Aras Innovator 2023 Release

Configurable User Interface Admin Guide

Document #: D-008095

Last Modified: *12/08/2022*

Copyright Information

Aras Corporation
100 Brickstone Square
Suite 100
Andover, MA 01810

Phone: 978-806-9400
Fax: 978-794-9826

E-mail: Support@aras.com
Website: https://www.aras.com

Table of Contents

Send Us Your Comments

Aras Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for future revisions.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where and what level of detail?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, indicate the document title, and the chapter, section, and page number (if available).

You can send comments to us in the following ways:

Email:
TechDocs@aras.com
Subject: Aras Product Documentation

Or,

Postal service:
Aras Corporation
100 Brickstone Square
Suite 100
Andover, MA 01810
Attention: Aras Technical Documentation

If you would like a reply, provide your name, email address, address, and telephone number.

If you have usage issues with the software, visit https://www.aras.com/support/

Document Conventions

The following table highlights the document conventions used in the document:

Convention	Description
Bold	This shows the names of menu items, dialog boxes, dialog box elements, and commands. Example: Click **OK**.
`Code`	Code examples appear in `courier` font. It may represent text you type or data you read.
`Yellow highlight`	Code highlighted in yellow draws attention to the code that is being indicated in the content.
`Yellow highlight with red text`	Red text highlighted in yellow indicates the code parameter that needs to be changed or replaced.
Italics	Reference to other documents.
Note:	Notes contain additional useful information.
Warning	Warnings contain important information. Pay special attention to information highlighted this way.
Successive menu choices	Successive menu choices may appear with a greater than sign (-->) between the items that you will select consecutively. Example: Navigate to **File** --> **Save** --> **OK**.

1 Introduction

This section explains the purpose and scope of the guide and describes the target audience.

1.1 Purpose and Scope of This Guide

1.1.1 Purpose

This guide describes how to configure the layout and behavior of the Aras Innovator user interface (UI). It also includes many examples to demonstrate how to implement common use cases.

1.1.2 Scope

The scope of this guide covers common UI controls in the Aras Innovator web client including the table of contents, toolbars, menus, keyboard shortcuts, item views, sidebars, headers, and title bars.

The scope of this guide does not include ItemTypes, Forms, Life Cycles, Workflows, or other general administration tasks the Aras Innovator web client.

1.2 Target Audience

This guide is written for administrators and developers who are responsible for configuring the user interface of the Aras Innovator software.

1.3 Guide Organization

The Configurable User Interface Administrator Guide is composed of the following content sections:

Table 1: Sections of the Configurable User Interface Administrator Guide

Convention	Description
Section 2: Configurable User Interface	This section provides a detailed description of the Configurable User Interface (CUI) which provides the underlying architecture for many of the configurable UI elements in the Aras Innovator web client. Understanding this framework may be helpful for developers who want to develop complex custom interfaces, but simple implementations may not require this level of technical detail.
Section 3: Table of Contents Editor	This section describes the Table of Contents Editor, a visual interface for administrators to configure the table of contents (TOC).
Section 4: Examples	This section contains step-by-step instructions on how to configure common use cases. These use cases are grouped by UI elements.

2 Configurable User Interface

This section describes the underlying technical details of configuring the Aras Innovator web client. For hands-on instructions, see Section 4.

2.1 Understanding the CUI Data Model

The Configurable User Interface, or CUI, is a modeling mechanism in Aras Innovator that allows administrators to define the layout and behavior of a client application. Standard features like the table of contents (TOC), sidebars, toolbars, menus, keyboard shortcuts, and accordion sections within item views are modeled with CUI controls.

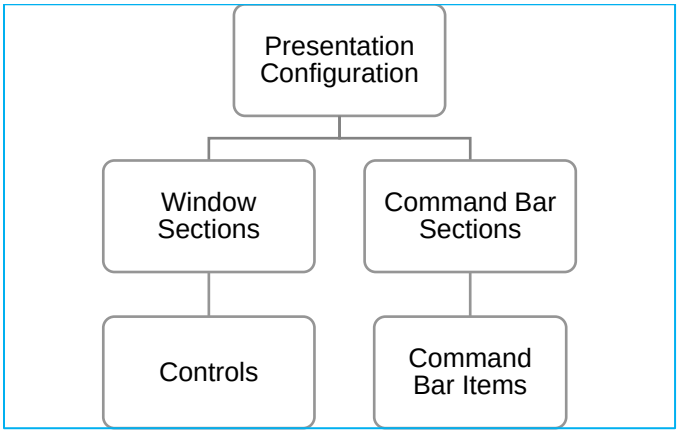

Figure 1. A simplified diagram of the CUI data model

2.2 Presentation Configurations

Presentation Configurations serve as containers for CUI configurations, defining the scope of the related CUI items as either *global* or *ItemType-specific*.

A *global* Presentation Configuration is associated with a specific client via the “presentation” item property on a Client Presentation item. In a standard Aras Innovator database, there is a single Client Presentation item to identify the global Presentation Configuration that defines the layout and UI elements that are inherited throughout the Aras Innovator web client.

Note: The CUI data model was architected with the intent that the user interface of any client application could be defined in the Aras Innovator database. That’s why there’s only one Client Presentation item in an out of the box Aras Innovator database – only the Aras Innovator web client is defined by default. However, an admin could create another Client Presentation to define the UI of a custom client application or connector.

The scope of an *ItemType-specific* Presentation Configuration is determined by relating it to an ItemType via the Client Style relationship tab. These Presentation Configurations allow administrators to override or augment the global configuration for a specific ItemType without affecting the UI of every ItemType.

As of Aras Innovator version 12.0, ItemType-specific Presentation Configurations are also used in conjunction with an ItemType’s TOC Access relationship(s) to determine where – and whether – the ItemType is displayed in the table of contents (TOC). Admins don’t need to manually create the CUI items

that display ItemTypes in the TOC; however, they will need to package the CUI configuration for any custom ItemTypes.

2.3 Window Sections

Window Sections are used to define the layout of a client application screen. Because they're related to Presentation Configuration items, Window Sections can be inherited globally or defined for an ItemType-specific scope to create different layouts.

If we look at the Window Sections related to the global Presentation Configuration in a standard Aras Innovator database, we can see examples of the two ways to define a Window Section – *declaratively* and *dynamically*.

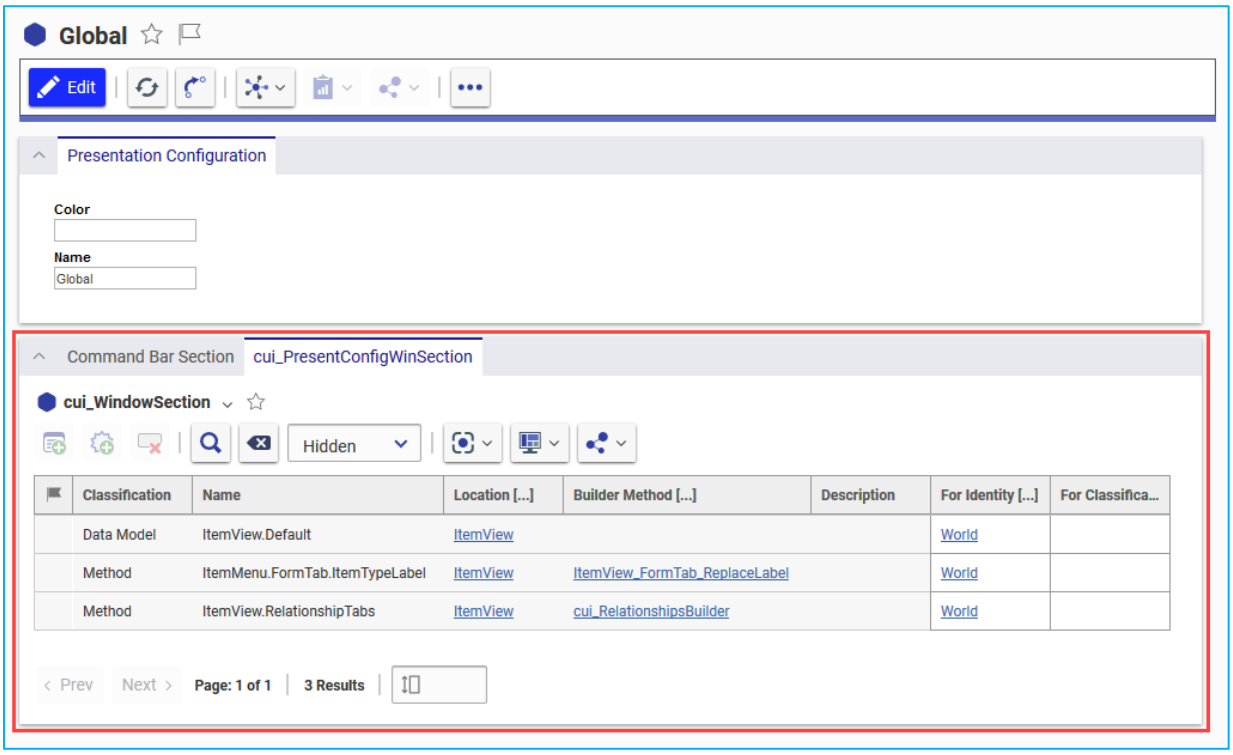

Figure 2. Globally scoped Window Sections

Declaratively defined Window Sections are configured by an administrator at "design time" and are identified with the *Data Model* classification. At runtime, the client application retrieves the Window Section and its child Controls to determine how the UI should be displayed. This approach is useful for defining client screens (or areas of client screens) when the desired layout is known in advance.

On the other hand, *dynamically-defined* Window Sections are populated by a Builder Method at runtime – hence the *Method* classification. This approach is often used for making minor adjustments, like updating the label displayed on a declaratively defined element, or completely populating a section based on the context item, as in the case of the default ItemView.RelationshipTabs Window Section.

2.4 Controls

Controls define the layout of the client UI within a parent Window Section. Using the Action property on the cui_WindowSectionControl, admins can *add*, *remove*, *replace*, or *clear all* Controls from a Window Section.

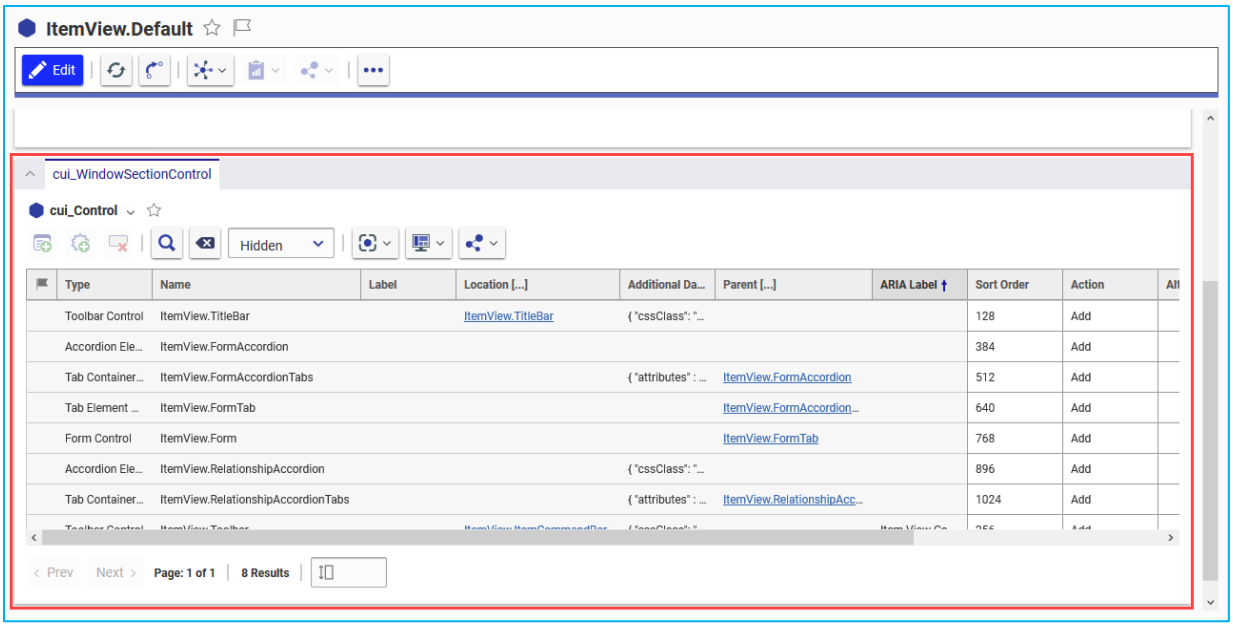

Figure 3. Example Window Section and Controls

See the following list for more details about the different types of Controls. Different classifications will support different relationship properties and different options in the Additional Data property.

2.4.1.1 Toolbar Control

The *Toolbar Control* type indicates where a command bar may be rendered in the client UI.

Use the Location property on the Control item to identify the Location that the associated CommandBarSection item should also use.

The Additional Data property optionally supports *cssClass* and *attributes* properties. See the default *ItemView.TitleBar* and *ItemView.Toolbar* items for examples.

2.4.1.2 Accordion Element Control

The *Accordion Element Control* type indicates where a collapsible accordion element may be rendered. An Accordion Element Control may be identified as the parent of one or more other controls.

The Additional Data property optionally supports *cssClass* property. See the default *ItemView.RelationshipAccordion* item for examples.

2.4.1.3 Tab Container Control

The *Tab Container Control* type indicates where a group of Tab Element Controls may be rendered. The end user can click through the tabs in a single container, alternating between the content of the child Tab Element Controls.

Use the Parent property on the cui_WindowSectionControl relationship to identify the Accordion Element Control the Tab Container Control should appear in.

The Additional Data property optionally supports the *cssClass* property. See the default *ItemView.RelationshipAccordionTabs* item for examples.

2.4.1.4 Tab Element Control

The *Tab Element Control* type indicates where a single tab may be rendered. A Tab Element Control may be identified as the parent of one or more other controls.

Use the Parent property on the cui_WindowSectionControl relationship to identify the Tab Container Control the Tab Element Control should appear in.

2.4.1.5 Form Control

The *Form Control* type indicates that a Form definition may be rendered in the client UI.

Use the Parent property on the cui_WindowSectionControl relationship to identify the Tab Element Control the Form content should appear in.

2.5 Command Bar Sections

On the surface, Window Sections and Command Bar Sections appear very similar – they're both related to Presentation Configurations, they can be defined either declaratively or dynamically, and they have many of the same properties. However, the two types serve different functions. While Window Sections and Controls define the *layout* of the client application screens, Command Bar Sections define the *content*.

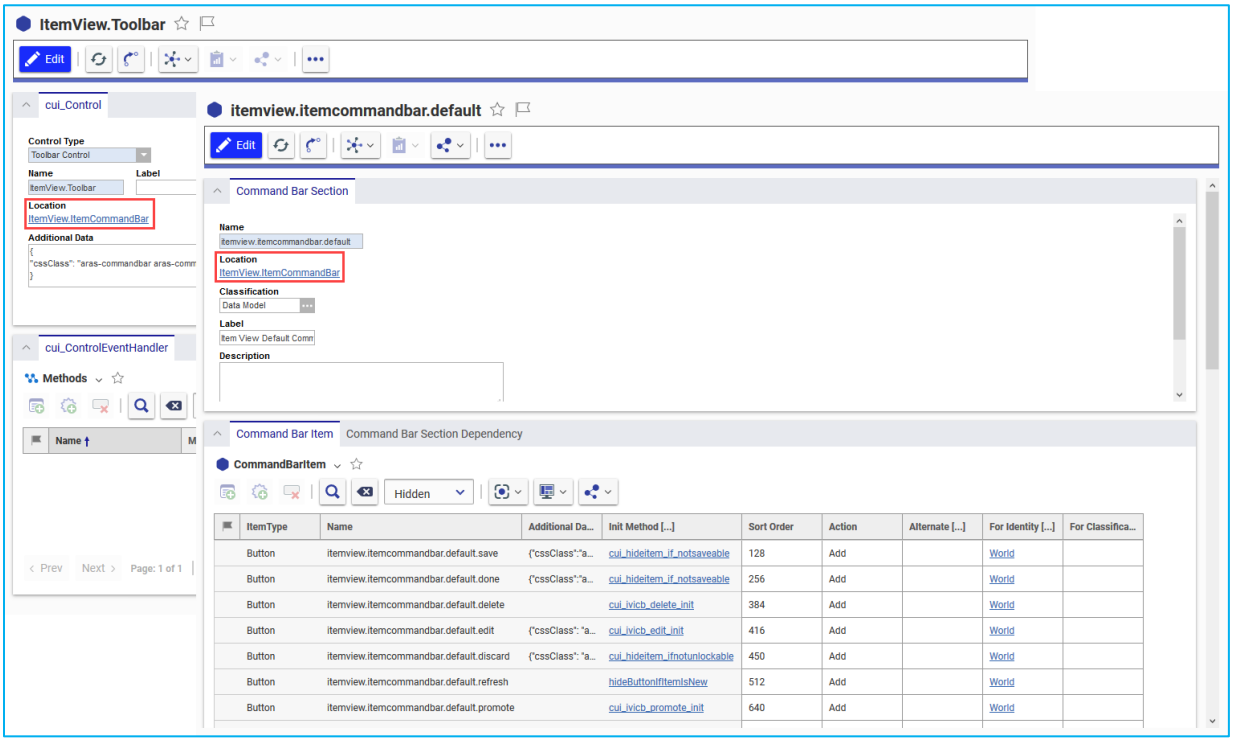

Figure 4. ItemView Toolbar Control and corresponding Command Bar Section

Consider this example based on the standard global CUI configuration. The *ItemView.Default* Window Section declares that the Aras Innovator web client has an "ItemView" screen, and one of the related Controls indicates that the ItemView has a toolbar element with the *ItemView.ItemCommandBar* Location. To determine the content of the layout defined by the Window Section and its Controls, we just need to find the Command Bar Section with the same Location and check out its related Command Bar Items.

2.6 Command Bar Items

So far, we've defined the *layout* of the client UI (Window Sections and Controls) and we've defined the *content* of the UI (Command Bar Sections and their relationships to Command Bar Items), but we haven't defined the *behavior* of the UI. That's where Command Bar Items differ from the rest of the data model we've reviewed so far – they provide the ability to execute logic based on user interaction, client state, and item context in addition to the add, remove, replace, and clear all actions on the relationship.

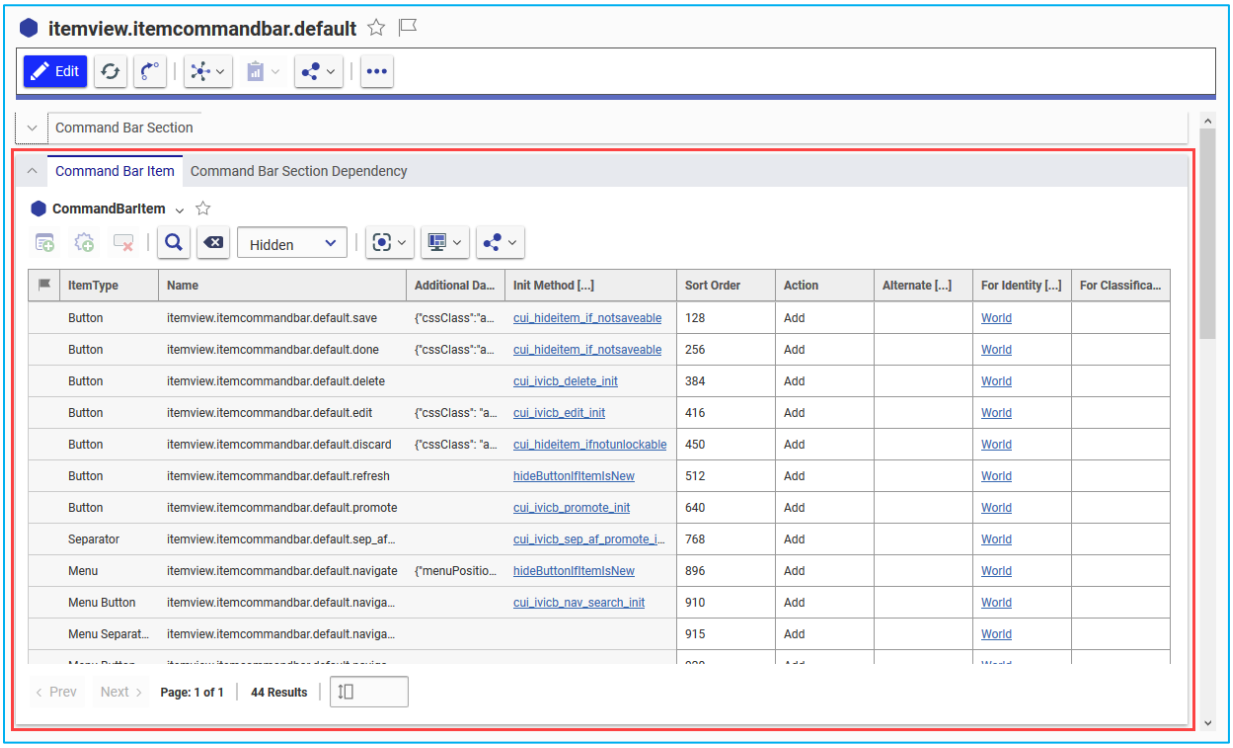

Figure 5. Default ItemView Command Bar Items

Command Bar Items are also implemented differently. While Window Sections, Controls, and Command Bar Sections all used classifications to differentiate items, the CommandBarItem ItemType is implemented as a polymorphic ItemType.

