dụng ng-include để import hai file template vào file index.

Để giải bài này ta sẽ làm các bước như sau.

**File template-a.html**

|  |
| --- |
| <h2 style="border:solid 1px red; margin: 10px;">      CHÀO MỪNG CÁC BẠN ĐẾN VỚI FREETUTS.NET <BR/>      ĐÂY LÀ TEMPLATE-A  </h2><br/><br/><br/> |

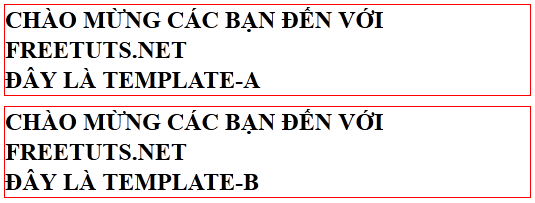
**File template-b.html**

|  |
| --- |
| <h2 style="border:solid 1px red; margin: 10px;">      CHÀO MỪNG CÁC BẠN ĐẾN VỚI FREETUTS.NET <BR/>      ĐÂY LÀ TEMPLATE-B  </h2> |

**File index.html**

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-include Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}          </style>          <script>              angular.module('ngInit', []).controller('myController',  ['$scope', function($scope) {              }]); |

Hai template chỉ có nhiệm vụ là hiển thị một đoạn text thôi, còn file index.html thì có nhiệm vụ import hai file kia vào. Chạy lên giao diện sẽ như sau:



**2. Các file được include có nhận được biến của $sope hay không ?**

Đây là câu hỏi mà chắc chắn bạn phải hỏi đấy nhé, nếu bạn không hỏi thì bạn sẽ không hiểu được toàn bộ **directive ng-include** này. Nếu như bạn để ý kỹ bằng cách bật firebug lên thì bạn sẽ thấy bản chất khi AngularJS import các template vào là nó đang sử dung [ajax](http://freetuts.net/tag/ajax-toan-tap) đấy nhé, và khi nhận kết quả nó sẽ xử lý các biểu thức khai báo trong kết quả đó, nếu có khai báo mã AngularJS và sử dụng các biến trong [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) thì nó vẫn nhận bình thường.

Bây giờ để các bạn hiểu rõ hơn thì ta làm lại ví dụ trên nhé, ở cả  hai template câu chào "*Chào mừng các bạn đến với freetuts.net*" sẽ được truyền tử giá trị $scope được gán trong [controller](http://freetuts.net/tim-hieu-controller-trong-angularjs-202.html). Chi tiết ta sẽ làm như sau:

**File template-a.html**

|  |
| --- |
| <h2 style="border:solid 1px red; margin: 10px;">      {{message}} <BR/>      ĐÂY LÀ TEMPLATE-A  </h2> |

**File template-b.html**

|  |
| --- |
| <h2 style="border:solid 1px red; margin: 10px;">      {{message}} <BR/>      ĐÂY LÀ TEMPLATE-B  </h2> |

**File index.html**

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Directive ng-include Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}          </style>          <script>              angular.module('ngInit', []).controller('myController',  ['$scope', function($scope) {                      $scope.message = 'CHÀO MỪNG CÁC BẠN ĐẾN VỚI FREETUTS.NET';              }]);          </script>      </head>      <body ng-app="ngInit" ng-author="thehalfheart@gmail.com" ng-website="freetuts.net">          <div ng-controller="myController">              <div ng-include=" 'template-a.html' "></div>              <div ng-include=" 'template-b.html' "></div>          </div>      </body>  </html> |

Chạy lên kết quả vẫn giống như ví dụ trước.

**3. Lời kết**

Vẫn còn vài phần nâng cao nữa đó là **các tham số và  events** của ng-include nhưng mình sẽ không trình bày ở bài này mà sẽ trình bày nó ở một bài khác vì nếu đưa vào bài này luôn bạn sẽ loạn đấy

**Bài 21: Services trong AngularJS và cách tạo Service mới**

Có lẽ bạn từng nghe qua khái niệm Service rồi nhỉ, chẳng hạn như Service trong PHP thì nhiệm vụ của nó giống như là một API dùng để trả kết quả về cho client. Đối với AngularJS thì nó cũng không có gì khác, service chính là những hàm thực hiện một nhiệm vụ cụ thể nào đó.

