Angular JS Custom Directive for Date Range

(Reusable component for all the projects)

VERSION CONTROL

Prepared by:	Neelakandan, R.
Date:	5/11/2015
Reviewed and Accepted by:	B.C.Subramanian
Date:	6/11/2015
Approved by:	Baskaran.Varadarajan, D.A.Soundararajan
Date:	29/12/2015

VERSION HISTORY

Date	Version	Author	Description
5/11/2015	0.1	Neelakandan, R.	Initial Draft Version
9/11/2015	1.0	B.C.Subramanian	Final Reviewed
21/12/2015	1.1	S.Kalaiyarasu	Reviewed
28/12/2015	1.2	Baskaran.Varadarajan	Final Approved

Contents

1. INTRODUCTION TO ANGULAR JS CUSTOM DIRECTIVE FOR DATE RANGE

1 INTRODUCTION

Custom directives are used in AngularJS to extend the functionality of HTML. Custom directives are defined using "directive" function. A custom directive simply replaces the element for which it is activated.

AngularJS application during bootstrap finds the matching elements and do one time activity using its compile () method of the custom directive then process the element using link () method of the custom directive based on the scope of the directive.

Directives are markers on a DOM element (such as an attribute, element name, comment or CSS class) that tell AngularJS's HTML compiler ($compile) to attach a specified behaviour to that DOM element (e.g. via event listeners), or even to transform the DOM element and its children.

- <projectName>DateRange is a custom directive used to select start and end dates from calendar.

- While clicking the Calendar button, two different calendar will be populated. Based on the user selection respective dates will be appear in the textbox.

1.1 PURPOSE

- <projectName>DateRange directive used to select start and end dates from calendar

1.2 SUPPORTING ELEMENTS BY ANGULAR JS TO CREATE DIRECTIVE FOR GRID

- **Element directives** − Directive activates when a matching element is encountered.
- **Attribute** − Directive activates when a matching attribute is encountered.
- **CSS** − Directive activates when a matching css style is encountered.
- **Comment** − Directive activates when a matching comment is encountered.

1.3 INTENDED AUDIENCE

- General Users who use the application:
  - ✓ General users will use the <projectName>DateRange directive for selecting start and end dates from calendar
- UI Developers:
  - ✓ UI Developers who want to work on refining or adding new functionalities to the directive will be able to understand the basic logic and mechanisms very swiftly upon referring this document.

1.4 DEFINING A DIRECTIVE

This section lists simple steps to define a custom directive in an AngularJS module. First, we need to define an Angular app.

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

Now, define a directive.

```
myApp.directive('myDirective', function() {
  return {
    restrict: 'E',
template: '<h1>I made a directive!</h1>'
  };
});
```

This defines a directive. restrict: 'E' means "restrict the usage of this directive to only Elements." Thus we embed this directive in the HTML page as

```
<body ng-app="myApp">
  <my-directive></my-directive>
</body>
```

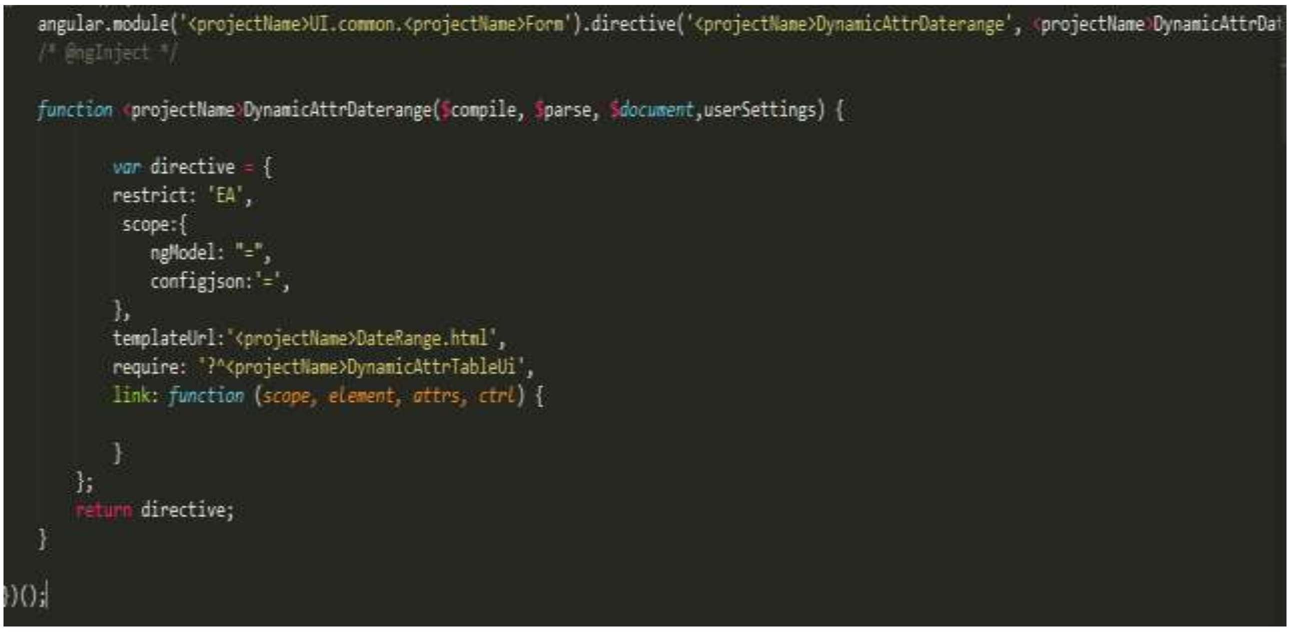
This code piece is equivalent to

```
<body ng-app="myApp">
  <h1>I made a directive!</h1>
</body>
```

