Designer

```
//$vcPlugin_demo_demoEchartsdemo
vsPluginComponentModule.factory('$vcPlugin_demo_demoFilterWidget',
['$vsPluginRegister', '$timeout', function ($vsPluginRegister, $timeout) {

    var factory = {

        //[]
        showDataCategory: true,
        //[]
        showBorderCategory: true,
        //[]
        showBasicCategory: true,
        //[]
        showFixedCategory: true,
        //[]
        showEventCategory: true,
        //[]
        showTitleCategory: false,
        //[]
        showThresholdCategory: false,

        /*  */
        init: function(scope, element, component, $compile){
            scope.element = element;
            scope.component = component;
            //
            component.config.pageFilter = true;
            //1
            scope.component.config.chartDimensionCount = 1;
            component.config.selectedItem = null;
            //
            if(component.config.inited == null){
                component.config.inited = true;
                component.config.itemHeight = 40;
       component.config.itemLineHeight = 40;
       //
       component.config.itemBgColor = "#ffffff";
       component.config.itemFontColor = "#333333";
       //
       component.config.selectedItemBgColor = "#2990EA";
       component.config.selectedItemFontColor = "#ffffff";
            }
```

```
        },

        /*
         *
         */
        buildDataDescription: function(dataDescription, scope, element,
component, $compile){
            //
            scope.$on(event_refreshComponentData, function(target, param){
                //
                if(param.component != null && scope.component.id ===
param.component.id){
     return;
    }
                scope.queryComponentData(param, {
                    onSuccess: function(){
                        refreshChartView(scope, element, component,
$compile);
                    }
                });
            });

            /*
             *
             * component.config.pageFiltertrue
             */
            scope.component.context.getPageFilter = function(){
    var result = [];
    if(component.config.selectedItem != null){
        //
     var value = component.config.selectedItem.value
     if(value != null && value !== vsLang.heji){
      result.push({
       column: scope.getLastDimension().name,
       exp: "=",
       value: value
      });
     }
    }
    return result;
   }

   //
   scope.onItemClicked = function(){
    var value = component.config.selectedItem.value;
    //
    scope.cacheDimensionValue(scope.getLastDimension().name, value);
    $timeout(function(){
     scope.notifyDimensionValueFilterEvent({
      queryConditionDimensions: true
     });
    });
```

```
   }

        },

        /*
         *
         */
        buildChartDescription: function(scope, element, component,
$compile, $sce){

   scope.isSelectedItem = function(item){
    return component.config.selectedItem != null && ""+item.value ===
""+component.config.selectedItem.value;
   }

   scope.getItemStyle = function(item){
    if(scope.isSelectedItem(item)){
     return {
            'background-color':component.config.selectedItemBgColor,
            'color': component.config.selectedItemFontColor
        }
    }else{
     return {
            'background-color':component.config.itemBgColor,
            'color': component.config.itemFontColor
        }
    }

   }

   var html = [];
   html.push("<div
style='height:100%;overflow-x:auto;overflow-y:hidden;'>");
   html.push(" <div ng-show='component.config.datasourceConfig.dimensions
== null' style='height:100%;width:100%;display:table;'>");
   html.push("  <div
style='text-align:center;background-color:#f0f0f0;vertical-align:middle;di
splay:table-cell;cursor:pointer;'>");
   html.push("   ");
   html.push("  </div>");
   html.push(" </div>");
   html.push(" <div ng-show='component.config.datasourceConfig.dimensions
!= null' style='height:100%;width:100%;display:table;'>");
   html.push("  <div
style='vertical-align:middle;display:table-cell;text-align:center;cursor:p
ointer;word-break: keep-all;white-space:nowrap;'
ng-style=\"getItemStyle(item)\" ng-repeat='item in
component.config.optionItems track by $index'
ng-model='component.config.selectedItem' ng-click='onItemClicked()'
uib-btn-radio=\"item\">");
   html.push("   {{item.label}}");
