



ProApp Designer

Version 14.10 - Package 32

Last Modified: 04/06/2026

Copyright Information

Copyright © 2025 Aras Corporation. All Rights Reserved.

Aras Corporation
100 Brickstone Square
Suite 100
Andover, MA 01810
Phone: 978-691-8900

Notice of Rights

Copyright © 2025 by Aras Corporation and/or its affiliates. All rights reserved.

This document is protected by U.S. and international copyright laws and conventions. No copyright may be obscured or removed from this document. This document may not be modified or altered, or reproduced or transmitted in any form, without the explicit permission of the copyright holder.

Aras Innovator, Aras, and the Aras Corp "A" logo are registered trademarks of Aras Corporation in the United States and other countries.

All other trademarks referenced herein are the property of their respective owners.

Notice of Liability

This document is provided for informational purposes only, and the contents hereof are subject to change without notice. The information contained in this document is distributed on an "as is" basis, without warranty of any kind, express or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose or a warranty of non-infringement. Aras shall have no liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this document or by the software or hardware products described herein.

Table of contents

Aras Background and Cron Jobs User Guide	11
Introduction	12
Scope	13
Target Audience	14
Feature Terms and Definitions	15
Guide Organization	16
Event Subscription and Notification User Guide	17
Introduction	18
Scope	19
Target Audience	20
Feature Terms and Definitions	21
Guide Organization	22
Event Subscriptions & Notifications	23
Architecture	24
Event Subscriptions & Notifications Roles	25
Types Of Events Available For Subscribe	26
Enable Subscription on an ItemType	27
Configure Subscriptions on an ItemType	28
Configure Subscriptions on an Item	30
Notification Templates	32
Aras ProApp Designer - SAAS Installation Guide	34
Introduction	35
Prerequisites	36
Definitions	37
Aras ProAppDesigner Installation	38
Aras ProAppDesigner Installation Instruction	39
Confirming the Installation	43



Aras ProApp Designer - Upgrade Guide	44
Overview	45
Prerequisites	46
Checking for Eligibility	47
Upgrading Aras ProAppDesigner	48
Upgrading Aras ProAppDesigner	50
Confirming the Installation	55
Aras ProAppDesigner - User Guide	56
Introduction	57
Feature Terms and Definitions	58
Content Overview	59
Introduction to Aras ProAppDesigner	60
Purpose of Aras ProAppDesigner Wizard	61
Process Overview	62
Global Dashboard	63
Type Dashboard	67
Property Filter	68
User Dashboard	70
Wizard	75
Wizard Actions	76
Single Page	78
Multi Page	79
Page Navigation	80
Page Controls	81
Layout Controls	82
Simple Controls	85
Collection Controls	98
Page & Control Features	115
Tooltip & Help Text	116

Display Conditions	118
Validation Rules	119
Computed Values	121
Conditional Formatting	122
Dynamic ReadOnly	123
Reset Dependencies	124
Data Export	125
PowerPoint Generation	126
Technical Data Package	127
Responsive UI & Mobile Views	128
Aras Presentation Generator - User Guide	130
Introduction	131
Scope	132
Target Audience	133
Feature Terms and Definitions	134
Guide Organization	135
Introduction to Presentation Generator	136
Presentation Generator Template Configurations	137
Template Selection	140
Presentation Template Creation	141
Configure Presentation Generator Command on Template View	144
Configure Custom action for Presentation Generator	147
Aras ProApp Designer - Installation Guide	149
Introduction	150
Prerequisites	151
Installing ProAppDesigner	152
Notification and Backup	153
Installing the Aras ProAppDesigner	155
Confirming the Installation	164



Activating ProAppDesigner	165
Setting Required Variables	166
Aras Technical Data Package - User Guide	167
Introduction	168
Scope	169
Target Audience	170
Feature Terms and Definitions	171
Guide Organization	172
Introduction to Technical Data Package (TDP)	173
TDP Template	174
TDP Template Management	175
Auto-Generate TDP Template	177
Excel Template Selection	179
Template File Assumptions	180
Export TDP	182
Import TDP	184
Aras Word PDF Generator - User Guide	185
Introduction	186
Scope	187
Target Audience	188
Feature Terms and Definitions	189
Guide Organization	190
Introduction to Word/PDF Generator	191
Document Generator Template Configurations	192
Template Selection	195
Word Template Creation	196
Using Controls: Plain Text Content Control with Locking Behavior	198
Using Controls: Repeating Text Content	200
Tag Naming Conventions and its meaning	202
Chart Support	203



Mapping Content Control to an Item Type in Server Method	204
Mapping to remove empty content controls along with Label	205
Steps to edit placeholder text directly from 'Design Mode' and to preserve the original style	208
Configure Word Generator Command on Template View	209
Configure Custom action for Word Generator	212
ProAppDesigner Rule Engine - User Guide	214
Introduction	215
Scope	216
Target Audience	217
Feature Terms and Definitions	218
Guide Organization	219
Introduction to ProAppDesigner Rule Engine	220
Schematic Diagram	221
Rule	222
Rule Validation	224
Rule Simulation	225
RuleSet	226
Associate Rule/RuleSet	227
Rule Execution	229
Program Management Extension - Installation guide	230
Introduction	231
Scope	232
Target Audience	233
Prerequisites	234
Checking for Eligibility	235
Installing Program Management Extension	236
Notification and Backup	237
Installing the Program Management Extension	239
Confirming the Installation	244
Activating Program Management Extension	245



Program Management Extension - User Guide	246
Introduction	247
Scope	248
Target Audience	249
Feature Terms and Definitions	250
Guide Organization	251
Introduction to Program Management Extension	252
Project View	253
Project Plan	255
Project Plan Views	256
Grid View	260
Timeline View	262
Resource View	263
Split View	264
Master and Subprojects	265
Project Baseline	267
Import Project Template	268
Calendars	270
Organization Calendar	271
Project Calendar	275
Team Calendar	277
User Calendar	280
Activity Completion Form in User Calendar	284
Event and Task Color Coding	285
Aras ProApp Designer - Developer Guide	286
Introduction	287
Scope	288
Target Audience	289
Feature Terms and Definitions	290
Guide Organization	291



ProAppDesigner API Overview	292
Events	293
OnClick Event	294
OnLoad Event	295
OnChange Event	296
OnBeforeLoad Event	297
OnSelect Event	298
OnSubmit Event	299
OnInsert Event	300
OnSelectFlyout Event	301
OnInsertFlyout Event	302
OnSave Event	303
OnBeforeSave Event	304
Event Handling	305
Wizard and Wizard Controls API	306
Wizard	307
Aras Object	309
Simple Controls	310
Collection Controls	311
Control Item	318
Layout Controls: Group & Section	322
List Control	323
Frame Control	324
CADViewer Control	325
DynamicTable Control	326
Messages	328
Progress Ring	329
ApplyMethod	330
Examples	332
How to Set and Get Value on Text Control	333



How to Set and Get Value on RichText Control	334
How to Set and Get Value for a Row on Table Control	335
How to Configure DynamicTable Control	336
Aras ProApp Designer - Administration Guide	341
Introduction	342
Feature Terms and Definitions	343
Content overview	344
Aras ProAppDesigner Overview	345
Roles	346
Typical Workflow	347
Architecture	348
Performance	350
Caching	351
Progressive Loading	352
Table and Worksheet Benchmarking	353
TOC	354
Template Studio	360
Features	361
Studio Layout	362
TemplateView Actions	363
TemplateView Settings	364
Control Actions	366
Template Studio Sidebar	367
Toolbox	374
Template Preview	375
Section Control	376
Group Control	379
Text Control	383
List Control	393
Checkbox Control	397



Image Control	399
Runtime View:	402
File Control	403
Runtime View:	405
Document Control	406
Runtime View:	408
RichText Control	409
RichText Control Toolbar:	411
Item Control	414
Runtime View:	416
Items Control	418
Runtime View:	420
xClassification Control	421
Runtime View:	423



Aras Background and Cron Jobs User Guide



Introduction

Purpose

This User Guide describes how to use Aras Background & Cron Jobs application for external users.



Scope

This document provides instructions for configurations and functionality of Aras Background & Cron Jobs application.



Target Audience

This document is intended for external users with limited access to data.



Feature Terms and Definitions

The following Feature terms and definitions are used in this document:

Term	Definition



Guide Organization

This document describes how to use the Aras Background & Cron Jobs application. This User Guide is organized into the following sections:

SECTION 1:	This section defines the purpose and scope of the User Guide, identifies its intended audience. It also supplies a brief overview of each section to help users in navigating the guide.
Introduction	
SECTION 2:	This section explains the purpose of the Background & Cron Jobs feature and offers an overview of the functionality available within Aras Background & Cron Jobs.
Introduction to	
Background & Cron Jobs	



Event Subscription and Notification User Guide



Introduction

Purpose

This guide describes how to use Aras Event Subscriptions and Notifications for external users.



Scope

This document provides instructions for configuration and functionality of Aras Event Subscriptions and Notifications.



Target Audience

This document is intended for users who will configure complex UIs and end users of these UI.



Feature Terms and Definitions

The following Feature terms and definitions are used in this document:

Term	Definition



Guide Organization

This document describes how to use the Aras Event Subscriptions and Notifications. This User Guide is organized into the following sections:

SECTION 1:

Introduction

This section defines the purpose and scope of the User Guide, identifies its intended audience, and provides a concise summary of each part to assist users in navigating the guide effectively.

SECTION 2:

Introduction to Event Subscriptions and Notifications

This section explains the purpose of Event Subscriptions and Notifications, offering an overview of the features available within the Aras Event Subscriptions and Notifications functionality.



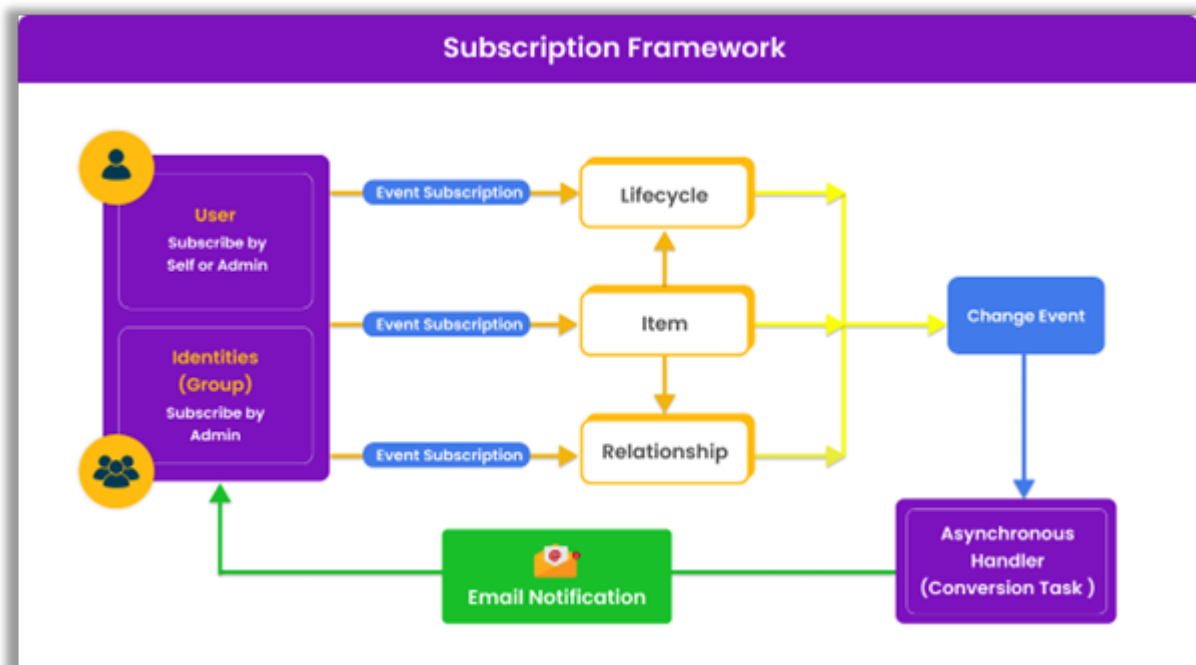
Event Subscriptions & Notifications

Enables administrators or users to subscribe individuals or groups to receive notifications for various events, either at the ItemType level or the specific Item level. Additionally, users have the option to self-subscribe for notifications at the Item level.



Architecture

The Subscription Framework supports two types of subscriptions: Item-Type-level and Item-Level Subscription. Administrators can subscribe multiple events to various alias or non-alias identities either at the ItemType level or for specific Items. Non-admin users also have the ability to self-subscribe to multiple events for individual Items. When a subscribed event is triggered in the system, an asynchronous handler on the conversion server sends email notifications to all subscribed users or groups. This handler operates in the background, delivering notifications to hundreds of users simultaneously without interrupting user actions. This asynchronous processing enhances the efficiency and scalability of the framework.



Event Subscriptions & Notifications Roles

When the Event Subscriptions & Notifications package is imported, it automatically deploys three default group identities: **SN Administrator**, **SN Subscription Administrators**, and **SN Employee**. The **SN Administrator** identity should be assigned to users responsible for enabling subscriptions at the ItemType level. The **SN Subscription Administrators** identity is intended for users who will manage administrative-level subscriptions, whether at the ItemType or individual Item level. Lastly, the **SN Employee** identity must be assigned to users who wish to self-subscribe to notifications for specific Items. These roles help ensure appropriate access and functionality within the subscription framework.

If all Word files have validation rules applied, the system will select the first file that successfully passes the validation. If none of the Word files meet the required validation criteria, the document generation process will be terminated.



Types Of Events Available For Subscribe

- **Admin Level Events Subscription:** Administrators have the ability to register themselves, as well as individual users or groups, to receive notifications for various events associated with a specific ItemType or Item.
- **User Level Events Subscription:** Non-admin users have the ability to register themselves to receive notifications for various events related to a specific Item.

Below are the list of events, user or admin can subscribe/unsubscribe on ItemType / Item level.

Basic Events

- **Create (Available on ItemType Level for Admin):** When this event is subscribed to, all users configured by the administrator will receive a notification whenever an Item of the specified ItemType is created in the system.
- **Modify:** When this event is subscribed to, all users configured by the administrator at the ItemType level, along with all users who have self-subscribed at the Item level, will receive a notification whenever an Item of the specified ItemType is modified in the system.
- **Revise:** When this event is subscribed to, all users configured by the administrator at the ItemType level, as well as all users who have self-subscribed at the Item level, will receive a notification whenever an Item of the specified ItemType is revised in the system.
- **Delete:** When this event is subscribed to, all users configured by the administrator at the ItemType level, along with all users who have self-subscribed at the Item level, will receive a notification whenever an Item of the specified ItemType is deleted from the system.

Relationship Events

- On subscription window, events are shown to the user/admin dynamically inside *Relationship* section.
- This section displays a list of non-TGV relationships associated with the context ItemType. It allows users and administrators to subscribe to one or more of these relationships, either at the ItemType level or the individual Item level.
- For all subscribed relationships, notifications will be sent to users configured by the administrator at the ItemType level, as well as to users who have self-subscribed at the Item level, under the following conditions.
 - When new relationship is created in the system.
 - When any relationship is deleted from the system.

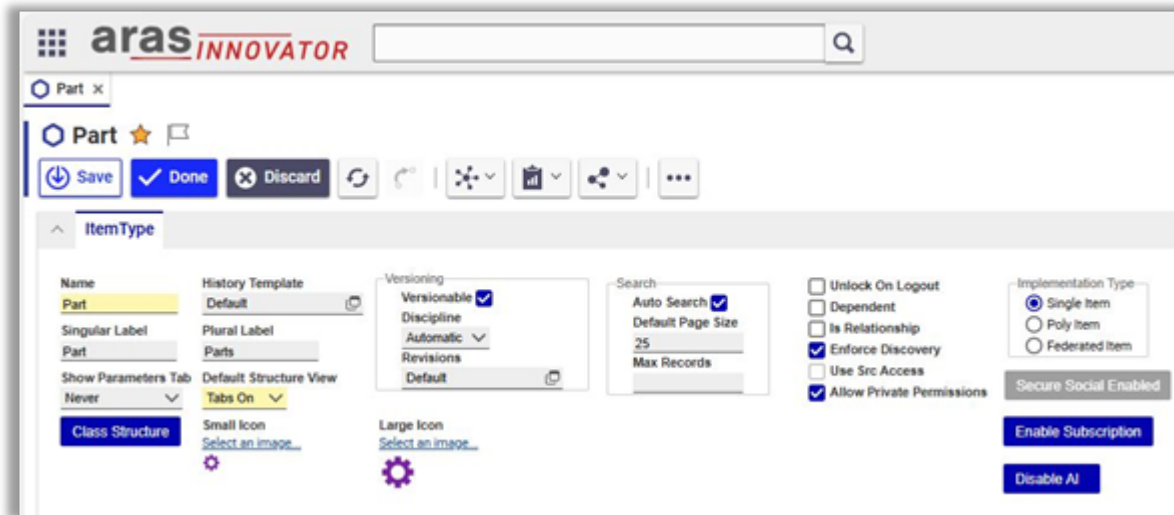
Lifecycle Events

- Within the subscription window, available events are dynamically displayed to the user or administrator inside *Lifecycle States* section.
- This section displays the list of lifecycle states associated with the context ItemType. It enables both users and administrators to subscribe to one or more of these lifecycle states, either at the ItemType level or the individual Item level.
 - For all subscribed lifecycle states, notifications will be sent to users configured by the administrator at the ItemType level, as well as to users who have self-subscribed at the Item level, whenever an Item enters a subscribed state.



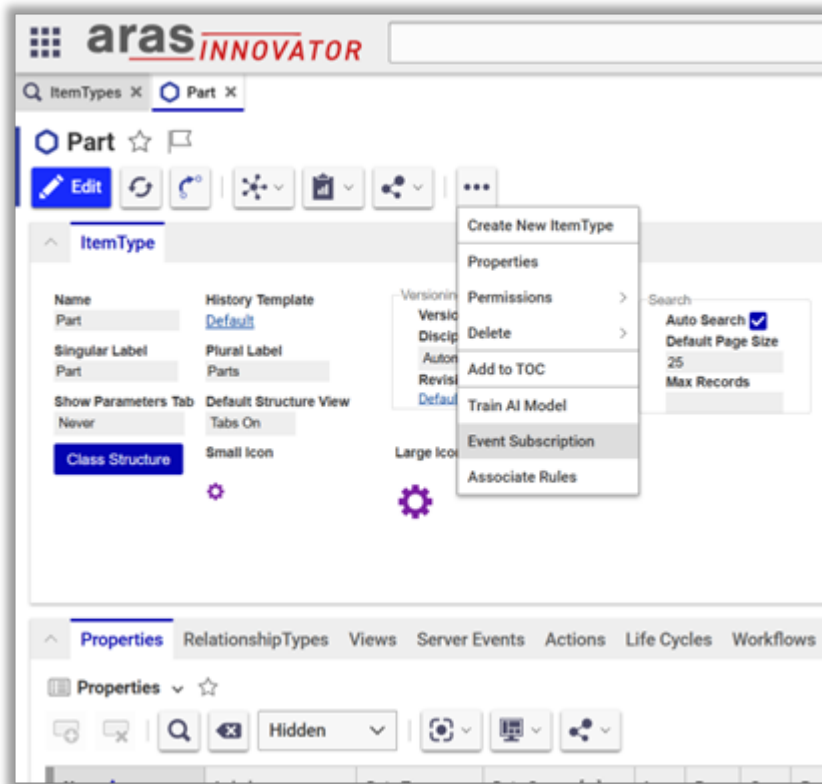
Enable Subscription on an ItemType

Admin user can enable the subscription on an ItemType by just opening the ItemType from TOC → Administration → ItemTypes → Search and open the ItemType → Click on null

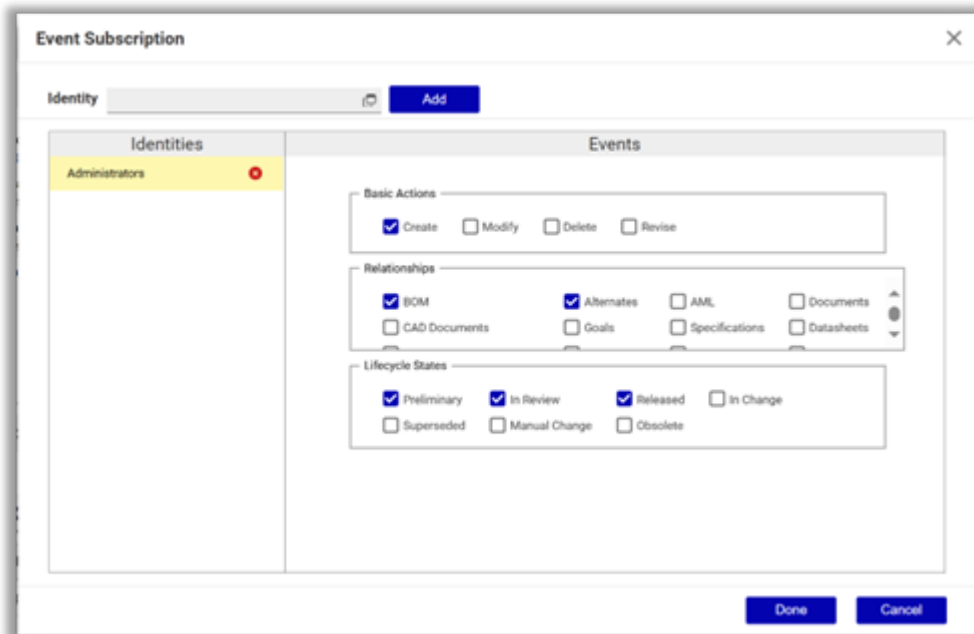


Configure Subscriptions on an ItemType

Once subscription is enabled on an ItemType, the administrator can utilize the **Event Subscription** action to configure admin-level subscriptions.



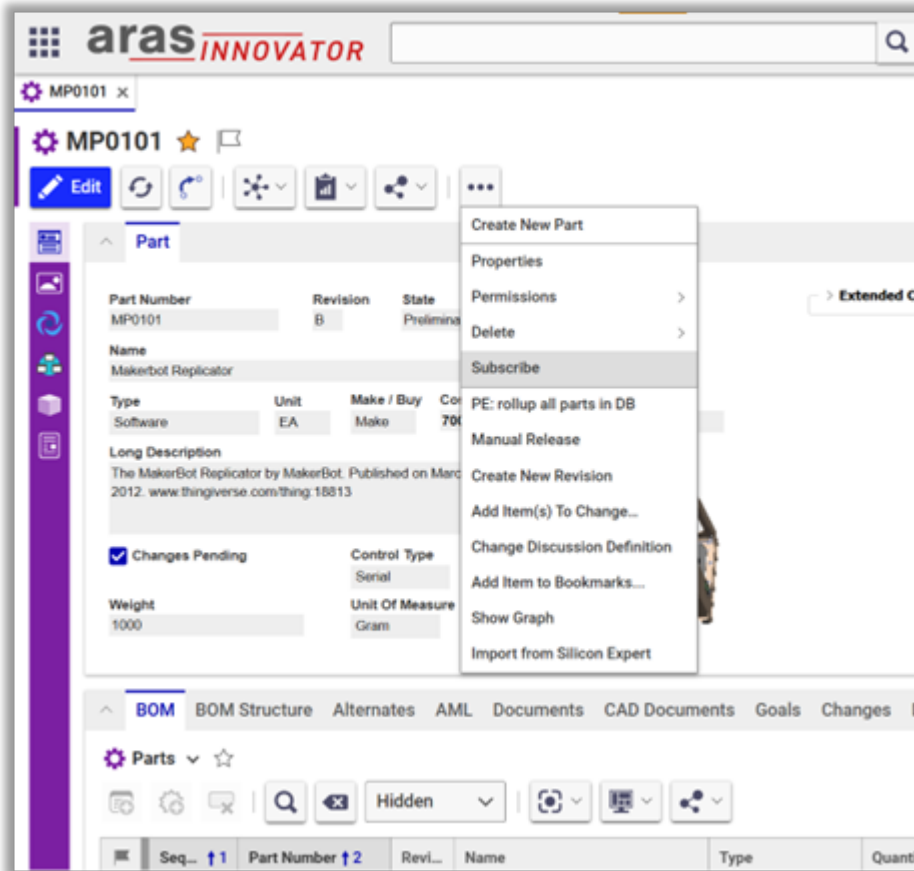
- This will open the below dialog box.



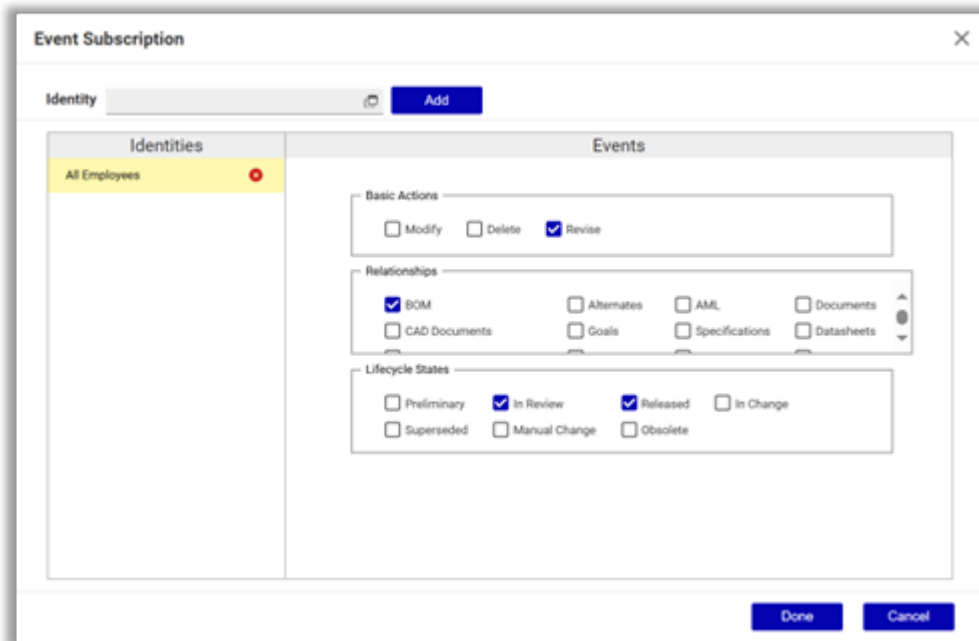
- When this dialog box is loaded, it automatically populates the list of identities along with their corresponding subscriptions.
- Admin user can add the new Identity (Alias or Non-alias) using the *Add* button.
- Admin user can select the Identity on left panel and add the events subscription on right panel.
- Admin user can update the event selections for the selected Identity.
- Admin user can remove/delete the selected Identity from admin subscriptions.
- Clicking on *Done* button saves the final state of event subscriptions for all identities into the system and closes the dialog box.
- Clicking on *Cancel* button will abort the operation and close the dialog box.

Configure Subscriptions on an Item

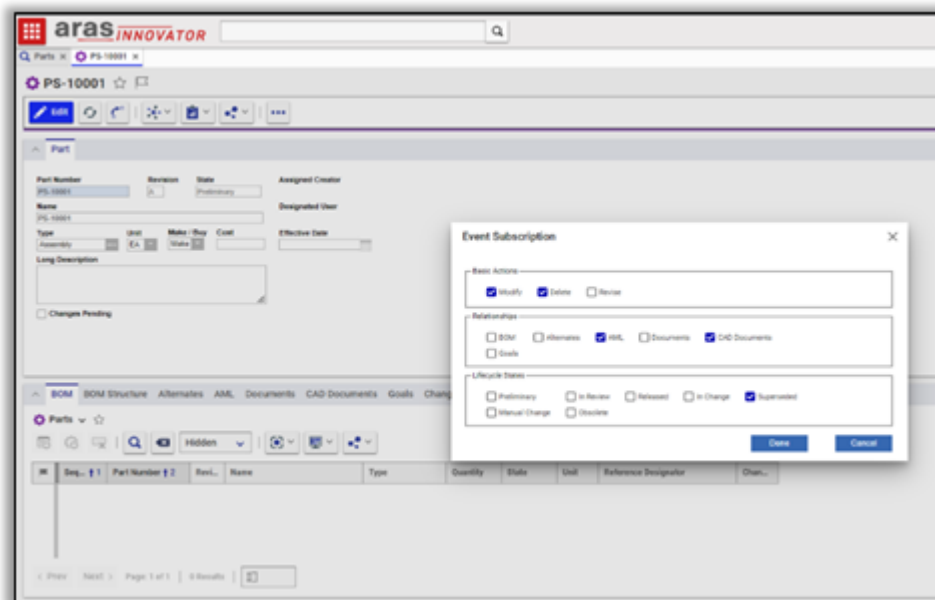
- Open any subscription enabled ItemType's Item using subscription admin user and click on *Subscribe* button.



- For admin subscription users, it will open below dialog box where admin user can add the admin level subscriptions on Item.

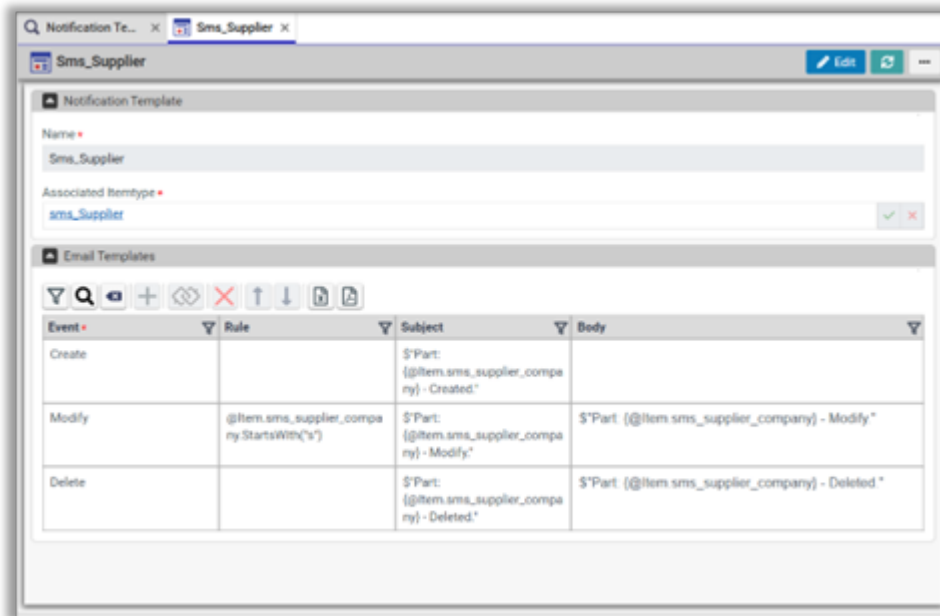


- For non-admin users, it will open below dialog box where user can add the subscriptions for themselves on an Item.



Notification Templates

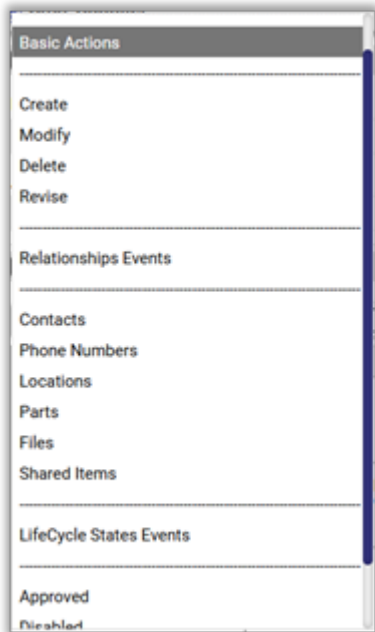
- Notification Templates are used to define the content of email notification messages. These templates allow you to configure both the subject and the body of the message for different Events mentioned in section [Types of events available for subscribe](#)
- Admin users can setup Notification Templates for each ItemType.
- To configure Notification Templates, go to TOC → Administrators → ProAppDesigner → Templates → Notification Templates



- Below are the properties that needs to be set:
 - **Name:** Name of the template
 - **Associated ItemType:** ItemType for which this template is configured.
- In the Email Templates section below, the administrator can configure the email message for each event. During this setup, the user is required to specify the following key properties.



- **Event:** You can select any of the Event listed in the dropdown.



- **Rule:** Admin can configure Rules based on Item's and User's properties.
- **Subject:** The Subject field defines the title of the email message. Administrators can customize it by including properties from the related Item using the following syntax:
`"Part: {@Item.sms_supplier_company} - Created."`
- **Body:** The Body of the email message contains the main content delivered to recipients. Administrators can customize this section by including properties from the related Item using the syntax:
`"Part: {@Item.sms_supplier_company} - Modify."`



Aras ProApp Designer - SAAS Installation Guide



Introduction

Purpose and Target Audience

The **Aras ProAppDesigner SaaS Installation Guide** for Aras DevOps provides instructions for Aras DevOps and Enterprise subscribers to add the Aras ProAppDesigner application to their Aras Innovator using SaaS. This document is intended for Aras DevOps users responsible for installing and configuring the applications and platform components for the Aras Innovator instance utilizing SaaS.



Prerequisites

To install Aras ProAppDesigner, **users must confirm** that the installation of Aras Innovator meets the minimum requirements to run ProAppDesigner.

Check the eligibility of code tree and database:

From the **TOC**, select **Administration, Variables**.

VersionMajor	VersionMinor	VersionServiceUpdate
14	0	32
14	0	33
14	0	34
14	0	35
14	0	36
14	0	37

If it does not meet the service pack requirements listed in the previous steps, contact Aras to discuss options.

Aras DevOps administrators or contributors **must have**:

- An installed Aras Innovator instance before following the steps outlined in this document.
- It is **recommended** that all administrators read the *Aras DevOps – User Guide*. It explains how to utilize the Aras DevOps service effectively.



Definitions

This section defines the terms used in this document.

Term	Definition
Code tree	The set of files that make up the application files for Aras Innovator.
Code tree patch	The set of files in an application installation package applied to the code tree to install the code tree component.
Config transformation	The transformation sourced into the project repository corresponding to changes in configuration files of Aras Innovator code tree.
Import package	The set of XML files and the corresponding import.mf file representing a package or set of packages that need to be installed to apply the database changes of an application.
Installation package	The set of files utilized to install an application (may include a code tree patch, an import package, and/or converters).
Platform Component	Not release-specific components that can be installed without a direct dependency on the installed Aras Innovator release.



Aras ProAppDesigner Installation

Aras Innovator needs to be installed before **Aras ProAppDesigner** component. It is **important** to notify users that Aras Innovator will experience downtime due to installation. Additionally, it is important to **create backups** for the Aras Innovator code tree and database.

Important

Please store backup files securely, to restore the system to its pre-installation state if installation fails.



Aras ProAppDesigner Installation Instruction

The following steps outline the process of installing **Aras ProAppDesigner**:

1. Download the **Aras ProAppDesigner CD Image** from the Aras File Sharing site.
2. Unzip the Aras ProAppDesigner CD Image on the local machine.
3. Navigate to the `\Files\` folder of the Aras ProAppDesigner CD Image.
4. Copy the `\Files\Innovator\Client` into the `\CodeTree\Innovator\Client` folder of the `Work.git` repository. When prompted, **overwrite** the existing folder and all of its contents.
5. From the `\Files\Innovator\Server` folder, copy all files matching **Prorigo.*** into the `\CodeTree\Innovator\Server` folder of the `Work.git` repository. When prompted, **overwrite** the existing files in the folder.
6. From the `\Files\Innovator\Server` folder, copy all remaining files **excluding** `Prorigo.*` into the `\CodeTree\Innovator\Server` folder of the `Work.git` repository. Do **not overwrite** any existing files during this step.
7. From the `\Files\Innovator\Server` folder, copy all remaining files **excluding** `Prorigo.*` into the `\CodeTree\Innovator\Server` folder of the `Work.git` repository. Do **not overwrite** any existing files during this step.
8. If any of the below DLLs are available in `\Innovator\Server\bin` or in `ConversionServer\bin` folder, delete them
 - o Prorigo.Protrak.ExpressionEvaluatorStd.Impl.dll
 - o Prorigo.Protrak.ExpressionEvaluatorStd.dll
 - o Prorigo.Plm.PresentationGenerator.dll
 - o Prorigo.Plm.PresentationGenerator.Aras.dll
 - o Prorigo.Plm.Logging.Framework.dll
 - o Prorigo.Plm.DocumentGenerator.dll
 - o Prorigo.Plm.DocumentGenerator.Aras.dll
 - o Prorigo.Plm.DiscussionPanel.dll
 - o Prorigo.Plm.Core.dll
 - o Prorigo.Plm.Aras.BackgroudCronJobFramework.dll
 - o Prorigo.Plm.AppStudio.RuleEngine.Impl.dll
 - o Prorigo.Plm.AppStudio.RuleEngine.dll
 - o Prorigo.Plm.AppStudio.IntegrationFramework.dll
 - o Prorigo.Plm.AppStudio.dll
9. Copy the Pro App Designer folder, except the `imports.mf` file, from the `..\Imports\Pro App Designer..` folder into the `AML-packages` folder of the `Work.git` repository.

Important

Do not copy the `imports.mf` file, as it will overwrite the existing `imports.mf` file in the `Work.git` repository.

10. Open the `..\Imports\imports.mf` file.
11. Copy the following <package> elements:



```
<imports>

  <package name="appstudio" path="ProAppDesigner\appstudio\Import" />

  <package name="documentgenerator" path="ProAppDesigner\documentgenerator\Import" />

  <package name="technicaldatapackage" path="ProAppDesigner\technicaldatapackage\Import" />

  <package name="businessruleengine" path="ProAppDesigner\businessruleengine\Import">

    <dependson name="backgroundcronjobframework" />

  </package>

  <package name="backgroundcronjobframework" path="ProAppDesigner\backgroundcronjobframework\Import">

  <package name="subscriptionframework" path="ProAppDesigner\subscriptionframework\Import" />

  <package name="powerpointgenerator" path="ProAppDesigner\powerpointgenerator\Import" />

  <package name="appstudiotemplates" path="ProAppDesigner\appstudiotemplates\Import">

    <dependson name="appstudio" />

  </package>

  <package name="integrationframework" path="ProAppDesigner\integrationframework\Import">

    <dependson name="backgroundcronjobframework" />

  </package>

  <package name="discussionpanel" path="ProAppDesigner\discussionpanel\Import" />

  <package name="proappdesigner_toc" path="ProAppDesigner\proappdesigner_toc\Import">

    <dependson name="appstudio" />

    <dependson name="backgroundcronjobframework" />

    <dependson name="businessruleengine" />

    <dependson name="documentgenerator" />

  </package>

</imports>
```




```
xdt:Transform="InsertAfter(/MethodConfig/ReferencedAssemblies/name[last()])">$(binpath)/Prorigo.Plm.A  
<name  
xdt:Transform="InsertAfter(/MethodConfig/ReferencedAssemblies/name[last()])">$(binpath)/Prorigo.Plm.L  
</ReferencedAssemblies>  
</MethodConfig>
```

Important

If method.config file does not exist at above location, add that file and put entire content given here.

Important

If any of the DLLs mentioned in the DLLs to be deleted section, please delete the corresponding entry from method-config.xml file.

17. Stage and review the changes to ensure that no customizations are overwritten.
18. Commit the changes.
19. Run the **continuous-integration** Pipeline.
20. Run the **deploy-innovator** Pipeline to complete the installation.



Confirming the Installation

Use the following procedure to check if the database has been updated correctly:

1. **Log in** to Aras Innovator as an administrator.
2. From the **Table of Contents**, expand **Administration** and select **Variables**.
3. Confirm the following variable:

Pro App Designer
14.14.32.10

If the installation fails at any time, **revert** to backups and **contact Aras Support** at support@aras.com.



Aras ProApp Designer - Upgrade Guide



Overview

This document outlines the steps for upgrading Aras ProAppDesigner from any previous release to 14.14.*. These changes affect both the server database and code tree.



Prerequisites

To upgrade from Aras ProAppDesigner 14.14.* confirm eligibility for the upgrade. Download and install any required software.



Checking for Eligibility

Please use the following procedure to check that your Aras Innovator code tree and database meet the minimum requirements to apply the Aras ProAppDesigner Solution AML.

1. Log in to Aras Innovator as an administrator.
2. From the TOC, select **Administration\Variables**.
3. Confirm following variable value:
Pro App Designer: 14.0.1



Upgrading Aras ProAppDesigner

Notification and Backup

1. Notify users that the system will be down at a scheduled time and that they should log out before the process starts.

Important

It is best to give at least 24-hour notice, as well as a reminder 15 minutes prior to the upgrade.

2. Backup the Code Tree.

The “Code Tree” refers to files and folders installed on the disk when Aras Innovator was first installed.

The default path for the Code Tree installation would be something like:

C:\Program Files (x86)\Aras\Innovator

With the following contents:

Name	Type	Size
AgentService	File folder	
ConversionServer	File folder	
DB	File folder	
Innovator	File folder	
OAuthServer	File folder	
SelfServiceReporting	File folder	
Tools	File folder	
VaultServer	File folder	
ConversionServerConfig	XML File	1 KB
InnovatorServerConfig	XML File	2 KB
SelfServiceReportConfig	XML File	1 KB
VaultServerConfig	XML File	1 KB



The following steps can be used to confirm the

installation folder:

1. From the Windows search field, enter **Control** to access the **Control Panel**.
2. Select **Programs and Features**.
3. Right-click the **Name** column and select **More...**
 - Select the **Comments** header if it has not been checked.
 - Click **OK**
4. Search for the **Aras Innovator** entry in the **Programs and Features** window.



5. View the value set in the **Comments** column for the Aras Innovator entry:

Organize ▾		Uninstall			
Name ^	Publisher	Installed On	Size	Version	Comments
 Aras Innovator	Aras Corporation	10/22/2024	1.24 GB	14.0.28	Installed to C:\Aras Installations\Innovator\Release32\
 Aras Update	Aras Corporation	7/26/2024	42.1 MB	1.23.1357	Aras Update

3. Disconnect all users from the database.

The easiest way to prevent client sessions from committing any further changes to the database is to change the database connection string in the `InnovatorServerConfig.xml` from `<DB-Connection...` to `<xDB-Connection...` and restart the **w3svc service (IIS)**. This expires all sessions and prevents all new connections to the Aras Innovator database through the existing instance.

4. Backup the database.

Store these files in a safe location, as they will need to be restored if the process fails.

5. Enable database connections.

After the database backup has been completed, enable the database connection string in the

`InnovatorServerConfig.xml` by changing it from `<xDB-Connection...` to `<DB-Connection...` and restarting the **w3svc service (IIS)**.



Upgrading Aras ProAppDesigner

1. Download the Aras ProAppDesigner CD Image from the Aras FTP site, in the `Aras ProAppDesigner Releases` folder, and unzip the file on the local computer.
2. From the root of your install directory, navigate to `Innovator\Client\scripts\AppStudio` folder and delete all its content
3. From the root of your install directory, navigate to `Innovator\Server\bin` folder and delete below listed DLLs
 1. `Prorigo.Protrak.ExpressionEvaluatorStd.Impl.dll`
 2. `Prorigo.Protrak.ExpressionEvaluatorStd.dll`
 3. `Prorigo.Plm.PresentationGenerator.dll`
 4. `Prorigo.Plm.PresentationGenerator.Aras.dll`
 5. `Prorigo.Plm.Logging.Framework.dll`
 6. `Prorigo.Plm.DocumentGenerator.dll`
 7. `Prorigo.Plm.DocumentGenerator.Aras.dll`
 8. `Prorigo.Plm.DiscussionPanel.dll`
 9. `Prorigo.Plm.Core.dll`
 10. `Prorigo.Plm.Aras.BackgroudCronJobFramework.dll`
 11. `Prorigo.Plm.AppStudio.RuleEngine.Impl.dll`
 12. `Prorigo.Plm.AppStudio.RuleEngine.dll`
 13. `Prorigo.Plm.AppStudio.IntegrationFramework.dll`
 14. `Prorigo.Plm.AppStudio.dll`
4. From the root of your install directory, navigate to `ConversionServer\bin` folder and delete below listed DLLs
 1. `Prorigo.Plm.Logging.Framework.dll`
 2. `Prorigo.Plm.Core.dll`
 3. `Prorigo.Plm.Aras.BackgroudCronJobFramework.dll`
 4. `Prorigo.Plm.AppStudio.IntegrationFramework.dll`
5. Copy the `\Files\Innovator\Client` folder to the root of your installation directory. When prompted, **overwrite** the existing `\Innovator\Client` folder and all of its contents.
6. From the `\Files\Innovator\Server` folder, copy all files matching `Prorigo.*` to the root of your installation directory. When prompted, **overwrite** the existing files in the `\Innovator\Server` folder.
7. From the `\Files\Innovator\Server` folder, copy all remaining files **excluding** `Prorigo.*` into the `\Innovator\Server` folder. Do **not overwrite** any existing files during this step.
8. Copy the `\Files\ConversionServer` folder to the root of your install directory, overwriting the existing `\ConversionServer` folder and all its contents.
9. From the root of your code tree, navigate to `\Innovator\Server` and edit the `method-config.xml` file.

1. Add below entries as shown:

...

```
<!-- ProAppDesigner DLLs : Start -->
<name>$(binpath)/Prorigo.Plm.Aras.dll</name>
```



```
<name>$(binpath)/Proriqo.Plm.Aras.AppStudio.dll</name>  
<name>$(binpath)/Proriqo.Plm.Aras.TDP.dll</name>  
<name>$(binpath)/Proriqo.Plm.Aras.SubscriptionFramework.dll</name>  
<name>$(binpath)/Proriqo.Plm.Aras.DocumentGenerator.dll</name>  
<name>$(binpath)/Proriqo.Plm.Aras.BackgroundCronJobFramework.dll</name>  
<name>$(binpath)/Proriqo.Plm.Aras.RuleEngine.dll</name>  
<name>$(binpath)/Proriqo.Plm.Aras.PresentationGenerator.dll</name>  
<name>$(binpath)/Proriqo.Plm.Aras.IntegrationFramework.dll</name>  
<name>$(binpath)/Proriqo.Plm.Aras.IntegrationFramework.Connector.dll</name>  
<name>$(binpath)/Proriqo.Plm.Aras.DiscussionPanel.dll</name>  
<name>$(binpath)/Proriqo.Plm.Aras.Logger.dll</name>  
<name>$(binpath)/Proriqo.Plm.LicensingFramework.dll</name>  
<!-- ProAppDesigner DLLs : End -->  
</ReferencedAssemblies>
```

2. Save the changes to the method-config.xml file and close it

Important

If any of the DLLs mentioned in the DLLs to be deleted section, please delete the corresponding entry from method-config.xml file.

10. Execute **Pre-Import Scripts**

Before you install Aras ProAppDesigner, execute below scripts:

1. _Navigate Aras Nash tool using URL <http://localhost/InnovatorServer/Client/scripts/nash.aspx>

2. Login in Nash using **admin** user.

The screenshot shows a web browser window with the URL `localhost/InnovatorServerR27/Client/X-salt=std_14.0.15.38102-X/scripts/nash.aspx`. The page header displays the **aras NASH** logo, the database name `InnovatorR27_New`, and the user `admin`. Below the header, there are two dropdown menus for `TimeZone` (set to `Eastern Standard Time`) and `Culture` (set to `en-US - English (United States)`). There are also input fields for `Action` (set to `ApplyAML`) and `Server` (set to `http://localhost/InnovatorServerR27/Server/InnovatorServer.aspx`). A large empty text area labeled `XML:` is present, with `Submit` and `Clear` buttons to its right. Below this, there is a `Run time, sec: 0,000` indicator and a `Result: read` status, with a `ShowXml` button to the right.

3. In the package go to folder `PreImportAMLScripts`, open each file in below sequence, copy the content in XML form and click on `Submit`.

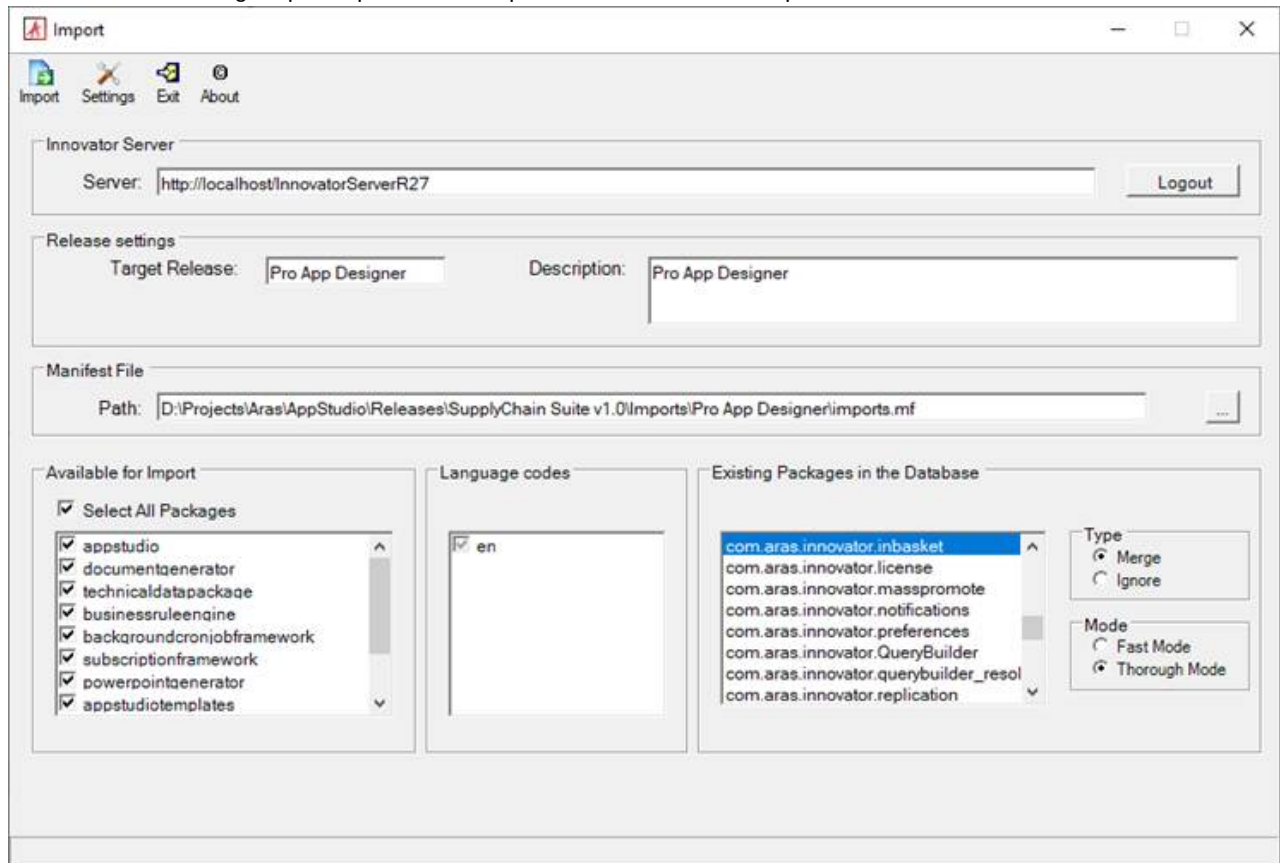
- `04500_TemplateDeploy_PreAML.xml`

11. Complete the **Aras ProAppDesigner** Installation: Import the **ProAppDesigner** database package with the `Package Import Export Utilities` (For more information using this tool, please see the *Aras Innovator - Package Import Export Utilities* documentation.)

1. You will need to enable the “Super User” login to perform this step.



- Browse to the \PackageImportExportUtilities\Import\ folder and run the Import.exe file.



- Enter the following information:

- **Server** = The connection URL for Aras Innovator
- **Database** = The target Aras Innovator database (login screen)
- **Username** = root (login screen)
- **Password** = Password for “root” login (Default is “innovator”) (login screen)
- **Target Release** = ProAppDesigner
- **Description** = ProAppDesigner
- **Manifest File Path** = The manifest is found in the unzipped patch folder in \Aras ProAppDesigner CD Image\Imports\ProAppDesigner\imports.mf
- **Available for Import** = Check all packages
- **Type** = Merge
- **Mode** = Thorough Mode

- Click the Import button.

12. Disable the “Super User” login.

13. Execute **Post Import Scripts**

After you install Aras ProAppDesigner, execute below scripts:

- _Navigate Aras Nash tool using URL <http://localhost/InnovatorServer/Client/scripts/nash.aspx>



2. Login in Nash using **admin** user.

The screenshot shows a web browser window with the URL `localhost/InnovatorServerR27/Client/X-salt=std_14.0.15.38102-X/scripts/nash.aspx`. The page header displays the Aras NASH logo, the database name `InnovatorR27_New`, and the user `admin`. Below the header, there are two dropdown menus for `TimeZone` (set to `Eastern Standard Time`) and `Culture` (set to `en-US - English (United States)`). The `Action` dropdown is set to `ApplyAML`, and the `Server` dropdown is set to `http://localhost/InnovatorServerR27/Server/InnovatorServer.aspx`. There are two large empty text areas labeled `XML:`. The first `XML:` area has `Submit` and `Clear` buttons to its right. Below the first `XML:` area, the `Run time, sec:` is `0,000` and the `Result:` is `read`. The second `XML:` area has a `ShowXml` button to its right.

3. In the package go to folder `PostImportAMLScripts`, open each file in below sequence, copy the content in XML form and click on `Submit`.

- `05000_TemplateDeploy_PostAML.xml`



Confirming the Installation

Use the following procedure to check if your database has been updated correctly:

1. Log in to Aras Innovator as an administrator.
2. From the TOC, select **Administration à Variables**.
3. Confirm the following sets of variables are listed:

Pro App Designer
14.14.*

Important

* here represents version of Innovator.

If at any time the installation fails, revert to your backups and contact Aras Support at support@aras.com.



Aras ProAppDesigner - User Guide



Introduction

Purpose and Target Audience

This User Guide describes how to use the Prophetier for external users with limited access data.



Feature Terms and Definitions

The following Feature terms and definitions are used in this document:

Term	Definition



Content Overview

This document describes how to use Aras ProAppDesigner. This User Guide is organized into the following sections:

SECTION 1: Introduction	This section outlines the purpose and scope of this User Guide, as well as its intended target audience. It supplies a brief overview of each section to help users in navigating the guide.
SECTION 2: Introduction to Aras ProAppDesigner	This section outlines Aras ProAppDesigner information.
SECTION 3: Global Dashboard	This section provides information about global dashboard
SECTION 4: Type Dashboard	This section provides information about type specific dashboard
SECTION 5: Wizard	This section describes wizard actions and pages.
SECTION 6: Page Controls	This section provides detailed information about page controls.
SECTION 7: Page & Control Features	This section provides detailed information about page controls.
SECTION 8: Data Export	This section provides information about data export from Aras ProAppDesigner.
SECTION 9: Responsive UI and Mobile Views	This section provides information about responsive UI and mobile views features of Aras ProAppDesigner.