Note that AngularJS maps the naming conventions from HTML's **my-directive** to JavaScript"s **myDirective**

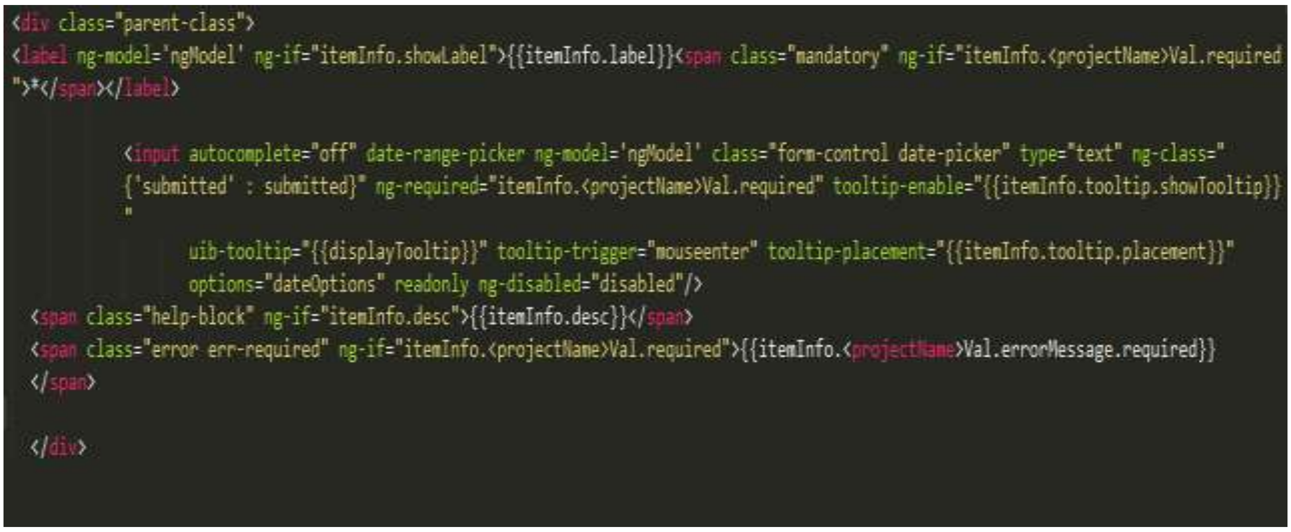
1.5 Sample Code

1.5.1 ANGULAR DIRECTIVE CODE



```
    angular.module('<projectName>UI.common.<projectName>Form').directive('<projectName>DynamicAttrDaterange', <projectName>DynamicAttrDat
    /* @ngInject */

    function <projectName>DynamicAttrDaterange($compile, $parse, $document,userSettings) {

            var directive = {
            restrict: 'EA',
             scope:{
                ngModel: "=",
                configjson:'=',
            },
            templateUrl:'<projectName>DateRange.html',
            require: '?^<projectName>DynamicAttrTableUi',
            link: function (scope, element, attrs, ctrl) {

            }
        };
        return directive;
    }

})();
```

