

Solicitud de Productos (Parte 2)

Ahora vamos a crear un modelo local para asignar la lista de productos previamente consultados desde el botón Buscar.

Agregamos la función **setProductModel** en el Helper **HomeHelper.js**:

```

1  sap.ui.define([
2      "com/bootcamp/sapui5/freestyle/utils/HomeService",
3      "sap/ui/model/json/JSONModel"
4  ], function (HomeService, JSONModel) {
5      "use strict";
6
7      return {
8          init: function (oNorthwindModel) {
9              this._oNorthwindModel = oNorthwindModel;
10         },
11
12         getDataProducts: async function() {
13             let oFilters = [];
14             return HomeService.readProducts(this._oNorthwindModel, oFilters);
15         },
16
17         setProductModel: async function (oController, oDatos) {
18             let oListModel = oController.getOwnerComponent().getModel('ProductCollection');
19             if(!oListModel){
20                 const oModel = new JSONModel([]);
21                 oModel.setSizeLimit(1000000);
22                 oController.getOwnerComponent().setModel(oModel, "ProductCollection");
23                 oListModel = oController.getOwnerComponent().getModel('ProductCollection');
24             }
25             oListModel.setData(oDatos);
26         },
27     },

```

En este punto usamos la librería [JSONModel](#)("sap/ui/model/json/JSONModel")

JavaScript

```

setProductModel: async function (oController, oDatos) {
    let oListModel =
oController.getOwnerComponent().getModel('ProductCollection');

```

```
if(!oListModel){
    const oModel = new JSONModel([]);
    oModel.setSizeLimit(1000000);
    oController.getOwnerComponent().setModel(oModel, "ProductCollection");
    oListModel =
oController.getOwnerComponent().getModel('ProductCollection');
}

oListModel.setData(oDatos);
},
```

Ahora Invocamos la función desde el controlador de la vista **Home.controller.js**:



Ahora agregamos en la vista Home una [tabla sap.m.table](#):

A esta tabla le asignamos en la propiedad `items` el modelo local creado previamente (`ProductCollection`)

```
Home.view.xml M X JS HomeHelper.js M i18n.properties M metadata.xml App.vi
freestyle > webapp > view > Home.view.xml
1 <mvc:View controllerName="com.bootcamp.sapui5.freestyle.controller.Home"
4 <Page id="page" title="{i18n>title}">
11 <VBox id="vTest2">
12
13 <Table id="idProductsTable"
14 inset="false"
15 items="{
16 path: 'ProductCollection/'
17 }">
18 <columns>
19 <Column id="ID_01">
20 <Text id="ID_I01" text="{i18n>Product}" />
21 </Column>
22 <Column id="ID_02">
23 <Text id="ID_I05" text="{i18n>UnitPrice}" />
24 </Column>
25 <Column id="ID_05">
26 <Text id="ID_I04" text="{i18n>UnitsInStock}" />
27 </Column>
28 <Column id="ID_03">
29 <Text id="ID_I02" text="{i18n>QuantityPerUnit}" />
30 </Column>
```

Se hace el binding con las diferentes columnas configuradas así:

```

Home.view.xml M X JS HomeHelper.js M i18n.properties M metadata.xml App.view.xml JS HomeSer
freestyle > webapp > view > Home.view.xml
1  <mvc:View controllerName="com.bootcamp.sapui5.freestyle.controller.Home"
4  <Page id="page" title="{i18n>title}">
11  <VBox id="vTest2">
13  <Table id="idProductsTable"
14  </columns>
15  <items>
16  <ColumnListItem id="ID_06" vAlign="Middle">
17  <cells>
18  <ObjectIdentifier id="ID_07"
19  title="{ProductCollection>ProductID}"
20  text="{ProductCollection>ProductName}" />
21  <ObjectNumber id="ID_11"
22  number="{
23  parts:[{path:'ProductCollection>UnitPrice'},{path:''}],
24  type: 'sap.ui.model.type.Currency',
25  formatOptions: {showMeasure: true}
26  }"
27  unit="USD" />
28  <Text id="ID_09"
29  text="{ProductCollection>UnitsInStock}" />
30  <Text id="ID_08"
31  text="{ProductCollection>QuantityPerUnit}" />
32  </cells>
33  </ColumnListItem>
34  </items>
35  </Table>
36  </VBox>
37  </Page>
38  </mvc:View>