See the list in the following section for more details on the various ItemTypes that are implemented as sources of the CommandBarItem poly ItemType.

2.6.1.1 Button

A *CommandBarButton* item represents an element that triggers an action. It can be single state, where a click executes the same action every time, or it can have two states – active and inactive.

Buttons support multilingual labels, multilingual tooltips, images, and the following properties:

- *Additional Data:* supports *cssClass*
- *Init Method:* a Method, often used to show, hide, enable, or disable an element when initialized
- *Click Method:* a Method used to execute some logic when the element is clicked by a user
- *Include Events:* state change events that should trigger the item's initialization method

See the default *itemview.itemcommandbar.default.save* item for an example.

2.6.1.2 Checkbox

A *CommandBarCheckbox* represents a single, atomic item in any non-menu container that can be toggled on/off. Can be styled as a checkbox or radio button.

Checkboxes support multilingual labels, multilingual tooltips, and the following properties:

- *Additional Data:* supports custom parameters
- *Init Method:* a Method, often used to show, hide, enable, or disable an element when initialized
- *Click Method:* a Method used to execute some logic when the element is clicked by a user
- *Include Events:* state change events that should trigger the item's initialization method

2.6.1.3 Dropdown

A *CommandBarDropDown* item represents a combobox element. The element's options are populated from the Additional Data property merged with the result of the Init Method.

Dropdowns support multilingual labels, multilingual tooltips, images, and the following properties:

- *Additional Data:* supports *cui_items*, an array of objects each containing an id and name or label
- *Init Method:* a Method, often used to show, hide, enable, or disable an element when initialized
- *Click Method:* a Method used to execute some logic when the element is clicked by a user
- *Include Events:* state change events that should trigger the item's initialization method

Note: If the Init Method returns false or an empty array, the dropdown control will not be created.

2.6.1.4 Separator

A *CommandBarSeparator* represents a divider to visually separate items in any non-menu container.

Separators support the following properties:

- *Additional Data:* supports custom parameters
- *Init Method:* a Method, often used to show, hide, enable, or disable an element when initialized
- *Include Events:* state change events that should trigger the item's initialization method

See the default *itemview.itemcommandbar.default.sep_af_promote* item for an example.

2.6.1.5 Menu

A *CommandBarMenu* represents a hierarchical menu that can contain CommandBarMenuButton items or other CommandBarMenu items.

Menus support multilingual labels, multilingual tooltips, images, and the following properties:

- *Parent Menu:* the parent CommandBarMenu item if this item is a sub-menu
- *Additional Data:* supports *menuPosition*
- *Init Method:* a Method, often used to show, hide, enable, or disable an element when initialized
- *Click Method:* a Method used to execute some logic when the element is clicked by a user
- *Include Events:* state change events that should trigger the item's initialization method

See the default *itemview.itemcommandbar.default.navigate* item for an example.

2.6.1.6 Menu Button

A *CommandBarMenuButton* represents a single, atomic button in a menu with some associated logic.

Menu Buttons support multilingual labels, multilingual tooltips, images, and the following properties:

- *Parent Menu:* the parent CommandBarMenu item
- *Additional Data:* supports custom parameters
- *Init Method:* a Method, often used to show, hide, enable, or disable an element when initialized
- *Click Method:* a Method used to execute some logic when the element is clicked by a user
- *Include Events:* state change events that should trigger the item's initialization method

See the default *itemview.itemcommandbar.default.navigate.structurebrowser* item for an example.

2.6.1.7 Menu Separator

A *CommandBarMenuSeparator* represents a divider to visually separate items in a menu.

Menu Separators support the following properties:

- *Parent Menu:* the parent CommandBarMenu item
- *Additional Data:* supports custom parameters
- *Init Method:* a Method, often used to show, hide, enable, or disable an element when initialized
- *Include Events:* state change events that should trigger the item's initialization method

See the default *itemview.itemcommandbar.default.navigate.aftersearch* item for an example.

2.6.1.8 Menu Checkbox

A *CommandBarMenuCheckbox* represents a single, atomic item in a menu that can be toggled on/off. Can be styled as a checkbox or radio button.

Menu Checkboxes support multilingual labels, multilingual tooltips, and the following properties:

- *Parent Menu:* the parent CommandBarMenu item
- *Additional Data:* supports custom parameters
- *Init Method:* a Method, often used to show, hide, enable, or disable an element when initialized
- *Click Method:* a Method used to execute some logic when the element is clicked by a user
- *Include Events:* state change events that should trigger the item's initialization method

A *CommandBarShortcut* item represents a keyboard shortcut that triggers an action. It can only be used with specific Locations intended for shortcuts.

Shortcuts support the following properties:

- *Additional Data:* supports *stopPropagation*, *useCapture*, *preventDefault*, *preventBlur*, and *context*
  - *stopPropagation:* Boolean property, prevents further propagation of the current event
  - *useCapture:* Boolean property, set to true if the “this” pointer must point to the context item
  - *preventDefault:* Boolean property, cancels the default event if it’s cancelable, without stopping further propagation
  - *preventBlur:* Boolean property, prevents the blur event if set to true and a domNode has the focus
  - *context:* Object property, the “this” pointer references this object if useCapture is true
- *Click Method:* a Method used to execute some logic when the shortcut is keyed in by a user
- *Shortcut:* the key combination for triggering the shortcut

Table 3 lists valid shortcut combinations:

Table 2: Valid shortcuts

Ctrl+	Shift+	Ctrl+Shift+	Button on keyboard
Supported	Supported	Supported	a-z
Supported	Supported	Supported	0-9
Supported	Supported	Supported	Num0-Num9
Supported	Supported	Supported	~
Supported	Supported	Supported	tab
Supported	Supported	Supported	enter
Supported	Supported	Supported	insert
Supported	Supported	Supported	delete
Supported	N/A	N/A	pageup, pagedown, home, end
Supported	N/A	N/A	F1-F12
Supported	N/A	N/A	arrows, Num-arrows
Supported	N/A	N/A	<, >
Supported	N/A	N/A	[,]
Supported	N/A	N/A	-, +, Num-, Num+

See the default *com.aras.innovator.cui_default.mws_delete* item for an example.

2.6.1.9 Edit

The Edit ItemType enables you to create a text box control and use it in a Command Bar to capture data relevant to a particular command.

The Edit option supports multilingual labels, multilingual tooltips, and the following properties:

- *Additional Data:* supports custom parameters
- *On Keydown Method:* a Method used to execute some logic when the shortcut is keyed in by a user
- *Init Handler:* a Method, often used to show, hide, enable, or disable an element when initialized

The following screenshot shows an example using the Class Structure dialog box:

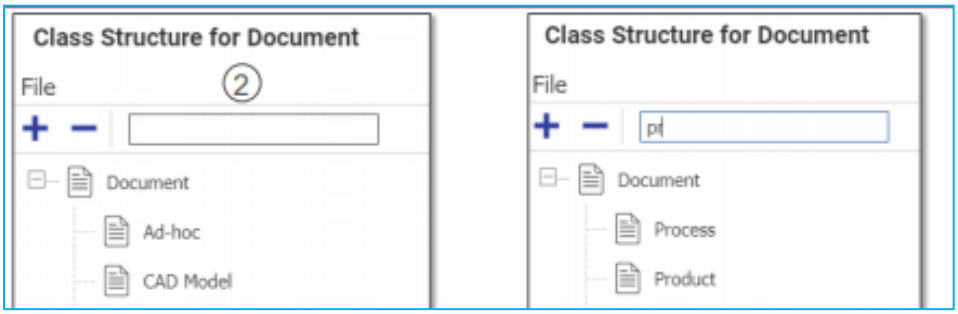

Figure 6.

In this example, entering “pr” in the search box displays only those documents that start with the letters “pr” enabling you to create a class structure for documents.

The Edit ItemType enables you to create a text box control which can be placed in a command bar to capture data relevant to a command. The following figure shows an example using the CommandBarEdit ItemType.

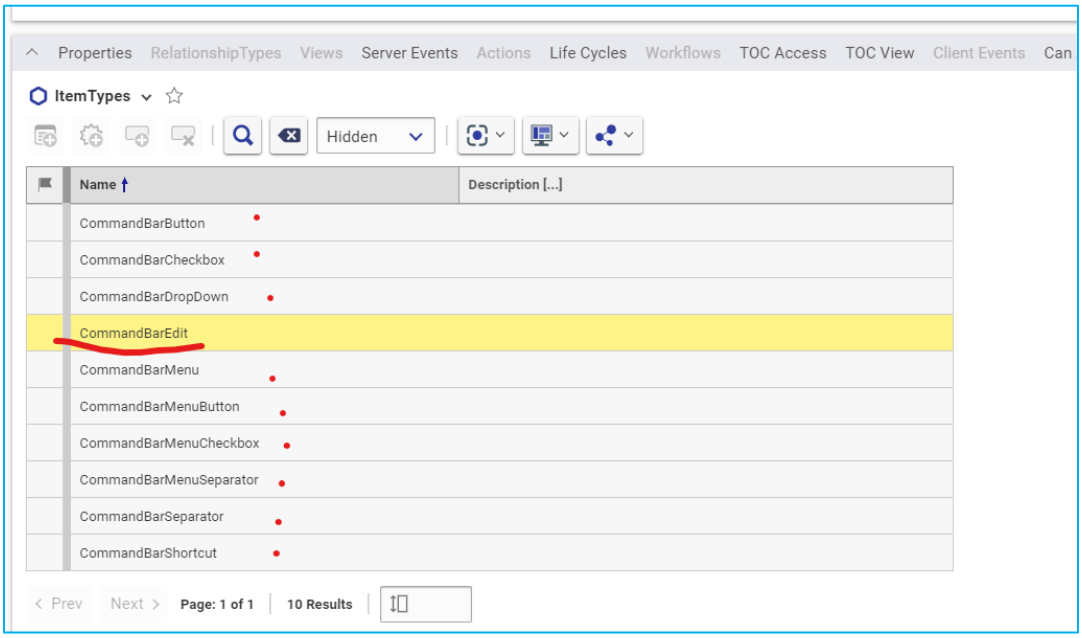

Figure 7.

2.7 Common CUI Properties

The following properties occur on most or all CUI ItemTypes.

2.7.1.1 Location

The *Location* property indicates the intended location or purpose of the CUI item. In Aras Innovator 11.0, this property was based on a List of predefined values. As part of the UX/UI enhancements included in Aras Innovator 12.0, the property was changed to an Item property with data source cui_Location. This update improves the ability for admins to define their own custom locations for client applications.

2.7.1.2 For Identity

The *For Identity* property indicates that the members of the identity will be affected by the CUI configuration item.

For example, if a Command Bar Section has a relationship to a Button with the *Add* action and the For Identity property is set to *World*, all users will see the button.

If a Command Bar Section has another relationship to the same Button with the *Remove* action and the For Identity property is set to *All Suppliers*, all users **except** members of the All Suppliers identity will see the button.

2.7.1.3 For Classification

The *For Classification* property indicates that the CUI configuration item will be evaluated when the context item has the specified classification.

For example, if a Command Bar Section has a relationship to a Button with the *Add* action and the For Classification property is set to *Component*, all users will see the button only when the context item has the Component classification.

3 Table of Contents Editor

This section describes the functionality of the Table of Contents (TOC) Editor.

3.1 Purpose of the Table of Contents Editor

The TOC Editor provides a WYSIWYG-style editing experience for administrators who configure the Aras Innovator table of contents. This feature is intended to meet the following goals.

1. **Reduce Context Switching:** Admins can configure all TOC content in a single interface. No need to switch between tabs and edit individual ItemTypes.
2. **Improve Visibility:** Admins can instantly see the results of their edits without logging out and back into Aras Innovator. The “View As” functionality also enables admins to see the TOC view for different users and groups, making it easier to test and troubleshoot their updates.
3. **Simplify Packaging:** The TOC Editor automatically packages the CUI objects that are used to render the TOC controls.

3.2 Understanding the Editor Interface

The TOC Editor is comprised of three main sections: the toolbar, the contents pane, and the details pane.

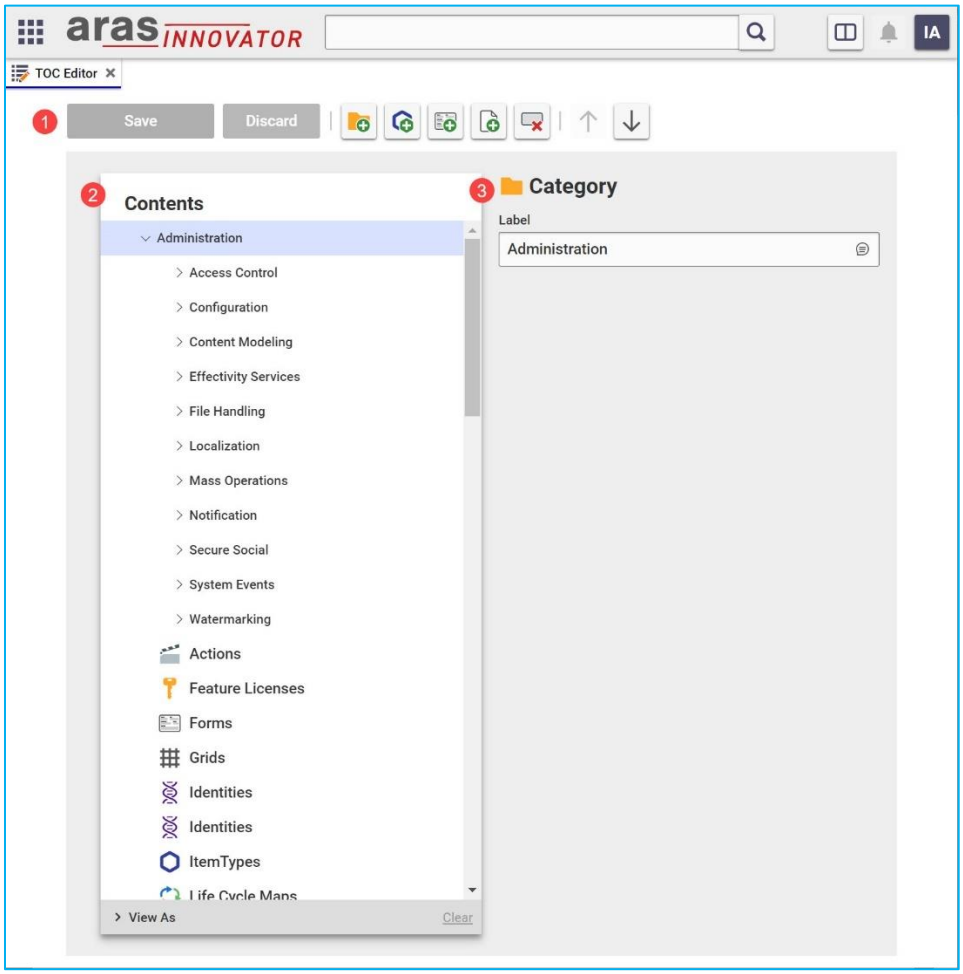

Figure 8. Viewing the TOC Editor

3.2.1 Editor Toolbar

The TOC Editor toolbar contains the following controls.

- **Save:** Saves the current changes in the editor. If the editor tab is closed before the user clicks Save, all changes made since the last save will be lost. This button will be disabled if there are no changes in the editor.
- **Discard Changes:** Discards the current changes in the editor. This button will be disabled if there are no changes in the editor.
- **Add Category:** Adds a new Category to the TOC. If another Category is selected when this button is clicked, the new Category will be created as a child of the selected Category. If no Category is selected, the new Category will be created at the bottom of the TOC.

- **Add ItemType:** Adds a new ItemType button to the TOC. If a Category is selected when this button is clicked, the new ItemType button will be created as a child of the selected Category. If no Category is selected, the new ItemType button will be created at the bottom of the TOC.
- **Add Form:** Adds a new Form button to the TOC. A Form button displays an Aras Innovator form in a new tab when a user clicks it.
- **Add Page:** Adds a new Page button to the TOC. A Page button displays an HTML page in a new tab when a user clicks it.
- **Delete:** Deletes the currently selected Category or ItemType button.
- **Move Up:** Moves the selected Category or ItemType button within its parent Category. This button will be disabled if the selected Category or ItemType button cannot be moved higher.
- **Move Down:** Moves the selected Category or ItemType button within its parent Category. This button will be disabled if the selected Category or ItemType button cannot be moved lower.

3.2.2 Contents Pane

The pane on the left side of the TOC Editor displays all Categories and ItemType buttons that are configured in the Aras Innovator database. It also includes several options for filtering the content.

3.2.2.1 View As: Identity

Using the Identity “View As” filter at the bottom of the Contents Pane, admins can choose to view only the TOC configuration for a specified user or group. The TOC Editor will enter a read-only mode when the View As filter is active.

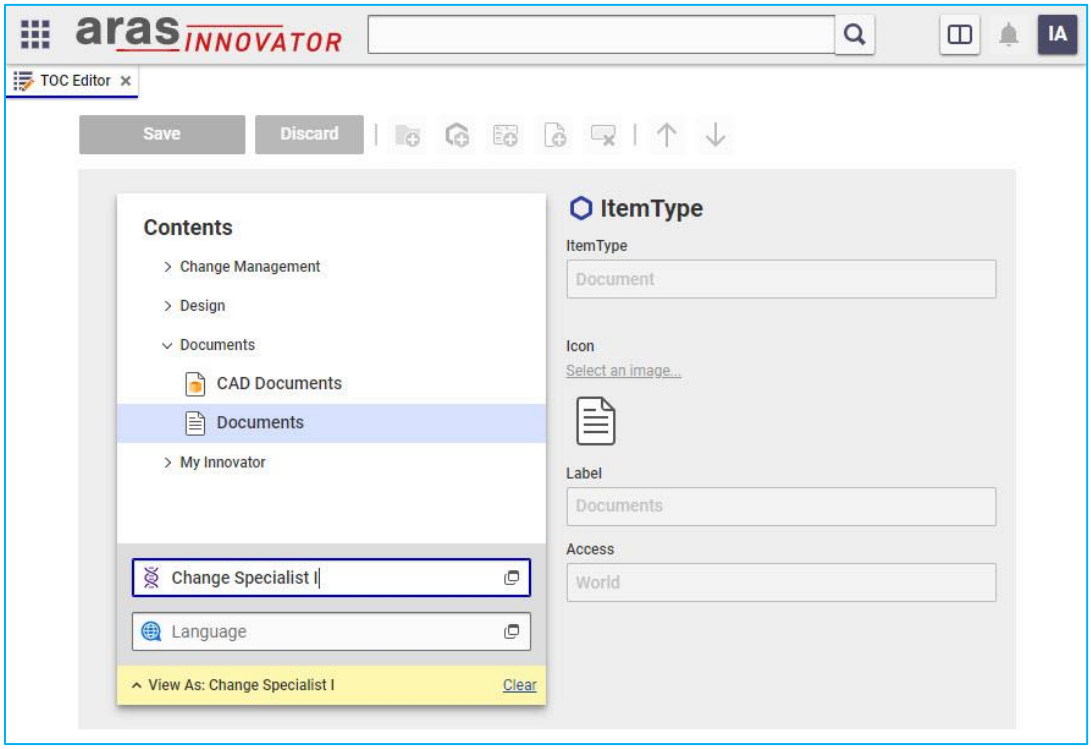

Figure 9. Viewing the TOC as a member of Change Specialist I

3.2.2.2 View As: Language

Using the Language “View As” filter at the bottom of the Contents Pane, admins can choose to view a user’s or group’s TOC configuration in a specific language. The TOC Editor will enter a read-only mode when the View As filter is active.