**1. Service trong angularjs**

Service trong AngularJS đóng một vai trò rất quan trọng trong các ứng dụng được viết bằng AngularJS, hiện có những service có sẵn mà Angular đã cung cấp cho chúng ta, tuy nhiên chúng ta hoàn toàn có thể mở rộng thêm bằng cách **tạo một Service mới**. Sau đây là danh sách các Service có sẵn trong AngularJS:

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

Trong danh sách này mỗi service sẽ có một nhiệm vụ nhất định. Ví dụ như với Service $http có nhiệm vụ gửi, trao đổi thông tin với Server thôn qua [ajax](http://freetuts.net/tag/ajax-toan-tap).Và các bạn để ý thấy tên của Service sẽ bắt đầu bằng ký hiệu đô la $, điều này có nghĩa rằng tất cả những service có sẵn chúng ta phải có ký hiệu này đằng trước.

**2. Các tạo Service mới trong AngularJS**

Có hai cách để chúng ta định nghĩa một Service mới như sau:

* Sử dụng Factory method
* Sử dụng Service method

Bây giờ chúng ta tìm hiểu từng cách nhé.

**Tạo service mới bằng Factory method**

**Ví dụ:** Viết chương trình gồm một ô input, khi nhập nếu giá trị chia hết cho 2 thì xuất ra màn hình thông báo chia hết cho 2, ngược lại thì không thông báo gì.

**Bài giải**:

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Service trong AngularJS - Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}          </style>          <script>              // Lưu module vào biến              var module = angular.module('myServiceModule', []);              // Khai báo controller              module.controller('MyController',  ['$scope','kiem\_tra\_so\_chan', function ($scope, kiem\_tra\_so\_chan) {                  $scope.xuat\_thong\_bao = function() {                      kiem\_tra\_so\_chan($scope.number);                  };              }]);                // Tạo service kiểm tra số chẵn              module.factory('kiem\_tra\_so\_chan', function($window) {                  return function(number) {                      if (number % 2 == 0) {                          $window.alert("Số " + number + " là số chẵn");                      }                  };              });          </script>      </head>      <body ng-app="myServiceModule" ng-author="thehalfheart@gmail.com" ng-website="freetuts.net">          <div id="simple" ng-controller="MyController">          <p>Nhập vào số chẵn thì sẽ thông báo lên màn hình</p>          <input ng-model="number" ng-change="xuat\_thong\_bao()"/>        </div>      </body>  </html> |

**Kết quả:** Khi chạy lên bạn nhập vào nếu là số chẵn thì sẽ có thông báo, ngược lại sẽ không có thông báo. Như vậy hoàn toàn giống với yêu cầu đề bài, bây giờ chúng ta mổ xẻ các hàm trong nó nhé. Trong đoạn code trên bạn cần chú ý tới đoạn code sau:

|  |
| --- |
| // Tạo service kiểm tra số chẵn  module.factory('kiem\_tra\_so\_chan', function($window) {      return function(number) {          if (number % 2 == 0) {              $window.alert("Số " + number + " là số chẵn");          }      };  }); |

Đây chính là đoạn mã tạo service với tên là  kiem\_tra\_so\_chan, service này sẽ sử dụng service có sẵn $window để xử lý alert thông tin lên. Trong thân hàm tham số truyền vào của đoạn mã return là biến number, như vậy khi [controller](http://freetuts.net/tim-hieu-controller-trong-angularjs-202.html) sử dụng service này sẽ phải truyền giá trị numbervào, cách truyền chính là đoạn code này :

|  |
| --- |
| kiem\_tra\_so\_chan($scope.number); |

**Tạo Service mới bằng Service method**

**Ví dụ**: Viết chương trình giống như ví dụ ở sử dụng Factory tạo service, thay vì dùng Factory method thì dùng Service method