```

Java

```

<Table id="idProductsTable"
  inset="false"
  items="{
    path: 'ProductCollection/'
  }">
  <columns>
    <Column id="ID_01">
      <Text id="ID_T01" text="{i18n>Product}" />
    </Column>
    <Column id="ID_02">
      <Text id="ID_T05" text="{i18n>UnitPrice}" />
    </Column>
    <Column id="ID_05">
      <Text id="ID_T04" text="{i18n>UnitsInStock}" />
    </Column>
    <Column id="ID_03">

```

```
        <Text id="ID_T02" text="{i18n>QuantityPerUnit}" />
    </Column>
</columns>
<items>
    <ColumnListItem id="ID_06" vAlign="Middle">
        <cells>
            <ObjectIdentifier id="ID_07"
                title="{ProductCollection>ProductID}"
                text="{ProductCollection>ProductName}" />
            <ObjectNumber id="ID_11"
                number="{
parts:[{path:'ProductCollection>UnitPrice'},{path:''}],
                type: 'sap.ui.model.type.Currency',
                formatOptions: {showMeasure: true}
            }"
            unit="USD" />
            <Text id="ID_09"
                text="{ProductCollection>UnitsInStock}" />
            <Text id="ID_08"
                text="{ProductCollection>QuantityPerUnit}" />
        </cells>
    </ColumnListItem>
</items>
</Table>
</VBox>
```

Ejecutamos la aplicación y damos click en el Botón Buscar:

SAP Freestyle SAPUI5

Control Text de Prueba con i18n

Buscar

Product	Unit Price	Units In Stock	Quantity Per Unit
1 Chai	18.00 USD	39	10 boxes x 20 bags
2 Chang	19.00 USD	17	24 - 12 oz bottles
3 Aniseed Syrup	10.00 USD	13	12 - 550 ml bottles
4 Chef Anton's Cajun Seasoning	22.00 USD	53	48 - 6 oz jars
5 Chef Anton's Gumbo Mix	21.35 USD	0	36 boxes
6 Grandma's Boysenberry Spread	25.00 USD	120	12 - 8 oz jars
7 Uncle Bob's Organic Dried Pears	30.00 USD	15	12 - 1 lb pkgs.
8 Northwoods Cranberry Sauce	40.00 USD	6	12 - 12 oz jars
9 Mishi Kobe Niku	97.00 USD	29	18 - 500 g pkgs.
10 Ikura	31.00 USD	31	12 - 200 ml jars
11 Queso Cabrales	21.00 USD	22	1 kg pkg.

Fragment

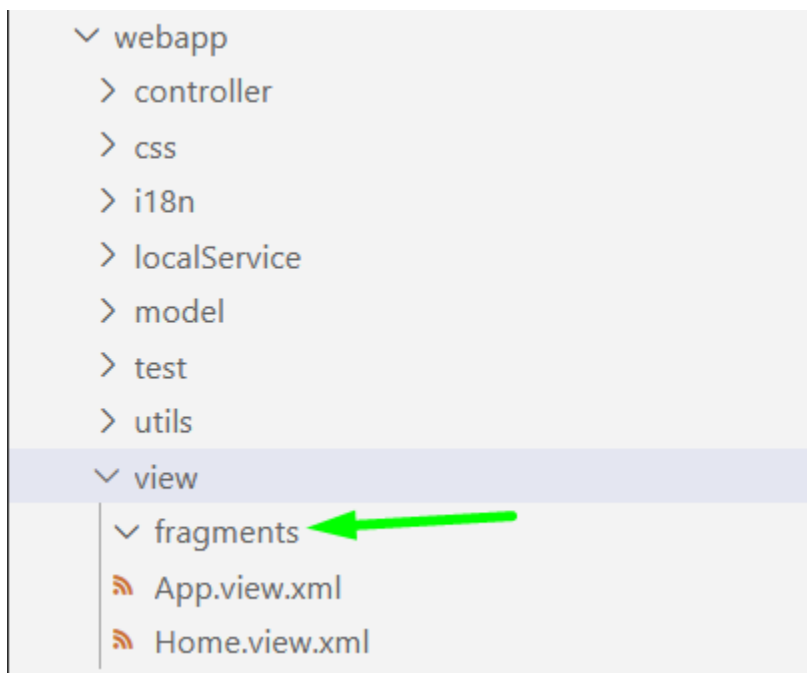
Un Fragment en SAPUI5 es una unidad reutilizable de interfaz de usuario que se utiliza para crear componentes de UI de manera modular. Los **fragments** permiten definir partes de una vista (view) que pueden ser reutilizadas en diferentes vistas o en el mismo contexto sin necesidad de duplicar código.

