



PUBLIC

# **Screen Personas for Mobile RF Devices in Warehouse Management in SAP S/4HANA Cloud**

## **How-To Guide**

## DOCUMENT HISTORY

Document Version	Date	Description
1.0	2022.02.08	First official release of this guide
1.1	2022.08.31	Added details about working in design mode
1.2	2023.07.26	Updated after major Screen Personas changes and for SAP S/4HANA Cloud 2308

# TABLE OF CONTENTS

1	BUSINESS SCENARIO.....	4
1.1	Introduction.....	4
2	OVERVIEW .....	5
2.1	Radio Frequency in Warehouse Management .....	5
2.2	UI Adaptation with Screen Personas .....	5
3	PREREQUISITES.....	6
4	STEP BY STEP PROCEDURES.....	7
4.1	Changing the Background Color.....	9
4.2	Changing the Font Size or Color .....	10
4.3	Hiding Fields and Menu Buttons .....	11
4.3.1	<i>Hiding a Field .....</i>	<i>11</i>
4.3.2	<i>Hiding a Menu Button.....</i>	<i>12</i>
4.4	Changing Button Size and Moving Buttons .....	13
4.5	Assigning Different Flavors to Different Users or User Groups.....	15
4.6	Simulating a Device in a Browser .....	15
4.7	Scripting .....	17
4.7.1	<i>Example 1 – Dynamically Hiding Verification Field on Destination Screen.....</i>	<i>17</i>
4.7.2	<i>Example 2 – Adding ODO Number to Picking Source Screen .....</i>	<i>19</i>
4.7.3	<i>Example 3 – Querying HU by Alternative Identification .....</i>	<i>21</i>
4.8	Templates for Adapting UIs for Classic Applications.....	25
4.8.1	<i>Example – Creating Header and Footer Template .....</i>	<i>25</i>
5	RF UI ADAPTATION FEATURES IN WAREHOUSE MANAGEMENT.....	29
5.1	RF Screens in Detail .....	29
5.1.1	<i>Changes to Menu Items.....</i>	<i>31</i>
5.1.2	<i>Changes to Buttons.....</i>	<i>32</i>
5.2	RF Design Mode.....	33
5.3	Creating Flavors – Tips & Tricks .....	34
5.4	Example Video .....	35
6	USEFUL LINKS.....	35

## 1 BUSINESS SCENARIO

### 1.1 Introduction

SAP GUI for HTML for browser-based radio frequency (RF) devices was introduced in SAP S/4HANA Cloud 2108. SAP GUI for HTML renders the ABAP Dynpros created in the RF framework directly in the browser on the RF device. With SAP GUI for HTML, you can also use Screen Personas to adapt RF screens starting with SAP S/4HANA Cloud 2202. This guide describes how you can set up UI adaptations, called flavors, for mobile RF devices working on SAP GUI for HTML.

## 2 OVERVIEW

### 2.1 Radio Frequency in Warehouse Management

The radio frequency (RF) framework is the standard solution for mobile devices that allows you to display an SAP application on GUI-based devices and browser-based devices using SAP GUI for HTML. The RF framework supports your warehouse in the following ways:

- Decouples business logic from the physical presentation of application data on a selected presentation device
- Supports a large variety of device sizes, device types, and data entry types
- Provides appropriate forms of data presentation according to application data, device capabilities, and user preferences
- Calls appropriate services according to resource inputs (such as verification/input data, keystroke, or logon request)

For more information, see [Working with Mobile RF Devices](#).

### 2.2 UI Adaptation with Screen Personas

You can use Screen Personas to transform classic Dynpro applications according to your specific business needs. Business analysts or key users can create simplified versions of screens called flavors. Flavors work on top of the underlying transactions and a single transaction can have multiple flavors to accommodate the needs of different user groups. The intuitive editor provides powerful features that allow flavor builders to simplify screens by hiding unneeded controls, rearranging controls with the drag and drop editor, and adjusting formatting and styles. The integrated scripting engine allows flavor builders to automate keystrokes to eliminate manual user interactions and thus increasing end-user productivity. An administrator manages the flavors by defining technical details and assigning flavors to users.

See chapter *RF UI Adaptation Features in Warehouse Management* for flavor recommendations in the RF context.

For more information about Screen Personas, see [Screen Personas Overview](#).

### 3 PREREQUISITES

- Warehouse Management in SAP S/4HANA Cloud 2308 (or later) with activated scope item 63V (*Mobile RF Devices in Warehousing*) and its prerequisites (see [Mobile RF Devices in Warehousing \(63V\)](#))
- Screen Personas with its user roles and set-up (see [Adapting UIs for Classic Applications](#) and [Administer UI Adaptations with Screen Personas](#))

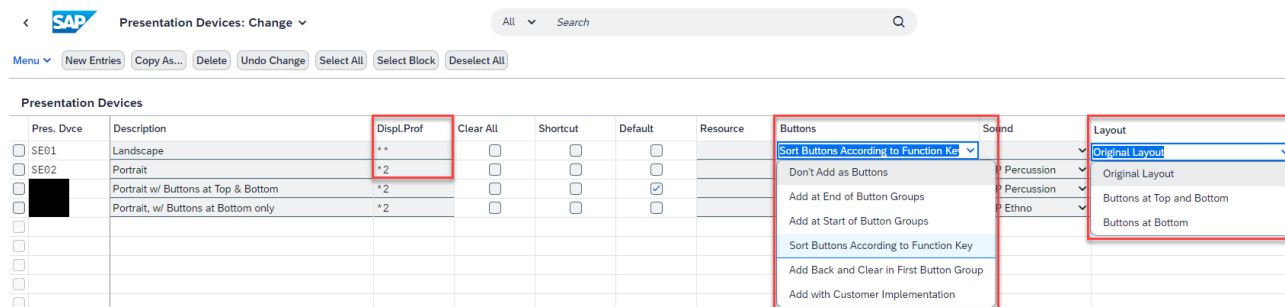
## 4 STEP BY STEP PROCEDURES

This chapter describes common steps to adapt radio frequency (RF) transactions using Screen Personas. For restrictions on adapting UIs in the context of the RF framework, see chapter *RF UI Adaptation Features in Warehouse Management*.

### Note

Select the right layout for your RF device before you start adapting the RF screens. In the *Maintain Presentation Devices* app, you can create a presentation device that suits your requirements, that is, the presentation device:

- Has the relevant page orientation (by using the relevant display profile)
- Sorts buttons and function key usage (by using the relevant buttons option)
- Has space for bigger buttons (by using the relevant layout option)



If you have different presentation devices according to the above-mentioned options, create several flavors - one for each presentation device.

If you create a flavor using a presentation device different than the one you use later, the flavor may not fit or work as expected.

1. Start SAP GUI for HTML on a desktop PC to launch the SAP GUI for HTML service.

We recommend using a Google Chrome browser.

2. Log onto the RF environment with your RF user profile.

On the SAP Fiori launchpad, open the *Test RF Environment* app.

You can ignore the header and footer area as they will be hidden automatically when logging onto the RF environment following the right URL (see [SAP Note 3048632](#) and chapter *Simulating a Device in a Browser*):

Whse No.:

Resource:

Dflt Dvc.:

Enter

3. Enter the warehouse number, resource, and presentation device name:

Whse No.:

Resource:

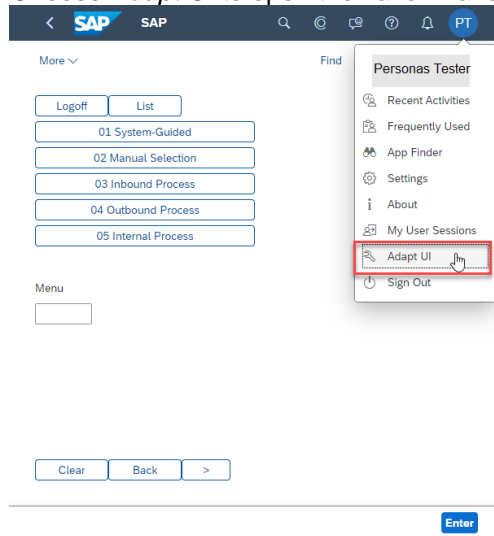
Dflt Dvc.:

The default presentation device you enter determines which screens are used when working in the RF environment.

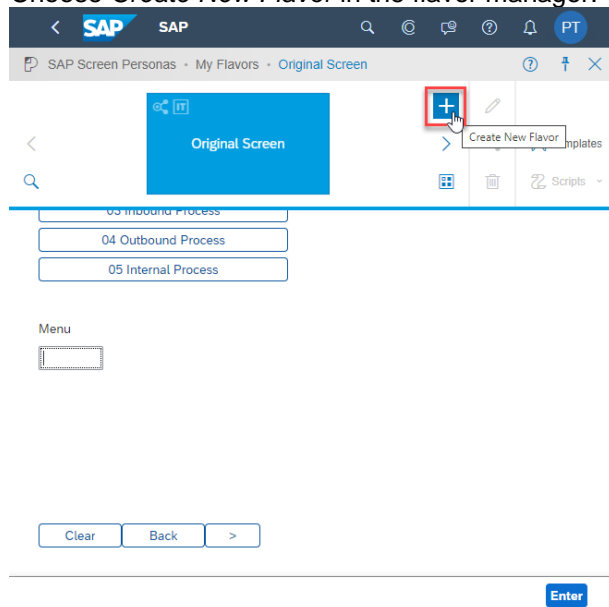
In Warehouse Management, you can display RF screens in portrait or landscape mode. The presentation profile setting for the device defines the screen format. For example, profile \*\* - standard screens in landscape mode, profile \*2 - standard screens in portrait mode.

