Sage 300 Web Screens

How to use the OptionsMenu Javascript Component

January 2021

Table of Contents

[1. Overview 3](#_Toc61271540)

[2. OptionsMenu HTML Markup 4](#_Toc61271541)

[3. How to include the OptionsMenu component 5](#_Toc61271542)

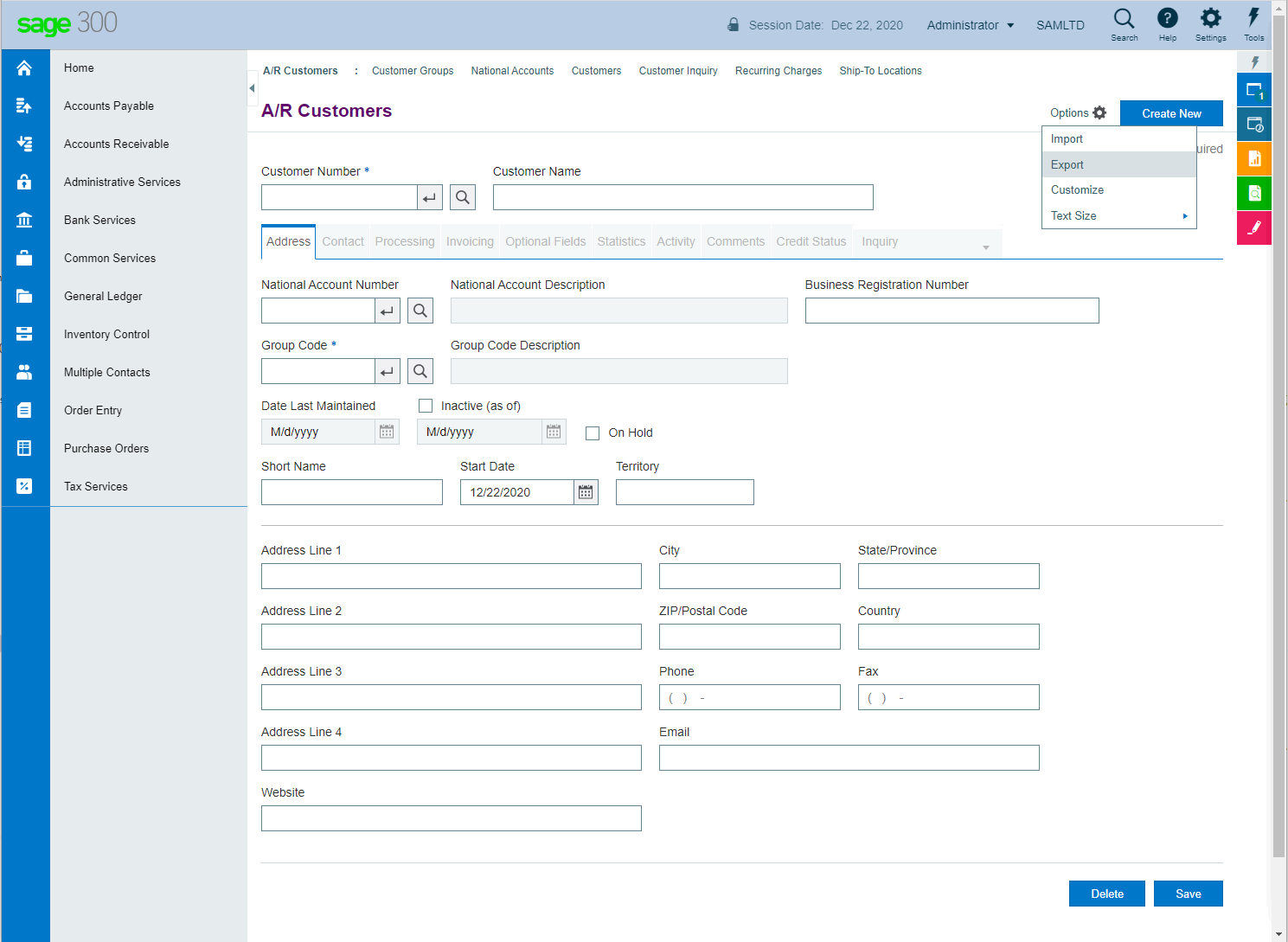
[4. Available Methods in the OptionsMenu Component 6](#_Toc61271543)

[5. Example of how to use the OptionsMenu Javascript Component 8](#_Toc61271544)

[6. Example code with updated Options Menu Screenshot 15](#_Toc61271545)