**Bài giải**:

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Service trong AngularJS - Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}          </style>          <script>              // Lưu module vào biến              var module = angular.module('myServiceModule', []);              // Khai báo controller              module.controller('MyController',  ['$scope','kiem\_tra\_so\_chan', function ($scope, kiem\_tra\_so\_chan) {                  $scope.xuat\_thong\_bao = function() {                      kiem\_tra\_so\_chan($scope.number);                  };              }]);              // Tạo service kiểm tra số chẵn              module.service('kiem\_tra\_so\_chan', function($window) {                  return function(number) {                      if (number % 2 == 0) {                          $window.alert("Số " + number + " là số chẵn");                      }                  };              });          </script>      </head>      <body ng-app="myServiceModule" ng-author="thehalfheart@gmail.com" ng-website="freetuts.net">          <div id="simple" ng-controller="MyController">          <p>Nhập vào số chẵn thì sẽ thông báo lên màn hình</p>          <input ng-model="number" ng-change="xuat\_thong\_bao()"/>        </div>      </body>  </html> |

Các bạn để ý kỹ thấy nội dung file không có gì thay đổi cả, sự khác nhau chỉ là thay chữ factory thành service và kết quả thì y chang nhau.

**3. Gọi service trong service**

Việc gọi một service có sẵn trong một service tạo mới thì không có gì đặc biệt, nhưng nếu hai service được tạo mới hoàn toàn thì gọi lẫn nhau có được không? Điều này hoàn toàn được nha bạn. Bây giờ chúng ta làm một ví dụ nhỏ để minh chứng cho điều này.

**Ví dụ:**Thực hiện lại ví dụ ở phần 2 nhưng chia làm hai service khác nhau, **service 1** dùng để kiểm tra một số có phải số chẵn hay không, **service 2** có nhiệm vụ validate giá trị và sử dụng **service 1** để kiểm tra. Điều này giống như bạn tạo một hàm validate, trong hàm validate sử dụng hàm kiem\_tra\_so\_chan.

**Bài giải**:

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng Service trong AngularJS - Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}          </style>          <script>              // Lưu module vào biến              var module = angular.module('myServiceModule', []);              // Khai báo controller              module.controller('MyController', ['$scope','validate', function ($scope, validate) {                  $scope.xuat\_thong\_bao = function() {                      validate($scope.number);                  };              }]);              // Tạo service kiểm tra số chẵn              module.service('validate', function($window, kiem\_tra\_so\_chan) {                  return function(number) {                      if (kiem\_tra\_so\_chan(number)) {                          $window.alert("Số " + number + " là số chẵn");                      }                  };              });                // Tạo service kiểm tra số chẵn              module.service('kiem\_tra\_so\_chan', function()              {                  return function(number)                  {                      return(number % 2 == 0);                  };              });          </script>      </head>      <body ng-app="myServiceModule" ng-author="thehalfheart@gmail.com" ng-website="freetuts.net">          <div id="simple" ng-controller="MyController">          <p>Nhập vào số chẵn thì sẽ thông báo lên màn hình</p>          <input ng-model="number" ng-change="xuat\_thong\_bao()"/>        </div>      </body>  </html> |

Chạy lên kết quả vẫn không có gì khác.

**4. Lời kết**

Bạn thấy service tuyệt vời phải không nào, các ví dụ cách sử dụng service trong angularjs này cũng không có gì khó nên hy vọng các bạn nắm được bài. Các bài  tìm hiểu Directive vẫn chưa hết nhưng tôi muốn chuyển sang service để các bạn không cảm thấy chán, đây cũng là bài đầu tiên tìm hiểu về service nên chặng đường còn rất dài nhưng đừng nản chí nhé

**Bài 22: Directive ng-class trong angularjs**

Trong bài này chúng ta sẽ tìm hiểu việc xử lý thêm hay thiết lập các class cho đối tượng HTML thông qua directive ngClass rất hay mà AngluarJS cung cấp cho chúng ta, việc dùng nó thực sự rất đơn giản mà bạn sẽ không ngờ tới, kết hợp với animation thì giống như thịt gà chấm muối tiêu.

**1. Tìm hiểu directive ng-class trong angularjs**

[Directive](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html) ng-class cho phép chúng ta thiết lập những class động trên các thẻ HTML trong ứng dụng website bằng cách dùng cơ chế **databinding** và các biểu thức để quyết định thêm hoặc không thêm class đó. Một ví dụ điển hình là khi các bạn làm một chức năng chuyển tabs bạn sẽ sử dụng nó để thiết lập class active cho tab hiện tại.