#### 4. Create a flavor for the RF environment.

##### a. Choose *Adapt UI* to open the flavor manager:

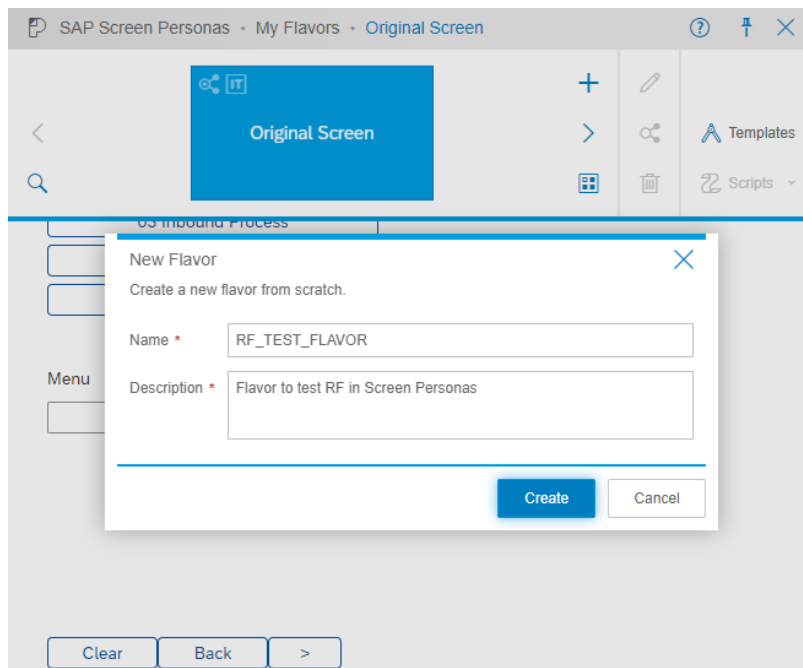


##### b. Choose *Create New Flavor* in the flavor manager:

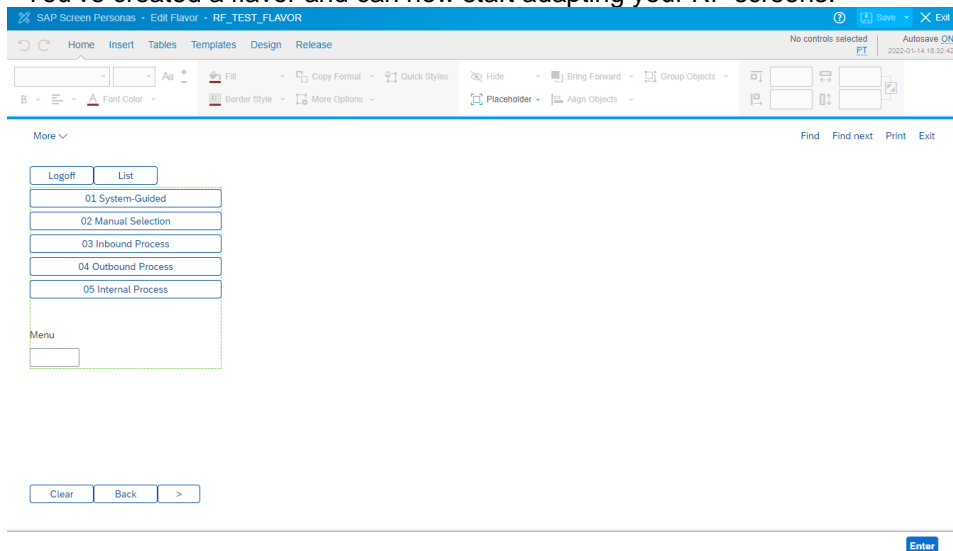


##### c. Enter a name and description for the flavor:



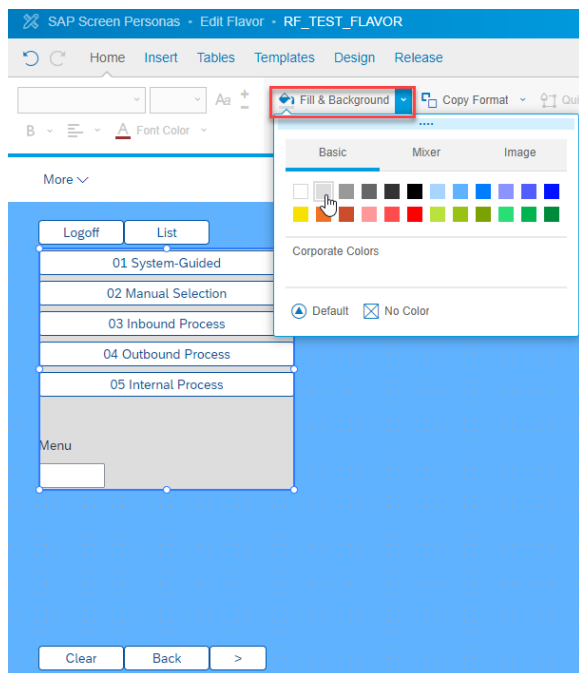
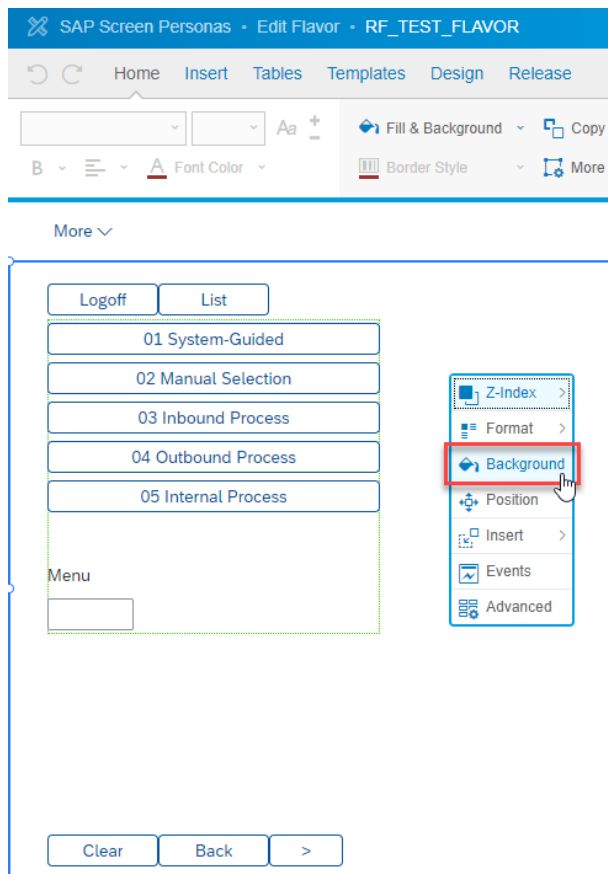


You've created a flavor and can now start adapting your RF screens:



## 4.1 Changing the Background Color

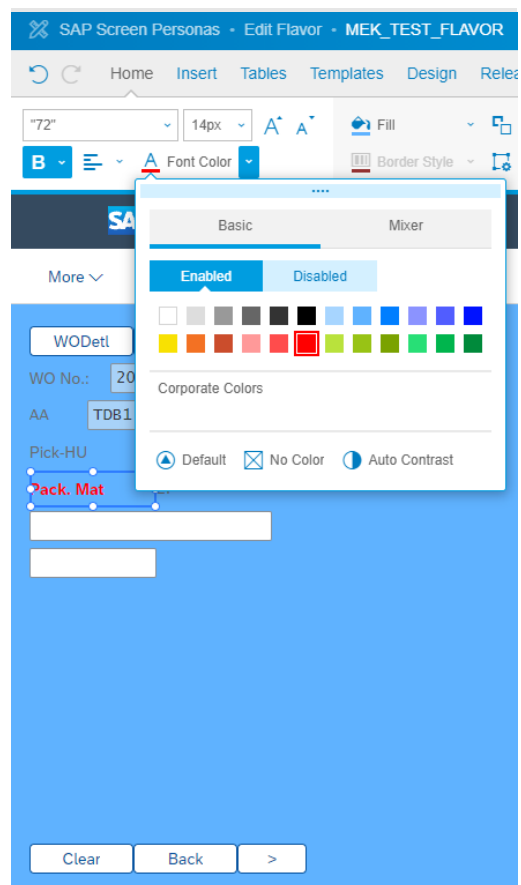
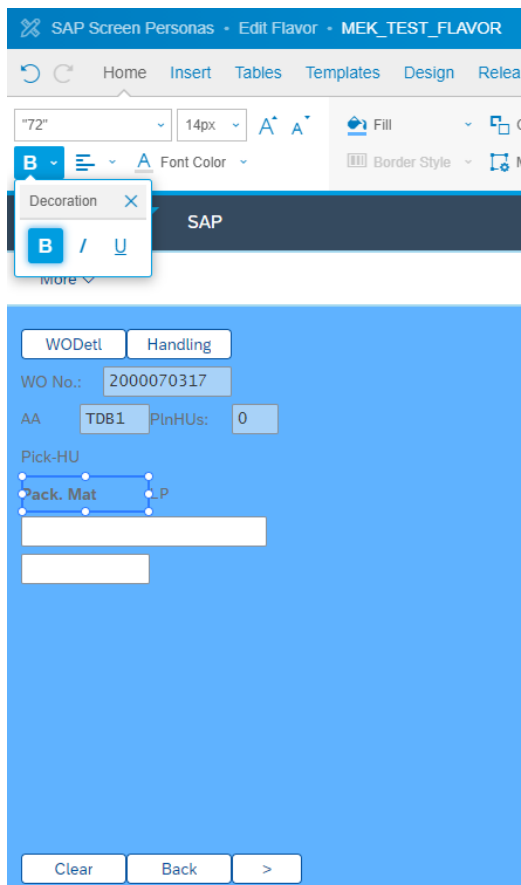
1. Select the screen element for the background.
2. Choose *Fill & Background* or right click and choose *Background*.
3. Choose a color from the palette:



If you don't get the expected result, check chapter *RF UI Adaptation Features in Warehouse Management* for restrictions in the RF context.

## 4.2 Changing the Font Size or Color

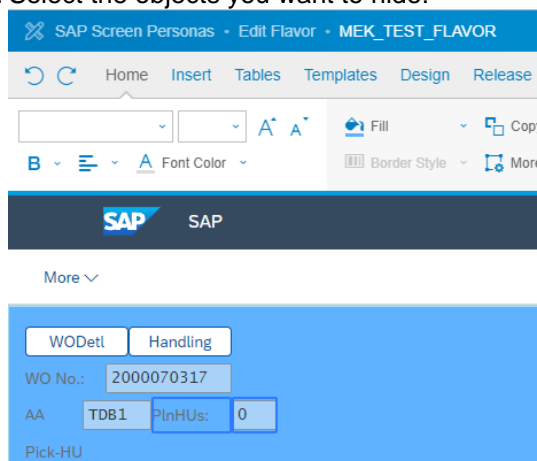
1. Select the element whose font size or color you want to change.
2. Choose *Decoration* to change the font to bold, italics, or underlined. Choose *Color* to change the font color:



