// Cal Confirmation messagebefore any action. Will not execute ng-click after cancel confirmation

app.directive('confirmationMessage', function () {

return {

priority: 1,

terminal: true,

link: function (scope, element, attr) {

var msg = attr.confirmationMessage || "Are you sure?";

var clickAction = attr.ngClick;

element.bind('click', function () {

if (window.confirm(msg)) {

scope.$eval(clickAction)

}

});

}

};

});

app.directive('onlyDigits', function () {

return {

restrict: 'A',

require: '?ngModel',

link: function (scope, element, attrs, ngModel) {

if (!ngModel) return;

ngModel.$parsers.unshift(function (inputValue) {

var digits = inputValue.split('').filter(function (s) {

return (!isNaN(s) && s != ' ');

}).join('');

ngModel.$viewValue = digits;

ngModel.$render(); //update the local view

return digits;

});

}

};

});

app.directive('onlyDigitsAndDecimals', function () {

return {

restrict: 'A',

require: '?ngModel',

link: function (scope, element, attrs, ngModel) {

if (!ngModel) return;

ngModel.$parsers.unshift(function (inputValue) {

var decimalCount = 0;

var digits = inputValue.split('').filter(function (s) {

if (s == '.' && decimalCount == 0) {

decimalCount++;

return (!isNaN(s) && s != ' ' || s == '.');

}

else

return (!isNaN(s) && s != ' ');

}).join('');

ngModel.$viewValue = digits;

ngModel.$render(); //update the local view

return digits;

});

}

};

});

app.directive('onlyNumeric', function () {

return {

require: 'ngModel',

link: function (scope, element, attr, ngModelCtrl) {

function fromUser(text) {

var transformedInput = text.replace(/[^0-9]/g, '');

if (transformedInput !== text) {

ngModelCtrl.$setViewValue(transformedInput);

ngModelCtrl.$render();

}

return transformedInput;

}

ngModelCtrl.$parsers.push(fromUser);

}

};

});

// Filter for converting any given bytes to MB,KB, GB, TB or PB

app.filter('bytes', function () {

return function (bytes, precision) {

if (isNaN(parseFloat(bytes)) || !isFinite(bytes)) return '-';

if (typeof precision === 'undefined') precision = 2;

var units = ['bytes', 'KB', 'MB', 'GB', 'TB', 'PB'];

if (bytes > 0) {

var number = Math.floor(Math.log(bytes) / Math.log(1024));

return (bytes / Math.pow(1024, Math.floor(number))).toFixed(precision) + ' ' + units[number];

}

else

return bytes + ' ' + units[0];

}

});

// End Filter

//Directive Implementaion for WysisHtml Editor

app.directive('richTextEditor', function ($log, $location) {

var self = this;

var directive = {

restrict: "A",

replace: true,

transclude: true,

require: '?ngModel',

template:

"<div>" +

"<textarea class=\"form-control\" id=\"richtexteditor-content\" placeholder=\"Enter text ...\"></textarea>" +

"</div>",

link: function ($scope, $element, $attrs) {

$scope.textarea = $('#richtexteditor-content').wysihtml5();

$scope.editor = $scope.textarea.data('wysihtml5').editor;

// $scope.editor = new wysihtml5.Editor( "richtexteditor-content", {

// toolbar : "richtexteditor-toolbar",

// parserRules: wysihtml5ParserRules

// });

$scope.$watch($attrs.ngModel, function (newValue, oldValue) {

$scope.textarea.innerHTML = newValue;

$scope.editor.setValue(newValue);

});

$scope.editor.on('change', function () {

$scope.content = $scope.editor.getValue();

//$scope.logValue($scope.editor.getValue());

});

$scope.cancel = function () {

$scope.$parent.cancel();

}

/\* $scope.save = function() {

var currentTemplateContent = $encryption.encodeHtml( $scope.editor.getValue() );

$scope.$parent.currentTemplate.content = currentTemplateContent;

$scope.$parent.save();

}

\*/

$scope.isClean = function () {

$scope.$parent.isClean();

}

$scope.editor.on('load', function () {

$('.wysihtml5-sandbox').contents().find('body').on("keyup", function (e) {

if ($scope.MatterEdit != undefined)

$scope.CountCharacters(e);

});

});

}

}

return directive;

});