Introduction to Aras ProAppDesigner

Aras ProAppDesigner Overview

It is a **designer** with a rich set of **UI controls and backend components** to build applications in Aras Innovator just by configuration. Its benefits are:

- It is a **no-code solution** to generate complex, responsive, mobile friendly UI.
- It has a rich set of controls for building complex and modern UI. Any multi-page application can be **built rapidly** in studio environment with drag-drop capabilities.
- Organizes complex UI in to **structured containers** like **Pages, Dialogs, Sections, Groups** which intern enable UI to adapt any screen size.
- It enables creating **guided UI** for showing long data entry forms with conditional display on UI elements based on the configuration.
- It has **rich features** like **conditional display, validation rules, default values, conditional formatting** all with configuration. You don't have to be a programmer to build application with Aras ProAppDesigner.



Purpose of Aras ProAppDesigner Wizard

Aras ProAppDesigner Wizard is a rich user interface used to show details of the item at runtime.



Process Overview

To render any item using Aras ProAppDesigner wizard one needs to follow the steps below:

- **Create Template** or **View for the ItemTypes** that need to be rendered using wizard.
- **Publish** the template through Lifecycle.
- **Create, view** and **edit** actions from Innovator now use published Template for creating or accessing the item.

The screenshot displays the Aras Innovator ProApp Designer interface for configuring a 'Supplier Template'. The interface is divided into several sections:

- Template Details:**
 - Basic:** Name: Supplier Template; Associated Type: Supplier; Revision: 0.
 - Template Type:** Template Type: Advanced View & Dashboard; State: InProgress.
 - Options:**
 - Show OpenItem Command
 - Show Promote Command
 - Enforce Validations
- Template Views:** A table listing various views and their configurations.

View Name	View Type	Rule	Reuse in Portal	Enable Config	Enable Sharing
Default Wizard	Wizard		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Mechanical Supplier View	Wizard	@Item.classification.Contains("Mechanical")	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Manager Dashboard View	Dashboard	@User.identities.Contains("Supply Manager")	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
Supplier Dashboard	Dashboard		<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
Create Wizard	Wizard	@User.identities.Contains("Supply Admin")	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
See View	Wizard	@User.identities.Contains("Innovator Admin")	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>



Global Dashboard

Global dashboard shows data related to multiple ItemTypes in the form of table, worksheet, and report controls. You can have a table or worksheet show list of items from a specific ItemType with pre-applied filter. You can also have reports generated based on different ItemTypes.

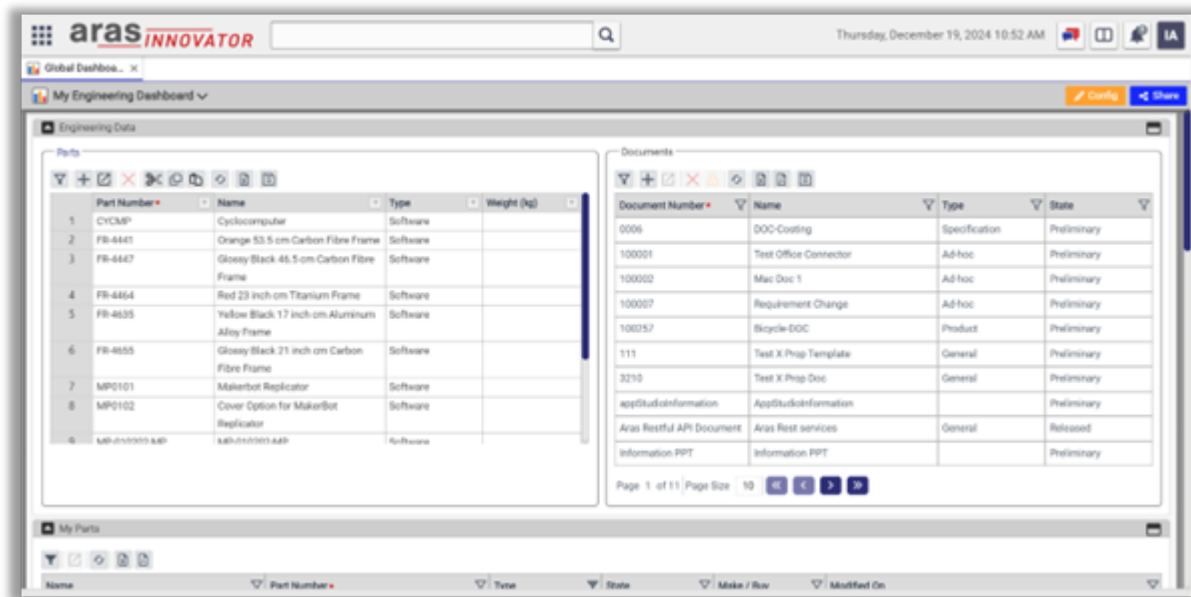


Table Control

Table control can render items of an ItemType based on the pre-applied filter. You can modify the filter if admin configures the table with "Enforce Filter" as false. It has features like:

- **Computed Columns**
- **Validation Rules**
- **Conditional Formatting**
- **Variable row height**; row height will be adjusted based on the cell content
- **Cell navigation using navigation keys** and all other shortcut keys are allowed for operations like Copy, Paste, Delete etc.
- **Row reordering** is supported with Drag & Drop operations

Table column widths can be configured in percentage or pixels by admin. With percentage option, horizontal scrollbar will never be shown. If admin configures the table with "Cascade Delete" setting, all child relationships and related items are deleted when the context item is deleted. Admin can also configure specific column as "Freeze Column" to fix columns that are always visible and don't move



along with horizontal scrollbar. Table can be configured to show either Scrolling or Pagination by admin.

The screenshot shows a window titled "My Suppliers" with a search and filter interface at the top. Below the interface is a table with the following data:

Supplier Name	Type	Weight(kg)	Max Order Quantity	Total Weight(kg)
Crompton	Electrical/Component	45	50	2,250.00
Aras	Electrical/Assembly	32	15	480.00
TECH CH	Electrical/Component	30	70	2,100.00
SKF	Mechanical/Component	20	1,000	20,000.00
Autodesk	Electrical/Assembly	100	16	1,600.00
BOSCH	Electrical/Component	23	45	1,035.00
Siemens	Mechanical/Assembly	50	99	4,950.00
Prongo PPTX	Electrical/Assembly	100	1	100.00

Worksheet Control

Worksheet control provides Excel experience for creating data sheets within Innovator based on ItemType data. Worksheet columns are defined based on ItemType properties and are constrained with property data types. It can show nested rows for a specific item with child relationships data. It has features like:

- **Computed Columns,**
- **Validation Rules,**
- **Conditional Formatting,**
- **Cell navigation using navigation keys** and all other shortcut keys are allowed for operations like Copy, Paste, Delete etc.
- **Excel type of sorting and filtering.** You can copy-paste between Excel and Worksheet control along with Drag & Drop operations for Row reordering,
- **Excel features** like Drag & Drop, Cell Copy by Drag and Undo & Redo etc.

Worksheet column widths can be configured in percentage or pixels by admin. With percentage option, horizontal scrollbar will never be shown. If admin configures the worksheet with "Cascade Delete" setting, all child relationships and related items are deleted when the context item is deleted. Admin can also configure specific column as "Freeze Column" to fix columns that are always visible and don't move along with horizontal scrollbar.

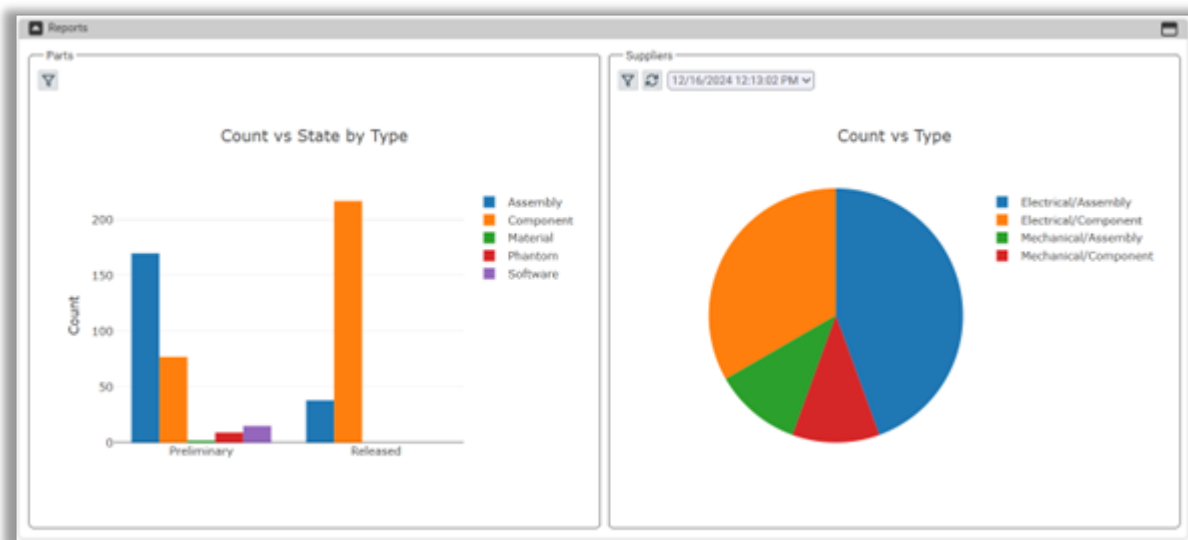


The screenshot shows the 'My Parts' application window. At the top, there is a 'Property Filter' section with an 'AI Prompt' field, a 'Property' dropdown, an 'Operator' dropdown, and a 'Value' input field. Below this is an 'Expression' field containing 'Type EqualTo Phantom'. A toolbar with buttons for 'AND', 'OR', and logical operators is visible. The main area contains a table with the following data:

Name	Part Number	Type	State	Make / Buy	Modified On
Housing Fab Assembly	MP1920	Phantom	Preliminary	Make	2/26/2024 10:08:00 AM
Heater Assembly	MP2718	Phantom	Preliminary	Make	2/26/2024 10:08:00 AM
Mount Assembly	P23124	Phantom	Preliminary	Make	2/26/2024 10:08:00 AM
Heat Sink Assembly	P4523	Phantom	Preliminary	Make	2/26/2024 10:08:00 AM
Printer Assembly	P5423	Phantom	Preliminary	Make	3/25/2024 5:03:00 PM
Fan Assembly	P6744	Phantom	Preliminary	Make	2/26/2024 10:08:00 AM
Thermal Cone	P7234	Phantom	Preliminary	Make	2/26/2024 10:08:00 AM
Filament Assembly	P8906	Phantom	Preliminary	Make	2/26/2024 10:08:00 AM
Drive Block	P9082	Phantom	Preliminary	Make	2/26/2024 10:08:00 AM

Report Control

Report control can be configured in Dashboard by the administrator using QueryDefinition. For Dashboards, report control should be configured with any QueryDefinition, which doesn't expect any item context. At runtime, report control executes QueryDefinition to get DataSource for the report. It can support any type of report like Pivot Table, Tree Table, Table Barchart, Heatmap, Row Heatmap, Column Heatmap, Line Chart, Bar Chart, Stacked Bar Chart, Area Chart, Stacked Area Chart, Pie Chart and PowerBI. All report types can show drilldown reports.





Type Dashboard

Type dashboard shows all the data related to an ItemType in the form of **table**, **worksheet** and **report controls**. You can have a table show list of items with pre-applied filter. You can also have reports generated based on that ItemType. Using Property filter, you can define complex filter criteria to fetch the items.

The screenshot displays the Aras Innovator interface for a 'Supplier Dashboard'. The main component is a table listing suppliers with columns for Supplier Name, Type, Weight, Max Order Quantity, and Total Weight. The table data is as follows:

Supplier Name	Type	Weight	Max Order Quantity	Total Weight
Crompton	Electrical/Component	45	50	2,250.00
Aras	Electrical/Assembly	32	15	480.00
TECH CH	Electrical/Component	30	70	2,100.00
SKF	Mechanical/Component	20	1,000	20,000.00
Autodesk	Electrical/Assembly	100	16	1,600.00
BOSCH	Electrical/Component	23	45	1,035.00
Siemens	Mechanical/Assembly	50	99	4,950.00
Prorigo PPTX	Electrical/Assembly	100	1	100.00
Prorigo	Electrical/Assembly	500	2	1,000.00

Below the table, there are two report controls, both labeled 'Count vs Type', one under a 'Line Chart' header and one under a 'Bar Chart' header. The interface also includes a search bar, navigation buttons, and a 'Page 1 of 1' indicator.



Property Filter

Property Filter is used in various controls of the Aras ProAppDesigner like Advanced Filter and Column Filter. Using Property Filter, you can create complex boolean expressions. Expressions can contain:

- Item properties,
- operators,
- values,
- grouping operator ('(')'),
- logical operators ('AND', 'OR').

Various editors are used for showing values based on the selected property type:

- Existing expression can be updated by selecting individual sub-expression or elements with mouse.
- You can also navigate expression elements using navigation keys. Selected keyword can be deleted by clicking on the delete icon.
- If no element is selected, by clicking on delete icon will remove the whole expression.
- If no element is selected in the expression, by inserting new property expression or other operators will insert at the beginning of the expression.
- If any element is selected in the expression, insert operation will insert the new element right after the selected element.

The screenshot shows a 'Select Filter' dialog box with the following components:

- Property:** A dropdown menu with 'Type' selected.
- Operator:** A dropdown menu with 'EqualTo' selected.
- Filter Type:** A dropdown menu with 'Static' selected.
- Value:** A text input field containing 'Asse', followed by a green checkmark icon, a red 'X' icon, a blue checkmark button, and a blue 'X' button.
- Expression:** A text input field containing 'Type EqualTo Assembly'. Above this field are buttons for 'AND', 'OR', '(', ')', and a red trash can icon.
- Enforce Filter:** A checkbox that is currently unchecked.

For the date property, Value field can take a specific date or a keyword. In order to build a boolean expression based on current date, value can contain keywords CURRENT_DATE and CURRENT_DATETIME. These keywords will be replaced at runtime before evaluating the expression with actual current date or datetime.

Ex1: **Created On** LessThanEqualTo CURRENT_DATE

Ex2: **Effective Date** LessThanEqualTo CURRENT_DATE + 2

Ex2: **Modified On** LessThanOrEqualTo CURRENT_DATE – 7

Select Filter

Property: Created C, Operator: LessThar, Filter Type: Static, Value: CURREN

Expression: Created On LessThanOrEqualTo CURRENT_DATE

Enforce Filter

For the user property, Value field can take any text or keyword. In order to build a boolean expression based on current user, value can contain keyword CURRENT_USER. This keyword will be replaced at runtime before evaluating the expression with the logged-in user.

Ex1: **Created By** EqualTo CURRENT_USER

Ex2: **Owned By** NoteEqualTo CURRENT_USER

Select Filter

Property: Modified, Operator: NotEqual, Filter Type: Static, Value: CURRENT_USE

Expression: Modified By NotEqualTo CURRENT_USER

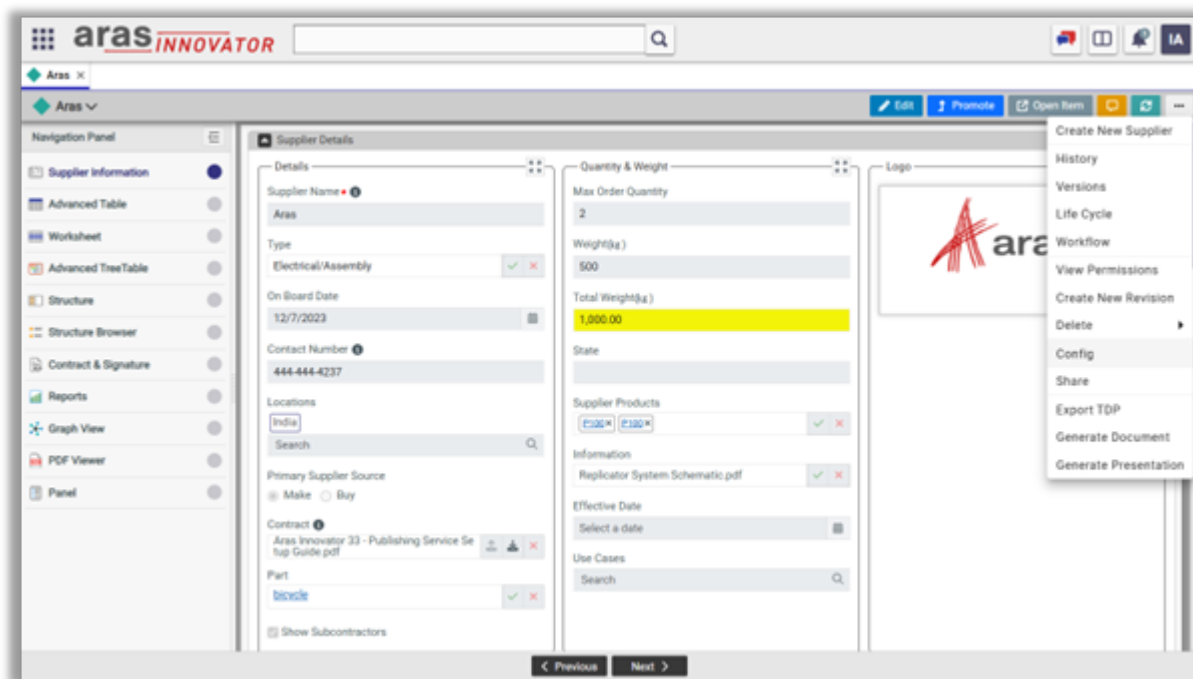
Enforce Filter



User Dashboard

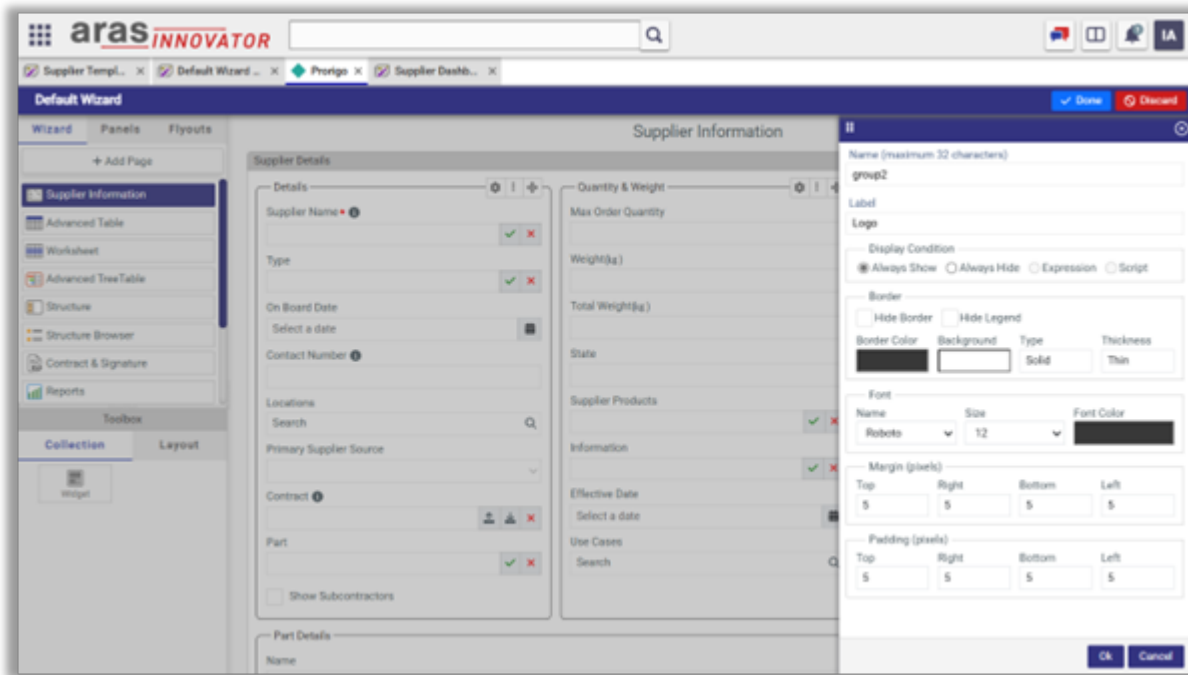
When admin selects “Enable Config” option on template, end user can configure his view based on the **default view** shown to him at runtime. When admin selects “Enable Sharing” option, end user can **share his customized view** with other users in the organization.

End users can create their own views based on the admin defined wizards and dashboards. For each ItemType, end users can create any number of views for the wizard and the dashboard. These views are shown along with the default views defined by the admin. To create the view upon clicking Config action, end user will be shown with the designer using which he can configure or hide existing controls from the layout. He can also add new widgets from widget libraries to the layout.



Upon selecting Config action from the dropdown, designer will be shown the current template, where end user can customize the view using designer capabilities. Using Display Condition option, end user can hide or show the control on the page. For collection controls like Table and Worksheet, by selection Settings icon, columns can be reordered or hidden from the view.

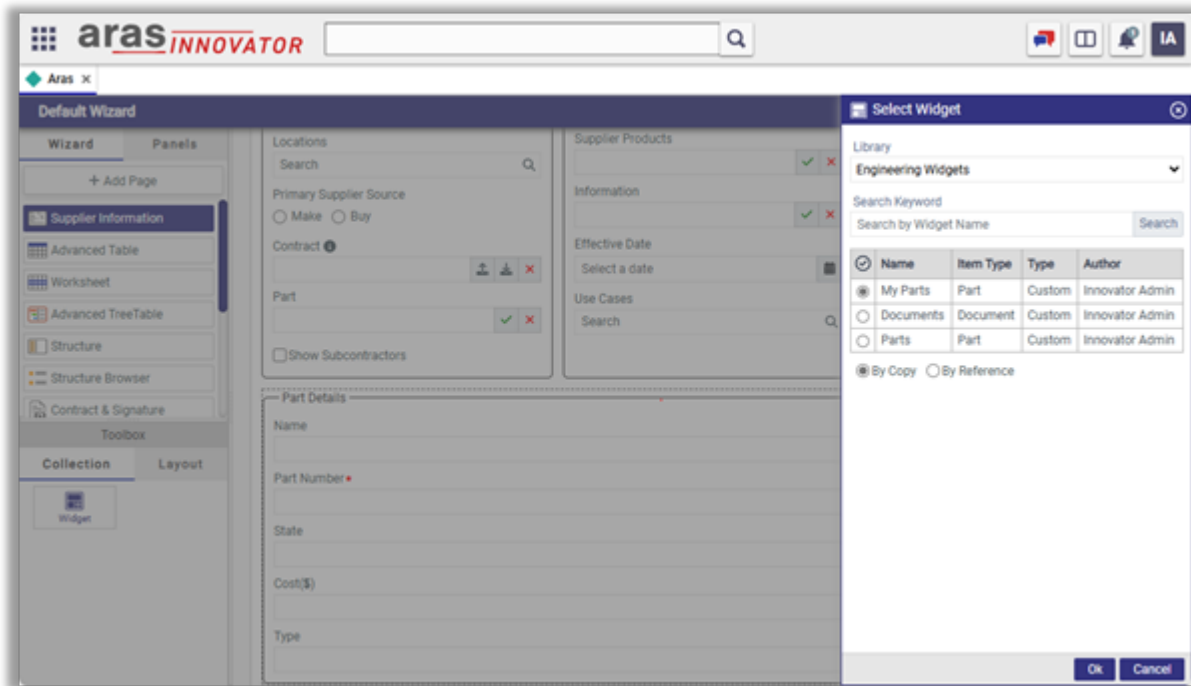




When admin creates any Widgets in the libraries, user can choose those widgets and place them on the page. Widgets are the predefined UI components, created from the existing collection controls of the template by the admin. Once defined can be used across many templates. Widgets are organized in libraries that are defined by admin. Admin can give access to the libraries based on user's roles. While creating wizard or dashboard template for a specific ItemType, admin can select a collection control and add it to the library.

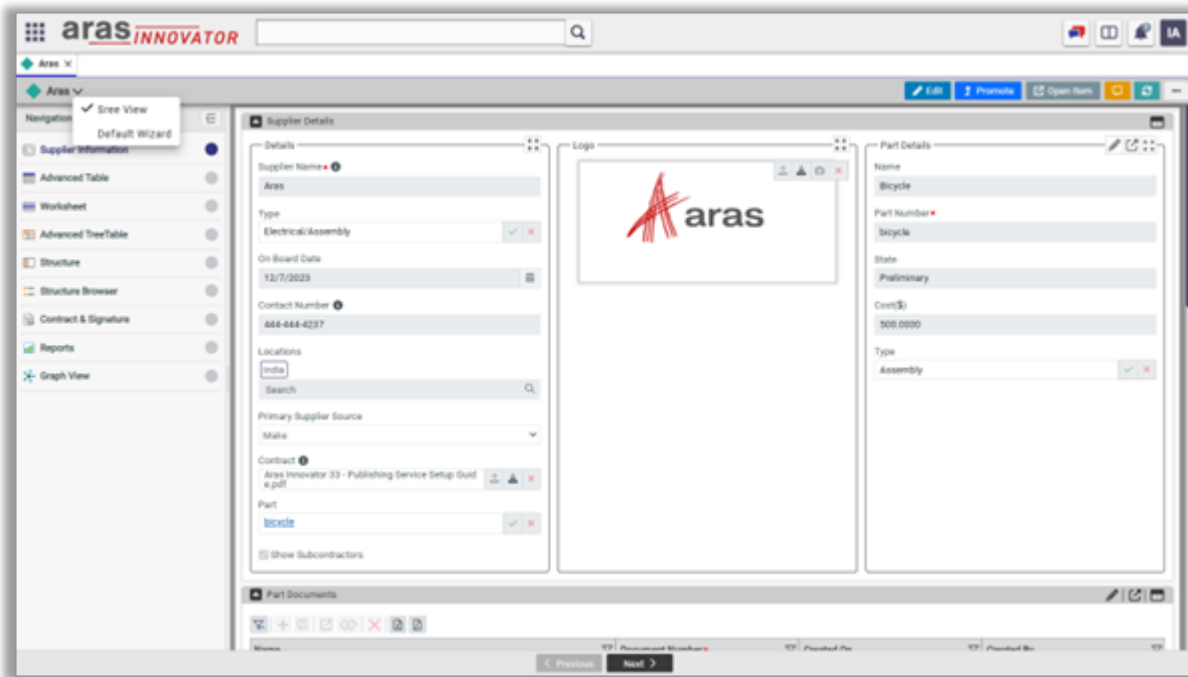
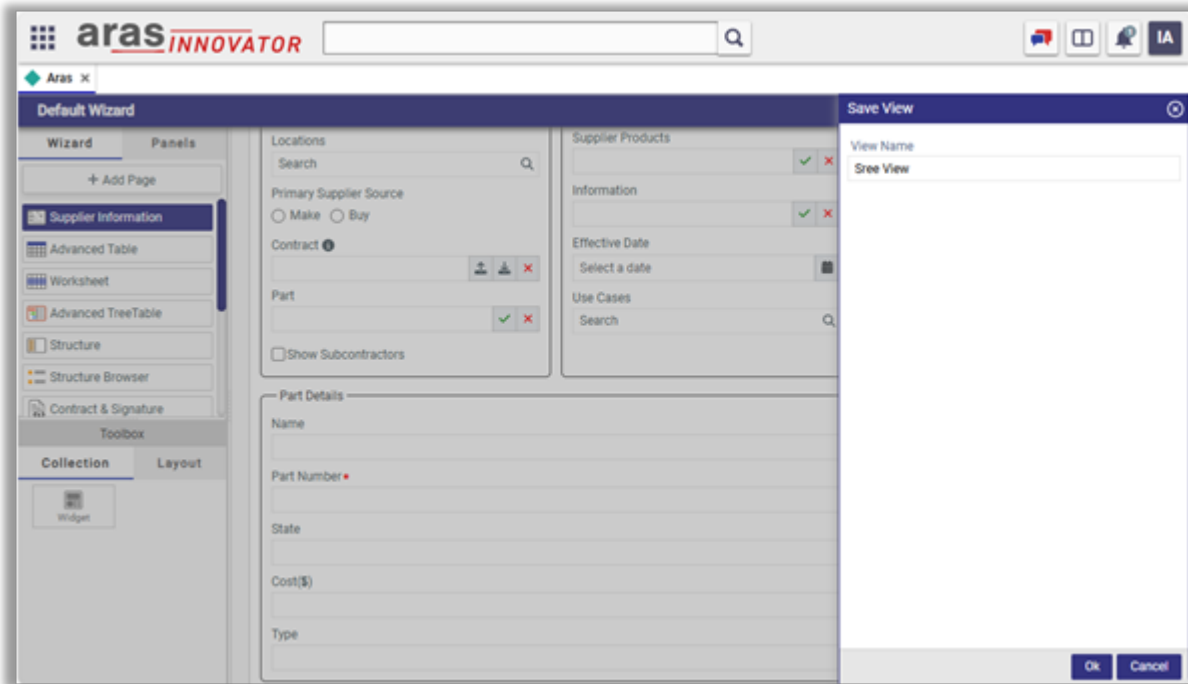
Adding a widget from library to the template can be done using Widget control from the toolbox. Widget can be added to template by-copy or by-reference by selecting appropriate option from 'Widget Settings' flyout. Adding widget by-copy, will copy the definition of the Widget to the template. Any subsequent updates done on the widget in the library will not reflect in the template if the widget is added by copy. By adding widget by-reference, will always show the latest widget based on its updates in the library.





After making the changes, when user clicks on the Save action, flyout will be shown to enter name for the customized view. A new template view will be created with the given view name for the current user.





End users can create custom views for Global and Type dashboards similar to the Wizard.



The screenshot displays the Aras Innovator Global Dashboard. The interface includes a search bar at the top, a navigation menu on the left, and several data tables. A dropdown menu is open over the 'Requirements' table, listing options like 'Requirements', 'Simulation', 'My Tasks', and 'Global Dashboard'. The 'Requirements' table lists items with columns for Requirement Number, State, Type, and Group. The 'Simulation Tasks' table lists tasks with columns for Task Number, Name, Type, Stage, and Task Owner. The 'Parts' table lists parts with columns for Part Number, Name, Type, Weight (g), and State. The 'Documents' table lists documents with columns for Document Number, Name, Type, and State.

Requirement Number	State	Type	Group
REQ-000000017	Draft	Requirement	Mechanical
REQ-000000018	Draft	Requirement	Mechanical
REQ-000000022	Draft	Requirement	Mechanical
REQ-000000023	Draft	Requirement	Mechanical
REQ-000000025	Draft	Requirement	Mechanical
REQ-000000026	Draft	Requirement	Mechanical
REQ-000000027	Draft	Requirement	Mechanical
REQ-000000028	Draft	Requirement	Mechanical
REQ-000000029	Draft	Requirement	Mechanical
REQ-000000030	Draft	Requirement	Mechanical

Task Number	Name	Type	Stage	Task Owner
SM-000021	Mechanical Plug Socket 2P+0	Service Task	Analysis	Dan Park
SM-000016	Frequencies Analysis	Manual Task	Analysis	Dan Park
SM-000009	Convective Heat Transfer Analysis P2	Manual Task	Analysis	Dan Park
SM-000011	Postprocessing p1-3 Results	Manual Task	Analysis	Dan Park
SM-000012	Test Data Curve Fitting	Manual Task	Report	Simulation Engineer
SM-000006	CAD model preparation for CFD analysis	Manual Task	CAD Preparation	Dan Park
SM-000019	Compare with As Designed	Manual Task	Analysis Post	Dan Park
SM-000004	Extruder Mesh with Contacts Refinement	Manual Task	Mesh	Dan Park
SM-000003	Extruder Thermal Analysis	Manual Task	Analysis	Dan Park
SM-000022	Socket Mechanical Unplug 2P+0	Service Task	Analysis	Dan Park

Part Number	Name	Type	Weight (g)	State
1	CYCOMP	Software		Preliminary
2	FR-4441	Orange 53.5 cm Carbon Fibre Frame		Preliminary
3	FR-4447	Glossy Black 46.5 cm Carbon Fibre Frame		Preliminary
4	FR-4464	Red 23 inch cm Titanium Frame		Preliminary
5	FR-4625	Yellow Black 17 inch cm Aluminum		Preliminary

Document Number	Name	Type	State
0008	DOC-Coating	Specification	Preliminary
100001	Test Office Connector	Ad-Hoc	Preliminary
100002	Mac Doc 1	Ad-Hoc	Preliminary
100003	Requirement Change	Ad-Hoc	Preliminary



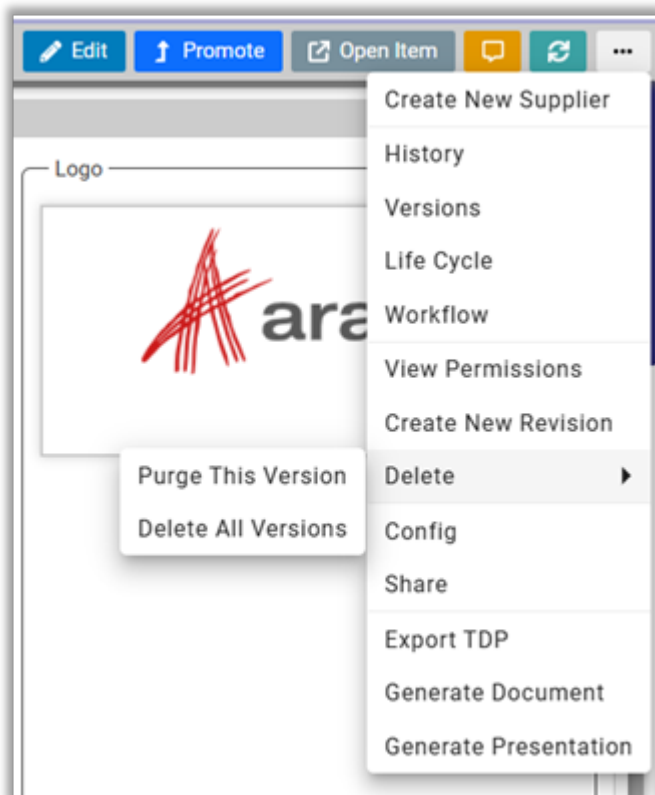
Wizard

Wizard is a separate single page application (SPA) which shows details of the item at runtime. When you select item from Innovator search grid, if the item has published Aras ProAppDesigner template, item details will be shown using the template instead of using default Innovator form. When you click on Create action from search grid, same template will be used to show the page with empty controls.



Wizard Actions

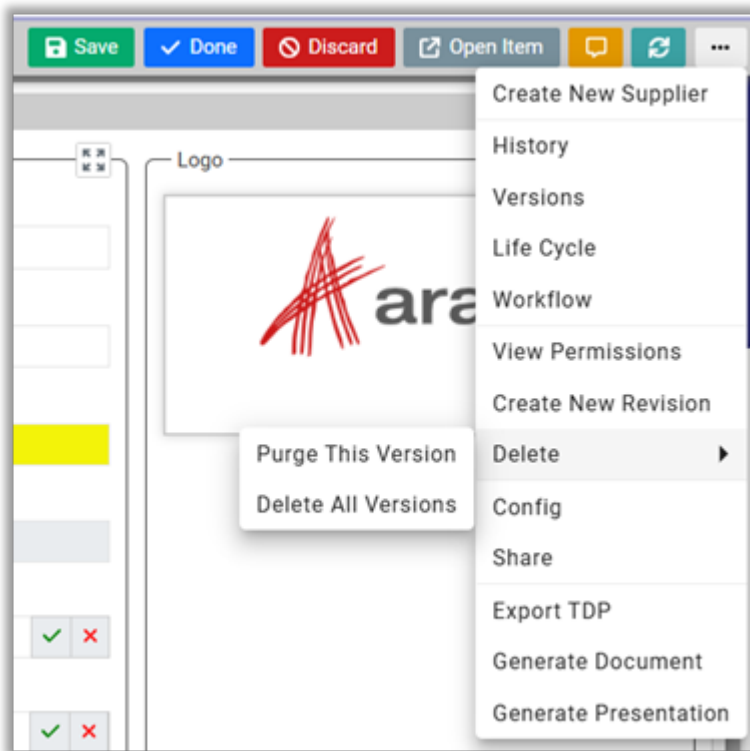
Wizard actions are shown in the wizard in the top-right corner when the item is opened. When you open item in read-only mode, you will see actions like Edit, Promote and Open Item.



- **Edit:** Applies lock on the item with exclusive access to make changes on the item.
- **Promote:** Promotes the item to the next state of the lifecycle associated with the item.
- **Open Item:** Shows new Innovator tab with default form that shows item details in classic view.
- **Discussion Panel:** Shows discussion panel as a flyout with all messages and their replies exchanged in the current item context.
- **Refresh:** Reloads the current item by making server request.
- **Create New Item:** Shows new item page in a different tab.
- **History:** Shows history of all the changes made to current item.
- **Versions:** Shows list of all previous generations of the current item.
- **Life Cycle:** Shows associated life cycle of the current item.
- **Workflow:** Show associated workflow of the current item.
- **View Permissions:** Shows permissions associated with the current item.
- **Create New Revision:** Creates a new revision for the current template item.
- **Delete:** Allows to delete current generation or all generations of the item.

- **Config:** Config action is shown only when admin has configured the template view in order to allow end user to customize the layout of the wizard of a given ItemType.
- **Share:** Share action is shown only when admin has configured the template view in order to allow end user to share the layout of the wizard of a give ItemType with other users.

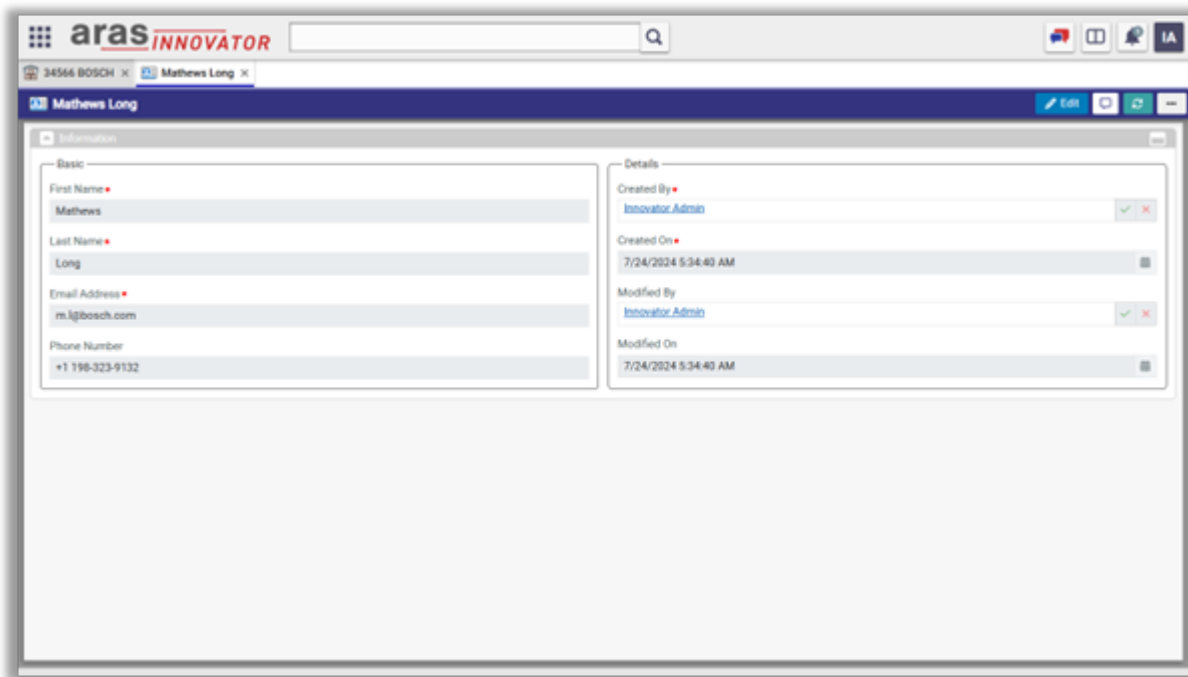
When you open the item in edit mode, following actions will be shown.



- **Save:** Saves the changes done so far on the wizard. Once changes are saved, you can't discard them using Discard action.
- **Done:** Saves the changes done so far on the item and unlocks the item so that other people can make changes.
- **Discard:** Discards all the changes done so far on the item after the last save and unlocks the item so that other people can make changes.

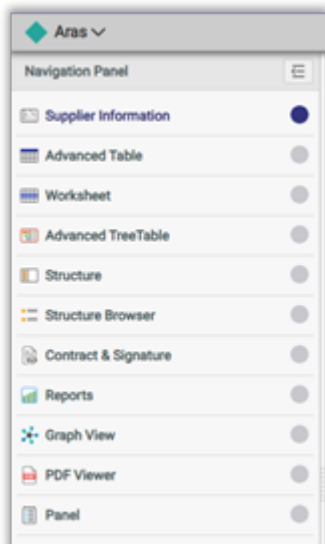
Single Page

The template created in Aras ProAppDesigner can be single page or multi page. If the template has only one page, then it is called single page application. In case of single page wizard, item keyed-name will be used to show the page title.



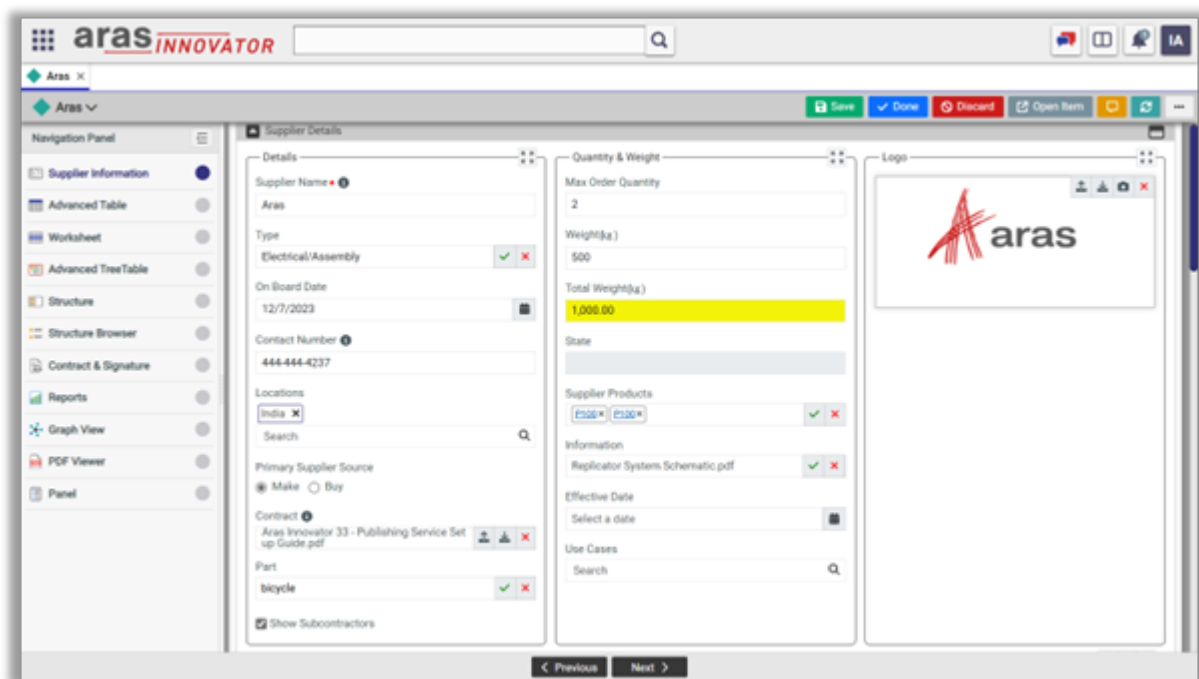
Multi Page

If the template created in Aras ProAppDesigner has more than one page, then it is called multi page application. Multi page application is shown with Navigation panel on the left which displays all the pages of the application. Active page is shown with blue-circle icon, already visited or filled pages are shown with green-circle with tick icon. If the page is not visited, then it will be shown with gray-circle with tick icon. If the page is visited but there are validation errors, then it will be shown with red-circle with tick icon. Along with the Navigation panel, it shows the Navigation Bar at the bottom with buttons Previous and Next. Previous button is disabled when you are on the first page and Next button is disabled when you are on the last page. Sometimes, all the pages may not be shown on the Navigation panel if those pages are defined with Display Condition. Display Condition expression is evaluated every time there is a change in contained property values.



Page Navigation

- You can navigate through the pages by clicking Next and Previous buttons in the bottom Navigation Bar. You can navigate to next page by clicking Next button, as part of moving away from the page, all validation rules on the page are evaluated. If any of the rules are not satisfied, it shows popup dialog with Validations Failed message and icons against each control.
- By hovering the mouse on the icon, specific errors or warning messages are shown based on the defined validation rules on the control. You can move away from the current page by clicking any other page from the Navigation Panel. If the current page has any validation errors, a popup dialog will be shown with options to ignore and move forward or fix the issues before moving forward.
- Every time when you save the item if you have "as_is_validated" property on the ItemType, it will save the status of validations against that property. That property value will be set to true if all pages in the wizard have no validations errors. Even if a single page has validation errors, property value will be set as false.



Page Controls

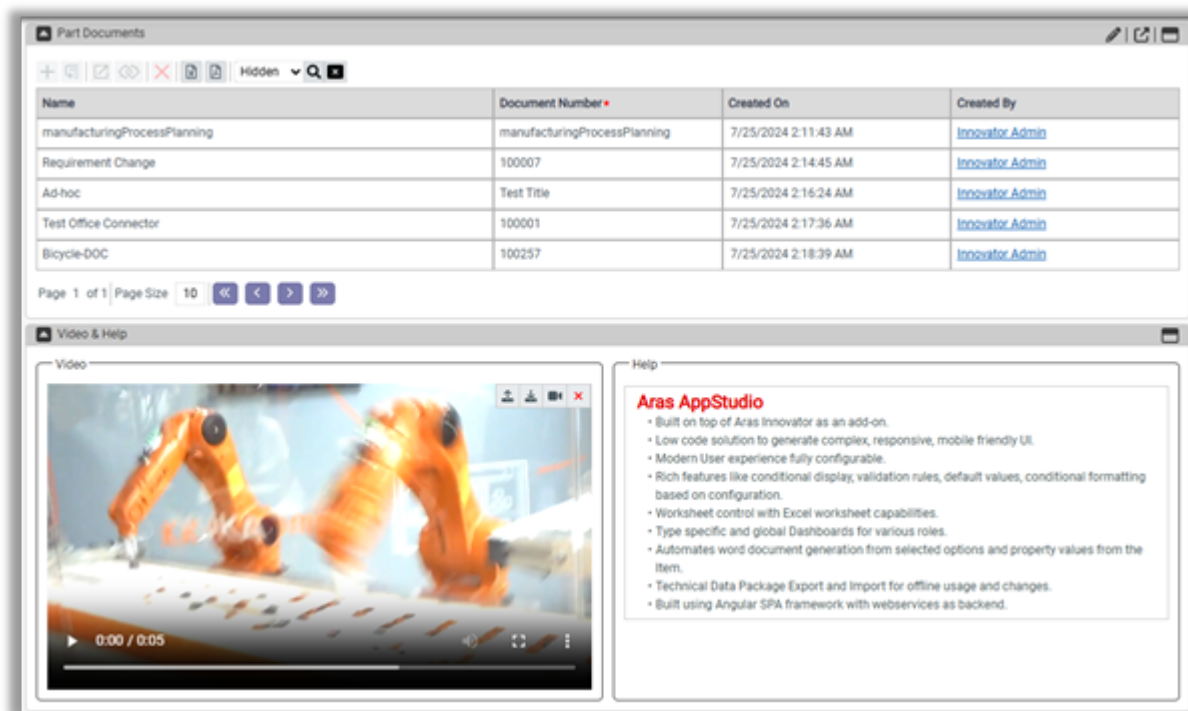
Every page is composed of layout controls, simple controls, collection controls. You can also define interdependencies between controls using features like **Display Conditions**, **Validation Rules**, **Conditional Formatting** and **Reset Dependencies**; these topics are covered later in the document.

Layout Controls

Layout controls can organize simple and collection controls with a proper layout in the page.

Section

Section control is a layout control to organize group controls placed inside it. Section controls are vertically stacked one over the other. Section controls are the top-level container controls exists in pages and flyouts. By defining display condition or access permissions, section control can be shown or hidden based on the evaluation criteria. In the picture below, Part Documents and Video & Help represents sections in the page.

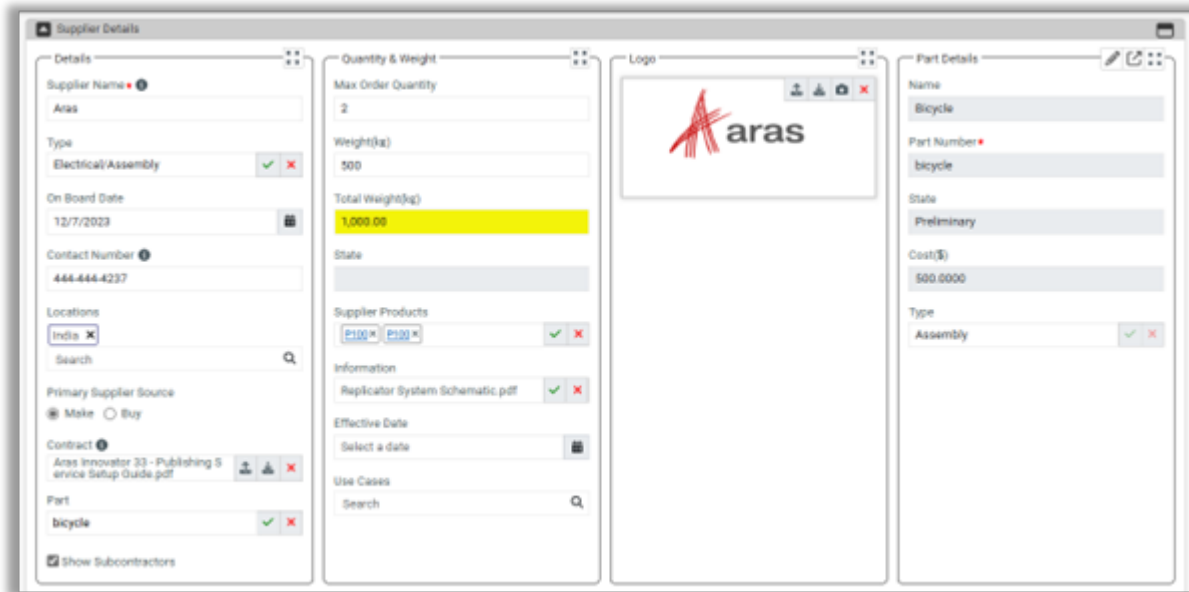


Group

Group control is a layout control to organize child controls placed inside it. Group controls are horizontally stacked beside each other inside a section control. When multiple group controls are placed inside section control, they equally take the available width. When you resize the page on desktop, based on the available width they can be shown in single row or multiple rows inside a section control. In mobile view, they will be vertically stacked inside a section control. By defining



display condition or access permissions, group control can be shown or hidden based on the evaluation criteria. In the picture below, Details, Quantity & Weight, Logo, Part Details represents groups within 'Supplier Details' section.



Panel

Panel control is the layout control that can help you to create a complex layout on the page. Panel control can be created just like a page with section controls within it. Panel control can be placed within group control just like any other child control. It can also be used as tabs within Tab control. In picture below, 'Supplier Files' represents panel control, that is placed inside the group.

The screenshot shows a form with two columns. The left column is titled 'Details' and contains the following fields: 'Supplier Name' (with a red dot icon), 'Type', 'On Board Date' (with a calendar icon), 'Contact Number' (with a red dot icon), 'Locations' (with a search icon), 'Primary Supplier Source' (with a dropdown arrow), 'Contract' (with a red dot icon and upload/download icons), 'Part', and a 'Show Subcontractors' checkbox. At the bottom of the left column is a 'Supplier Files' section with a toolbar and a table with columns 'File Name', 'File Type', 'major_rev', and 'Created On'. The right column is titled 'Quantity & Weight' and contains: 'Max Order Quantity', 'Weight(kg)', 'Total Weight(kg)', 'State', 'Supplier Products', 'Information', 'Effective Date' (with a calendar icon), and 'Use Cases' (with a search icon).

Tab

Tab control can be used to show associated panels as tabs. In the picture below, Information and Parts together represents Tab control built based on the panels.

The screenshot shows a window titled 'Supplier Details' with a tab control at the top. The 'Information' tab is selected, showing a 'Details' panel with fields for 'Supplier Name' (Aras), 'Type' (Electrical/Assembly), 'Weight' (500), 'Max Order Quantity' (2), 'On Board Date' (12/7/2023), 'Contact Number' (444-444-4237), and 'Part' (btcs1e). The 'Parts' tab is also visible. To the right of the 'Information' panel is a 'Logo' panel displaying the Aras logo.

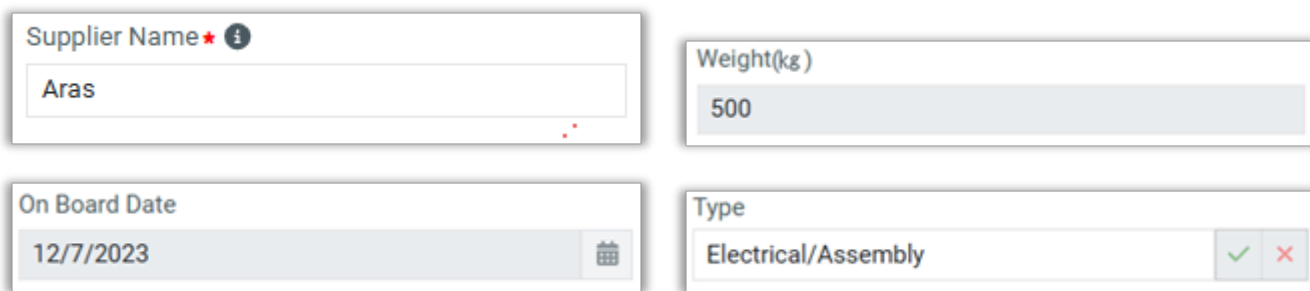


Simple Controls

Simple controls can show data related to an item or its properties.

Text

Text control is used to display properties of datatypes String, Text, Integer, Float, Decimal, Date, Sequence and Classification. Text control honors Format Specifier set on the ItemType definition. Text control honors the default value set through ItemType definition. You can also set default value for Text control by defining text expression with properties from other controls from the current page or previous pages. Text control has additional settings like Display Condition, Validation Rules, Conditional Formatting etc. that are covered in the subsequent sections.

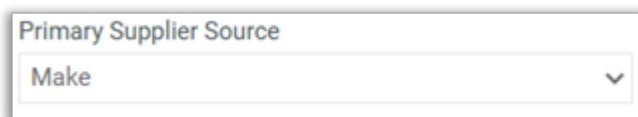


The image displays four examples of text controls in a grid layout:

- Supplier Name**: A text input field containing the value "Aras". It includes a red asterisk and an information icon to the left of the label.
- Weight(kg)**: A text input field containing the value "500".
- On Board Date**: A date picker control showing the date "12/7/2023" and a calendar icon to the right.
- Type**: A dropdown menu showing the selected value "Electrical/Assembly". It includes a green checkmark and a red 'X' icon to the right of the field.

