Angular JS Custom Directive for Text

(Reusable component for all projects)

VERSION CONTROL

Prepared by:	Meenakshi Sankar
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	Meenakshi Sankar	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 TEXT

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.

- projectNametext is a custom directive is a section of a page that enables a user to enter text . Non-editable text boxes can serve the purpose of simply displaying text.

1.1 PURPOSE

- The projectNametextbox allows user to enter text, based on that DOM manipulation will be done.

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 projectNametext to add or edit listed items from source lists to destination lists.
- 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>
```

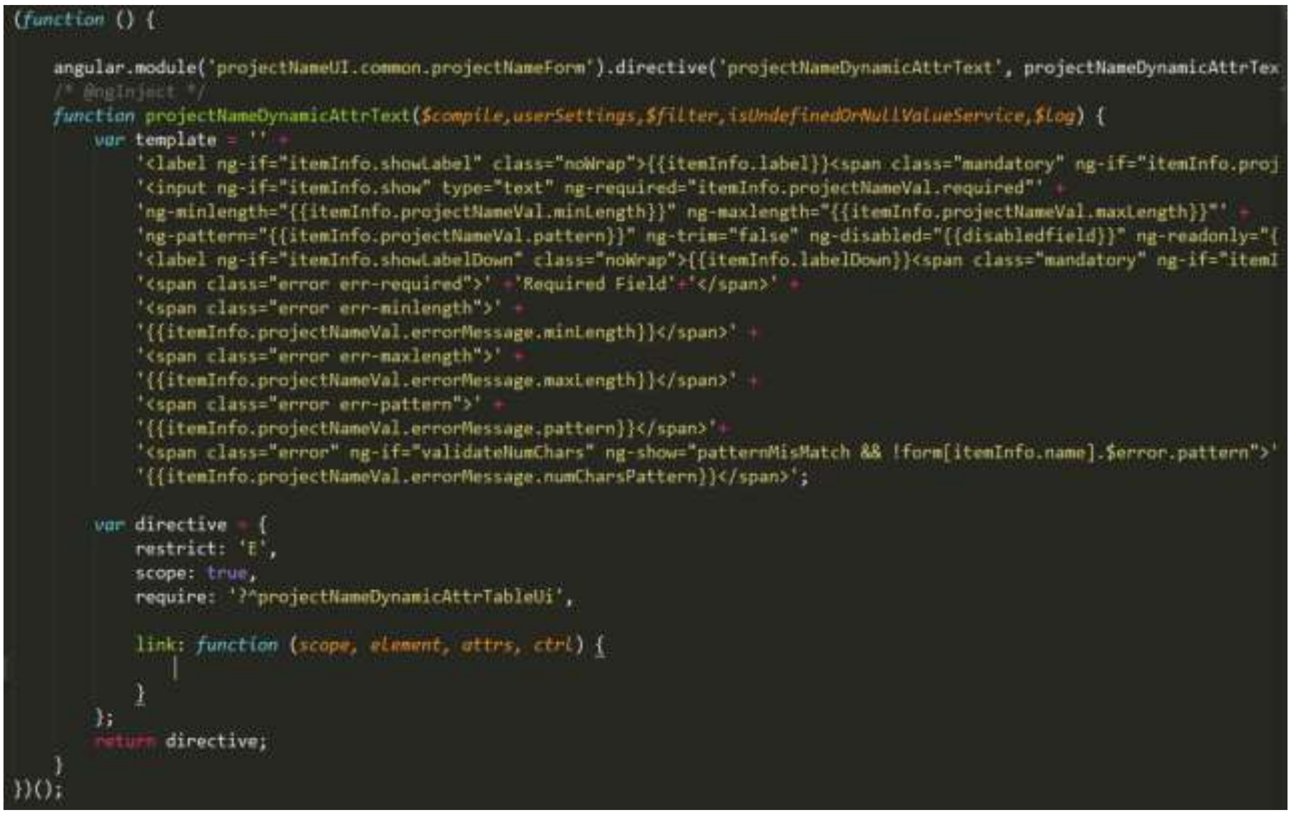
This code piece is equivalent to

```
<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 DATA:

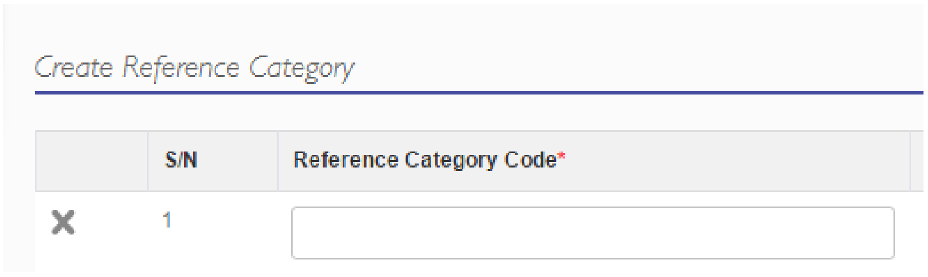
1.5.1 ANGULAR AND HTML DIRECTIVE CODE:

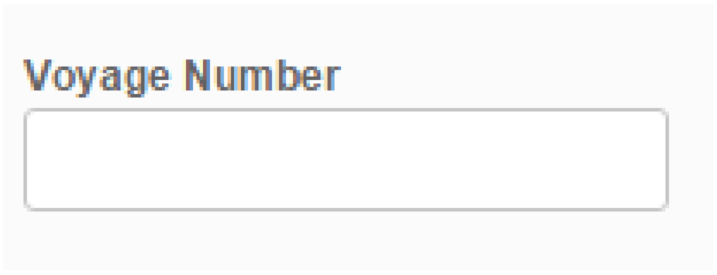


```
(function () {

    angular.module('projectNameUI.common.projectNameForm').directive('projectNameDynamicAttrText', projectNameDynamicAttrTex
    /* @ngInject */
    function projectNameDynamicAttrText($compile,userSettings,$filter,isUndefinedOrNullValueService,$log) {
        var template = '' +
            '<label ng-if="itemInfo.showLabel" class="noWrap">{{itemInfo.label}}<span class="mandatory" ng-if="itemInfo.proj
            '<input ng-if="itemInfo.show" type="text" ng-required="itemInfo.projectNameVal.required"' +
            'ng-minlength="{{itemInfo.projectNameVal.minLength}}" ng-maxlength="{{itemInfo.projectNameVal.maxLength}}"' +
            'ng-pattern="{{itemInfo.projectNameVal.pattern}}" ng-trim="false" ng-disabled="{{disabledfield}}" ng-readonly="{
            '<label ng-if="itemInfo.showLabelDown" class="noWrap">{{itemInfo.labelDown}}<span class="mandatory" ng-if="itemI
            '<span class="error err-required">' +'Required Field'+'</span>' +
            '<span class="error err-minlength">' +
            '{{itemInfo.projectNameVal.errorMessage.minLength}}</span>' +
            '<span class="error err-maxlength">' +
            '{{itemInfo.projectNameVal.errorMessage.maxLength}}</span>' +
            '<span class="error err-pattern">' +
            '{{itemInfo.projectNameVal.errorMessage.pattern}}</span>'+
            '<span class="error" ng-if="validateNumChars" ng-show="patternMisMatch && !form[itemInfo.name].$error.pattern">'
            '{{itemInfo.projectNameVal.errorMessage.numCharsPattern}}</span>';

        var directive = {
            restrict: 'E',
            scope: true,
            require: '?^projectNameDynamicAttrTableUi',

            link: function (scope, element, attrs, ctrl) {
                |
            }
        };
        return directive;
    }
})();
```

2. TEXT DIRECTIVE INFORMATION USING ANGULAR JS

2.1 VIEW (SAMPLE TEXTBOX)





2.2 GENERAL INFORMATION ABOUT THE DIRECTIVE

- **Directive Name:**
  projectName-textbox

- **Directive Type:**
  Element Level Directive

- **Attributes:**
  1. Field : String value to get the layout out detail of that particular field
  2. ng-model : Provides two way data binding
  3. configjson : Contains all the layout details for all the fields
  4. disabledfield-Disables or enables the textbox
  5. readonlyfield-User can't edit data in the textbox

- **Location:**

  Directive:
  PROJECTNAME_CustApps_Web_UI_Client\app\app_modules\common\projectNameForm\directives\projectNameTextbox.js

  View:
  PROJECTNAME_CustApps_Web_UI_Client\app\app_modules\common\projectNameForm\directives\views\projectNameTextbox.html

2.3 DB JSON OBJECT STRUCTURE

```
"workQueues_dueThreshold" : {
        "type" : "text",
        "label" : "Due Threshold (Days)",
        "name" : "dueThreshold",
        "show" : true
        }
```

2.4 JSON OBJECT VARIABLE DETAILS

Variable Name	Accepts type	Explanation
type	String	Directive Type “textbox”
name	String	
showLabel	Boolean	Display or hide label
show	Boolean	Display or hide directive
default	String	

2.5 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.
- Parent dependency have been removed with the usage of isolate scope.
- Pattern needs to changes in the mongoscript.eg : /(0-9)+$/ . Remove the / and gives as “(0-9)+$”

2.6 Pros and Cons

Pros:

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

Cons:

- Compulsory attributes not mentioned in the document leads to error.
- Need to have some understanding in the directive code to use it to good effect.