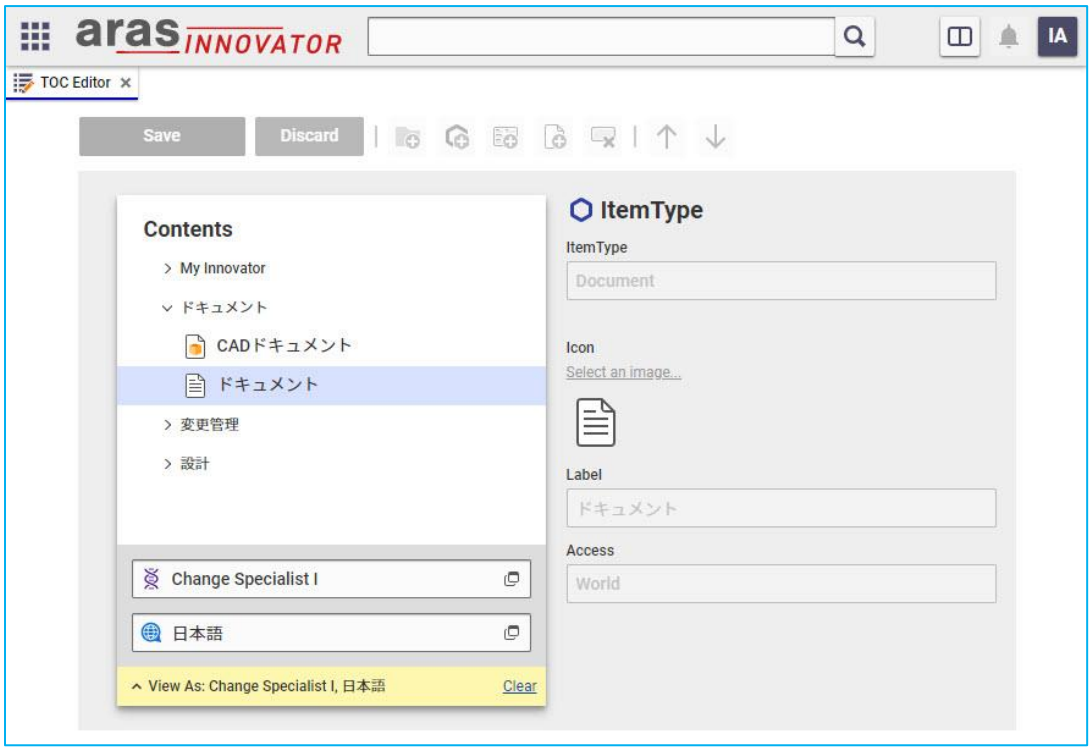

Figure 10. Viewing the TOC as a Change Specialist I with Japanese client settings

Note: The Language filter will be enabled only when an Identity is specified and there are two or more languages configured in the Aras Innovator database. For more details on configuring Aras Innovator for multiple languages, please refer to the Configuring Internationalization documentation.

3.2.3 Details Pane

The pane on the right side of the TOC Editor displays a form for viewing and editing the properties of the selected row in the Contents Pane.

3.2.3.1 No Item Selected

When no items are selected in the Contents Pane, the Details Pane shows general properties for the TOC. This is the default view when the TOC Editor opens.

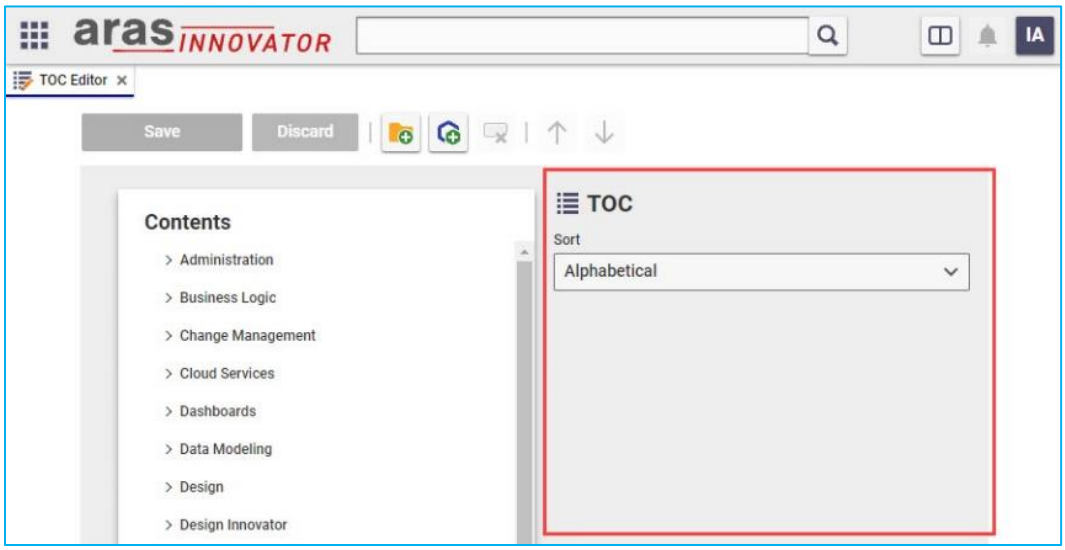

Figure 11. Properties for the TOC are shown when no items are selected

- **Sort:** Determines the order that the Aras Innovator web client renders the TOC elements.
  - **Alphabetical:** The default setting is *Alphabetical*, which sorts TOC Categories and their child items in alphabetical order. This setting is consistent with the behavior of Aras Innovator 12 and earlier releases.

    Note: The order of items in the Contents Pane and the actual TOC may appear different when the sort properties is set to *Alphabetical*.

  - **Custom:** This setting sorts the TOC Categories and ItemType buttons according to the order defined in the TOC Editor.

To deselect an item in the Contents Pane and show the TOC properties, click the **Contents** header in the Contents Pane.

3.2.3.2 Category Selected

When a Category is selected in the Contents Pane, the Details Pane shows a form for viewing and editing the properties of that Category.

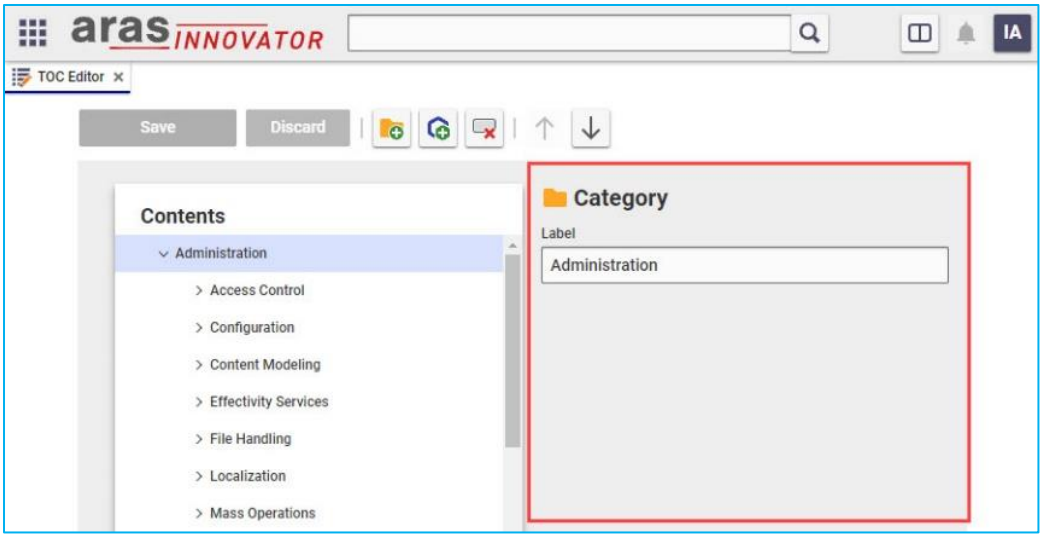

Figure 12. Properties for the selected Category

- **Label:** This multilingual property defines the text that is shown for the Category in the TOC.

3.2.3.3 ItemType Button Selected

When an ItemType button is selected in the Contents Pane, the Details Pane shows a form for viewing and editing the properties of that ItemType button.

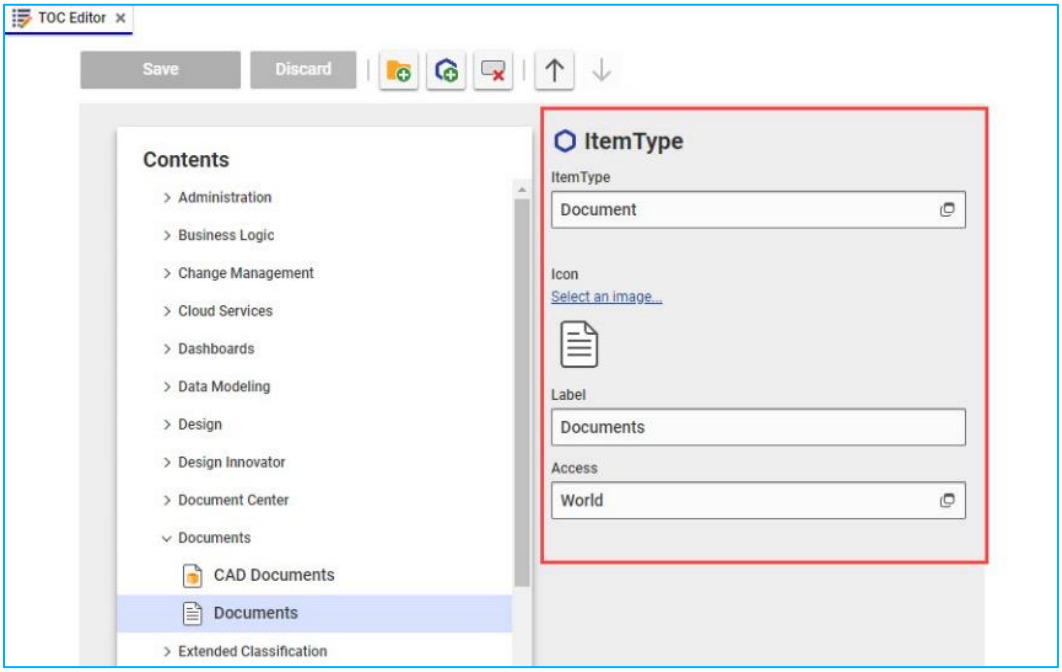

Figure 13. Properties for the selected ItemType button

- **ItemType:** Sets the ItemType associated with the button. When the button is clicked in the TOC, the Navigation Pane will display the user's favorite searches, favorite items, and common actions for that ItemType.
- **Icon:** Defines the icon that appears on the button in the TOC.

- **Label:** This multilingual property defines the text that is shown on the button in the TOC.
- **Access:** Defines which Identity members will see this button in the TOC.

Note: Each button can only be configured for a single Identity. To display an ItemType in the TOC for multiple Identities, either create a separate button for each Identity or use a group Identity that encompasses all individuals or groups who should see that ItemType in the TOC.

3.2.3.4 Form Button Selected

When a Form button is selected in the Contents Pane, the Details Pane shows a form for viewing and editing the properties of that Form button.

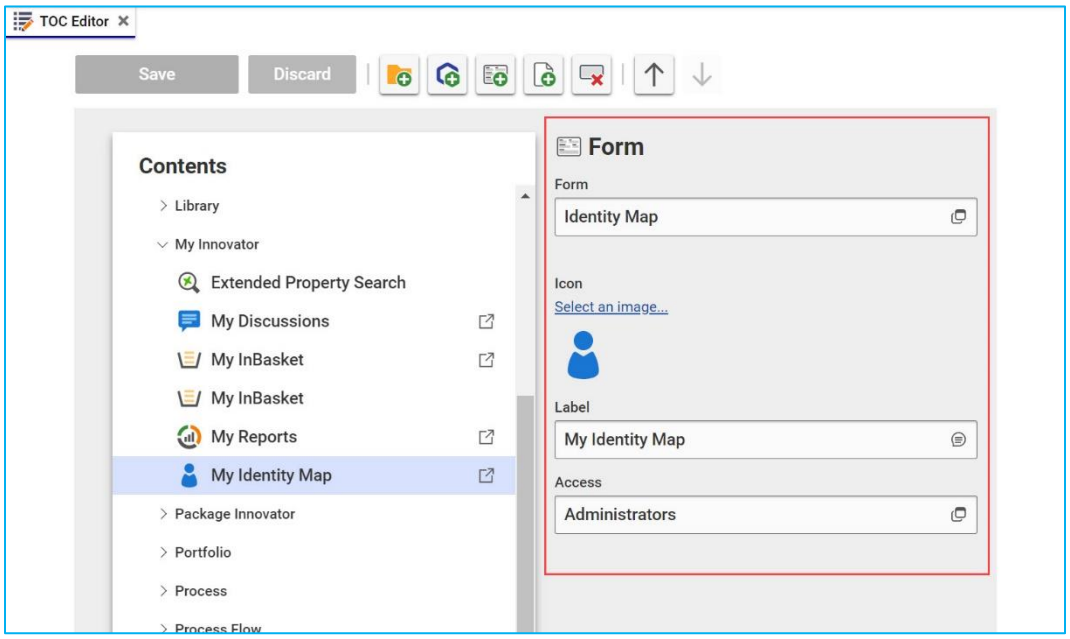

Figure 14. Properties for the selected ItemType button

- **Form:** Sets the Form associated with the button. When the button is clicked in the TOC, the Form will display in a new Aras Innovator tab.
- **Icon:** Defines the icon that appears on the button in the TOC.
- **Label:** This multilingual property defines the text that is shown on the button in the TOC.
- **Access:** Defines which Identity members will see this button in the TOC.

3.2.3.5 Page Button Selected

When a Page button is selected in the Contents Pane, the Details Pane shows a form for viewing and editing the properties of that Page button.

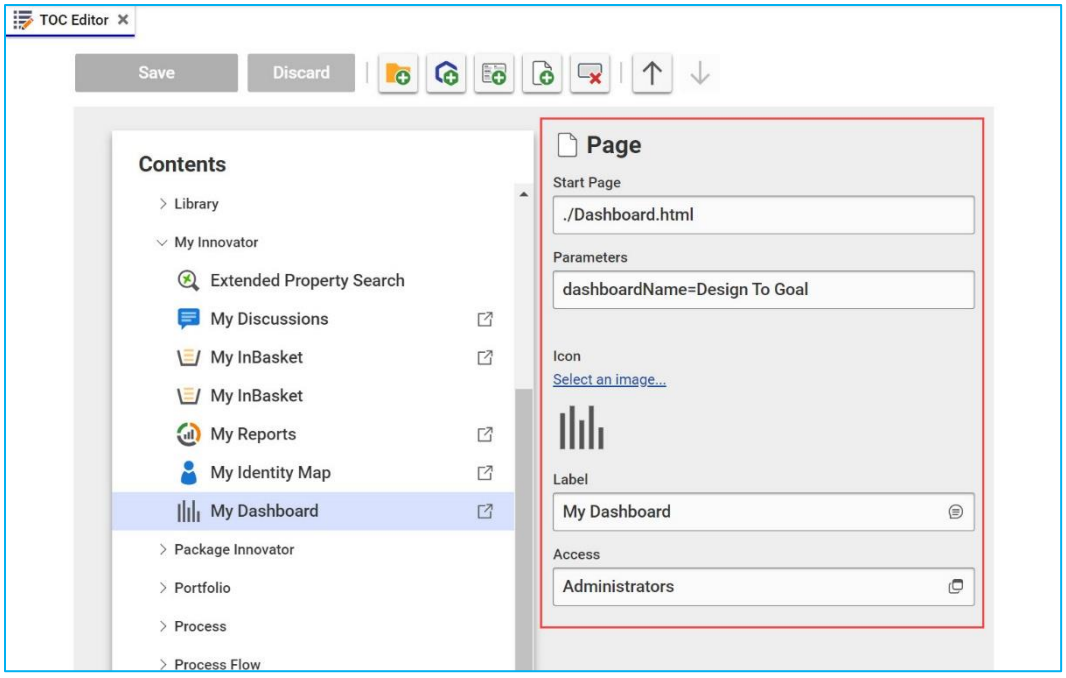

Figure 15. Properties for the selected ItemType button

- **Start Page:** Specifies the path of an HTML page to display when the button is clicked. This should be an HTML file in the code tree, and the path should be relative to the *Innovator/Client/scripts* directory.
- **Parameters:** Optional. Passes parameters for the HTML page to use.
- **Icon:** Defines the icon that appears on the button in the TOC.
- **Label:** This multilingual property defines the text that is shown on the button in the TOC.
- **Access:** Defines which Identity members will see this button in the TOC.

3.3 Importing and Exporting

Packaging Aras Innovator configuration data to share changes between developers or different environments, like test and production. This section provides information on how to handle packages that include TOC configuration data.

3.3.1 Exporting TOC Elements

The TOC Editor manages the underlying CUI items that define and render the TOC controls in the Aras Innovator web client. These CUI items need to be packaged to export the TOC configuration to another Aras Innovator instance.

3.3.1.1 ItemType Buttons

The TOC Editor automatically adds each ItemType button to the package that contains its associated ItemType. For ItemTypes that belong to package definitions, admins can simply export the packages and import them into another environment.

If an ItemType button references an ItemType that does not belong to a package, there are a couple options. One option is to remove the button from the TOC, package the ItemType, and then recreate the button in the TOC Editor. This will automatically package the button.

The second option is to manually package the CUI items that define the ItemType button. The following steps demonstrate how to manually package a custom “Part” ItemType button.

Note: To simplify these steps, it’s recommended to download and install the CUI Add to Package community project from the Aras Labs GitHub page. It’s possible to perform the same tasks without the project’s helpful “Add to Package” action. However, this requires adding the CommandBarSection and CommandBarMenuButton ItemTypes to the TOC and searching for the relevant items to package.

1. Navigate to **Administration > ItemTypes** in the TOC.
2. Search for the *Part* ItemType and open it.
3. Click the **Client Style** relationship tab and open the Presentation Configuration item.
4. In the Command Bar Section grid, open the **Part_TOC_Content** item.
5. In the Command Bar Item grid, open the Menu Button item that corresponds with the custom ItemType button in the TOC. The format of the name will resemble com.aras.innovator.cui_default.toc_Part_<GUID>.
6. Click the “**…**” button in the toolbar and select **Add to Package Definition**.
7. **Choose the package** from the list that the Menu Button will be assigned to.
8. **Close** the Menu Button item, Command Bar Section item, Presentation Configuration item, and ItemType tabs.

The custom ItemType button is now packaged and can be exported with the export utility.

Warning The CUI items that define ItemType buttons should be packaged with the ItemType they’re assigned. If they are assigned to a different package, that package definition must have a “depends on” relationship to the package definition containing the ItemType.

3.3.1.2 Categories

Since Categories are not explicitly assigned an ItemType, they must be packaged manually. The following steps demonstrate how to package a custom Category called “Design Pros”.

Note: To simplify these steps, it’s recommended to download and install the CUI Add to Package community project from the Aras Labs GitHub page. It’s possible to perform the same tasks without the project’s helpful “Add to Package” action. However, this requires adding the CommandBarSection and CommandBarMenuButton ItemTypes to the TOC and searching for the relevant items to package.