1. Overview

This document is intended to serve as a guide for illustrating how to use the OptionsMenu Javascript component. This component will allow developers to easily add, remove and otherwise manage the available options in the OptionsMenu. The following image illustrates where the OptionsMenu lives within a typical Web Screens page. The OptionsMenu is located here



1. OptionsMenu HTML Markup

The following is an example of how the OptionsMenu is rendered in HTML markup. The menu options available in this example are as follows:

* Import [**btnOptionImport**]
* Export [**btnOptionExport**]
* Customize UI [**btnCustomizeUI**]
* Text Size [**menuOptionsTextSize**]

Primary Menu Container

<ul class="sub-menu k-group k-menu-group k-popup k-reset" id="**OptionsMenu**" role="menu" data-role="popup">

<li class="k-item k-state-default k-first" role="menuitem" id="**btnOptionImport**"><a class="k-link k-menu-link">Import</a></li>

<li class="k-item k-state-default" role="menuitem" id="**btnOptionExport**"><a class="k-link k-menu-link">Export</a></li>

<li class="k-item k-state-default" role="menuitem" id="**btnCustomizeUI**"><a class="k-link k-menu-link">Customize</a></li>

<li class="menu-with-submenu k-item k-state-default k-last" aria-haspopup="true" role="menuitem">

<a id="btnFormSizeUILevel" class="k-link k-menu-link">”Text Size"<span class="k-icon k-i-arrow-60-right k-menu-expand-arrow"</span></a>

<ul class="sub-menu k-group k-menu-group" id="**menuOptionsTextSize**" role="menu" aria-hidden="true">

<li id="**btnFormSizeUILevelLarge**" class="menu-active k-item k-state-default k-first" role="menuitem">

<a class="k-link k-menu-link" id="labelFormSizeUILevelLarge" onclick="">Large</a>

</li>

<li id="**btnFormSizeUILevelMedium**" class="menu-active k-item k-state-default" role="menuitem">

<a class="k-link k-menu-link" id="labelFormSizeUILevelMedium" onclick="">Medium</a>

</li>

<li id="**btnFormSizeUILevelSmall**" class="menu-active k-item k-state-default k-last" role="menuitem">

<a class="k-link k-menu-link" id="labelFormSizeUILevelSmall" onclick="">Small</a>

</li>

</ul>

</li>

</ul>

Menu Items

SubMenu Items

SubMenu

1. How to include the OptionsMenu component

The OptionsMenu javascript component lives in the following folder and file:

\Web\Sage.CA.SBS.ERP.Sage300.Web\Areas\Shared\Scripts\Sage.CA.SBS.ERP.Sage300.Common.OptionsMenu.js

1. Available Methods in the OptionsMenu Component

The OptionsMenu component currently contains the following methods:

|  |  |  |
| --- | --- | --- |
| Namespace | Method | Description |
| sg.utls.OptionsMenu | getOptionsMenu | Get a reference to the base Options Menu container. |
| sg.utls.OptionsMenu | \_fixCSS | Ensure that the CSS classes k-first and k-last are correctly placed after insertion/deletion of items. Developer Note: This is meant to be called by other methods, not called directly. |
| sg.utls.OptionsMenu | \_fixItemClass | Ensure the list of list items have their classes correctly specified. First item has 'k-first' and last item has 'k-last' |
| sg.utls.OptionsMenu | getTopLevelMenuItemCount | Get a count of top level menu items (Ignores submenu and their items) |
|  |  |  |
| sg.utls.OptionsMenu.MenuItem | add | Add a new item to the bottom of the OptionsMenu |
| sg.utls.OptionsMenu.MenuItem | remove | Remove an existing menu item from the Options Menu |
| sg.utls.OptionsMenu.MenuItem | setLabel | Set the visible label text for an existing menu item |
| sg.utls.OptionsMenu.MenuItem | insertBefore | Insert a new menu item before an existing item |
| sg.utls.OptionsMenu.MenuItem | insertAfter | Insert a new menu item after an existing item |
| sg.utls.OptionsMenu.MenuItem | \_getDefaultTemplate | Build the standard MenuItem template markup |
|  |  |  |
| sg.utls.OptionsMenu.SubMenu | getItemCount | Get a count of the number of items in a SubMenu |
| sg.utls.OptionsMenu.SubMenu | setLabel | Set the visible label text for an existing SubMenu |
| sg.utls.OptionsMenu.SubMenu | add | Add a new SubMenu container and Optional MenuItems to the bottom of the OptionsMenu |
| sg.utls.OptionsMenu.SubMenu | remove | Remove an existing SubMenu and all of it's MenuItems |
| sg.utls.OptionsMenu.SubMenu | insertBefore | Insert a new SubMenu before an existing menu item |
| sg.utls.OptionsMenu.SubMenu | insertAfter | Insert a new SubMenu after an existing menu item |
| sg.utls.OptionsMenu.SubMenu | \_getDefaultTemplate | Build the standard SubMenu template markup |
| sg.utls.OptionsMenu.SubMenu | \_insertItems | Insert new items into an existing SubMenu  **Developer Note**: This method is meant to be called by the public methods   * add * insertBefore * insertAfter |
|  |  |  |
| sg.utls.OptionsMenu.SubMenu.MenuItem | \_getDefaultTemplate | Build the standard SubMenu MenuItem template markup |
| sg.utls.OptionsMenu.SubMenu.MenuItem | add | Add a new MenuItem or MenuItems to an existing SubMenu |
| sg.utls.OptionsMenu.SubMenu.MenuItem | remove | Remove an existing MenuItem from an existing SubMenu |
| sg.utls.OptionsMenu.SubMenu.MenuItem | insertBefore | Insert a new MenuItem or MenuItems into an existing SubMenu before an existing MenuItem |
| sg.utls.OptionsMenu.SubMenu.MenuItem | insertAfter | Insert a new MenuItem or MenuItems into an existing SubMenu after an existing MenuItem |