Chúng ta có ba cách sử dụng ngClass như sau:

* Nếu biểu thức truyền vào là một chuỗi string thì giá trị của nó nó có thể là một class hoặc nhiều cách nhau bởi khoảng trắng
* Nếu biểu thức truyền vào là một mảng array thì mỗi phần tử của mảng đó phải là một chuỗi chứa một hoặc nhiều class name, nếu nhiều thì cách nhau bởi khoảng trắng.
* Nếu biểu thức truyền vào là đối tượng object thì các cặp key:value như sau:
  + Sẽ thêm class vào nếu giá trị value = true

Lưu ý: Nếu giá trị trường hợp object, nếu của className là true thì className sẽ được thêm vào, ngược lại nó sẽ không được thêm vào.

Cú pháp sử dụng rất đơn giản như sau:

|  |
| --- |
| <ANY class="ng-class: ;"> ... </ANY> |

**2. Các ví dụ sử dụng directive ng-class trong angularjs**

**Trường hợp giá trị truyền vào là chuỗi**

**Ví dụ**:

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng ngClass trong AngularJS - Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}              .red{                  padding: 20px;                  background-color: red;                  color:#FFF;              }              .border{                  border:solid 1px blue;              }          </style>          <script>              angular.module('ngClassExample', [])                      .controller('ngControllerExample', function($scope){                          $scope.className = 'red border';                      });          </script>      </head>      <body ng-app="ngClassExample">          <div ng-controller="ngControllerExample">              <p ng-class="className">Học angularjs miễn phí tại freetuts.net</p>          </div>      </body>  </html> |

Trong ví dụ này mình tạo hai class như sau:

|  |
| --- |
| .red{      padding: 20px;      background-color: red;      color:#FFF;  }  .border{      border:solid 1px blue;  } |

Sau đó mình tạo một thẻ p và truyền ng-class vào như sau:

|  |
| --- |
| <p ng-class="className">Học angularjs miễn phí tại freetuts.net</p> |

Và cuối cùng trong Controller mình gán giá trị cho classNamenhư sau:

|  |
| --- |
| $scope.className = 'red border'; |

Như vậy khi chương trình chạy lên thì hai class **red, border** sẽ được thêm vào thẻ P nên giao diện sẽ như sau:



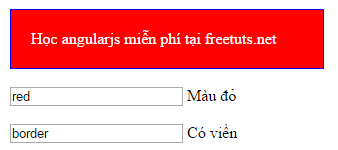
**Trường hợp giá  trị truyền vào là mảng**

**Ví dụ**:

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng ngClass trong AngularJS - Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}              .red{                  padding: 20px;                  background-color: red;                  color:#FFF;              }              .border{                  border:solid 1px blue;              }          </style>          <script>              angular.module('ngClassExample', [])                      .controller('ngControllerExample', function($scope){              });          </script>      </head>      <body ng-app="ngClassExample">          <div ng-controller="ngControllerExample">              <p ng-class="[border, red]">Học angularjs miễn phí tại freetuts.net</p> <br/>              <input type="text" ng-model="red" placeholder="Nhập vào red"/> Màu đỏ <br/><br/>              <input type="text" ng-model="border" placeholder="Nhập vào border"/> Có viền <br/>          </div>      </body>  </html> |

Trường hợp này sự khác biệt ở ở thẻ P tôi truyền vào là một mảng, nên lúc này các giá trị trong mảng sẽ truy xuất đến giá trị tương ứng trong $scope, mà hai key red và border chính là 2 input (do có ng-model) nên nếu bạn nhập vào hai ô đó tức là bạn thay đổi trong $scope. Chính vì vậy thẻ p sẽ nhận được các class tương ứng.