1. Navigate to **Administration > Configuration > Client Presentation** in the TOC.
2. Click the **Search** button to run the Client Presentation search.
3. Click the **Global** item property in the single search result.
4. In the Command Bar Section grid, open the **com.aras.innovator.cui_default.toc** item.
5. In the Command Bar Item grid, open the Menu item that corresponds with the custom Category in the TOC. The format of the name will resemble com.aras.innovator.cui_default.toc_<GUID>.
6. Click the “**…**” button in the toolbar and select **Add to Package Definition**.
7. **Choose the package** from the list that the Menu will be assigned to.

8. **Close** the Menu item, Command Bar Section item, Presentation Configuration item, and ItemType tabs.

The custom Category is now packaged and can be exported with the export utility.

Warning The CUI items that define Categories should be packaged in the com.aras.innovator.cui_default package or a package definition with a “depends on” relationship to the com.aras.innovator.cui_default package.

3.3.2 Importing TOC Elements

3.3.2.1 Packages Containing TOC Access or TOC View

Prior to Aras Innovator Release 14, admins defined the TOC using the legacy TOC Access and TOC View relationships on ItemTypes. These relationships are no longer used as of Release 14, however, packages exported from Aras Innovator 12 SP20, or earlier releases may still include this data. These packages can be imported into Release 14+, but they will need a few updates to properly capture the TOC configuration.

Note: Subscribers upgrading to Aras Innovator Release 14+ from an earlier release of Aras Innovator will not need to perform these steps for any packages in the scope of an Aras Upgrades service project. For questions about the Aras Upgrade subscription benefit, please refer to support@aras.com.

The following steps are required to update a package that contains TOC Access or TOC View data.

1. Import the package into Aras Innovator (Release 14 or greater).
2. For each TOC Access relationship in the package, create a new ItemType button using the TOC Editor. The “Access” property of each button should match the Identity related to the TOC Access relationship in the package.

   Note: The TOC Access and TOC View relationships will not be visible on the ItemType form in the Aras Innovator web client. To find this data, review the AML files in the ItemType subdirectory of the package.

3. For each TOC View relationship in the package, create a new Form or Page button following the steps in 4.4.7 or 4.4.8 in this guide.
4. Export the package using the export utility.

The package can now be imported to Aras Innovator Release 14+ without additional manual steps.

4 Examples

Warning Because many of the standard UI elements in the Aras Innovator web client are defined via CUI, it's important to back up your database before changing any of the out of the box controls.

4.1 Toolbars

4.1.1 Add a button to an item toolbar

"I want to add a button to the Part item toolbar for all users."

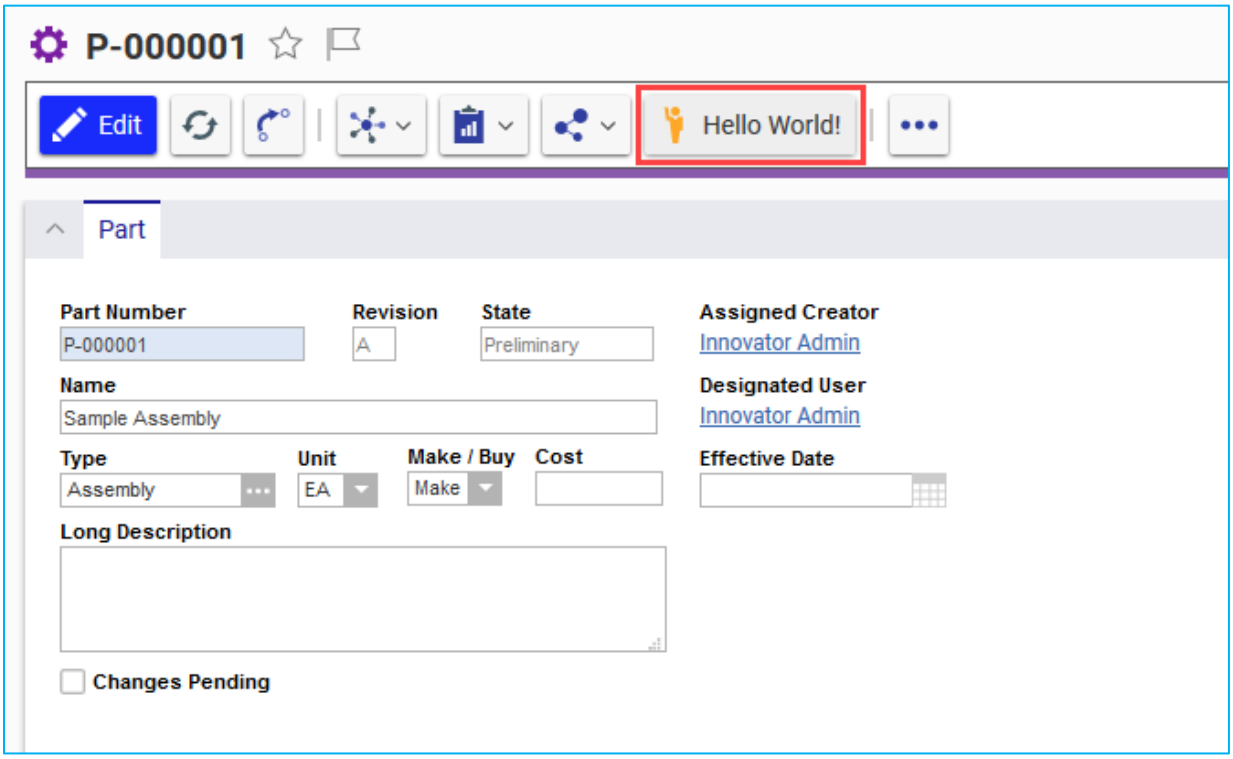

Figure 16. A custom button appears in the Part item toolbar

Note: This section demonstrates how to add a Command Bar Item to the Item View toolbar for a specific ItemType. See the "Add a separator to a toolbar" section for steps to add a Command Bar Item to the Item View toolbar for all ItemTypes.

1. Open the **Part** ItemType and select the **Client Style** relationship tab.
2. Right click the entry in the grid and select **Open** to open the Presentation Configuration for the Part ItemType.
3. Click the **Edit** button.

4. Click the **New** button in the Command Bar Section grid to create a new section with the following properties:
    a. Classification: **Data Model**
    b. Name: **Part_Custom_Buttons**
    c. Location: **ItemView.ItemCommandBar**
    d. Sort Order: **256**
    e. For Identity: **World**
5. Click the **Save** button, then right click the new Command Bar Section, and select **Open**.
6. Click the **Edit** button.
7. Click the **New** button in the Command Bar Item grid, then select **Button > OK** in the dialog to create a new button with the following properties:
    a. Name: **My_Button**
    b. Sort Order: **2000**
    c. Action: **Add**
    d. For Identity: **World**
8. Click **Save**, then right click the new Button, and select **Open**.
9. Click the **Edit** button.
10. Set the following properties on the item form:
    a. Label: **enter the label** you want to appear on the button
    b. Click Method: **choose a Method** you want to execute when the button is clicked
    c. Image: **choose an icon** to display in the button
11. Click **Save**, then navigate to **Design > Parts** in the TOC.
12. Open a Part item to see the new button in the toolbar.

4.1.2 Add a separator to all item toolbars

“I want to add a separator before the Refresh button in the toolbar for all ItemTypes and users.”

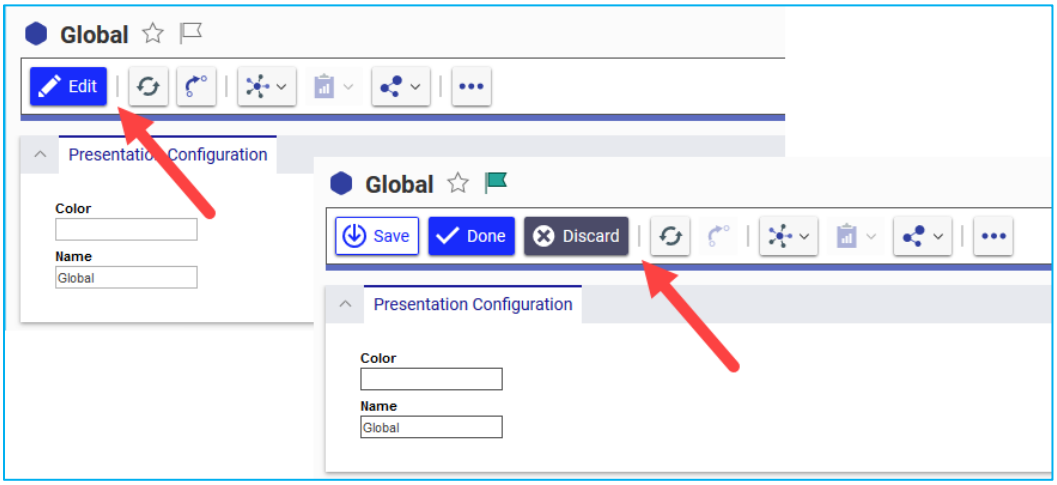

Figure 17. A custom separator appears before the refresh button in all item toolbars

Note: This section demonstrates how to add a Command Bar Item to the Item View toolbar for all ItemTypes. See the “Add a button to a toolbar” section for steps to add a Command Bar Item to the Item View toolbar for a specific ItemType.

1. Navigate to **Administration > Configuration > Client Presentation** in the TOC.
2. Click the **Search** button to run the Client Presentation search.
3. Click the **Global** item property in the single search result.
4. In the Command Bar Section grid, open the **itemview.itemcommandbar.default** item.
5. Click the **Edit** button.
6. Click the **New** button in the Command Bar Item grid, then select **Separator > OK** in the dialog to create a new separator with the following properties:
   a. Name: **My_Separator**
   b. Sort Order: **500**
   c. Action: **Add**
   d. For Identity: **World**
7. Click the **Save** button.
8. Open any item to see the new separator appear before the Refresh button in the toolbar.

4.1.3 Remove a button on a relationship toolbar

“I want to hide the Share button on the Part BOM relationship toolbar for members of All Suppliers.”

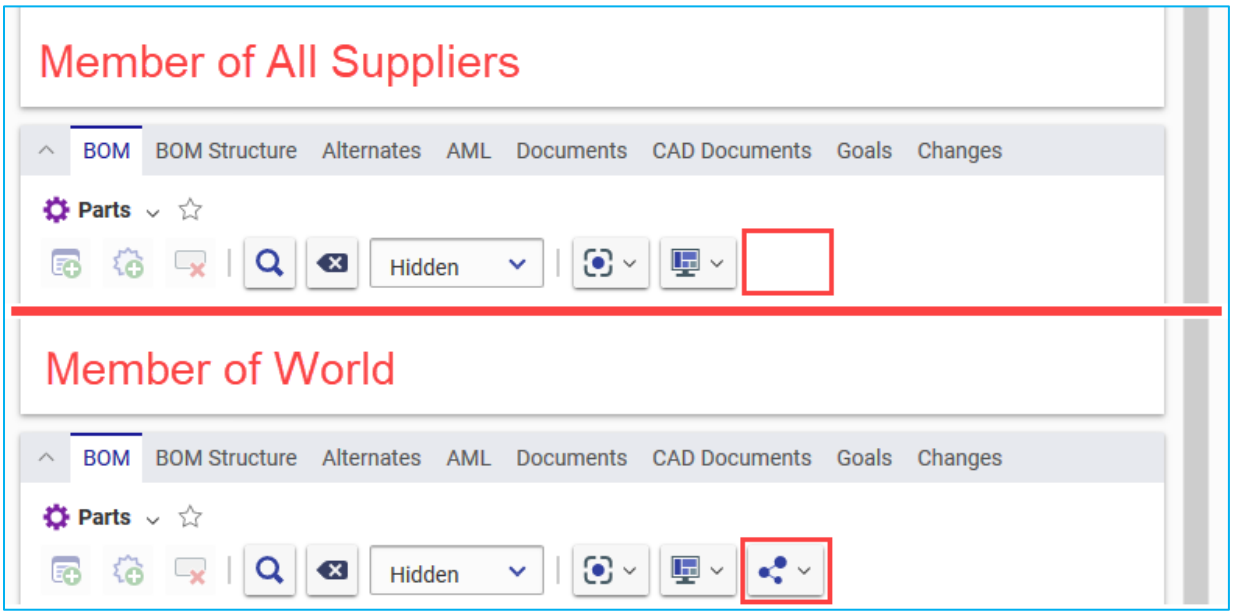

Figure 18. The default “share” button does not appear on Part BOM for All Suppliers

1. Open the **Part BOM** ItemType and click the **Edit** button.
2. In the **Client Style** relationship tab, click the **New** button to create a new Presentation Configuration with the following properties:

   Name: **Part_BOM**
3. Click the **Save** button, then right click the new Presentation Configuration, and select **Open**.
4. Click the **Edit** button.
5. Click the **New** button in the Command Bar Section grid to create a new section with the following properties:
   a. Classification: **Data Model**
   b. Name: **Part_BOM_Custom**
   c. Location: **ItemView.RelationshipsCommandBar**
   d. Sort Order: **128**
   e. For Identity: **All Suppliers**
6. Click the **Save** button, then right click the new Command Bar Section, and select **Open**.
7. Click the **Edit** button.
8. Click the **Add** button in the Command Bar Item grid, then search for the **commonitems.commandbar.sharemenu** item. Click **OK** in the search dialog to add the item to the Command Bar Items grid.

9. Set the following properties for the Command Bar Item:
    a. Action: **Remove**
    b. For Identity: **All Suppliers**
10. Click **Save**, then logout of Aras Innovator.
11. Login to Aras Innovator as a member of the All Suppliers group identity.
12. Open a Part item to see that the Share button no longer appears in the Part BOM relationship toolbar.

4.1.4 Replace a button on a toolbar

"I want to replace the default "New" button with a custom button on the Part search toolbar for all users."

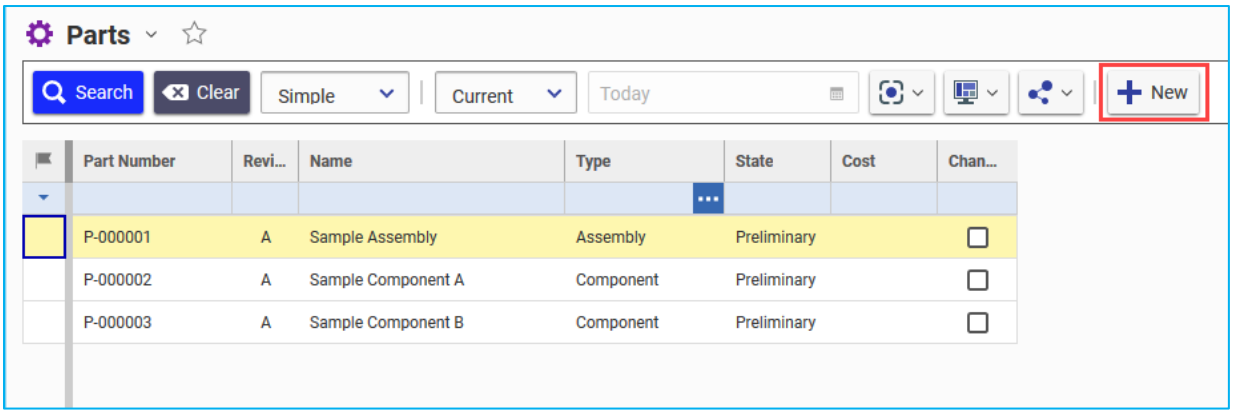

Figure 19.

The default "new" button is replaced with a custom button in the search toolbar.

1. Open the **Part** ItemType and select the **Client Style** relationship tab.
2. Right click the entry in the grid and select **Open** to open the Presentation Configuration for the Part ItemType.
3. Click the **Edit** button.
4. Click the **New** button in the Command Bar Section grid to create a new section with the following properties:
    a. Classification: **Data Model**
    b. Name: **Part_Search_Custom_Buttons**
    c. Location: **SearchView.CommandBar**
    d. Sort Order: **512**
    e. For Identity: **World**
5. Click the **Save** button, then right click the new Command Bar Section, and select **Open**.
6. Click the **Edit** button.
7. Click the **Add** button in the Command Bar Item grid, then search for the **searchview.commandbar.default.new** item. Click **OK** in the search dialog to add the item to the Command Bar Items grid.

8. Set the following properties for the Command Bar Item:
    a. Sort Order: **2000**
    b. Action: **Replace**
    c. For Identity: **World**
9. Click **Save**, then right click the Button, and select **Open**.
10. Click the "**…**" button in the toolbar and select **Create New Button** to create a new Button with the following properties:
    a. Name: **My_New_Button**
    b. Label: **New**
    c. Tooltip Template: **New {0}**
    d. Init Method: **cui_svcb_new_init**
    e. Click Method: **cui_ivicb_more_new_click**
    f. Image: **choose an icon** to display in the button
11. Click **Save**, then navigate back to the **Part_Search_Custom_Buttons** Command Bar Section tab.
12. In the Alternate property of the **searchview.commandbar.default.new** Command Bar Item relationship, enter **My_New_Button**.
13. Click **Save**, then navigate to **Design > Parts** in the TOC to see the custom "New" button in the search toolbar.

4.2 Menus

4.2.1 Add a button to a menu

“I want to add an action to the global user menu for all administrators.”

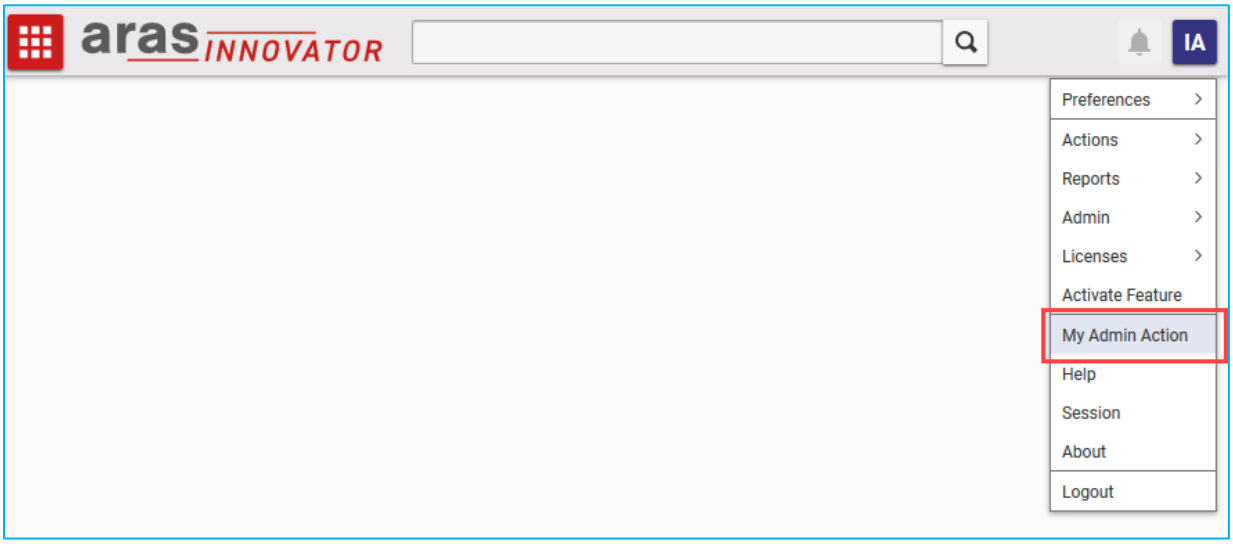

Figure 20. A new action appears in the user menu for administrators

1. Navigate to **Administration > Configuration > Client Presentation** in the TOC.
2. Click the **Search** button to run the Client Presentation search.
3. Click the **Global** item property in the single search result.
4. In the Command Bar Section grid, open the **com.aras.innovator.cui_default.mwh_header_user_menu** item with the MainWindowHeader location.
5. Click the **Edit** button.
6. Click the **New** button in the Command Bar Item grid, then select **Menu Button > OK** in the dialog to create a new menu button with the following properties:
   a. Name: **My_Custom_Admin_Action**
   b. Sort Order: **3100**
   c. Action: **Add**
   d. For Identity: **Administrators**
7. Right click the new button row and select **Open**.
8. Enter the following properties for the new Menu Button:
   a. Label: **My Admin Action**
   b. Parent Menu: **com.aras.innovator.cui_default.mwh_user_menu**
   c. Click Method: **select a Method** you want to run when the button is clicked
9. Click the **Save** button and close the Menu Button tab.

