Angular JS Custom Directive for Button Radio

(Reusable component for all the projects)

VERSION CONTROL

Prepared by:	Dhivya Arinathan
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	Dhivya Arinathan	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 BUTTON RADIO

1.0 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.

- The projectNameDynamicAttrRadio(also referred as projectNameRadio in this document) is a custom directive that allows the user to choose only one of a predefined set of options.
- Each radio button is accompanied by a label describing the choice that the radio button represents.
- Selecting a radio button is done by clicking the mouse on the button or by using keyboard keys.

1.1 PURPOSE

- Use the projectNameDynamicAttrRadio directive only when there is a list of two or more options that are mutually exclusive and the user must select exactly one choice. In other words, clicking a non-selected radio button will deselect whatever other button was previously selected in the list.

1.2 SUPPORTING ELEMENTS BY ANGULARJS TO CREATE DIRECTIVE FOR DROPDOWN

- **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 projectNameRadio button to select one option among the many options.

- UI Developers

  UI developers who are working on refining the directive or adding new functionalities to the directive will be able to understand the basic logic of the behaviour and functionality of projectNameRadio after reading 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

```
(function () {
    angular.module('projectNameUI.common.projectNameForm').directive('projectNameDynamicAttrRadio', function () {

        return {
            templateUrl: 'app_modules/common/projectNameForm/directives/views/projectNameRadio.html',
            restrict: 'E',
            scope: {
                model: '=ngModel',
                configJSON: '=configjson',
                field: '@field',
                disabled: '=ngDisabled',
                changeFunction: '&ngChange',
                submitted: '='
            },

            link: function (scope, element, attrs) {

                };
            }
        };
    });
})();
```

1.5.2 HTML VIEW CODE

```
<input type="radio"
                    ng-required="itemInfo.projectNameVal.required"
                    name="{{attr}}"
                    ng-change="change()"
                    ng-disabled="disabled"
                    value="{{items.value}}"
                    ng-class="{'submitted' : submitted}"
                    ng-model="$parent.model"
                    tooltip-enable="{{itemInfo.tooltip.showTooltip}}"
                    uib-tooltip="{{displayTooltip}}" tooltip-trigger="mouseenter"
                    tooltip-placement="{{itemInfo.tooltip.placement}}"/>
```

2. BUTTON RADIO INFORMATION USING ANGULAR JS

2.1 VIEW (SAMPLE DATA)

2.2 GENERAL INFORMATION ABOUT THE DIRECTIVE

Param	Type	Details
ngModel	string	Assignable angular expression to data-bind to.
configjson	string	configjson=configJSON is used to pass the mongoScript values to the directive.
field	string	Property name of the button.
ngDisabled *(optional)*	string	Angular expression which disables the radio button when the value is true.
ngChange *(optional)*	string	Angular expression to be executed when input changes due to user interaction with the input element.

2.3 DB JSON OBJECT STRUCTURE

```
"status": {
    "type": "radio",
    "label": "Status",
    "name": "status",
    "default": "Active",
    "optionLabels": [
        {
            "value": "Active",
            "item": "Active"
        },
        {
            "value": "Inactive",
            "item": "Inactive"
        }
    ]
}
```

2.4 JSON OBJECT VARIABLE DETAILS

VARIABLE NAME	ACCEPTS TYPE	EXPLANATION
type	String	Specify the type of the field. In this case mention the type as “radio”.
label	String	This field holds the label value that is to be displayed in the UI screen.
name	String	This field hold the name of the object.
default	String	If by default, a particular radio button has to be selected, then use this variable to mention which button has to be selected.
optionLabels	Object	This object has details of the multiple options of the radio button
value	String	This variable holds the value of the button.
item	String	This variable contains the name of the options of the radio button that is to be displayed in the UI screen.

2.5 CURRENT ISSUES AND ADDRESSES

- Watch count:

  In the existing directive, there are too many watches. This has been reduced by implementing isolated scope in the modified directive.

- Unused scope variables:

  The unused scope variables have been removed.

- View and Controller:

  The HTML and controller have been separated into different files in the modified directive.