Características de los Fragments:

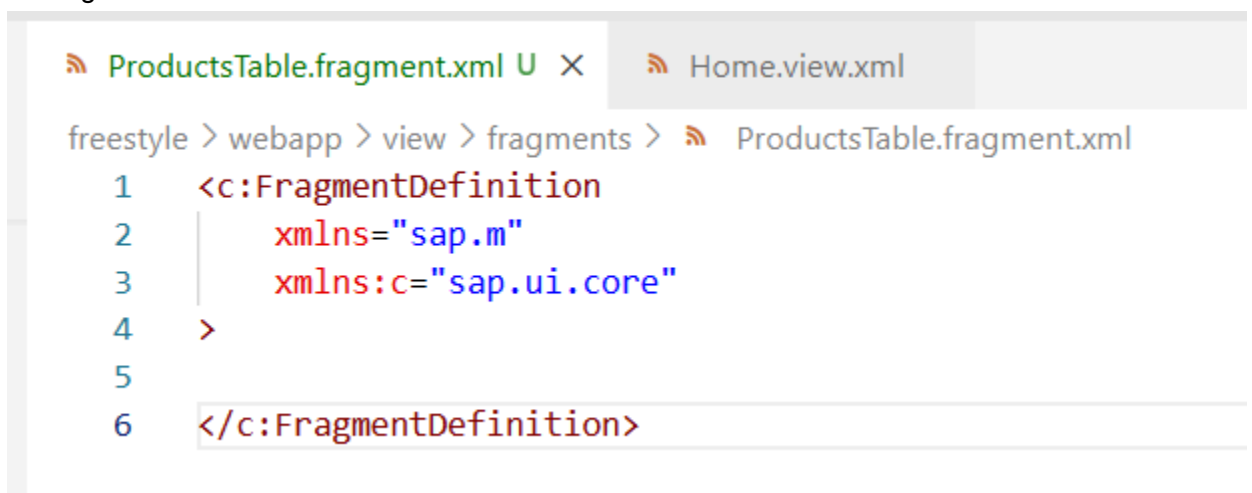
- **Reutilización:** Los fragments pueden ser utilizados en múltiples vistas, lo que ayuda a mantener el código limpio y DRY (Don't Repeat Yourself).
- **No Tienen Controlador Propio:** A diferencia de las vistas, los **fragments** no tienen su propio controlador. En su lugar, utilizan el controlador de la vista que los incluye.
- **Definición en XML o HTML:** Pueden ser definidos en archivos XML o HTML, lo que permite una fácil integración con el resto de la aplicación.

Apliquemos a nuestro proyecto:

Creamos una carpeta llamada **fragments** en la ruta **webapp/view/fragments**



- Creamos un nuevo archivo **ProductsTable.fragment.xml** y agregamos la definición inicial del fragment así:



JavaScript

```
<c:FragmentDefinition
  xmlns="sap.m"
  xmlns:c="sap.ui.core"
>

</c:FragmentDefinition>
```

- Ahora copiamos todo el contenido de la definición de la tabla de la vista Home al **fragment** que acabamos de definir:

JavaScript

```
<c:FragmentDefinition
  xmlns="sap.m"
  xmlns:c="sap.ui.core"
>

  <Table id="idProductsTable"
    inset="false"
    items="{
      path: 'ProductCollection/'
    }">
    <columns>
      <Column id="ID_01">
        <Text id="ID_T01" text="{i18n>Product}" />
      </Column>
      <Column id="ID_02">
        <Text id="ID_T05" text="{i18n>UnitPrice}" />
      </Column>
      <Column id="ID_05">
        <Text id="ID_T04" text="{i18n>UnitsInStock}" />
      </Column>
      <Column id="ID_03">
        <Text id="ID_T02" text="{i18n>QuantityPerUnit}" />
      </Column>
    </columns>
    <items>
      <ColumnListItem id="ID_06" vAlign="Middle">
        <cells>
          <ObjectIdentifier id="ID_07"
            title="{ProductCollection>ProductID}"
            text="{ProductCollection>ProductName}" />
          <ObjectNumber id="ID_11"
            number="{
              parts:[{path:'ProductCollection>UnitPrice'},{path:''}],
              type: 'sap.ui.model.type.Currency',
              formatOptions: {showMeasure: true}
            }"
            unit="USD" />
          <Text id="ID_09"
```