**Giao diện**:



**Trường hợp giá trị truyền vào là object**

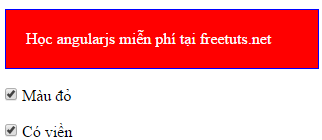
**Ví dụ**:

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng ngClass trong AngularJS - Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}              .red{                  padding: 20px;                  background-color: red;                  color:#FFF;              }              .border{                  border:solid 1px blue;              }          </style>          <script>              angular.module('ngClassExample', [])                      .controller('ngControllerExample', function($scope){                      });          </script>      </head>      <body ng-app="ngClassExample">          <div ng-controller="ngControllerExample">              <p ng-class="{red:scope\_red, border:scope\_border}">  Học angularjs miễn phí tại freetuts.net  </p><br/>              <input type="checkbox" ng-model="scope\_red"/> Màu đỏ <br/><br/>              <input type="checkbox" ng-model="scope\_border"/> Có viền <br/>          </div>      </body>  </html> |

Trong ví dụ này giá trị ng-class="{red:scope\_red, border:scope\_border}". Mà theo như phần 2 tôi trình bày thì:

* Nếu scope\_redcó giá trị là true thì class red sẽ được thêm vào
* Nếu scope\_bordercó giá trị là true thì class border sẽ được thêm vào

**Giao diện**:



**Tạo hiệu ứng animate với ng-class và CSS3**

Để tạo hiệu ứng animate thì chúng ta phải kết hợp CSS3 và ng-class để xử lý thêm class animate. Trước tiên chúng ta khởi tạo hai class như sau:

|  |
| --- |
| .base-class {      -webkit-transition:all cubic-bezier(0.250, 0.460, 0.450, 0.940) 0.5s;      transition:all cubic-bezier(0.250, 0.460, 0.450, 0.940) 0.5s;  }  .base-class.my-class {      color: red;      font-size:3em;  } |

Tiếp theo tạo thẻ P và 2 button:

|  |
| --- |
| <p class="base-class" ng-class="variable">Học angularjs miễn phí tại freetuts.net</p> <br/>  <input type="button" value="Animate" ng-click="variable = 'my-class'"/>  <input type="button" value="Not Animate" ng-click="variable = ''"/> |

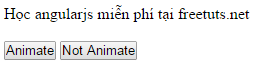
**Trong đó**:

* Thẻ P có nhiệm vụ hiển thị text và có ng-class="variable"
* Button thứ nhất có sự kiện click ng-click="variable = 'my-class'", sự kiện này sẽ gán giá trị cho variable = my-class, như vậy thẻ P sẽ lấy variable và thêm **class my-class** nên có hiệu ứng animate lớn lên
* Button thứ hai có sự kiện click ng-click="variable = ''", sự kiện này gán variable = '', như vậy thẻ P thấy variable lúc này không còn gì nên nó bỏ **class my-class**đi nên sẽ có hiệu ứng animate nhỏ xuống

**Toàn file như sau**:

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>Ví dụ sử dụng ngClass trong AngularJS - Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}              .base-class {                  -webkit-transition:all cubic-bezier(0.250, 0.460, 0.450, 0.940) 0.5s;                  transition:all cubic-bezier(0.250, 0.460, 0.450, 0.940) 0.5s;              }              .base-class.my-class {                  color: red;                  font-size:3em;              }          </style>          <script>              angular.module('ngClassExample', [])                      .controller('ngControllerExample', function($scope){                      });          </script>      </head>      <body ng-app="ngClassExample">          <div ng-controller="ngControllerExample">          <p class="base-class" ng-class="variable">Học angularjs miễn phí tại freetuts.net</p> <br/>          <input type="button" value="Animate" ng-click="variable = 'my-class'"/>          <input type="button" value="Not Animate" ng-click="variable = ''"/>          </div>      </body>  </html> |

**Giao diện chưa click**:



**Giao diện sau khi click**:



**Lời kết**

Việc sử dụng ng-class được ứng dụng rất rộng rãi trong thực tế, ví dụ các trường hợp cần active như tabs, menu thì nếu ta kết hợp ngClass để xử lý thì cực kì nhanh. Bài này tới đây là kết thúc, chúc các bạn học tốt.

**Bài 23: Directive ng-click trong angularjs**

Trong các đối tượng HTML (các thẻ HTML) có một sự kiện mà đối tượng nào cũng có đó là onclick, sự kiện này xảy ra khi chúng ta dùng chuột click vào phạm vi hiển thị của thẻ đó, nếu các thẻ chồng lên nhau thì khi chúng ta click thì cấp cao nhất sẽ được xảy ra trước. Với angular thì nó cung cấp cho chúng ta một directive tương tự đó là ng-click, cách hoạt động nó cũng không khác gì sự kiện onclick. Chi tiết chúng ta sẽ tìm hiểu qua các ví dụ dưới đây nhé.