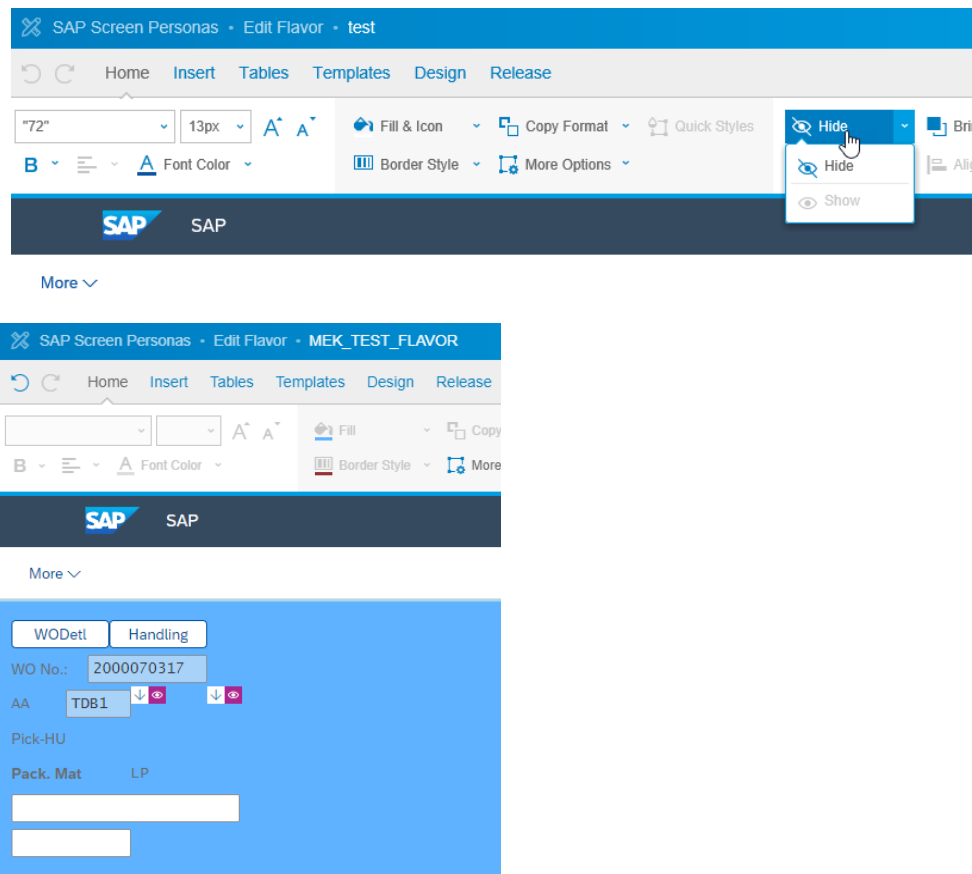
## 4.3 Hiding Fields and Menu Buttons

### 4.3.1 Hiding a Field

1. Select the objects you want to hide:



2. Choose *Hide*:

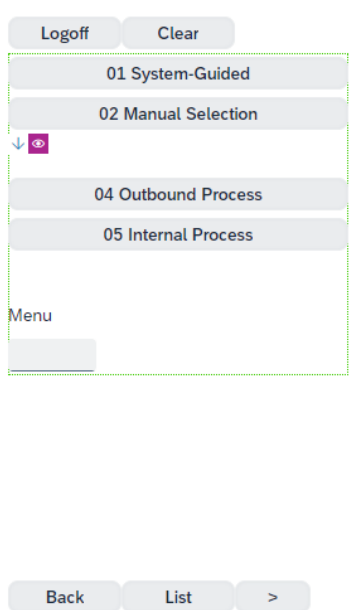


### 4.3.2 Hiding a Menu Button

By default, all available RF transactions are part of the RF menu. If you want to limit access to some RF transactions for RF users, you can hide single menu buttons.

#### Note

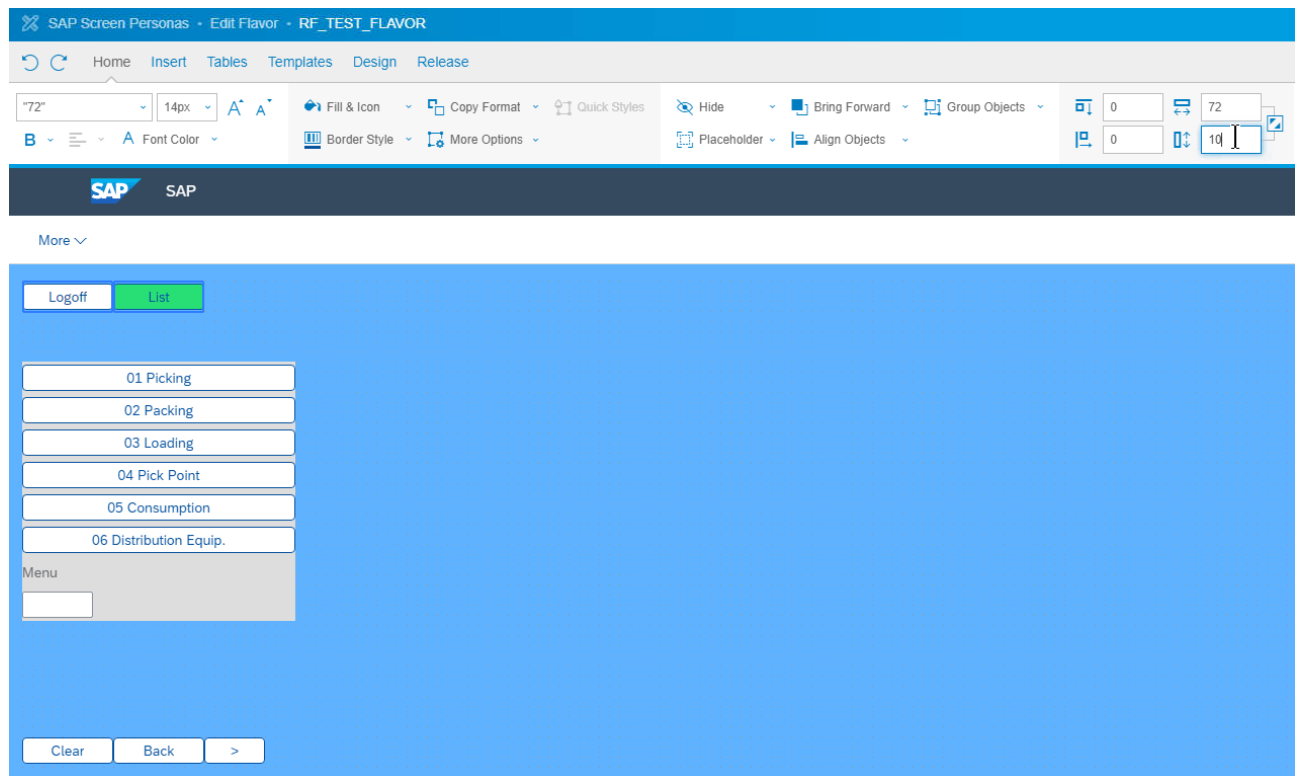
The remaining visible menu buttons are not automatically rearranged, and shortcut numbers are not automatically renumbered. Users can still access hidden RF transactions by entering the hidden shortcut number in the *Menu* field. You can avoid this by hiding the *Menu* field:



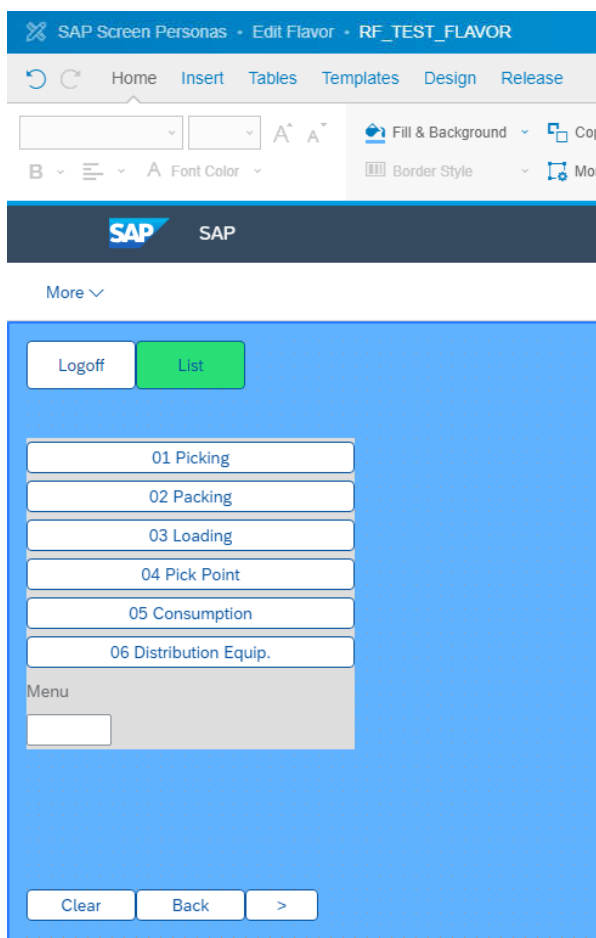
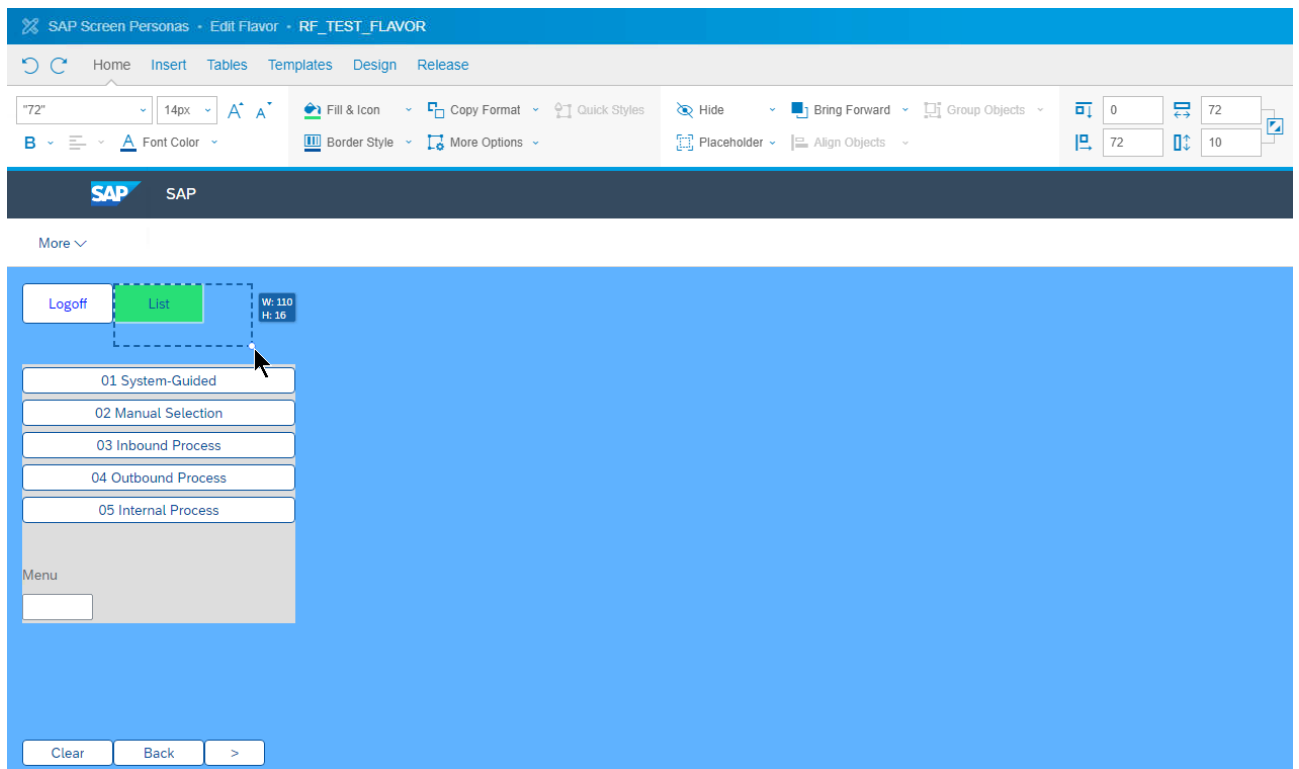
## 4.4 Changing Button Size and Moving Buttons

- 4.5** You can change a single button in the layout area of an RF screen. For general changes to buttons in the layout area, we recommend using templates that you create with the template editor as part of adapting UIs with Screen Personas. For more information about templates, see chapter *Templates for Adapting UIs for Classic Applications* **Templates for Adapting UIs for Classic Applications**

1. Select the button whose size you want to change.
2. Move the element borders or use the size palette:



3. Drag and drop the button to move it to where you want it.



For more information about buttons, see chapter *RF UI Adaptation Features in Warehouse Management*.

