leftshift

8 Tips for Angular.js Beginners

created on 08 September 2012 and tagged under angularjs tips development

We started working with Angular.js recently and after spending a few days on it, I realised that there a big need for beginner tutorials on it. I've tried to document some of the things you might need on day 1.

1. The documentation still sucks so it's okay if you're taking more time.

Yes, it’s really worth it. So do spend a little extra time. Here’s an example of the issues you would face:

> *Discussion from StackOverflow: http://stackoverflow.com/questions/10486769/cannot-get-to-rootscope*
>
> *Thanks, it makes perfect sense, but how did you know that? Was it in the docs? – Malvolio May 7 at 21:55*
>
> *@Mavolio No, he is one the 3 core developers. – ChrisOdney Jun 6 at 19:36*

The documentation sucks and you have to assume stuff. But there is light at the end of the tunnel.

If you need resources, you should try out the following

a. Run through the videos first – this should get your hooked. The following two are essential

- http://www.youtube.com/watch?v=WuiHuZq_cg4
- http://www.youtube.com/watch?v=IRelx4-ISbs

b. Run through the tutorial – http://docs.angularjs.org/tutorial/step_00

c. Run through the concepts – http://docs.angularjs.org/guide

d. Finally, keep this open – http://docs.angularjs.org/api/ – it won’t help

much other than just to remember what some function did.

e. Read this blog post – http://deansofer.com/posts/view/14/AngularJs-Tips-and-Tricks-UPDATED

f. If it did look interesting, add AngularUI to your project – https://github.com/angular-ui/angular-ui/

g. Go and join the AngularJS google group. It's quite active.

2. How to divide code

I have divided the code into two files. The first is the app.js and the second is the controllers. App.js contains code for setting up routes, app.factory functions and app.run to setup $rootScope Controllers.js contains all controllers so far

3. How to initialize the app

I do this in the App.js file

```
var app = angular.module('beta', [],
function($routeProvider, $locationProvider) {
  $routeProvider.when('/home', {
    templateUrl: '/partials/home',
    controller: HomeController
  });
  // When you put /home, it also automatically handles /home/ as well
  $routeProvider.when('/login', {
    templateUrl: '/partials/login',
    controller: LoginController
  });
  $routeProvider.otherwise( { redirectTo: '/login'} );

  // configure html5 to get links working
  // If you don't do this, you URLs will be base.com/#/home rather than base.com/home
  $locationProvider.html5Mode(true);
});
```

4. Create a set of functions that you can reuse

This is again done in the app.js file

```
app.factory('db', function() {
  var items = [];

  var modify = {};
  var modify.addItem = function(item) {
    items.push(item);
    return 'added item';
  };
  var modify.getItems = function() {
    return items;
  }
  return modify; // returning this is very important
});
```

now, in your controller, you can access these as follows

```
function MainCtrl = function ($scope, db) {
  $scope.save = function() {
    db.addItem('hello');
    console.log( db.getItems() );
  };
}
```

5. Controller are just for defining things

This might seem a little stupid for people who have been doing this for long, but well I stumbled on this, so this makes the cut.

Basically, when this means is that whenever you're trying to test your controller to try out something new, don't try this

```
function MainCtrl = function($scope, db) {
  db.addItem('hello');
}
```

This won’t work for obvious reasons What you need to do is this

```
function MainCtrl = function($scope, db) {
  $scope.save = function() {
    console.log( db.addItem('hello') );
  }
}
```

and now to run the save function, go to JADE and add

```
input(type='submit', name='submit', value='Submit',
ng-click='save()')
```

Now open the form, click on submit and check out console.log

6. Define functions in the $rootScope

The $rootScope is a global, which means that anything you add here, automatically becomes available in $scope in all controller. Nice eh! To set it up, you need to do something like this (I do it in app.js)

```
app.run(function($rootScope) {
  $rootScope.hello = function() {
    console.log('hello');
  }
});
```

This should now be available in controllers

```
function MainCtrl = function($scope) {
  $scope.save = function() {
    $scope.hello();
  }
};
```

7. Form validation

To use the validation which comes by default with Angular, you need to follow the following steps

a. give a “name” to your form e.g. <form name=”loginForm”>

b. mark all required input boxes as required e.g. <input type=’email’ required />

c. to turn on say email validation, you need to set type=’email’

d. check if the form is validating or not by checking loginForm.$invalid. To check this inside your controller, do $scope.loginForm.$invalid

8. Handling the menu via ng-controller

If you have defined ng-app in the HTML tag, and have defined an ng-view in the body somewhere. However, you want to keep the menu outside ng-view and still want to have access to it programatically to change menu based the fact that the user is logged in or not, you can define a ng-controller on the menu and make it work like a normal controller

Another thing I did was to put my menu in $rootScope so that each controller can mark which menu should be used and which one is active.

- *Sudhanshu Raheja*

We help companies create loveable apps for iOS, Android and the Web. If you would like to contact us for a consulation, a quote or to take us out for a beer, you can send us an email at pune@leftshift.io

Like this post? Subscribe to our newsletter

Find out more about our services
Find out who’s on our team
Subscribe to our newsletter
Check out our portfolio
Here’s how to join us
Follow us on Twitter
Get in touch with us
Check out our blog
Go back to home