**1. Tìm hiểu directive ng-click trong angularjs**

Sự kiện này sẽ xảy ra khi chúng ta click vào thẻ hiện tại. Cú pháp là <element ng-click="statment"></element>,  trong đó:

* ng-click là tên của [directive](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html) ngClick
* statment là những dòng lệnh sẽ chạy. Bạn có thể truyền vào hàm hoặc những đoạn code rời rạc

**2. Các ví dụ về ng-click trong angularjs**

Chúng ta sẽ xây dụng một ứng dụng đơn giản đó là thực hiện đếm số lần click vào thẻ button bằng hai cách (Gọi đến hàm và xử lý trực tiếp)

**Gọi đến hàm**

Cách này chúng ta sẽ gọi đến một hàm khai báo trong [$scope](http://freetuts.net/tim-hieu-scope-va-rootscope-trong-angularjs-204.html) ở controller.

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>NG-CLICK trong AngularJS - Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}          </style>          <script>              angular.module('ngCLickExample', [])                  .controller('ngClickController', function($scope){                      // Gán giá trị khởi đầu cho count                      $scope.count = 0;                      // Hàm tăng count lên 1 đơn vị                      $scope.TangBienCount = function(){                          $scope.count++;                      };                  });          </script>      </head>      <body ng-app="ngCLickExample" ng-controller="ngClickController">          <h1>Freetuts.net - ng-click</h1>          <input type="button" value="Click Me" ng-click="TangBienCount()" /> <br/><br/>          <span>Bạn đã click {{count}} lần</span>      </body>  </html> |

**Xử lý trực tiếp**

Cách này chúng ta xử lý trực tiếp ngay tại vị trí ng-click luôn.

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>NG-CLICK trong AngularJS - Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}          </style>          <script>              angular.module('ngCLickExample', [])                  .controller('ngClickController', function($scope){                      // Gán giá trị khởi đầu cho count                      $scope.count = 0;                  });          </script>      </head>      <body ng-app="ngCLickExample" ng-controller="ngClickController">          <h1>Freetuts.net - ng-click</h1>          <input type="button" value="Click Me" ng-click="count = count + 1" /> <br/><br/>          <span>Bạn đã click {{count}} lần</span>      </body>  </html> |

Sự khác biệt ở hai cách này là cách 1 ta gọi đến hàm khai báo trong [controller](http://freetuts.net/tim-hieu-controller-trong-angularjs-202.html), cách hai thì ta tăng biến count trực tiếp ngay ng-click luôn. Cả hai cách trên chạy lên giao diện sẽ giống nhau:



**3. Lời kết**

Quá đơn giản cho phần này, bản chất nó hoạt động giống như sự kiện onclick trong javascript thôi nên bạn rất dễ hiểu đối với directive này.

**Bài 24: Directive ng-class-odd và ng-class-even trong Angularjs**

Có một số layout hiển thị danh sách theo dạng row lẻ thì có background khác với màu chẵn, lúc này giải pháp của chúng ta là tạo hai class là odd dành cho row lẻ và evendành cho row chẵn. Nếu chúng ta dùng [vòng lặp foreach trong php](http://freetuts.net/vong-lap-foreach-trong-php-9.html) để lặp thì sẽ kiểm tra điều kiện index chẵn hay lẻ để hiển thị, nhưng chúng ta đang làm angular nên sẽ sử dụng [ng-repeat](http://freetuts.net/directive-ngrepeat-trong-angularjs-221.html) và kết hợp với $index để kiểm tra điều kiện.

Tuy nhiên làm vậy thì chúng ta đa đi sai hướng rồi vì trong AngularJS có hỗ trợ hai directive giải quyết vấn đề này là **ng-class-odd** và **ng-class-even**, chi tiết các bạn xem phần dưới đây.