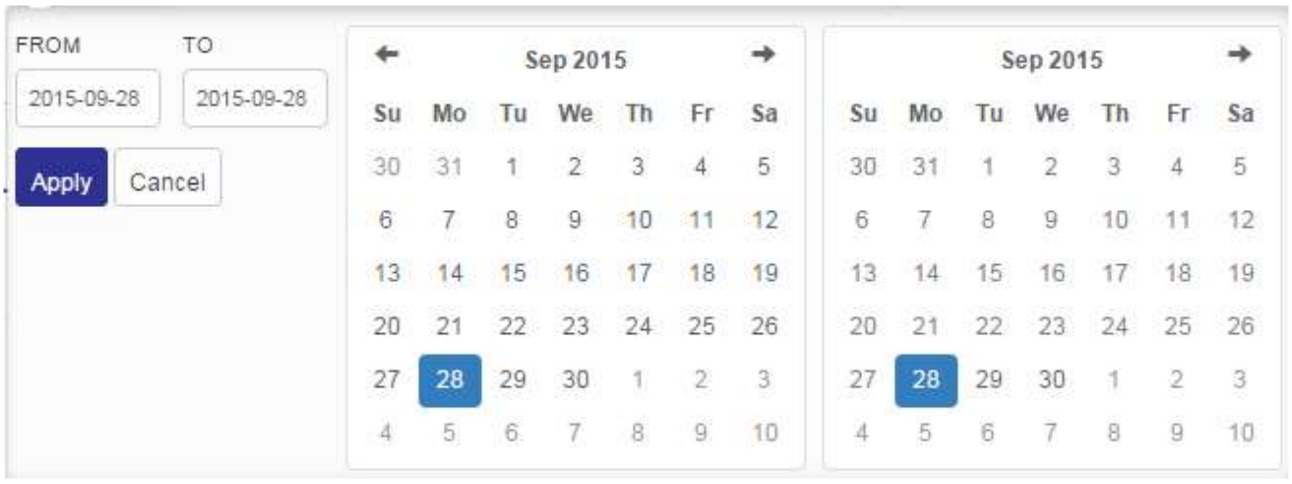
1.5.2 HTML VIEW CODE



```
<div class="parent-class">
<label ng-model='ngModel' ng-if="itemInfo.showLabel">{{itemInfo.label}}<span class="mandatory" ng-if="itemInfo.<projectName>Val.required
">*</span></label>

            <input autocomplete="off" date-range-picker ng-model='ngModel' class="form-control date-picker" type="text" ng-class="
            {'submitted' : submitted}" ng-required="itemInfo.<projectName>Val.required" tooltip-enable="{{itemInfo.tooltip.showTooltip}}
            "
                   uib-tooltip="{{displayTooltip}}" tooltip-trigger="mouseenter" tooltip-placement="{{itemInfo.tooltip.placement}}"
                   options="dateOptions" readonly ng-disabled="disabled"/>
  <span class="help-block" ng-if="itemInfo.desc">{{itemInfo.desc}}</span>
  <span class="error err-required" ng-if="itemInfo.<projectName>Val.required">{{itemInfo.<projectName>Val.errorMessage.required}}
  </span>

  </div>
```

2. DATE RANGE DIRECTIVE INFORMATION USING ANGULAR JS

2.1 VIEW (SAMPLE)



.

2.2 LIST OF BUTTONS

- Calendar Button
- Apply
- Cancel

2.3 DETAILED INFORMATION ABOUT THE 4 BUTTONS

2.3.1 Calendar Button

While clicking this button, two calendars will be appear.

2.3.2 Apply

While clicking this button, select date will be clear from textbox.

2.3.3 Cancel

While clicking this button, Calendar will be closed.

2.4 GENERAL INFORMATION ABOUT THE DIRECTIVE

- **Directive Name:**
  <projectName>-dynamic-attr-daterange

- **Directive Type:**
  Element Level Directive

- **Attributes:**
  date-options="dateOptions" ng-disabled="disable" configjson="configJSON"

2.5 DB JSON OBJECT STRUCTURE

```
"submittedDateRange" : {
        "type" : "date",
        "label" : "Submitted Date Range ",
        "name" : "submittedDateRange",
        "show" : true
         "<projectName>Val": {
                    "required": false
        }
```

2.6 JSON OBJECT VARIABLE DETAILS

Variable Name	Accepts type	Explanation
type	String	Directive Type “date”
name	String	
Label	String	Display
show	Boolean	Display or hide directive
required	String	Mandatory Field

2.7 Current issues and Addresses

- All Console.log( ) usages have been removed from the directive.
- HTML view and the Directive have been separated.
- Unused scope variables have been removed.

2.8 Pros and Cons

Pros:

- Easy way of creating dropdown with the required functionalities with minimal HTML code.
- Re-usable component which will be used across the application.

Cons:

- Need to have some understanding in the directive code to use it to good effect.