List

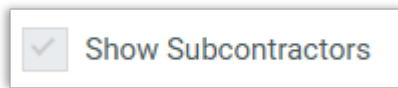
List control is used to display properties of datatypes List, Filter List and Multi Value List. While displaying list, Aras ProAppDesigner uses same underlining list with options defined in the Innovator. List control comes with Control Style setting, with this, options can be shown as RadioButtonGroup in case of single selection list or CheckboxGroup in case of multi selection list. If the Control Style is not set, default Dropdown will be used to show the options. List control honors the default value set through ItemType definition.



The image shows a single dropdown list control. The label is "Primary Supplier Source" and the selected value is "Make". A small downward arrow is visible on the right side of the dropdown box.

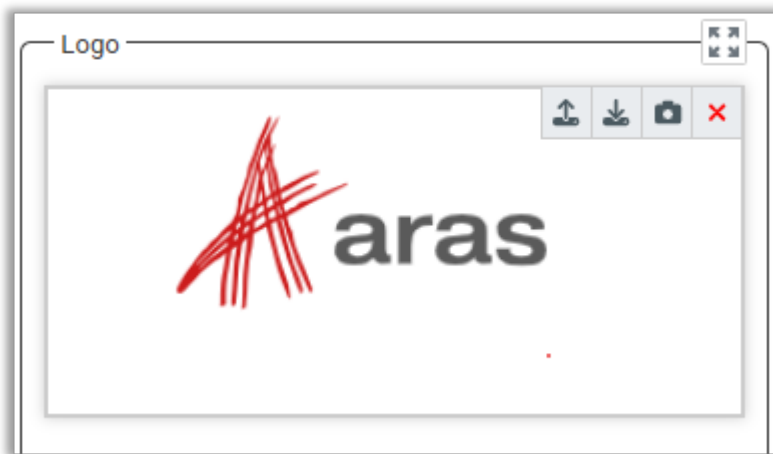
Checkbox

Checkbox control is used to display properties of datatype Boolean from the associated ItemType.



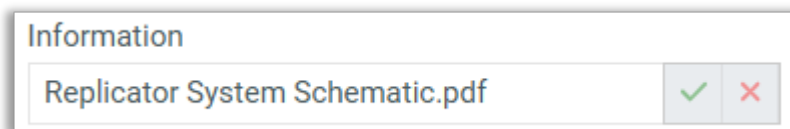
Image

Use Image control to display properties of datatype Image from the associated ItemType. Image control will show actions to upload picture, download picture, capture picture, and remove picture. Capture picture can be used to capture picture directly from the camera while using the application in the mobile device.



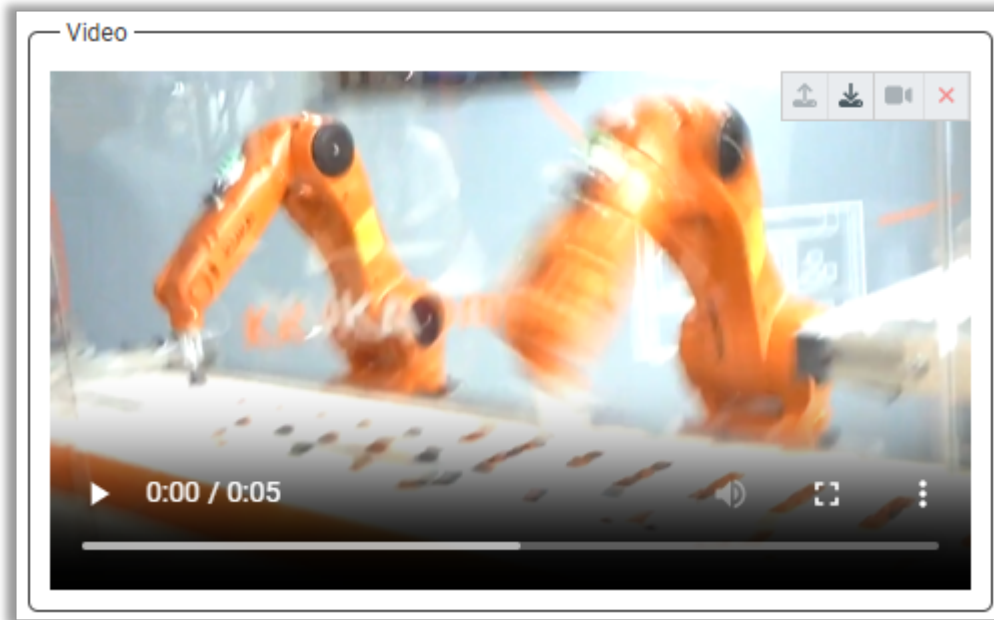
File

Use File control to display properties of datatype File from the associated ItemType. Admin can setup the file control in File or Video mode inside template definition.



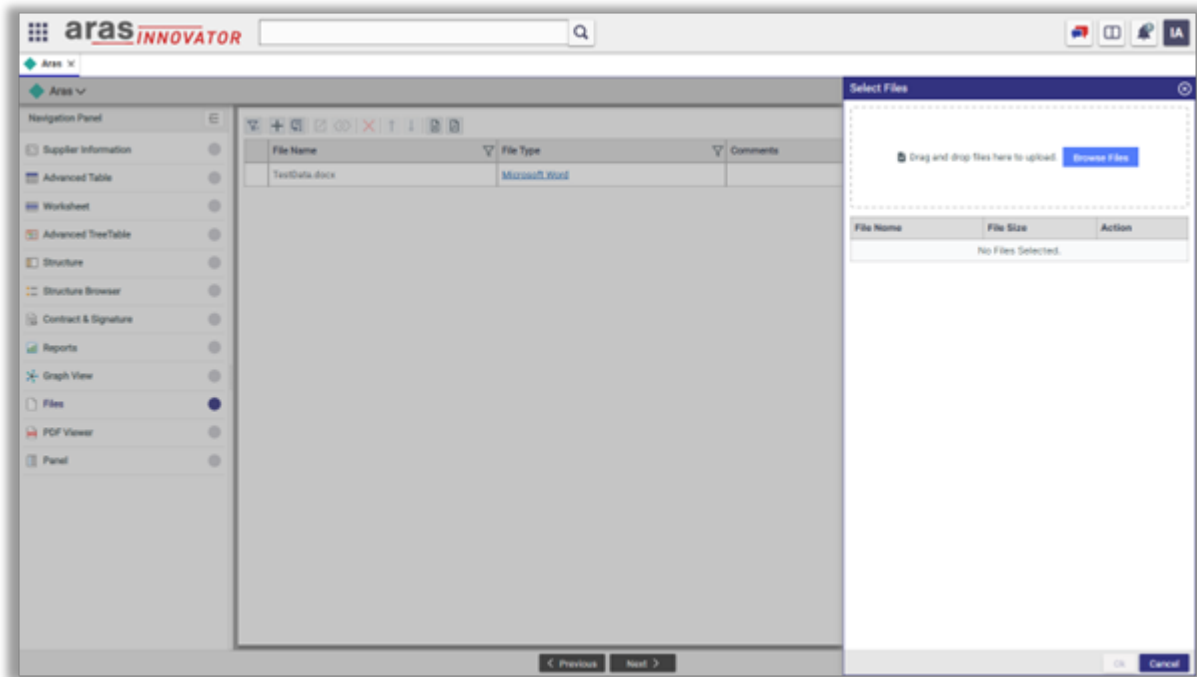
When it is set in video mode, it will show actions to upload video, download video, capture video, and remove video. Capture video can be used to capture video directly from the camera while using the application in the mobile device. It also has inbuilt support to play the video.





File control on the form support drag & drop of files from the local file system. When you drag & drop a file from the windows explorer to file control on the form, it will show dropped file path in the control.




File property used in collection controls columns shows a file selection icon in the column header to support bulk file upload in the collection control. Upon selecting the icon shows 'Select Files' flyout with drop area. It also shows 'Select Files' flyout when you select Add action from Files table.





Document

Document control allows you to select the document which is stored in Innovator. Upon clicking on tick icon of the control, Select Document dialog will be shown. After entering search criteria, Select Document dialog shows all the files of type Word, PDF and Image. Selected document is stored on the context item as file property. Document control is useful in case you want to generate Word/PDF document from the context item.


Contract ⓘ


Aras Portal - User Guide.docx   

Select Document 

Document Type
Document 

Property Relationship

Select Relationship Type
Document File 

Select Property of Relationship
Related File 

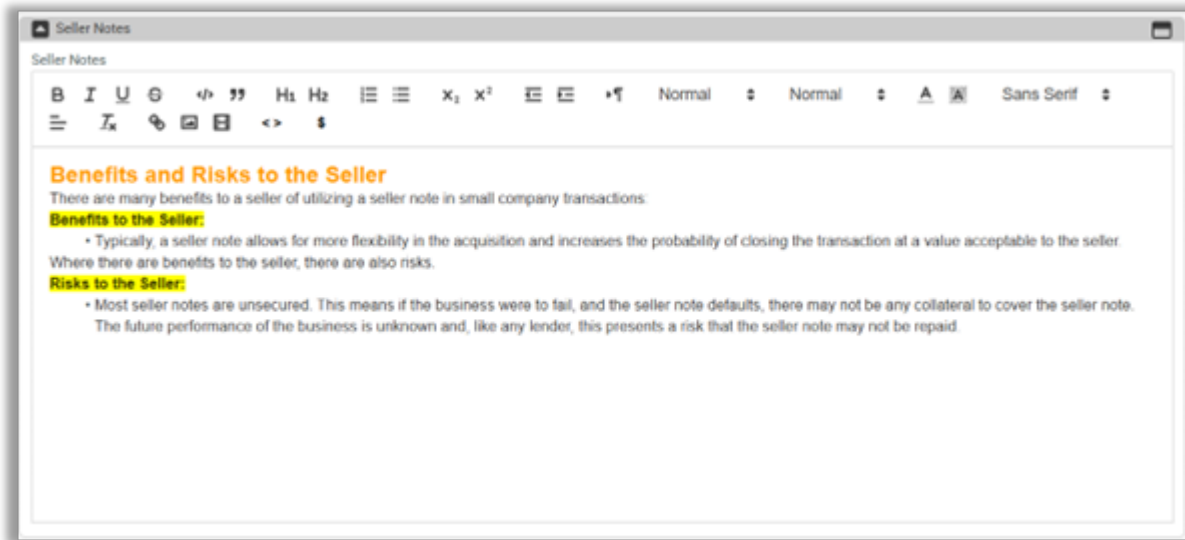
Search Keyword (Displays only top 100 matches)
Search by document name

<input checked="" type="checkbox"/>	Name	Related File
<input type="checkbox"/>	PLM System Migration	PLM System Migration.pptx
<input type="checkbox"/>	SM0014	MP0101CCCC Geometry.wcax
<input type="checkbox"/>	SM2000002	Convective Heat transfer within 3D printer STUDY Report.pdf
<input type="checkbox"/>	0006	Costing Sheet.txt
<input type="checkbox"/>	SM0013	Tuned PID Controller.png
<input type="checkbox"/>	SM000002	SF4_Mechanical_Plug.py
<input type="checkbox"/>	SM0015	Mesh002c.wcax
<input type="checkbox"/>	SM0012	CurveFitting001.sce
<input type="checkbox"/>	SM000003	SF4_Mechanical_Unplug.py

RichText

RichText control displays properties of datatype FormattedText from the associated ItemType. In edit mode, RichText control shows toolbar, using it you can format the text with bold, italic, font, color etc. It also allows you to insert items like images, hyperlinks and videos, upon saving all these items are embedded inside FormattedText on the property.





xClassification

xClassification control associates xClassification tree nodes and xProperties with the item. It works exactly same way as you work with regular Innovator forms.

Signature

Signature control captures your digital signature as image and can be saved to image property of the context item. Signature control shows a signature pad, you can use mouse to draw your signature.



Item

Item control displays properties of datatype Item from the associated ItemType. Using this control, you can select existing item of the property's Data Source ItemType through Select Dialog.

Part

bicycle ✓ ✗

Select Part ✕

Search Keyword (Displays only top 100 matches)

Search by Item name Search

Property Filter

Property Operator Value

+

Expression AND OR () ✖

+

<input checked="" type="checkbox"/>	Name	Part Number	State
<input type="checkbox"/>	Polyurethane Cover	100013	Preliminary
<input type="checkbox"/>	Velcro Belt	10012	Preliminary
<input type="checkbox"/>	Acetate Filter	Acetate Filter	Preliminary
<input type="checkbox"/>	Aluminum Foil	Aluminum Foil	Preliminary
<input type="checkbox"/>	Axle	AXL	Preliminary
<input type="checkbox"/>	Axle coming from GM V	AXL-GMV	Preliminary
<input type="checkbox"/>	Axle coming from SVC and light weight	AXL-SVC	Preliminary
<input type="checkbox"/>	Axle coming from TVS - Heavy Duty	AXL-TVS	Preliminary
<input type="checkbox"/>	Non-Threaded 63mm Bottom Bracket	BB-9090	Released
<input type="checkbox"/>	Threaded 63mm Bottom Bracket	BB-9091	Released

Ok Cancel

Items

Use Items control to select and display list of items with their keyed-names from the associated ItemType. Using this control, you can select existing items of the property's Data Source ItemType through Select Dialog. Selected items will be saved as relationships upon saving the context item.



Supplier Products

P100, P100 ✓ ✗

Select Product

Search Keyword (Displays only top 100 matches)

Search by item name Search

Property Filter

Property	Operator	Value
<input type="text"/>	<input type="text"/>	<input type="text"/>

Expression AND OR () +

<input checked="" type="checkbox"/>	Product Number	Name
<input checked="" type="checkbox"/>	P100	Toolbox
<input checked="" type="checkbox"/>	P100	engine2stroke screw
<input type="checkbox"/>	P101	ToolboxStand
<input type="checkbox"/>	P78	windshaft
<input type="checkbox"/>	P98	cableline 98

Ok Cancel

ButtonGroup

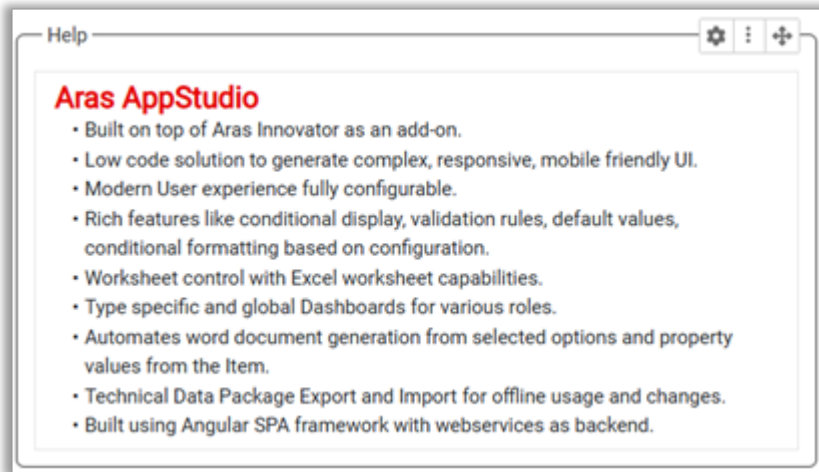
ButtonGroup control displays a group of buttons or a single button. Button control can be associated with a custom action defined by admin. Each button can be configured to show a dialog or a shortcut to the existing Innovator page. Button can be also be configured to call a custom script.



Banner

Banner control shows static content like instructions or help on the page. Banner control with static content is added through rich-text editor inside the template. At runtime, it shows rich-text to the user.





Frame

Frame control displays content from the external page. It can be used inside the page to show a report generated from the external BI tool.



HTML

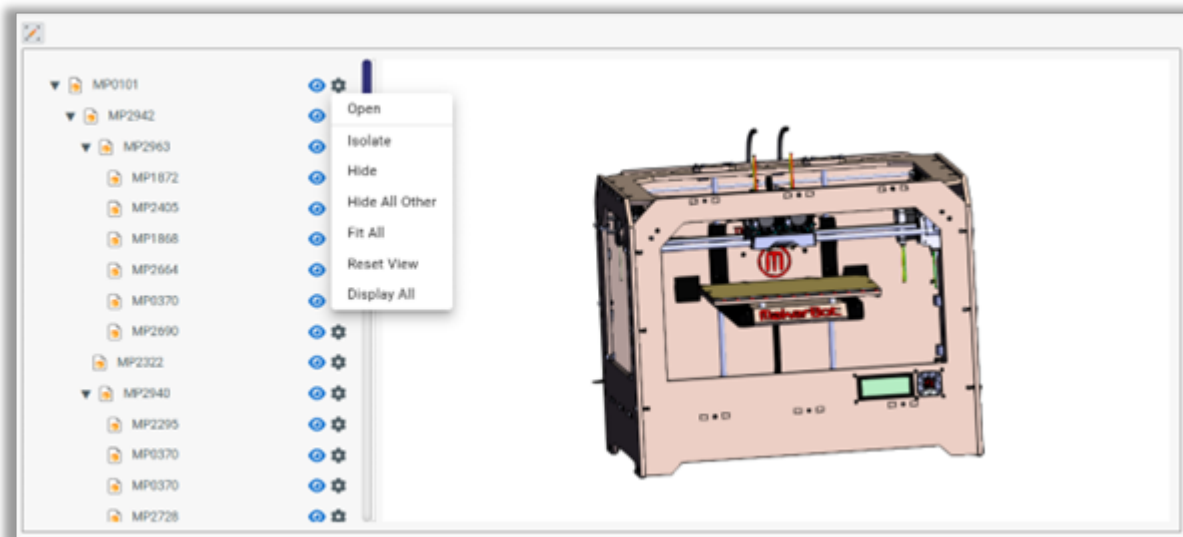
HTML control displays custom UI in the pages and dialogs. HTML content added to this control will be rendered in the same DOM as of the Aras ProAppDesigner Wizard.






CAD Viewer

3D Viewer displays 3D Models developed on software related to CAD. It provides the external users with a detailed view of the item. 3D Models are shown on the right panel and a Tree View of all the individual items is shown on the left. 3D Models can be moved in any direction and can be zoomed in or out, by using the basic mouse and touch (mobile) actions.



Toolbar Options:

In the Tree View each tree item has an option to show or hide it, along with other options stated below:

- **Open:** Opens a new tab for the individual item, to access all its information.
- **Isolate:** Focuses and shows the current item, by decreasing the opacity of all other items.
- **Show / Hide:** Shows or hides the current item.
- **Hide All Other:** Hides all items excluding the current item.
- **Fit All:** Resizes the 3D model based on the size of the screen.
- **Reset View:** Reloads the 3D model.
- **Display All:** Shows all items 

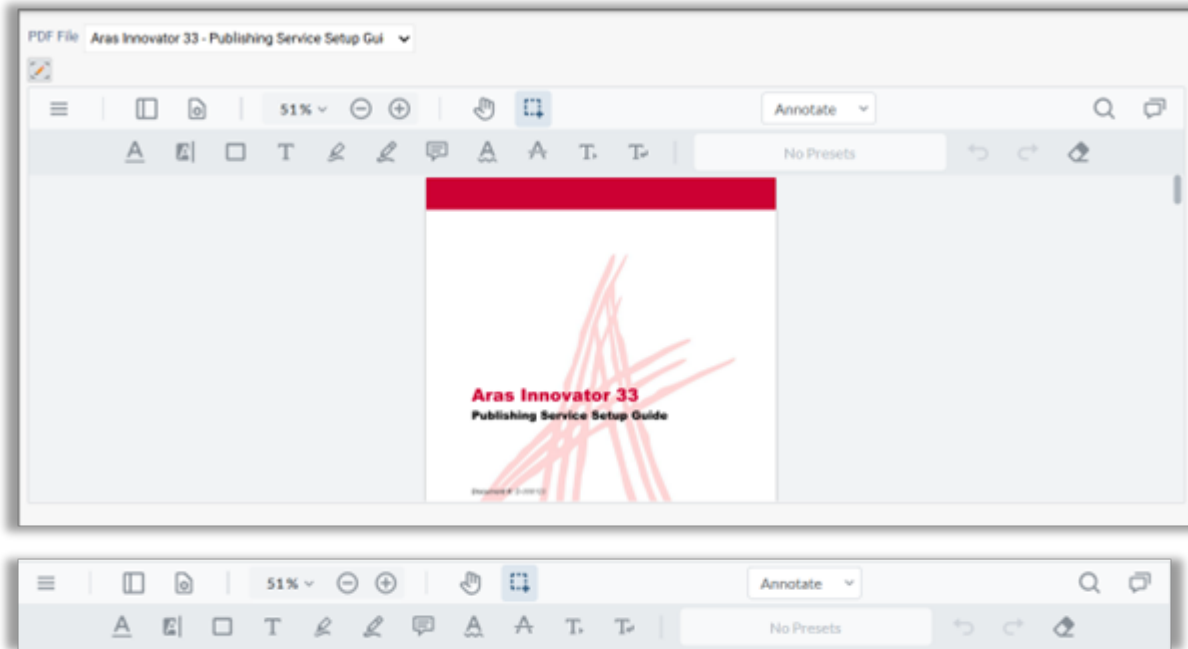
Markup:

Click on the Markup button to take a snapshot of the 3D model view and open it in the Image Viewer. You can perform all the basic image editing actions and download the image from the Image Viewer. You can return to the 3D Viewer by clicking the top left button of the Image Viewer.

PDF Viewer

PDF Viewer allows to view 3D PDF and PDF documents associated to that item. PDF Viewer consists of a Toolbar with Panel, View Controls, Zoom, Pan and Select options on the left. While Search and Menu options are on the right





Toolbar Options:

All Toolbar options are briefly explained below.

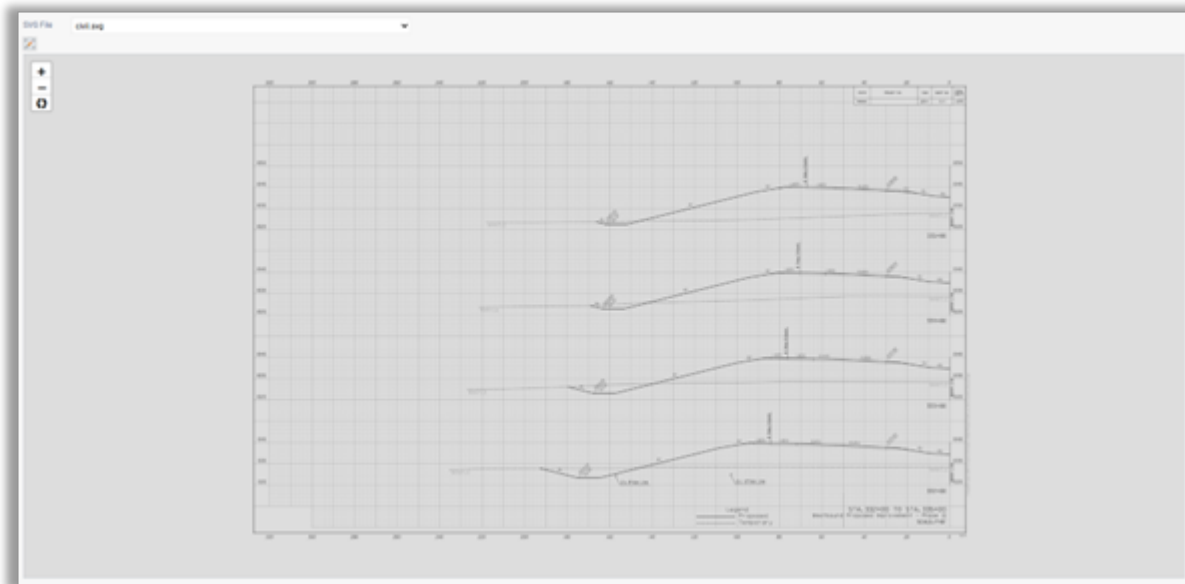
- **Panel:** Opens left panel showing preview of all the pages in the PDF, can be used to move across various pages.
- **View Controls:** Provides options to change the Transition, Orientation & Layout of the page view.
- **Zoom:** Provides options to Zoom in, Zoom out, Fit to width, Fit to page, Marquee Zoom and other fixed Zoom options.
- **Pan:** Moves the models or pages.
- **Select:** Selects text or models in the page.
- **Search:** Opens a right panel where user can perform search operations on the PDF document.
- **Menu:** A dropdown shows up containing the options below:
 - **Full Screen:** View the PDF in full screen mode.
 - **Download:** Download the PDF document.
 - **Print:** Take a printout of the PDF document.
 - **Dark / Light Mode:** Change the theme of the viewer.
 - **Languages:** Change the language of all the options available in the viewer.

Markup:

Click on the Markup button to take a snapshot of the page view and open it in the Image Viewer. You can perform all the basic image editing actions and download the image from the Image Viewer. You can return to the PDF Viewer by clicking the top left button of the Image Viewer.

Image Viewer

Use ImageViewer control to render SVG and image files associated with a Document item. Image viewer show list of all files in the dropdown associated with document item. It also shows controls to zoom in, zoom out, and refresh.



Markup:

Click on the Markup button in the toolbar to take a snapshot of the image and open it in the editor. You can perform all the basic image editing actions and download the image from editor. You can return to the Image Viewer by clicking View button from the toolbar.

Collection Controls

Collection controls can show data related to collection of items and their relationships.

Table

Table control shows relationship or reference data of the context item. At the time of adding table control to the form, properties of RelationshipType or ReferenceType are mapped to table columns. Table control has multiple settings that control its appearance and behavior at runtime.

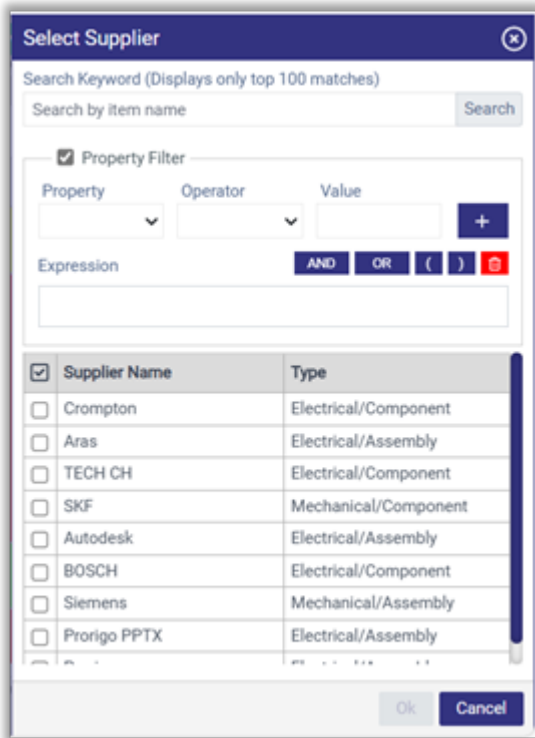
Supplier Name	Type	Weight(kg)	Max Order Quantity	Total Weight(kg)	Seller Notes	File
Aras	Electrical/Assembly	32	15	480.00	All parts need to be inspected for acceptance	Issue Tracker.xlsx
Siemens	Mechanical/Assembly	50	99	4,950.00	Siemens is a German multinational conglomerate corporation and the largest industrial manufacturing company in Europe. It is headquartered in Munich and has several foreign branch offices. Customer service: 1800 209 1850	
Autodesk	Electrical/Assembly	100	16	1,600.00	For more information visit Autodesk site	
Crompton	Electrical/Component	45	50	2,250.00	Crompton Greaves Consumer Electricals Limited is an Indian	

Above the table control, on the left end you will find the label and the Toolbar on the right end. If the label is absent the Toolbar is on the left.



Below are all the actions available in the Toolbar.

- **Add:** Add a row after the selected row. Added rows appear in light green color in the table. Updated rows will be shown in light orange color.
- **Insert:** An item selection flyout shows up. You can add rows to the Table from the items which are already available.



Components of item selection flyout will be as follows:

- **Search Options:** A search input field with a search button is available at the top within the flyout. By clicking on the search button, you filter the items grid.
- **Property Filter:** By clicking over the Property Filter check box available below the search, you perform advanced search operations on group containing various options to create custom expressions.
- **Items Grid:** All the items of that ItemType are shown in the grid with respect to the Search & Property Filter conditions.
- **Inserting Rows:** By clicking on the Ok button at the bottom right, you add all the items selected to the Table after the selected row or at the end. Added rows appear in light green color in the table.
- **Close:** By clicking on the Cancel button at the bottom right or the close button at the top right, you close the flyout without inserting.
- **Open Related Item:** By clicking on the Open Related Item action, you open a new tab by loading the related item of the selected row.
- **Open Relationship Item:** By clicking on the Open Relationship Item action, you open a new tab by loading the relationship item of the selected row.
- **Up:** By clicking on the Up action, you move the selected row above the previous row.
- **Down:** By clicking on the Down action, you move the selected row below the next row.
- **Delete:** By clicking on the Delete action, you delete the selected row from the Table.
- **Lock:** By clicking on the Lock action, you lock the selected row for exclusive edit. If the associated item with row is locked by someone else, you won't be able to lock the row.
- **Export Excel:** By clicking on the Export Excel action, you download all the visible data shown in the table to an excel file.
- **Export PDF:** By clicking on the Export PDF action, you download all the visible data shown in the table to a PDF file.



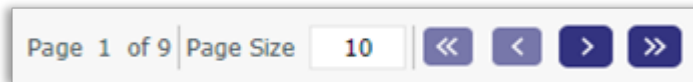
- **Filter:** By clicking on the Hidden dropdown, you select Simple option to show column filters on the table. You enter filter values on individual columns and click on the search icon beside the hidden dropdown to filter the rows based on the criteria. Beside the search icon there is also a clear option to clear all the filters
- **Custom Action Buttons:** Additionally, based on the requirements admins can configure custom action buttons for the table.

These two actions below are **exclusively** available in the **Toolbar for the Dashboards:**

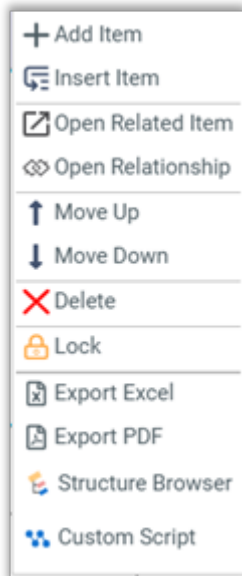
- **Property Filter:** By clicking over the Property Filter check box available below the search, you perform advanced search operations on group containing various options to create custom expressions. For more information regarding Property Filter, you can refer to section **Error! Reference source not found.** of this document.
- **Save:** By clicking over the save action button, save all the changes performed related to that table.

Below are other functionalities which are available for the table:

- **Sort:** Sort the table rows in ascending or descending order with respect to that column.
- **Drag & Drop:** Move and reorder the rows with ease. This feature is only available when the table is in scroll mode and reordering is enabled.
- **Column Filter:** By clicking over the filter icon in the column header, property filter containing various options to create custom expressions shows up automatically selecting property name and default first operator. You can enter filter values on individual columns and select search icon to filter the rows based on RelationshipType or ReferenceType properties. For table with RelationshipType properties, only “AND” logical operator is supported. Once property filter is applied, filter icon turns solid for the column on which filter is applied.
- **Pagination:** Pagination controls are shown at the bottom to navigate to other pages of the table. You can enter Page Size and select the search icon from the top action Toolbar to adjust the number of rows per page.



- **Context Menu:** By right clicking on any of the table rows, context menu shows up. All the Toolbar actions excluding the simple filter actions can also be accessed through the Context Menu.



- **Multi Select:** You can also use CONTROL & SHIFT buttons to select multiple rows based on the requirement.

If Table columns have a fixed width, the following features are enabled:

- **Freeze Columns:** The columns which are frozen will be visible to access at the left of the Table, while you scroll horizontally to other columns.
- **Column Resizing:** Based on requirement, you can increase or decrease the width of the columns. Columns can be resized by dragging the right edge of the column header.
- **Column Re-Ordering:** Drag-drop columns to re-order them based on the requirement.

Worksheet

Worksheet control provides Excel experience for creating datasheet. You use Worksheet control to show relationship data or reference data of the context item as datasheet. When you add worksheet control to the form, properties of Relationship and Related ItemType are mapped to worksheet columns. Columns in the Worksheet are constrained with property data types. It supports nested rows from child relationships. Worksheet control has multiple settings that control its appearance and behavior at runtime.



	Name	Part Number	Total Cost (\$)	(Note Text created_on)
1	Bicycle cover	Cover	1250	Material: Cloth Custom Stitched Multi color option
2	Frame	frame	1875	Frame purchased as welded part. To be painted black at supplier site. supplier to pack it properly.
3	Brake	BRKE	6750	Rim brakes and disc brakes are operated by brake levers are mounted on the handle bars
4	Axle	AXL	7000	a rod that serves to attach a wheel to a bicycle provides support for bearings on which the wheels rotates.
5	Belt Drive	BLT	9000	alternative to chain-drive
6	Chain	CHN	7000	a system of interlocking rings, plates and rollers that

Above the worksheet control, on the left end you will find the label and the Toolbar on the right end. If the label is absent the Toolbar is on the left.



Below are all the actions available in the Toolbar:

- **Add:** Add a new row after the selected row.
- **Insert:** Item selection flyout shows up. You can add rows to the Worksheet from the items which are already available.
- **Open Related Item:** Open a new tab by loading the related item of the selected row.
- **Open Relationship Item:** Open a new tab by loading the relationship item of the selected row.
- **Up:** Move the selected row above the previous row.
- **Down:** Move the selected row below the next row.
- **Delete:** Delete the selected row.
- **Cut:** Cut the selected cell or row.
- **Copy:** Copy the selected cells or rows to the clipboard.
- **Paste:** Paste the selected cells or rows from the clipboard.
- **Export:** Download all the data shown in the worksheet to an excel file.
- **Custom Action Buttons:** Admins can configure custom action buttons for the Worksheet.

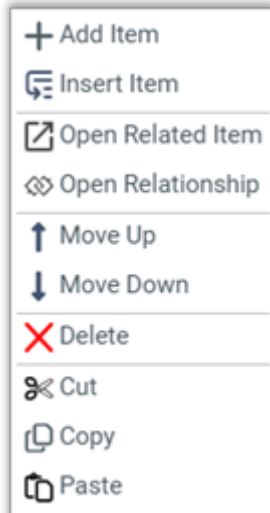
These two actions below are **exclusively** available in the **Toolbar for the Dashboards**:

- **Property Filter:** By clicking over the Property Filter check box available below the search, you perform advanced search operations on group containing various options to create custom expressions.
- **Save:** Save all the changes performed related to that Worksheet.

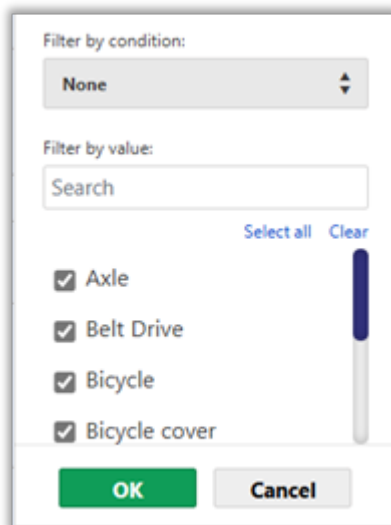


Below are other functionalities which are available for the Worksheet:

- **Context Menu:** By right clicking anywhere on the worksheet, context menu shows up. You access Toolbar actions excluding the Export action through the Context Menu.



- **Sort:** Sort the rows in ascending or descending order with respect to that column.
- **Drag & Drop:** Select a row or multiple rows and use the drag & drop to move and reorder the rows with ease.
- **Multi Select:** Use CONTROL & SHIFT buttons to select multiple rows or cells based on the requirement.
- **Filter:** Available at the column level; access it by clicking on the arrow down icon in the column header to filter the rows with ease.



If the Worksheet columns have a fixed width, the following features are enabled:

- **Freeze Columns:** Frozen columns will be visible to access at the left of the Worksheet, while you scroll horizontally to other columns.
- **Column Resizing:** Increase or decrease the width of the columns. Columns can be resized by dragging the right edge of the column header.
- **Column Re-Ordering:** Drag-drop columns to re-order them based on the requirement.
- **Nested Rows:** Double click to edit nested rows data from the flyout.

	Note Text	created_on
1	Rim brakes and disc brakes are operated by brake levers	7/25/2024 5:03:41 AM
2	are mounted on the handle bars	7/25/2024 5:03:41 AM

TreeTable

TreeTable control shows the structure of items of the same ItemType. At the time of adding tree-table control to the form, properties of Relationship and Related Item Type are mapped to tree-table columns. TreeTable control has multiple settings that control its appearance and behavior at runtime.

Name	Part Number	Cost(\$)	Max Order Quantity	Total Cost(\$)	Created On	Author
Wheels	wheels	20.0000	40	800	7/25/2024 2:05:07 AM	Innovator Admin
Spokes	spokes	20.0000	60	1,200	7/25/2024 3:36:57 AM	Innovator Admin
Tyre	tyre	20.0000	20	400	7/25/2024 3:36:57 AM	Innovator Admin
Tyre Valve	tyre_valve	20.0000	20	400	7/25/2024 3:48:51 AM	Innovator Admin
Rim	rim	20.0000			7/25/2024 3:45:54 AM	Innovator Admin
Hub	hub	20.0000			7/25/2024 3:32:16 AM	Innovator Admin
Frame	frame	20.0000			7/25/2024 2:05:08 AM	Innovator Admin
Seat	seat	20.0000			7/25/2024 2:55:13 AM	Innovator Admin
Handle	handle	20.0000			7/25/2024 3:21:14 AM	Innovator Admin
Main Frame	main_frame	20.0000			7/25/2024 3:32:16 AM	Innovator Admin

Expand Level 1

Above the TreeTable control, on the left you will find the label and the Toolbar on the right. If the label is absent the Toolbar is on the left.



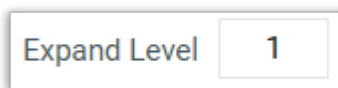


Below are all the actions available in the Toolbar:

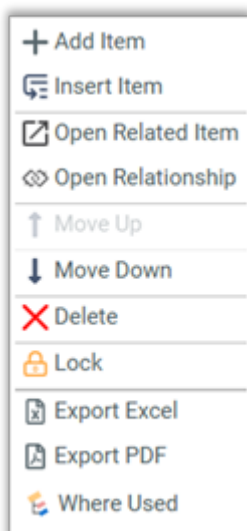
- **Add:** Add an item as a child or a sibling of the selected node.
- **Insert:** Item selection flyout shows up. Add rows to the TreeTable from the items which are already available.
- **Open Related Item:** Open a new tab by loading the related item of the selected row.
- **Open Relationship Item:** Open a new tab by loading the relationship item of the selected row.
- **Up:** Move the selected row above the previous row. Up action will be enabled only when the parent item is locked.
- **Down:** Move the selected row below the next row. Down action will be enabled only when the parent item is locked.
- **Delete:** Delete the selected row from the TreeTable. Delete action will be enabled only when the item is locked.
- **Lock:** Lock the selected row for exclusive edit. If the associated item with the row is locked by someone else, you won't be able to lock the row.
- **Export Excel:** Download all the visible data shown in the TreeTable to an excel file.
- **Export PDF:** Download all the visible data shown in the TreeTable to a PDF file.
- **Custom Action Buttons:** Admins can configure custom action buttons for the TreeTable.

Below are other functionalities which are available for the TreeTable:

- **Expand Level:** When you expand the node, it only gets its first level children. If you want to expand to multiple levels, the Expand Level value can be set. Expand Level is available at the bottom left of the TreeTable.



- **Context Menu:** Right click on the rows of TreeTable to see Context Menu; access Toolbar actions.



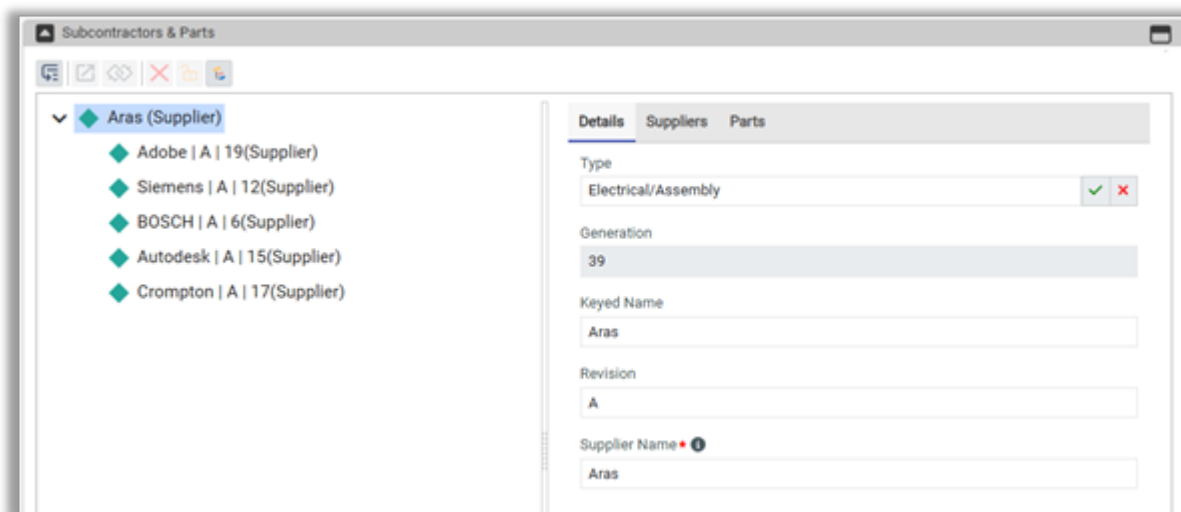
If the TreeTable columns have a fixed width, the following features are enabled:

- **Freeze Columns:** Frozen columns will be visible to access at the left of the Worksheet, while you scroll horizontally to other columns.
- **Column Resizing:** Increase or decrease the width of the columns. Columns can be resized by dragging the right edge of the column header.

Structure

Use Structure control to show any structure with items of different ItemTypes and Relationship Types:

- It is defined on Query Definition which requires context item Id. Use it as StructureBrowser and WhereUsed controls for a given ItemType by defining appropriate Query Definition.
- In edit mode, you can modify and save fields returned from the Query Definition back to Innovator.
- In edit mode, you can insert children of any allowed type at any level in the structure and save back to Innovator.
- You can configure Structure control to show relationships in the structure as nodes in the tree or rows in the table based on the configuration.



Above the Structure control, at the left you will find the label and the Toolbar on the right. If the label is absent the Toolbar is on the left.



Below are all the actions available in the Toolbar:

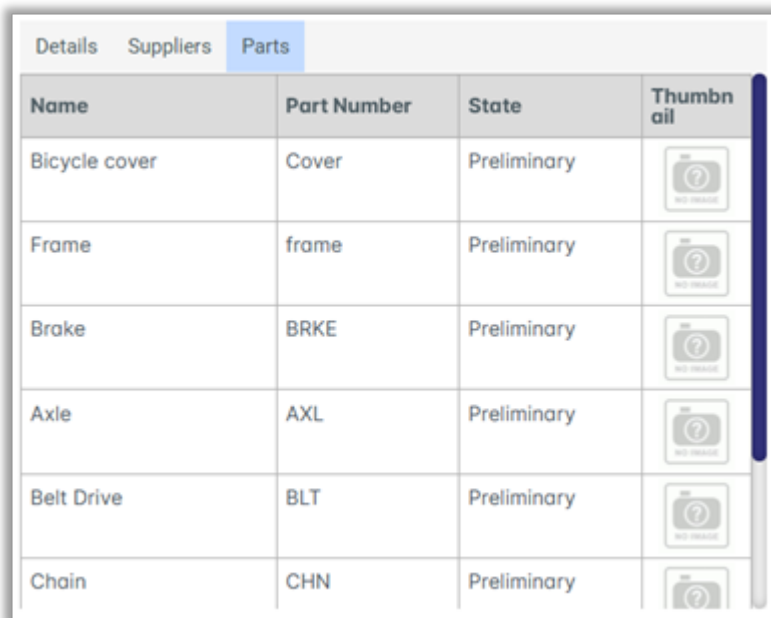
- **Insert:** Item selection flyout shows up. Add rows to the Structure from the items which are already available.



- **Open Related Item:** Open a new tab will be opened by loading the related item of the selected row.
- **Open Relationship Item:** Open a new tab will be opened by loading the relationship item of the selected row.
- **Delete:** Delete the selected node. Delete action will be enabled only when the item is locked.
- **Lock:** Lock the selected node for exclusive edit. If someone locks the associated item of the node, you won't be able to lock the row.
- **Custom Action Buttons:** Admins can configure custom action buttons for the Structure.

Other functionalities which are available for the TreeTable:

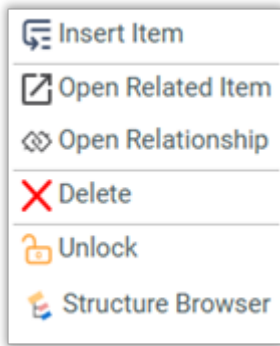
- **Separator:** Structure control is divided into two sections by the Separator. On the left, you have a Tree View of nodes and on the right there is a Details view of the selected node. By dragging the Separator, you can adjust width on both sections.
- **Tree View:** A classic tree view, with the current item as root node and all its relationships, references & related items configured are shown as its child nodes.



Name	Part Number	State	Thumbnail
Bicycle cover	Cover	Preliminary	
Frame	frame	Preliminary	
Brake	BRKE	Preliminary	
Axle	AXL	Preliminary	
Belt Drive	BLT	Preliminary	
Chain	CHN	Preliminary	

- **Details View:** Shows all the tabs configured. Details tab shows all the configured properties of the selected node item. Other tabs show relationships, references and related ItemTypes ("*" next to the tab name).
- **Sort:** Sort rows in ascending & descending order.
- **Context Menu:** Right click on the nodes of Structure to see Context Menu; access Toolbar actions.





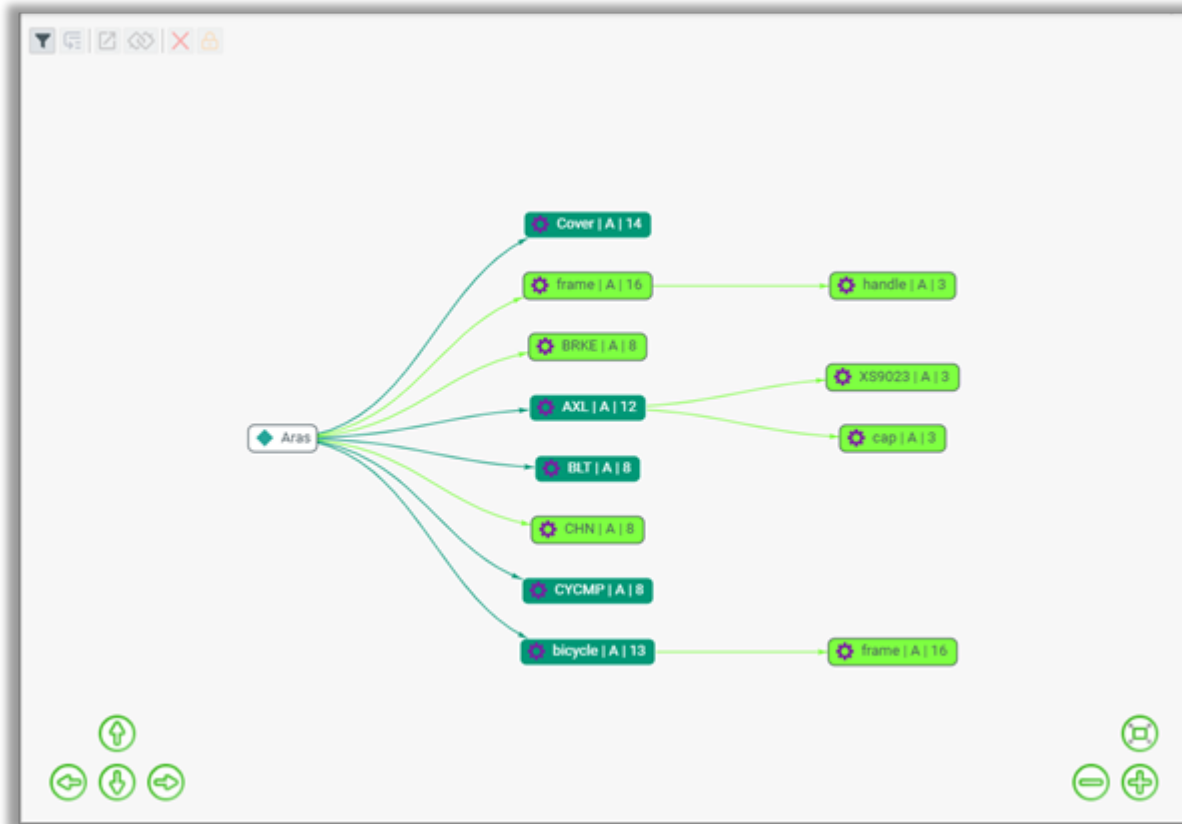
Graph

Use Graph control to show structure data of the context item in the graphical structure. Graph control is bound to a Query Definition which defines which Relationship Types, Related Item Types and properties to show in the control. By double clicking any node in the Graph, a flyout will show with the selected item properties along with its relationships. Each node in the graph will be shown with different color based on the background color set at RelationshipType or conditional formatting defined for the RelationshipType in the template definition.

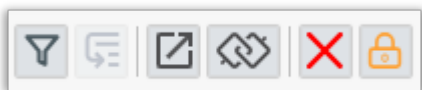
Graph control comes with two modes:

- **read-only** mode shows structure and properties, you can only navigate the structure and see properties of the selected node. You can also open the specific node in its own tab by selecting Open icon from the toolbar.
- in the **edit mode**, you can insert new nodes in the structure by selecting the Insert icon from the toolbar. In the InsertNode Flyout, allowed RelationshipTypes of the selected node are shown, by selecting specific RelationshipType and keyword. You can also edit properties or delete selected node.



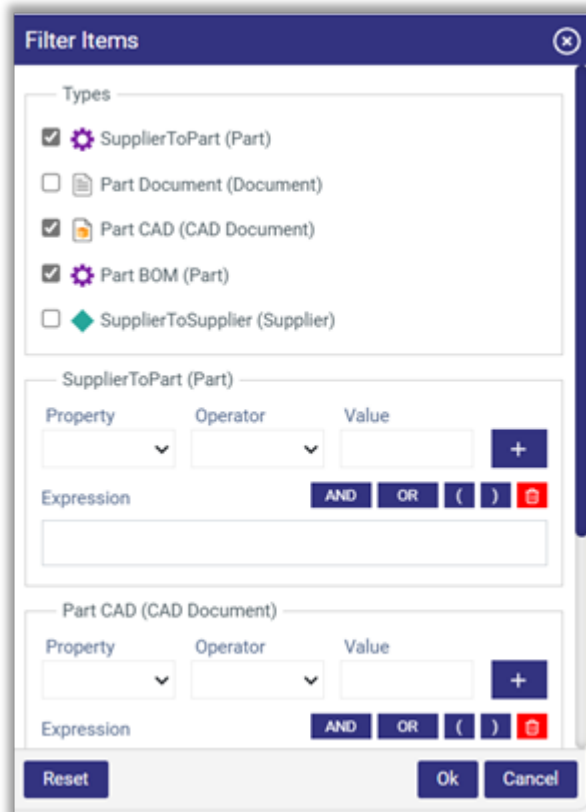


Above the Graph control, at the left you will find the label and the Toolbar on the right. If admin hides the label, the Toolbar is on the left.



Below are all the actions available in the Toolbar.

- **Item Filter:** A filter flyout shows up from which you can filter the items in the graph view. All the ItemTypes shown in the graph view are in the first group and for each selected ItemType, property filter is shown in its individual group.



- Filter flyouts layout will be as follows:
 - **Reset:** Reset all the filter selections to default.
 - **Ok:** Apply selection filters over the Graph and close the flyout
 - **Cancel:** Close the flyout.
- **Insert:** An item selection flyout appears. You can add nodes to the Graph from the items which are already available. All functionality and features are similar to those in the Table.
- **Open Related Item:** Open a new tab by loading the related item of the selected node.
- **Open Relationship Item:** Open a new tab by loading the relationship item of the selected node.
- **Delete:** Delete the selected node. Delete action will be enabled only when the item is locked.
- **Lock:** Lock the selected node for exclusive edit. If someone locks the associated item of the node, you will not be able to lock the row.
- **Custom Action Buttons:** Admins can configure custom action buttons for the Graph.

Other functionalities of the Graph are as following:

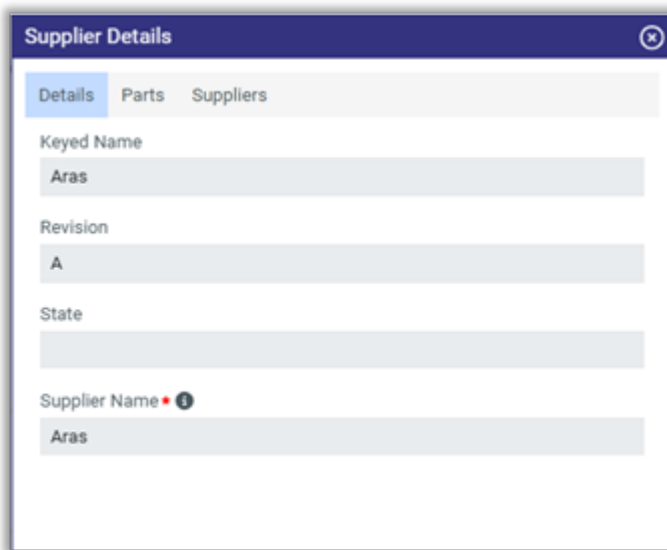
- **Graph Buttons:**
There are arrow buttons available at the bottom left of the graph that are used to move the graph in their respective directions.



There are three other buttons available at the bottom right of the graph, those are Zoom In, Zoom Out & Fit to Size.



- **Node Details Flyout:** By double clicking on the node of the graph, Node Details Flyout shows up. Shows all the tabs configured. Details tab shows all the configured properties of the selected node item. Other tabs show relationships, references and related ItemTypes ("*" next to the tab name).

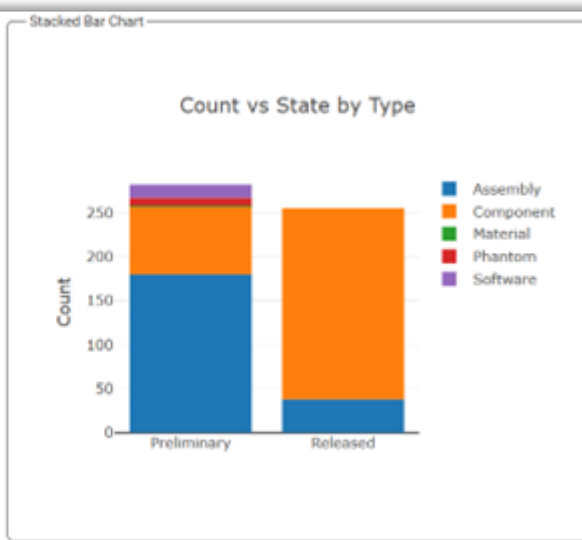
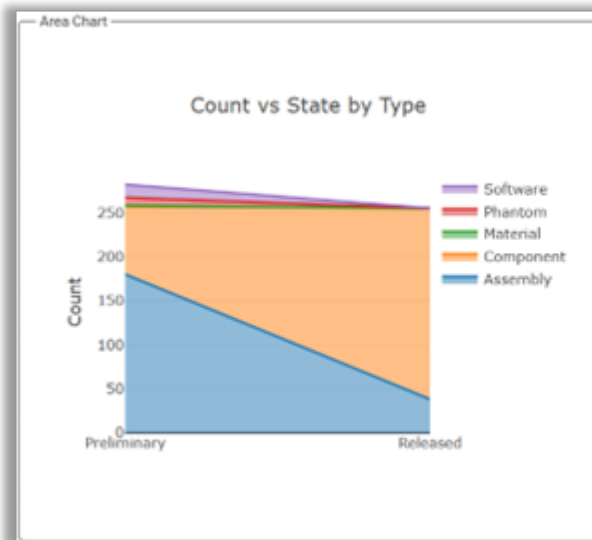


- **Sort:** Sort the rows in ascending and descending order.
- **Ok:** Apply all changes made in the flyout to the Graph and close the flyout.
- **Cancel:** Close the flyout.

