Ahm NG tech

. UI

In ahm-ng, angular is adopted for rich ui framework. As we all know, the big challenge is to manage tons of javascript files. it forced ahm adopting another framework called requires help. RequireJS is AMD (Asynchronous Modules Definition) compatible framework. See link <http://www.integralist.co.uk/posts/AMD.html> to understand it.

AMD (Asynchronous Module Definition) requires us to modularize javascripts. Each javascript file defines its own dependencies. It decoupled with other files and improved the reusability. RequireJS is AMD javascript loader. More and more JS frameworks support AMD. Jquery1.7+/angularjs has a special version supporting AMD.

Before we really jump into the detail, we must understand the difference of injection between requireJS and angularjs.  [Angular DI involves components](https://docs.angularjs.org/guide/di). Examples of Angular components are factories, directives, and filters. Angular provides several ways to inject a component into something else, but is not concerned about how to load these components or where they are stored. They could be all stored in a single .js file, in multiple files, or inline in the HTML document. The general assumption is that each component is loaded at page load, but that might not be the case.

[RequireJS is a file and module loader](http://www.requirejs.org/). Each individual module is in it's own file. RequireJS uses ajax methods to load modules on demand as required.

Where Angular is more concerned with the IOC portion of the DI picture, RequireJS is more focused on file loading, data transfer, and memory conservation portions.

It is possible, and common, to use RequireJS to enhance the DI functionality of Angular, but unless it's a large scale app with hundreds of script files, it's normally not necessary.

Great example of requireJS <https://kielczewski.eu/2013/04/integrating-angularjs-with-requirejs/>

RequireJs provides 2 approaches to load dependent scripts.

1. Require(‘[module1’,’module2’….],function(){….});
2. Define(‘[module1’,’module2’….],function(){….return object/function/…});

We can see these two approaches having different purposes. Require() function is the entry point to start the javascript after the dependent javascript files loaded. It is usually included in main.js file which is defined in the index.html (see below)

<script data-main="Assets/Scripts/main.js" src="Assets/Scripts/Require.js"></script>

The approach of integrating angularjs with requireJS is

1. Only load requireJS in index.html file and define data-main property
2. In main.js file, all dependencies are declared in here. If any dependency doesn’t support AMD feature (non-AMD compliant), it must be specified where to find it.



What we do here is that we specify *paths* on where angular.js and angular-resource.js can be found and the property name for each path defines a module name for [RequireJS](http://requirejs.org/) to recognize it by. Notice there are omitted file extensions (.js) and it's not a mistake, you can see [RequireJS](http://requirejs.org/) docs why as it's irrelevant here. The shim section specifies dependencies between the modules we just defined. Additionally, for angular.js an exports property is required to give a variable name under which [AngularJS](http://angularjs.org/) API will be available.

1. Define angularjs app module



What it does is that it defines module app that requires module angular and angular-resource, and after they load, the function is executed with angular parameter that is needed to access [AngularJS](http://angularjs.org/) API (see exports property in configuration above). Inside this function we create angular module app and return it, so it can be available to controllers.

1. Create angularjs controller