10. Click the **Save** button in the Command Bar Section tab.
11. Logout of Aras Innovator and log back in as a member of the Administrators identity.
12. Click the global user menu button in the top right corner of the window and select **My Admin Action** to run your custom action.

4.2.2 Disable a menu

“I want to disable the Share menu button for items that are not yet released.”

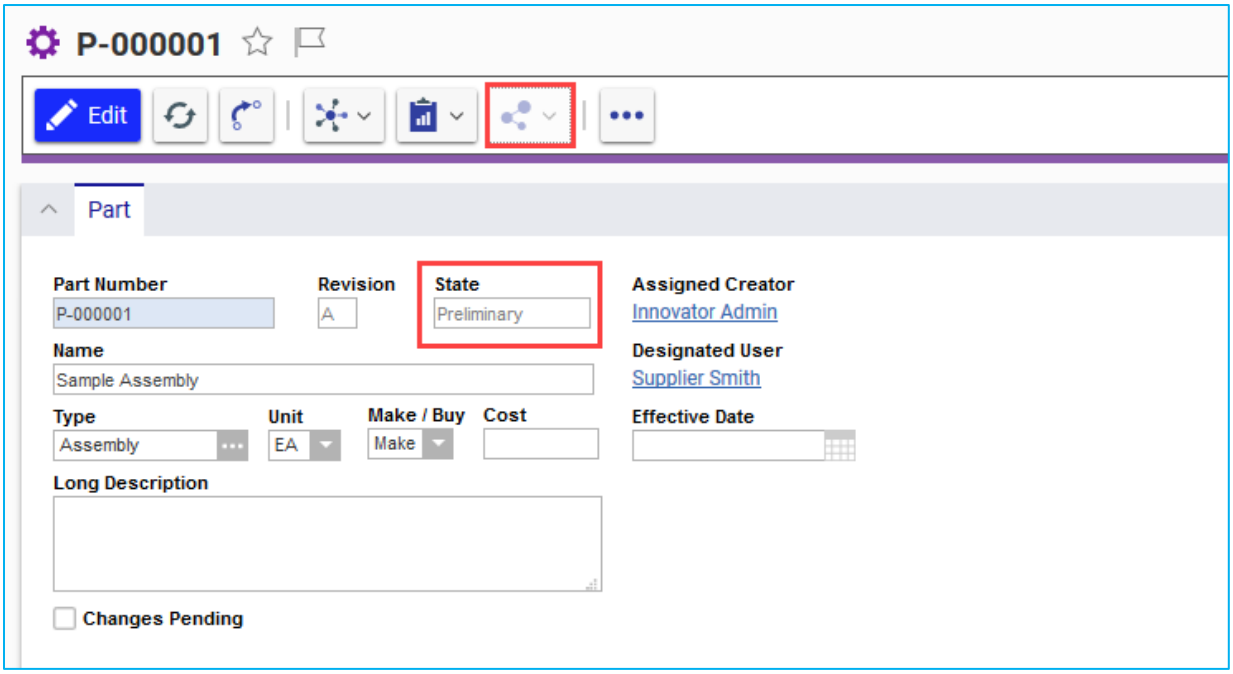

Figure 21. The default “share” menu is disabled for non-released items

1. Navigate to **Administration > Configuration > Client Presentation** in the TOC.
2. Click the **Search** button to run the Client Presentation search.
3. Click the **Global** item property in the single search result.
4. In the Command Bar Section grid, open the **itemview.itemcommandbar.default** item.
5. In the Command Bar Item grid, open the **itemview.itemcommandbar.default.share** item.
6. Click the **Edit** button.

7. Replace the Init Method with a custom Method that contains the following code:

```
if (!options.eventState) {
   options.eventState = aras.evalMethod('cui_reinit_calc_tearoff_states');
}

// get the release state of the context item
var contextItem = aras.getItemById(options.itemTypeName, options.itemId);
var isReleased = aras.getItemProperty(contextItem,'is_released','0');

// disable the menu when the item is not released
return {
    'hidden': options.eventState.isNew,
    'disabled': (isReleased === '0')
};
```

8. Click **Save** and close the Menu item.
9. Open any non-released item to confirm that the Share menu button is disabled. The button will be enabled for any released item.

4.2.3 Add a submenu to a menu

"I want to move 'Structure Browser' and 'Where Used' to a submenu of the main grid context menu."

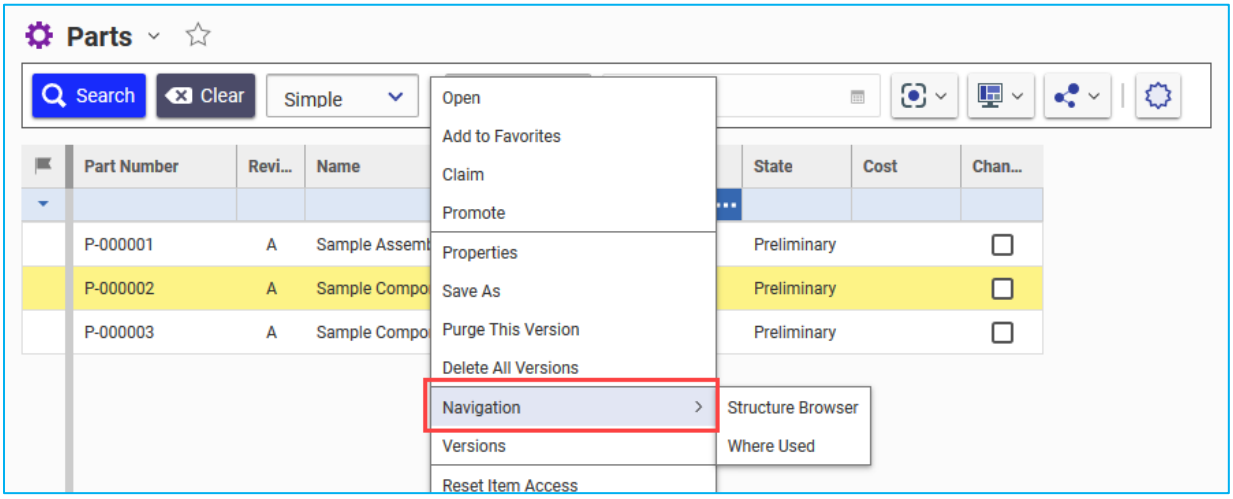

Figure 22. The default structure browser and where used actions appear in a new submenu

1. Navigate to **Administration > Configuration > Client Presentation** in the TOC.
2. Click the **Search** button to run the Client Presentation search.
3. Click the **Global** item property in the single search result.
4. In the Command Bar Section grid, open the **com.aras.innovator.cui_default.popup_menu_for_item_grid** item.
5. Click the **Edit** button.

6. Click the **New** button in the Command Bar Item grid, then select **Menu > OK** in the dialog to create a new menu button with the following properties:
    a. Name: **My_Nav_Menu**
    b. Init Method: **initPopupItemInItemsGrid**
    c. Sort Order: **1300**
    d. Action: **Add**
    e. For Identity: **World**
7. Right click the new button row and select **Open**.
8. Set the following properties:

    Label: **Navigation**
9. Click **Save** and close the new Menu tab.
10. Move the Structure Browser action to the new submenu:
    a. On the Command Bar Section form, open the **com.aras.innovator.cui_default.pmig_Structure Browser** Menu from the Command Bar Item grid.
    b. Click the **Edit** button.
    c. Set the Parent Menu property to **My_Nav_Menu**.
    d. **Save** and close the Structure Browser Menu item tab.
11. Move the Where Used action to the new submenu:
    a. On the Command Bar Section form, open the **com.aras.innovator.cui_default.pmig_Where Used** Menu from the Command Bar Item grid.
    b. Click the **Edit** button.
    c. Set the Parent Menu property to **My_Nav_Menu**.
    d. **Save** and close the Where Used Menu item tab.
12. Click the **Save** button for the Command Bar Section.
13. Open any search grid from the TOC and right click on a row to see the new “Navigation” submenu.

4.2.4 Display an icon on a menu button

“I want to show an icon next to the ‘Delete’ action in the main grid context menu.”

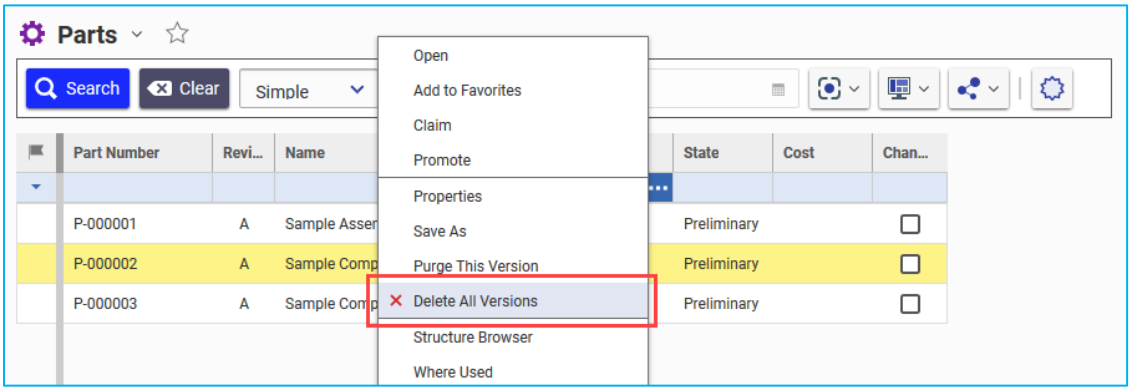

Figure 23. An icon appears next to the delete action in the main grid menu

1. Navigate to **Administration > Configuration > Client Presentation** in the TOC.
2. Click the **Search** button to run the Client Presentation search.
3. Click the **Global** item property in the single search result.
4. In the Command Bar Section grid, open the **com.aras.innovator.cui_default.popup_menu_for_item_grid** item.
5. Click the **Edit** button.
6. In the Command Bar Item grid, open the **com.aras.innovator.cui_default.pmig_Delete** item.
7. Click the **Edit** button.
8. Use the **Image** property to choose the icon you want to show in the menu.
9. Click **Save** and close the Menu item.
10. Click **Save** and close the Command Bar Section item.
11. Open any search grid from the TOC and right click on a row to see your selected icon appear next to the “Delete All Versions” action.

4.3 Shortcuts

4.3.1 Add an item form keyboard shortcut

“I want a shortcut to allow users to create a new item from an item form.”

1. Navigate to **Administration > Configuration > Client Presentation** in the TOC.
2. Click the **Search** button to run the Client Presentation search.
3. Click the **Global** item property in the single search result.
4. In the Command Bar Section grid, open the **com.aras.innovator.cui_default.itemWindowShortcuts** item.
5. Click the **Edit** button.

6. Click the **New** button in the Command Bar Item grid, then select **Shortcut > OK** in the dialog to create a new keyboard shortcut with the following properties:
   a. Name: **New_Item**
   b. Sort Order: **1024**
   c. Action: **Add**
   d. For Identity: **World**
7. Right click the new button row and select **Open**.
8. Set the Shortcut property to **alt+ctrl+n**.
9. Set the Handler property to a custom Method containing the following code:

```
var itemtype = window.itemTypeName;
if (itemtype) {
    aras.uiNewItemEx(itemtype);
}
```

10. Click **Save** and close the new Shortcut tab.
11. **Save** and close the Command Bar Section tab.
12. Open an item, like a Part, and enter the keyboard shortcut **CTRL+ALT+N**. A new Part item will open in a new tab.

4.4 Table of Contents

4.4.1 Add a new ItemType Button to the TOC

“I want the All Employees identity to see a TOC button for Purchase Orders.”

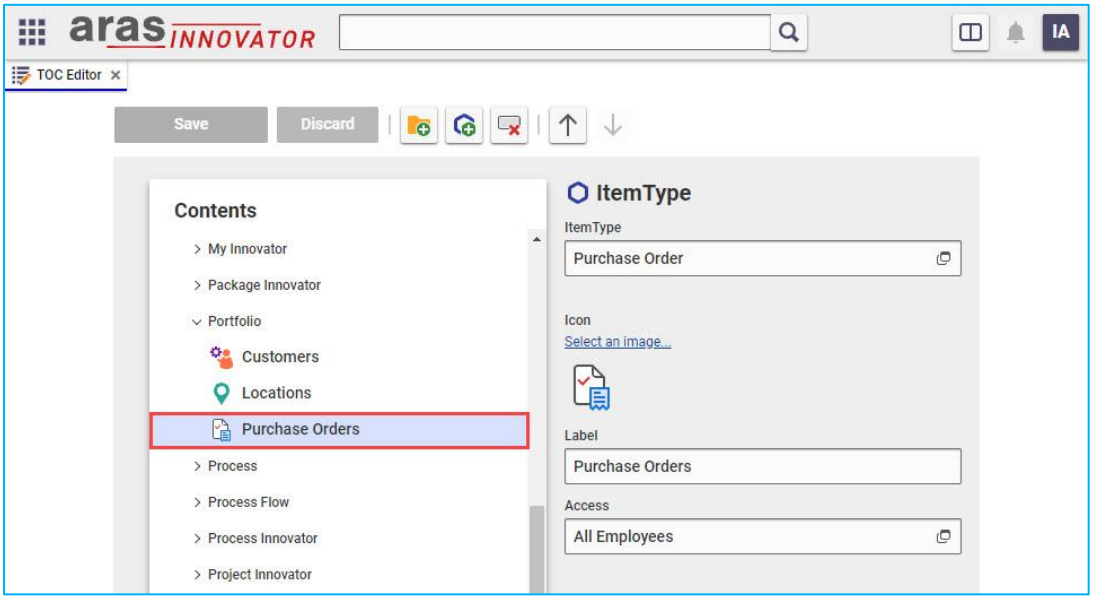

Figure 24. Configuring a new button for the custom ItemType “Purchase Orders”

1. Navigate to **Administration > Configuration > TOC Editor** in the TOC.
2. Click the **Add ItemType** button in the toolbar.
3. In the search dialog, choose the ItemType you want to add and click **OK**.

> **Note:** A new entry for your ItemType will appear in the Contents pane of the editor. If no categories were selected when you added the button, the new entry will appear at the bottom of the list. If a category was selected, the new entry will appear in that category.

4. In the righthand pane, you may customize any of the following properties:
   a. ItemType: This ItemType will appear in the navigation pane when a user clicks the button.
   b. Icon: This image is displayed as an icon on the button.
   c. Label: This multilingual property determines the button text.
   d. Access: This property determines which identity will see this button in the TOC. If you want multiple identities to see a button for this ItemType, pick a group identity that includes all the identities that should have access, or add a button for each identity.
5. If you want to change where the button appears in the TOC, you can drag and drop it to your preferred location. Alternatively, you can select the button and use the **Move Up**/**Move Down** buttons in the toolbar.
6. When you're finished making your changes, click **Save** and close the tab for the TOC Editor.

4.4.2 Add a new category to the TOC

"I want to add a 'Finance' category for my Invoices and Purchase Orders."

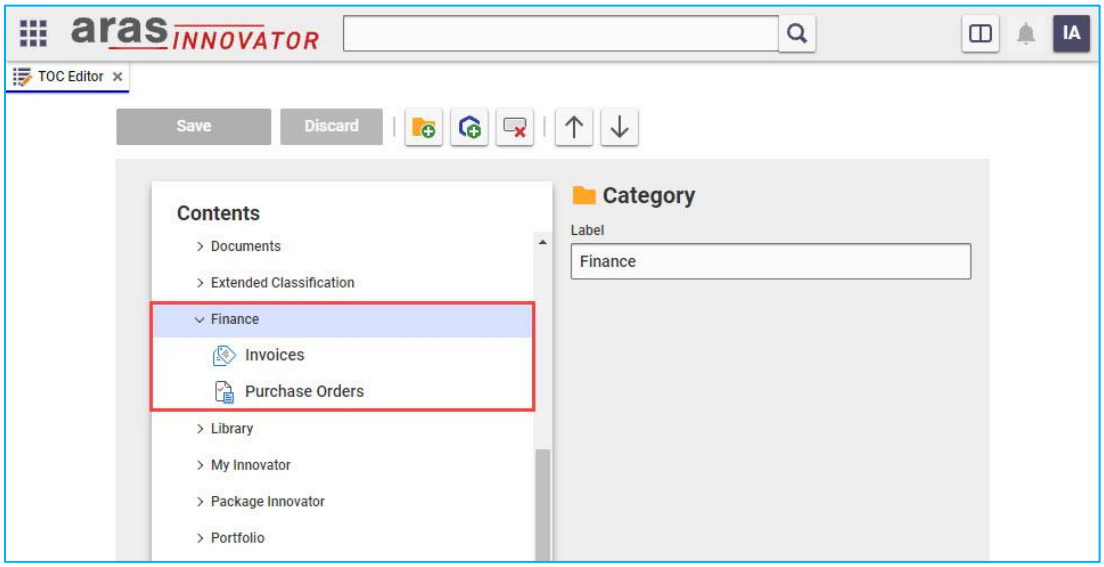

Figure 25. Adding a Finance category to group two custom ItemType buttons

1. Navigate to **Administration > Configuration > TOC Editor** in the TOC.

2. Click the **Add Category** button in the toolbar.

Note: A new category will appear in the Contents pane of the editor. If no existing categories were selected when you added the new category, the new entry will appear at the bottom of the list. If a category was selected, the new entry will appear in that category.

3. In the righthand pane, you can customize the multilingual **Label** property to set the category text.
4. If you want to change where the category appears in the TOC, you can drag and drop it to your preferred location in the Contents pane. Alternatively, you can select the category and use the **Move Up**/**Move Down** buttons in the toolbar.
5. When you're finished making your changes, click **Save** and close the tab for the TOC Editor.

4.4.3 Set a custom button label and icon for a specific identity

"I want the All Suppliers identity to see a different label and icon for Purchase Orders."

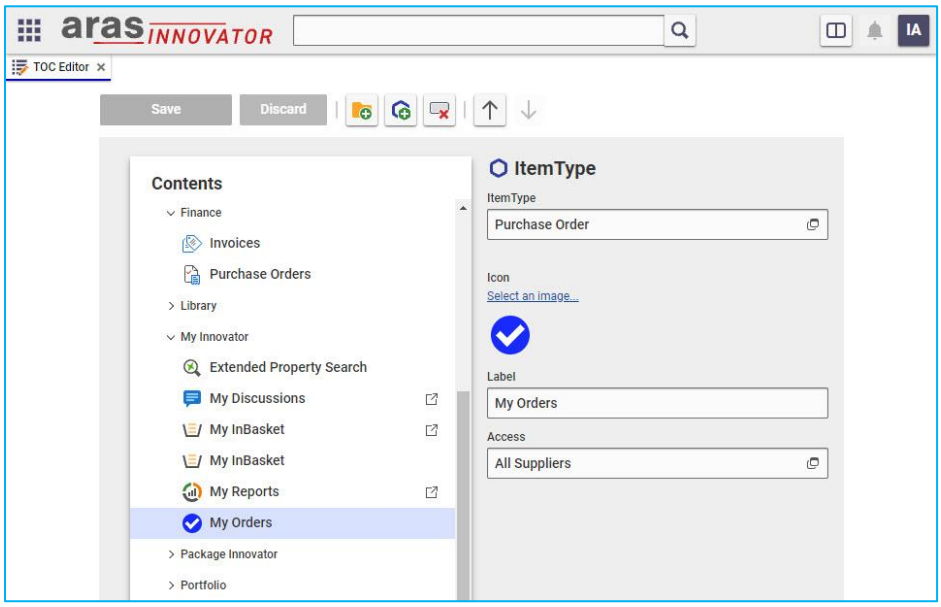

Figure 26. Configuring a custom label and icon for All Suppliers' "Purchase Orders"

Note: The steps in this section are largely the same as section 2.4.1 "Add a new ItemType button to the TOC". Here, we are adding two buttons with the same ItemType but different labels, icons, and access identities.