Report

Use Report control to generate reports from the available data. Data is fetched for a report control based on the Query Definition selected for the control. Report is generated based on the data associated with the context item. It also can be generated based on global data of any other ItemType. Report control shows the data in the form of PivotTable, Line Chart, Bar Chart, Area Chart TreeTable, PowerBI. Report control is displayed on Item Details and Dashboard pages.

	State			
Type		Preliminary	Released	Totals
Assembly		182	36	218
Component		97	197	294
Material		2		2
Phantom		9		9
Software		14		14
Totals		304	233	537

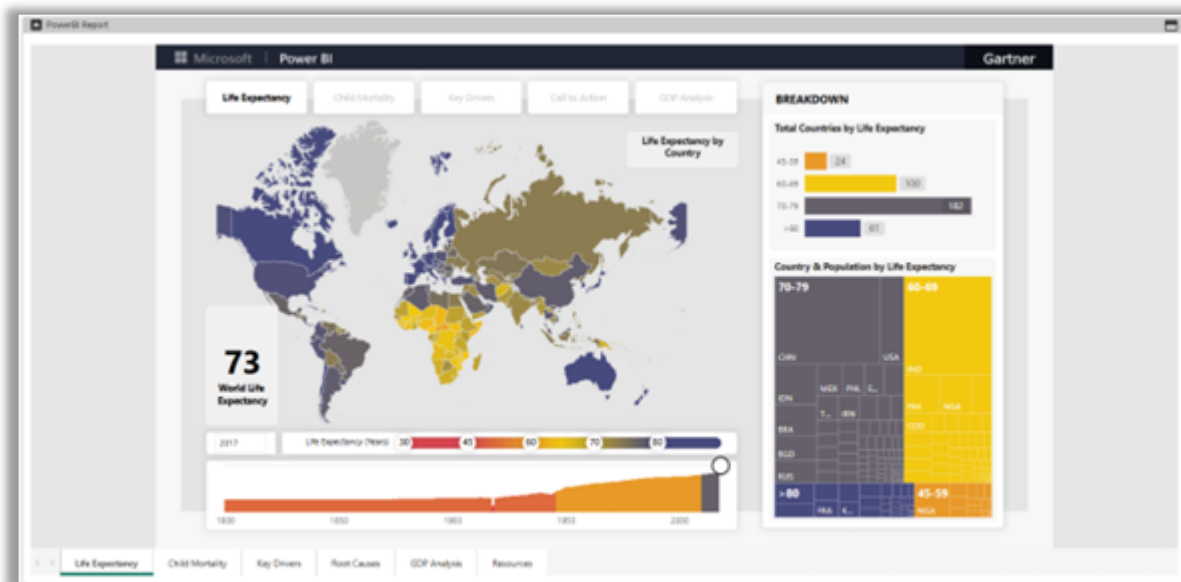


Default Wizard

12/24/2024 11:51:24 AM v

Keyed Name	Quantity	Value (%)	Unit of Measure	REACH Compliance	RoHS Compliance
✖ Cigarette (Part)				Non-Compliant	Compliant
✖ Cig Blended Tobacco (Part)	1			Compliant	Compliant
✖ Cig Tobacco (Part)	1			Compliant	Compliant
> Surety Tobacco (Part)					
✖ Oriental(Turkish) Tobacco (Part)	1				
✖ Dark Tobacco (Part)	1				
✖ Cadmium (Substance)		0.01	Gram	<= 1.5 (Compliant)	<= 3.5 (Compliant)
✖ Copper (Cu) (Substance)		0.02	Gram	<= 0.5 (Compliant)	<= 3.5 (Compliant)
✖ Cadmium (Substance)		0	Gram	<= 1.5 (Compliant)	<= 3.5 (Compliant)
✖ Copper (Cu) (Substance)		0.01	Gram	<= 0.5 (Compliant)	<= 3.5 (Compliant)
> Cig Foil (Part)	1			Compliant	Compliant
> Cig Filter (Part)	1				
> Cig Flavors (Part)	1				

< Previous Next >



Drilldown Report is displayed by double-clicking on the row or column data for each subcategory. Individual item data can be opened by clicking hyperlink of the item from the Flyout.



Drilldown Report ✕

Axis (Categories) Legend (Series)

State : Released Type : Component

Name	Part Number	State	Type	Make / Buy
Non-Threaded 63mm Bottom Bracket	BB-9090	Released	Component	Buy
Threaded 63mm Bottom Bracket	BB-9091	Released	Component	Buy
Non-Threaded 73mm Bottom Bracket	BB-9092	Released	Component	Buy
Threaded 73mm Bottom Bracket	BB-9093	Released	Component	Buy
Bell	BC-BELL	Released	Component	Make
Bell 2	BC-BELL-2	Released	Component	Make
4mm Steel Allen Bolt	BLT-0001	Released	Component	Buy
17.3" Aluminum Drop Bar	HB-1623	Released	Component	Buy
16.5" Aluminum Drop Bar	HB-2303	Released	Component	Buy

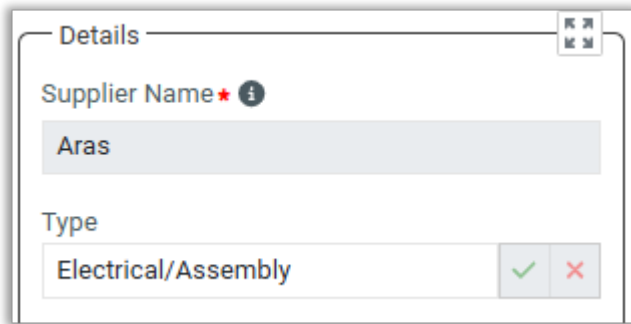
Cancel



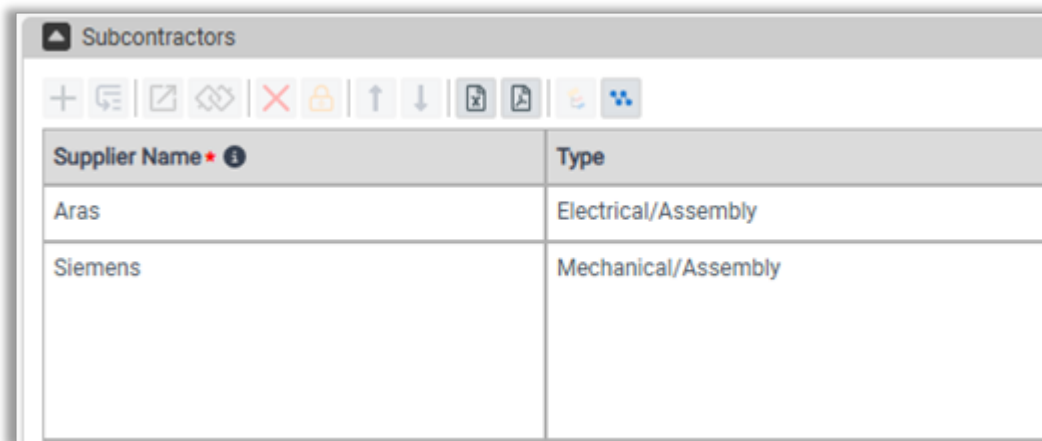
Page & Control Features

Mandatory Properties

Mandatory properties on the form and collection controls are displayed with '*' after property label.



The screenshot shows a 'Details' form with two fields. The first field is 'Supplier Name', which is mandatory (indicated by a red asterisk and an information icon) and contains the text 'Aras'. The second field is 'Type', which contains the text 'Electrical/Assembly' and has a green checkmark and a red 'X' button next to it.



The screenshot shows a 'Subcontractors' collection control with a toolbar and a table. The table has two columns: 'Supplier Name' (mandatory) and 'Type'. The first row contains 'Aras' and 'Electrical/Assembly'. The second row contains 'Siemens' and 'Mechanical/Assembly'.

Supplier Name *	Type
Aras	Electrical/Assembly
Siemens	Mechanical/Assembly

Tooltip & Help Text

Tooltip is displayed when you hover the mouse on the information icon on the form field labels or collection control column headers if admin has setup the tooltip on the ItemType property definition.

The screenshot shows a 'Supplier Details' form with the following fields:

- Supplier Name:** A text input field containing 'Aras'. A tooltip is displayed over the information icon (i) next to the label, stating 'Supplier Name means the name of the entity'.
- Type:** A dropdown menu showing 'Electrical/Assembly' with a green checkmark and a red 'X' button.
- On Board Date:** A date input field showing '12/7/2023' with a calendar icon.

The screenshot shows a 'Subcontractors' table with a toolbar at the top containing various icons (add, edit, delete, etc.). The table has two columns: 'Supplier Name' and 'Type'. A tooltip is displayed over the information icon (i) next to the 'Supplier Name' column header, stating 'Supplier Name means the name of the entity'.

Supplier Name	Type
Aras	Electrical/Assembly
Siemens	Mechanical/Assembly

Help Text flyout will be shown when you click on the information icon on the form field labels or collection control column headers if admin has setup the help text on the ItemType property definition.

Supplier Details

Details

Supplier Name • ⓘ

Aras

Type

Electrical/Assembly ✓ ✕

Quantity & Weight

Max Order Quantity

2

Weight(kg)

500

Supplier Name ⓘ

Supplier Name means the name of the entity providing the product or service that is part of the contract requirements. This could include any entity involved in producing products or services that are part of the contract.

Subcontractors

+ - ↺ ↻ ✕ 🔒 ↑ ↓ 📄 📄 🔍

Supplier Name • ⓘ	Type	Weight(kg)	Max Order Quantity	Total Weight(kg)
Aras	Electrical/Assembly	32	15	480.00
Siemens	Mechanical/Assembly	50	99	4,950.00

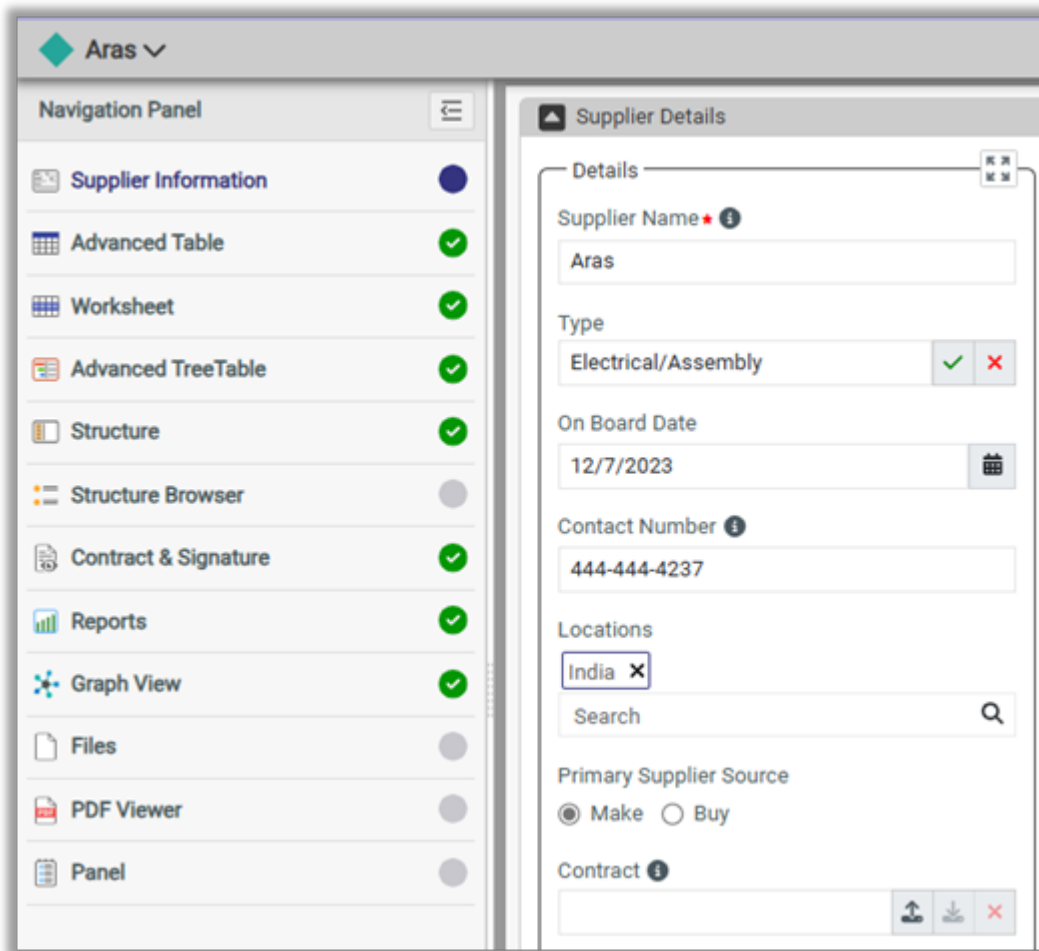
Supplier Name ⓘ

Supplier Name means the name of the entity providing the product or service that is part of the contract requirements. This could include any entity involved in producing products or services that are part of the contract.



Display Conditions

Display Conditions on a control are defined with expression to dynamically render or hide controls based on property values. Display conditions defined on the control will decide whether control should be shown or hidden from the page. Display conditions can be defined on the pages as well as all control types. In the following example, 'Contract & Signature' page is visible only when 'Show Subcontractors' is selected.

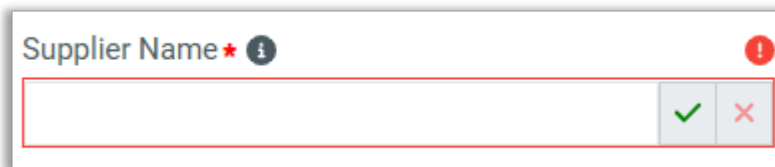


Validation Rules

Validation Rules are defined on a control to validate entered data before submitted request to server:

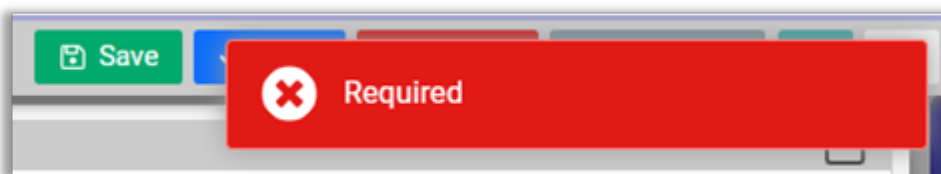
- Rules can be built based on the expression from the properties of the current and previous pages. As part of moving away from the page, all validation rules on the page are evaluated.
- If any of the rules are not satisfied, it shows popup dialog with Validations Failed message and icons against each control.
- Hovering the mouse on the icon to display specific errors or warning messages based on the defined validation rules on the control.
- You can move away from the current page by clicking any other page from the Navigation Panel.
- If the current page has any validation errors, a popup dialog will display with options to ignore and move forward or fix the issues before moving forward.
- If multiple rules are defined on the controls, all rules should be satisfied in order to move forward.

Validation can be of **Error** or **Warning** type. When validation error rule fails, error will be displayed in the popup dialog with options to ignore and move forward or fix the issues before moving forward. If you move forward without fixing validation errors for that page, in the navigation panel, page title with be shown with red-circle with tick icon to indicate there are validation errors in the page.

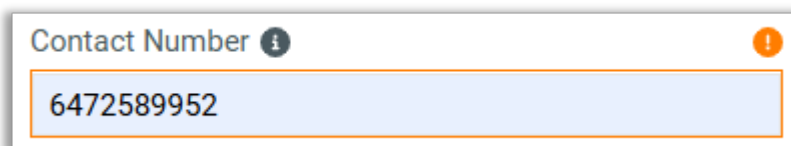


Supplier Name * ⓘ

✓ ✗

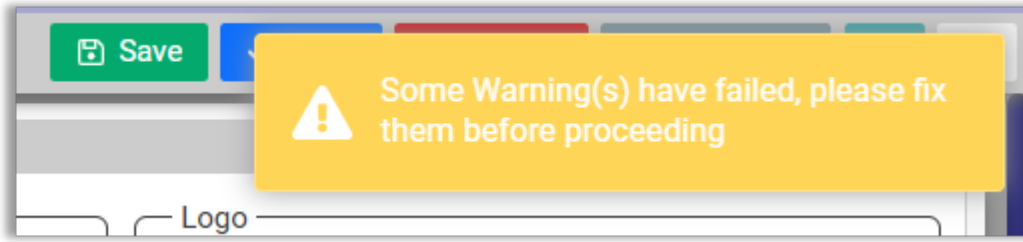


When validation warning rule fails, warning will be shown in the flash notification with orange background color. Warnings can be ignored, by clicking on Save, Done, Next buttons second time to move forward with the warning



Contact Number ⓘ

6472589952



Computed Values

Computed values are defined on the Text control to show calculated value when control is rendered. Computed value can be calculated based on the expression, expression can refer other properties from the current page or previous pages. If Text control is defined with computed value and it is marked as read-only, it can dynamically change its value based on the values entered in other controls on the page. In the following example, 'Total Weight' is calculated based on the values set for 'Max Order Quantity' and 'Weight' at real-time.

Supplier Name	Type	Weight (kg)	Max Order Quantity	Total Weight (kg)
Aras	Electrical/Assembly	32	15	480.00
Autodesk	Electrical/Assembly	100	16	1,600.00
Siemens	Mechanical/Assembly	50	99	4,950.00

Max Order Quantity

Weight (kg)

Total Weight (kg)



Conditional Formatting

Conditional Formatting can be defined on Text control and Table control column to highlight the fields or cells with defined styles. Rules are built with expression based on thresholds set on property value. In the following example, Conditional Formatting is set on 'Total Weight' to highlight control with red color if the value is greater than 2000.

Max Order Quantity
3
Weight (kg)
1,000
Total Weight (kg)
3,000.00



Dynamic ReadOnly

Dynamic ReadOnly allows any control to be marked as read-only at runtime even if you can edit the property associated with that control with lock. Dynamic read-only is a Boolean expression defined using properties from the current and previous pages. At runtime, Dynamic expression will be evaluated and control will be marked as read-only if the value is true.



Reset Dependencies

Reset dependencies can be defined on a control, to reset its value when any of the controls defined in reset dependencies have been changed at runtime.

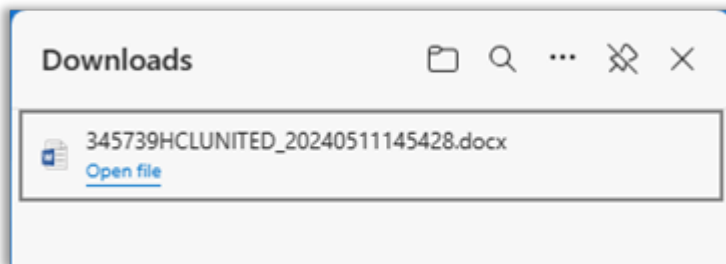
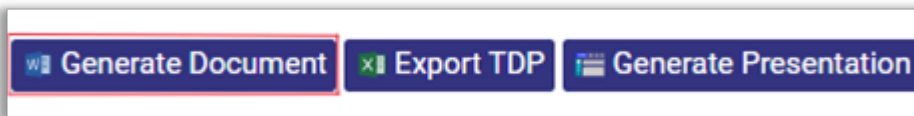


Data Export

Word/PDF Generation

Automates creation of MS Word and PDF documents based on properties from the item:

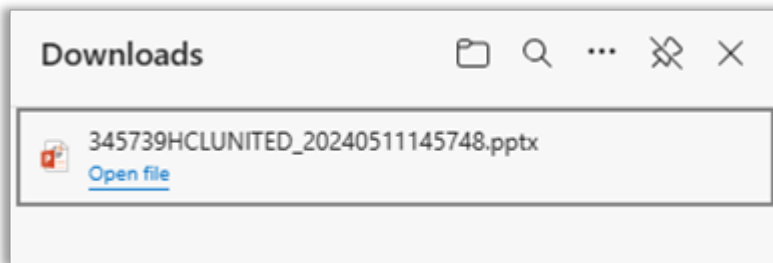
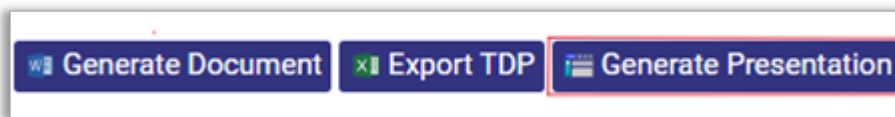
- Word document is created based on the Word template defined in the system.
- Word template has content controls mapped to properties of ItemTypes and RelationshipTypes.
- Can be defined with chart, data-bound with Query Definition to dynamically generate chart with data in the generated document.
- Document can be composed from the documents stored in Innovator.
- Can have a repeating content controls with DataSource generated from relationships collection or ServerMethod output.
- Document generation with watermarking and PDF can be configured based on lifecycle states.



PowerPoint Generation

Automates creation of MS PowerPoint document based on the properties from the item:

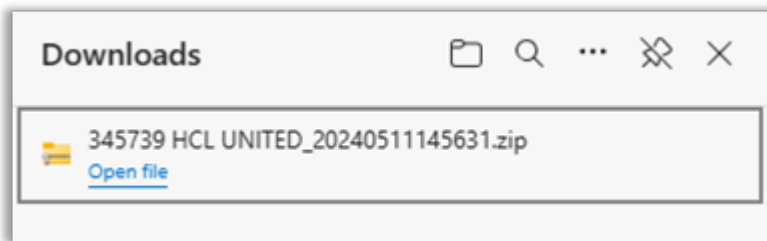
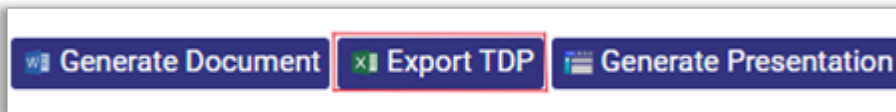
- PowerPoint document is created based on the PowerPoint template defined in the system.
- PowerPoint template has content controls mapped to properties of ItemTypes and Relationship Types.
- Can be defined with chart, data-bound with Query Definition to dynamically generate chart with data in the generated document.
- Can be composed from the PowerPoint documents stored in Innovator.
- Template can have table and dynamic list controls with DataSource generated from relationships collection or ServerMethod output.



Technical Data Package

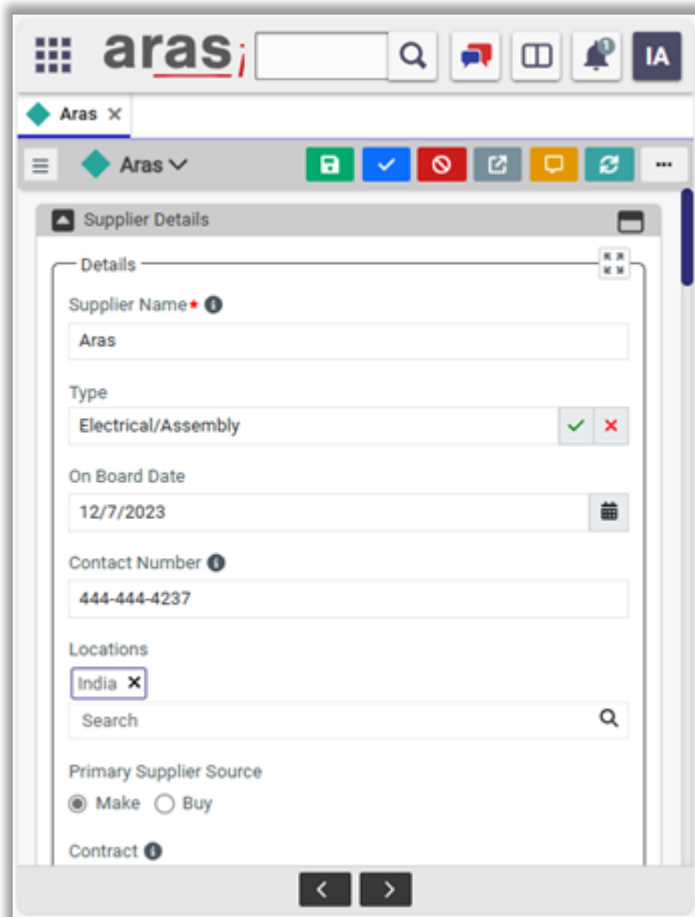
TDP provides offline access to outside suppliers for specific data and files:

- TDP contains items data, relationships data and files bundled in a zip file.
- TDP is created based on configured excel template with multiple worksheets, having columns with properties from ItemTypes, Relationship Types, and Files.
- Package is generated on the server based on user action, lifecycle promotion, or scheduled job.
- Generated package has an excel file with all items and relationships data along with the files.



Responsive UI & Mobile Views

Controls rendered on the page are adjusted to the available page width to avoid showing horizontal scrollbars. In the mobile device, all the pages are shown with a mobile layout.



The screenshot displays the Aras ProApp Designer interface. At the top, there is a header with the Aras logo, a search bar, and several utility icons. Below the header, a navigation bar shows the current application context 'Aras' and a set of action icons. The main workspace is divided into two panels. The top panel, titled 'Pivot Table', contains a data table with the following structure:

	State	Preliminary	Released	Totals
Type				
Assembly		203	36	239
Component		106	195	301
Material		3		3
Phantom		9		9
Software		14		14
Totals		335	231	566

The bottom panel, titled 'Charts', contains an 'Area Chart' with the title 'Count vs State by Type'. The chart area is currently blank, with only the title text visible. At the bottom of the interface, there are navigation arrows.



Aras Presentation Generator - User Guide



Introduction

Purpose

This User Guide describes how to use the Aras Presentation Generator for external users.



Scope

This document provides instructions for configurations and functionality of the Aras Presentation Generator application.



Target Audience

This document is intended for external users with limited access to data.



Feature Terms and Definitions

The following Feature terms and definitions are used in this document:

Term	Definition



Guide Organization

This document describes how to use the Aras Presentation Generator application. This User Guide is organized into the following sections:

SECTION 1:	This section defines the purpose and scope of the User Guide and identifies its intended audience. It also provides a concise summary of each part to assist users in navigating the guide effectively.
Introduction	
SECTION 2:	This section explains the purpose of the Presentation Generator and provides an overview of the features available in the Aras Presentation Generator. It also includes a description of the template creation process.
Introduction to	
Presentation Generator	



Introduction to Presentation Generator

The Presentation Generator automates the creation of presentations based on user selections made through wizard pages. Presentations are generated using a configured template and do not require any plugins or Office connectors. Generation occurs on the server and can be triggered by user actions, lifecycle promotions, or scheduled jobs.

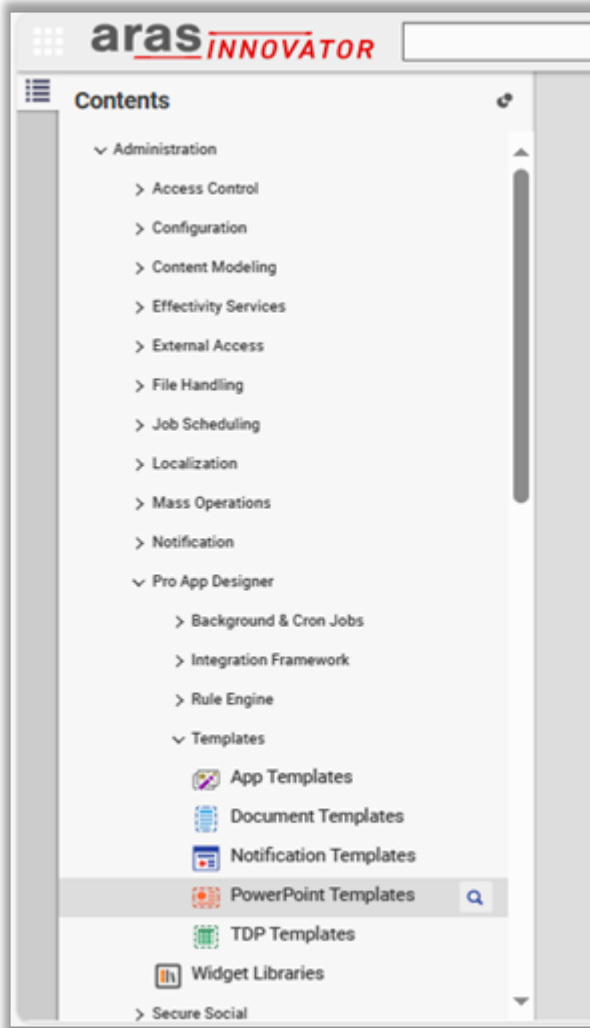
Templates include content controls mapped to properties of ItemTypes and RelationshipTypes. Presentations can also be composed from existing presentations stored in Innovator. Templates may contain repeating content controls, which are linked to data sources derived from relationship collections or ServerMethod outputs. The appropriate template for generation is selected based on predefined rules.

Presentation generation can be initiated by any user action or system event. The process utilizes a template associated with the relevant ItemType.



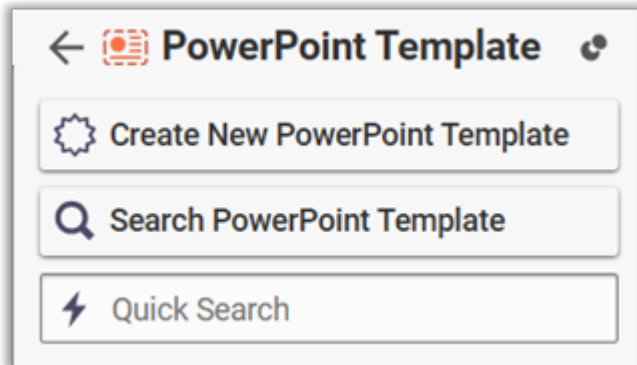
Presentation Generator Template Configurations

1. Go to TOC → Administration → Pro App Designer → Templates → PowerPoint Templates.

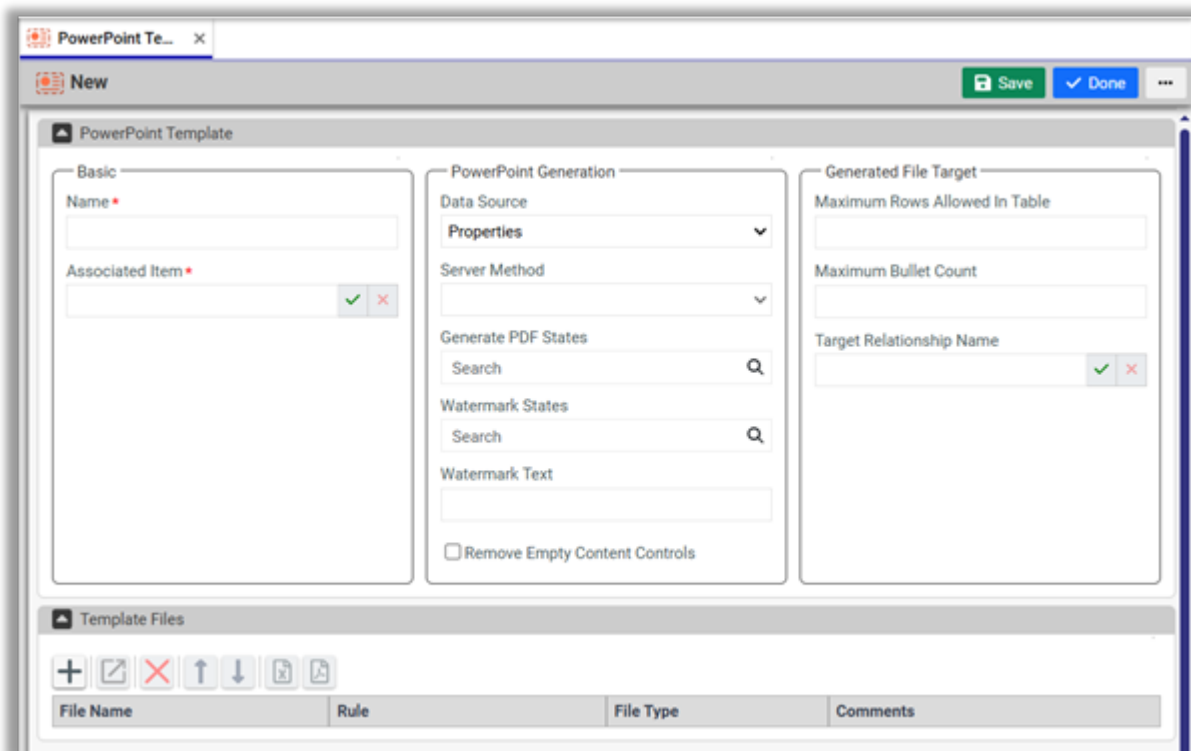


2. Click on Create New PowerPoint template.





3. Enter the necessary information, save your work, and add the template file accordingly.



Following are the details for PowerPoint Template properties:

Basic:

- **Name:** The name of the PowerPoint template is a required property and must be specified in order to save the template.
- **Associated Item:** The context item for which the presentation template is being created is a required property and must be specified to save the template.



PowerPoint Generation:

- **Data Source:** There are two options available for selection under Data Source. The first option is 'Properties' (selected by default), and the second option is 'Server Method'.
- **Server Method:** Server method selection is available when 'Server Method' is chosen as the data source. All methods configured as Server Events on the associated item will be listed for selection.
- **Generate PDF States:** A list of states for which the PDF will be generated must be specified.
- **Watermark States:** Specifies the list of states during which watermarks should be applied when generating the PDF.
- **Watermark Text:** Specifies the text to be used for watermarking. This attribute is applicable only when the 'Watermark States' attribute is defined.
- **Remove Empty Content Control:** Determines whether empty content controls should be removed from the generated PDF. This setting is applied during the presentation generation process.

Generated File Target:

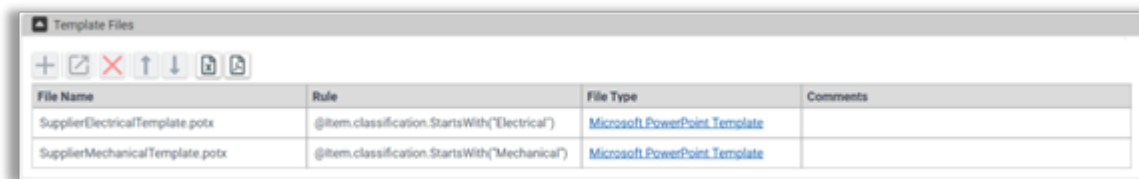
- **Maximum Rows Allowed in Table:** Specifies the maximum number of table rows that should be displayed on a single slide.
- **Maximum Bullet Count:** Specifies the maximum number of bullet points that should be displayed on a single slide.
- **Target Relationship Name:** Specifies the name of the relationship to which the files will be attached after processing.



Template Selection

Each Presentation Generator template can have multiple Presentation files attached. During the Presentation creation process, the appropriate PowerPoint document is automatically selected based on predefined rules:

- If a presentation file does not have any rule defined, it will be selected by default. In cases where multiple files lack rules, the first available presentation file will be chosen automatically.
- If all presentation files have rules defined, the first file that successfully passes validation will be selected. If none of the files meet the validation criteria, the presentation generation process will be aborted.



File Name	Rule	File Type	Comments
SupplierElectricalTemplate.potx	@item.classification.StartsWith("Electrical")	Microsoft PowerPoint Template	
SupplierMechanicalTemplate.potx	@item.classification.StartsWith("Mechanical")	Microsoft PowerPoint Template	

Presentation Template Creation

Presentation generation application uses following content controls:

Plain text Control

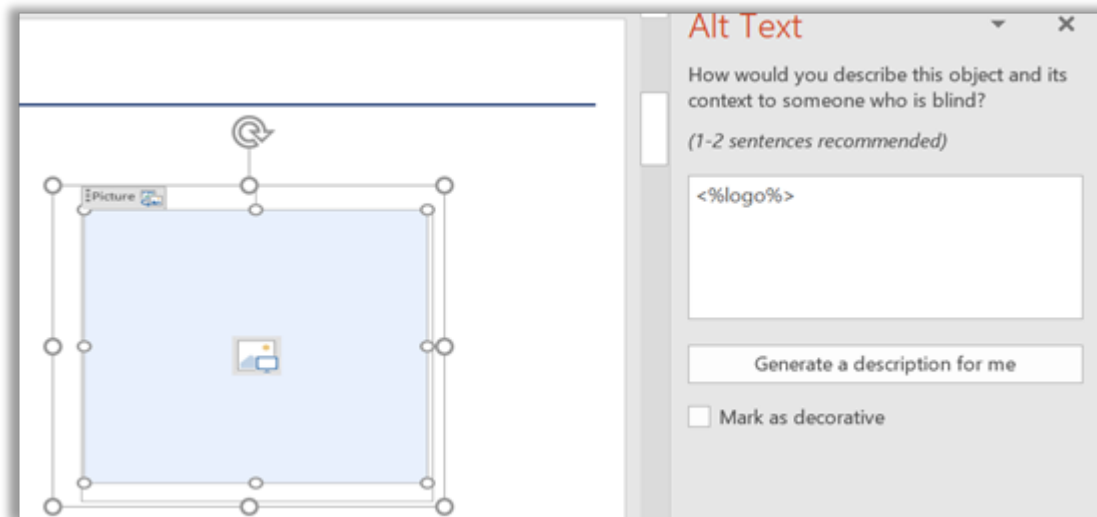
This is used to display plain text.

Examples:

- **Contact Number:** <%contact_number%>
- **Supplier Products:** <%SupplierToProduct%>
- **Locations:** <%locations%>

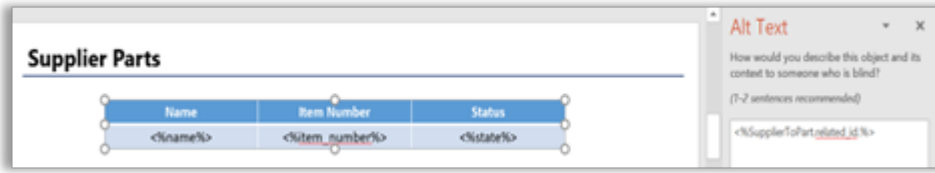
Image

The 'Alt Text' property of an image object is used to define the mapping to an Innovator Item property. This mapping enables the system to retrieve the image from the specified property during presentation generation.



Repeating Text Content: Tabular Presentation

Repeating content controls are used to populate tables in the presentation. The 'Alt Text' property of the table object defines the related path, which is used to retrieve related items. Each column in the table is then mapped to a property of the related item.



Repeating Text Content: Per Slide

The 'Alt Text' property of the slide title is used to define the path to related items. Properties of these related items can be referenced within the content of the slide. Slides will be repeated dynamically for each related item retrieved.

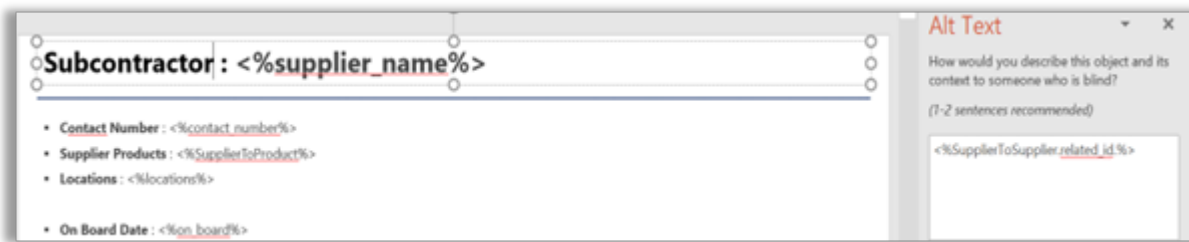
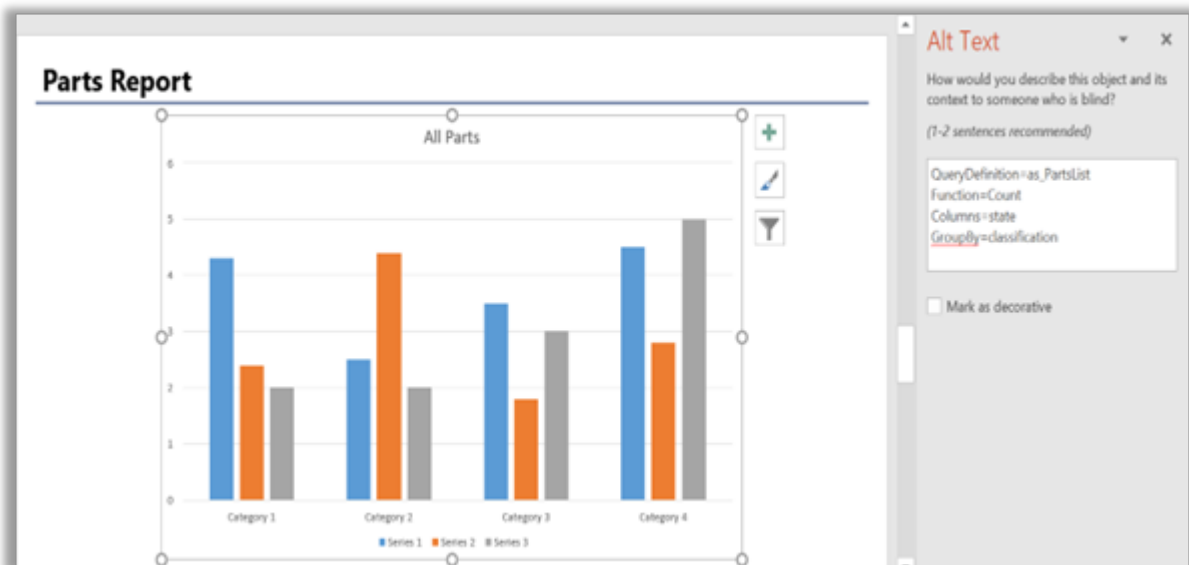


Chart Control

Query definitions are used to retrieve data for chart controls. All chart configurations are specified within the 'Alt Text' property of the chart object. Please refer to the image below.



Tag Naming Conventions and its Meaning

In the above image tag value is given as `<%supplier_name%>`. Let us see what it means

1. `<%..%>` Indicates, processing at server end. Please note that it is mandatory to follow this convention.
2. `<%supplier_name%>` In this context, 'supplier_name' refers to the name of a property within the Supplier ItemType. By using this tag, the application identifies the corresponding property value in Aras Innovator and substitutes it into the presentation at the designated placeholder.
3. `<%SupplierToPart.related_id.name%>` Here involves relationship.
 1. 'SupplierToPart': is the RelationshipType of Supplier Item Type.
 2. 'related_id' is the property pointing to Part Item Type.
 3. 'name': is the property of Part Item Type.

So, this tag value actually fetches the value of Property 'name' attached to 'Part' item type.

Using this naming convention, the server follows a specific sequence when processing the tag:

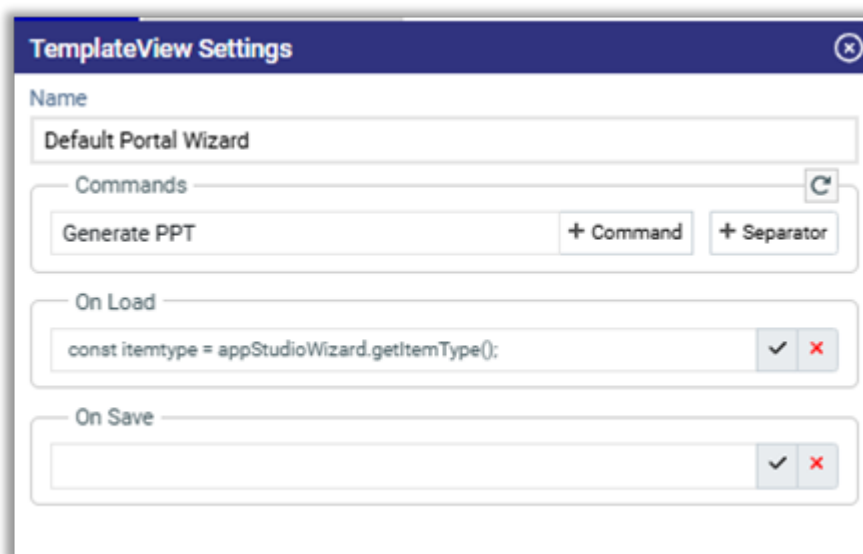
1. Go to 'SupplierToPart'.
2. Search for related_id and get the Data Source value which is 'Part' in this case.
3. Go to Part and access the 'name' property value.



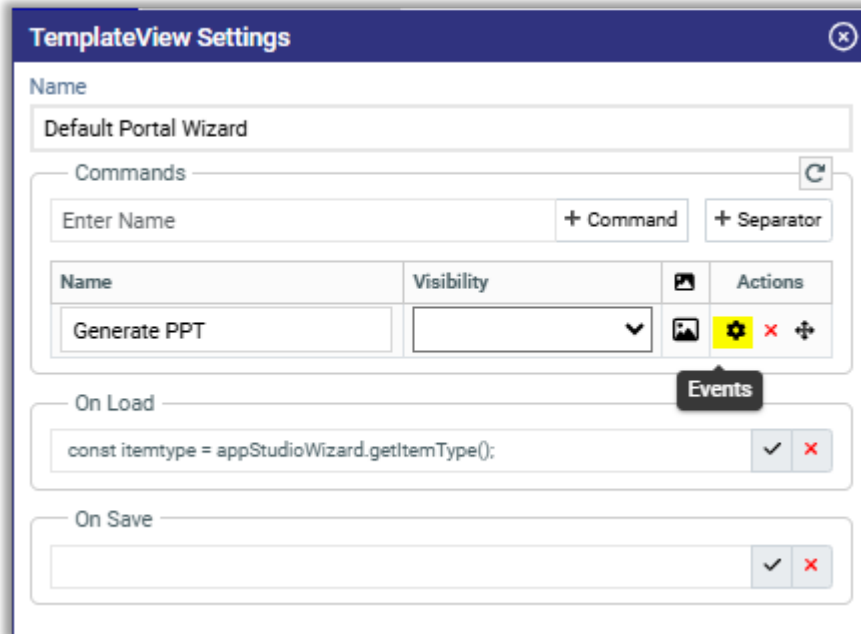
Configure Presentation Generator Command on Template View

To enable presentation generation using this generator, it can be configured as a command on the ItemType template by following these steps:

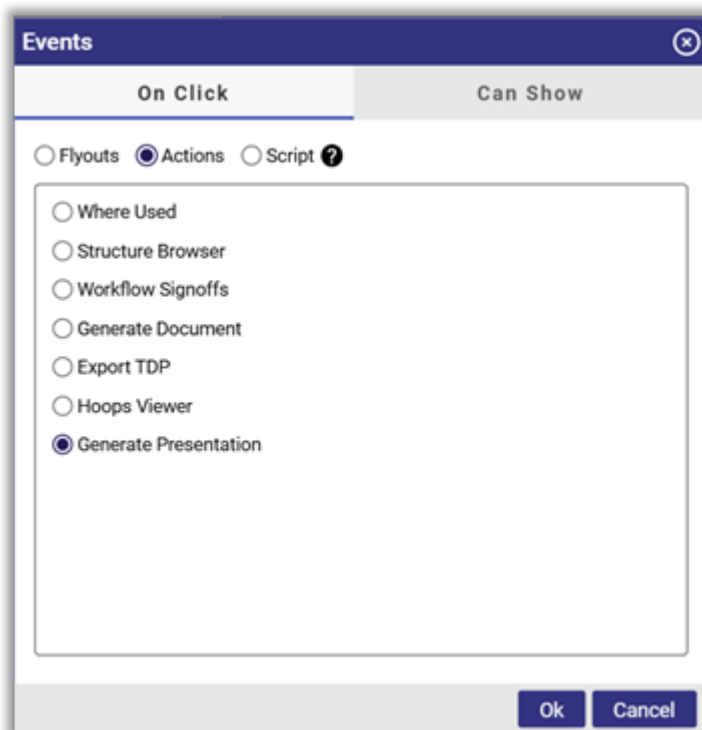
1. Go to TOC → Administration → Pro App Designer → Templates → App Templates → Open App Template and then Open Template View
2. Go to More → Settings → Add Command name (like below image) → Click on Command button.



3. Click on Events (highlighted in below image).



4. Select Actions and then Select Generate Presentation, click Ok.



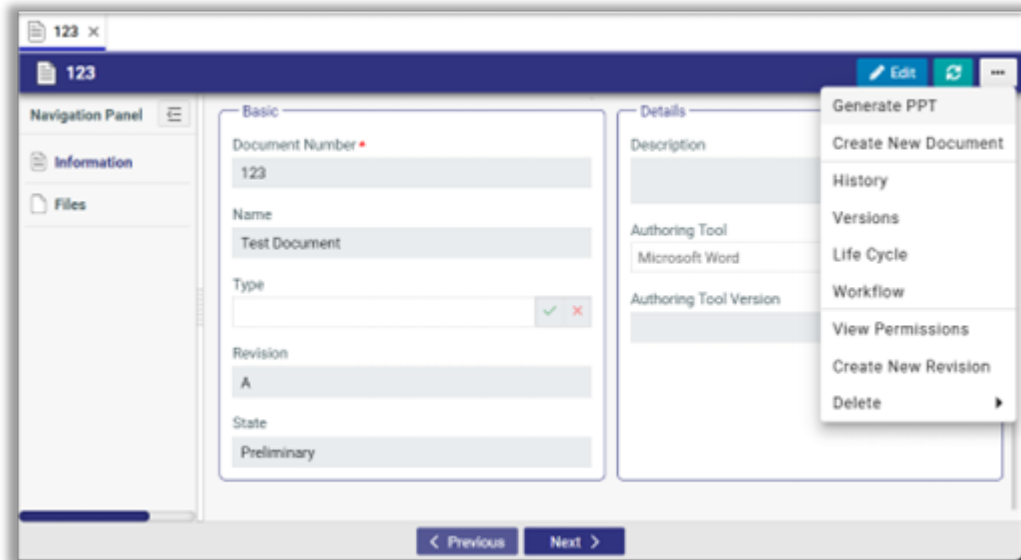
5. Add Visibility then click on Ok



6. Example of Commands with different visibility

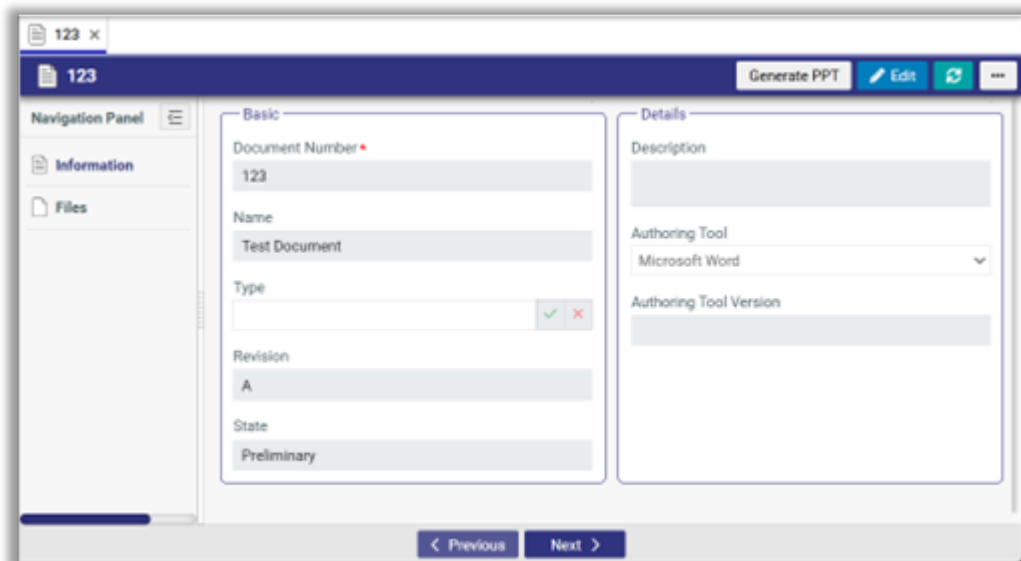
Visibility: Show In Dropdown

eg.



Visibility: Show In ButtonGroup

eg.

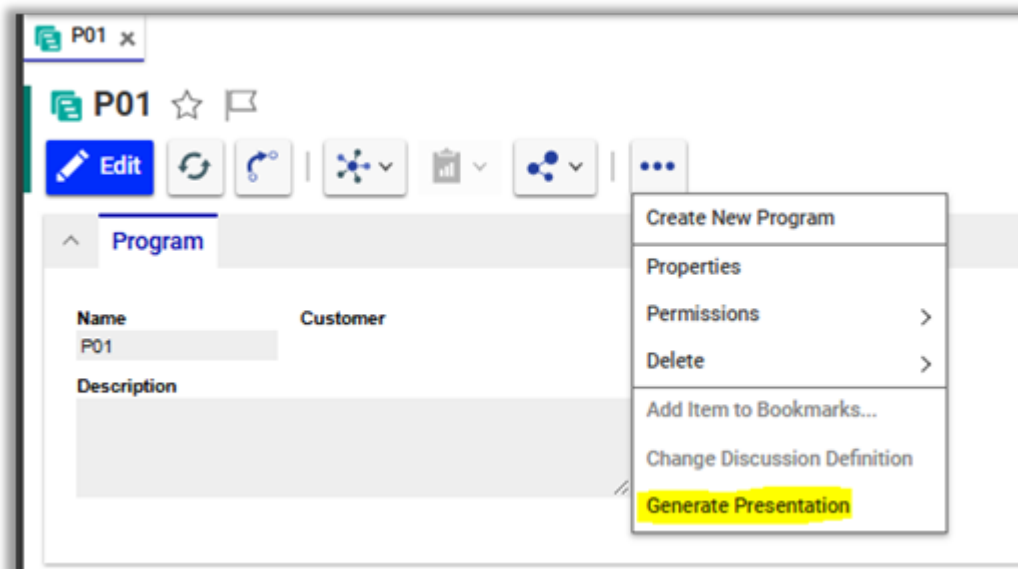


Configure Custom action for Presentation Generator

To generate presentations using the Presentation Generator, it can be configured within Aras Innovator as either custom actions or CUI (Client User Interface) menu items. Steps to use it through custom action:

1. Create a custom action on the context item type.
2. Create and attach a client method that invokes the server method named “as_GeneratePresentation”, which internally utilizes the PresentationService provided by the Presentation Generator.
3. The “as_GeneratePresentation” method can be executed on a context item, which is then used to replace property values within the presentation template.

e.g.