   html.push("  </div>");
   html.push(" </div>");
```

```
    html.push("</div>");

    var el = $compile(html.join(""))( scope );
    element.html(el);

    {
     //
                 var categoryDesc = {
      name: "unselected",
      title: "",
      groups: []
     };

     categoryDesc.groups.push({
      name: "itemBg",
      title: {
       text: ""
      },
      elements: [{
       title: "",
       type: "colorpicker",
       bind: "itemBgColor"
      }]
     });

     categoryDesc.groups.push({
      name: "itemText",
      title: {
       text: ""
      },
      elements: [{
       title: "",
       type: "configSlide",
       bind: "itemFontSize",
       config: {
        slideEnd: 100
       }
      },{
       title: "",
       type: "colorpicker",
       bind: "itemFontColor"
      }]
     });

     component.description.categories.push(categoryDesc);

    }

    {
     //
                 var categoryDesc = {
      name: "selected",
      title: "",
```

```
     groups: []
    };

    categoryDesc.groups.push({
     name: "itemBg",
     title: {
      text: ""
     },
     elements: [{
      title: "",
      type: "colorpicker",
      bind: "selectedItemBgColor"
     }]
    });

    categoryDesc.groups.push({
     name: "itemText",
     title: {
      text: ""
     },
     elements: [{
      title: "",
      type: "configSlide",
      bind: "selectedItemFontSize",
      config: {
       slideEnd: 100
      }
     },{
      title: "",
      type: "colorpicker",
      bind: "selectedItemFontColor"
     }]
    });

    component.description.categories.push(categoryDesc);

   }
  }
    };

    var buildSettingDescription = function(scope, element, component,
$compile){
        //[]
        var category = {
            name: "setting",
            title: "",
            groups: []
        };
        component.description.categories.push(category);

        category.groups.push({
            name: "text",
            title: {
```

```
                text: ""
            },
            elements: [{
                title: "",
                type: "text-area",
                bind: "content"
            }]
        });
        category.groups.push({
            name: "font",
            title: {
                text: vsLang.font
            },
            elements: [{
                title: "",
                type: "configSlide",
                bind: "fontSize",
                config: {
                    slideStart: 10,
                    slideEnd: 100
                }
            },{
                title: "",
                type: "horizontal-align",
                bind: "textAlign"
            },{
                title: "",
                type: "colorpicker",
                bind: "color"
            }]
        });

    }

     //
    var refreshChartView = function(scope, element, component, $compile){
        var dimensions = component.config.datasourceConfig.dimensions;
        //context
var data = component.context.data;
//
var initValue = scope.parseInitValue();
//
var optionItems = [];
//
var selectedItem = null;
//
for(var i = 0; i < data.length; i++){
 var value = data[i][dimensions[dimensions.length-1].name];
 var item = {
  label: ""+value,
  value: ""+value
 };
 //'',"All"
```

```
   if(value != null && value === vsLang.heji){
       //
    item.label =
scope.getDimensionSummaryAlias(dimensions[dimensions.length-1].name);
   }
   optionItems.push(item);
  }
  //
  if(initValue != null && component.context.first_render_init_value ==
null){
   component.context.first_render_init_value = initValue;
   selectedItem = {
    value: initValue,
    label: initValue
   };
  }else{
      //
   var cachedSelectedValue =
scope.getCachedDimensionValue(dimensions[dimensions.length-1].name);
   for(var i = 0; i < optionItems.length; i++){
    if(""+optionItems[i].value === ""+cachedSelectedValue){
     selectedItem = optionItems[i];
     break;
    }
   }
  }
  scope.component.config.optionItems = optionItems;
  //
  if(selectedItem == null && optionItems.length > 0){
   selectedItem = optionItems[0];
  }
  component.config.selectedItem = selectedItem;
  //
  scope.cacheDimensionValue(dimensions[dimensions.length-1].name,
selectedItem == null ? null : selectedItem.value);
    };

    //"demo"
```

```
    $vsPluginRegister.register("demo", "demoFilterWidget", factory);

