Week 2 Unit 4

SORTING, GROUPING AND FILTERING

Please perform the exercises below in your app project as shown in the video.

Table of Contents

[1 Filtering 2](#_Toc449110175)

[2 Sorting and Grouping 4](#_Toc449110176)

## Preview

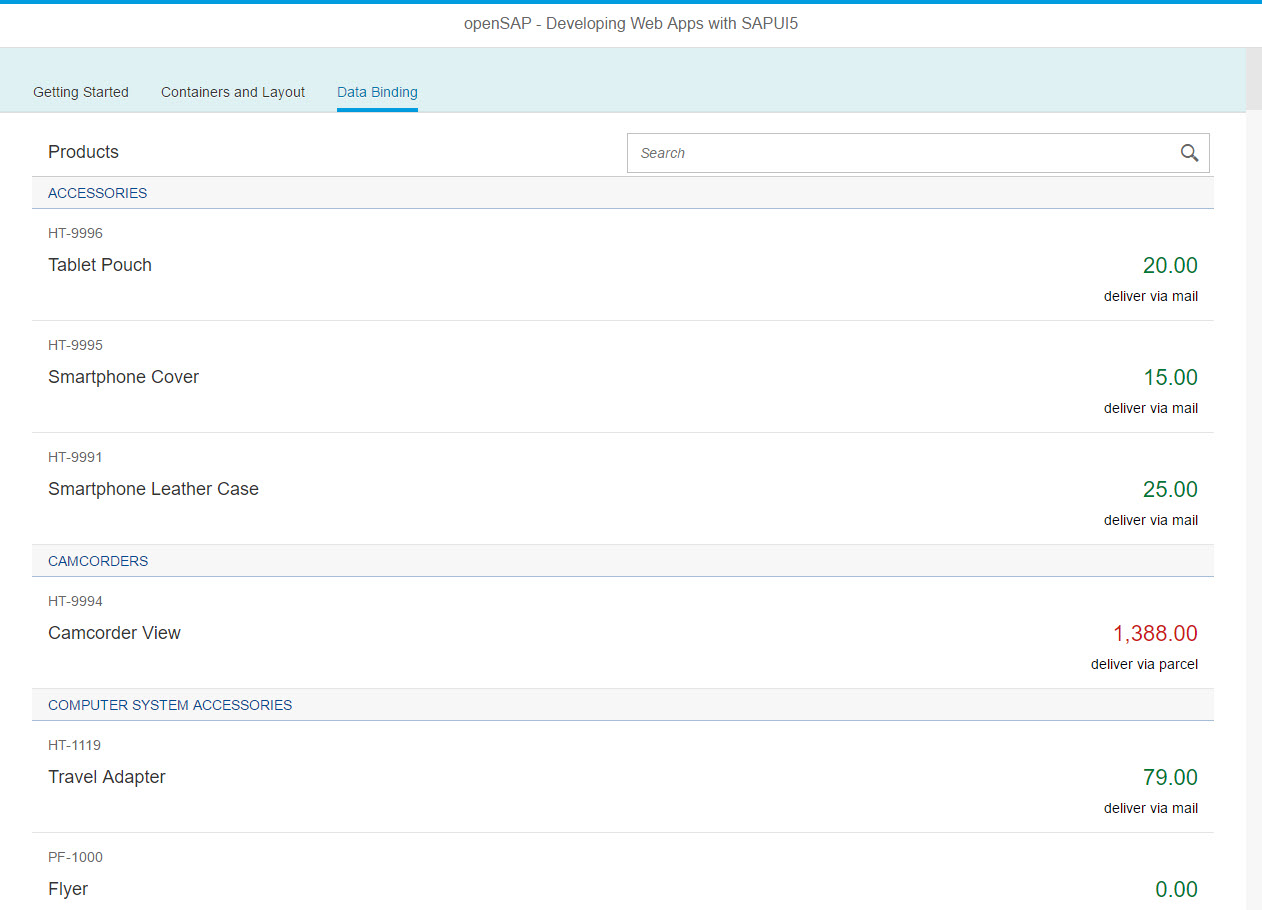


Figure 1 - Preview of the app after doing this unit’s exercises

# Filtering

Let us first add the sap.m.SearchField to the list header.

### Preview

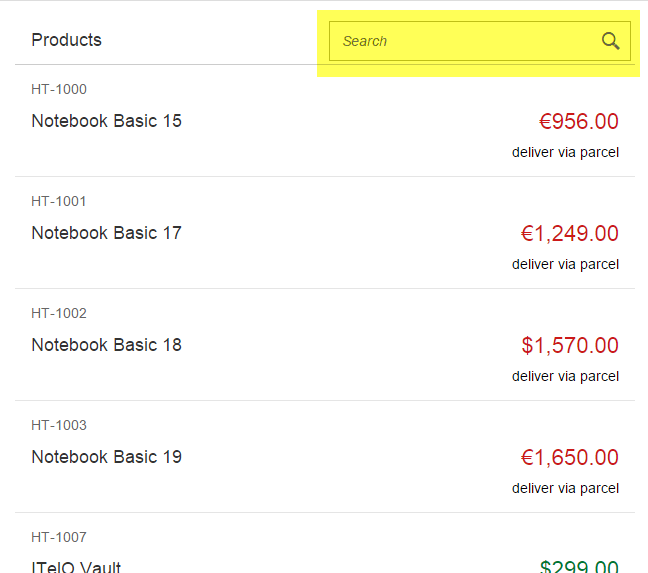


Figure 2 - List with a SearchField in its header

### webapp/view/App.view.xml

|  |
| --- |
| …  <IconTabFilter  text="{i18n>dataBindingFilter}" key="db">  <content>  **<List**  **id="productsList"**  **items="{/ProductSet}">**  **<headerToolbar>**  **<Toolbar>**  **<Title text="{i18n>productListTitle}"/>**  **<ToolbarSpacer/>**  **<SearchField width="50%" search="onFilterProducts"/>**  **</Toolbar>**  **</headerToolbar>**  <items>  …  </items>  </List>  </content>  </IconTabFilter>  … |