4. Additionally, a custom server method can be introduced to modify the values of the context item before initiating presentation generation. Instead of directly calling the “as_GeneratePresentation” method, this custom method can include logic to set or transform property values on the context item. Once the necessary adjustments are made, a PresentationService object can be instantiated, and its APIs can be invoked.

e.g.

```
Item contextItem = inn.newItem(itemType, "get");
contextItem.setID(itemId);
contextItem = contextItem.apply();

Prorigo.Plm.PresentationGenerator.IPresentationService
presentationService = new
Prorigo.Plm.PresentationGenerator.Aras.PresentationService(inn, CCO);

var fileItem = inn.newItem("File");
var fileId = presentationService.GeneratePresentation( contextItem,
"as GetDocumentAssociationJson", false ); //Boolean parameter to
indicate file to be checked in to item as per configuration or not
fileItem.setID(fileId);
var resultItem = fileItem.apply("get");
return resultItem;
```

Aras ProApp Designer - Installation Guide



Introduction

Purpose and Target Audience

This installation guide provides the required information for successfully installing and configuring Aras ProAppDesigner for Aras Innovator administrators. The installation involves updates to both the code tree and database.



Prerequisites

To install ProAppDesigner, **users need to confirm** that the installation of Aras Innovator meets the minimum requirements to run ProAppDesigner.

Check the eligibility of code tree and database:

From the **TOC**, select **Administration, Variables**.

VersionMajor	VersionMinor	VersionServiceUpdate
14	0	28
14	30	0
14	34	0
14	35	0
14	36	0
14	37	0

If it does not meet the service pack requirements listed in the previous steps, contact Aras to discuss options.

Make sure to have the **following versions** installed:

- Aras Update Tool **1.17+**
or
- Aras Innovator **Package Import Export Utilities**



Installing ProAppDesigner

ProAppDesigner must modify the Aras Innovator database with the metadata required to run the application.



Notification and Backup

1. **Notify users** that the system will be down at a scheduled time, they should log out prior to that.

Important

It is best to give at least **24-hour notice**, as well as a reminder 15 minutes prior to the upgrade.













2. **Backup** the code tree.

The **Code Tree** refers to files and folders installed to the disk when Aras Innovator was first installed.

The default path for the Code Tree installation would be something like:

C:\Program Files (x86)\Aras\Innovator

With the following contents:

Name	Type	Size
 AgentService	File folder	
 ConversionServer	File folder	
 DB	File folder	
 Innovator	File folder	
 OAuthServer	File folder	
 SelfServiceReporting	File folder	
 Tools	File folder	
 VaultServer	File folder	
 ConversionServerConfig	XML File	1 KB
 InnovatorServerConfig	XML File	2 KB
 SelfServiceReportConfig	XML File	1 KB
 VaultServerConfig	XML File	1 KB

Follow these steps to confirm the installation folder:



1. From the Windows search field, enter **Control** to access the **Control Panel**.
2. Select **Programs and Features**.
3. Right-click the **Name** column and select **More...**
 1. Select the **Comments** header if it is not checked.
 2. Click **OK**.
4. Search for the **Aras Innovator** entry in the **Programs and Features** window.
5. View the value set in the **Comments** column for the Aras Innovator entry:

Name	Publisher	Installed On	Size	Version	Comments
Aras Innovator	Aras Corporation	1/20/2022	1.09 GB	14.0.0	Installed to C:\Program Files (x86)\Aras\I...
Aras Innovator	Aras Corporation	4/12/2023	0.99 GB	14.0.17	Installed to C:\Aras\14.0.15\
Aras Innovator	Aras Corporation	3/7/2022	899 MB	12.0.0	Installed to C:\Program Files (x86)\Aras\I...

After the tree code installation path has been verified, back up the folder and all its contents:

- **Disconnect** all users from the database.

The easiest way to prevent client sessions from committing any further changes to the database is to change the database connection string in the `InnovatorServerConfig.xml` from `<DB-Connection ...` to `<xDB-Connection` and restart the `w3svc` service (IIS). This expires all session and prevents all new connections to the Aras Innovator database through the existing instance.

- **Backup** the database.
- Place these files in a **safe location**, as they will be needed to restore if the process fails.
- **Enable** database connections.

After the database backup has been completed, enable the database connection string in the `InnovatorServerConfig.xml` by changing it from `<xDB-Connection ...` to `<DB-Connection` and restarting the `w3svc` service (IIS).

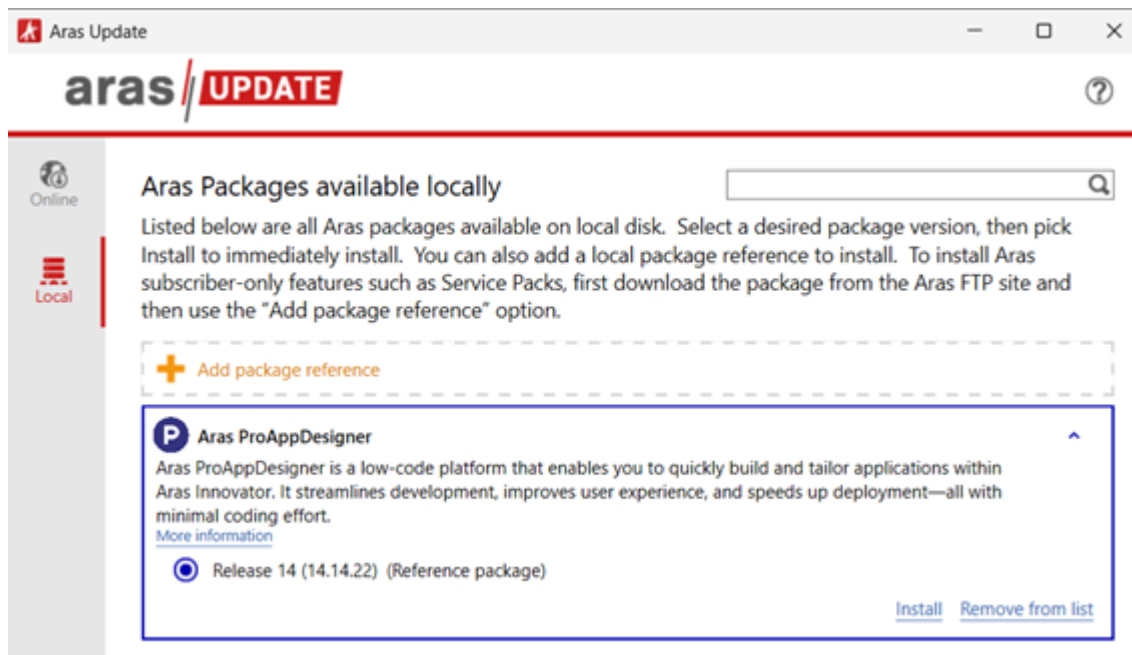


Installing the Aras ProAppDesigner

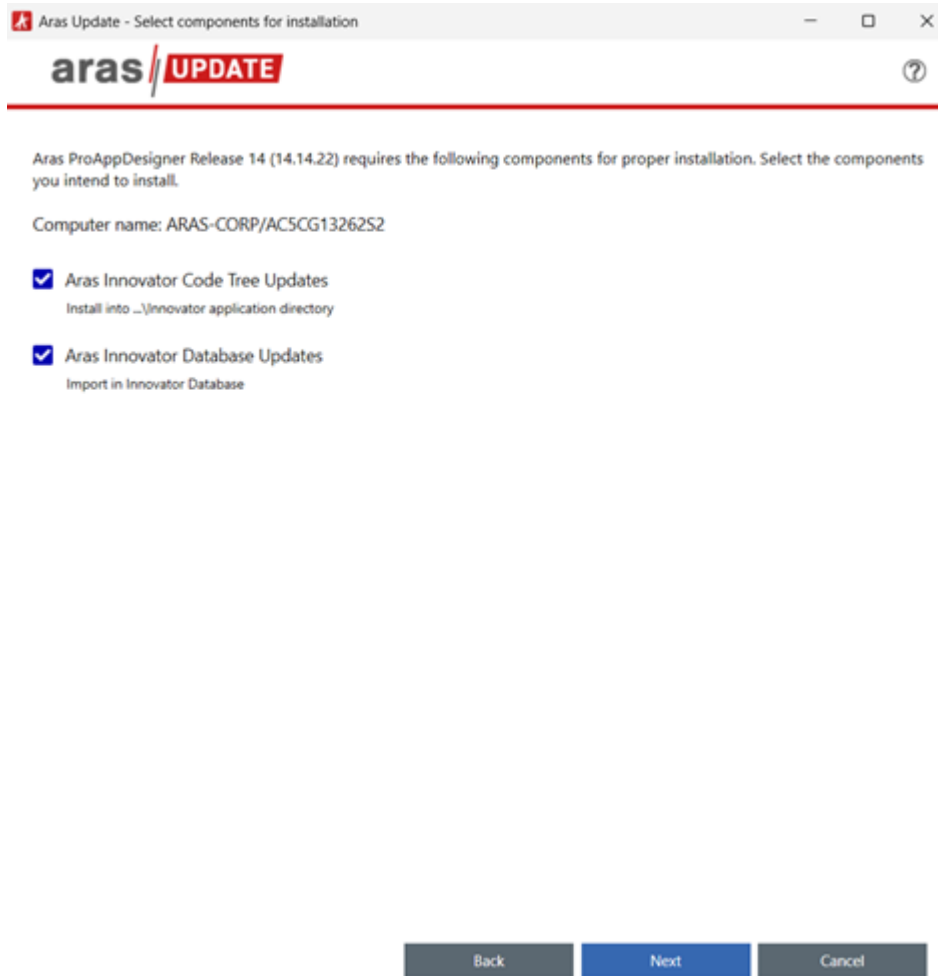
The ProAppDesigner can be installed using either the **Automated Installation** option (using the Aras Update tool), or the **Manual Installation** option (using the Aras Import Tool).

Automated Installation Option

1. Download the **Aras ProAppDesigner CD Image** from the Aras File Sharing site.
2. Unzip the Aras ProAppDesigner CD Image on the local computer.
3. Enable the **Super User** login.
4. Launch the **ArasUpdate.exe** file as administrator.
5. On a default installation, the ArasUpdate.exe file can be found in the C:\Program Files (x86)\Aras\Aras Update \ directory.
6. Locate and select the **Aras ProAppDesigner** package.



7. Click **Install**.



8. Select the **Aras Innovator Code Tree Updates** and **Aras Innovator Database Updates** options.
9. Click **Next**.
10. Select the appropriate logging option and click **Next**.
The logging option is used to record the installation attempt and can be used to troubleshoot issues.
11. Input the connection information and click **Install**.
 - o Server URL = The connection URL for Aras Innovator
 - o Database = The target Aras Innovator database
 - o Username = root
 - o Password = Password for “root” login (Default is “innovator”)

Aras Update should import everything required for Aras Innovator ProAppDesigner.

Aras Update

aras // UPDATE

Enter the following parameters for each component you are going to install and then click Install button to continue.

✓ Aras Innovator Code Tree Updates

Install Path (example - ...\Innovator application directory)

C:\Program Files (x86)\Aras\Innovator\Innovator Browse...

✓ Aras Innovator Database Updates

Server URL (example - 'http://localhost/InnovatorServer')

http://localhost/InnovatorServer

Database Name (example - 'InnovatorSolutions')

InnovatorSolutions

Username (should use 'root' User)

root

Password for 'root' User

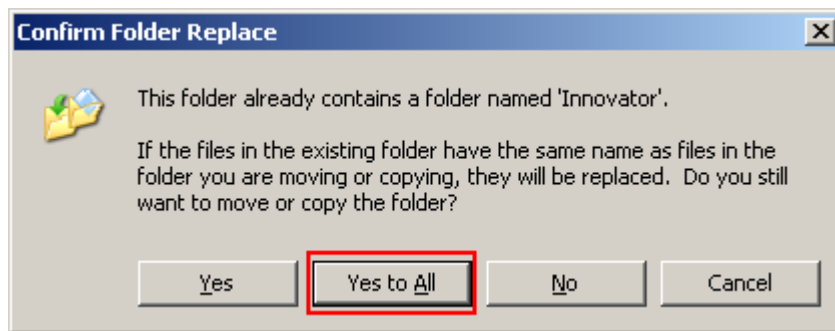
Back Install Cancel

12. After the import is complete, **disable** the Super User login.

Manual Installation Option

1. Download the **Aras ProAppDesigner** 14 CD image from the Aras File Sharing site, in the Aras ProAppDesigner Releases folder, and unzip the file on the local computer.
2. Copy the \ Files\Innovator\Client folder to the root of your installation directory. When prompted, **overwrite** the existing \Innovator\Client folder and all of its contents.

It is **recommended** that this step be performed by an administrator on the server.



- From the `\Files\Innovator\Server` folder, copy all files matching `Prorigo.*` to the root of your installation directory. When prompted, **overwrite** the existing files in the `\Innovator\Server` folder.

It is **recommended** that this step be performed by an administrator on the server.

- From the `\Files\Innovator\Server` folder, copy all remaining files **excluding** `Prorigo.*` into the `\Innovator\Server` folder. Do **not overwrite** any existing files during this step.
- Copy the `\Files\ConversionServer` folder to the root of your install directory, overwriting the existing `\ConversionServer` folder and all its contents.

It is recommended that this step be performed by an administrator on the server.

- If any of the below DLLs are available in `\Innovator\Server\bin` or in `ConversionServer\bin` folder, delete them
 - o Prorigo.Protrak.ExpressionEvaluatorStd.Impl.dll
 - o Prorigo.Protrak.ExpressionEvaluatorStd.dll
 - o Prorigo.Plm.PresentationGenerator.dll
 - o Prorigo.Plm.PresentationGenerator.Aras.dll
 - o Prorigo.Plm.Logging.Framework.dll
 - o Prorigo.Plm.DocumentGenerator.dll
 - o Prorigo.Plm.DocumentGenerator.Aras.dll
 - o Prorigo.Plm.DiscussionPanel.dll
 - o Prorigo.Plm.Core.dll
 - o Prorigo.Plm.Aras.BackgroundCronJobFramework.dll
 - o Prorigo.Plm.AppStudio.RuleEngine.Impl.dll
 - o Prorigo.Plm.AppStudio.RuleEngine.dll
 - o Prorigo.Plm.AppStudio.IntegrationFramework.dll
 - o Prorigo.Plm.AppStudio.dll
- From the root of your code tree, navigate to `\Innovator\Server` and edit the `method-config.xml` file.



1. Add below entries as shown:

...

```

<!-- ProAppDesigner DLLs : Start -->
<name>$(binpath)/Proriqo.Plm.Aras.dll</name>
<name>$(binpath)/Proriqo.Plm.Aras.AppStudio.dll</name>
<name>$(binpath)/Proriqo.Plm.Aras.TDP.dll</name>
<name>$(binpath)/Proriqo.Plm.Aras.SubscriptionFramework.dll</name>
<name>$(binpath)/Proriqo.Plm.Aras.DocumentGenerator.dll</name>
<name>$(binpath)/Proriqo.Plm.Aras.BackgroundCronJobFramework.dll</name>
<name>$(binpath)/Proriqo.Plm.Aras.RuleEngine.dll</name>
<name>$(binpath)/Proriqo.Plm.Aras.PresentationGenerator.dll</name>
<name>$(binpath)/Proriqo.Plm.Aras.IntegrationFramework.dll</name>
<name>$(binpath)/Proriqo.Plm.Aras.IntegrationFramework.Connector.dll</name>
<name>$(binpath)/Proriqo.Plm.Aras.DiscussionPanel.dll</name>
<name>$(binpath)/Proriqo.Plm.Aras.Logger.dll</name>
<name>$(binpath)/Proriqo.Plm.LicensingFramework.dll</name>
<!-- ProAppDesigner DLLs : End -->

```

</ReferencedAssemblies>

2. Save the changes to the method-config.xml file and close it

Important

If any of the DLLs mentioned in the DLLs to be deleted section, please delete the corresponding entry from method-config.xml file.

8. From the root of your code tree, navigate to `\Innovator\Client` and edit the `InnovatorClient.config` file.
 1. Change the value of the filesRevision attribute. The value should be changed from std to 2, 3, 4, etc

```

<oauth configSource="OAuth.config" />
<cachingModule moduleEnabled="true" filesRevision="2" />
<staticContent>

```

...

2. Save the changes to the `InnovatorClient.config` file and close it.

9. From the root of your code tree, open `ConversionServerConfig.xml` file.



1. Add below entries in Converters.

```
<Converters>
  <Converter name="hci CRON JOBS"
    type="Prorion.Plm.Aras.BackgroundCronJobFramework.BCJ_Converter,
    Prorion.Plm.Aras.BackgroundCronJobFramework" />
  <Converter name="hci Background Job Conversion Type"
    type="Prorion.Plm.Aras.BackgroundCronJobFramework.BCJ_Converter,
    Prorion.Plm.Aras.BackgroundCronJobFramework" />
</Converters>
```

...

2. Save the changes to the ConversionServerConfig.xml file and close it.

10. Restart the World Wide Web Service (IIS).

11. Execute Pre Import Scripts

Before you install Aras ProAppDesigner, execute the following scripts:

1. Navigate Aras Nash tool using URL <http://localhost/InnovatorServer/Client/scripts/nash.aspx>
2. Login in Nash using **admin** user.



The screenshot shows a web browser window with the URL `localhost/InnovatorServerR27/Client/X-salt=std_14.0.15.38102-X/scripts/nash.aspx`. The page header includes the Aras NASH logo, the database name `InnovatorR27_New`, and the user `admin`. Below the header, there are two dropdown menus for `TimeZone` (set to `Eastern Standard Time`) and `Culture` (set to `en-US - English (United States)`). The `Action` dropdown is set to `ApplyAML`, and the `Server` dropdown is set to `http://localhost/InnovatorServerR27/Server/InnovatorServer.aspx`. A large text area labeled `XML:` is present, with `Submit` and `Clear` buttons to its right. Below the XML area, the `Run time, sec:` is displayed as `0,000` and the `Result:` is `read`. A `ShowXml` button is located to the right of the result area.

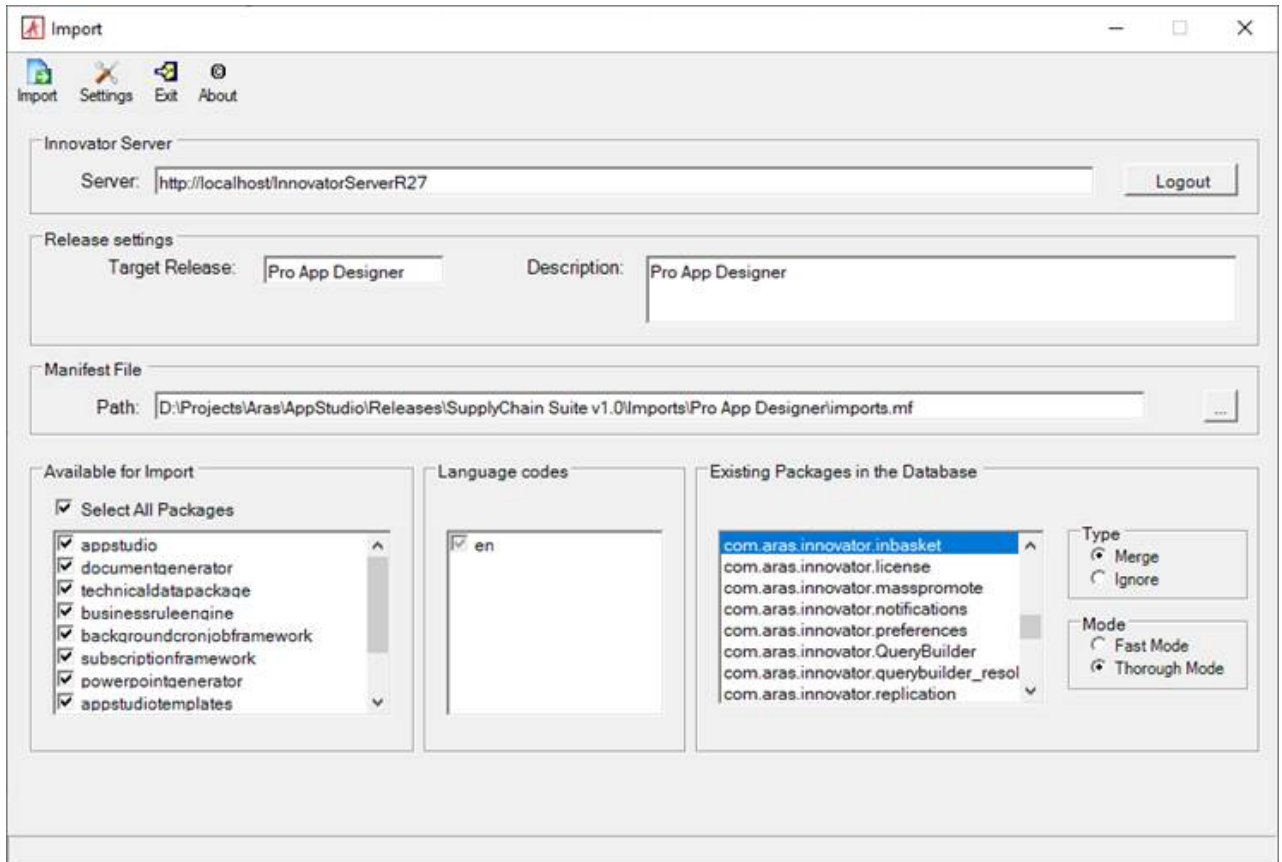
3. In the package go to folder `PreImportAMLScripts\Manual Scripts`, open each file in below sequence, copy the content in XML form and click on Submit.

- `04500_TemplateDeploy_PreAML.xml`

12. Complete the Installation of ProAppDesigner: Import the **ProAppDesigner** database package with the Package Import Export Utilities (For more information using this tool, please see the *Aras Innovator - Package Import Export Utilities* documentation.)

1. You will need to **enable** the Super User login to perform this step.
2. Browse to the `\PackageImportExportUtilities\Import\` folder and run the `Import.exe` file.





3. Enter the following information:

- **Server** = The connection URL for Aras Innovator
- **Database** = The target Aras Innovator database (login screen)
- **Username** = root (login screen)
- **Password** = Password for “root” login (Default is “innovator”) (login screen)
- **Target Release** = ProAppDesigner
- **Description** = ProAppDesigner
- **Manifest File Path** = The manifest is found in the unzipped patch folder in Aras ProAppDesigner CD Image\Imports\ProAppDesigner\imports.mf
- **Available for Import** = Check all packages
- **Type** = Merge
- **Mode** = Thorough Mode

4. Click the Import button.

13. **Disable** the Super User login.

14. Execute **Post Import Scripts**

After you install Aras ProAppDesigner, execute the following scripts:

1. Navigate Aras Nash tool using URL <http://localhost/InnovatorServer/Client/scripts/nash.aspx>



2. Login in Nash using **admin** user.

The screenshot shows a web browser window with the URL `localhost/InnovatorServerR27/Client/X-salt=std_14.0.15.38102-X/scripts/nash.aspx`. The page header displays the **aras NASH** logo, the database name `InnovatorR27_New`, and the user `admin`. Below the header, there are two dropdown menus for `TimeZone` (set to `Eastern Standard Time`) and `Culture` (set to `en-US - English (United States)`). The `Action` dropdown is set to `ApplyAML`, and the `Server` dropdown is set to `http://localhost/InnovatorServerR27/Server/InnovatorServer.aspx`. There are two large empty text areas for XML input. To the right of the first XML area are `Submit` and `Clear` buttons. Below the XML areas, the `Run time, sec:` is `0,000` and the `Result:` is `read`. A `ShowXml` button is located to the right of the second XML area.

3. In the package go to folder `PostImportAMLScripts\Manual Scripts`, open each file in below sequence, copy the content in XML form and click on Submit.

- `05000_TemplateDeploy_PostAML.xml`



Confirming the Installation

Use the following procedure to check if your database is updated correctly:

1. Log in to Aras Innovator as an administrator.
2. From the TOC, select **Administration à Variables**.
3. Confirm the following sets of variables are listed:

Pro App Designer
14.14.32.10

If at any time the installation fails, revert to your backups and contact Aras Support at support@aras.com.



Activating ProAppDesigner

You require any one of the listed feature licenses for ProAppDesigner. Once you have received the keys, activate them from **User Menu** → **Activate Feature**.

1. Aras.ProAppDesigner
2. Aras.SupplierManagementAndPortal



Setting Required Variables

Make sure to create and set values for below variables:

Variable Name	Value Example	Remarks
Prorigo.Plm.InstanceURL	http://localhost/InnovatorServer	This variable is used by subscription framework to send the item link in mail body. If not created or left empty, the mail body will not have correct item link.



Aras Technical Data Package - User Guide



Introduction

Purpose

This guide describes how to use Aras Technical Data Package for external users.



Scope

This document provides guidance on the configuration and functionality of the Aras Technical Data Package application.



Target Audience

This document is intended for external users with limited access to data.



Feature Terms and Definitions

The following Feature terms and definitions are used in this document:

Term	Definition



Guide Organization

This document describes how to use the Aras Technical Data Package application. This User Guide is organized into the following sections:

SECTION 1: Introduction	This section outlines the purpose and scope of this User Guide, as well as its intended target audience. It supplies a brief overview of each section to help users in navigating the guide.
SECTION 2: Introduction to Technical Data Package (TDP)	This section outlines the purpose of the Technical Data Package (TDP) and provides an overview of the key functionalities available within the Aras Technical Data Package application.
SECTION 3: TDP Template	This section outlines how to select and generate TDP Excel Template which are the fundamental building blocks in TDP.
SECTION 4: Export TDP	This section outlines how to create custom action and export TDP.
SECTION 5: Import TDP	This section outlines how to create custom action and import TDP.



Introduction to Technical Data Package (TDP)

Purpose of Technical Data Package

The Aras Technical Data Package (TDP) application enables offline access for external suppliers by allowing specific data and files to be securely shared. Through the Export TDP action, users can export items, relationships, and associated files into a ZIP archive. This package includes an Excel file containing structured data across multiple worksheets, with columns representing properties from ItemTypes, RelationshipTypes, and Files. The package is generated on the server based on user actions, lifecycle promotions, or scheduled jobs. When updates are made, the Import TDP action allows users to re-import the modified data and files back into Aras Innovator. The Excel template used for package generation is selected according to predefined configuration rules.



TDP Template

Introduction

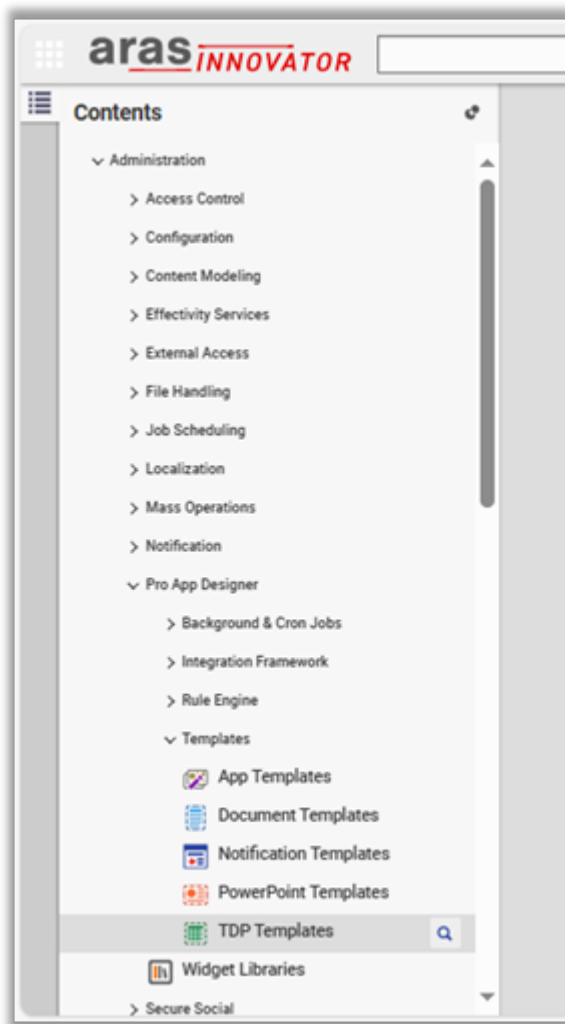
The Technical Data Package (TDP) is generated based on a configured Excel template that includes multiple worksheets. Each worksheet contains columns mapped to properties from ItemTypes, RelationshipTypes, and Files, ensuring structured and comprehensive data representation. Only one TDP template item can be associated with a specific ItemType, maintaining a clear and consistent configuration. However, each TDP template can include multiple Excel files, allowing flexibility in how data is organized and presented within the package.

TDP Template Management

Create Template

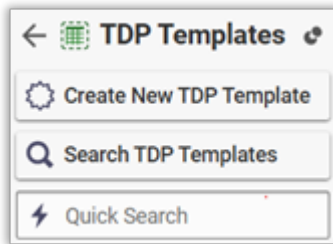
The following steps outline the process of creating **TDP Template**:

1. From the Table of Contents select **TDP Templates**.
(Go to TOC → Administration → Pro App Designer → Templates → TDP Templates.)

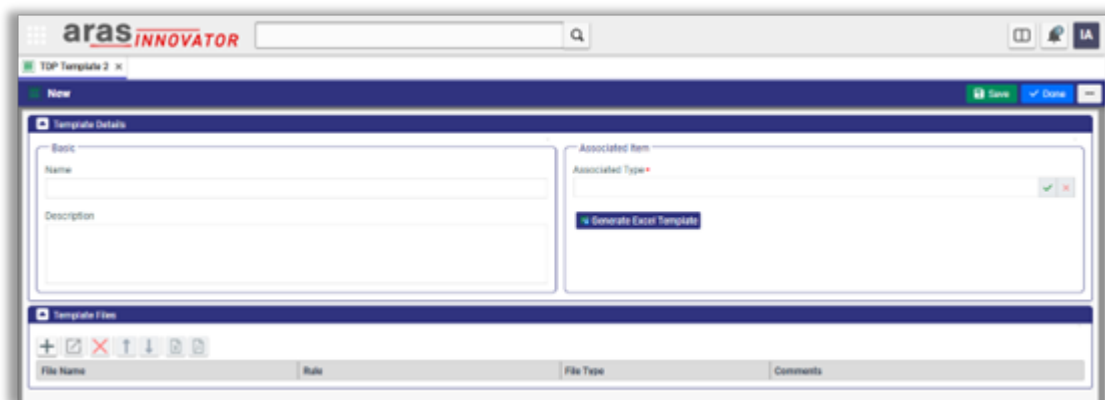


2. The **TDP Templates Table of Contents** appears. Click **Create New TDP Template**.





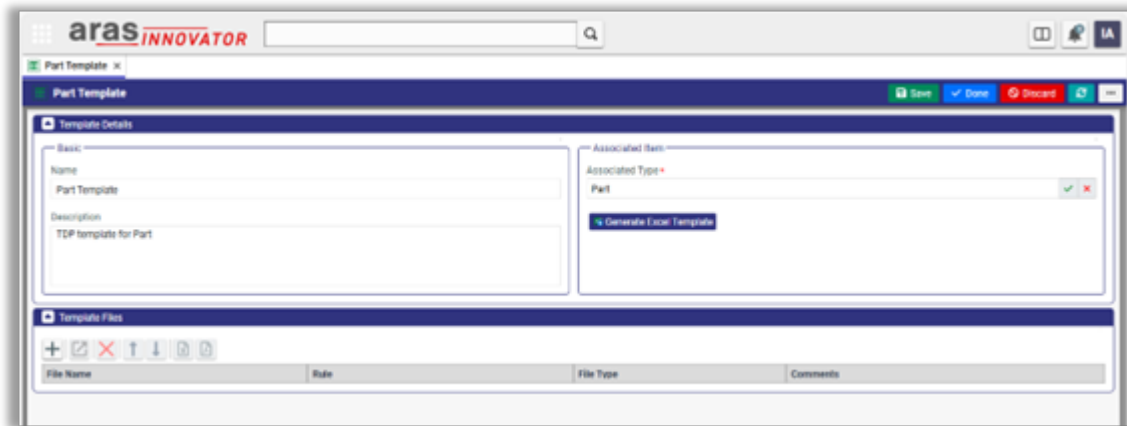
3. Fill in the required properties, add the appropriate template file, and then click Save or Done to complete the process.



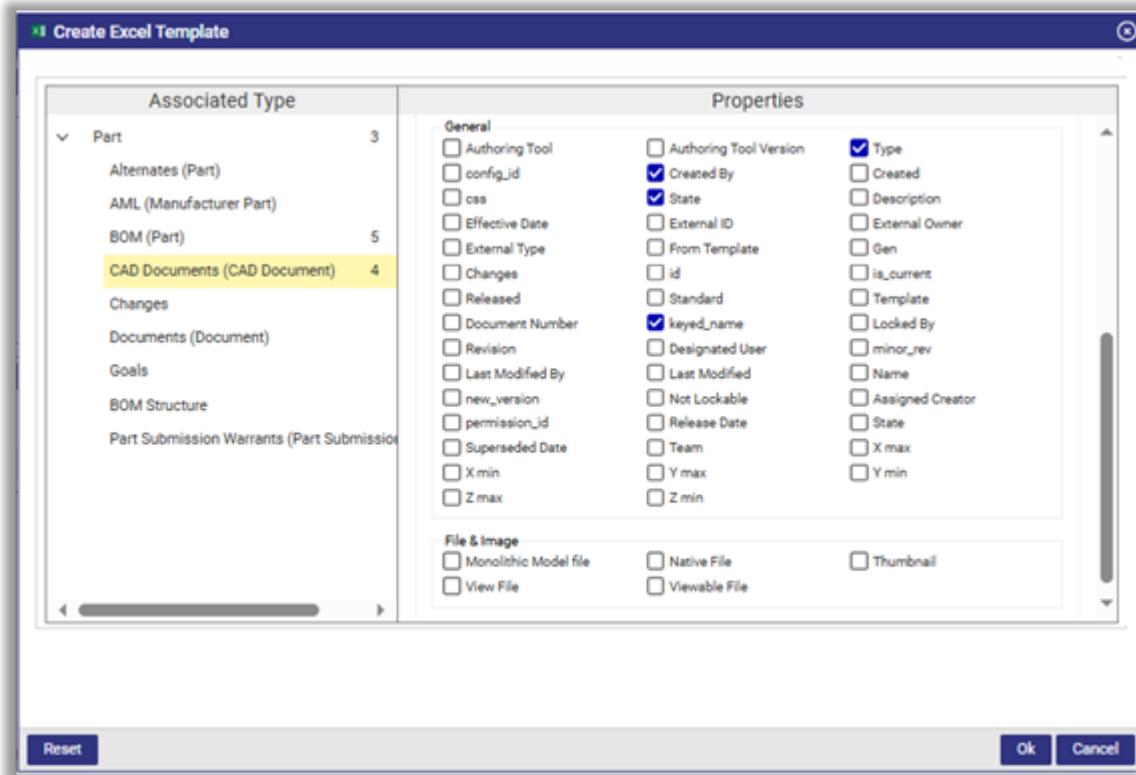
Auto-Generate TDP Template

The steps below detail the process for automatically generating a TDP template file:

1. Click on **Generate Excel Template** button under Associate Item group.

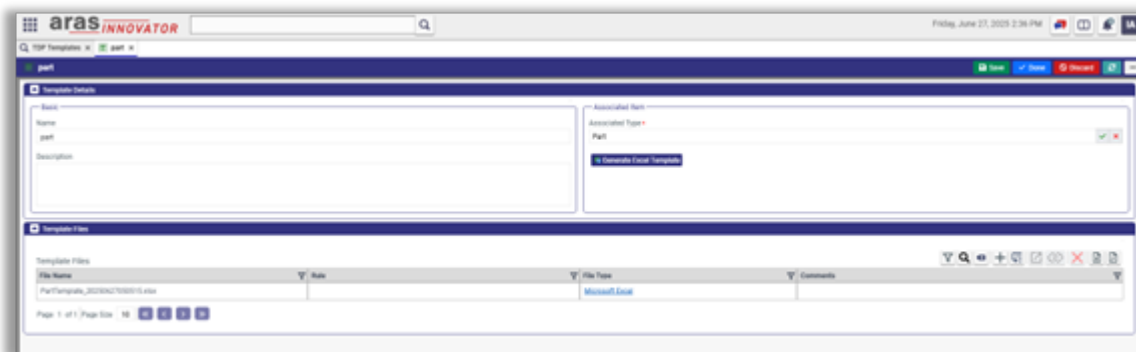


2. The **Create Excel Template** dialog will open.



3. Select the properties and click on **Ok** button.

4. New **Template Files** relationship will create.



Excel Template Selection

Each TDP template can have multiple Excel files attached. During the Export TDP operation, the appropriate Excel file is automatically selected based on predefined configuration rules:

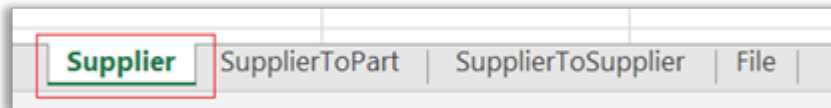
1. If an Excel file within a TDP template does not have any rule configured, it will be selected by default during the Export TDP operation. In cases where multiple Excel files lack rules, the system will automatically select the first available file.
2. If all Excel files within a TDP template have rules configured, the system will select the first file that satisfies its rule during the Export TDP operation. If none of the files meet their respective rule conditions, the Export TDP operation will be aborted.



Template File Assumptions

To ensure successful and accurate export of a Technical Data Package (TDP), the Excel files attached to the TDP template must adhere to specific formatting and structural guidelines.

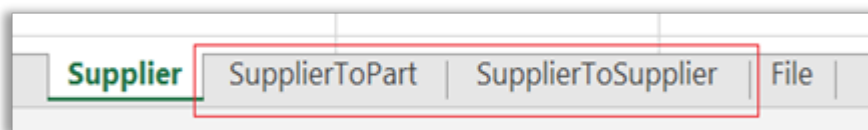
1. The name of the first worksheet must exactly match the name of the context ItemType. The property columns names should be exactly same as ItemType property names.



2. The final worksheet in the Excel file must include fully qualified paths to the file locations required for the Technical Data Package (TDP). These file paths can originate from ItemType properties, Relationship properties, or File relationships. If the file location is defined through a File relationship, the "Property" column in the worksheet may be left blank. An example of how to describe file location is shown below.

FileItemTypePath	Property
Supplier/SupplierToPart/Part/Part Document/Document/Document File/File	
Supplier/SupplierToPart/Part/Part CAD/CAD	native_file
Supplier/SupplierToPart/Part/Part CAD/CAD	viewable_file
Supplier/SupplierToPart/Part/Part CAD/CAD/CADFiles	attached_file

3. Intermediate worksheets can be added as needed to export data related to relationships and associated items. Each of these worksheets must be named exactly after the corresponding Relationship name to ensure proper mapping. Additionally, the column headers within each intermediate sheet must match the property names defined in the Relationship.



4. In each worksheet, the columns should display the property labels, while the corresponding property names must be included as comments within the header cells.



F	G	H
Keyed Name	s keyed_name	Workspace Number

5. If the property type is item then add itemtype name in comment.

B	C	D
Created By	related.created_by_id(keyed_name:User)	Description

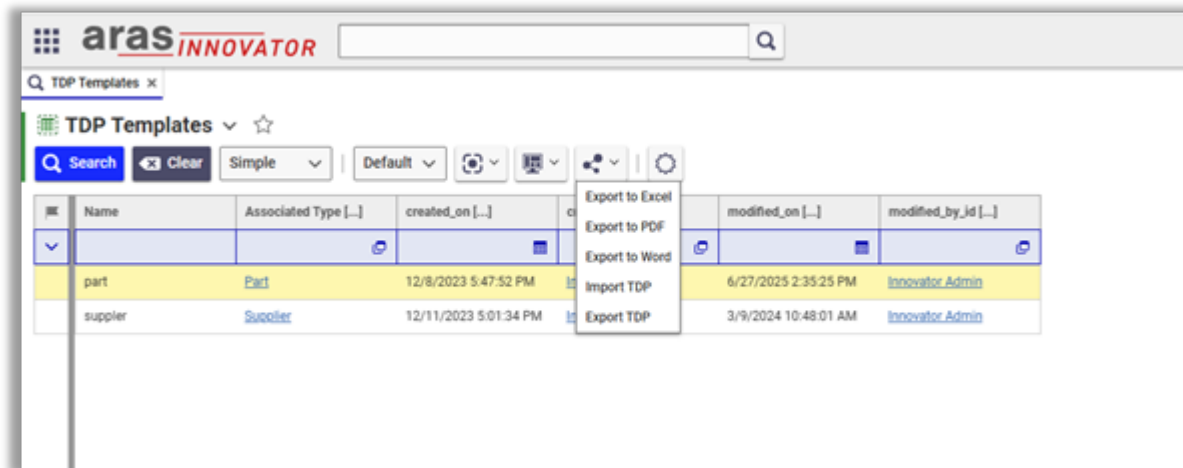
6. The RelatedItem properties can be exported by adding the prefix “related.” to the property name.

A	B	C
Classification	related.classification	Created On

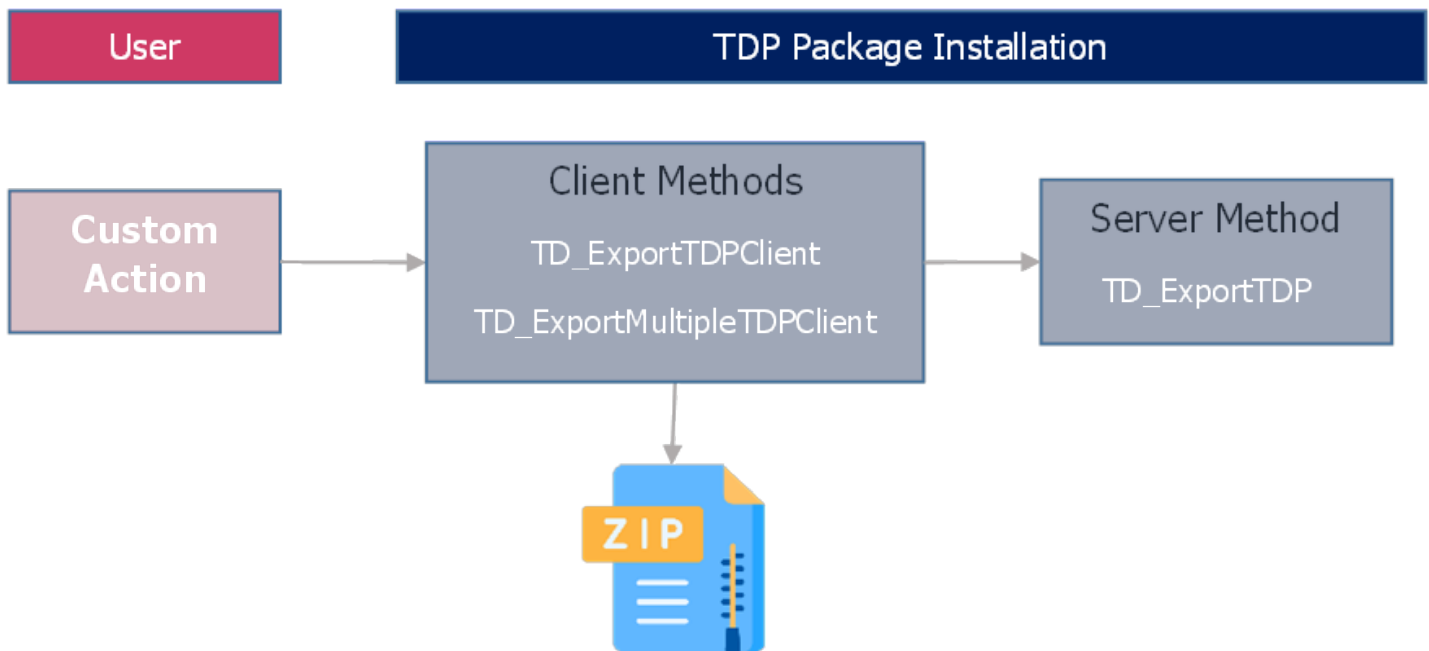


Export TDP

The Export TDP action can be configured for any ItemType and made available on both the Item Details page and the Item Search page.



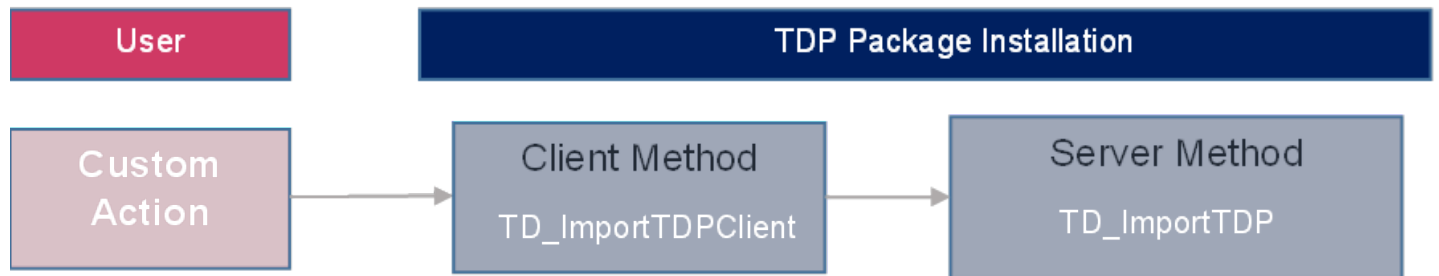
Once the standard Export TDP UI action is configured by the user, the associated client method can invoke server-side methods provided with the TDP installation to generate the Technical Data Package. The server method processes the request and returns the TDP item to the client, which then uses standard Aras functionality to download the package to the user's default local download location. The exported package is delivered as a ZIP file containing an Excel document with all relevant items and relationships data, along with any associated files. When exporting from the Search Grid, users can select multiple items, and the system will compile all related data and files into a single Excel file and ZIP archive for export.



The Excel file generated as part of the Technical Data Package (TDP) can be edited offline, allowing users to update data as needed before re-importing it into Aras Innovator. To prevent accidental modifications, the File worksheet within the Excel file is password-protected. The default password can be changed by updating the value stored in the `td_TDPPassword` variable. Additionally, each worksheet contains certain columns highlighted in grey—these are essential for the re-import process and must not be edited or deleted to ensure successful data integration.

Import TDP

Using the 'Import TDP' action, the updated package can be re-imported into Innovator. Users must configure the 'Import TDP' action within the Search page of any ItemType.



Aras Word PDF Generator - User Guide



Introduction

Purpose

This User Guide describes how to use the Aras Word/Pdf Generator for external users.



Scope

This document provides instructions for configurations and functionality of Aras Word/Pdf Generator application.



Target Audience

This document is intended for external users with limited access to data.



Feature Terms and Definitions

The following Feature terms and definitions are used in this document:

Term	Definition



Guide Organization

This document describes how to use the Aras Word/Pdf Generator application. This User Guide is organized into the following sections:

SECTION 1:	This section defines the purpose and scope of the User Guide, identifies its intended audience, and provides a concise summary of each part to assist users in navigating the document effectively.
Introduction	
SECTION 2:	This section explains the purpose of the Aras Word/PDF Generator and provides an overview of its key features. It also includes guidance on creating templates.
Introduction to	
Aras Word/pdf Generator	



Introduction to Word/PDF Generator

The Aras Word/PDF Generator automates the creation of Microsoft Word and PDF documents based on user selections made through wizard pages. Documents are generated using pre-configured Word templates and do not require any plugins or Office connectors. Generation occurs server-side and can be triggered by user actions, lifecycle promotions, or scheduled jobs.

Templates are designed with content controls mapped to properties of ItemTypes and RelationshipTypes, allowing dynamic data population. Documents can also be assembled from existing files stored within Aras Innovator. Templates may include repeating content controls, which are linked to data sources derived from relationship collections or server method outputs.

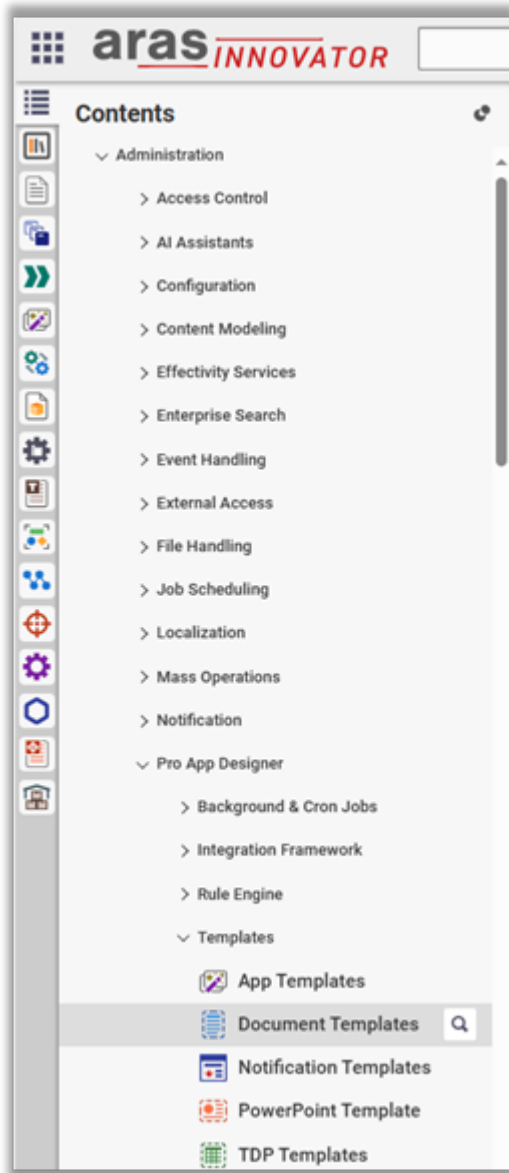
Additional features include watermarking based on lifecycle states and rule-based template selection for document generation.

Document generation can be initiated by any user action or system event. The process utilizes the template associated with the relevant ItemType.



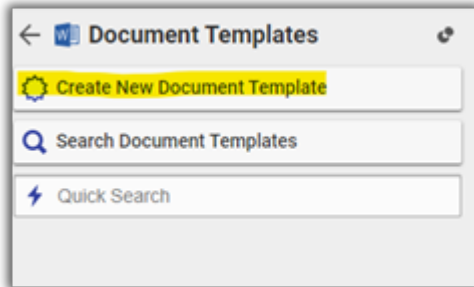
Document Generator Template Configurations

1. Go to TOC → Administration → Pro App Designer → Templates → Document Templates.

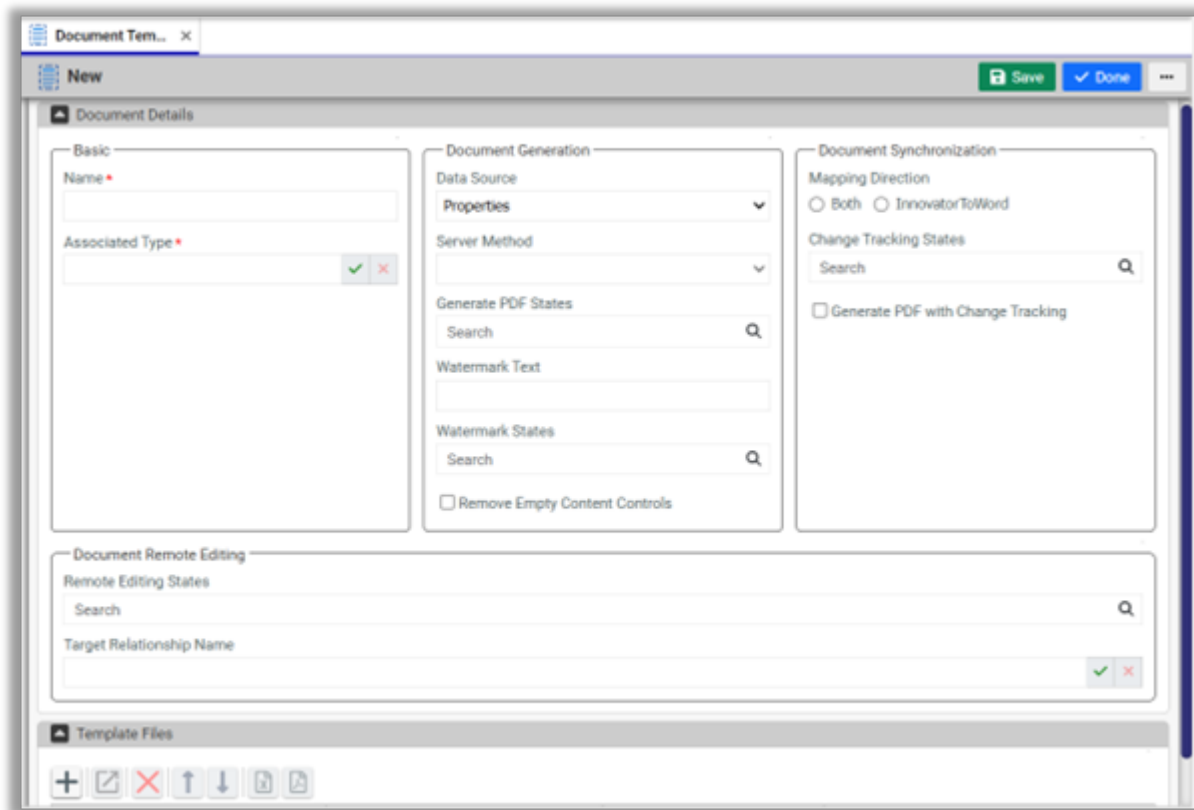


2. Click on Create New Document template.





3. Enter the necessary information, save your work, and add the template file accordingly.



Following are the details for Document Template properties:

Basic:

- **Name:** Template Name is a required property used to identify the document template. A valid name must be provided in order to save the template.
- **Associated Item:** Context Item specifies the ItemType for which the document template is being created. This is a required property and must be defined in order to save the template.



Document Generation:

- **Data Source:** Supported values are 'Properties' (selected by default, Content variables are mapped using the properties of the selected ItemType), 'Server Method' (Content variables are mapped using the output of a specified server method).
- **Server Method:** Server Method Selection option becomes available when 'Server Method' is chosen as the data source for content mapping. All server methods attached as Server Events to the associated ItemType will be listed for selection. The selected method will be used to map content variables within the document template.
- **Generate PDF States:** States list for which PDF will be generated.
- **Watermark States:** Specifies the list of states for which watermarks should be applied when generating PDF documents.
- **Watermark Text:** Text to be used for watermarking. This attribute is applicable only if the watermark States attribute is defined.
- **Remove Empty Content Control:** Determines whether empty content controls should be removed from the generated PDF document.

Document Synchronization:

- **Mapping Direction:** Supported values are 'Both' (Content variables will be mapped from Innovator to Word and Vice versa), 'InnovatorToWord' (Content variables mapping will be only from Innovator to Word)
- **Change Tracking States:** States list for which mapping will be done after changes.
- **Generate PDF with Change Tracking:** Determines whether a change-tracked PDF should be generated, based on whether any modifications have occurred.