}]);
```

View

```
//
var build_demoFilterWidget_component = function(scope, element, $compile,
$timeout){

    var component = scope.component;

    component.config.selectedItem = null;

    scope.isSelectedItem = function(item){
  return component.config.selectedItem != null && ""+item.value ===
""+component.config.selectedItem.value;
 }

 scope.getItemStyle = function(item){
  if(scope.isSelectedItem(item)){
   return {
          'background-color':component.config.selectedItemBgColor,
          'color': component.config.selectedItemFontColor
      }
  }else{
   return {
          'background-color':component.config.itemBgColor,
          'color': component.config.itemFontColor
      }
  }

 }

 var html = [];
 html.push("<div style='height:100%;overflow-x:auto;overflow-y:hidden;'>");
 html.push(" <div ng-show='component.config.datasourceConfig.dimensions ==
null' style='height:100%;width:100%;display:table;'>");
 html.push("  <div
style='text-align:center;background-color:#f0f0f0;vertical-align:middle;di
splay:table-cell;cursor:pointer;'>");
 html.push("   ");
 html.push("  </div>");
 html.push(" </div>");
```

```
 html.push(" <div ng-show='component.config.datasourceConfig.dimensions !=
null' style='height:100%;width:100%;display:table;'>");
 html.push("  <div
style='vertical-align:middle;display:table-cell;text-align:center;cursor:p
ointer;word-break: keep-all;white-space:nowrap;'
ng-style=\"getItemStyle(item)\" ng-repeat='item in
component.config.optionItems track by $index'
ng-model='component.config.selectedItem' ng-click='onItemClicked()'
uib-btn-radio=\"item\">");
 html.push("   {{item.label}}");
 html.push("  </div>");
 html.push(" </div>");
 html.push("</div>");

 var el = $compile(html.join(""))( scope );
 element.html(el);

    //designer.js
    var refreshChartView = function(){
        var dimensions = component.config.datasourceConfig.dimensions;
        //context
  var data = component.context.data;
  //
  var initValue = scope.parseInitValue();
  //
  var optionItems = [];
  //
  var selectedItem = null;
  //
  for(var i = 0; i < data.length; i++){
   var value = data[i][dimensions[dimensions.length-1].name];
   var item = {
    label: ""+value,
    value: ""+value
   };
   //'',"All"
   if(value != null && value === vsLang.heji){
       //
    item.label =
scope.getDimensionSummaryAlias(dimensions[dimensions.length-1].name);
   }
   optionItems.push(item);
  }
  //
  if(initValue != null && component.context.first_render_init_value ==
null){
   component.context.first_render_init_value = initValue;
   selectedItem = {
    value: initValue,
    label: initValue
   };
  }else{
      //
```

```
   var cachedSelectedValue =
scope.getCachedDimensionValue(dimensions[dimensions.length-1].name);
   for(var i = 0; i < optionItems.length; i++){
    if(""+optionItems[i].value === ""+cachedSelectedValue){
     selectedItem = optionItems[i];
     break;
    }
   }
  }
  scope.component.config.optionItems = optionItems;
  //
  if(selectedItem == null && optionItems.length > 0){
   selectedItem = optionItems[0];
  }
  component.config.selectedItem = selectedItem;
  //
  scope.cacheDimensionValue(dimensions[dimensions.length-1].name,
selectedItem == null ? null : selectedItem.value);
    };

    //
    scope.$on(event_refreshComponentData, function(target, param){
        //
        if(param.component != null && scope.component.id ===
param.component.id){
   return;
  }
        scope.queryComponentData(param, {
            onSuccess: function(){
                refreshChartView(scope, element, component, $compile);
            }
        });
    });

    /*
     *
     * component.config.pageFiltertrue
     */
    scope.component.context.getPageFilter = function(){
  var result = [];
  if(component.config.selectedItem != null){
      //
   var value = component.config.selectedItem.value
   if(value != null && value !== vsLang.heji){
    result.push({
     column: scope.getLastDimension().name,
     exp: "=",
     value: value
    });
   }
  }
  return result;
 }
```

```
//
scope.onItemClicked = function(){
 var value = component.config.selectedItem.value;
 //
 scope.cacheDimensionValue(scope.getLastDimension().name, value);
 scope.notifyDimensionValueFilterEvent({
  queryConditionDimensions: true
 });
```

```
    }

  }
```