## 4.6 Assigning Different Flavors to Different Users or User Groups

You can assign a flavor indirectly to users using the *Manage Flavors* app. Flavors are grouped by assignment category and the assignment categories are then used in the mapping between user groups and business roles. For more information, see the [Assignment Categories](#) section of the [Administration Guide](#) for Screen Personas.

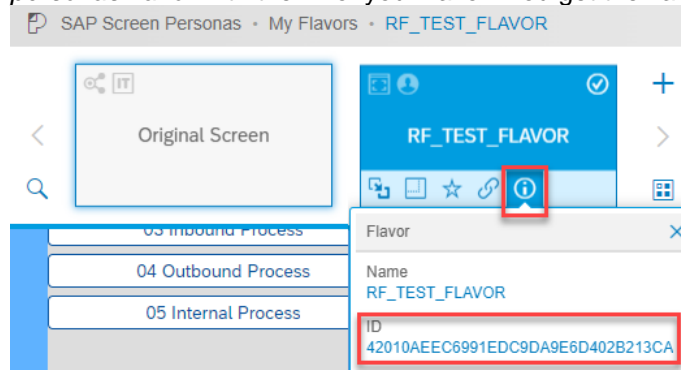
## 4.7 Simulating a Device in a Browser

After creating a flavor, you can check the result by simulating a device with the browser's developer tools. The steps below describe how to do this for the Google Chrome browser.

1. Call a URL that uses the following pattern:

```
https://<server>/sap/bc/gui/sap/its/ewm_mobgui?~transaction=/scwm/rfui&sap-language=EN
```

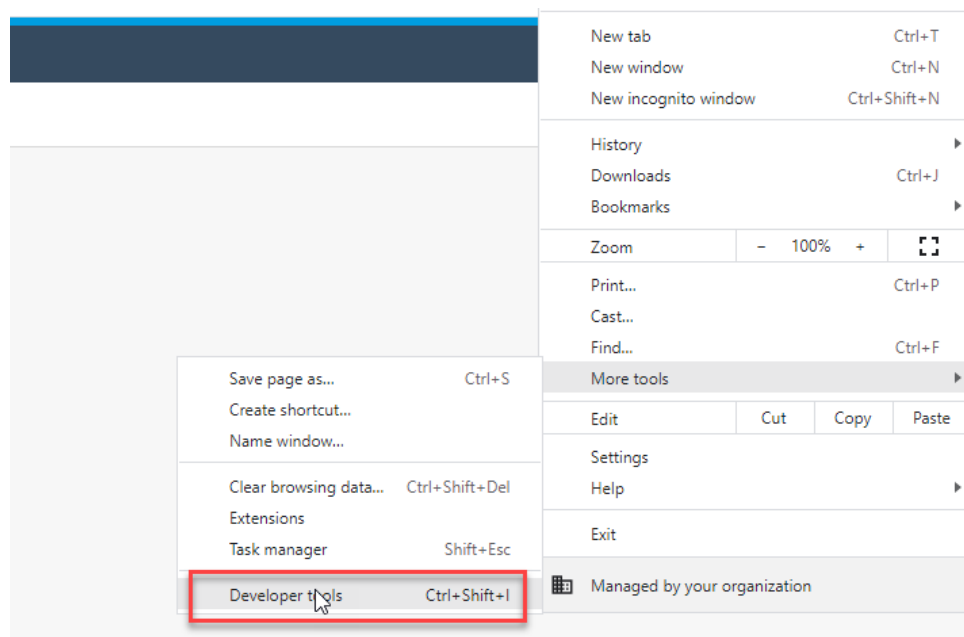
Note that you can add your flavor from the previous steps by using the additional URL parameter *sap-personas-flavor* with the ID of your flavor. You get the flavor's ID in the flavor manager:



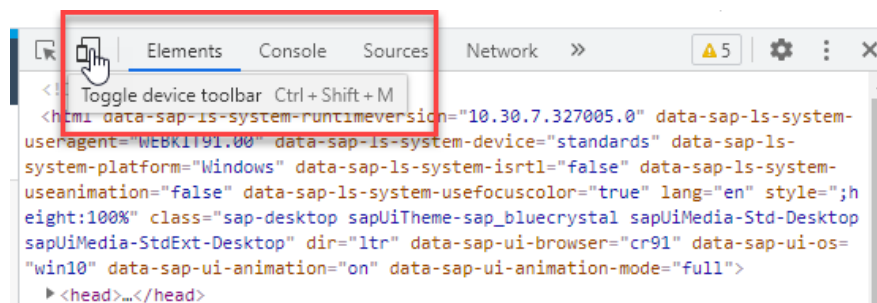
Example of URL with flavor ID added:

```
https://<server>/sap/bc/gui/sap/its/ewm_mobgui?~transaction=/scwm/rfui&sap-language=EN&sap-personas-flavor=42010AEEC6991EDC9DA9E6D402B213CA
```

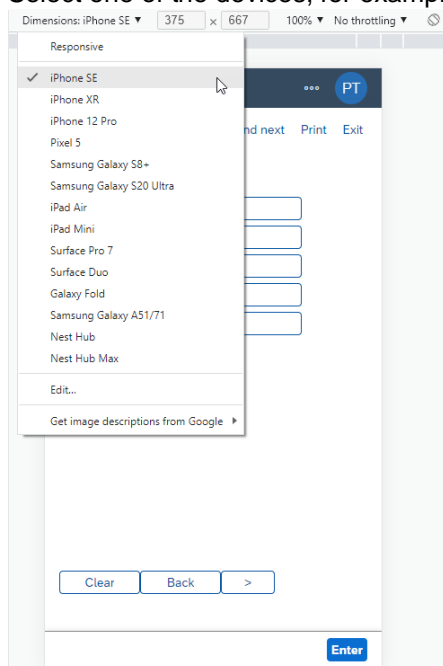
2. Log onto the RF with your RF user profile.
3. Open the developer tools using the Google Chrome menu in the browser window and selecting *More Tools > Developer Tools*.  
Alternatively, you can press Ctrl + Shift + I.



4. Toggle to the device toolbar by selecting a corresponding button or pressing Ctrl + Shift + M:



5. Select one of the devices, for example, iPhone SE:





## 4.8 Scripting

You can use scripting to automate or manipulate screen actions, for example:

- Screens can be skipped assuming mandatory fields can be filled automatically by the script
- Fields can be prefilled
- Additional data can be retrieved and displayed

Scripts created as part of adapting UI features using Screen Personas are small pieces of JavaScript. You can attach these scripts to individual UI elements with script events or entire screens with screen events. For more information, see [Script Editor](#).

### Note

Scripting can have some unwanted side effects, for example:

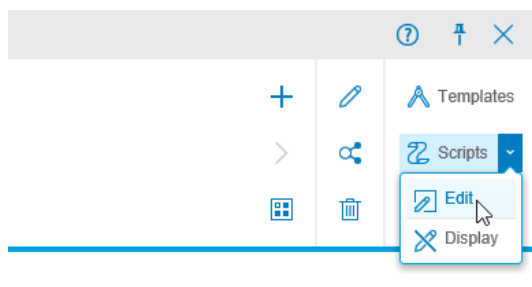
- Depending on what you do in JavaScript and how complex the flavors become, system performance can be affected significantly.  
Therefore, we recommend that you evaluate every script carefully before productive usage, that is, evaluate the expected costs (runtime) and compare it with the benefit for end users. Keep in mind that usually every millisecond counts when working on RF devices. If you want to perform complex steps using scripting, the added runtime must be acceptable to end users.  
For example, if you change the color of a button using scripting, at runtime there's a visible delay in the button changing its color.
- Any changes to a screen definition, that is, the objects/elements on the screen, might have a negative impact on your script.  
Depending on the impact, you may need to revise the script.
- If you use object-related content in your script, for example, the text/label of a button for a query, the corresponding coding might be language-dependent.  
That is, if the script checks for the label text, you must consider different languages used in your warehouses.

The scripting examples below are added for illustration purposes only. They do not claim to be perfect JavaScript code. Additionally, they focus on specific aspects and don't cover screen processing in a holistic way (for example, complete error handling).

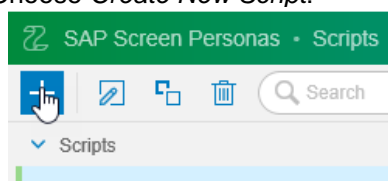
### 4.8.1 Example 1 – Dynamically Hiding Verification Field on Destination Screen

This example describes how to use a script to dynamically hide the quantity fields on the destination screen of a product warehouse task in RF picking. Besides hiding the fields, the script also automatically verifies the quantity on the destination screen. This assumes that the quantity was already verified on the source screen.

1. Start the script editor from the flavor manager by choosing *Scripts > Edit*.



2. Choose *Create New Script*.



### 3. Add the following coding:

```

wnd[0]/scriptPersonas_506B4BC345F41EEC9CC52F6AEFF4EEA7
Execute Start Recording Format script Aa
1 // An OnLoad or OnAfterRefresh script was executed.
2
3 if (session.idExists("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/1b1/SCWM/S_RF_ORDIM_CONFIRM-NISTA_VERIF")) {
4     session.findById("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/txt/SCWM/S_RF_ORDIM_CONFIRM-NISTA_VERIF").text = session.findById("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/txt/SCWM/S_RF_ORDIM_CONFIRM-NISTA_CHR").text;
5     session.findById("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/1b1/SCWM/S_RF_ORDIM_CONFIRM-NISTA_VERIF").hide();
6     session.findById("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/txt/SCWM/S_RF_ORDIM_CONFIRM-NISTA_VERIF").hide();
7     session.findById("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/txt/SCWM/S_RF_ORDIM_CONFIRM-ALTME").hide();
8 }

```

```

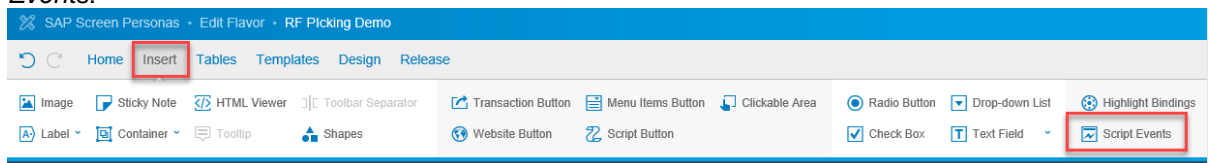
// OnAfterRefresh

if (session.idExists("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/1b1/SCWM/S_RF_ORDIM_CONFIRM-NISTA_VERIF")) {
    session.findById("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/txt/SCWM/S_RF_ORDIM_CONFIRM-NISTA_VERIF").text = session.findById("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/txt/SCWM/S_RF_ORDIM_CONFIRM-NISTA_CHR").text;
    session.findById("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/1b1/SCWM/S_RF_ORDIM_CONFIRM-NISTA_VERIF").hide();
    session.findById("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/txt/SCWM/S_RF_ORDIM_CONFIRM-NISTA_VERIF").hide();
    session.findById("wnd[0]/usr/subX:/SCWM/SAPLRF_PICKING_PM:0302/txt/SCWM/S_RF_ORDIM_CONFIRM-ALTME").hide();
}

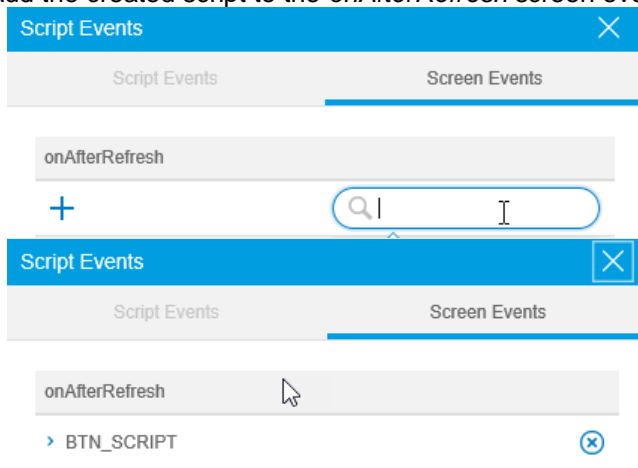
```

