

Angular JS Custom Directive for Label

(Reusable component for all the projects)

VERSION CONTROL

Prepared by:	Kumar Sahoo, Srikant
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	Kumar Sahoo, Srikant	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 LABEL	4
1	Introduction.....	4
1.1	Purpose	4
1.2	Supporting elements by angular js to create directive for Grid	4
1.3	Intended Audience	5
1.4	Defining a directive.....	5
1.5	SAMPLE CODE	6
1.5.1	ANGULAR AND HTML DIRECTIVE CODE	6
2.	LABEL DIRECTIVE INFORMATION USING ANGULAR JS	7
2.1	VIEW	7
2.2	general information about the directive	7
2.3	DB JSON object structure	8
2.4	JSON object variable details	8
2.5	Current issues and Addresses	8

1. INTRODUCTION TO ANGULAR JS CUSTOM DIRECTIVE FOR LABEL

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>Label is a custom directive used to do label in page.

1.1 PURPOSE

- The <projectName>Label is mostly used for show the value in view page.

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>Label for view purpose.
- 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**

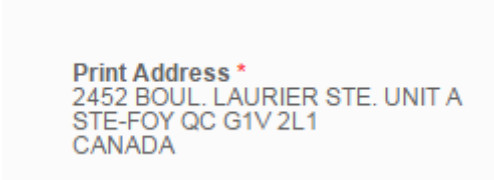
1.5 SAMPLE CODE

1.5.1 ANGULAR AND HTML DIRECTIVE CODE

```
(function () {  
  angular.module('<projectName>UI.common.<projectName>Form').directive('<projectName>DynamicAttrLabel', <projectName>DynamicAttrLabel);  
  /* @ngInject */  
  function <projectName>DynamicAttrLabel(userSettings, $filter, isUndefinedOrNullValueService) {  
    var template = '' +  
      '<label ng-if="itemInfo.show">{{itemInfo.label}}<span class="mandatory" ng-if="itemInfo.<projectName>Val.required">*</span></label>'+  
      '<span ng-if="itemInfo.show">{{label}}</span>';  
    var directive = {  
      restrict: 'E',  
      scope: true,  
      template: template,  
      link: function (scope, element, attrs, ctrl) {  
          
      }  
    };  
    return directive;  
  }  
})();
```

2. LABEL DIRECTIVE INFORMATION USING ANGULAR JS

2.1 VIEW



Print Address *
2452 BOUL. LAURIER STE. UNIT A
STE-FOY QC G1V 2L1
CANADA

2.2 GENERAL INFORMATION ABOUT THE DIRECTIVE

- **Directive Name:**
<projectName>-dynamic-attr-label
- **Directive Type:**
Label Level Directive
- **Attributes:**

Mandatory Attributes:

Add configjson="configJSON" this at the time of directive implementation

Example:

```
<<projectName>-dynamic-attr-label field="customer" configjson="configJSON" >
```

```
Ex:- <<projectName>-dynamic-attr-button field="Submit" configjson="configJSON"  
click="$eval(configJSON['Submit'].link)" disabledfield="true" show="{{showButton}}"  
model="ModelName" />
```

2.3 DB JSON OBJECT STRUCTURE

```
"customer" : {  
    "type" : "label",  
    "label" : "Customer",  
    "name" : "customer",  
    "show" : true  
}
```

2.4 JSON OBJECT VARIABLE DETAILS

Variable Name	Accepts type	Explanation
type	label	Directive Type "<projectName>label"
label	String	
name	String	Used in view page
show	Boolean	Display or hide

2.5 CURRENT ISSUES AND ADDRESSES

- All console.log () usages have been removed from the directive.
- Unused scope variables have been removed.
- Parent dependency have been removed with the usage of isolate scope.