Document Remote Editing:

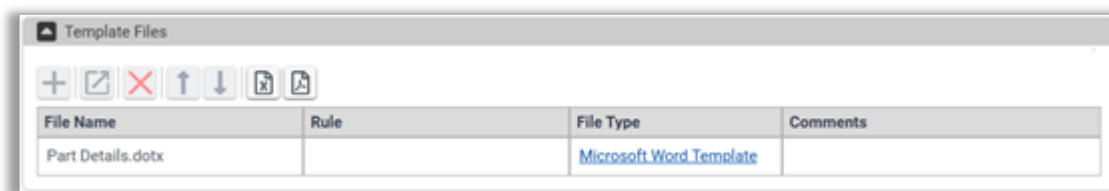
- **Remote Editing States:** States list for which remote editing is allowed.
- **Target Relationship Name:** Name of relationship where the files will attach after processing.



Template Selection

Each Document Generator template can have multiple Microsoft Word files associated with it. During the document creation process, the appropriate Word file is automatically selected based on predefined rules:

- If a Word file is attached to the template without any selection rule configured, it will be chosen by default during document generation. In cases where multiple Word files lack rules, the system will automatically select the first available file.
- If all attached Word files have selection rules defined, the system evaluates each rule in order. The first Word file that passes its validation criteria is selected for document generation. If none of the files satisfy their respective rules, the document generation process is aborted.



File Name	Rule	File Type	Comments
Part Details.dotx		Microsoft Word Template	

Word Template Creation

Enabling Developer Tab in Microsoft Word

In Microsoft Word, the Developer tab provides access to content controls, which are essential for creating dynamic templates. However, this tab is not visible by default. To enable it, navigate to **File > Options > Customize Ribbon**. In the right-hand panel of the customization window, locate and check the Developer checkbox, then click OK. Once activated, the Developer tab will appear in the ribbon, offering a variety of content controls such as text boxes, drop-down lists, and combo boxes. Each of these controls comes with configurable properties that allow users to tailor their behavior and data bindings to suit specific template requirements.

The WordSynchronizer application utilizes three types of content controls in Microsoft Word: Plain Text, Rich Text, and Date Picker. Each control type serves a specific purpose in template design and data mapping. The table below outlines the content control types used in WordSynchronizer along with their general descriptions:

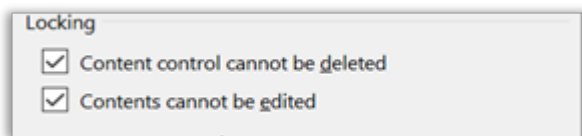
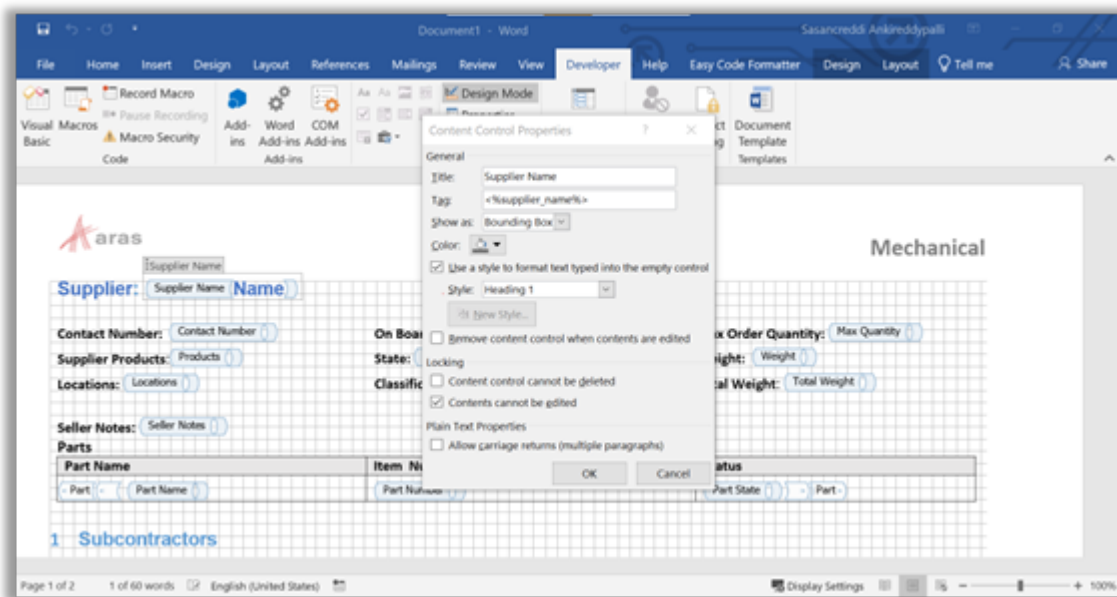
Type	General Description
Plain Text	A plain text content control is designed to hold only plain text, either in a single paragraph or across multiple paragraphs. It cannot contain complex elements such as tables, images, or other embedded content controls. It is used for Aras Innovator à MS Word mapping and MS Word à Aras mapping as well.
Rich Text	A rich text control allows users to enter and format text with styles such as bold, italic, and underline. It can also contain complex elements like tables, images, and even other content controls. However, in the context of MS Word → Aras, this control is only supported for content flowing from Word to Aras. To store the HTML-formatted content in Aras, an additional text property must be configured to handle the rich text data. Please note that if you have summary as a Property and you want to have its html equivalent property it has to be 'summary_html'.
Date Picker	Displays a calendar from which the user can click a date. It is used for Aras Innovator à MS Word mapping and MS Word à Aras as well.



Repeating	This control allows the duplication of its entire contents, including nested content controls. It is typically used to repeat structured elements such as paragraphs or table rows. null This content control is available onwards MS Office 2013.
-----------	--



Using Controls: Plain Text Content Control with Locking Behavior



Content control cannot be deleted

1. This restricts the user to delete the content control.
2. Recommended to set this option 'ON' once template design is freeze.

Contents cannot be edited set to True [Used for Aras à MS Word mapping]

When the “Content Cannot be Edited” option is True, it prevents users from modifying the content externally. This setting is used to insert property values from Aras Innovator directly into the Word document at designated placeholders. For instance, if the tag `<%supplier_name%>` is used and the option is set to True, the system will automatically substitute this tag with the value of the `supplier_name` property from Aras Innovator.

It's important to note that this configuration supports a one-way mapping from Aras Innovator to Microsoft Word. Whenever the goal is to retrieve and display a property value from Aras Innovator in



a Word document, this option should always be set to True. For this behavior, only Plain Text and Date Picker content controls are supported—Rich Text controls are not compatible.

Contents cannot be edited set to False [Used for MS Word à Aras mapping]

When the “Content Cannot Be Edited” option is False, users can enter or modify content within the control, and the updated value will be saved to the corresponding property of the Aras Innovator item during the Save operation. This type of mapping supports Rich Text, Plain Text, and Date Picker content controls. It’s important to note that this mapping is only valid for direct properties of the associated item and does not apply to relationship properties or derived data.



Using Controls: Repeating Text Content

Repeating Content Controls allow entire tables or specific sections within a Word document to be repeated multiple times, each populated with a different dataset. This functionality relies on input provided in XML format, leveraging Microsoft Word's custom XML feature to dynamically generate content.

For more details on Word's custom XML capabilities, please refer to the following link:

<https://docs.microsoft.com/en-us/visualstudio/vsto/how-to-add-custom-xml-parts-to-document-level-customizations?view=vs-2019>

The WebSynchronizer application processes JSON input received from an Aras Innovator Server Method, which is configured as the data source property of the Document Template item. When the associated content control in the Word template is a repeating control, WebSynchronizer automatically converts the JSON data into XML format. This XML is then used to populate the repeating sections in the document. For example, in the case of a table, the data row will be repeated as many times as there are entries in the XML dataset.

Parts		
Part Name	Item Number	Status
Test1	1211	Preliminary

Parts		
Part Name	Item Number	Status
- Part - (Part Name Test1)	(Part Number 1211)	(Part State Preliminary) - Part -

Please refer following XML for condition table which is returning single condition. Rows will be repeated number of times for `<SupplierToPart.related_id>` tag in XML.

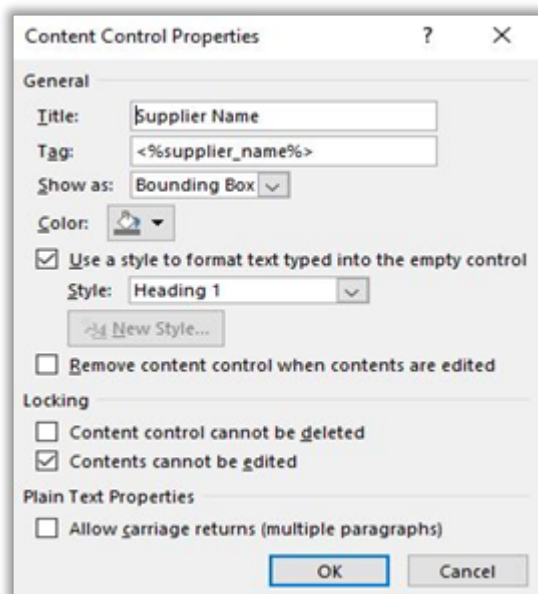
```
<root>
<SupplierToPart.related_id>
<name>Test1</name>
<item number>1211</item number>
<state>Preliminary<state>
```

```
</SupplierToPart.related_id>
</root>
```

For Reference, please also find the relevant JSON below.

```
"\" SupplierToPart.related_id \"
:[{\"name\": \"Test1\"
, \"item number\": \"1211\"
, \"state\": \"Preliminary \"}]"
```

Following image shows how repeating content control is mapped to `<%SupplierToPart.related_id%>` which is tag from XML.



Tag Naming Conventions and its meaning

In the above image tag value is given as `<%supplier_name%>`. Let us see what it means

1. `<%.%>` Indicates, processing at server end. Please note that it is mandatory to follow this convention.
2. `<%supplier_name%>` Here 'supplier_name' is actually the name of the Property of 'Supplier' Item Type. By this tag, application recognizes this property value to be fetched from Aras Innovator and substitute in the MS Word at relevant placeholder
3. `<%SupplierToPart.related_id.name%>` Here involves relationship.
 1. 'SupplierToPart': is the RelationshipType of Supplier Item Type
 2. 'related_id' is the property pointing to Part Item Type
 3. 'related_id' is the property pointing to Part Item Type

So, this tag value actually fetches the value of Property 'name' attached to 'Part' item type.

Using this naming convention, the server follows a specific sequence when processing a tag at server

1. Go to 'SupplierToPart'
2. Search for related_id and get the Data Source value which is 'Part' in this case
3. Go to Part and access the 'name' property value.



Chart Support

Query Definition is used to retrieve data for chart generation in Aras Innovator. Refer following example snippet for fetching part status chart

QueryDefinition=as_PartsList

Function=Count

Columns=state

GroupBy=classification

This configuration will be provided as a “Alt Text” property of chart object in Word Template.



Mapping Content Control to an Item Type in Server Method

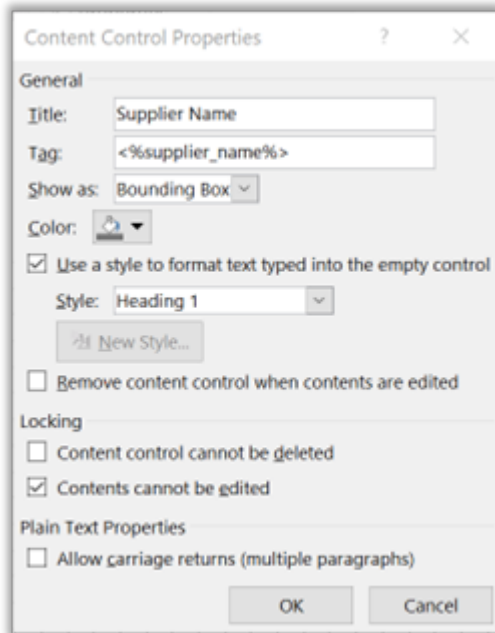
Let's consider the ItemType "Part". If you want to fetch specific properties from this item, the following snippet demonstrates how to retrieve the name which is property of Part item. These retrieved properties are intended to be mapped to content variables in the Word document, as described in the previous section.

```
var partItemQuery = innovator.newItem("Part", "get");  
partItemQuery.setID(partId);  
partItemQuery.setAttribute("select", "name");  
var partItem = partItemQuery.apply();  
partObject.Add("name", partItem.getProperty("name "));  
In case of multiple properties, please refer the code snippet below  
var partItemQuery = innovator.newItem ("Part", "get");  
partItemQuery.setID(partId);  
partItemQuery.setAttribute("select", "name,item_number");  
var partItem = partItemQuery.apply();  
partObject.Add("name", partItem.getProperty("name"));  
partObject.Add("item_number ", partItem.getProperty("item_number"));
```

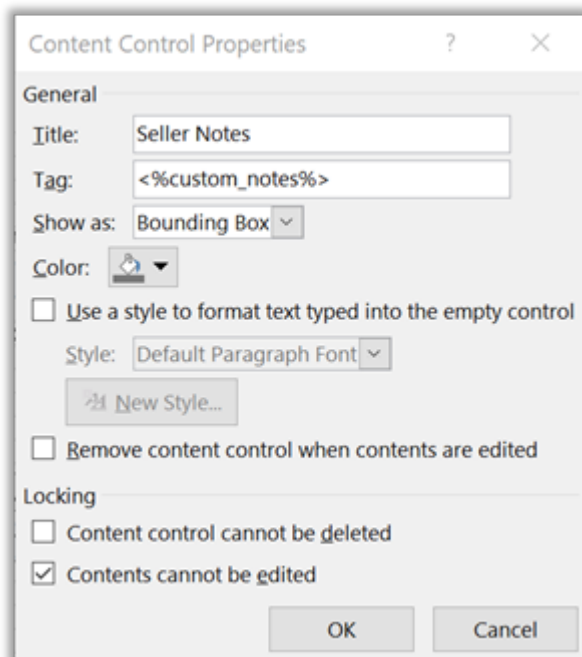
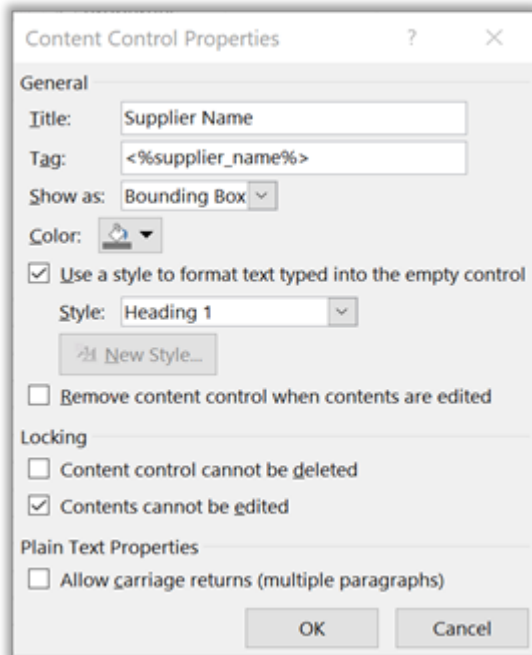


Mapping to remove empty content controls along with Label

1. Add label as plain text content control. In the properties dialog, check the option “Contents cannot be edited” .

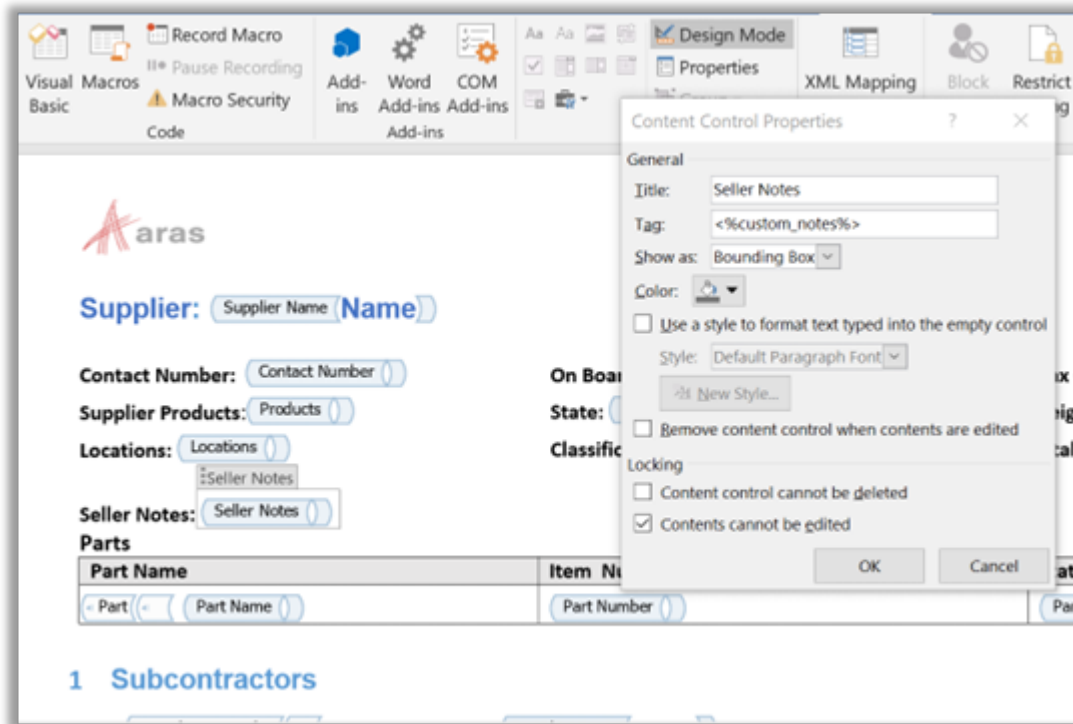


2. Label content control and data content control title should be same as shown below



3. Template designers can use line and spacing settings in Microsoft Word to add extra space after a content control. As shown in below image for Seller Note data content control.





Steps to edit placeholder text directly from 'Design Mode' and to preserve the original style



Click "in" the existing placeholder text as shown and modify the text to preserve formatting. Alternatively, you can also select the whole text and write yours

Clicking inside the placeholder text preserves its formatting. However, if you delete the placeholder text or begin typing at its start, the placeholder style will be lost..



Notice that Starting at the beginning text is '**Black - Bold**'. Here the original placeholder text style is lost. To avoid this you can also select the whole text and write yours

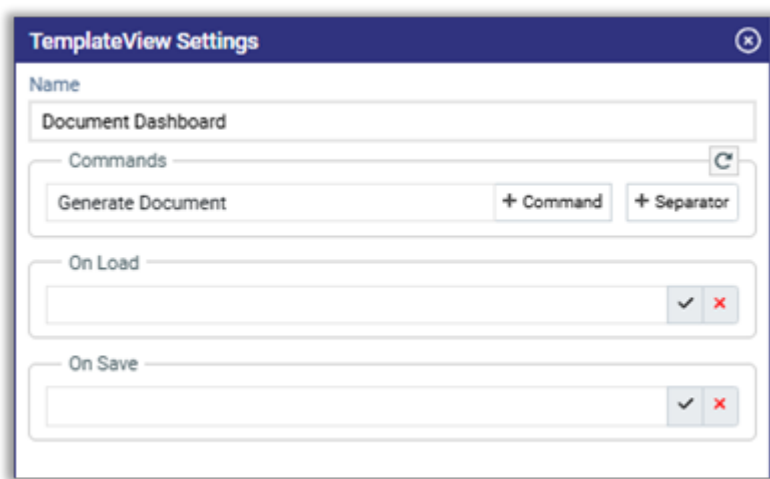


Starting "in" the existing text preserve the placeholder text style. We can later delete the unwanted text

Configure Word Generator Command on Template View

To generate documents using this generator, it can be configured as a command on Item template with following steps

1. Go to TOC → Administration → Pro App Designer → Templates → App Templates → Open App Template and then Open Template View
2. Go to More → Settings → Add Command name (like below image) → Click on Command button



3. Click on Events (highlighted in below image)

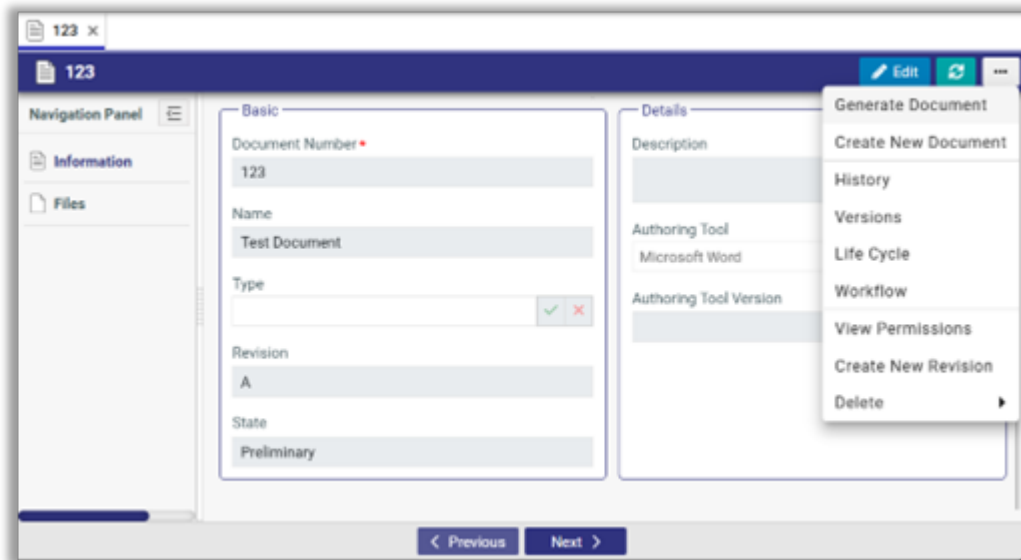
The screenshot shows the 'Name' field set to 'Document Dashboard'. Below it is a 'Commands' section with an 'Enter Name' input field, '+ Command', and '+ Separator' buttons. A table below lists commands with columns for 'Name', 'Visibility', and 'Actions'. The 'Generate Document' command is selected in the 'Actions' column. Below the table are 'On Load' and 'On Save' event configuration sections, each with a text input field and check/clear buttons. An 'Events' tooltip is visible over the 'Generate Document' command's actions.

4. Select Actions and then Select Generate Document, click Ok.

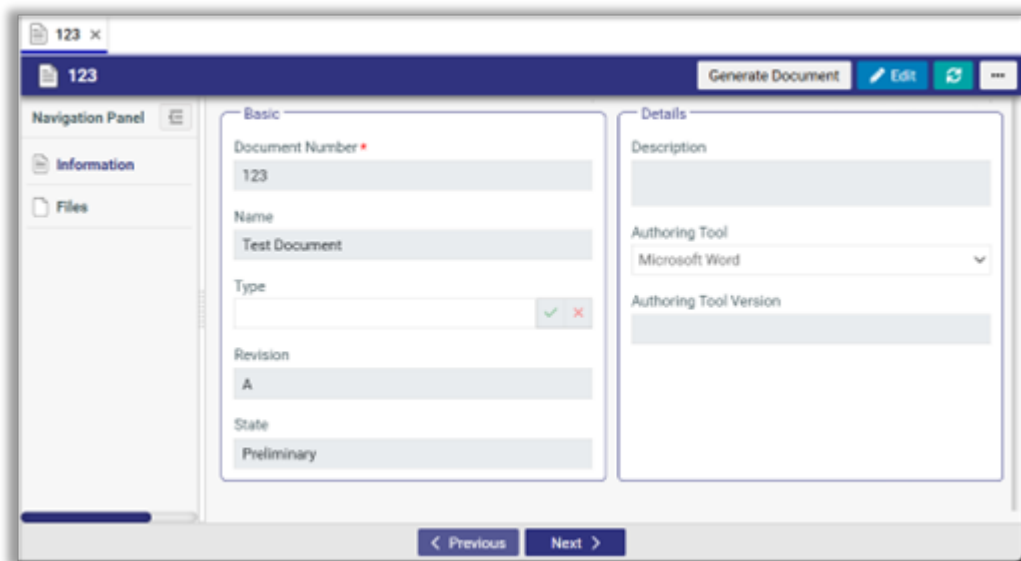
The 'Events' dialog box has two tabs: 'On Click' (selected) and 'Can Show'. Under 'On Click', there are three radio buttons: 'Flyouts', 'Actions' (selected), and 'Script ?'. Below these is a list of actions with radio buttons: 'Where Used', 'Structure Browser', 'Workflow Signoffs', 'Generate Document' (selected), 'Export TDP', 'Hoops Viewer', and 'Generate Presentation'. 'Ok' and 'Cancel' buttons are at the bottom.



- 5. Add Visibility, then click on Ok.
- 6. Example of Commands with different visibility
Visibility: Show In Dropdown
eg:



Visibility: Show In ButtonGroup
eg:

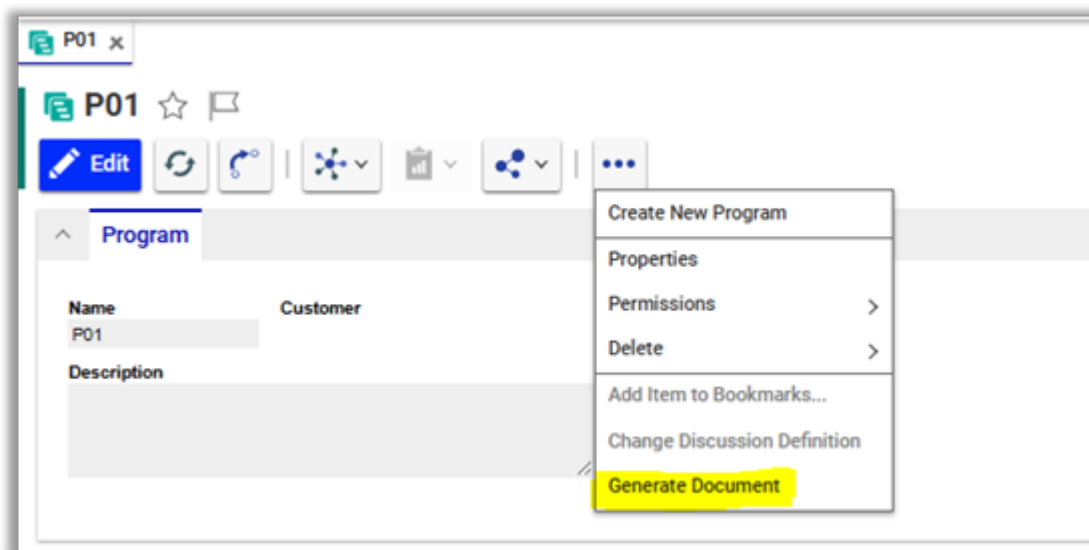


Configure Custom action for Word Generator

To generate Word or PDF documents using this generator, it can be configured as a custom action or added to CUI menus within Innovator. Below are the steps to use it via a custom action:

1. Create a custom action on the context item type.
2. Attach a client method that calls the server method "dg_WordDocument_GenerateDocument". This server method internally utilizes the DocumentService from the Document Generator to process and generate the Word or PDF document.
3. The method "dg_WordDocument_GenerateDocument" can be executed on context item, which will be used to replace the properties values in Document template.

e.g.



4. Additionally, a custom server method can be introduced to modify values on the context item before processing. Instead of directly invoking the 'dg_WordDocument_GenerateDocument' method, you can create a separate method that sets the required values on the context item. After that, the DocumentService object can be instantiated and the relevant APIs can be called.

e.g.

```
Item contextItem = inn.newItem(itemType, "get");
contextItem.setID(itemId);
contextItem = contextItem.apply();
```

```
Prorigo.Plm.DocumentGenerator.IDocumentService documentService = new
Prorigo.Plm.DocumentGenerator.Aras.DocumentService(inn,CCO);
var fileItem = inn.newItem("File");
var fileId
=documentService.GenerateWordDocument(contextItem,"GetDocumentAssociationJsc
//Boolean parameter to indicate file to be checked in to item as per
configuration or not
fileItem.setID(fileId);
var resultItem = fileItem.apply("get");
return resultItem;
```



ProAppDesigner Rule Engine - User Guide



Introduction

Purpose

This User Guide describes how to use the ProAppDesigner Rule Engine for external users.



Scope

This document provides instructions for configurations and functionality of Rule Engine application.



Target Audience

This document is intended for external users with limited access to data.



Feature Terms and Definitions

The following Feature terms and definitions are used in this document:

Term	Definition



Guide Organization

This document describes how to use the ProAppDesigner Rule Engine application. This User Guide is organized into the following sections:

SECTION 1:	This section outlines the purpose and scope of this User Guide, as well as its intended target audience. It supplies a brief overview of each section to help users in navigating the guide.
Introduction	
SECTION 2:	This section outlines the purpose of Rule Engine, it supplies an overview of the functionality provided in the Rule Engine. It also includes description for Creation of rule.
Introduction to Rule Engine	



Introduction to ProAppDesigner Rule Engine

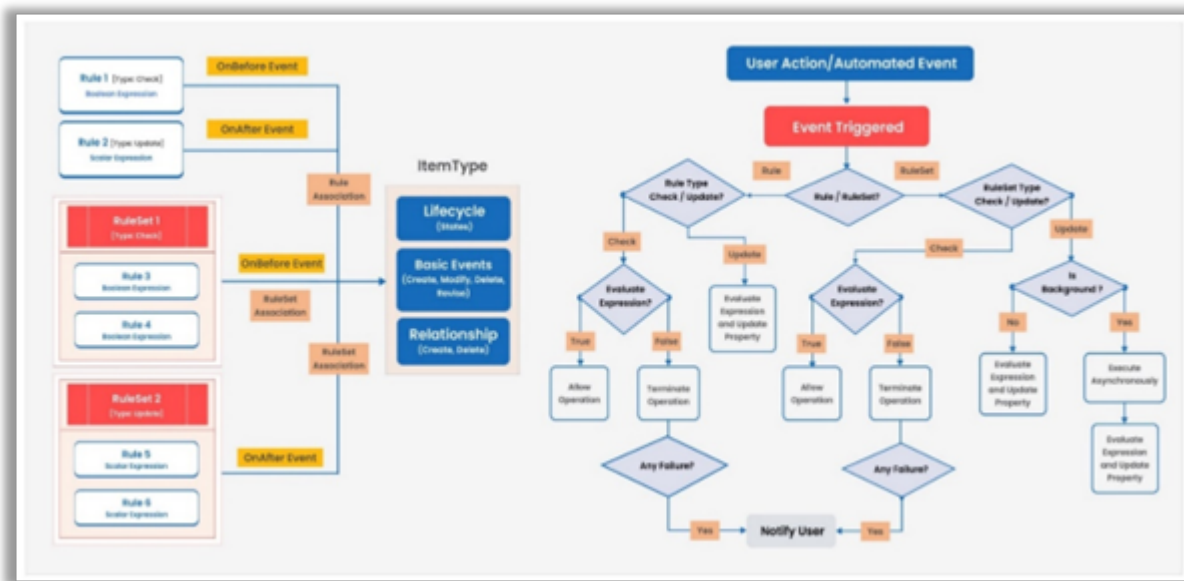
ProAppDesigner Rule Engine is a No-Code Platform that enables business users to define logic within rules, which are triggered by specific server events.

Below are key elements:

1. Rule
2. Rule Validation/Simulation
3. RuleSet
4. Associating Rule/RuleSet with ItemType Events
5. Rule Execution

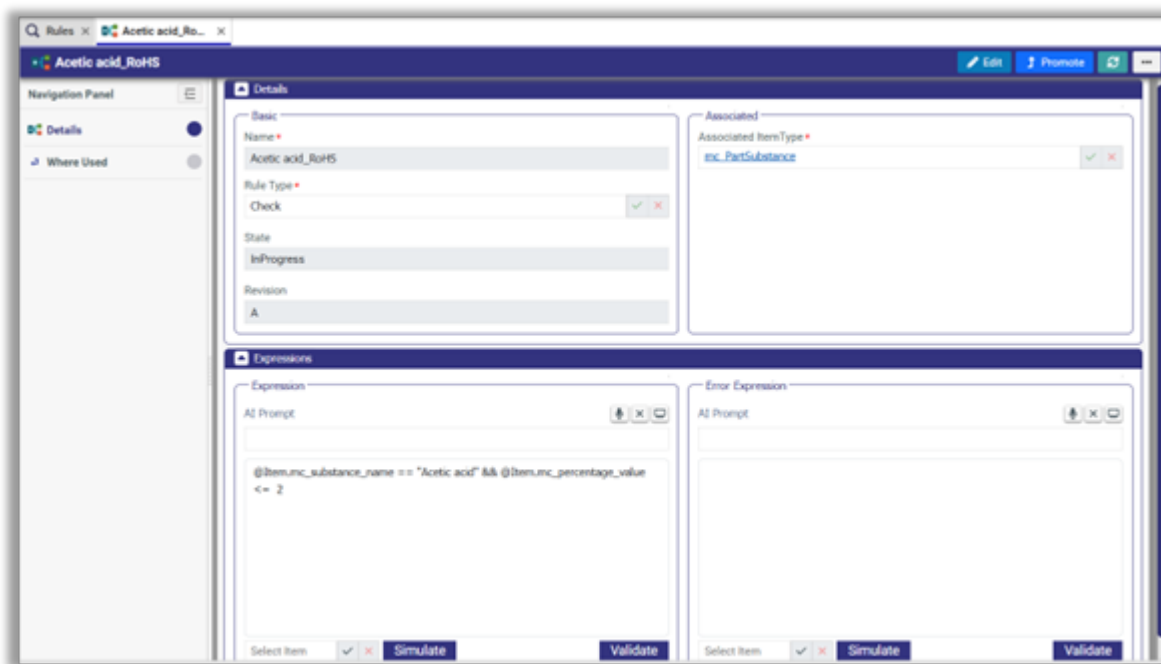


Schematic Diagram



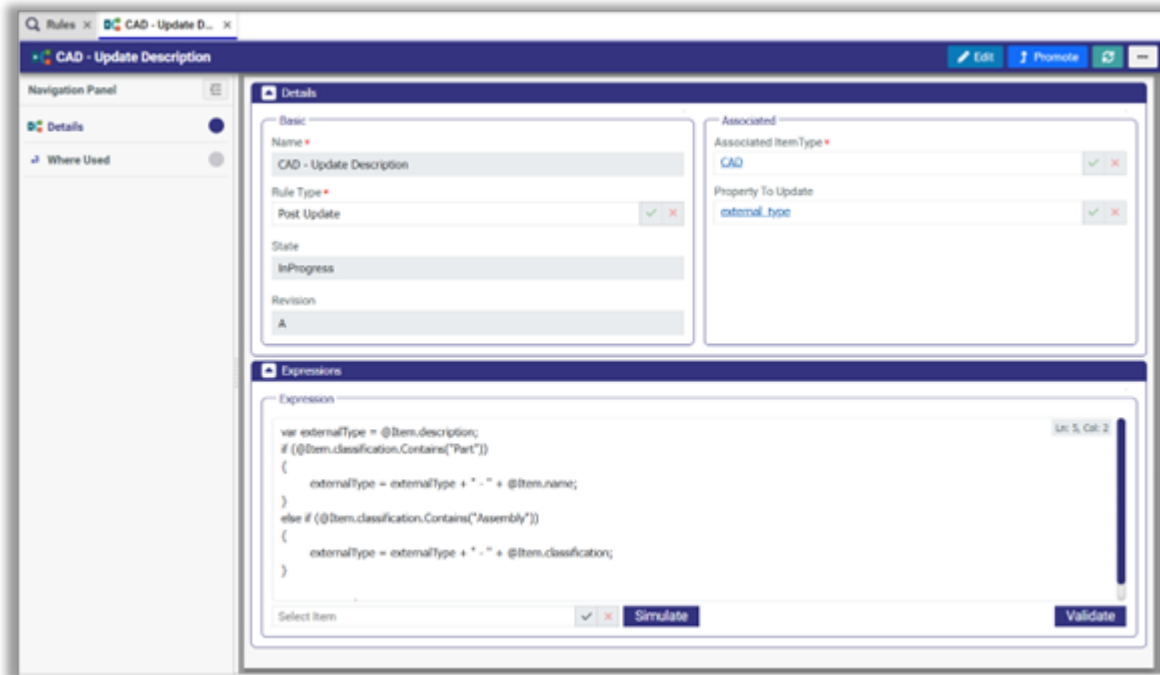
Rule

1. Associated ItemType: Name of ItemType for which this rule is defined
2. Rule Type
 - Check – To perform any kind of validations before any action. {create/edit/delete/revision/promote/relationship(add/delete)}
 - Pre-Update – To evaluate the rule expression and update the context item with the result of rule expression.
 - Post Update – To update the properties of the context item with the result of rule execution.
3. Property to Update
 - In case of Pre/Post Update rule type, select the property of associated ItemType which needs to be updated.
4. Expression
 - Capability to write any C# compatible code inside the expression. It can be any complex code or a mathematical expression.
 - Supports Item Properties, User Properties & Aras Server Methods.
5. Error Expression
 - In case of Check rule type, the error message to be shown to user. This can be an expression itself.
6. Check Rule



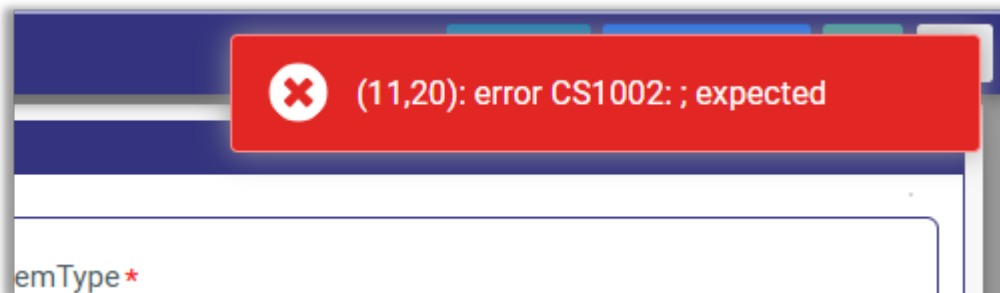
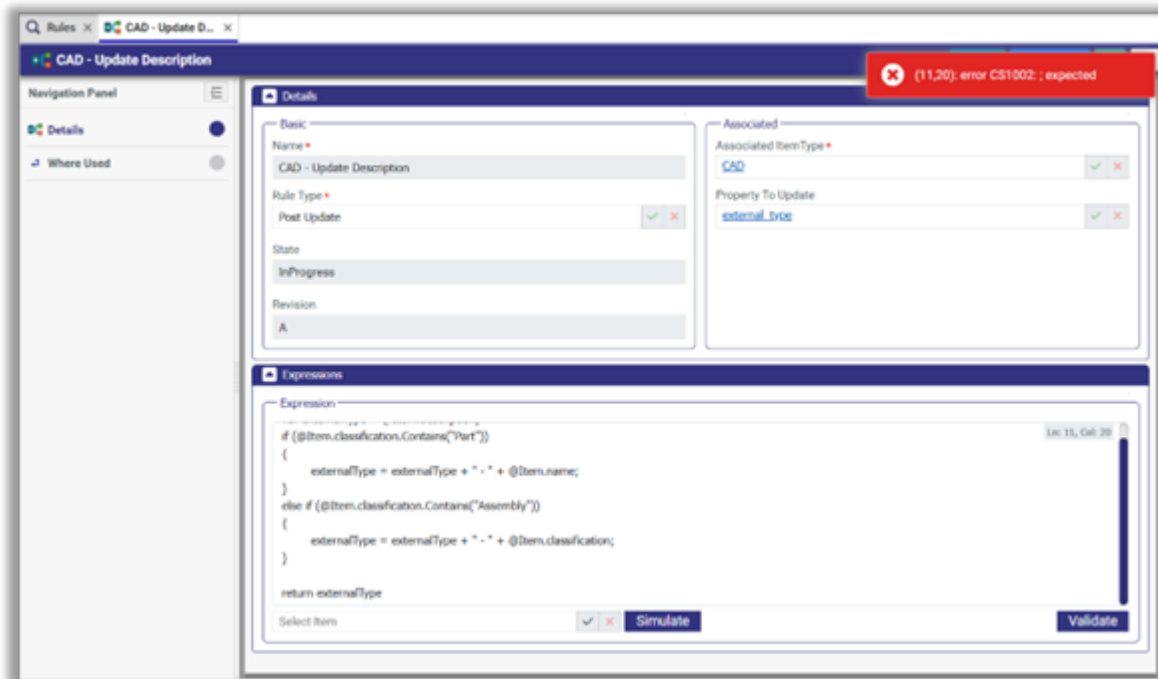
7. Update Rule





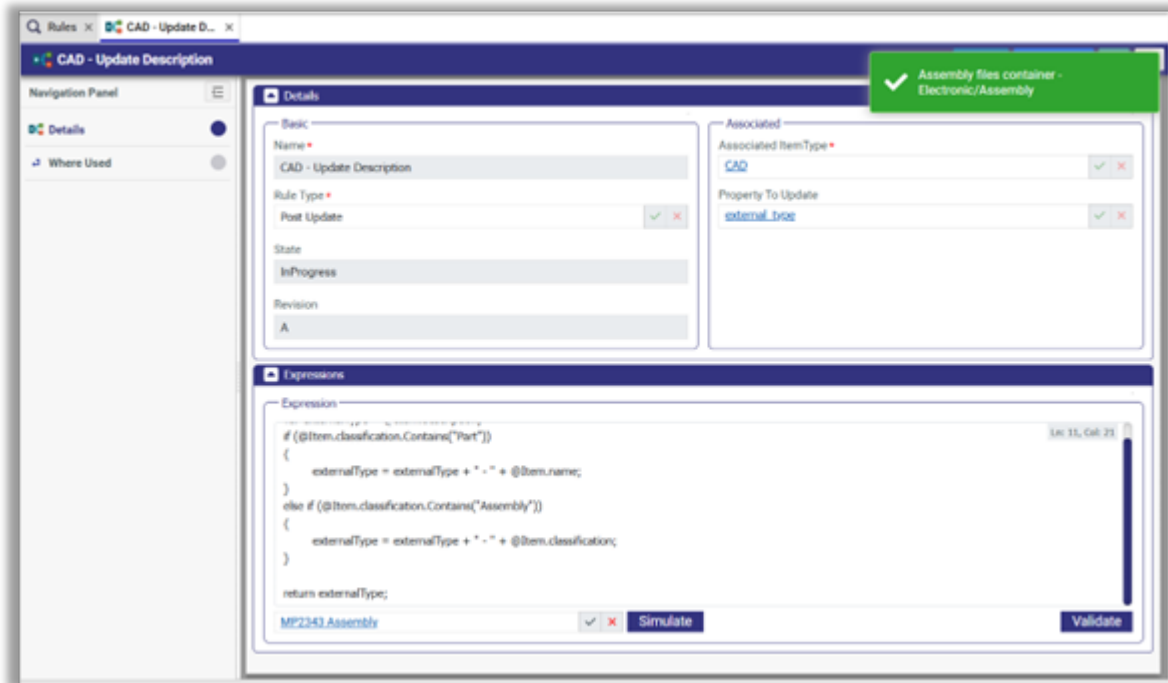
Rule Validation

- Rule Validation is available to user to check if expression is correct. It will check for any syntax errors, property and method validation.
- If there are any errors, user will be presented with error message.



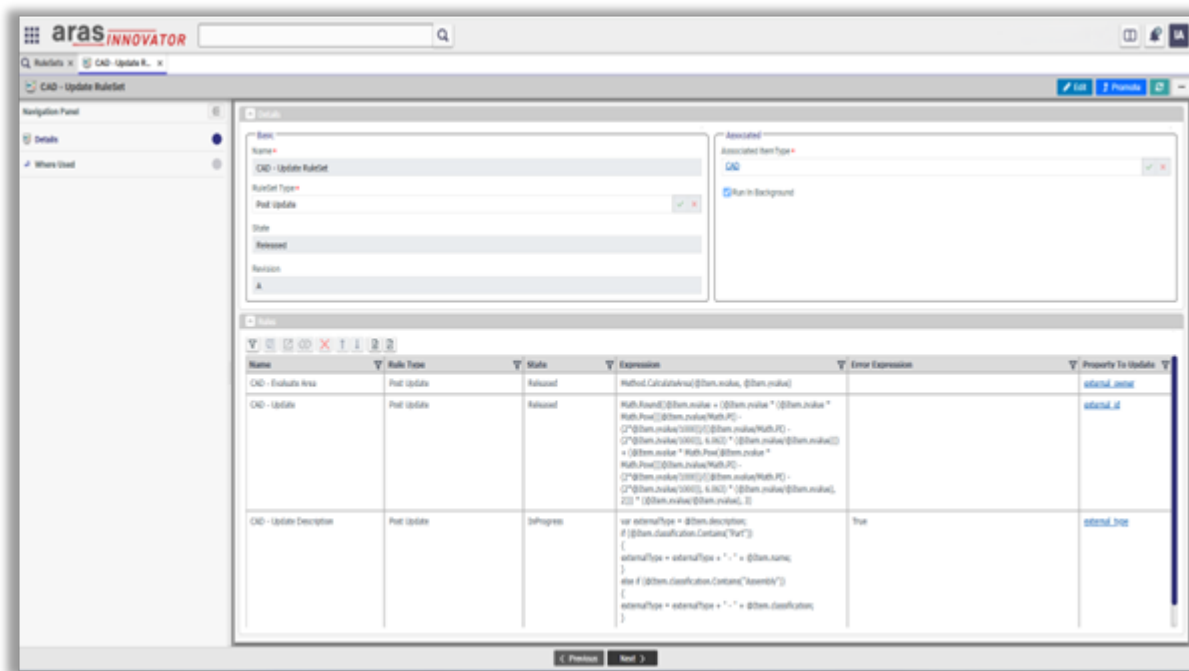
Rule Simulation

Rule Simulation is available to user to check if expression is returning desired value.



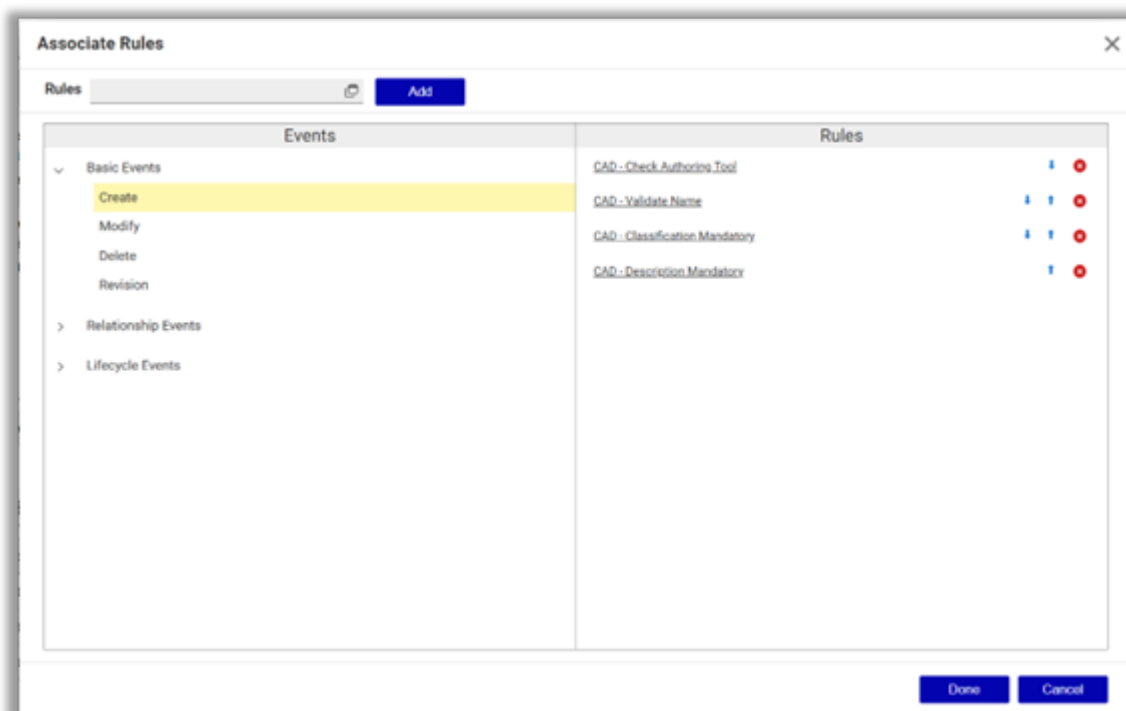
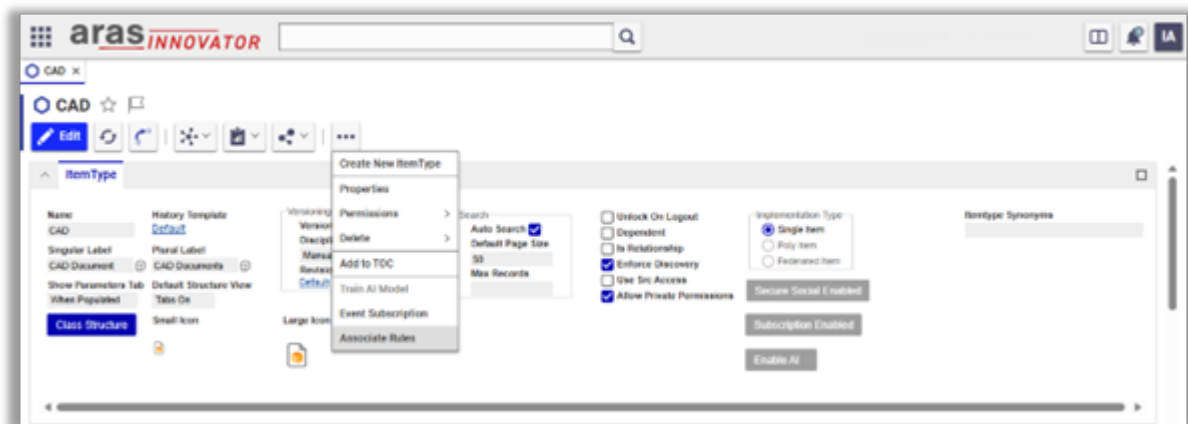
RuleSet

1. It is collection of Rules
2. Associated ItemType: Name of ItemType for which this ruleset is defined
3. Rule Type
 - o Check – To perform any kind of validations before any action {create/edit/delete/revision/promote/relationship(add/delete)}
 - o Post Update – To update the properties of the context item with the result of rule execution.
4. Run In Background
 - o In case of Post Update ruleset type, execution of rules will be done asynchronously using background job. User will be notified upon completion of job.



Associate Rule/RuleSet

- “Associate Rules” action is available on ItemType
- User can select Rules/RuleSets which are defined for that particular itemtype and are in Released state



- Available Events
 - Basic Events

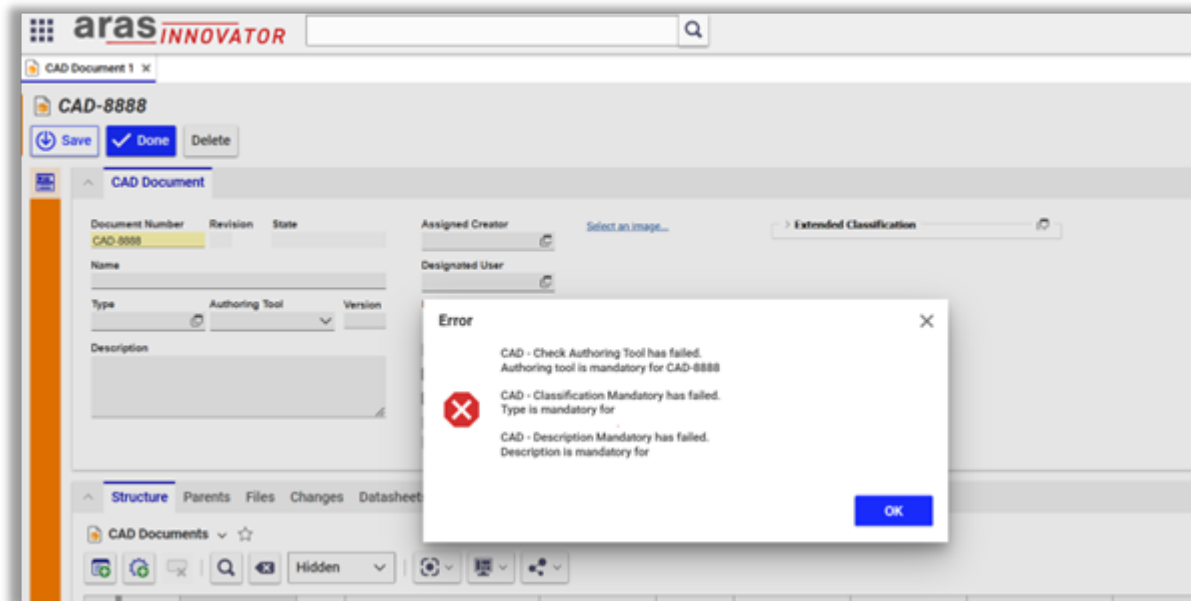
- Create
- Modify
- Delete
- Revision
- Lifecycle Events – Promotion on Lifecycle state

- Relationship Events – Add/Delete of relationship items



Rule Execution

- Once the rules/rulesets are associated with particular server events, framework takes care of executing those rules on those server events.
- In case of failure user will be notified.



Program Management Extension - Installation guide



Introduction

Purpose

This installation guide provides the required information for successfully installing and configuring Program Management Extension.



Scope

The Installation Guide explains the installation process for Program Management Extension for Aras Innovator. The installation involves updates to both the code tree and database.



Target Audience

This Installation Guide is intended for Aras Innovator administrators.



Prerequisites

To install Program Management Extension, users need to confirm that the installation of Aras Innovator meets the minimum requirements to run Program Management Extension.



Checking for Eligibility

The following steps outline the process to verify the code tree and database are eligible for installing the Program Management Extension:

From the **TOC**, select **Administration, Variables**.

VersionMajor	VersionMinor	VersionServiceUpdate
14	0	8
14	0	9
14	0	10
14	0	11
14	0	12

If it does not meet the service pack requirements listed in the previous steps, contact Aras to discuss options.



Installing Program Management Extension

Program Management Extension must modify the Aras Innovator database with the metadata required to run the application. Before applying these changes, verify the following requirements:

Requirements:

- Aras ProAppDesigner
- Aras Program Management 14
- Aras Update Tool 1.17+ or
- Aras Innovator Package Import Export Utilities



Notification and Backup

Note: It is best to give at least 24-hour notice, as well as a reminder 15 minutes prior to the upgrade.

1.

Notify users that the system will be down at a scheduled time, and they should log out of the system prior to the start of the process.













2. Backup the code tree.

The “Code Tree” refers to files and folders installed to the disk when Aras Innovator was first installed.

The default path for the Code Tree installation would be something like:

C:\Program Files (x86)\Aras\Innovator

With the following contents:

Name	Type	Size
 AgentService	File folder	
 ConversionServer	File folder	
 DB	File folder	
 Innovator	File folder	
 OAuthServer	File folder	
 SelfServiceReporting	File folder	
 Tools	File folder	
 VaultServer	File folder	
 ConversionServerConfig	XML File	1 KB
 InnovatorServerConfig	XML File	2 KB
 SelfServiceReportConfig	XML File	1 KB
 VaultServerConfig	XML File	1 KB

The following steps can be used to confirm the installation folder:

- o From the Windows search field, enter **Control** to access the **Control Panel**
- o Select **Programs and Features**
- o Right-click the **Name** column and select **More...**
 - Select the **Comments** header if it is not checked
 - Click **OK**
- o Search for the **Aras Innovator** entry in the **Programs and Features** window.
- o View the value set in the **Comments** column for the Aras Innovator entry:

Name	Publisher	Installed On	Size	Version	Comments
Aras Innovator	Aras Corporation	1/25/2022	1.09 GB	14.0.0	Installed to C:\Program Files (x86)\Aras\I...
Aras Innovator	Aras Corporation	4/12/2023	0.99 GB	14.0.17	Installed to C:\Aras\14.0.17\
Aras Innovator	Aras Corporation	3/7/2022	899 MB	12.0.0	Installed to C:\Program Files (x86)\Aras\I...