### 4. Save the implementation.

### 5. Assign the script to the screen event by editing the flavor, choosing the *Insert* tab, and choosing *Script Events*:



### 6. Add the created script to the *onAfterRefresh* screen event:



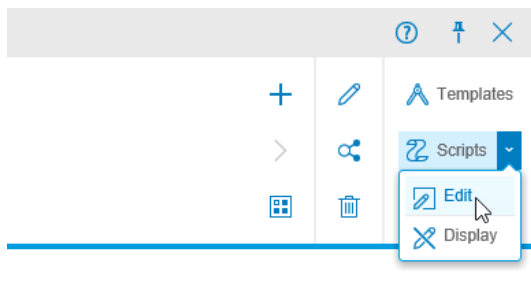
When the script runs, the quantity fields won't be visible on the destination screen for product warehouse tasks in RF picking:

#### 4.8.2 Example 2 – Adding ODO Number to Picking Source Screen

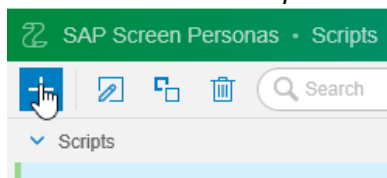
This example describes how to create a script to add the outbound delivery order (ODO) number to the RF picking source data screen. This can be done although the ODO number is not available as standard on the source data screen.

Note that you must add a custom field and label to the screen (see [txtPersonas\\_168726056454688](#) below). For your script, you will need to use the ID of your custom field.

1. Start the script editor from the flavor manager by choosing *Scripts > Edit*.



2. Choose *Create New Script*.



3. Name the script and create the following implementation in the coding window:

```

1  if (session.idExists("wnd[0]/usr/txtPersonas_168726056454688")) {
2      //start query to get ODO number the WT is linked to
3      session.findById('wnd[0]/usr/btn/SCWM/S_RF_SCRELM-PB2').press();
4      //query WT by WT
5      session.findById('wnd[0]/usr/subX:/SCWM/SAPLRF_INQUIRY_PM:0400/txt/SCWM/S_RF_INQ_SEL2-SELNO').text = '6';
6      session.findById('wnd[0]').sendVKey(0);
7      session.findById('wnd[0]').sendVKey(0);
8      //go to detail
9      session.findById('wnd[0]/usr/btn/SCWM/S_RF_SCRELM-PB1').press();
10     //get ODO number
11     var ODO = session
12         .findById('wnd[0]/usr/subX:/SCWM/SAPLRF_INQUIRY_PM:0102/txt/SCWM/S_RF_INQ_TO-RDOCID')
13         .text;
14     // go back to original screen
15     session.findById('wnd[0]').sendVKey(7);
16     session.findById('wnd[0]').sendVKey(7);
17     session.findById('wnd[0]').sendVKey(7);
18     //set ODO in custom field on the picking WT source screen
19     session.findById("wnd[0]/usr/txtPersonas_168726056454688").text = ODO;
20 }

```

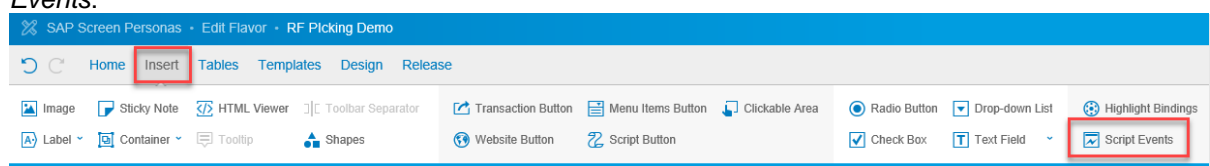
```

if (session.idExists("wnd[0]/usr/txtPersonas_168726056454688")) {
    //start query to get ODO number the WT is linked to
    session.findById('wnd[0]/usr/btn/SCWM/S_RF_SCRELM-PB2').press();
    //query WT by WT
    session.findById('wnd[0]/usr/subX:/SCWM/SAPLRF_INQUIRY_PM:0400/txt/SCWM/S_RF_INQ_SEL2-SELNO').text = '6';
    session.findById('wnd[0]').sendVKey(0);
    session.findById('wnd[0]').sendVKey(0);
    //go to detail
    session.findById('wnd[0]/usr/btn/SCWM/S_RF_SCRELM-PB1').press();
    //get ODO number
    var ODO = session
        .findById('wnd[0]/usr/subX:/SCWM/SAPLRF_INQUIRY_PM:0102/txt/SCWM/S_RF_INQ_TO-RDOCID')
        .text;
    // go back to original screen
    session.findById('wnd[0]').sendVKey(7);
    session.findById('wnd[0]').sendVKey(7);
    session.findById('wnd[0]').sendVKey(7);
    //set ODO in custom field on the picking WT source screen
    session.findById("wnd[0]/usr/txtPersonas_168726056454688").text = ODO;
}

```

4. Save the implementation.

5. Assign the script to the screen event by editing the flavor, choosing the *Insert* tab, and choosing *Script Events*:



6. Add the created script to the *onAfterRefresh* screen event:

The flavor containing the script uses RF functionality to execute a warehouse task query to get the ODO number linked to the warehouse task and displays it on the RF picking source screen:

Original Layout	Changed Layout
<p>Detail Queries</p> <p>SrcBin <input type="text" value="S001-01-01"/></p> <p>Prod. <input type="text" value="TG11"/> <input type="text" value="Trad.Good 11,PD,Reg.Trading"/></p> <p>5 <input type="text" value=""/> ActQty: <input type="text" value=""/> PC</p> <p>DestHU <input type="text" value=""/></p> <p>HUWd WTLst &gt; <input type="text" value=""/></p>	<p>Detail Queries</p> <p>SrcBin <input type="text" value="S001-01-01"/></p> <p>Prod. <input type="text" value="TG11"/> <input type="text" value="Trad.Good 11,PD,Reg.Trading"/></p> <p>5 <input type="text" value=""/> ActQty: <input type="text" value=""/> PC</p> <p>ODO: <input type="text" value="80006392"/></p> <p>DestHU <input type="text" value=""/></p> <p>HUWd WTLst &gt; <input type="text" value=""/></p>

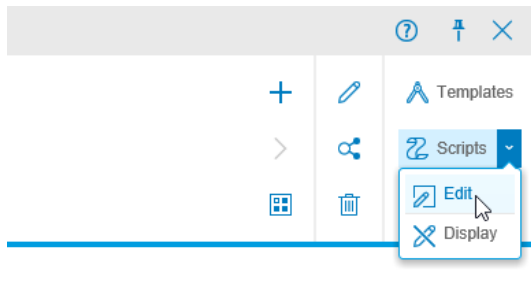
Note that the script performs several steps in the background. This affects runtime that must be acceptable by end users. We recommend that you don't add too many steps.

If the information you need is not provided on an existing RF screen that you can reach out of the current logical transaction (for example, picking), you can use APIs. For more information, see [SAP Business Accelerator Hub](#) and chapter *Example 3 – Querying HU by Alternative Identification*.

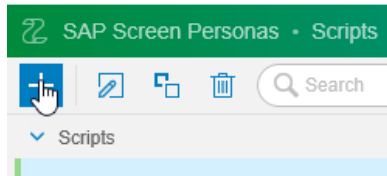
### 4.8.3 Example 3 – Querying HU by Alternative Identification

This example describes how to create a script to select handling unit (HU) information by entering the alternative HU identification instead of the Warehouse Management HU number.

1. Start the script editor from the flavor manager by choosing *Scripts > Edit*.



2. Choose Create New Script:



3. Name the script and create the following implementation in the coding window:

```

1  if (session.idExists('wnd[0]/usr/txtPersonas_168387506699773')) {
2      if (session.findById('wnd[0]/usr/txtPersonas_168387506699773').text != '') {
3          var altID = session.findById('wnd[0]/usr/txtPersonas_168387506699773').text;
4          session.utils.log(altID);
5
6          //Create the request object
7          var http = new XMLHttpRequest();
8
9          //Specify the type of request, the url and if the request should be handled asynchronously or not
10         http.open(
11             'GET',
12             "https://[REDACTED]/sap/opu/odata4/sap/api_handlingunit/srvd_a2x/sap/handlingunit/0001/
HandlingUnitAlternativeID?$filter=Warehouse eq '1050' and EWMHndlgUnitAltID eq '' +
13                 altID +
14                 '',
15             false
16         );
17
18         //set request header attributes
19         http.setRequestHeader('DataServiceVersion', '2.0');
20         http.setRequestHeader('Accept', 'application/json');
21
22         //Send the request
23         http.send();
24
25         //Process the response
26         if (http.readyState == 4 && http.status == 200) {
27             //The response is available in the http.response property
28             session.utils.log(http.response);
29             try {
30                 var HU = JSON.parse(http.response).value[0].HandlingUnitExternalID;
31                 session.utils.log(HU);
32                 session.findById('wnd[0]/usr/subX:/SCWM/SAPLRF_INQUIRY_PM:0200/txt/SCWM/S_RF_SELECTION-RFHU').text = HU;
33             } catch (error) {
34                 session.utils.log('HU number could not be determined based on alternative HU ID ' + altID);
35                 session.utils.alert('HU number could not be determined based on alternative HU ID ' + altID);
36             }
37         } else {
38             session.utils.log('OData API query failed');
39             session.utils.alert('OData API query failed');
40         }
41
42         session.utils.log('API call finished');
43     } else {
44         session.utils.log('API NOT called');
45     }
46 }