**We will put the SearchField into the list’s ‘HeaderToolbar’ where we can also place the ‘headerText’. The ‘headerText’ can therefore be removed from the list properties. For the ‘search’ event we provide the handler function ‘onFilterProducts’ which we need to implement in the controller.**

webapp/controller/App.controller.js

|  |
| --- |
| sap.ui.define([  "sap/ui/core/mvc/Controller",  "sap/m/MessageToast",  **"sap/ui/model/Filter",  "sap/ui/model/FilterOperator",**  "opensap/myapp/model/formatter" ], function (Controller, MessageToast, Filter, FilterOperator, formatter) {  "use strict";   return Controller.extend("opensap.myapp.controller.App", {   formatter : formatter,   onShowHello : function () {  // read msg from i18n model  var oBundle = this.getView().getModel("i18n").getResourceBundle();  var sRecipient = this.getView().getModel("helloPanel").getProperty("/recipient/name");  var sMsg = oBundle.getText("helloMsg", [sRecipient]);   // show message  MessageToast.show(sMsg);  }**,   onFilterProducts : function (oEvent) {   // build filter array  var aFilter = [], sQuery = oEvent.getParameter("query"),  // retrieve list control  oList = this.getView().byId("productsList"),  // get binding for aggregation 'items'  oBinding = oList.getBinding("items");   if (sQuery) {  aFilter.push(new Filter("ProductID", FilterOperator.Contains, sQuery));  }  // apply filter. an empty filter array simply removes the filter  // which will make all entries visible again  oBinding.filter(aFilter);  }**  }); }); |

**Applying filters (and sorters) can be done using the list’s ‘items’ binding. So we first retrieve the list control using its id suffix ‘productList’. The control object gives us access to the list binding object. Now we can create our filter object for field ‘ProductId’. We choose a filter operator and fetch the search field value from the handler function’s event parameter. Finally, we can call the binding’s filter method with our new filter.**

**Please note: With the above code we have implemented a search which is case-sensitive. When you try searching for a product ID, take care to write the same letters in upper or lower case as they are in the actual product ID.**

### Related Information

[API Reference: sap.ui.model.Filter](https://sapui5.hana.ondemand.com/docs/api/symbols/sap.ui.model.Filter.html)

[API Reference: sap.ui.model.FilterOperator](https://sapui5.hana.ondemand.com/docs/api/symbols/sap.ui.model.FilterOperator.html)

[API Overview and Samples: sap.m.SearchField](https://sapui5.hana.ondemand.com/explored.html#/entity/sap.m.SearchField/samples)

# Sorting and Grouping

Now, we will add sorting and grouping to our list.

### Preview

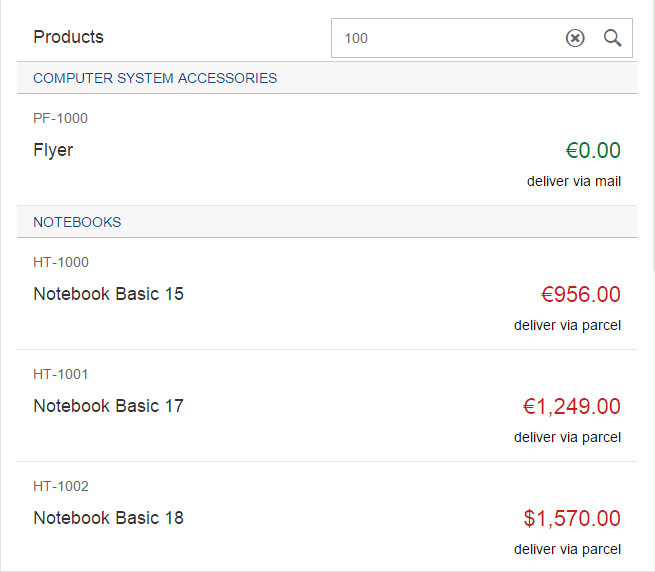


Figure 3 - List sorted and grouped by ‘Category’

### webapp/view/App.view.xml

|  |
| --- |
| …  <IconTabFilter  text="{i18n>dataBindingFilter}" key="db">  <content>  <List id="productsList" items**="{  path : '/ProductSet',  sorter : {  path : 'Category',  group : true  }  }**">  <headerToolbar>  <Toolbar>  <Title text="{i18n>productListTitle}"/>  <ToolbarSpacer/>  <SearchField width="50%" search="onFilterProducts"/>  </Toolbar>  </headerToolbar>  … |

All we need to do is to add sorting and grouping to the list’s binding information:

### Related Information

[API Reference: sap.ui.model.Sorter](https://sapui5.hana.ondemand.com/docs/api/symbols/sap.ui.model.Sorter.html)

**Coding Samples**

Any software coding or code lines/strings (“Code”) provided in this documentation are only examples and are not intended for use in a productive system environment. The Code is only intended to better explain and visualize the syntax and phrasing rules for certain SAP coding. SAP does not warrant the correctness or completeness of the Code provided herein and SAP shall not be liable for errors or damages cause by use of the Code, except where such damages were caused by SAP with intent or with gross negligence.