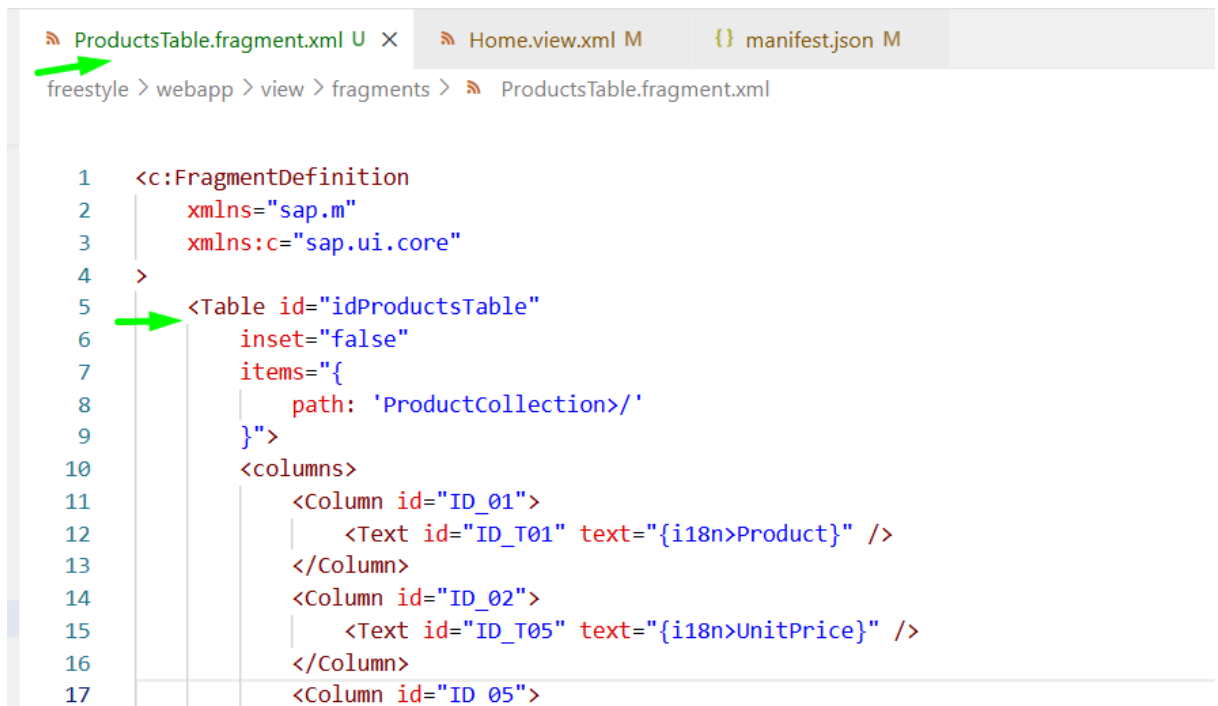


```

        text="{ProductCollection>UnitsInStock}" />
    <Text id="ID_08"
        text="{ProductCollection>QuantityPerUnit}" />
    </cells>
</ColumnListItem>
</items>
</Table>
</c:FragmentDefinition>

```

La definición del Fragment queda ya con toda la adición de la tabla [sap.ui.table](#)



```

1  <c:FragmentDefinition
2      xmlns="sap.m"
3      xmlns:c="sap.ui.core"
4  >
5      <Table id="idProductsTable"
6          inset="false"
7          items="{
8              path: 'ProductCollection>'
9          }">
10         <columns>
11             <Column id="ID_01">
12                 <Text id="ID_T01" text="{i18n>Product}" />
13             </Column>
14             <Column id="ID_02">
15                 <Text id="ID_T05" text="{i18n>UnitPrice}" />
16             </Column>
17             <Column id="ID_05">

