

AngularJS and AJAX

Estimated time for completion: 30 minutes

Overview:

In this lab you will convert an AngularJS SPA style application to load and persist its resources using AJAX calls.

Goals:

* Load resources using the $http service.
* Load and update resources using the $resource service.

Lab Notes:

Don’t forget to reference the AngularJS documentation at <http://docs.angularjs.org/api>.

Update the countryLoader service to use AJAX

In this part of the lab, you will update the countryLoader service to request the list of countries using AJAX. The project contains a WebAPI REST service at **api/coutries** which will return the list of country names.

**Helpful links:**

* [The AngularJS documentation](http://docs.angularjs.org/api).
* [The $http service](http://docs.angularjs.org/api/ng.$http).

Steps:

1. Run and test the application. It should be completely functional with the exception of loading the resources from the REST services.
2. Open the “Countries.js” and locate the “countryLoader” service. Currently this service contains a hardcoded list of country codes.
3. Update this service to use the “$http” service to load the country data from the REST country service. The URL for this REST service is “/api/countries”. You should not need to make any changes to the “ContactsEditorCtrl” for this change.
4. As an AngularJS service is a singleton you should make sure that the data is only loaded once when the application starts.
5. Run and test your application.

.

Update the peopleLoader service to use AJAX

In this part of the lab, you will update the “peopleLoader” service to use the People REST service. Because we are going to load as well as update data we are going to use the $resource .

**Helpful links:**

* [The AngularJS $resource service](http://docs.angularjs.org/api/ngResource.$resource).

Steps:

1. Open the “People.js” and locate the “peopleLoader” service. Currently this service contains a hardcoded list of people.
2. Remove this hard coded list of people and replace this with a call to “$resource()” to create a REST resource service. The URL for the WebAPI REST service is “api/people/”.
3. Update the “ContactsEditorCtrl” constructor to use the updated “peopleLoader” service. Note that the copy action is no longer needed, this was only required for the undo action with local contacts.
4. Update the “save()” function to use the updated “peopleLoader” service to save changes. Make sure to redirect to the contacts list only if the update on the server was successful and to display the error message returned if the updated failed. Note you can test this using the first and last name properties as both are required.
5. Run and test your application.

Optional: Add actions to add new contacts or delete existing ones.

In this part of the lab, you update the application to support adding new contacts and deleting existing ones.

Steps:

1. Implement the insert and delete functionality.
2. Run and test your application.

Solutions:

The final solution for this lab is available in the ~/after directory.