1. Example of how to use the OptionsMenu Javascript Component
2. Add a new menu item to the Options Menu (without optional href argument)

Note: The component is responsible for finding the OptionsMenu within the DOM. This is the case for ALL methods in this component.

sg.utls.OptionsMenu.MenuItem.add(“NewMenuItemId”, “This is the menu label”);

This is the simplest way to add a new menu item.

1. Remove an existing menu item from the Options Menu

sg.utls.OptionsMenu.MenuItem.remove(“NewMenuItemId”);

1. Set the label for an existing menu item

sg.utls.OptionsMenu.MenuItem.setLabel(“NewMenuItemId”, “This is the new label”);

1. Insert a new menu item BEFORE an existing menu item

var targetItemId = “btnOptionExport”;

var itemId = “NewItemId”;

var label = “This is the label”;

var href = “”;

sg.utls.OptionsMenu.MenuItem.insertBefore(targetItemId, itemId, label, href);

1. Insert a new menu item AFTER an existing menu item

var targetItemId = “btnOptionImport”;

var itemId = “NewItemId”;

var label = “This is the label”;

var href = “”;

sg.utls.OptionsMenu.MenuItem.insertAfter(targetItemId, itemId, label, href);

1. Get the number of items in a SubMenu

sg.utls.OptionsMenu.MenuItem.SubMenu.getItemCount(“SubMenuId”);

1. Set the label for an existing SubMenu

sg.utls.OptionsMenu.SubMenu.setLabel(“SubMenuId”, “This is the new label”);

1. Add a new SubMenu with optional MenuItems to the bottom of the OptionsMenu

var subMenuItems =

[

{ listItemId: "btnItem1", anchorId: "labelItem1", label: "Large", onClick: "" },

{ listItemId: "btnItem2", anchorId: "labelItem2", label: "Medium", onClick: "" },

{ listItemId: "btnItem3", anchorId: "labelItem3", label: "Small", onClick: "" },

];

var subMenuId = "SubMenu1Id";

var subMenuLabelId = "SubMenuLabelId1";

var subMenuLabel = "New Sub Menu";

sg.utls.OptionsMenu.SubMenu.add(subMenuId, subMenuLabelId, subMenuLabel, subMenuItems);

1. Remove an existing SubMenu and all of it’s MenuItems.

sg.utls.OptionsMenu.SubMenu.remove(“SubMenuId”);

1. Insert a new SubMenu BEFORE another SubMenu

var subMenuItems =

[

{ listItemId: "btnNewSubMenu1", anchorId: "labelItem3\_1", label: "Extreme", onClick: "" },

{ listItemId: "btnNewSubMenu2", anchorId: "labelItem3\_2", label: "High", onClick: "" },

{ listItemId: "btnNewSubMenu3", anchorId: "labelItem3\_3", label: "Medium", onClick: "" },

{ listItemId: "btnNewSubMenu4", anchorId: "labelItem3\_4", label: "Low", onClick: "" },

{ listItemId: "btnNewSubMenu5", anchorId: "labelItem3\_5", label: "Poverty", onClick: "" },