After the tree code installation path has been verified, back up the folder and all its contents.

3. Disconnect all users from the database.

The easiest way to prevent client sessions from committing any further changes to the database is to change the database connection string in the `InnovatorServerConfig.xml` from

`<DB-Connection ...` to `<xDB-Connection` and restart the w3svc service (IIS). This expires all session and prevents all new connections to the Aras Innovator database through the existing instance.

4. Backup the database.

5. Place these files in a safe location, as they will be needed to restore if the process fails.

6. Enable database connections.

After the database backup has been completed, enable the database connection string in the

`InnovatorServerConfig.xml` by changing it from `<xDB-Connection ...` to `<DB-Connection` and restarting the w3svc service (IIS).

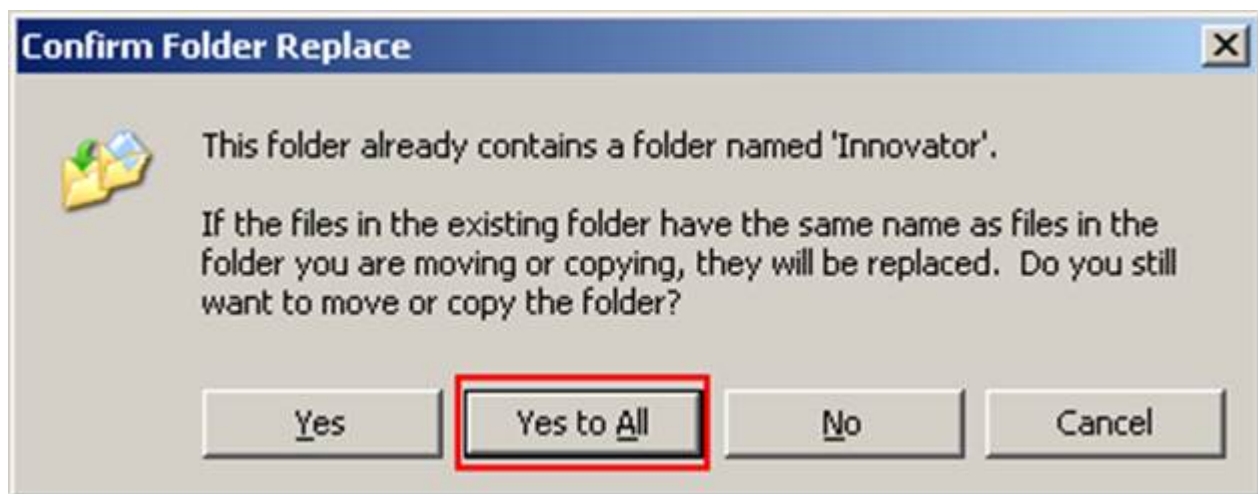
Installing the Program Management Extension

The Program Management Extension can be installed using the Manual Installation option (using the Aras Import Tool).

Manual Installation Option

1. Download the Aras ProAppDesigner CD Image from the Aras FTP site, in the Aras ProAppDesigner Releases folder, and unzip the file on the local computer.
2. Navigate to \AddOns\Program Management Extension folder
3. Copy the \Files\Innovator folder to the root of your install directory, overwriting the existing \Innovator folder and all its contents.

It is recommended that this step be performed by an administrator on the server.



4. From the root of your code tree, navigate to \Innovator\Server and edit the `method-config.xml` file.

1. Add below entries as shown:

```

...
<!-- Start: Program Management Extension-->
<name>$(binpath)/Proriqo.Plm.Apps.ProgramManagement.dll</name>
<!-- END: Program Management Extension -->
</ReferencedAssemblies>

```

2. Save the changes to the method-config.xml file and close it

5. From the root of your code tree, navigate to \Innovator\Client and edit the

InnovatorClient.config file.

1. Change the value of the filesRevision attribute. The value should be changed from std to 2, 3, 4, etc

```
<oauth confidSource="0Auth.confid" />  
<cacheInModule moduleEnabled="true" filesRevision="2" />  
<staticContent>  
...
```

2. Save the changes to the InnovatorClient.config file and close it.

6. Restart the World Wide Web Service (IIS).

7. Execute **Pre-Import Scripts**

Before you install Program Management Extension, execute below scripts:

1. Navigate Aras Nash tool using URL <http://localhost/InnovatorServer/Client/scripts/nash.aspx>
2. Login in Nash using **admin** user.



3. In the package go to folder PreImportAMLScripts, open each file in below sequence, copy the content in XML form and click on Submit.

- 0001_PM_TemplateDeploy_PreAML.xml
- 0002_MethodPackageCorrection.xml

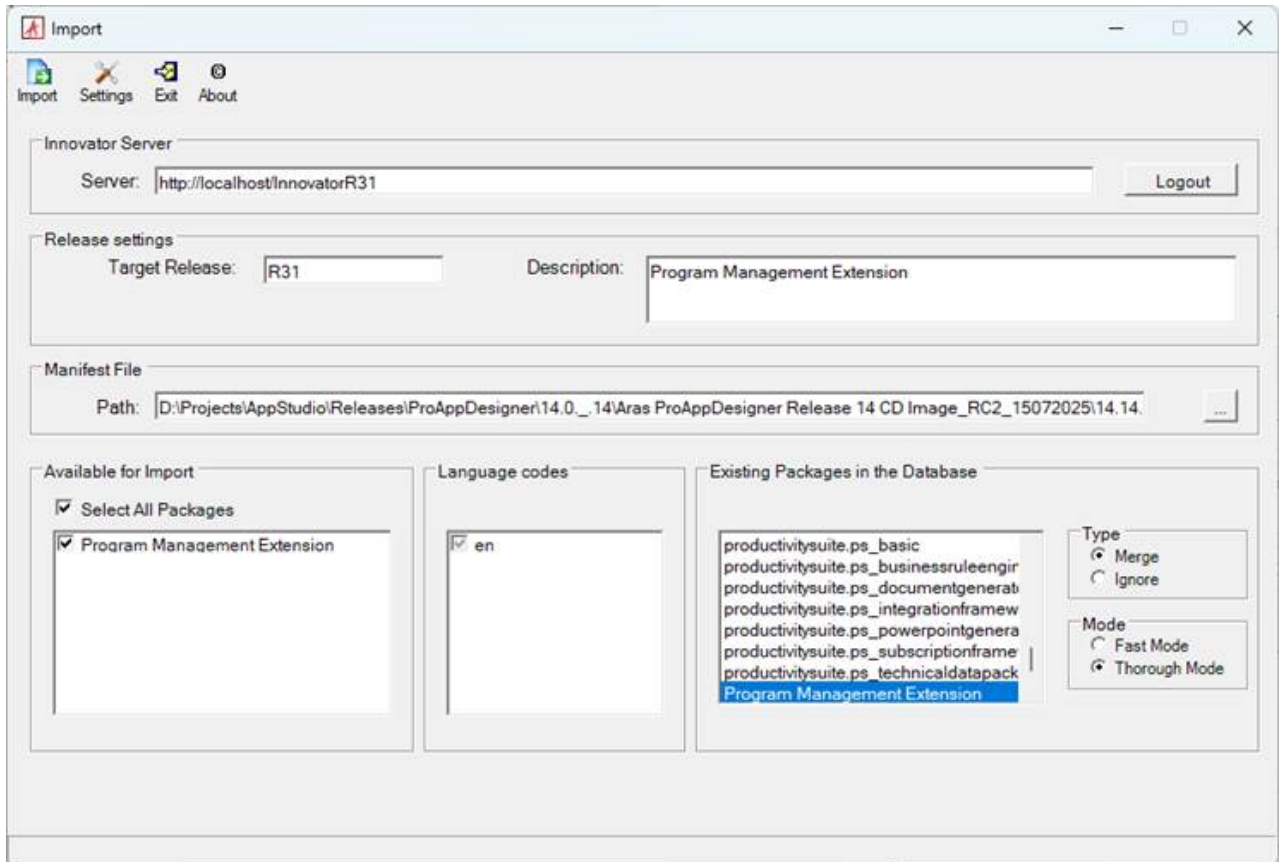
8. Import the **Program Management Extension** database package with the Package Import Export Utilities (For more information using this tool, please see the *Aras Innovator - Package Import Export Utilities* documentation.)

1. You will need to enable the “Super User” login to perform this step.

Browse to the \PackageImportExportUtilities\Import\ folder and run the Import.exe file.

2. Disable the “Super User” login.





3. Enter the following information:

- **Server** = The connection URL for Aras Innovator
- **Database** = The target Aras Innovator database (login screen)
- **Username** = root (login screen)
- **Password** = Password for “root” login (Default is “innovator”) (login screen)
- **Target Release** = Program Management Extension
- **Description** = Program Management Extension
- **Manifest File Path** = The manifest is found in the unzipped patch folder in AddOns\Program Management Extension\Imports\imports.mf
- **Available for Import** = Check all packages
- **Type** = Merge
- **Mode** = Thorough Mode

4. Click the **Import** button.

9. Disable the “Super User” login.

10. Execute **Post Import Scripts**

After you install Aras Program Management Extension, execute below scripts:

1. a. Navigate Aras Nash tool using URL <http://localhost/InnovatorServer/Client/scripts/nash.aspx>
2. Login in Nash using **admin** user.



The screenshot shows the Aras NASH web interface. At the top, the browser address bar shows the URL: localhost/InnovatorServerR27/Client/X-salt=std_14.0.15.38102-X/scripts/nash.aspx. The interface header includes the Aras NASH logo, the database name 'InnovatorR27_New', and the user 'admin'. Below the header, there are configuration options for TimeZone (Eastern Standard Time) and Culture (en-US - English (United States)). The Action is set to 'ApplyAML' and the Server is 'http://localhost/InnovatorServerR27/Server/InnovatorServer.aspx'. There are 'Submit' and 'Clear' buttons next to the XML input field. Below the XML field, the Run time is 0,000 seconds and the Result is 'read'. A 'ShowXml' button is also present. At the bottom, a table lists installed Aras Innovator versions:

Name	Publisher	Installed On	Size	Version	Comments
Aras Innovator	Aras Corporation	1/20/2022	1.09 GB	14.0.0	Installed to C:\Program Files (x86)\Aras\...
Aras Innovator	Aras Corporation	4/12/2023	0.99 GB	14.0.17	Installed to C:\Aras\14.0.17\
Aras Innovator	Aras Corporation	3/7/2022	899 MB	12.0.0	Installed to C:\Program Files (x86)\Aras\...

3. In the package go to folder PostImportAMLScripts, open each file in below sequence, copy the content in XML form and click on Submit.
- 0001_PM_ExecuteSQL.xml
 - 0002_PM_TemplateDeploy_PostAML.xml



Confirming the Installation

Use the following procedure to check if your database has been updated correctly:

1. Log in to Aras Innovator as an administrator.
2. From the TOC, select **Administration** è **Variables**.
3. Confirm the following sets of variables are listed:

Program Management Extension
14.14.22.1

If at any time the installation fails, revert to your backups and contact Aras Support at support@aras.com.

Activating Program Management Extension

You require any one of the below mentioned feature license for Program Management Extension. Once you have received the keys, activate those features from User Menu → Activate Feature.

1. Aras.ProAppDesigner
2. Aras.SupplierManagementAndPortal



Program Management Extension - User Guide



Introduction

Purpose

This guide describes how to use Program Management Extension for external users.



Scope

This document describes functionality of Program Management Extension.



Target Audience

This document is intended for external users with limited access to data.



Feature Terms and Definitions

The following Feature terms and definitions are used in this document:

Term	Definition



Guide Organization

This document describes how to use the Program Management Extension. This User Guide is organized into the following sections:

SECTION 1: Introduction	This section outlines the purpose and scope of this User Guide, as well as its intended target audience. It supplies a brief overview of each section to help users in navigating the guide.
SECTION 2: Introduction to Program Management Extension	This section outlines the purpose of Program Management Extension, it provides an overview of the functionality provided in the Program Management Extension.
SECTION 3: Project View	This section outlines the purpose of Project View in Program Management Extension.
SECTION 4: Project Plan	This section outlines the purpose of Project Plan and various Project Plan Views in Program Management Extension.
SECTION 5: Calendars	This section outlines the purpose of Calendar Management System in Program Management Extension.



Introduction to Program Management Extension

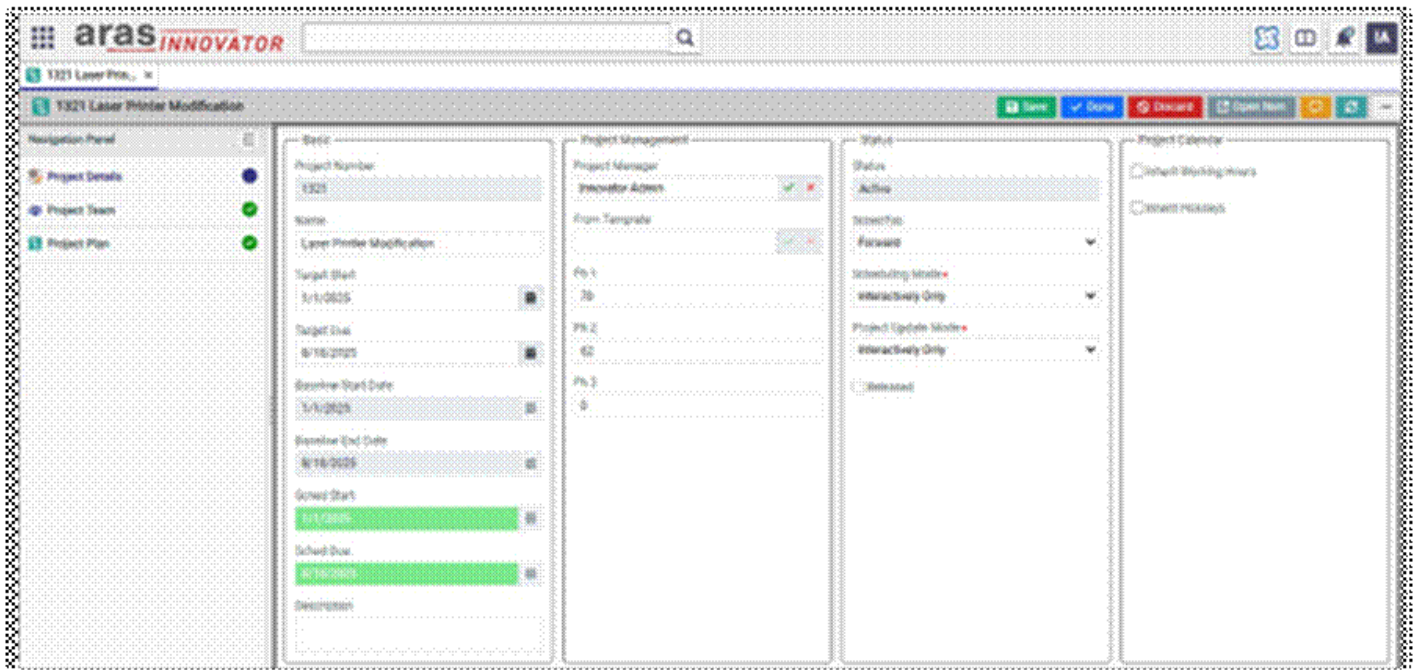
Purpose of Program Management Extension

The Program Management Extension is designed to be installed atop the Program Management module, enriching its capabilities by introducing a suite of advanced UI controls that elevate the project management experience.



Project View

Introduction



When starting a new project, users can choose from pre-existing templates within the Aras Project Template, which they can tailor to their needs and save for future use. The project view is generated using the ProAppDesigner template, offering a structured and organized layout of project information across multiple pages. The Project Details page displays essential project information, neatly grouped into categories for enhanced clarity.

The Project Team page offers a clear and organized snapshot of everyone involved in the project. Team members are listed systematically, each accompanied by their designated role—such as "Project Manager," "Developer," or "Designer"—making it easy to understand their key responsibilities at a glance. Meanwhile, the Project Plan page presents the full project roadmap, detailing tasks along with their dependencies to support effective sequencing and scheduling. The intuitive interface allows smooth navigation between these sections, streamlining project tracking and overall management.



The screenshot shows the ProApp Designer interface. At the top, there are tabs for 'Projects' and '1321 Laser Prin...'. Below the tabs is a toolbar with buttons for 'Save', 'Done', 'Discard', 'Open Item', and a refresh icon. On the left is a 'Navigation Panel' with three items: 'Project Details' (grey circle), 'Project Team' (blue circle), and 'Project Plan' (green circle with a checkmark). The main area contains a table with two columns: 'Name' and 'Role'. The table data is as follows:

Name	Role
Innovator Admin	Manager
Mike Miller	Design
peter smith	Process
Chris Brayn	Quality
Mike Beck	Assemble

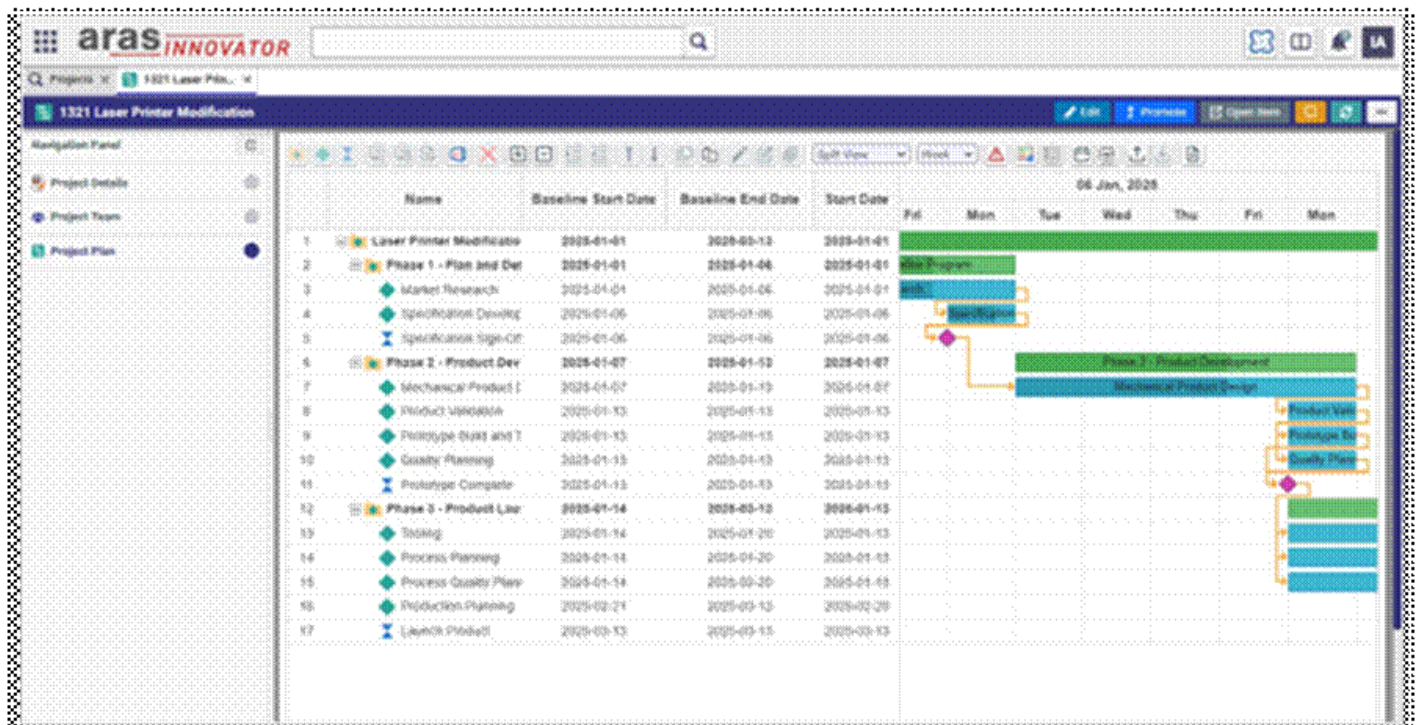


Project Plan

Introduction

The Project Plan utilizes a sophisticated Gantt Chart control featuring multiple interactive views. In the Timeline View, users can directly manage the project by adding dependencies, adjusting activity durations, and reassigning tasks with ease. Meanwhile, the Resource View delivers a clear overview of resource allocation, identifying potential overloads to help maintain a balanced and efficient workload distribution.

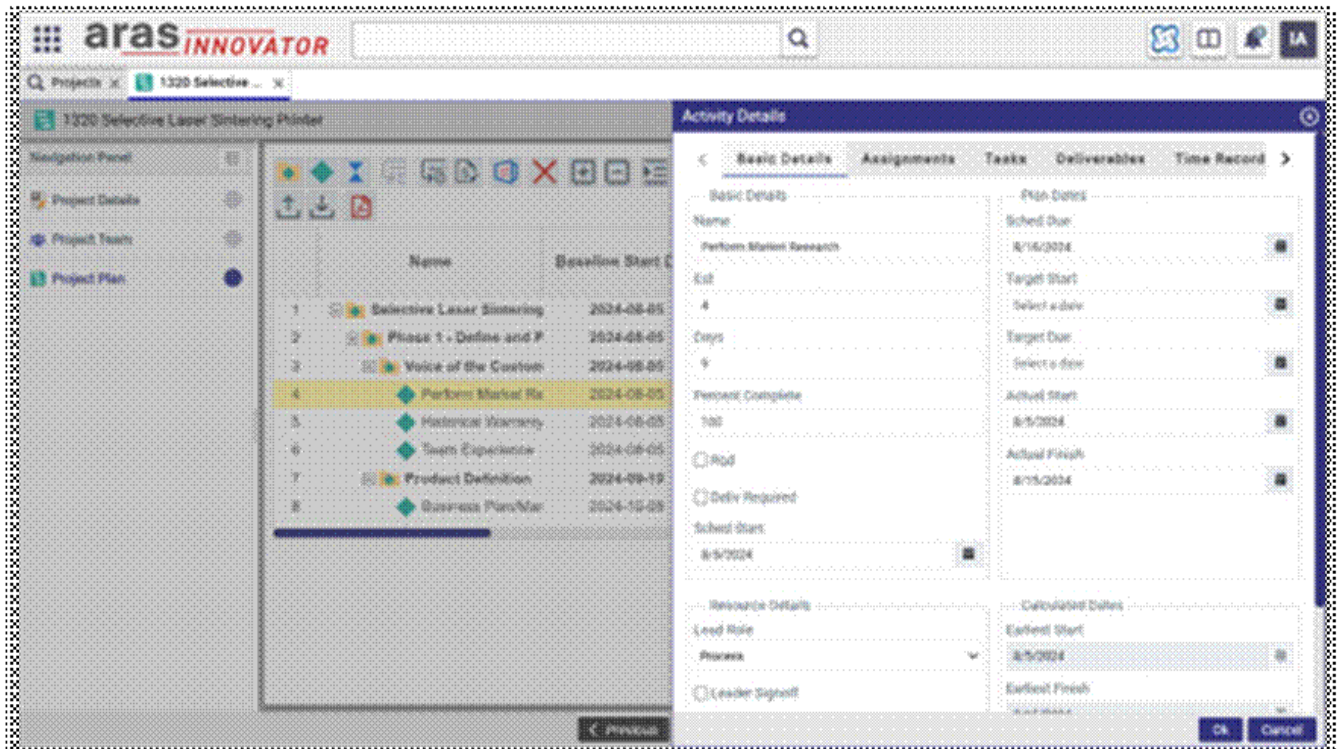
When assigning tasks in the Project Plan, the system intelligently integrates the User Calendar to detect scheduling conflicts, ensuring assignments align with each user's availability and established work schedules. The advanced Gantt Chart control further elevates the experience by providing real-time updates and smooth, intuitive navigation—enhancing project oversight and control. Together, these features empower project managers to maintain a conflict-free, well-coordinated timeline that supports optimal resource utilization and streamlined execution.



- Open the subproject in a new tab, allowing users to work on it independently while maintaining its connection to the main project.
- Change the lifecycle state of the subproject to reflect its progression through various project phases, ensuring accurate tracking and phase alignment within the overall project structure.
- D-Link the subproject from the master project if needed.
- **Update Project:** The system automatically updates the percentage completion of each WBS element based on the progress of its child activities, ensuring a more precise reflection of the overall project status. Additionally, it applies color coding to visually represent task progress, enhancing clarity and making it easier to assess project health at a glance.
- **Delete:** Select this action to delete a task or element from the project structure.
- **Expand All:** Use this action to expand the entire project hierarchy, revealing all levels—including tasks, templates, and subprojects—for a complete and unobstructed view of the project's structure.
- **Collapse All:** Select this action hides all sub-level tasks, activities, and subprojects, showing only the top-level project phases for a simplified view.
- **Indent:** This action moves the selected task or element under another, creating a parent-child relationship within the project hierarchy.
- **Outdent:** The outdent action promotes an indented task or element to a higher level in the hierarchy.
- **Up:** Select up action moves the selected activity, milestone, or element above the preceding item in the list.
- **Down:** Select Down action moves the selected item below the next task, activity, or milestone in sequence.
- **Copy:** Use this action to copy the selected activity or WBS element from the project. Users can select multiple items using multiselect for efficient bulk operations.
- **Paste:** Select Paste action to Inserts the copied content from the clipboard either into or after the selected task or element.

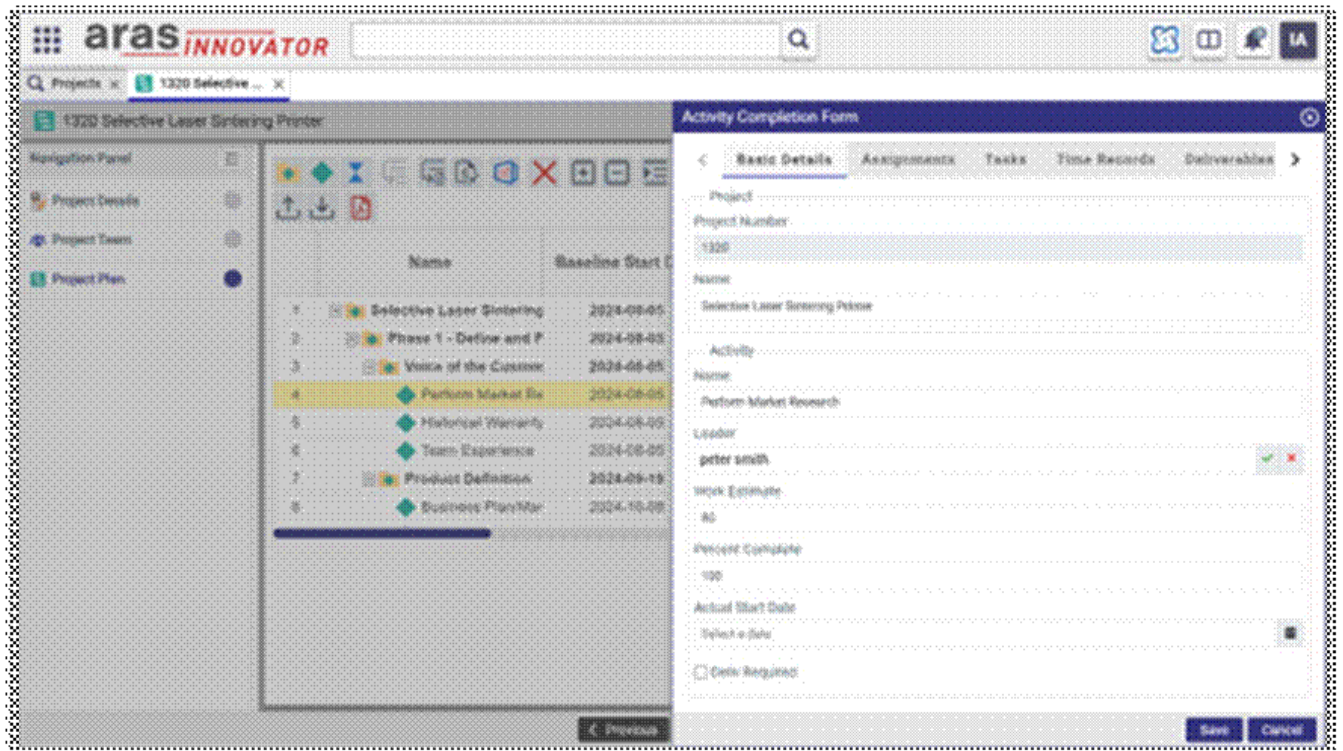
- **Timeline View Options:** Allows users to customize the Gantt Chart view by switching between day, week, month, quarter, or year scales—enhancing schedule visibility.





- **Activity Details:** Select activity details action opens a detailed flyout for the selected task, activity, or milestone, displaying key information such as assignments, deliverables, time records, and comments.
- **Basic Details:** Displays the activity's leader, lead role percentage, scheduling, and duration. These values are editable, making it easy to monitor and adjust project timelines.
- **Assignment:** Allows users to efficiently manage team roles by adding or deleting assignments for the selected activity.
- **Task:** Enables users to add or remove tasks within an activity, supporting flexible and up-to-date task management.
- **Deliverables:** Shows documents attached to the activity. Users can insert or delete files, helping maintain organized and accessible deliverables.
- **Time Record:** Select the Time Record tab to let users add or remove time entries in the time record table—crucial for tracking effort spent on specific tasks.
- **Comment:** Select the Comment tab to attach files and enter comments related to the activity. This feature supports clear documentation and collaboration.
- **Activity Completion Form:** Select the activity completion action to present a read-only summary of key details for the selected element, activity, or milestone. Users can still update the status of assignments and tasks directly from the form, making activity closure streamlined and efficient.



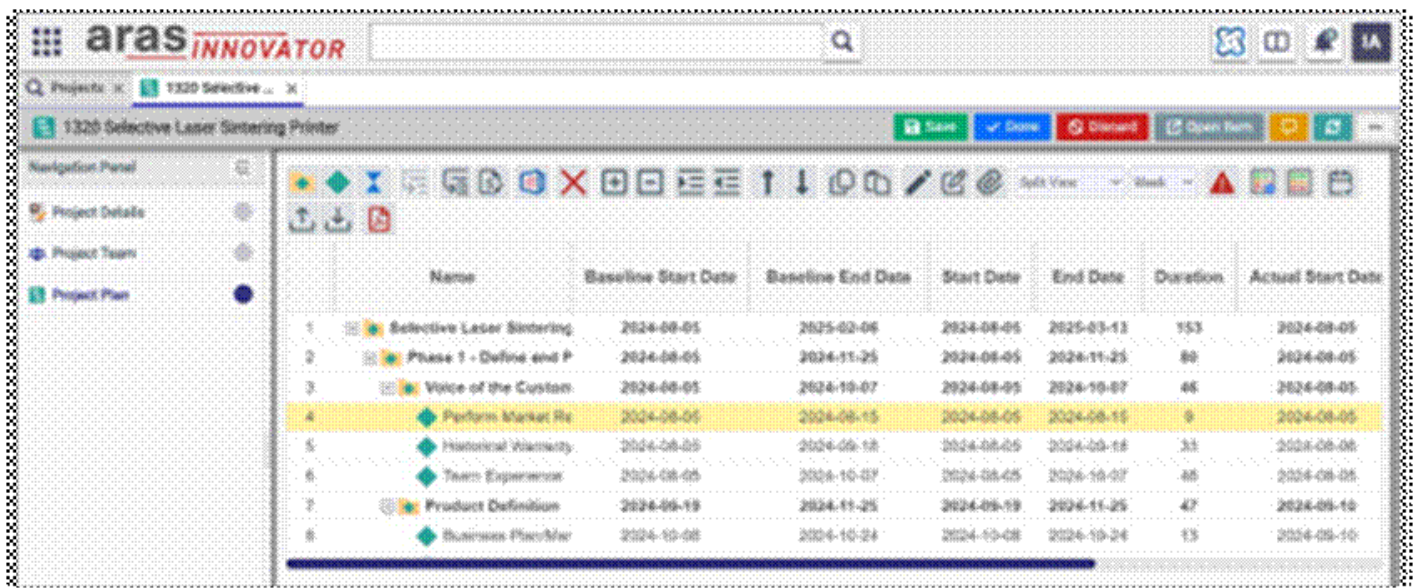


In the Assignments tab, users can mark assignments and tasks as complete and verify their completion status. This functionality supports accurate progress tracking and reinforces accountability across the project team.

- **Add Existing Deliverables:** Select the add deliverables action to Attach a document to an activity by selecting it from the search flyout. Deliverables serve as essential documentation for each activity, supporting project goals and fulfilling stakeholder expectations.
- **Show Critical Path:** Highlights the project's critical path across all views in red. The critical path reflects the sequence of tasks that directly affect the shortest project duration. It is automatically updated when task dates are modified, ensuring visibility into the most current project schedule.
- **Update Baseline:** Use Update Baseline to overwrite baseline start and end dates with the current planned dates for each task, keeping the baseline aligned with the latest approved schedule. Refer to Section [Project Baseline](#) for detailed instructions.
- **Show Deviation:** This action visually identifies differences between baseline and planned task dates using intuitive color coding, helping users quickly spot delays or advancements. See [Project Baseline](#) for more details.
- **Project Calendar:** The Project Calendar defines the overall project's working days, non-working days, and holidays, guiding consistent scheduling throughout all phases. Refer to [Project Calendar](#) for customization details.
- **Teams Calendar:** The **Teams Calendar** presents team members' individual schedules in one view to help manage workloads, avoid conflicts, and track team-wide progress. For information on configuring and managing the Teams Calendar, please refer to [Team Calendar](#).
- **Export PDF:** Select this action to generate a PDF file containing all visible data displayed in the structured layout of the project plan.

Grid View

Selecting Grid View from the toolbar presents a structured, table-based layout for managing project tasks and milestones. Users can manually input or edit essential attributes such as Leader, Lead Role, Status, Duration, and Predecessor for each item. Based on the entered Actual Start Date and Actual End Date, the system automatically updates the corresponding Start Date, End Date, and Duration, ensuring accurate and efficient timeline tracking.



	Name	Baseline Start Date	Baseline End Date	Start Date	End Date	Duration	Actual Start Date
1	Selective Laser Sintering	2024-08-05	2025-02-06	2024-08-05	2025-03-13	153	2024-08-05
2	Phase 1 - Define and P	2024-08-05	2024-11-25	2024-08-05	2024-11-25	80	2024-08-05
3	Voice of the Custom	2024-08-05	2024-10-07	2024-08-05	2024-10-07	46	2024-08-05
4	Perform Market Re	2024-08-05	2024-08-15	2024-08-05	2024-08-15	9	2024-08-05
5	Historical Warranty	2024-08-05	2024-08-18	2024-08-05	2024-08-18	33	2024-08-05
6	Team Experience	2024-08-05	2024-10-07	2024-08-05	2024-10-07	46	2024-08-05
7	Product Definition	2024-08-19	2024-11-25	2024-08-19	2024-11-25	47	2024-08-19
8	Business Plan/Rev	2024-10-08	2024-10-24	2024-10-08	2024-10-24	13	2024-08-10

When a value is entered in the Predecessor field, the system automatically applies a default Finish-to-Start (FS) dependency. These streamlines scheduling logic by eliminating the need for manual configuration of common task relationships.

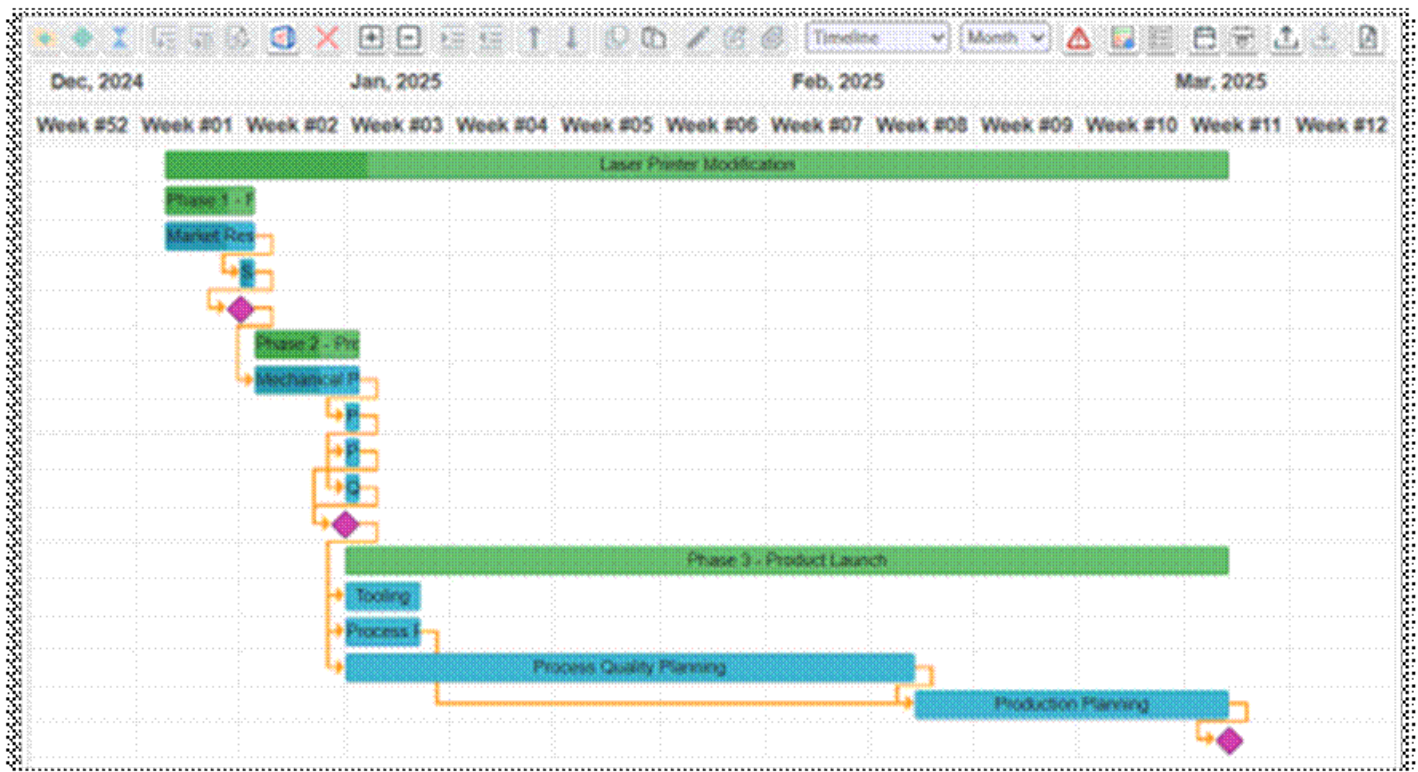
Additionally, The Grid View offers flexible customization options to enhance usability and data management. By right-clicking any column header, users can choose to show or hide specific columns based on their preferences. Additionally, columns can be dragged and reordered, allowing users to tailor the layout to suit their workflow and improve efficiency.

	Name	Baseline Start Date	Baseline End Date
1	Hide Name	2025-01-01	2025-03-13
2		2025-01-01	2025-01-06
3	Show Columns	<input checked="" type="checkbox"/> Name	2025-01-06
4	Specification Device	<input checked="" type="checkbox"/> Baseline Start Date	2025-01-06
5	Specification Sign-Off		2025-01-06
6	Phase 2 - Product Del	<input checked="" type="checkbox"/> Baseline End Date	2025-01-13
7	Mechanical Product	<input checked="" type="checkbox"/> Start Date	2025-01-13
8	Product Validation		2025-01-13
9	Prototype Build and	<input checked="" type="checkbox"/> End Date	2025-01-13
10	Quality Planning	<input checked="" type="checkbox"/> Duration	2025-01-13
11	Prototype Complete		2025-01-13
12	Phase 3 - Product Las	<input checked="" type="checkbox"/> Actual Start Date	2025-03-13
13	Tooling	<input checked="" type="checkbox"/> Actual End Date	2025-01-20
14	Process Planning	<input checked="" type="checkbox"/> Status	2025-01-20
15	Process Quality Plan		2025-02-20
16	Production Planning	<input checked="" type="checkbox"/> Predecessors	2025-03-13
17	Launch Product	<input checked="" type="checkbox"/> Leader	2025-03-13
		<input checked="" type="checkbox"/> Lead Role	
		<input checked="" type="checkbox"/> Attach	



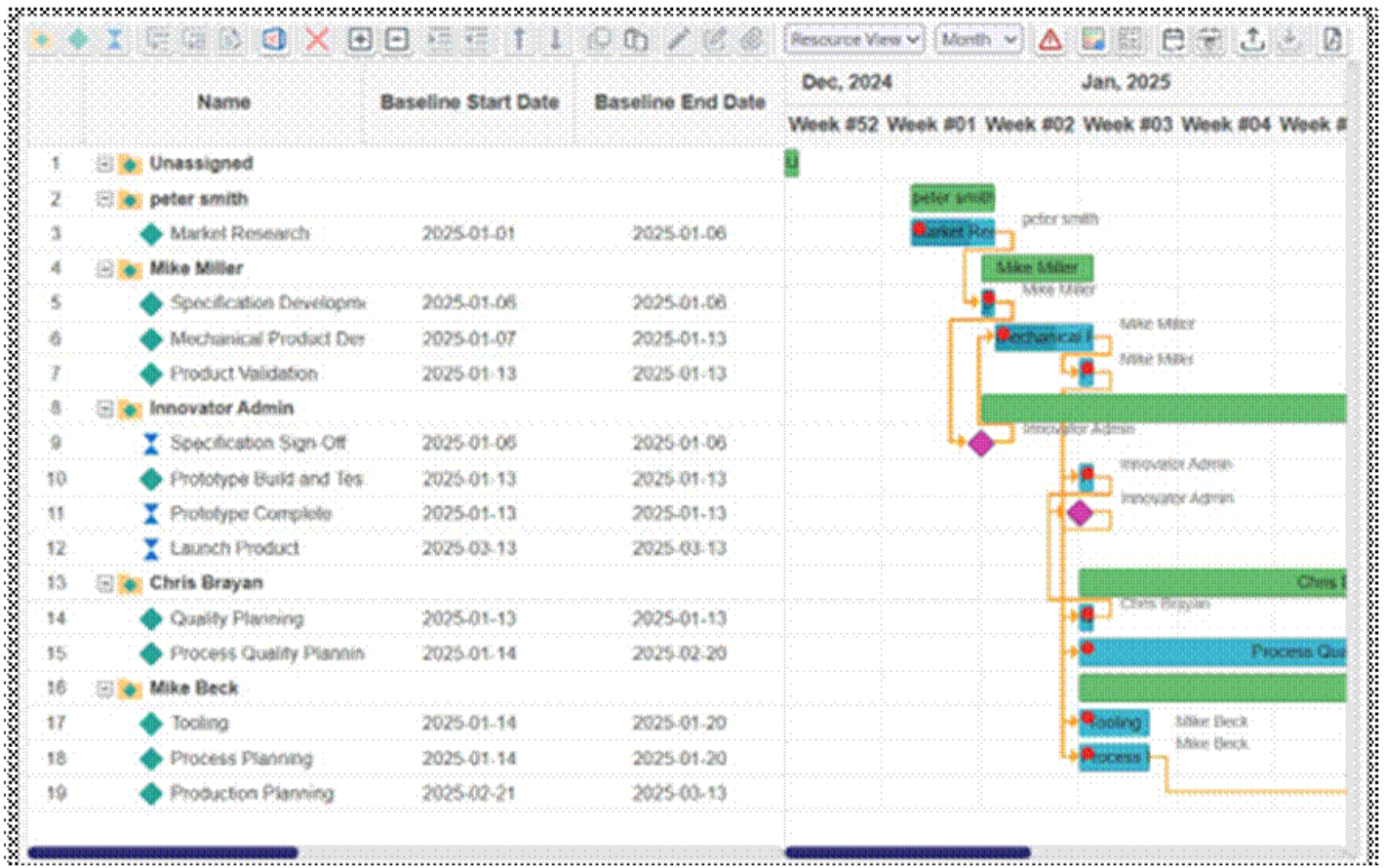
Timeline View

Select Timeline View from the toolbar to display the project timeline. In this view, WBS Elements appear in green, Activities in blue, and Milestones in pink, offering clear visual differentiation. Activity bars show percentage completion in a darker shade for quick progress tracking. Users can add Predecessors by connecting activity nodes directly within the timeline. Activity durations can be modified by dragging the respective bars. If an assignment override occurs, a red circle icon appears atop the activity; hovering over the icon reveals a warning message for immediate attention.



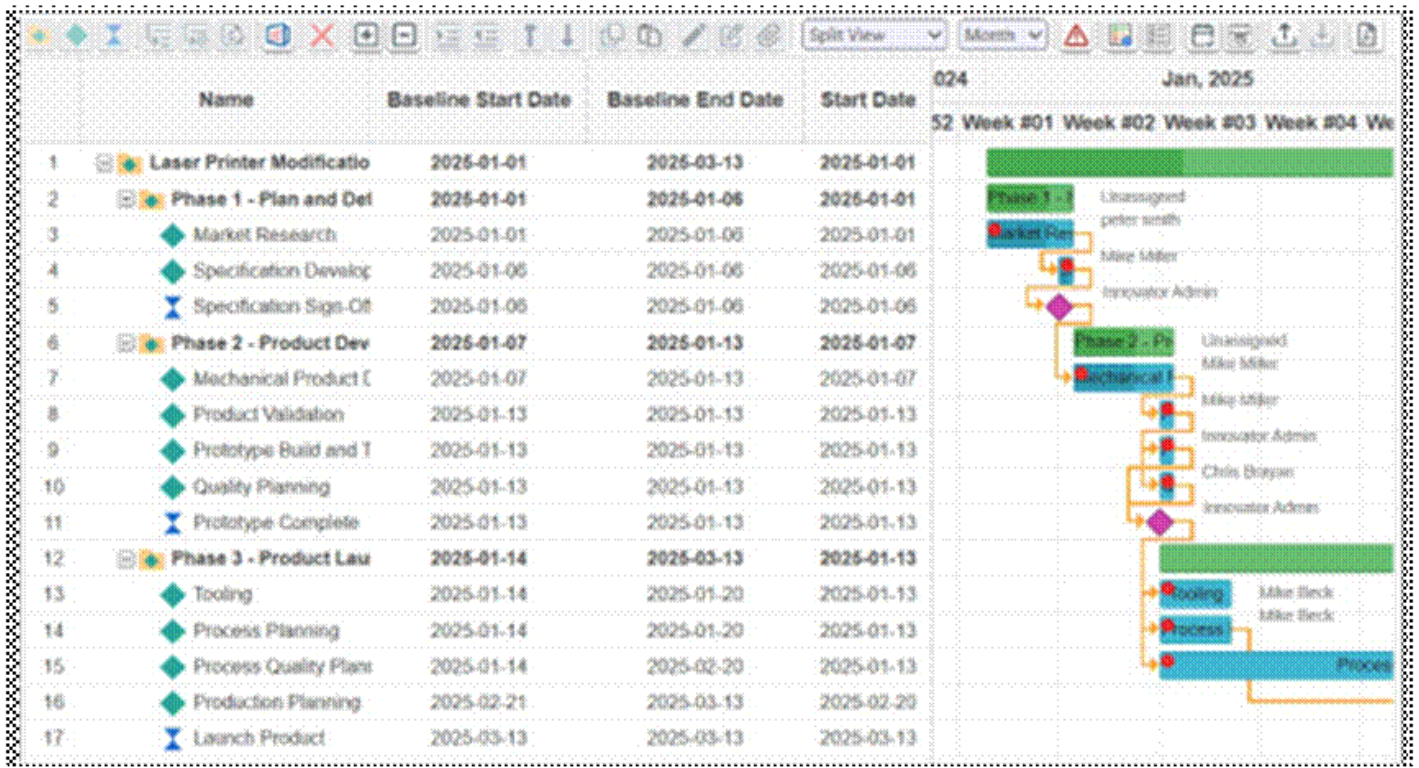
Resource View

Combines elements of Grid View and Split View while presenting a resource-centric layout. Activities are organized beneath their assigned leaders, clearly displaying each team member's workload. If an activity lacks an assigned leader, it appears under the Unassigned category. When a user is allocated multiple tasks, a warning message is displayed upon opening the Resource View, alerting project managers to potential overloads and promoting balanced task distribution.



Split View

Combines the Grid View on the left with the Timeline View on the right, offering a unified interface for comprehensive task management. Users can add key attributes—such as Leader, Lead Role, Status, Duration, and Predecessor—to activities or milestones. The Deliverables button enables users to insert supporting documents into project elements. When users input Actual Start and Actual End dates, the system automatically updates the corresponding planned Start and End dates, ensuring accurate timeline alignment.



Master and Subprojects

Complex projects can be structured using a master project and its associated subprojects, mirroring traditional hierarchical project planning. This modular approach breaks large initiatives into manageable sections while preserving overall cohesion.

- **Insert Subproject:** Use the Insert Subproject action to add one or more subprojects to the main project. Subprojects can be selected from the search flyout if they have already been created.
- **Manage Subproject:** - After insertion, the Subproject action becomes available—allowing users to update the subproject's state, remove it from the master, or open it in a separate tab for focused editing.

Master Projects: A master project serves as the top-level container for coordinating multiple subprojects, offering a unified, high-level view of progress and dependencies.

Subprojects: Subprojects are independently managed standalone projects linked to a master project. They allow teams to concentrate on specific deliverables while maintaining alignment with the overarching project objectives.

The screenshot displays the Aras Innovator project management interface. The main window shows a project plan for '1320 Selective Laser Smoothing Printer' with a list of tasks and their dates. A 'Manage Subprojects' dialog box is open on the right, showing a table of subprojects.

Subproject Name	Plan Start	Plan Finish	Status	Project Manager
Laser Printer Modification	2023-01-01	2023-03-31	Active	INNOVATOR Admin

The main project plan table includes the following data:

ID	Name	Baseline Start Date	Baseline End Date	Start
1	Selective Laser Smoothing	2024-08-05	2025-02-05	2024
2	Phase 1 - Define and P	2024-08-05	2024-11-25	2024
3	Voice of the Customer	2024-08-05	2024-10-07	2024
4	Perform Market Re	2024-08-05	2024-08-15	2024
5	Historical Win/Loss	2024-08-05	2024-09-18	2024
6	Team Experience	2024-08-05	2024-10-07	2024
7	Product Definition	2024-08-19	2024-11-25	2024
8	Business Plan/Mar	2024-10-01	2024-10-24	2024
9	ProductProcess B	2024-10-25	2024-10-31	2024
10	ProductProcess A	2024-10-25	2024-10-31	2024
11	Product Reliability	2024-09-19	2024-09-27	2024
12	Customer Input	2024-10-08	2024-10-08	2024
13	Design Costs	2024-11-01	2024-11-08	2024
14	Reliability and Cos	2024-11-05	2024-11-07	2024
15	Preliminary Bill of	2024-11-08	2024-11-14	2024
16	Preliminary Procus	2024-11-19	2024-11-19	2024
17	Preliminary Listin	2024-11-19	2024-11-20	2024
18	Product Assurance	2024-11-21	2024-11-22	2024



Users can perform all toolbar actions on subprojects, ensuring consistent functionality across all project levels. In the Timeline View, subprojects are visually highlighted in yellow, making them easy to distinguish within a complex project structure. Additionally, the system supports nested subprojects, allowing users to insert one subproject within another.

Project Baseline

A baseline captures the original planned schedule, tasks, and milestones of a project, serving as a benchmark for measuring actual progress. Once the project is created, users can select the Update Baseline action to refresh the Baseline Start Date and Baseline End Date based on the current plan.

Show Deviation Action: The "Show Deviation" action compares baseline dates with actual task dates and highlights schedule variances using intuitive color codes on start and end dates:

- **Red highlighted:** Task is significantly delayed beyond the planned schedule.
- **Orange highlighted:** Task is slightly delayed beyond the planned schedule.
- **Green highlighted:** Task is on track or completed within the planned timeframe.

	Name	Baseline Start Date	Baseline End Date	Start Date	End Date	Duration	Actual Start Date
1	Laserc Printer Modificatio	2025-01-01	2025-03-13	2025-01-01	2025-03-13	51	2024-09-11
2	Phase 1 - Plan and Def	2025-01-01	2025-01-06	2025-01-01	2025-01-06	4	2024-09-11
3	Market Research	2025-01-01	2025-01-06	2025-01-01	2025-01-06	4	2024-09-11
4	Specification Develop	2025-01-06	2025-01-06	2025-01-06	2025-01-06	1	2024-09-15
5	Specification Sign-Off	2025-01-06	2025-01-06	2025-01-06	2025-01-06	0	2024-09-23
6	Phase 2 - Product Dev	2025-01-07	2025-01-13	2025-01-07	2025-01-13	5	2024-10-01
7	Mechanical Product E	2025-01-07	2025-01-13	2025-01-07	2025-01-13	5	2024-10-01
8	Product Validation	2025-01-13	2025-01-13	2025-01-13	2025-01-13	1	
9	Prototype Build and T	2025-01-13	2025-01-13	2025-01-13	2025-01-13	1	
10	Quality Planning	2025-01-13	2025-01-13	2025-01-13	2025-01-13	1	
11	Prototype Complete	2025-01-13	2025-01-13	2025-01-13	2025-01-13	0	
12	Phase 3 - Product Laur	2025-01-14	2025-03-13	2025-01-13	2025-03-13	43	
13	Tooling	2025-01-14	2025-01-20	2025-01-13	2025-01-17	5	
14	Process Planning	2025-01-14	2025-01-20	2025-01-13	2025-01-17	5	
15	Process Quality Plans	2025-01-14	2025-02-20	2025-01-13	2025-02-19	28	
16	Production Planning	2025-02-21	2025-03-13	2025-02-20	2025-03-12	15	
17	Launch Product	2025-03-13	2025-03-13	2025-03-13	2025-03-13	0	

This visual representation enables quick identification of scheduling issues, empowering teams to take timely corrective actions.

Import Project Template

Project templates containing standard activities can be imported directly into the project plan. Once imported, the activities become fully integrated and editable as part of the current project.

Steps to import template:

- Select WBS element where user want to insert template.
- Click on Insert Activities action from the toolbar.
- In the search flyout, choose an existing template and click OK.

After import, users can perform all available actions on the inserted activities, just as they would with manually created tasks.