**1. Directive ng-class-odd trong AngularJS**

Ta sẽ làm ví dụ hiển thị danh sách sinh viên, nếu row lẽ thì cho background màu đỏ, ta sẽ làm các bước như sau:

* Khai báo một class background màu đỏ
* Dùng ng-class-odd để xử lý

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>ng-class-odd và ng-class-even trong AngularJS - Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}              div{height: 30px; line-height: 30px; padding: 0px 20px;}              .odd{                  background: red;                  color:#FFF;              }          </style>          <script>              angular.module('Example', [])                  .controller('Controller', function($scope){                      // Danh sách sinh viên                      $scope.students = [                          {name : "Nguyễn Văn Cường"},                          {name : "Nguyễn Văn Kính"},                          {name : "Nguyễn Văn Chính"},                          {name : "Nguyễn Văn Quyền"},                          {name : "Nguyễn Văn Trường"},                          {name : "Nguyễn Văn Hùng"},                          {name : "Nguyễn Văn Hiếu"},                          {name : "Nguyễn Văn Nghĩa"}                      ];                  });          </script>      </head>      <body ng-app="Example" ng-controller="Controller">          <h1>Freetuts.net - ng-class-odd</h1>          <div ng-repeat="student in students" ng-class-odd="'odd'">              {{student.name}}          </div>      </body>  </html> |

**Giao diện**:

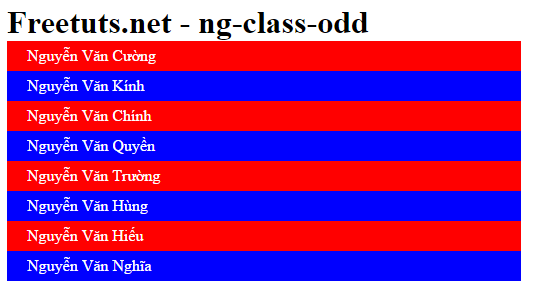


**2. Directive ng-class-even trong AngularJS**

Kết hợp với ví dụ trên là hiển thị row lẽ màu đỏ, chúng ta sử dụng ng-class-even để thêm class background màu xanh cho row chẵn. Chúng ta chỉ việc tạo thêm một class event có background màu xanh và thêm [directive](http://freetuts.net/tim-hieu-angularjs-directives-danh-sach-directives-205.html)ng-class-even vào các thẻ div ở mỗi vòng lặp là xong.

|  |
| --- |
| <!doctype html>  <html lang="vi">      <head>          <meta charset="UTF-8">          <title>ng-class-odd và ng-class-even trong AngularJS - Freetuts.net</title>          <script src="//ajax.googleapis.com/ajax/libs/angularjs/1.3.2/angular.min.js"></script>          <style>              \*{margin:0;padding:0}              body{margin:20px;}              div{height: 30px; line-height: 30px; padding: 0px 20px;}              .odd{                  background: red;                  color:#FFF;              }              .even{                  background: blue;                  color:#FFF;              }          </style>          <script>              angular.module('Example', [])                  .controller('Controller', function($scope){                      // Danh sách sinh viên                      $scope.students = [                          {name : "Nguyễn Văn Cường"},                          {name : "Nguyễn Văn Kính"},                          {name : "Nguyễn Văn Chính"},                          {name : "Nguyễn Văn Quyền"},                          {name : "Nguyễn Văn Trường"},                          {name : "Nguyễn Văn Hùng"},                          {name : "Nguyễn Văn Hiếu"},                          {name : "Nguyễn Văn Nghĩa"}                      ];                  });          </script>      </head>      <body ng-app="Example" ng-controller="Controller">          <h1>Freetuts.net - ng-class-odd</h1>          <div ng-repeat="student in students" ng-class-odd="'odd'" ng-class-even="'even'">              {{student.name}}          </div>      </body>  </html> |

**Giao diện**:



**3. Lời kết**

Sự kiết hợp hai directive này rất là hay, nếu không có nó thì chúng ta phải kiểm tra điều kiện nữa nên sẽ rất là rối.

**B. AngularJS căn bản (thực hành)**

<http://gockinhnghiem.com/category/lap-trinh/javascript-cac-thu-vien/angular-js/>

<http://dammelaptrinh.com/category/angularjs-tutorial>