1. Navigate to **Administration > Configuration > TOC Editor** in the TOC.
2. Click the **Add ItemType** button in the toolbar.
3. In the search dialog, choose the ItemType you want to add and click **OK**.
4. In the righthand pane, you may customize the following properties or leave them set to their default values:
   a. **Icon**: This image is displayed as an icon on the button.
   b. **Label**: This multilingual property determines the button text.

c. **Access**: This property determines which identity will see this button in the TOC. If you want multiple identities to see a button for this ItemType, pick a group identity that includes all the identities that should have access, or add a button for each identity.

5. If you want to change where the button appears in the TOC, you can drag and drop it to your preferred location. Alternatively, you can select the button and use the **Move Up**/**Move Down** buttons in the toolbar.
6. Repeat steps 2-5, choosing a different **Access** identity, **Icon**, and **Label** in step 4.
7. When you're finished making your changes, click **Save** and close the tab for the TOC Editor.

4.4.4 Sort the TOC Buttons Alphabetically

"I want all TOC categories and buttons to display in alphabetical order."

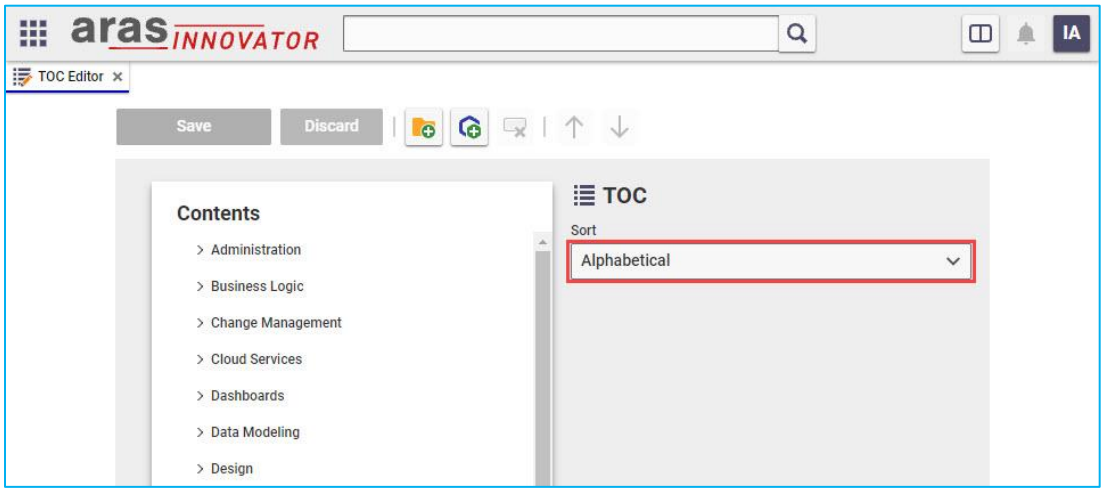

Figure 27. Configuring the TOC for alphabetical sort order

1. Navigate to Administration > Configuration > TOC Editor in the TOC.
2. In the righthand pane, set the Sort property to *Alphabetical*.

Note: The TOC sort order is set to Alphabetical by default.

3. When you're finished making your changes, click Save and close the tab for the TOC Editor.

4.4.5 Arrange the TOC Buttons in a Custom Order

“I want the 'My Innovator' category to appear at the top of the TOC.”

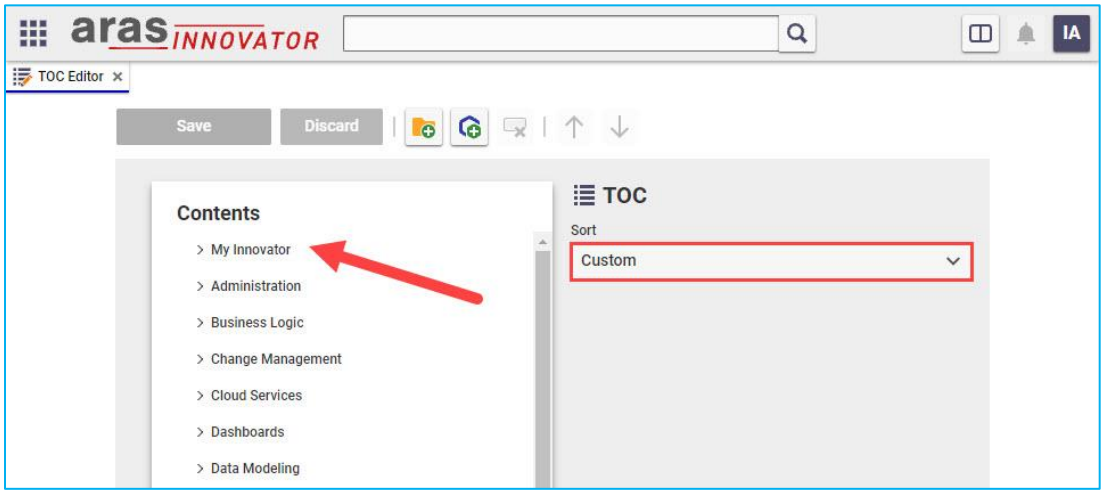

Figure 28. Configuring the TOC for a custom sort order

1. Navigate to **Administration > Configuration > TOC Editor** in the TOC.
2. In the righthand pane, set the Sort property to *Custom*.

Note: The TOC sort order is set to Alphabetical by default.

3. When you’re finished making your changes, click Save and close the tab for the TOC Editor.

4.4.6 View the TOC for a Specific Identity

“I want to see what the TOC looks like for user Mike Miller.”

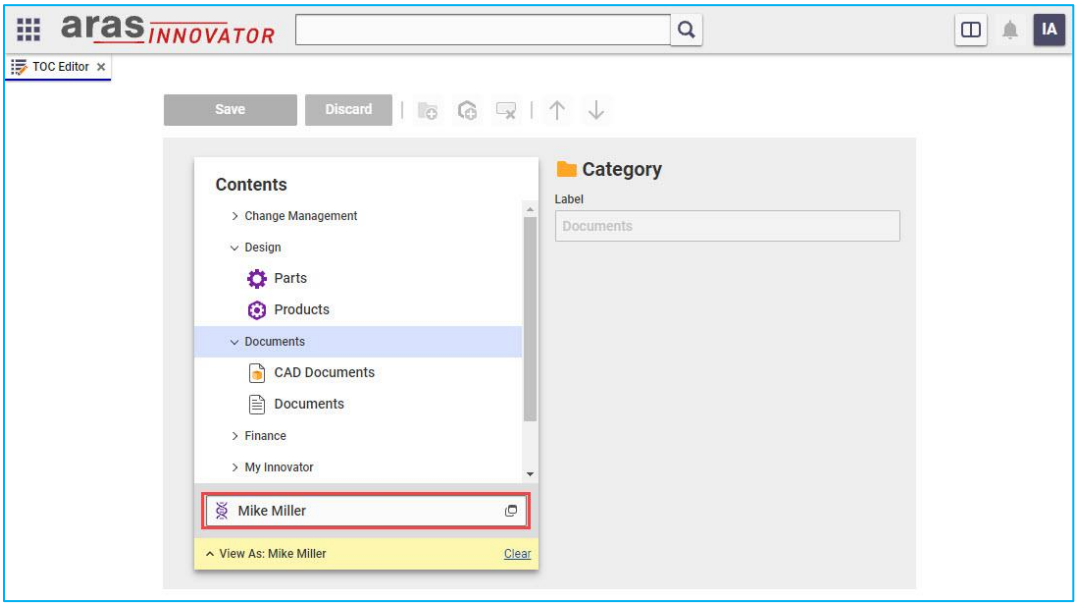

Figure 29. Viewing the content for Mike Miller’s TOC

1. Navigate to **Administration > Configuration > TOC Editor** in the TOC.
2. Click View As at the bottom of the contents pane.
3. Enter or search for an identity in the Identity field. This can be a group identity like “All Employees” or an alias identity for a user like “Innovator Admin”.
4. The contents pane will now display the TOC for your chosen identity. When you are done viewing the content, click **Clear** to exit viewing mode.

Note: Viewing mode is read-only. To make edits to the TOC, you must clear the View As field.

4.4.7 Display a Form with a TOC Button

“I want the user to see a special form when they click a button in the TOC.”

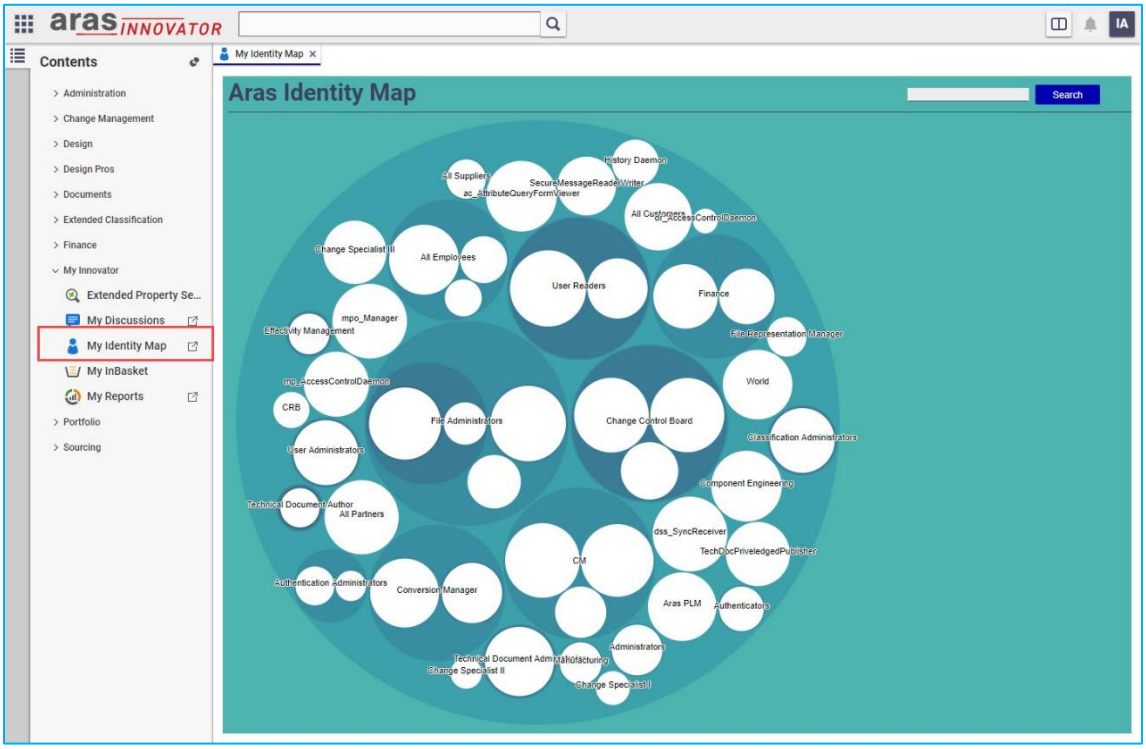

Figure 30. The user clicks “My Identity Map” to see an interactive graph in a custom form

1. Navigate to **Administration > Configuration > TOC Editor** in the TOC.
2. Click the **Add Form** button in the toolbar.
3. In the search dialog, choose the Form to display and click **OK**.
4. In the righthand pane, you may customize the following properties or leave them set to their default values:
   a. **Icon**: This image is displayed as an icon on the button.
   b. **Label**: This multilingual property determines the button text.
   c. **Access**: This property determines which identity will see this button in the TOC. If you want multiple identities to see a button for this ItemType, pick a group identity that includes all the identities that should have access, or add a button for each identity.

5. If you want to change where the button appears in the TOC, you can drag and drop it to your preferred location. Alternatively, you can select the button and use the **Move Up**/**Move Down** buttons in the toolbar.

4.4.8 Display an HTML Page with a TOC Button

"I want the user to see an HTML page when they click a button in the TOC."

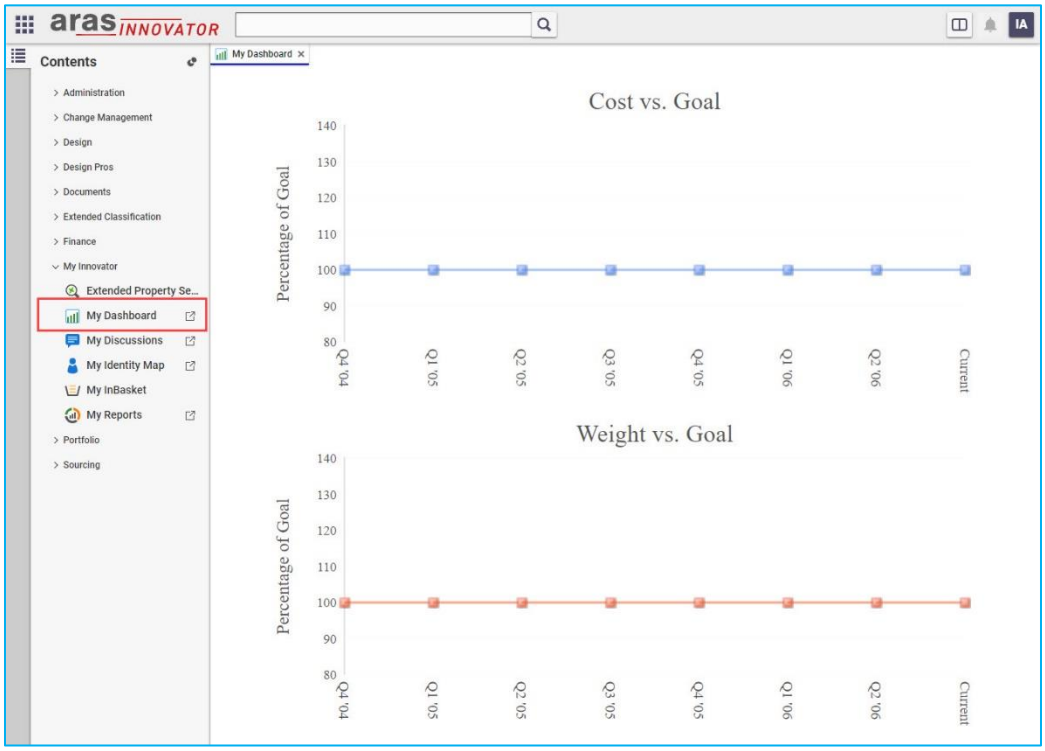

Figure 31. The user clicks "My Dashboard" to see a graph in a custom HTML page

1. Navigate to **Administration > Configuration > TOC Editor** in the TOC.
2. Click the **Add Page** button in the toolbar.
3. In the righthand pane, you may customize the following properties or leave them set to their default values:
   a. **Start Page:** Specifies the path of an HTML page to display when the button is clicked. This should be an HTML file in the code tree, and the path should be relative to the *Innovator/Client/scripts* directory.
   b. **Parameters:** Optional. Passes parameters for the HTML page to use.
   c. **Icon**: This image is displayed as an icon on the button.
   d. **Label**: This multilingual property determines the button text.
   e. **Access**: This property determines which identity will see this button in the TOC. If you want multiple identities to see a button for this ItemType, pick a group identity that includes all the identities that should have access, or add a button for each identity.
4. If you want to change where the button appears in the TOC, you can drag and drop it to your preferred location. Alternatively, you can select the button and use the **Move Up**/**Move Down** buttons in the toolbar.

4.4.9 View the TOC in a Specific Language

“I want to see what the TOC looks like for a Change Specialist I member in Japan.”

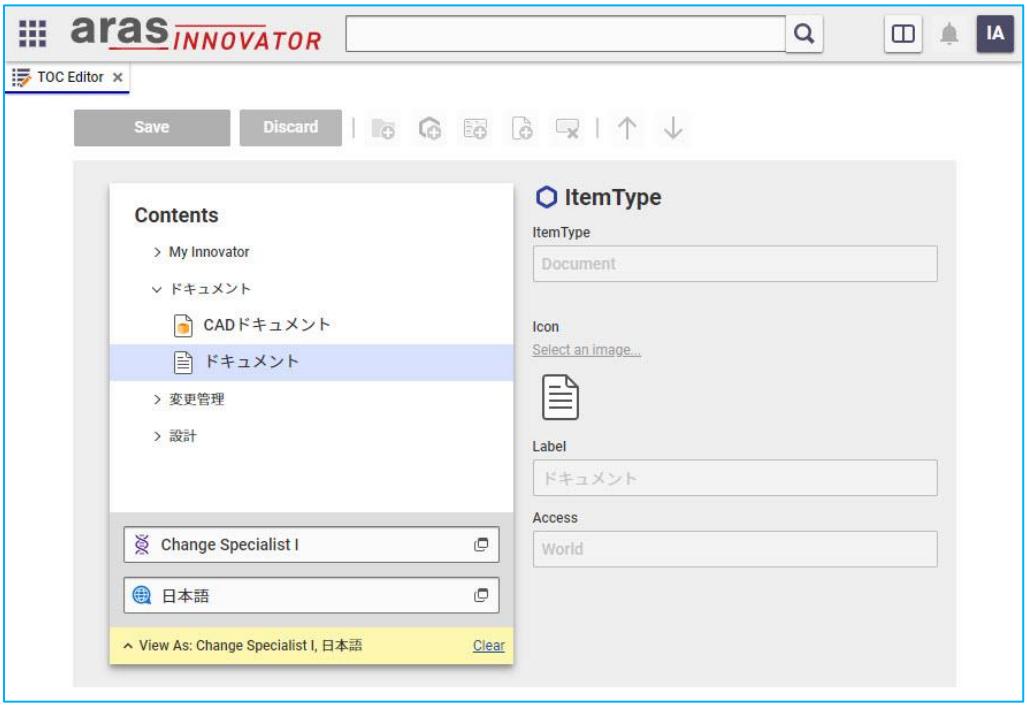

Figure 32. Viewing the content for Mike Miller’s TOC

1. Navigate to **Administration > Configuration > TOC Editor** in the TOC.
2. Click View As at the bottom of the contents pane.
3. Enter or search for an identity in the Identity field. This can be a group identity like “All Employees” or an alias identity for a user like “Innovator Admin”.
4. The contents pane will now display the TOC for your chosen identity.
5. Enter or search for a language in the Language field.
6. The contents pane will now display the identity’s TOC content in the specified language. When you are done viewing the content, click **Clear** to exit viewing mode.

Note:	Viewing mode is read-only. To make edits to the TOC, you must clear the View As field.
Note:	The Language filter will be enabled only when an Identity is specified and there are two or more languages configured in the Aras Innovator database. For more details on configuring Aras Innovator for multiple languages, please refer to the Configuring Internationalization documentation.

4.4.10 Add a Button to the TOC from an ItemType

“I want to quickly add a new TOC button while defining my ItemType.”

