AngularJS - Compiling dynamic HTML strings from database

The Situation

Nested within our Angular app is a directive called Page, backed by a controller, which contains a div with an ng-bind-html-unsafe attribute. This is assigned to a $scope var called 'pageContent'. This var gets assigned dynamically generated HTML from a database. When the user flips to the next page, a called to the DB is made, and the pageContent var is set to this new HTML, which gets rendered onscreen through ng-bind-html-unsafe. Here's the code:

Page directive

```
angular.module('myApp.directives')
    .directive('myPage', function ($compile) {

        return {
            templateUrl: 'page.html',
            restrict: 'E',
            compile: function compile(element, attrs, transclude) {
                // does nothing currently
                return {
                    pre: function preLink(scope, element, attrs, controller) {
                        // does nothing currently
                    },
                    post: function postLink(scope, element, attrs, controller) {
                        // does nothing currently
                    }
                }
            }
        };
    });
```

Page directive's template ("page.html" from the templateUrl property above)

```
<div ng-controller="PageCtrl" >
   ...
   <!-- dynamic page content written into the div below -->
   <div ng-bind-html-unsafe="pageContent" >
   ...
</div>
```

Page controller

```
angular.module('myApp')
  .controller('PageCtrl', function ($scope) {

        $scope.pageContent = '';

        $scope.$on( "receivedPageContent", function(event, args) {
            console.log( 'new page content received after DB call' );
            $scope.pageContent = args.htmlStrFromDB;
        });

});
```

That works. We see the page's HTML from the DB rendered nicely in the browser. When the user flips to the next page, we see the next page's content, and so on. So far so good.

The Problem

The problem here is that we want to have interactive content inside of a page's content. For instance, the HTML may contain a thumbnail image where, when the user clicks on it, Angular should do something awesome, such as displaying a pop-up modal window. I've placed Angular method calls (ng-click) in the HTML strings in our database, but of course Angular isn't going to recognize either method calls or directives unless it somehow parses the HTML string, recognizes them and compiles them.

In our DB

Content for Page 1:

```
<p>Here's a cool pic of a lion. <img src="lion.png" ng-click="doSomethingAwesome('lion',
'showImage')" > Click on him to see a large image.</p>
```

Content for Page 2:

```
<p>Here's a snake. <img src="snake.png" ng-click="doSomethingAwesome('snake',
'playSound')" >Click to make him hiss.</p>
```

Back in the Page controller, we then add the corresponding $scope function:

Page controller

```
$scope.doSomethingAwesome = function( id, action ) {
    console.log( "Going to do " + action + " with "+ id );
}
```

I can't figure out how to call that 'doSomethingAwesome' method from within the HTML string from the DB. I realize Angular has to parse the HTML string somehow, but how? I've read vague mumblings about the $compile service, and copied and pasted some examples, but nothing works. Also, most examples show dynamic content only getting set during the linking phase of the directive. We would want Page to stay alive throughout the life of the app. It constantly receives, compiles and displays new content as the user flips through pages.

In an abstract sense, I guess you could say we are trying to dynamically nest chunks of Angular within an Angular app, and need to be able to swap them in and out.

I've read various bits of Angular documentation multiple times, as well as all sorts of blog posts, and JS Fiddled with people's code. I don't know whether I'm completely misunderstanding Angular, or just missing something simple, or maybe I'm slow. In any case, I could use some advice.

html dom dynamic angularjs compilation

asked Aug 9 '13 at 23:55

giraffe_sense
313 1 4 4

$compile and the docs blogs surrounding it make me feel that I am also slow - even though I feel my js is quite strong - I think if I do get to grips with this I will make an idiots style blog - thats my specialty ! – landed Jun 4 at 9:10

3 Answers

`ng-bind-html-unsafe` only renders the content as HTML. It doesn't bind Angular scope to the resulted DOM. You have to use `$compile` service for that purpose. I created this plunker to demonstrate how to use `$compile` to create a directive rendering dynamic HTML entered by users and binding to the controller's scope. The source is posted below.