```

Ahora vamos modificar nuestra vista **Home** para que se cargue el fragment que definimos así:

freestyle > webapp > view > Home.view.xml

You, 41 minutes ago | 1 author (You)

```

1 <mvc:View controllerName="com.bootcamp.sapui5.freestyle.controller.Home"
2   xmlns:mvc="sap.ui.core.mvc"
3   xmlns:c="sap.ui.core"
4   xmlns="sap.m">
5   <Page id="page" title="{i18n>title}">
6
7       <VBox id="vTest">
8           <Text id="textPrueba" text="{i18n>textPruebai18n}"></Text>
9           <Button id="btnBuscar" text="{i18n>textBuscar}" press="onPress" />
10       </VBox>
11
12       <VBox id="vTest2">
13           <c:Fragment
14             fragmentName="com.bootcamp.sapui5.freestyle.view.fragments.ProductsTable"
15             type="XML"
16           />
17       </VBox>
18   </Page>
19 </mvc:View>

```

You, 3 weeks ago • 1 Hello World

JavaScript

```

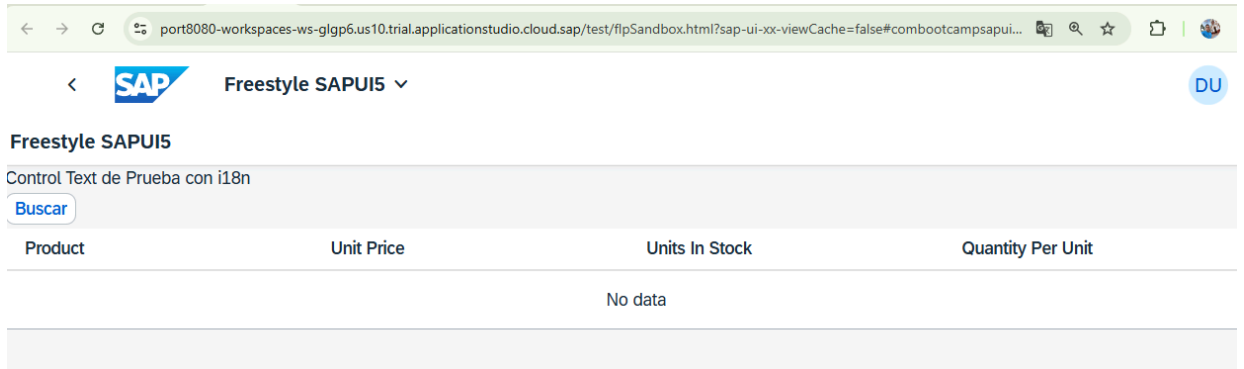
<mvc:View controllerName="com.bootcamp.sapui5.freestyle.controller.Home"
  xmlns:mvc="sap.ui.core.mvc"
  xmlns:c="sap.ui.core"
  xmlns="sap.m">
  <Page id="page" title="{i18n>title}">

    <VBox id="vTest">
      <Text id="textPrueba" text="{i18n>textPruebai18n}"></Text>
      <Button id="btnBuscar" text="{i18n>textBuscar}"
press="onPress" />
    </VBox>

    <VBox id="vTest2">
      <c:Fragment
        fragmentName="com.bootcamp.sapui5.freestyle.view.fragments.ProductsTable"
        type="XML"
      />
    </VBox>
  </Page>
</mvc:View>

```

Al ejecutar la aplicación se presenta de la misma manera al usuario:



The screenshot shows a web browser window with the URL: `port8080-workspaces-ws-glgp6.us10.trial.applicationstudio.cloud.sap/test/f?pSandbox.html?sap-ui-xx-viewCache=false#combootcampsapui...`. The browser's address bar and navigation icons are visible at the top. Below the browser, the application header features the SAP logo, the text "Freestyle SAPUI5" with a dropdown arrow, and a user profile icon labeled "DU". The main content area is titled "Freestyle SAPUI5" and contains a subtitle "Control Text de Prueba con i18n". Below this is a search bar with the placeholder text "Buscar". A table with four columns is displayed: "Product", "Unit Price", "Units In Stock", and "Quantity Per Unit". The table is currently empty, showing "No data" in the center. The bottom of the screenshot shows a light gray footer area.

Freestyle SAPUI5

Control Text de Prueba con i18n

Buscar

Product	Unit Price	Units In Stock	Quantity Per Unit
No data			