```

```

if (session.idExists('wnd[0]/usr/txtPersonas_168387506699773')) {
    if (session.findById('wnd[0]/usr/txtPersonas_168387506699773').text != '') {
        var altID = session.findById('wnd[0]/usr/txtPersonas_168387506699773').text;
        session.utils.log(altID);

        //Create the request object
        var http = new XMLHttpRequest();

        //Specify the type of request, the url and if the request should be handled
        asynchronously or not
        http.open(
            'GET',
            "https://xxx-
x.xxx.xxx.xxx/sap/opu/odata4/sap/api_handlingunit/srvc_a2x/sap/handlingunit/0001/HandlingUn
itAlternativeID?$filter=Warehouse eq '1050' and EWMHndlgUnitAltvID eq '' +
            altID +
            '",
            false
        );

        //set request header attributes
        http.setRequestHeader('DataServiceVersion', '2.0');
        http.setRequestHeader('Accept', 'application/json');

        //Send the request
        http.send();

        //Process the response
        if (http.readyState == 4 && http.status == 200) {
            //The response is available in the http.response property
            session.utils.log(http.response);
            try {
                var HU = JSON.parse(http.response).value[0].HandlingUnitExternalID;
                session.utils.log(HU);
                session.findById('wnd[0]/usr/subX:/SCWM/SAPLRF_INQUIRY_PM:0200/txt/SCWM/S_RF
_SELECTION-RFHU').text = HU;
            } catch (error) {
                session.utils.log('HU number could not be determined based on alternative HU
ID ' + altID);
                session.utils.alert('HU number could not be determined based on alternative
HU ID ' + altID);
            }
        } else {
            session.utils.log('OData API query failed');
            session.utils.alert('Odata API query failed');
        }

        session.utils.log('API call finished');
    } else {
        session.utils.log('API NOT called');
    }
}

```

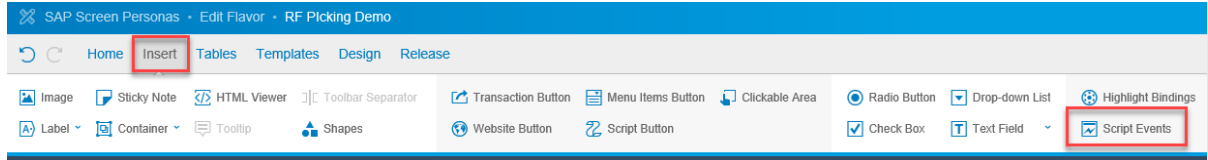
```

}
}

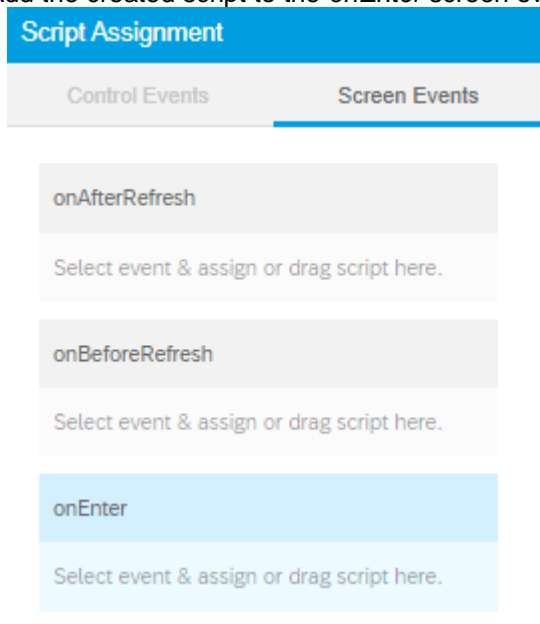
```

4. Save the implementation.

5. Assign the script to the screen event by editing the flavor, choosing the *Insert* tab, and choosing *Script Events*:



6. Add the created script to the *onEnter* screen event:



7. Create a second script with the following implementation:

```

1 //reset original field
2 if (session.idExists('wnd[0]/usr/subX:/SCWM/SAPLRF_INQUIRY_PM:0200/txt/SCWM/S_RF_SELECTION-RFHU') && session.idExists('wnd[0]/usr/txtPersonas_168387506699773')) {
3     session.findById('wnd[0]/usr/subX:/SCWM/SAPLRF_INQUIRY_PM:0200/txt/SCWM/S_RF_SELECTION-RFHU').text = '';
4 }
5
6 //set focus to alternative HU ID field
7 if (session.idExists('wnd[0]/usr/txtPersonas_168387506699773')) {
8     session.findById('wnd[0]/usr/txtPersonas_168387506699773').setFocus();
9 }
10

```

```

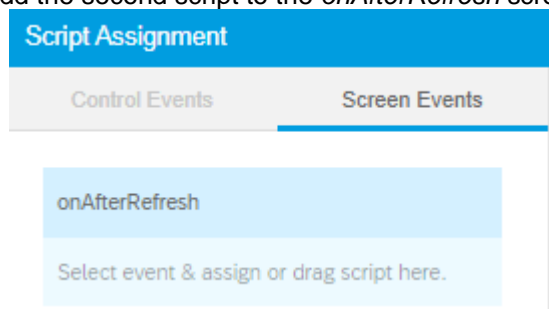
//reset original field
if (session.idExists('wnd[0]/usr/subX:/SCWM/SAPLRF_INQUIRY_PM:0200/txt/SCWM/S_RF_SELECTION-RFHU')
&& session.idExists('wnd[0]/usr/txtPersonas_168387506699773')) {
    session.findById('wnd[0]/usr/subX:/SCWM/SAPLRF_INQUIRY_PM:0200/txt/SCWM/S_RF_SELECTION-
RFHU').text = '';
}
//set focus to alternative HU ID field
if (session.idExists('wnd[0]/usr/txtPersonas_168387506699773')) {
    session.findById('wnd[0]/usr/txtPersonas_168387506699773').setFocus();
}

```



Note that the Screen Personas text element IDs (here `txtPersonas_168387506699773`) must fit to your flavor.

8. Add the second script to the `onAfterRefresh` screen event:



The flavor containing the scripts changes the selection screen of RF transaction *HU Query*:

Original Layout	Changed Layout
<div> <div>ClearBack</div> <div>           HU  <input type="text"/> </div> <div>Message</div> </div>	<div> <div>ClearBack</div> <div>           Alternative HU Identifier  <input type="text"/> </div> <div>Message</div> </div>

The flavor hides the original field label and input field of the HU number and instead uses the field label and input field of the alternative HU identifier. You must add these custom fields to the screen as part of your flavor (see above).

The script for event `onAfterRefresh` sets the focus to the new input field.

The script for event `onEnter` uses an OData service to read the HU data based on the entered alternative HU identifier. For more information, see [SAP Business Accelerator Hub](#).

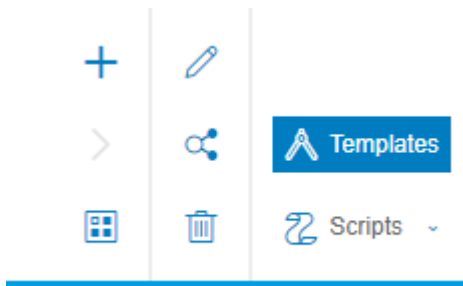
## 4.9 Templates for Adapting UIs for Classic Applications

Templates that you create with the template editor as part of adapting UIs allow you to use predefined design patterns, for example, the header and footer section for the RF screens.

### 4.9.1 Example – Creating Header and Footer Template

Create the Template:

1. Start the template editor from the flavor manager by choosing *Templates*:

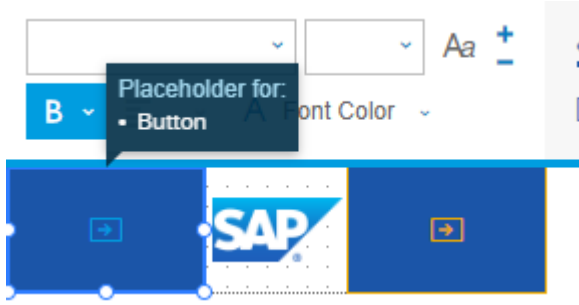


2. Choose *Create New*:



3. Enter a name, group, and description and choose *Create*:

4. Insert a *Placeholder for Button*, an *Icon*, and set additional attributes such as background color and size:



5. Save the template.

6. Create another template for the buttons in the footer section.

Use the Template in a Flavor:

#### Note

To be able to add the template with its bigger size, you must select a presentation device with a RF layout containing empty lines. For more information about RF layouts, see chapter *RF UI Adaptation Features in Warehouse Management*.

1. In the flavor editor, select *Templates -> Insert Template*:

[Insert Template](#)
[Edit Template](#)

[Create Template](#)

Menu ▾

Logoff

Clear

01 System-Guided

02 Manual Selection

2. Select the previously created template and move it aside for further processing:

Logoff Clear

01 System-Guided

02 Manual Selection

03 Inbound Process

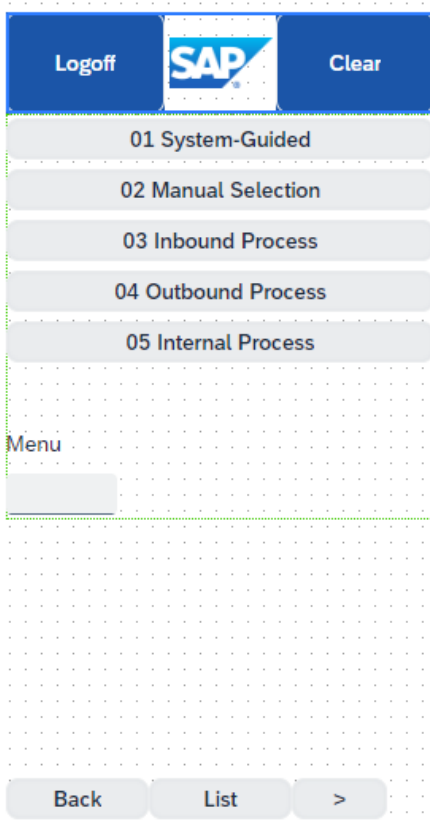
04 Outbound Process

05 Internal Process

Menu

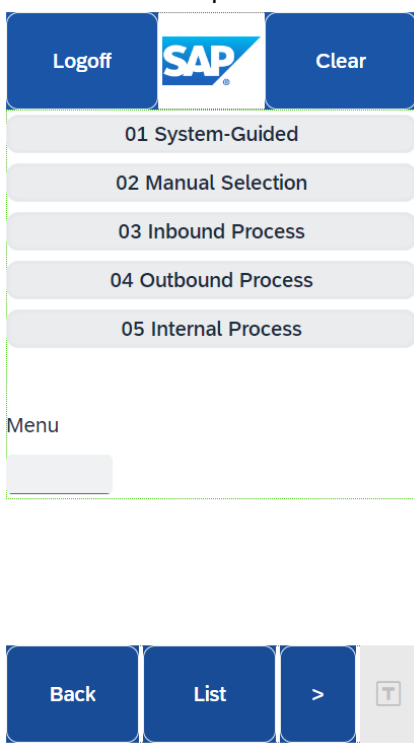
Back List >

3. Drag and drop the buttons onto the template and move it back to the header position:



Note: Make sure that the template fits onto the RF screen without moving the subscreen area.

4. Repeat the steps for the footer template, that is, drag and drop the buttons onto the template and move it back to the footer position:



5. Save the flavor.

For more information about buttons and the correct RF layout, see chapter *RF UI Adaptation Features in Warehouse Management*.

