

# Advanced Internet Programming

## Project no.10 – Angular and JS applications

Hubert Beczkowski, ID: 307290

Given task:

- Run and explain functionalities of given AngularJS examples

### Example 24 – Sorting in descending order

Given application function is to:

- Create a searchable list of population in world, continents and countries, which will be sorted on the population number

The application consists of index.html which invokes the angular module, and a json file containing all the data required (name of the places, population count).

Search:

| Country   | Population |
|---|------------|
| WORLD   | 6916183000 |
| Less developed regions                                      | 5675249000 |
| Less developed regions, excluding least developed countries | 4836442000 |
| Less developed regions, excluding China                     | 4284697000 |
| ASIA  | 4165440000 |
| South-Central Asia  | 1743101000 |
| Southern Asia   | 1691407000 |

```
<script>
var countryApp = angular.module('countryApp', []);
countryApp.controller('CountryCtrl', ['$scope', '$http', function (scope, http){
    http.get('countries.json').success(function(data) {
        scope.countries = data;
    });
}]);
</script>
```

The script invokes the angular.module, and describes the controller, while pointing at the json file from which data is taken.

```

<body ng-controller="CountryCtrl">
  Search:<input ng-model="query" type="text"/>
  <table>
    <tr>
      <th>Country</th>
      <th>Population</th>
    </tr>
    <tr ng-repeat="country in countries | filter:query | orderBy:'-population'">
      <td>{{country.name}}</td>
      <td>{{country.population}}</td>
    </tr>
  </table>
</body>

```

The body of the html consists of three parts:

1. The Search engine, which takes the text as query, whenever text is input in the search window, the query is looked for through the list.
2. The table description, which divides it into two columns- country and population
3. The descending order controller, which are filtered by given query and ordered by the "population field".

The sorting in descending order is applied in fairly simple and easy way thanks to the angularJS.

### Example 37- using links with routes for navigation between views

In example 37 we explore the routing options with use of angular module. Given 3 countries, we create route for each, which will function in a way, that whenever clicked, it will redirect to page with contents regarding given nation (in that case, the content is described by the country detail HTML, which only takes the name of the country from json file, and puts it in a header).

```

countryApp.config(function($routeProvider) {
  $routeProvider.
    when('/', {
      templateUrl: 'country-list.html',
      controller: 'CountryListCtrl'
    }).
    when('/:countryName', {
      templateUrl: 'country-detail.html',
      controller: 'CountryDetailCtrl'
    }).
    otherwise({
      redirectTo: '/'
    });
});

countryApp.controller('CountryListCtrl', function ($scope, $http){
  $http.get('countries.json').success(function(data) {
    $scope.countries = data;
  });
});

countryApp.controller('CountryDetailCtrl', function ($scope, $routeParams){
  $scope.name = $routeParams.countryName;
});

```

Given script defines what happens depending on given route. If the route is ended with a slash ("/") the user is redirected to the view with country list, containing three linked, clickable country names. If a country is clicked, the user is redirected to "/:countryName" which contains the name of the nation contained in a Header of another HTML page.

#### Example 41- Extracting the country details query into a service

Example 41 is somewhat an extension to example 37. In this one, the functionality is expanded to include additional information except for the name of the country.

```
countryApp.config(function($routeProvider) {
    $routeProvider.
        when('/', {
            templateUrl: 'country-list.html',
            controller: 'CountryListCtrl'
        }).
        when('/:countryName', {
            templateUrl: 'country-detail.html',
            controller: 'CountryDetailCtrl'
        }).
        otherwise({
            redirectTo: '/'
        });
});
```

The route provider in this case is pretty similar to one from the previous example, and works on the same principle (depending on whether or not there is "/" or "/:countryname" in the route, the page calls for different page to be invoked).

```
countryApp.factory('countries', function($http){
    return {
        list: function(callback){
            $http.get('countries.json').success(callback);
        },
        find: function(name, callback){
            $http.get('countries.json').success(function(data) {
                var country = data.filter(function(entry){
                    return entry.name === name;
                })[0];
                callback(country);
            });
        }
    };
});


countryApp.controller('CountryListCtrl', function ($scope, countries){
    countries.list(function(countries) {
        $scope.countries = countries;
    });
});

countryApp.controller('CountryDetailCtrl', function ($scope, $routeParams, countries){
    countries.find($routeParams.countryName, function(country) {
        $scope.country = country;
    });
});
```

The first part of the script describes from where the data is to be taken from. The callback makes it so that the function is passed to another one as a parameter and executed when the outer function is completed. So, the callback creates a parameter out of a function that invokes the data from the json file. That way we can give the countryApp.controllers a shorter route from which it can invoke data.

The other two parts are the controllers for the routing cases. In case of the route ending in slash, the countryApp.controller 'CountryListCtrl' is used, which controls the creation of list. If it ends in '/:countryName', then the countryApp.controller 'CountryDetailCtrl' is applied, which includes the route parameters included in country name, so the scope of the data that is invoked is expanded to the amount given in country-detail.html.

## United States of America

- Flag: 
- Population: 312,247,000
- Capital: Washington, D.C.
- GDP: \$16,244.00

Depending on what is described in country-detail.html, the output will change. In this case, the parameters inside it are flag which is invoked through the URL that is given in countries.json, the population, capital and GDP (all of those taken from countries.json).