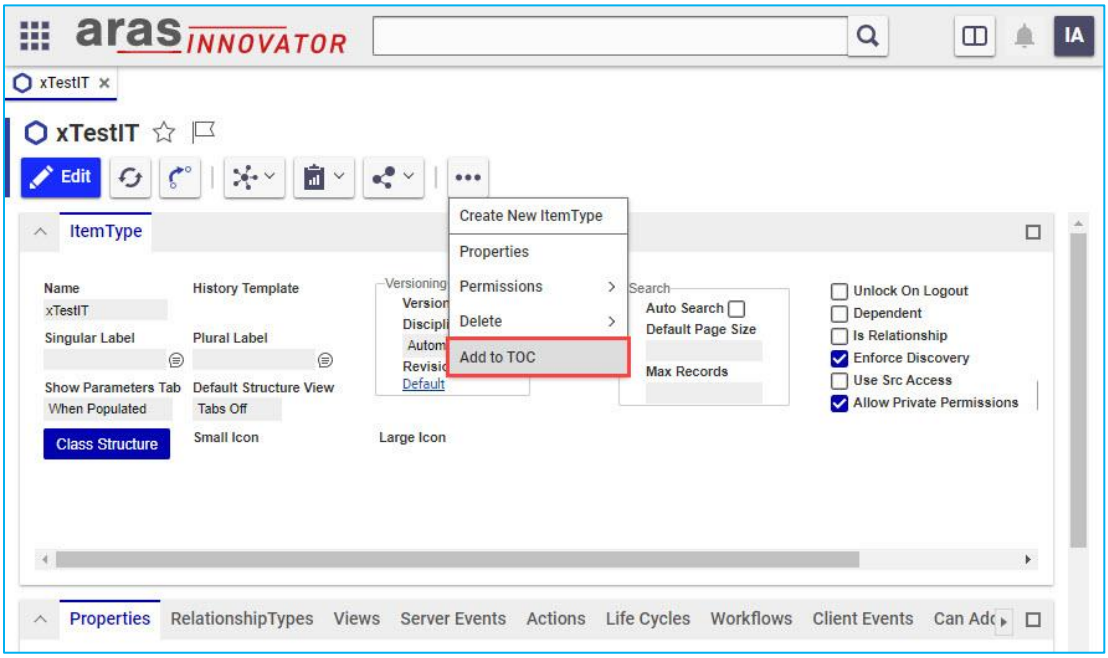

Figure 33. Quickly create a new TOC button with the “Add to TOC” action

1. If the ItemType isn’t already open:
   a. Navigate to **Administration > ItemTypes** in the TOC.
   b. Search for the ItemType you want to work with.
   c. Double click the ItemType in the grid to view it.
2. Select the **‘…’** button from the ItemType toolbar and click **Add to TOC**. This action will create a new ItemType button in the TOC with the following settings:
   a. ItemType: Current **ItemType**
   b. Icon: **Small Icon** of the ItemType
   c. Label: **Plural Label** of the ItemType (or ItemType name if no plural label is defined)
   d. Access: **Administrators**
3. To see the new TOC button, navigate to the TOC and scroll to the bottom. This is the default location for new buttons added via the **Add to TOC** action.
4. To change the default settings for the new TOC button, navigate to **Administration > Configuration > TOC Editor** in the TOC.

5. Scroll to the bottom of the contents pane and select the new ItemType button.

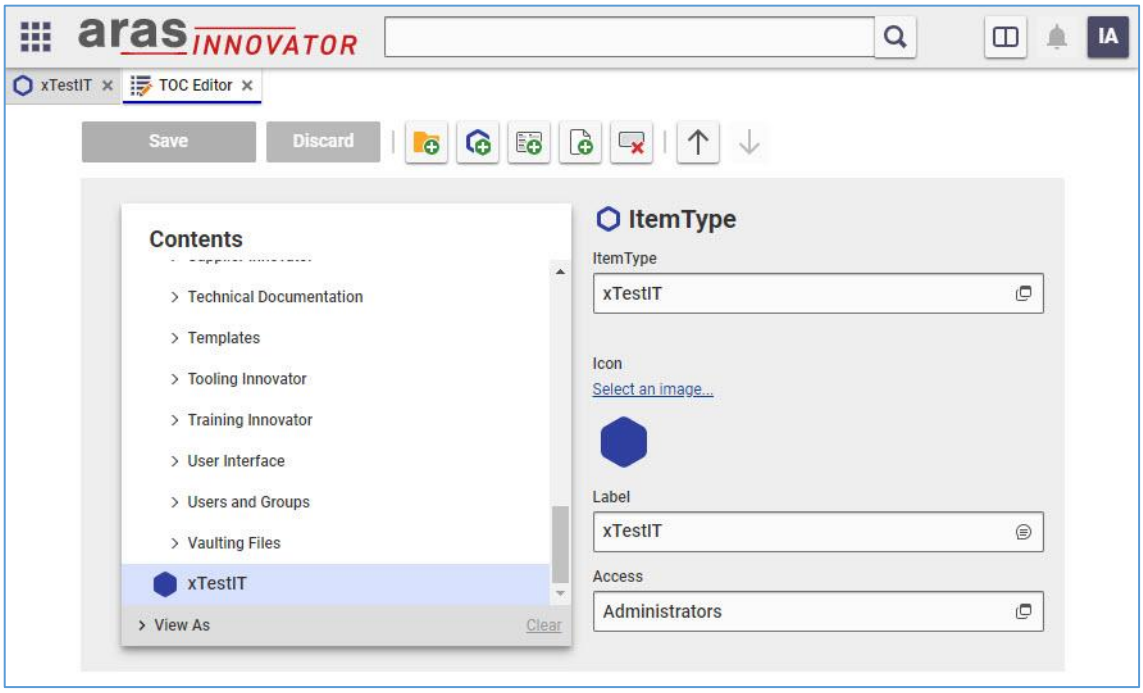

Figure 34. Default settings for a button created from the xTestIT ItemType

6. Change any default settings as needed.

Note: Changing the settings of an ItemType button on the TOC will not affect the settings of the associated ItemType. For example, changing the button icon will not change the icons defined on the ItemType.

7. If you want to change where the button appears in the TOC, you can drag and drop it to your preferred location. Alternatively, you can select the button and use the **Move Up**/**Move Down** buttons in the toolbar.

8. When you're finished making your changes, click **Save** and close the tab for the TOC Editor.

4.5 Item Views

4.5.1 Add a tab to the first accordion

“I want to display the Part BOM tab in the top accordion for all users.”

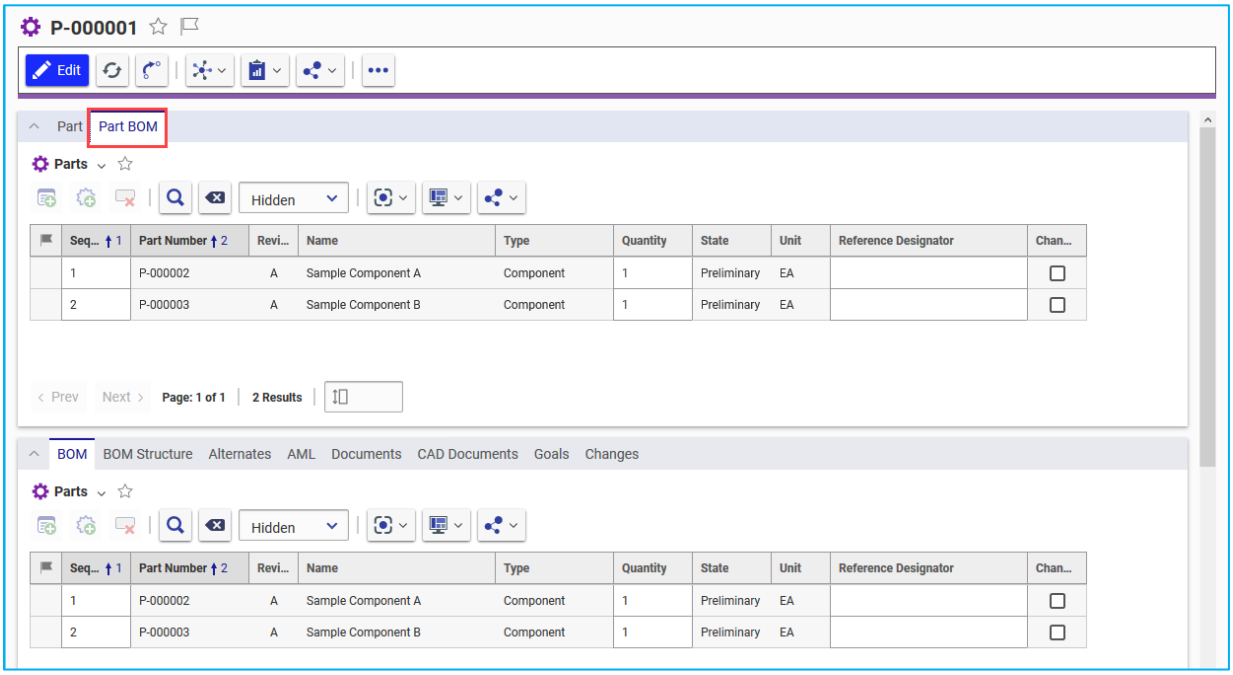

Figure 35. The Part BOM tab is added to the top accordion on the Part item view

1. Open the **Part** ItemType and select the **Client Style** relationship tab.
2. Right click the entry in the grid and select **Open** to open the Presentation Configuration for the Part ItemType.
3. Click the **Edit** button.
4. Select the **cui_PresentConfigWinSection** tab.
5. Click the **New** button in the grid toolbar to create a new window section with the following properties:
   a. Classification: **Data Model**
   b. Name: **Accordion1**
   c. Location: **ItemView**
   d. For Identity: **World**
6. Click the **Save** button, then right click the new window section, and select **Open**.
7. Click the **Edit** button.

8. Click the **New** button in the grid toolbar to create a new control with the following properties:
   a. Type: **Tab Element Control**
   b. Name: **Acc1_BOM_Tab**
   c. Label: **Part BOM**
   d. Additional Data: **{"relTypeId": "159C6D88795B4A86864420863466F728"}**
   e. Parent: **ItemView.FormAccordionTabs**
   f. Sort Order: **1024**
   g. Action: **Add**
   h. For Identity: **World**
9. Click **Save**, then navigate to **Design > Parts** in the TOC and open a Part item to see the new Part BOM tab in the top accordion.

Note: The default BOM tab will still appear in the relationship accordion. See the next section for steps to hide the default BOM tab from the second accordion.

4.5.2 Hide a tab from the second accordion

"I want to display the Part BOM tab in the top accordion instead of the relationship accordion."

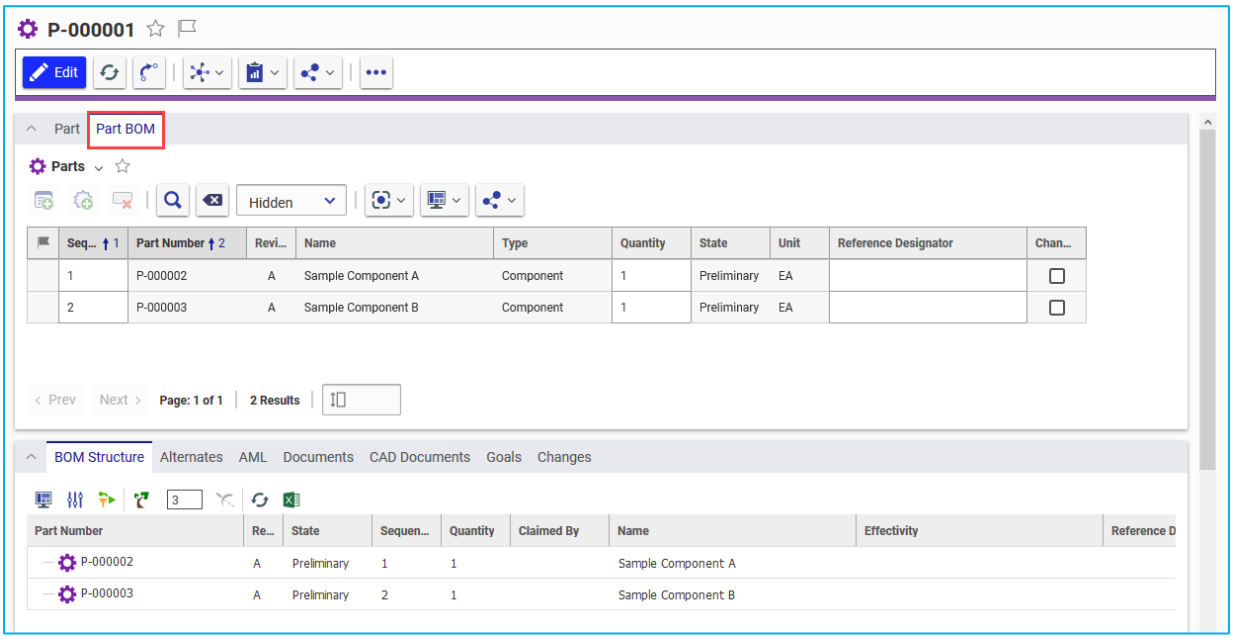

Figure 36. The Part BOM *only* appears in the top accordion on the Part item view

1. Follow the steps in the previous section to add the Part BOM tab to the top accordion. The following steps outline how to hide the default BOM tab from the relationship accordion.
2. Navigate to **Administration > RelationshipTypes** in the TOC.
3. Search for the **Part BOM** RelationshipType and open the item.
4. Click **Edit**.

5. Check off the **Hide In All** box to prevent the default tab from appearing in a relationship accordion.
6. Click **Save**, then navigate to **Design > Parts** in the TOC and open a Part item to confirm the default BOM tab doesn't appear in the relationship accordion.

4.5.3 Add a third accordion with a tab

"I want to show the Documents and CAD Documents tabs in a third accordion."

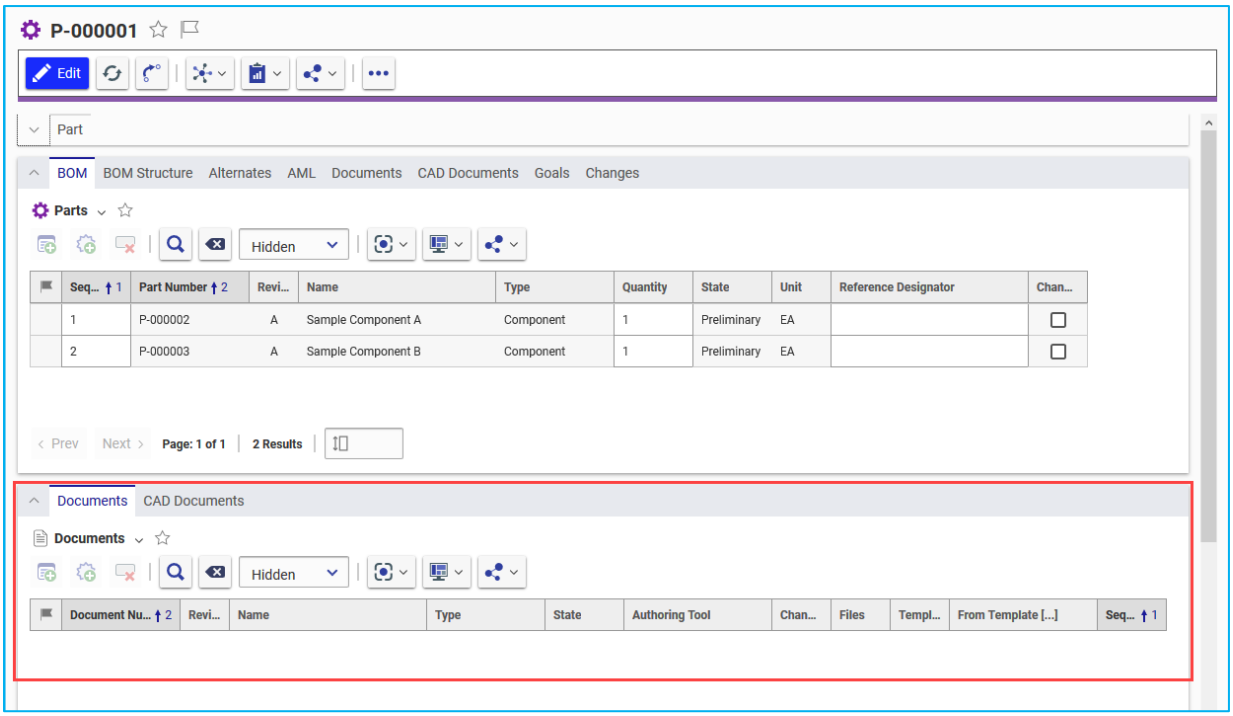

Figure 37. A third accordion is added displaying the Document and CAD Document tabs

1. Open the **Part** ItemType and select the **Client Style** relationship tab.
2. Right click the entry in the grid and select **Open** to open the Presentation Configuration for the Part ItemType.
3. Click the **Edit** button.
4. Select the **cui_PresentConfigWinSection** tab.
5. Click the **New** button in the grid toolbar to create a new window section with the following properties:
   a. Classification: **Data Model**
   b. Name: **Accordion3**
   c. Location: **ItemView**
   d. For Identity: **World**
6. Click the **Save** button, then right click the new window section, and select **Open**.
7. Click the **Edit** button.

8. Click the **New** button in the grid toolbar to create a new control with the following properties:
    a. Type: **Accordion Element Control**
    b. Name: **Acc3_Container**
    c. Additional Data: **{"cssClass": "aras-item-view__relationship-accordion"}**
    d. Sort Order: **2000**
    e. Action: **Add**
    f. For Identity: **World**
9. Click **Save**.
10. Click the **New** button in the grid toolbar to create a new control with the following properties:
    a. Type: **Tab Container Control**
    b. Name: **Doc_Tab_Container**
    c. Additional Data: **{"attributes" : {"slot": "header"}}**
    d. Parent: **Acc3_Container**
    e. Sort Order: **2050**
    f. Action: **Add**
    g. For Identity: **World**
11. Click **Save**.
12. Click the **New** button in the grid toolbar to create a new control with the following properties:
    a. Type: **Tab Element Control**
    b. Name: **Document_Tab**
    c. Label: **Documents**
    d. Additional Data: **{"relTypeId": "09CBB0294FAB477AA7300906DC035462"}**
    e. Parent: **Doc_Tab_Container**
    f. Sort Order: **3000**
    g. Action: **Add**
    h. For Identity: **World**
13. Click the **New** button in the grid toolbar to create a new control with the following properties:
    a. Type: **Tab Element Control**
    b. Name: **CAD_Tab**
    c. Label: **CAD Documents**
    d. Additional Data: **{"relTypeId": "C65FB7DC8C3A4EFC97A6EE8196C0B448"}**
    e. Parent: **Doc_Tab_Container**
    f. Sort Order: **3050**
    g. Action: **Add**
    h. For Identity: **World**

14. Click **Save**, then navigate to **Design > Parts** in the TOC and open a Part item to see Document and CAD Document tabs appear in a new accordion at the bottom of the page.

4.6 Item View Sidebar

4.6.1 Display the ad hoc Graph View from an Item View Sidebar button

“I want to add a button to the sidebar of all items that will show an ad hoc Graph View in the item view.”

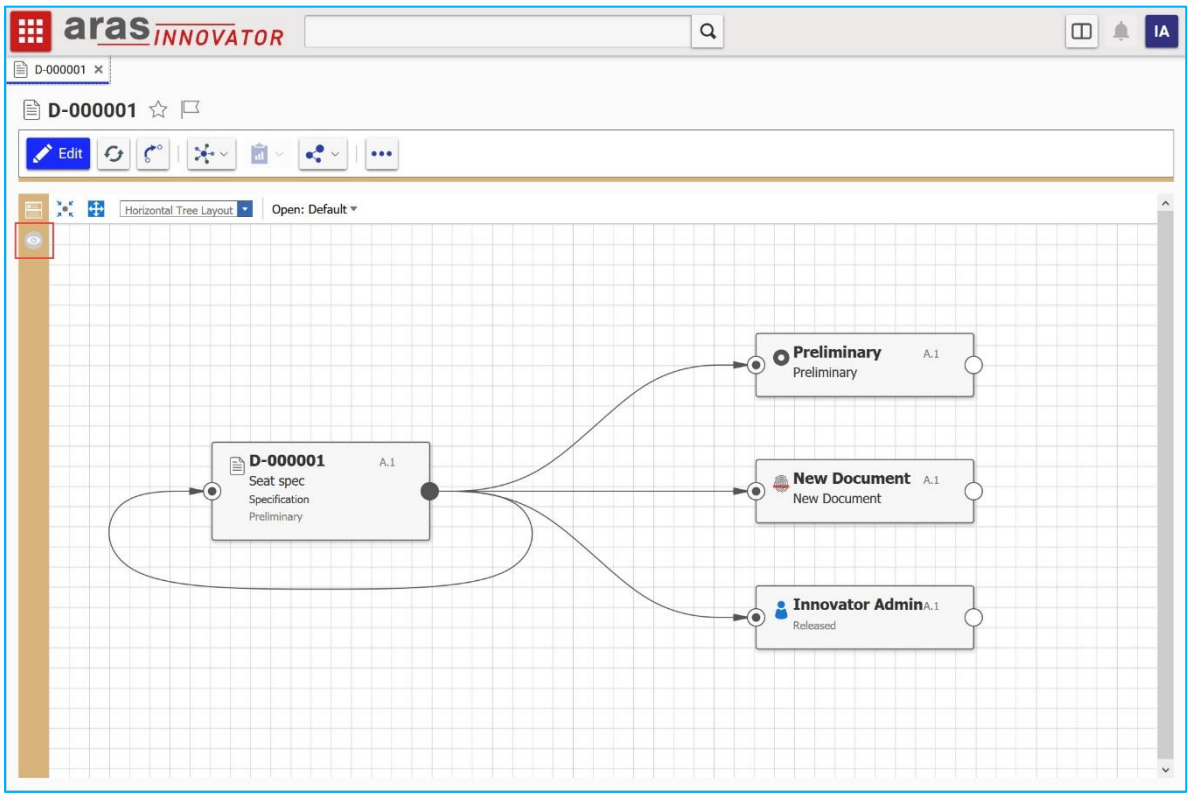

Figure 38. A sidebar button for displaying the ad hoc graph is added to the global Item View definition

1. Navigate to **Administration > Configuration > Client Presentation** in the TOC.
2. Click the **Search** button to run the Client Presentation search.
3. Click the **Global** item property in the single search result.
4. In the Command Bar Section grid, open the **Global Sidebar** item.
5. Click the **Edit** button.
6. Click the **New** button in the Command Bar Item grid, then select **Button > OK** in the dialog to create a new button with the following properties:
   a. Name: **My_Ad_Hoc_Graph_Button**
   b. Sort Order: **256**
   c. Action: **Add**
   d. For Identity: **World**

7. Right click the new button row and select **Open**.
8. Enter the following properties for the new Menu Button:
   a. Click Method: **sidebar_default_gv_click**
   b. Image: choose the icon you want to show when the sidebar button is inactive
   c. Additional Image: choose the icon you want to show when the button is active
9. Click the **Save** button and close the Button tab.