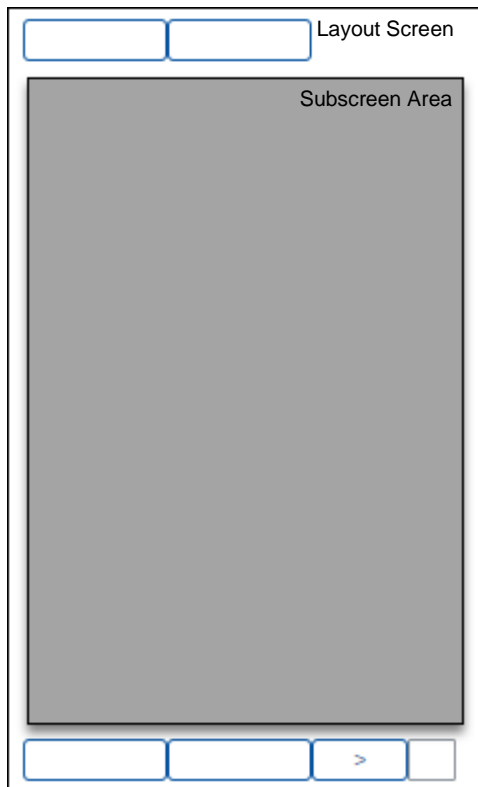
## 5 RF UI ADAPTATION FEATURES IN WAREHOUSE MANAGEMENT

### 5.1 RF Screens in Detail

When adapting UIs for classic applications using Screen Personas together with radio frequency (RF) screens, it's important to pay attention to the ID of the screen element that you're adjusting.

RF screens have two main parts:

- Layout screen with buttons – this screen contains buttons with the element IDs /SCWM/S\_RF\_SCRELM-PB1 through to /SCWM/S\_RF\_SCRELM-PB4.
- Subscreen area with dynamic content – the subscreen area is called based on the maintained screen and program for the current RF step in the RF Customizing or the menu subscreen.



#### Layout screen (White area)

- Contains buttons
- Contains subscreen area
- Portrait mode – 15 rows and 26 columns
- Landscape mode – 8 rows and 40 columns

#### Subscreen area (Grey area)

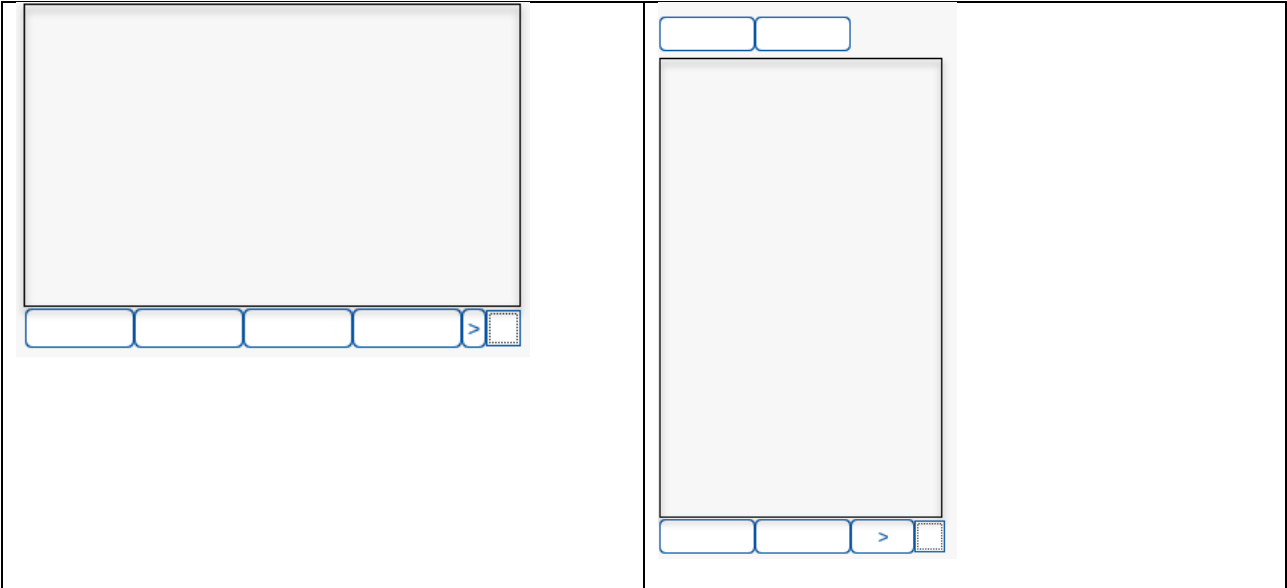
- Contains dynamic content
- Portrait mode – 13 rows and 26 columns
- Landscape mode – 7 rows and 40 columns

If you adjust an element in the subscreen area, the changes are applied only for the transaction that uses this subscreen area. If you adjust a button of a layout screen, the changes are applied to all RF transactions that contain this button. The UI adaptation logic uses the function code that's assigned to the button as the ID, not the button name. This assigns a unique ID to a button that allows you to, for example, give the *BACK* button a unique layout. This is also valid for buttons on a subscreen, such as the menu buttons.

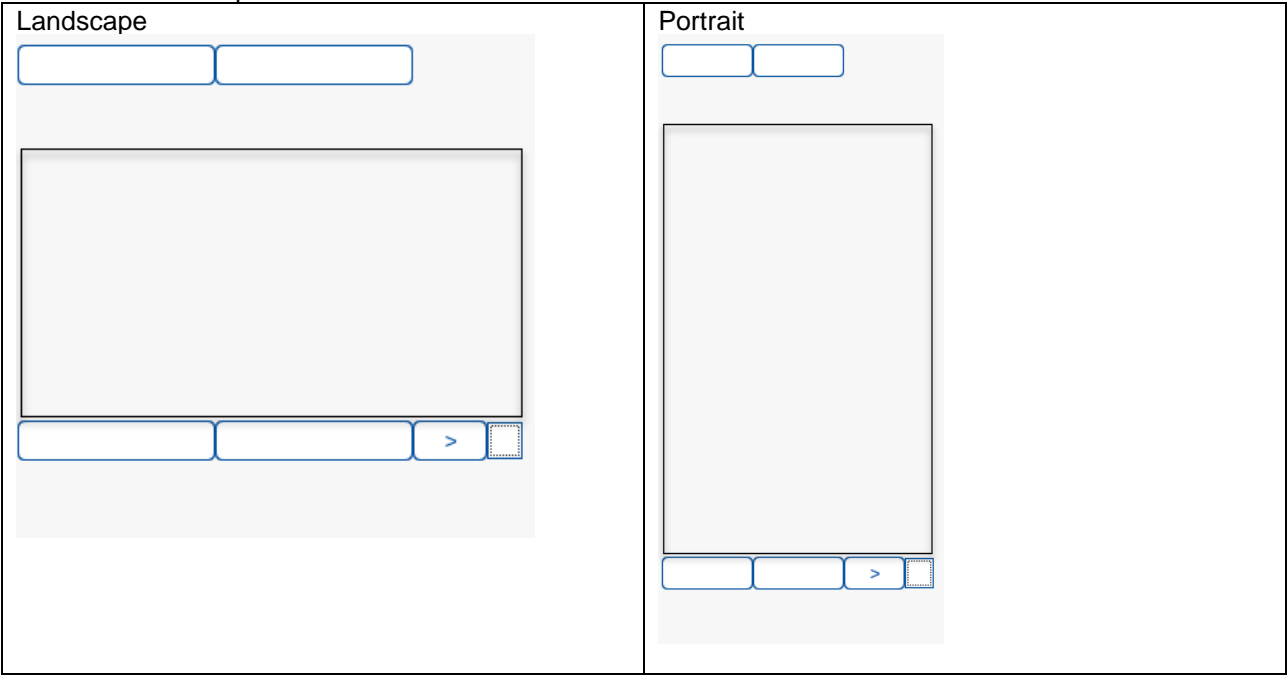
**Recommendation:** Before adapting UIs using Screen Personas, select the screen layout that fits best to your planned changes. You select the layout in the *Maintain Presentation Devices* app in Warehouse Management.

- Original Layout

Landscape	Portrait
-----------	----------

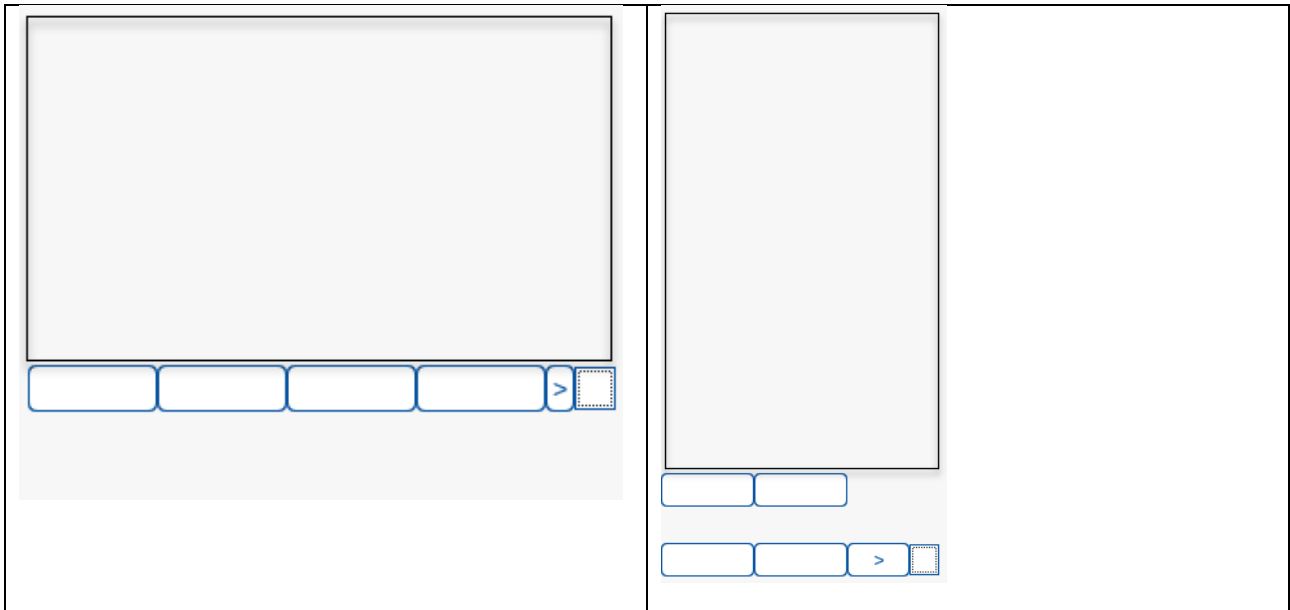


• Buttons at Top and Bottom



• Buttons at Bottom

Landscape	Portrait
-----------	----------



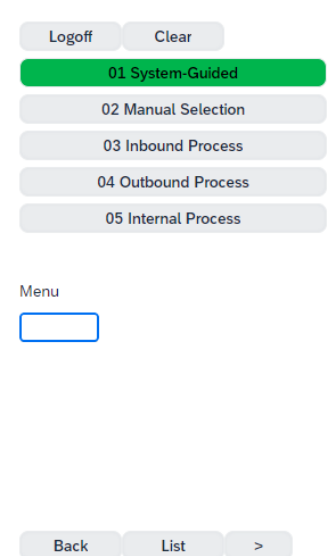
The additional lines make it much easier to enlarge the button size in Screen Personas.

Note that the UI adaptations you make with Screen Personas on the new layout screen will only be visible if you use a presentation device linked to this layout screen. Using another layout screen may corrupt the flavor and changes to the layout area will not be visible.