];

var targetItemId = "btnOptionImport";

var subMenuId = "SubMenuId";

var subMenuLabelId = "SubMenuLabelId";

var subMenuLabel = "Income Level";

sg.utls.OptionsMenu.SubMenu.insertBefore(targetItemId, subMenuId, subMenuLabelId, subMenuLabel, subMenuItems);

1. Insert a new SubMenu AFTER another SubMenu

var subMenuItems =

[

{ listItemId: "btnNewSubMenu1", anchorId: "labelItem3\_1", label: "Extreme", onClick: "" },

{ listItemId: "btnNewSubMenu2", anchorId: "labelItem3\_2", label: "High", onClick: "" },

{ listItemId: "btnNewSubMenu3", anchorId: "labelItem3\_3", label: "Medium", onClick: "" },

{ listItemId: "btnNewSubMenu4", anchorId: "labelItem3\_4", label: "Low", onClick: "" },

{ listItemId: "btnNewSubMenu5", anchorId: "labelItem3\_5", label: "Poverty", onClick: "" },

];

var targetItemId = "btnOptionImport";

var subMenuId = "SubMenuId";

var subMenuLabelId = "SubMenuLabelId";

var subMenuLabel = "Income Level";

sg.utls.OptionsMenu.SubMenu.insertAfter(targetItemId, subMenuId, subMenuLabelId, subMenuLabel, subMenuItems);

1. Add a new MenuItem to the bottom of an existing SubMenu

var subMenuItems =

[

{ listItemId: "btnNewSubMenu1", anchorId: "labelItem3\_1", label: "Label 1", onClick: "" },

{ listItemId: "btnNewSubMenu2", anchorId: "labelItem3\_2", label: "Label 2", onClick: "" },

];

var subMenuId = "ExistingSubMenuId";

sg.utls.OptionsMenu.SubMenu.MenuItem.add(subMenuId, subMenuItems);

1. Remove an existing MenuItem from an existing SubMenu

var subMenuId = "SubMenu1";

var subMenuItemId = "SubMenuItem1";

sg.utls.OptionsMenu.SubMenu.MenuItem.remove(subMenuId, subMenuItemId);

1. Insert one or more new MenuItems into an existing SubMenu BEFORE an existing MenuItem

var subMenuItems =

[

{ listItemId: "btnNewItem1", anchorId: "labelItem3\_1", label: "Item 1", onClick: "" },

{ listItemId: "btnNewItem2", anchorId: "labelItem3\_2", label: "Item 2", onClick: "" },

];

var subMenuId = "SubMenu1";

var targetSubMenuItemId = "SubMenuItem1";

sg.utls.OptionsMenu.SubMenu.MenuItem.insertBefore(subMenu1, targetSubMenuItemId, subMenuItems);

1. Insert one or more new MenuItems into an existing SubMenu AFTER an existing MenuItem

var subMenuItems =

[

{ listItemId: "btnNewItem1", anchorId: "labelItem3\_1", label: "Item 1", onClick: "" },

{ listItemId: "btnNewItem2", anchorId: "labelItem3\_2", label: "Item 2", onClick: "" },

];

var subMenuId = "SubMenu1";

var targetSubMenuItemId = "SubMenuItem1";

sg.utls.OptionsMenu.SubMenu.MenuItem.insertAfter(subMenu1, targetSubMenuItemId, subMenuItems);

1. Example code with updated Options Menu Screenshot

// Add a new item to the bottom of the Options Menu

sg.utls.OptionsMenu.MenuItem.add("CustomMenuItemId", "Custom Menu Item 2");

// Insert a new item before the 'Import' options menu

var targetItemId = "btnOptionImport";

var itemId = "NewItemId";

var label = "Custom Menu Item 1";

var href = "";

sg.utls.OptionsMenu.MenuItem.insertBefore(targetItemId, itemId, label, href);

// Add a new SubMenu to the bottom of the Options Menu

var subMenuItems = [

{ listItemId: "btnItem1", anchorId: "labelItem1", label: "Option 1", onClick: "" },

{ listItemId: "btnItem2", anchorId: "labelItem2", label: "Option 2", onClick: "" },

{ listItemId: "btnItem3", anchorId: "labelItem3", label: "Option 3", onClick: "" },

];

var subMenuId = "SubMenu1Id";

var subMenuLabelId = "SubMenuLabelId1";

var subMenuLabel = "Custom Sub Menu";

sg.utls.OptionsMenu.SubMenu.add(subMenuId, subMenuLabelId, subMenuLabel, subMenuItems);

The above javascript will alter the OptionsMenu as shown in the following screen shot.