4.6.2 Display a query-based Graph View from an Item View Sidebar button

"I want to display the Part BOM Graph View when the user clicks a sidebar button on a Part."

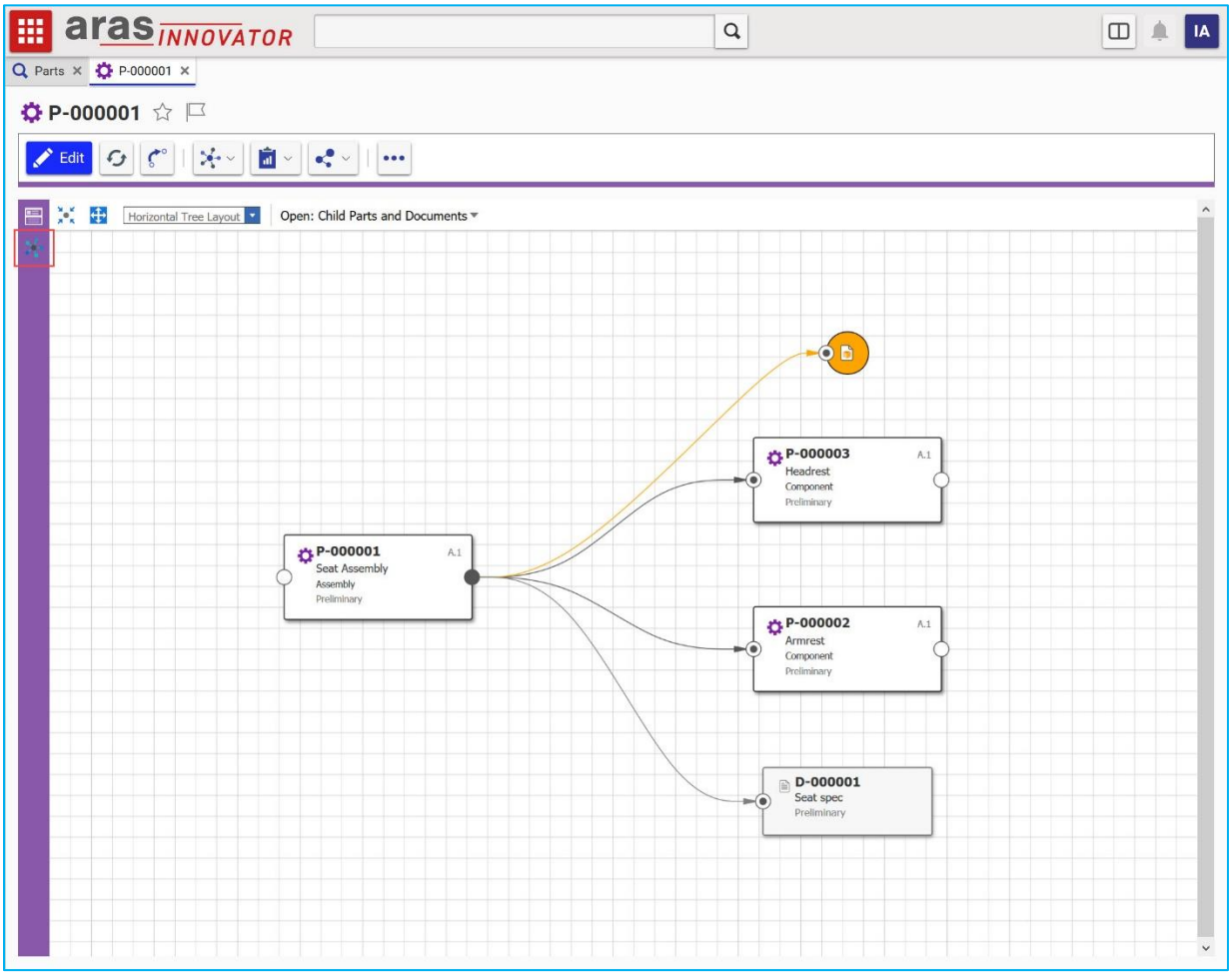

Figure 39. A sidebar button for displaying a specific graph is added to the Part form

1. Open the **Part** ItemType and select the **Client Style** relationship tab.
2. Right click the entry in the grid and select **Open** to open the Presentation Configuration for the Part ItemType.
3. Click the **Edit** button.

4. Click the **New** button in the Command Bar Section grid to create a new section with the following properties:
    a. Classification: **Data Model**
    b. Name: **Part_Custom_Sidebar**
    c. Location: **ItemWindowSidebar**
    d. Sort Order: **256**
    e. For Identity: **World**
5. Click the **Save** button, then right click the new Command Bar Section, and select **Open**.
6. Click the **Edit** button.
7. Click the **New** button in the Command Bar Item grid to create a new item with the following properties:
    a. Type: **Button**
    b. Name: **My_Graph_Button**
    c. Sort Order: **1100**
    d. Action: **Add**
    e. For Identity: **World**
8. Click **Save**, then right click the new Button, and select **Open**.
9. Click the **Edit** button.
10. Set the following properties on the item form:
    a. Additional Data: **{ "gvdId": "<graph view id>" }**
    b. Click Method: **sidebar_default_gv_click**
    c. Image: **choose the icon you want to show when the sidebar button is inactive**
    d. Additional Image: **choose the icon you want to show when the button is active**
11. Click **Save**, then navigate to **Design > Parts** in the TOC.
12. Open a Part item to see the new button in the sidebar.
13. Click the new sidebar button to view the specified Graph View.

4.6.3 Display a Tree Grid View from an Item View Sidebar button

“I want to display the BOM as a Tree Grid View when the user clicks a sidebar button on a Part.”

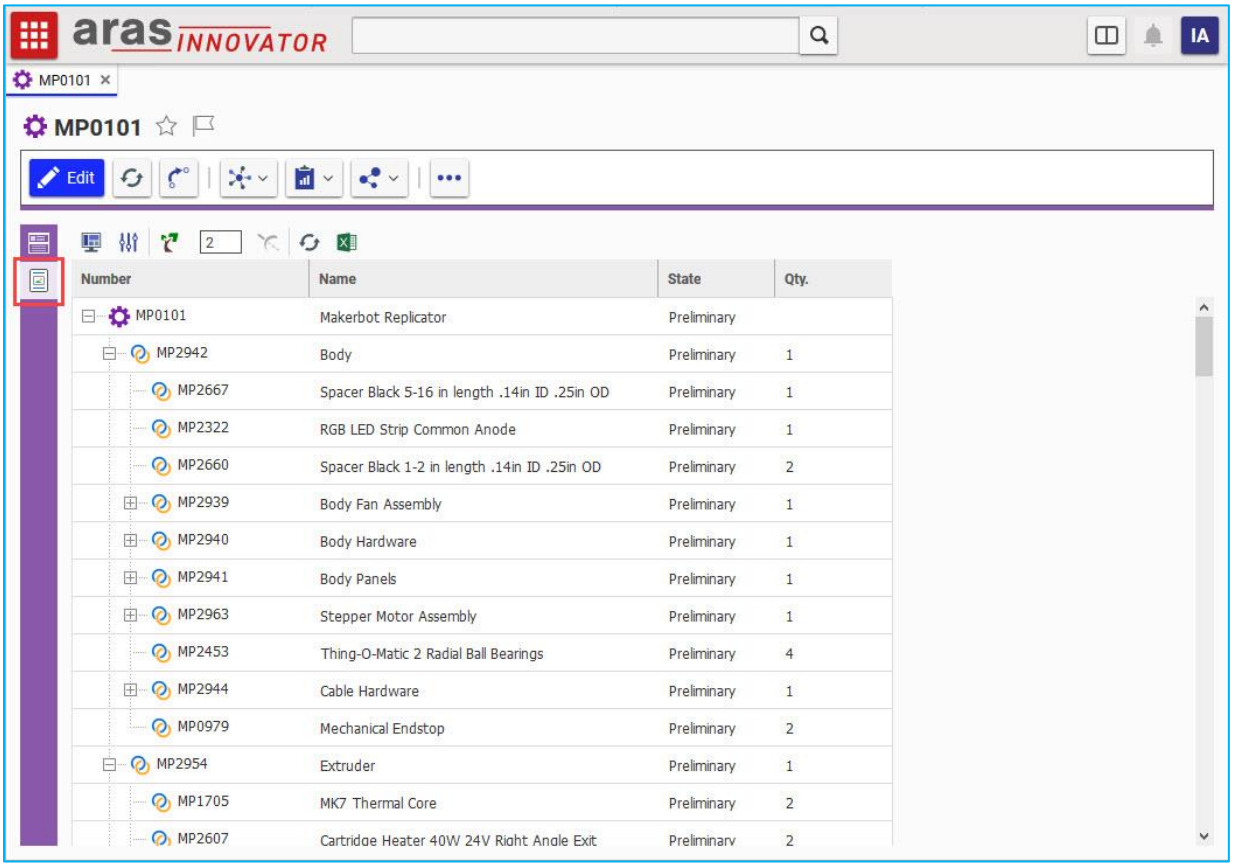

Figure 40. A sidebar button for displaying a Tree Grid View is added to the Part form

1. Open the **Part** ItemType and select the **Client Style** relationship tab.
2. Right click the entry in the grid and select **Open** to open the Presentation Configuration for the Part ItemType.
3. Click the **Edit** button.
4. Click the **New** button in the Command Bar Section grid to create a new section with the following properties:
    a. Classification: **Data Model**
    b. Name: **Part_Custom_Sidebar**
    c. Location: **ItemWindowSidebar**
    d. Sort Order: **256**
    e. For Identity: **World**
5. Click the **Save** button, then right click the new Command Bar Section, and select **Open**.
6. Click the **Edit** button.

7. Click the **New** button in the Command Bar Item grid to create a new item with the following properties:
    a. Type: **Button**
    b. Name: **My_TGV_Button**
    c. Sort Order: **1200**
    d. Action: **Add**
    e. For Identity: **World**
8. Click **Save**, then right click the new Button, and select **Open**.
9. Click the **Edit** button.
10. Set the following properties on the item form:
    a. Additional Data: **{ "tgvdId": "<tree grid view id>", "startConditionProvider": "ItemDefault({\"id\":\"id\"})" }**
    b. Click Method: **sidebar_default_tgv_click**
    c. Image: **choose the icon you want to show when the sidebar button is inactive**
    d. Additional Image: **choose the icon you want to show when the button is active**

Note:	The *startConditionProvider* property in Additional Data is optional. The view will use the context item's id as the starting condition if this property is not provided. It's only necessary to include the *startConditionProvider* property if your Tree Grid View has a start condition that is not the id of the context item.

11. Click **Save**, then navigate to **Design > Parts** in the TOC.
12. Open a Part item to see the new button in the sidebar.
13. Click the new sidebar button to view the specified Tree Grid View.

4.7 Headers and Title Bars

4.7.1 Add a Button to the Main Window Header

"I want to display the 'help' content when the user clicks a button in the main window header."

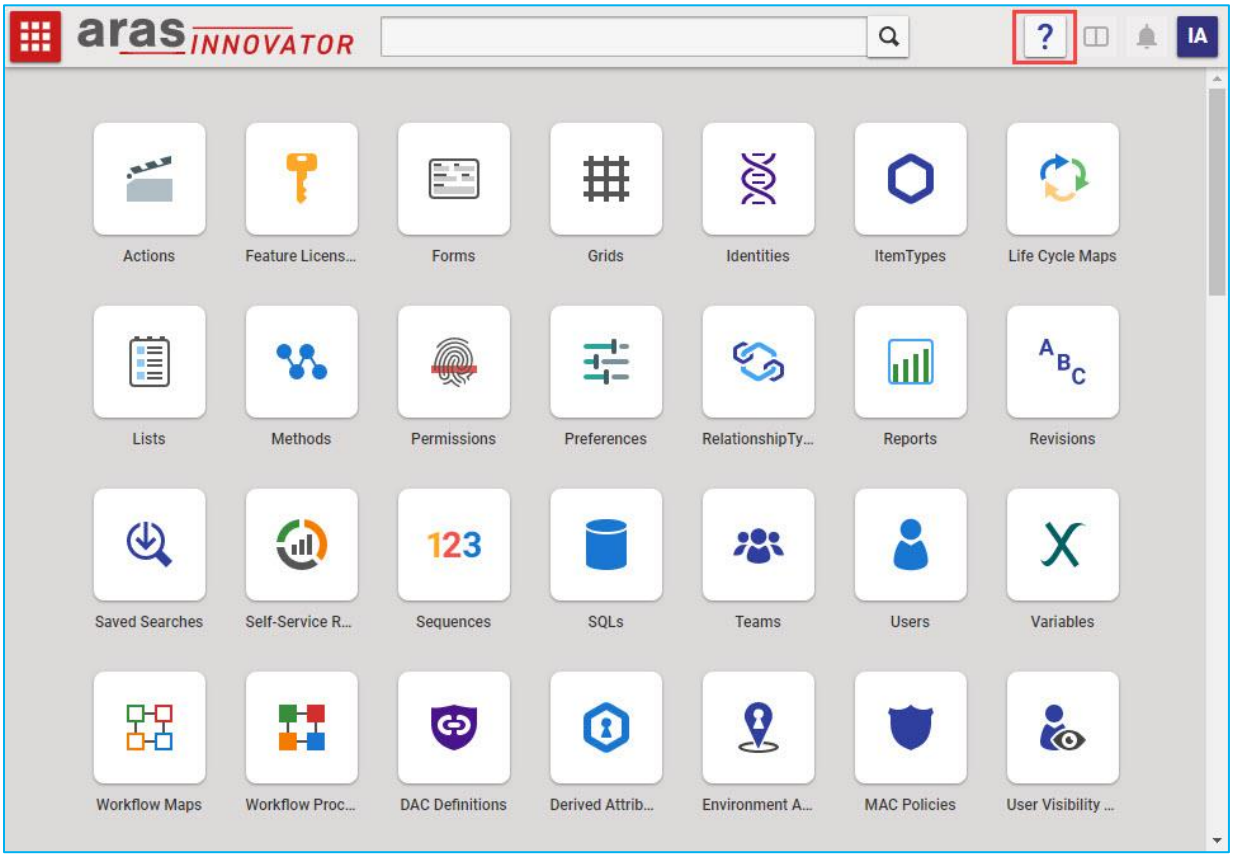

Figure 41. A main window header button for displaying the default Aras Innovator "help" content

1. Navigate to **Administration > Configuration > Client Presentation** in the TOC.
2. Click the **Search** button to run the Client Presentation search.
3. Click the **Global** item property in the single search result.
4. In the Command Bar Section grid, open the **com.aras.innovator.cui_default.mwh_header** item.
5. Click the **Edit** button.
6. Click the **New** button in the Command Bar Item grid, then select **Button > OK** in the dialog to create a new button with the following properties:
   a. Name: **Custom_Help_Button**
   b. Sort Order: **75**
   c. Action: **Add**
   d. For Identity: **World**
7. Right click the new button row and select **Open**.

8. Enter the following properties for the new Menu Button:
   a. Click Method: **cui_default_mwh_onHelpCommand**
   b. Image: choose the icon you want to show on the button
   c. Additional Data: **{"right":true}**
9. Click the **Save** button and close the Button tab.
10. Logout and login to see the new button in the main window header.

4.7.2 Add a Button to the Search View Title Bar

"I want to display the context ItemType when an admin clicks a button in the search view title bar."

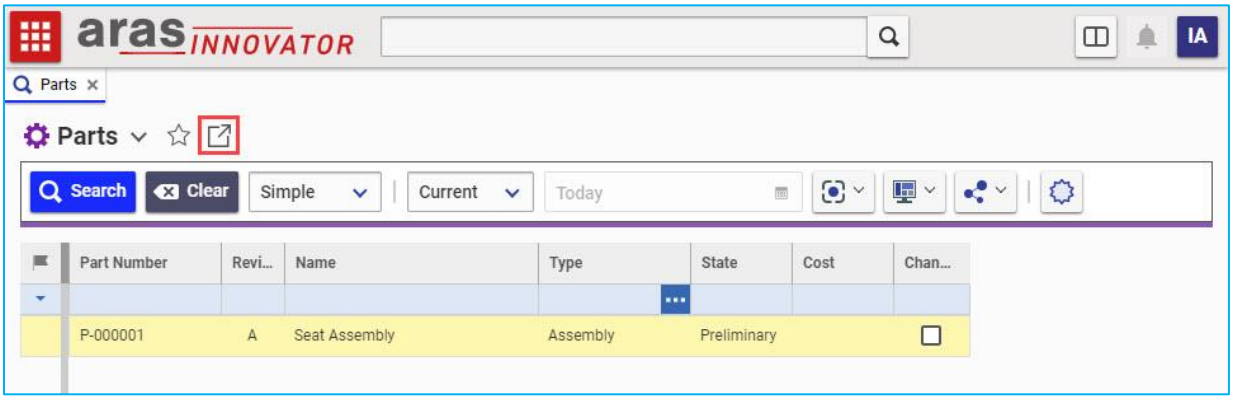

Figure 42. A search view title bar button for displaying the context ItemType definition

1. Navigate to **Administration > Methods** and create a new JavaScript Method with the following code:

```
var topWindow = aras.getMostTopWindowWithAras(window);
var workerFrame = topWindow.work;
if (workerFrame && workerFrame.itemTypeID) {
   aras.uiShowItem('ItemType',workerFrame.itemTypeID);
}
```

2. Click **Save** and close your new Method.
3. Navigate to **Administration > Configuration > Client Presentation** in the TOC.
4. Click the **Search** button to run the Client Presentation search.
5. Click the **Global** item property in the single search result.
6. In the Command Bar Section grid, open the **searchview.titlebar.default** item.
7. Click the **Edit** button.
8. Click the **New** button in the Command Bar Item grid, then select **Button > OK** in the dialog to create a new button with the following properties:
   a. Name: **Open_ItemType_From_SearchView**
   b. Sort Order: **768**
   c. Action: **Add**
   d. For Identity: **Administrators**

9. Right click the new button row and select **Open**.
10. Enter the following properties for the new Menu Button:
    a. Click Method: choose the **new Method** you created
    b. Image: choose the icon you want to show on the button
    c. Additional Data: **{ "cssClass": "aras-button_d" }**
11. Click the **Save** button and close the Button tab.

Note: The same title bar content can also be configured for the RelationshipsView, SearchDialog, and GraphView locations.