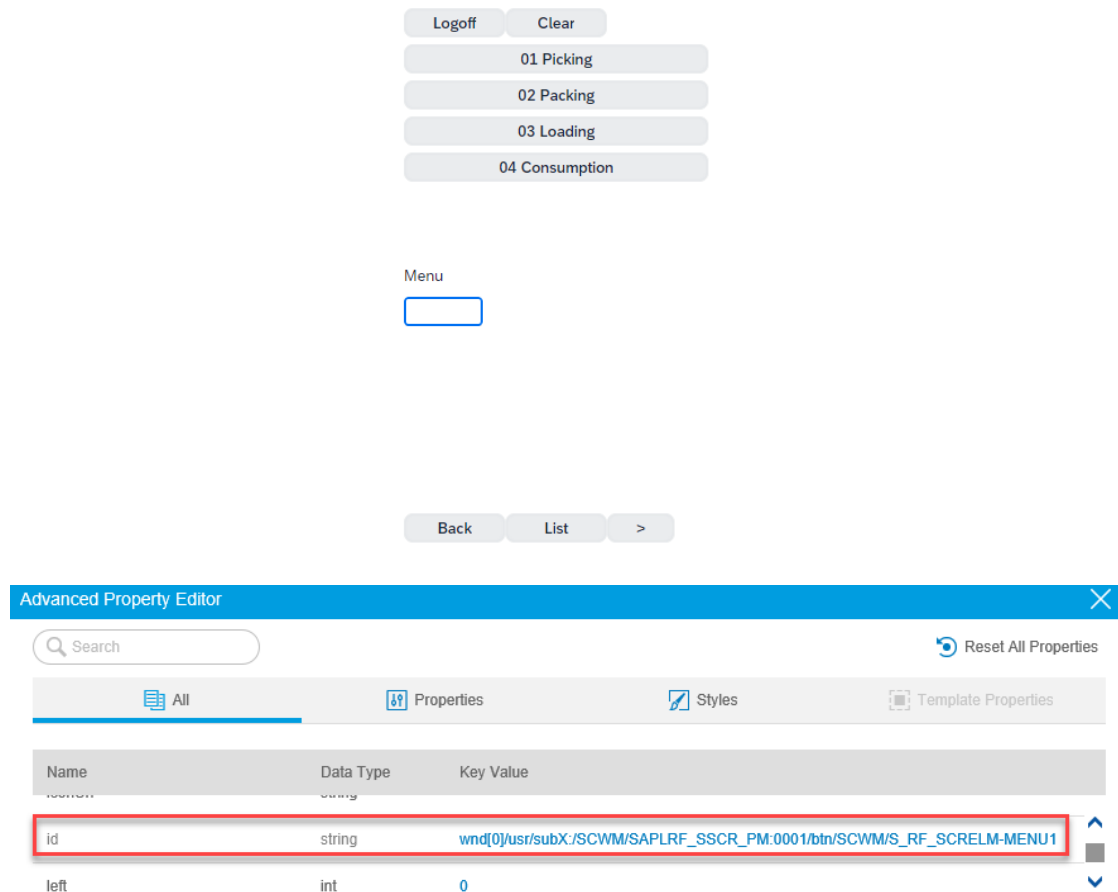
### 5.1.1 Changes to Menu Items

When applying changes to an RF menu item, for example, *System-Guided*, the changes are applied only to the single menu item and NOT to all menu items with the same button ID, for example, *Picking*. As you can see in the advanced properties of a screen element, the IDs of RF menu items share the same root as the element name on the dynpro, for example, `/SCWM/S_RF_SCRELM-MENU1`. For the Screen Personas change list, the menu ID from the RF framework (for example, `OBMAIN` for *Outbound Process*) is used to identify a menu entry uniquely.

The following screenshots show an example of coloring one specific menu item based on its unique ID. Here, the first menu item is green:



In the following screenshot, although *01 Picking* shares the same dynpro ID (`/SCWM/S_RF_SCRELM-MENU1`), it's not green because Screen Personas uses the RF menu ID for unique identification (`SYGUID` vs. `PIMAIN`):



### 5.1.2 Changes to Buttons

When applying changes for a button on the layout screen (for example, *BACK*), the changes are only applied to the button that has the same function code, independent of where the button is located (see screenshots below).

As you can see in the advanced properties of a screen element, the buttons share the same technical ID, which is the element name on the dynpro, for example, `/SCWM/S_RF_SCRELM-PB2`. For the change list that was created when adapting the UI features, the function code ID from the RF framework (for example, *BACK*) is taken.



Logoff
Clear

01 System-Guided
02 Manual Selection
03 Inbound Process
04 Outbound Process
05 Internal Process

Menu

Back
List
>

Clear
Back

Product

Message

Advanced Property Editor

Search
Reset All Properties

All
Properties
Styles
Template Properties

Name	Data Type	Key Value
id	string	wnd[0]/usr/btn/SCWM/S_RF_SCRELM-PB2

## 5.2 RF Design Mode

Adapting UIs for classic applications with Screen Personas allows you to adjust only screen elements that are displayed on a screen when the screen is being used. If some of the elements or screens are not displayed based on context (for example, serial numbers), you can't adjust these elements. Consequently, if you want to influence serial number fields, you must work with a serialized product. To overcome this limitation, you can use design mode on a desktop to adjust RF screens more easily.

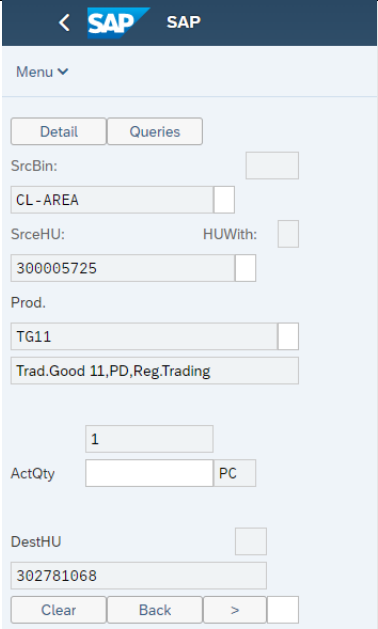
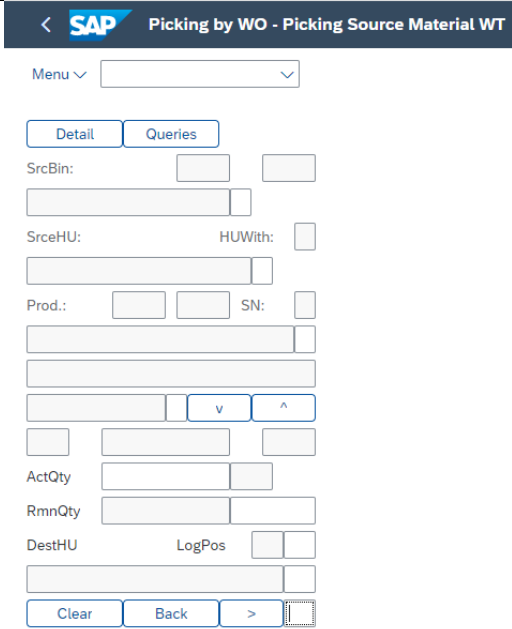
With design mode, you can see all the RF screens in your current presentation format (portrait or landscape) and all possible fields that belong to the RF transaction, without needing to prepare the necessary data upfront.

### Tips

- Start design mode by pressing Ctrl + Shift + F2 from any screen in the RF transaction that you want to adjust.
- Go to the next screen by pressing Ctrl + Shift + F3.
- Return to the previous screen by pressing Ctrl + Shift + F4.

- You can identify the screen you're adjusting by the description of the RF transaction and the RF transaction step in the header, for example, *Picking by WO – Picking Source Material WT*.
- Start adapting UIs for classic applications features by choosing *Adapt UI* to open the flavor manager.
- End design mode by restarting the *Test RF Environment* app.

#### Example: RF Picking Screen

Normal Mode - some fields hidden from RF transaction based on current application data	Design Mode - all fields visible
	

### 5.3 Creating Flavors – Tips & Tricks

- Before creating a flavor, it's important to think about which transactions and screens you want to adapt.
- Warehouse Management provides the RF screens in two presentation formats (portrait and landscape mode). If you use RF devices with only one presentation format, make sure that you enter the right presentation device type at the RF logon screen. If you use RF devices with both presentation formats, you might have to make the Screen Personas change twice.
- When changing the content of the layout screen (see chapter *RF UI Adaptation Features in Warehouse Management*), check all the subscreens that you use. The layout change you make could negatively affect the subscreens. For example, if you increase the size of buttons on the top of the screen, you must rearrange the subscreen area so that it's not hidden by the content of the layout screen. This can be avoided by using the layout screens with predefined empty lines. First, decide on the layout you want to use. Second, use templates to change the button size. For information about templates, see chapter *Templates for Adapting UIs for Classic Applications*.
- SAP recommends preparing necessary test data for the screens that need to be adjusted. Then create a flavor and apply to your RF screens as described in chapter *Step by Step Procedures*.
- Test your RF screens in a browser and then validate the flavor with a real RF device. Distribute the flavors to the relevant target system. For information about transports in relation to adapting UI artifacts, see [Administer UI Adaptations with Screen Personas](#).
- SAP recommends creating one flavor for the complete RF experience. Nevertheless, depending on your needs, it's also possible to create separate flavors, for example, for different user groups (experienced users as opposed to temporary employees) and application areas (inbound, outbound, and internal processes).
- If the screen orientation of your RF device is portrait, we recommend setting the RF logon screen to portrait orientation too. By default, the logon screen is set to landscape orientation when you start the *Test RF Environment* app in a browser. Set the RF logon screen to portrait orientation in one of the following ways:

- In the Google Chrome browser, select a device type with portrait orientation and restart the transaction.
- In the *Maintain Presentation Devices* app, set a presentation device with portrait orientation as the default presentation device (for example, with display profile \*2) and restart the *Test RF Environment* app.

## 5.4 Example Video

In this [video](#), there's a demo of the result of a UI adaptation applied to the RF picking transaction. The video compares the RF picking transaction with and without a custom flavor in SAPGUI for HTML and without any adaptation in ITSmobile.

## 6 USEFUL LINKS

- Screen Personas
  - Specifics for adapting UIs in SAP S/4HANA Cloud with Screen Personas
    - [Screen personas app help](#)
    - [Administration guide](#)
    - [Blog posts](#) on the SAP Community
    - [SAP Note 3322808](#) (*SAP Screen Personas/Adapt UI for Classic Applications - Enhanced Support for Warehouse Management Radio Frequency Applications*)
  - Learn more about Screen Personas for SAP S/4HANA
    - <https://open.sap.com/courses/sps5> (*Adapting the UI in SAP S/4HANA Cloud with SAP Screen Personas*)
    - <https://open.sap.com/courses/sps3> (*Using SAP Screen Personas for Advanced Scenarios*)
    - <https://open.sap.com/courses/sps4> (*Building Mobile Applications with SAP Screen Personas*)
- Warehouse Management
  - [Working with Mobile RF Devices](#)
  - [SAP Note 3048632](#) (*Information and Restrictions for Mobile Data Terminals in Warehouse Management in S/4HANA Cloud or EWM in S/4HANA On Premise*)

[www.sap.com](http://www.sap.com)