demo.html

```
<!DOCTYPE html>
<html ng-app="app">

  <head>
    <script data-require="angular.js@1.0.7" data-semver="1.0.7"
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.0.7/angular.js"></script>
    <script src="script.js"></script>
  </head>

  <body>
    <h1>Compile dynamic HTML</h1>
    <div ng-controller="MyController">
      <textarea ng-model="html"></textarea>
      <div dynamic="html"></div>
    </div>
  </body>

</html>
```

script.js

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

app.directive('dynamic', function ($compile) {
  return {
    restrict: 'A',
    replace: true,
    link: function (scope, ele, attrs) {
      scope.$watch(attrs.dynamic, function(html) {
        ele.html(html);
        $compile(ele.contents())(scope);
      });
    }
  };
});

function MyController($scope) {
  $scope.click = function(arg) {
    alert('Clicked ' + arg);
  }
  $scope.html = '<a ng-click="click(1)" href="#">Click me</a>';
}
```

answered Aug 10 '13 at 1:55

Buu Nguyen
26.1k 2 39 68

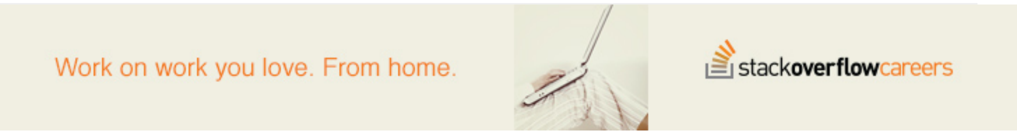
4 Thanks so much, Buu! Creating the attribute directive and adding the scope watch function were the two things I was missing. Now that this is working, guess I'll read up again on directives and $compile, to better understand what's going on under the hood. – giraffe_sense Aug 12 '13 at 18:29

@giraffe_sense you're welcome. I'm glad it helps. – Buu Nguyen Aug 12 '13 at 21:16

7 Me too!The Angular team could really do with improving the docs on this. – Craig Morgan Nov 19 '13 at 8:46

2 Wow, works perfectly. This was a huge help. I appreciate you writing this :) – ac360 Jan 15 '14 at 0:37

5 Thanks! Finally! angularjs official documentation sucks. – FlamesoFF Mar 11 '14 at 13:34



In angular 1.2.10 the line `scope.$watch(attrs.dynamic, function(html) {` was returning an invalid character error because it was trying to watch the value of `attrs.dynamic` which was html text.

I fixed that by fetching the attribute from the scope property

```
scope: { dynamic: '=dynamic'},
```

My example

```
angular.module('app')
  .directive('dynamic', function ($compile) {
    return {
      restrict: 'A',
      replace: true,
      scope: { dynamic: '=dynamic'},
      link: function postLink(scope, element, attrs) {
        scope.$watch( 'dynamic' , function(html){
          element.html(html);
          $compile(element.contents())(scope);
        });
      }
    };
  });
```

edited Jan 30 '14 at 0:44
krillgar
4,339 5 14 42

answered Jan 30 '14 at 0:25
Alexandros Spyropoulos
506 4 16

Hello, If I use *element.html* it return me TypeError: Cannot call method 'insertBefore' of null. So after some googling about that I find that I must use *element.append* But If I use that directive on multiple places - it generate multiplicate HTML. So 2 directives generate 4 same HTML code. Thanks for your answer. – DzeryCZ Feb 27 '14 at 23:57

I wouldn't use append in your place, I will have a look on that tonight and I'll get back to you. To be honest, I used this directive in quite a few places in a page without any issue. I'll try to reproduce the problem and I'll get back to you. – Alexandros Spyropoulos Mar 6 '14 at 15:34

1 @AlexandrosSpyropoulos I just test and see that my code runs okay even with 1.2.12. I think you probably missed the declaration <div dynamic="html"> in the HTML? (With that declaration, $watch watches the 'html' property in scope, not the actual HTML as you mentioned, so there should be no invalid char error.) If not, send me the plunkr that shows it doesn't work, I'll see what's wrong. – Buu Nguyen Mar 31 '14 at 18:46

You're probably right. I've been expecting back then, that html is actually a variable that contains html :P. It's good idea though to set a scope on your Directives. umur.io/… – Alexandros Spyropoulos Apr 4 '14 at 0:17

`$compile(ele.contents())(scope);` - this line solved my issue of not compiling angular components which are added dynamically. Thanks. – Mital Pritmani Apr 10 '14 at 12:40

Found in a google discussion group. Works for me.

```
var $injector = angular.injector(['ng', 'myApp']);
$injector.invoke(function($rootScope, $compile) {
  $compile(element)($rootScope);
});
```

answered Oct 15 '14 at 19:04
kwerle
560 4 16

□