The screenshot displays the Aras Innovator software interface. The main window shows a project plan for '1922 Primer M...'. The project plan is organized into phases: 'Printer Modification', 'Phase 1 - Plan and Def', and 'Phase 2 - Product Dev'. A 'Select Activity' dialog box is open on the right side of the screen, showing a search for 'Approved' templates. The dialog box contains a table with the following data:

Name	Status
APQP	Approved
PPAP	Approved
Workshop Product Development	Approved

The main project plan table has the following columns: Name, Baseline Start Date, Baseline End Date, Start Date, and End Date. The data in the table is as follows:

ID	Name	Baseline Start Date	Baseline End Date	Start Date	End Date
1	Printer Modification	2024-05-21	2024-08-27	2024-05-21	2024-07-14
2	Phase 1 - Plan and Def	2024-05-21	2024-05-21	2024-05-21	2024-05-29
3	Market Research	2024-05-21	2024-05-21	2024-05-21	2024-05-21
4	Specification Create	2024-05-28	2024-05-21	2024-05-28	2024-05-28
5	Specification Sign-Off	2024-05-31	2024-05-31	2024-05-28	2024-05-28
6	Phase 2 - Product Dev	2024-06-03	2024-07-18	2024-05-28	2024-07-09
7	Mechanical Product I	2024-06-03	2024-06-14	2024-06-30	2024-06-11
8	Product Validation	2024-06-17	2024-06-18	2024-06-12	2024-06-12
9	Process Build and T	2024-06-19	2024-06-20	2024-06-12	2024-06-15
10	Quality Planning	2024-06-05	2024-07-09	2024-06-16	2024-07-08
11	Prototype Complete	2024-07-16	2024-07-16	2024-07-06	2024-07-06
12	Phase 3 - Prestart L	2024-07-17	2024-08-27	2024-07-06	2024-07-18
13	Testing	2024-07-17	2024-07-20	2024-07-06	2024-07-18
14	Process Planning	2024-07-17	2024-07-20	2024-07-06	2024-07-18
15	Process Quality Plan	2024-07-17	2024-07-20	2024-07-06	2024-07-18



The screenshot displays the Aras Innovator software interface for a project named "1321 Laser Printer Modification". The main window shows a project plan with a table of tasks and a Gantt chart. A modal window titled "Import MPP" is open, showing a file named "1321 Laser Printer Modification.mpp".

Name	Baseline Start Date	Baseline End Date	Start Date	End Date
1 Laser Printer Modification	2025-01-01	2025-03-13	2025-01-01	2025-03-13
2 Phase 1 - Plan and Def	2025-01-01	2025-01-06	2025-01-01	2025-01-06
3 Market Research	2025-01-01	2025-01-06	2025-01-01	2025-01-06
4 Specification Drawing	2025-01-06	2025-01-06	2025-01-06	2025-01-06
5 Specifications Sign Off	2025-01-06	2025-01-06	2025-01-06	2025-01-06
6 Phase 2				
7 Mechanical				
8 Product				
9 Prototype				
10 Quality Planning	2025-01-13	2025-01-13	2025-01-13	2025-01-13
11 Prototype Complete	2025-01-13	2025-01-13	2025-01-13	2025-01-13
12 Phase 3 - Product Dev	2025-01-14	2025-03-13	2025-01-13	2025-03-13
13 Tooling	2025-01-14	2025-01-20	2025-01-13	2025-01-13
14 Process Planning	2025-01-14	2025-01-20	2025-01-13	2025-01-13
15 Process Quality Plan	2025-01-14	2025-02-20	2025-01-13	2025-02-13
16 Production Planning	2025-02-21	2025-03-13	2025-02-20	2025-03-13
17 Launch Product	2025-03-13	2025-03-13	2025-03-13	2025-03-13



Calendars

Introduction

The calendar management system operates across three levels—Organization, Project, and User—to support efficient scheduling, resource planning, and conflict resolution. This multi-tiered structure allows seamless coordination across hierarchical layers by aligning organization-wide policies, project-specific requirements, and individual user preferences. As a result, calendars enhance productivity, reduce scheduling conflicts, and streamline project execution. Whether managing enterprise-wide activities, tracking project milestones, or coordinating personal workloads, the calendar system ensures optimal visibility and control.



Organization Calendar

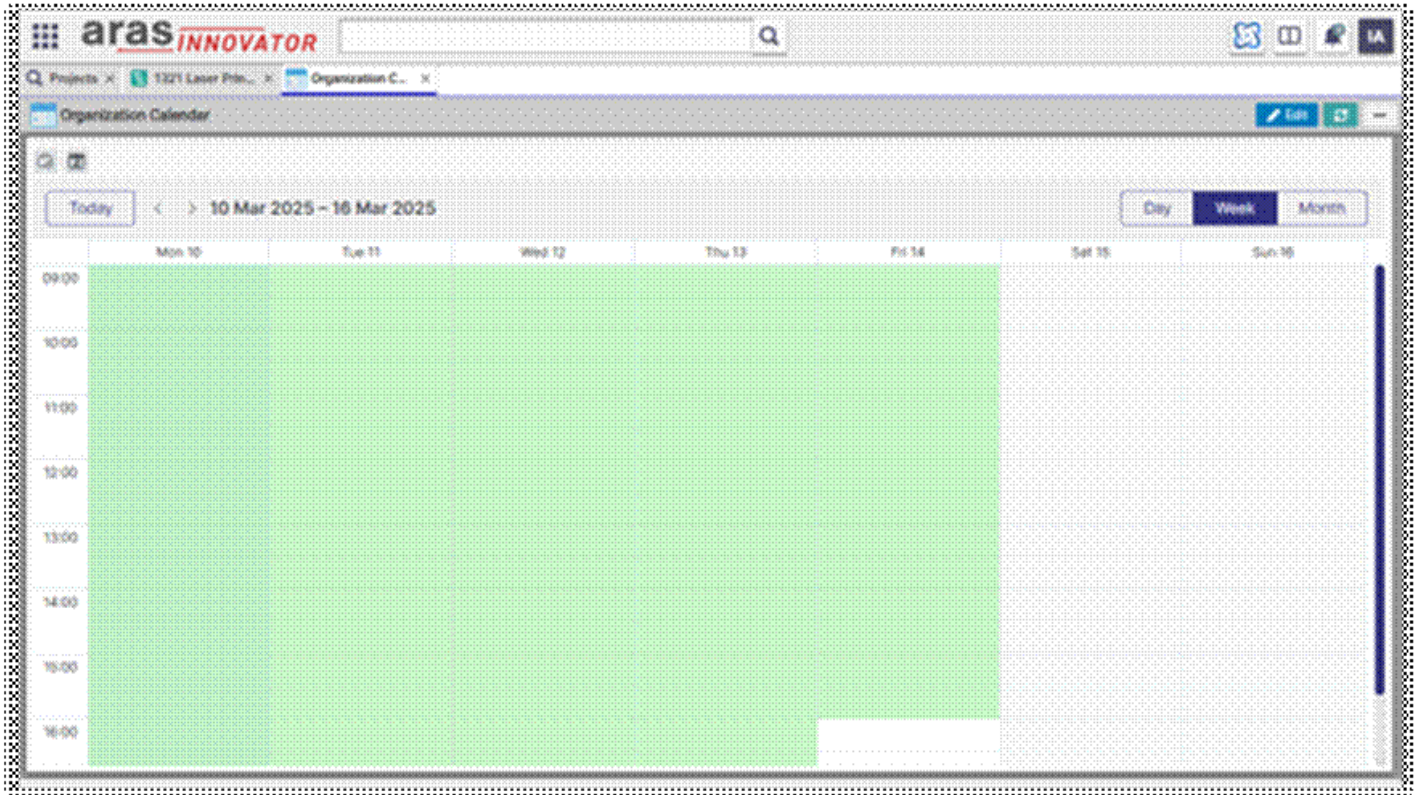
The **Organization Calendar** establishes the foundational working schedule that applies across the entire organization. It acts as the foundational reference for Project and User calendars, promoting consistency in scheduling and aligning all timelines with organizational policies and expectations.

	Name	Baseline Start Date	Baseline End Date	Start Date	End Date
1	Laser Printer Modification	2025-01-01	2025-03-13	2025-01-01	2025-03-1
2	Phase 1 - Plan and Del	2025-01-01	2025-01-06	2025-01-01	2025-01-6
3	Market Research	2025-01-01	2025-01-06	2025-01-01	2025-01-6
4	Specification Drawing	2025-01-06	2025-01-06	2025-01-06	2025-01-6
5	Specification Sign-Off	2025-01-06	2025-01-06	2025-01-06	2025-01-6
6	Phase 2 - Product Dev	2025-01-07	2025-01-13	2025-01-07	2025-01-1
7	Mechanical Product E	2025-01-07	2025-01-13	2025-01-07	2025-01-1
8	Product Validation	2025-01-13	2025-01-13	2025-01-13	2025-01-1
9	Prototype (built and 1)	2025-01-13	2025-01-13	2025-01-13	2025-01-1
10	Quality Planning	2025-01-13	2025-01-13	2025-01-13	2025-01-1
11	Prototype Complete	2025-01-13	2025-01-13	2025-01-13	2025-01-1
12	Phase 3 - Product Lev	2025-01-14	2025-03-13	2025-01-13	2025-03-1
13	Tooling	2025-01-14	2025-01-20	2025-01-13	2025-01-1
14	Process Planning	2025-01-14	2025-01-20	2025-01-13	2025-01-1
15	Process Quality Plan	2025-01-14	2025-02-20	2025-01-13	2025-02-1
16	Production Planning	2025-02-21	2025-03-13	2025-02-20	2025-03-1
17	Final Sign-off	2025-03-13	2025-03-13	2025-03-13	2025-03-1

Steps to Configure Organization Calendar

- **Define Working Days:** Define the organization's standard working days (e.g., Monday to Friday) to establish the baseline for operational planning and scheduling.
- **Set Work Hours:** Specify the organization's standard working hours (e.g., 9:00 AM to 6:00 PM with a 1-hour lunch break from 1:00 PM to 2:00 PM). Establishing consistent work hours across departments.
- **Add Holidays:** List all holidays observed at the organizational level. These may include national holidays, company-wide shutdowns, and special occasions.





null

Toolbar Actions for Organization Calendar:

- **Manage Working Hours:** Configure the organization's standard working days and working hours for the organization. Users can adjust default schedules and apply changes across all projects.



The screenshot displays the Aras Innovator interface. At the top left, the logo "aras INNOVATOR" is visible. Below it, there is a search bar and a tab labeled "Organization C...". The main area is divided into two panels. The left panel, titled "Organizational Calendar", shows a calendar view for the period "17 Mar 2025 - 23 Mar 2025". The calendar grid is currently empty and has a light green background. The right panel, titled "Working Hours", contains a table with the following data:

Week Day	Start Time	End Time
<input checked="" type="checkbox"/> Monday	09:00:00 AM	06:00:00 PM
<input checked="" type="checkbox"/> Tuesday	09:00:00 AM	06:00:00 PM
<input checked="" type="checkbox"/> Wednesday	09:00:00 AM	06:00:00 PM
<input checked="" type="checkbox"/> Thursday	09:00:00 AM	06:00:00 PM
<input checked="" type="checkbox"/> Friday	09:00:00 AM	06:00:00 PM
<input type="checkbox"/> Saturday	12:00:00 AM	12:00:00 AM
<input type="checkbox"/> Sunday	12:00:00 AM	12:00:00 AM

At the bottom right of the interface, there are "OK" and "Cancel" buttons.

- **Add/Edit Holidays:** Maintain a list of company-wide holidays, ensuring they are reflected in all schedules.

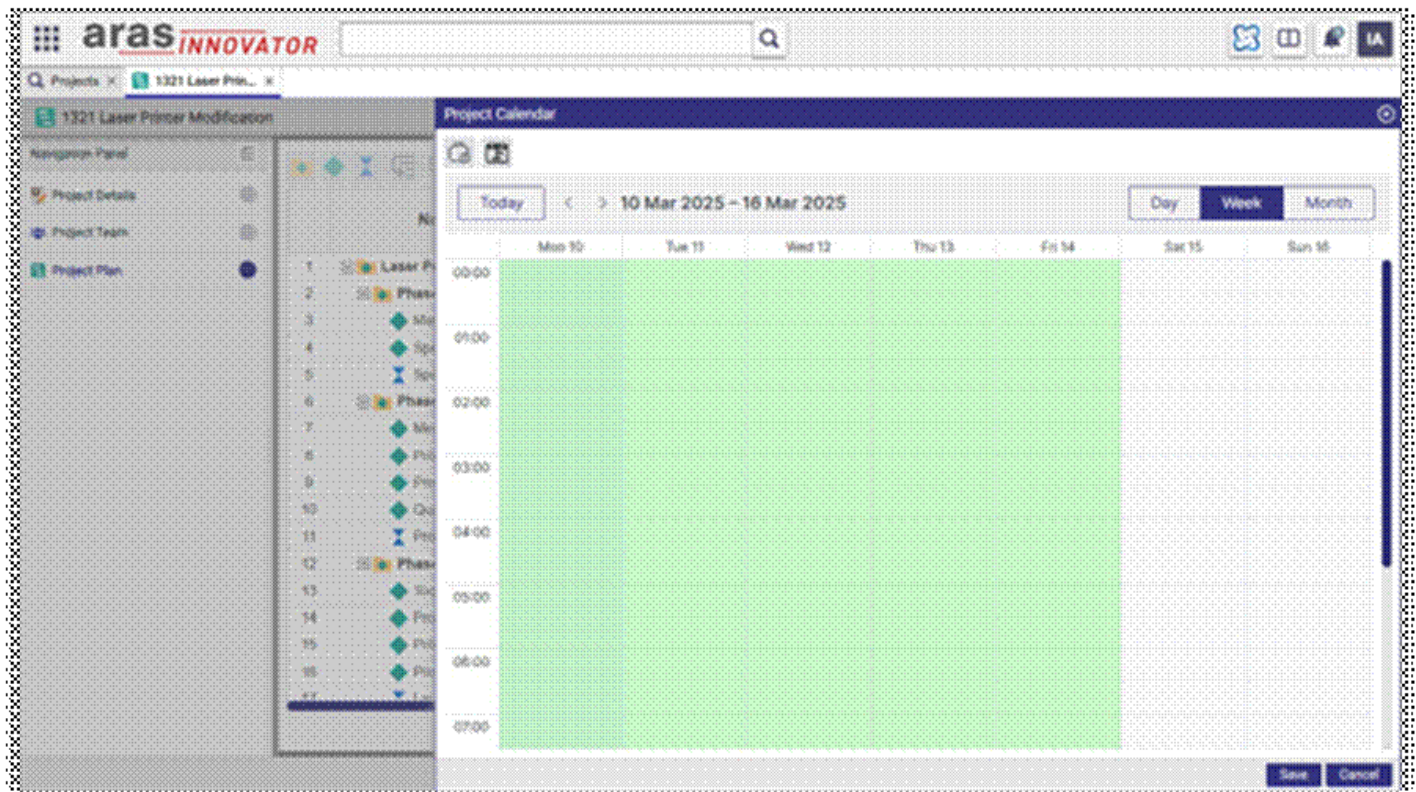


The screenshot displays the Aras Innovator Organization Calendar. The main calendar view shows a grid for the week of March 17-23, 2025. The time axis on the left ranges from 09:00 to 16:00. A 'Holidays' panel on the right lists the following events:

Name	Start Date	End Date
New Year's Eve	12/31/2024	12/31/2024
Christmas	12/25/2024	12/25/2024
Thanksgiving Day	11/28/2024	11/28/2024
May Day	5/1/2025	5/1/2025
Organization Transition Day	1/26/2025	1/26/2025



Project Calendar



The **Project Calendar** inherits default working days, hours, and holidays from the Organization Calendar, but allows project-specific customization. Project managers can adjust the calendar to accommodate unique timelines, resource availability, or regional considerations, while still maintaining overall consistency with organizational policies.

Steps to Configure Project Calendar

- **Inherit from Organization Calendar:** By default, the project calendar adopts the organization-wide settings, including working days, hours, and holidays.
- **Override Working Days:** If a project requires an alternative working schedule—such as weekend shifts, non-standard work hours, or region-specific holidays. These modifications can be made directly within the Project Calendar.
- **Adjust Work Hours:** Project managers can modify work hours to accommodate project-specific needs, such as extended work hours for urgent deadlines.
- **Add Project-Specific Holidays:** Define non-working days that apply exclusively to a specific project such as milestone-related breaks, scheduled maintenance windows, or region-specific observances.



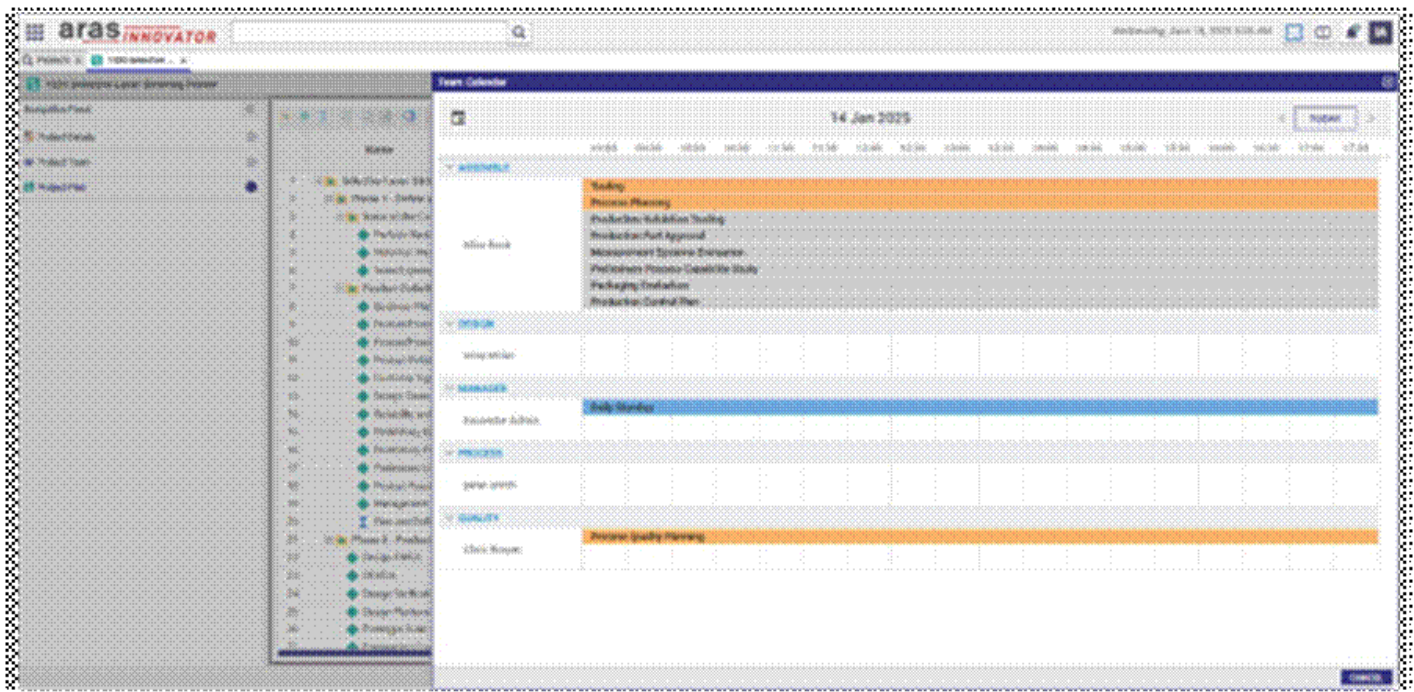
Toolbar Actions for Project Calendar:

- **Manage Working Hours:** Modify both working days and work hours specifically for the project. Changes here override organizational settings.
- **Add Project-Specific Holidays:** Customize the holiday list for a particular project without affecting other schedules.



Team Calendar

The **Team Calendar** offers a consolidated, real-time view of all team members' schedules within a project. It aggregates data from individual User Calendars and project assignments, enabling project managers to monitor workloads, assess team progress, and proactively identify scheduling conflicts. Serving as a centralized coordination hub, the Team Calendar enhances visibility, improves collaboration, and supports effective resource allocation across the project team.



Steps to Configure Team Calendar

- **View Team Member Availability:** Project managers gain real-time visibility into each team member's availability including standard working hours, scheduled vacations, and personal events through the Team Calendar.
- **Monitor Task Progress:** The Team Calendar displays the real-time status of tasks assigned to each team member such as In Progress, Completed, or Pending. This consolidated view offers project managers a clear snapshot of the team's workload, task distribution, and overall project progress.
- **Identify Resource Conflicts:** The system proactively identifies and highlights cases where team members are over-allocated or have conflicting schedules within the Team Calendar allowing project managers to reallocate tasks and resolve issues proactively.
- **Facilitate Collaboration:** - By providing clear visibility into each team member's schedule, the Team Calendar fosters stronger coordination and more effective communication. This shared awareness of colleagues' availability and commitments helps reduce miscommunication, streamline collaboration.

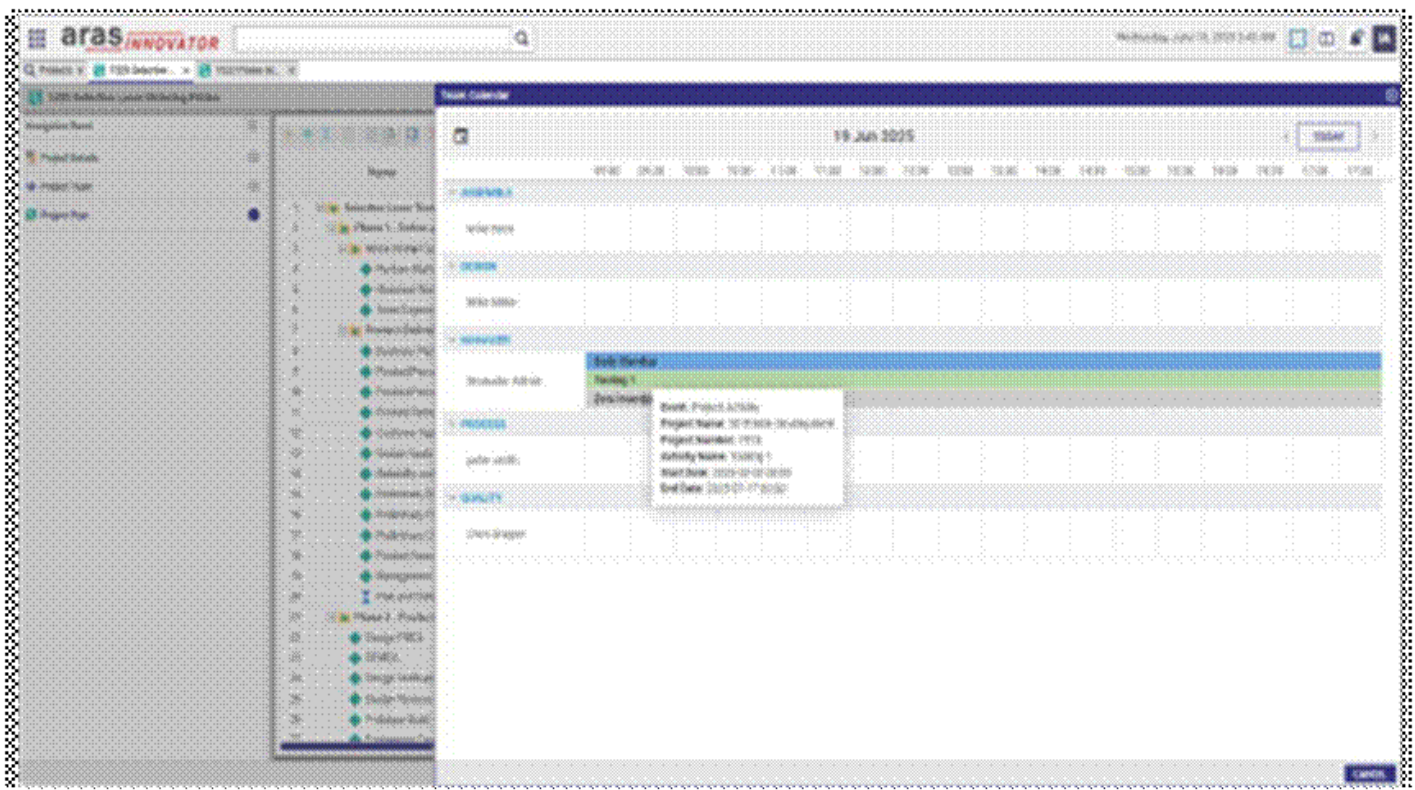


Navigation in Team Calendar

The Team Calendar is presented exclusively in a day view and provides multiple navigation tools for streamlined browsing:

- **Next Button:** Click the "Next" button to advance the calendar to the following day. This allows for a quick progression through daily schedules.
- **Back Button:** Click the "Back" button to move the calendar to the previous day. This is useful for reviewing past events and tasks.
- **Today Button:** Click the "Today" button to quickly return to the current date's view. This ensures you can always snap back to the present schedule.
- **Mini Calendar:** A mini calendar is located at the top-left corner of the interface, the mini calendar acts as a date picker. Users can click any specific date to quickly jump to that day's schedule, making it easy to view non-sequential dates on demand.

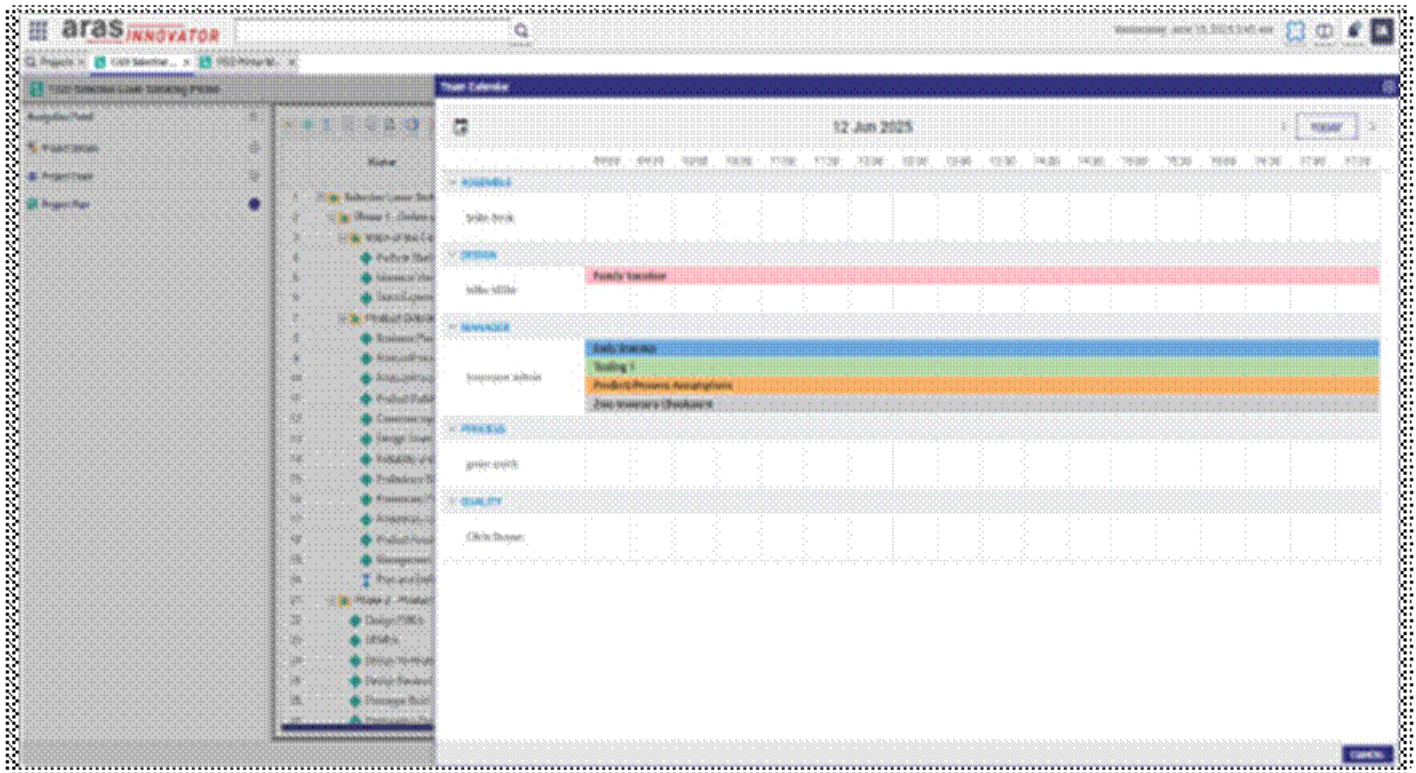
Tooltips for Events and Activities



When users hover the mouse cursor over any event, activity, task, or vacation entry within the Team Calendar, a tooltip automatically appears. This tooltip reveals key information such as task name,

assigned user, status, or time details providing quick, contextual insight without requiring clicks or additional navigation.

Color Coding for Events, Activities, and Vacations for team



To improve clarity and organization within the Team Calendar, a consistent set of color codes is applied—aligned with those used in the User Calendar where applicable. These visual cues allow users to quickly interpret the status of team activities and personal schedules at a glance.

- **Orange:** Project tasks currently in progress, helping users monitor ongoing work.
- **Grey:** Project tasks that are not in an active state, helping users track pending work.
- **Blue:** Custom created events or meetings, allowing easy identification of manually scheduled activities.
- **Green:** Completed project tasks, making it easy to identify successfully finished activities.
- **Pink:** Vacation or personal leave entries, helping teams account for individual availability.



User Calendar

The **User Calendar** is tailored to each individual within the organization, the User Calendar provides a personalized view of the user's working schedule, task assignments, vacations, and other events. It integrates seamlessly with project calendars, allowing individuals to manage their availability while contributing to team and organizational schedules. This synchronization ensures that personal workload, time off, and task visibility are accurately reflected across planning tools, helping optimize resource allocation and minimize conflicts.

The screenshot displays the User Calendar interface. At the top right, there are icons for navigation and user identification. Below these is a toolbar with buttons for 'Edit', 'Promote', and 'Open Item'. A secondary toolbar contains icons for communication, editing, and viewing options like 'Split View' and 'Week'. The main area is a calendar grid for the week ending 31 Dec, 2024. The grid shows task assignments for Wednesday and Thursday. A dropdown menu is open on the right side, listing various system functions, with 'User Calendar' highlighted.

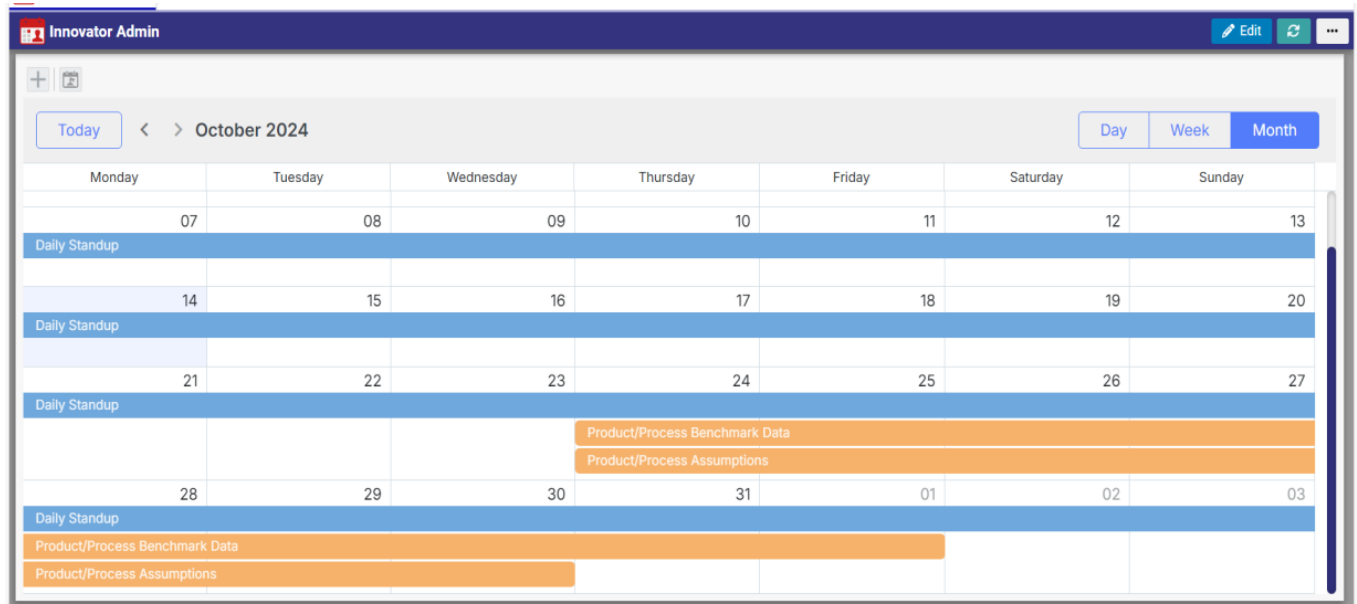
End Date	Start Date	31 Dec , 2024			
		Tue	Wed	Thu	Fri
03-13	2025-01-01				
01-06	2025-01-01			Phase 1 - Plan an	
01-06	2025-01-01			Market P	
01-06	2025-01-06				
01-06	2025-01-06				
01-13	2025-01-07				

Steps to Configure User Calendar

- **View Assigned Tasks:** Allows users to directly access their assigned project tasks from within the calendar interface, supporting better time management and task prioritization.



- **Create Events and Vacations:** Enables users to schedule personal events, meetings, training sessions, and vacations, ensuring accurate reflection of availability across project planning tools.
- **Check Project Task Completion:** - Users can seamlessly update the status of their project tasks marking them as Completed, In Progress, or Delayed without leaving the calendar view.



- **Detect Assignment Conflicts:** The system automatically detects scheduling conflicts between user availability and assigned tasks, providing real-time alerts to enable quick resolution.

Toolbar Actions for User Calendar:

null



The screenshot displays the Aras Innovator application interface. At the top, the 'aras INNOVATOR' logo is visible on the left, and navigation icons are on the right. Below the header, there are two browser tabs labeled 'Innovator Admin'. The main content area is split into two panes. The left pane shows a calendar view for the week of March 17 to 23, 2025. A 'Daily Standup' event is highlighted in blue on Monday, March 17, from 00:00 to 09:00. The right pane is a 'Create Event' dialog box with the following fields:

- Basic Details** (Section Header)
- Title *** (Text input field)
- New Event** (Text input field)
- Start Date *** (Date picker with 'Select a date' text and a calendar icon)
- End Date *** (Date picker with 'Select a date' text and a calendar icon)

At the bottom right of the dialog box, there are 'Ok' and 'Cancel' buttons.

- **Manage Events:** - Users can add or edit personal events such as meetings, training sessions, or reminders within the User Calendar. This ensures individual schedules are accurately reflected and remain in sync with broader company operations and project timelines.
- **Schedule Vacations:** Users can input their vacation days directly within the User Calendar. These entries are automatically reflected in project scheduling tools, ensuring tasks are not assigned during periods of unavailability.



The screenshot displays the aras INNOVATOR application interface. At the top, there is a search bar and navigation icons. Below the header, the main area is split into two panels. The left panel shows a calendar for the week of March 17-20, 2025, with a 'Daily Standup' event highlighted. The right panel, titled 'Vacations', contains a table with the following data:

Name	Start Date	End Date
Family Vacation	10/23/2024	10/23/2024
New year	1/1/2025	1/1/2025
Republic Day	1/26/2025	1/26/2025

- **Update Task Status:** Users can update the status of their assigned tasks directly within the User Calendar by marking them as Completed, In Progress, or Delayed. These real-time updates are immediately visible to project managers, ensuring accurate progress tracking and enabling timely adjustments to workloads or schedules as needed.



Activity Completion Form in User Calendar

The **Activity Completion Form** enables users to finalize and update the status of assigned tasks directly within their **User Calendar**. This integrated feature streamlines the reporting process by automatically syncing updates with the project schedule, reducing the need for manual entries and enhancing the accuracy and timeliness of project tracking.

Features of Activity Completion Form

- **Basic Details** – Displays core information including the task leader, assigned role, percentage of completion, schedule, and duration.
- **Assignment Verification** – Users can confirm task completion by checking a confirmation box and entering the actual start date.
- **Task Overview** – Lists all tasks associated with the activity, enabling users to update individual task statuses as needed.
- **Deliverables Management** – Shows any documents or files attached to the task, ensuring visibility and easy access to required deliverables.
- **Time Record Logging** – Allows users to log time entries corresponding to the completed activity for accurate time tracking.
- **Comment Section** – Allows users to leave remarks, upload supporting documents, or attach relevant files for future reference.

The Activity Completion Form available within the User Calendar mirrors the version used in the Project Plan, ensuring a consistent experience for users across both interfaces.

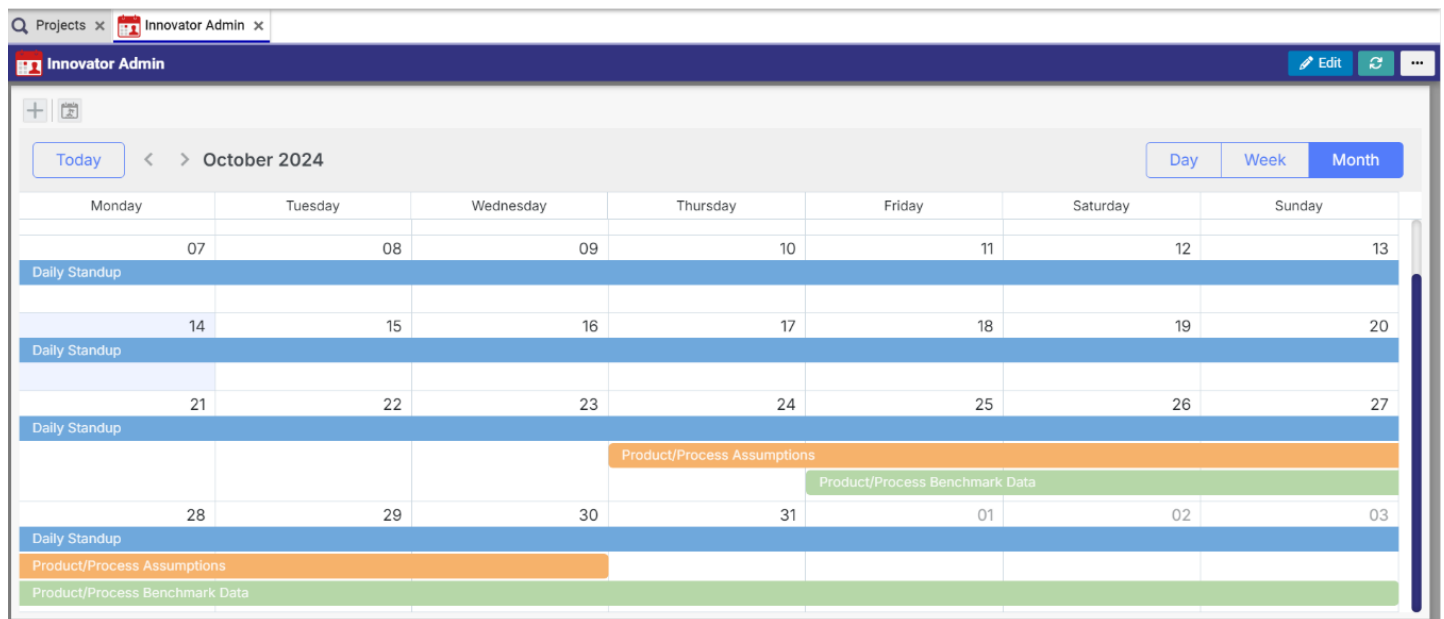


Event and Task Color Coding

To improve visual clarity and streamline scheduling, the calendar employs the following color codes to represent event types and task statuses:

- **Blue** – Custom-created events or meetings, offering clear differentiation of manually scheduled activities.
- **Grey** – Project tasks that are not yet active or pending, helping users anticipate upcoming responsibilities.
- **Orange** – Project tasks currently in progress, allowing easy identification of active workstreams.
- **Green** – Completed project tasks, making it easy to identify successfully finished activities.

This intuitive color system offers users and project managers a high-level view of progress and availability, fostering more efficient calendar navigation and team coordination.



Aras ProApp Designer - Developer Guide



Introduction

Purpose

This User Guide describes how to use the ProAppDesigner API's for developers.



Scope

This document provides information about APIs supported in ProAppDesigner application.



Target Audience

This document is designed for developers seeking to customize the template using JavaScript.



Feature Terms and Definitions

The following Feature terms and definitions are used in this document:

Term	Definition



Guide Organization

This guide explains how to integrate and utilize the ProAppDesigner APIs within your application. This Developer Guide is organized into the following sections:

SECTION 1:	This section outlines the purpose and scope of this Developer Guide, as well as its intended target audience. It supplies a brief overview of each section to help users in navigating the guide.
Introduction	
SECTION 2:	This section defines the purpose of the ProAppDesigner API.
ProAppDesigner API Overview	
SECTION 3:	This section details the range of events that can be triggered from the Wizard and Wizard Controls.
Events	
SECTION 4:	This section covers handling events using custom scripting.
Event Handling	
SECTION 5:	This section provides an overview of the various Wizard Control APIs available in ProAppDesigner.
Wizard and Wizard Control API	
SECTION 6:	This section outlines the different Message APIs and their functions.
Messages	
SECTION 7:	This section outlines the Progress Ring APIs and their usage.
Progress Ring	
SECTION 8:	This section outlines various server methods and how to invoke them from the client side.
ApplyMethod	
SECTION 9:	This section provides examples of how to set and retrieve data from various UI controls.
Examples	



ProAppDesigner API Overview

Aras ProAppDesigner Developer Guide is a comprehensive resource for customizing templates created using ProAppDesigner.

ProAppDesigner is a no-code solution for building user interfaces in Aras Innovator. Most of the UI can be created through configuration in TemplateStudio, using a rich set of controls available in the Toolbox. However, certain use cases may require writing code to implement custom logic.

The Developer Guide explains the events and JavaScript APIs needed to interact with the controls used in templates. It also provides APIs for reading data from and writing data to a control, provided the control's specific details are known.

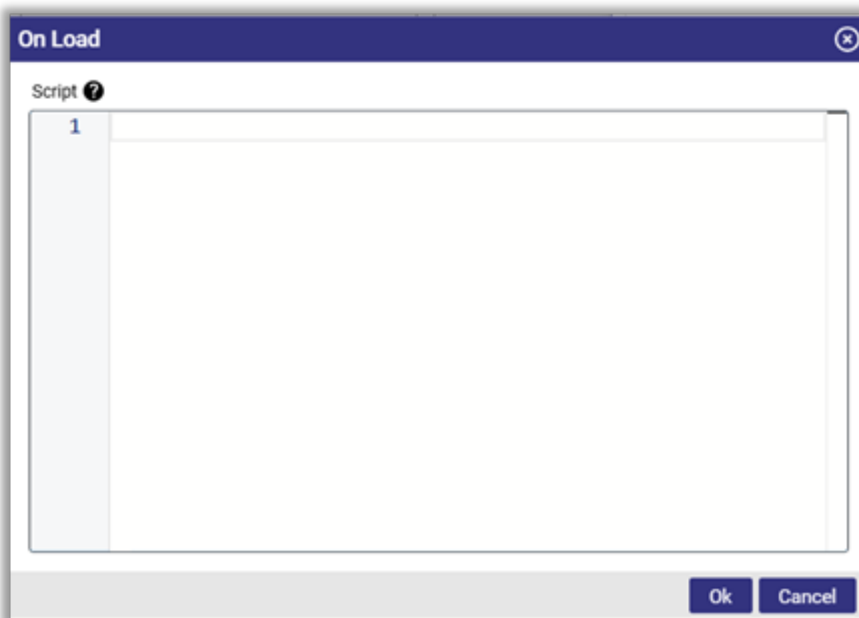


Events

ProAppDesigner enables customization through a variety of event types that are triggered by the Wizard at different points during runtime. While configuring a ProAppDesigner template, administrators can register custom scripts for any of the supported events. These events are available on various controls within TemplateStudio, and custom scripts can be associated with them through the control's settings.



The image shows two rows of configuration fields. The first row is labeled 'On Load' and contains a text input field followed by a checkmark icon and a red 'X' icon. The second row is labeled 'On Change' and also contains a text input field followed by a checkmark icon and a red 'X' icon.



The image shows a dialog box titled 'On Load' with a close button in the top right corner. Inside the dialog, there is a section labeled 'Script ?' with a question mark icon. Below this label is a list box containing the number '1'. To the right of the list box is a large text area for entering the script. At the bottom of the dialog are 'Ok' and 'Cancel' buttons.



OnClick Event

The OnClick event is exclusively supported on the Button control. This event is triggered whenever the button is clicked, allowing you to execute custom logic in response to the user interaction.

// on button click opening custom flyout

```
appStudioWizard.showDialogFlyout(pageDialogName);
```



OnLoad Event

The OnLoad event is triggered after the data has been loaded for either the Wizard page or individual collection controls. This allows you to execute custom logic once the data is available, enabling dynamic behavior or additional initialization based on the loaded content.

//Setting list options on a column of the Table control:

```
const itemId = appStudioWizard.getItemId();
```

*// calling a server method & passing the **itemId** to get the required options. Refer to **2.7 Examples** on how to call server method.*

// building options map

```
let options = {  
  optionValue: "OptionLabel",  
  optionValue2: "OptionLabel2"  
};  
appStudioWizard.setCollectionListOptions("table_name", "column_name",  
options);
```



OnChange Event

The OnChange event is triggered whenever the value of a control is changed. It is supported on both simple controls (like text boxes, checkboxes, etc.) and columns within collection controls (such as tables or worksheets).

For **collection controls**, when the **OnChange** event is triggered, the eventData object includes both the **selectedItem** and the **columnName**. These allow you to access the value that was changed.

// On value change of control1, access its value and set the value of control2

```
let control1Value = appStudioWizard.getControlValue("control1");
if (control1Value) {
  let value = control1Value === "Value1" ? "Discoverable" : "Hidden";
  appStudioWizard.setControlValue("control2", value);
}
```



OnBeforeLoad Event

The OnBeforeLoad event is triggered before data is loaded into collection controls on a page. This event is supported only on table, worksheet, treeTable controls. OnBeforeLoad event can be used to set filterExpression on the collection control to load only required dataset.

```
//Setting filterExpression on Table control
```

```
// creating an expression string
```

```
let expressionString =  
"classification EqualTo Item AND name StartsWith AS_";
```

```
// Refer to section filterCollectionItems on how to build your expression string
```

```
// calling filter API for table control
```

```
appStudioWizard.filterCollectionItems(controlName, expressionString);
```



OnSelect Event

The OnSelect event is triggered when an item within a collection control is selected by the user.

// getting the selected item to access its data

```
const selectedControlItems = appStudioWizard.getControlSelectedItems(
"supplierParts");
if (selectedItems) {
const selectedItem = selectedItems[0];
```

// accessing data of selected item

```
const itemId = selectedItem.getItemId();
const relatedId = selectedItem.getRelatedId();
const partWeight = selectedItem.getValue("weight");
}
```



OnSubmit Event

The OnSubmit event is triggered after the Submit button is clicked within a flyout. This event is supported exclusively for flyout.

// accessing form values in the flyout & saving them

```
let price = appStudioWizard.getControlValue("price");  
let date = appStudioWizard.getControlValue("purchased on");  
let managedBy = appStudioWizard.getControlValue("managed_by_id");
```

*// calling a server method to save the data. Refer to **2.7 Examples** on how to call server method.*

```
if (onSuccess) {  
    appStudioWizard.showSuccessMessage("Item saved successfully");  
} else {  
    appStudioWizard.showErrorMessage("Failed to save the Item.");  
}
```



OnInsert Event

The **OnInsert** event is triggered after an item has been inserted into a collection control. The event handler receives the details of the inserted item through the `eventData` object, which is available within the scope of the event handler code.

// Accessing data of inserted item

```
const insertedItemCount = eventData.length;
for (let i = 0; i < insertedItemCount; i++) {
```

// accessing data of selected item

```
    const relationshipId = eventData[i].getItemId();
    const relatedItemId = eventData[i].getRelatedId();
    const partWeight = eventData[i].getValue("weight");
```

// setting a cell value on inserted item

```
    eventData[i].setValue("classification", "item");
}
```



OnSelectFlyout Event

The **OnSelectFlyout** event is triggered when opening the SelectItem flyout and is supported by the Item and Items controls. In this event handler, the admin can implement custom scripts and use the `setSelectFilterExpression` API to apply a custom filter expression.

// creating a custom expression string

```
let expressionString = "classification EqualTo Item AND id NotEqualTo  
8D58C2D8DC1649279E64CCF5BD2FF7A7";
```

*// Refer to **filterCollectionItems** section on how to build your expression string*

```
appStudioWizard.setSelectFilterExpression(controlName, expressionString,  
enforce);
```



OnInsertFlyout Event

The **OnInsertFlyout** event is triggered when opening the InsertItem flyout and is supported in Table, Worksheet, and TreeTable controls. In the event handler, the admin can write custom scripts and use the `setInsertFilterExpression` API to apply a custom filter expression.

// creating a custom expression string

```
let expressionString = "classification EqualTo Item AND id NotEqualTo  
8D58C2D8DC1649279E64CCF5BD2FF7A7";
```

*// Refer to **filterCollectionItems** section on how to build your expression string*

```
appStudioWizard.setInsertFilterExpression(controlName, expressionString,  
enforce);
```



OnSave Event

The OnSave event is triggered after saving the item within the wizard or in section/group controls that have their own associated items.

```
const itemId = appStudioWizard.getItemId();  
const itemType = appStudioWizard.getItemType();
```

*// calling a server method to save custom data. Refer to **2.7 Examples** on how to call server method.*

```
if (onSuccess)  
appStudioWizard.showSuccessMessage("Data saved successfully");  
else  
appStudioWizard.showErrorMessage("Failed to save the custom data.");
```



OnBeforeSave Event

The OnBeforeSave event is triggered before the actual save request is sent to the server. Within its event handler, you can implement code to validate changes and cancel the save operation by returning false if any issues are detected.

```
var isItemNew = appStudioWizard.isItemNew();  
var controlValue = appStudioWizard.getControlValue('controlName');
```

// Before calling a server method to save data. Refer to ApplyMethod Examples on how to stop server method to save data.

```
if (isNew && !controlValue)  
true;  
else  
appStudioWizard.showErrorMessage("Failed to save the custom data.");  
false
```



Event Handling

A custom script can be defined as an event handler using the predefined variable `appStudioWizard`. This script can include any JavaScript code that reads values from wizard controls, makes requests to the server using the Innovator IOM API, and updates controls based on the server's response. It can also make requests to third-party servers via JavaScript HTTP clients. ProAppDesigner Wizard provides a simple API accessible through `appStudioWizard` to interact with different control types. Additionally, the wizard offers APIs to display various message types to users during event handling. If the event handler's code takes time to execute, there is also an API to show a progress ring to indicate ongoing processing.



Wizard and Wizard Controls API

Wizard controls are categorized into two types: simple controls and collection controls. For simple controls, you can easily get or set values using the `getControlValue` and `setControlValue` APIs. For collection controls, you first need to retrieve the selected items by calling the `getControlSelectedItems` API. Each selected item is represented by an `ItemControl`, which allows you to get or set column values using the `getValue` and `setValue` APIs.

To set or get values from controls, you can pass the control name to the API methods. However, if there are multiple controls with the same name, you need to use a qualified path instead of just the control name to uniquely identify the specific control when calling the API methods.

QualifiedPath for controls used in the group control:

pageName/sectionName/groupName/controlName

QualifiedPath for controls used in the panel control:

pageName/sectionName/groupName/panelControlName/pageName/sectionName/groupName/controlName

Example:

```
appStudioWizard.getControlValue("pageName/sectionName/groupName/controlName");
```



Wizard

The Wizard object can be accessed in scripts using the keyword `AppStudioWizard`, which provides all the APIs needed to interact with the various controls within the Wizard.

getItemId

Get `itemId` of the item loaded in the Wizard.

```
var itemId = appStudioWizard.getItemId()
```

getItemType

Get `itemType` of the item loaded in the Wizard.

```
var itemType = appStudioWizard.getItemType()
```

getItemConfigId

Get `itemConfigId` of the item loaded in the Wizard.

```
var itemConfigId = appStudioWizard.getItemConfigId()
```

isItemNew

Returns true if the item in the Wizard is never saved to server.

```
var isItemNew = appStudioWizard.isItemNew()
```

isItemEditable

Returns true if the item in the Wizard is editable (in edit mode).

```
var isItemEditable = appStudioWizard.isItemEditable()
```

Section and Group controls on a page can have their own `ItemType` context. The following API returns true if the associated item of the layout control (Section or Group) is editable (i.e., the page is in edit mode).

```
var isLayoutEditable = appStudioWizard.isItemEditable(layoutControlName)
```

isControlUpdated



Checks control with the given name is updated (has unsaved changes)

```
var isControlUpdated = appStudioWizard.isControlUpdated(controlName)
```

reload

Reload the entire wizard with the latest item details from the server, discarding all local changes.

```
appStudioWizard.reload();
```

reloadControl

Reload a specific collection control's data from the server. discarding all local changes.

```
appStudioWizard.reloadControl(controlName);
```

showDialogFlyout

Shows the flyout given the flyout name.

```
appStudioWizard.showDialogFlyout(pageDialogName);
```

showInsertFlyout

Shows the associated InsertItem flyout on the collection controls

```
appStudioWizard.showInsertFlyout(controlName);
```

showWorkflowActivityFlyout

Shows WorkflowActivity flyout given the activityId, workflowId, itemId and tableControlName.

```
appStudioWizard.showWorkflowActivityFlyout(activityId, workflowId, itemId, tableControlName);
```

downloadFile

Downloads a file from vault server given the fileId and fileName.

```
appStudioWizard.downloadFile(fileId, fileName);
```



Aras Object

Aras object can be accessed in scripts using the keyword Aras. It provides APIs to interact with Innovator on the client side. However, it is only available for templateViews used to display the Wizard within the Innovator UI and does not function when the templateView is used to show the UI in the Portal.



Simple Controls

Following API works with all simple controls like Text, List, Checkbox, RichText, Item and Items.

getControlValue

Gets the value of the simple control based on its name.

```
var controlValue = appStudioWizard.getControlValue(controlName);
```

setControlValue

Sets value for the simple control based on its name.

```
appStudioWizard.setControlValue(controlName, controlValue);
```

setSelectFilterExpression

This API sets a default filter expression inside the SelectItem flyout when it opens for selecting an item. It is supported for Item and Items controls. Additionally, an optional boolean parameter, enforce, can be passed to hide the configured filter expression from the user.

// creating a custom expression string

```
let expressionString = "classification EqualTo Item AND id NotEqualTo  
8D58C2D8DC1649279E64CCF5BD2FF7A7";
```

*// Refer to **filterCollectionItems** section on how to build your expression string*

```
appStudioWizard.setSelectFilterExpression(controlName, expressionString,  
enforce);
```

setControlLabel

Control label can be updated. given its control name & label value.

```
appStudioWizard.setControlLabel(controlName, label);
```



Collection Controls

The following API functions with collection controls such as Table, Worksheet, TreeTable, Structure, DashboardTable, and DashboardWorksheet. These collection control APIs can only be used when the controls are visible on the current page.

getControlSelectedItems

Gets list of selected items from a collection control given the collection control name.

```
var controlItems =  
appStudioWizard.getControlSelectedItems(collectionControlName);
```

setControlSelectedItems

This API selects items in a collection control by specifying the collection control name along with an array of relationship IDs when the table displays relationship data. For dashboard tables, dashboard worksheets, and reference items tables, you can pass item IDs directly instead.

```
const arrayOfIds = ["131706F6B80D47A5A68ED00BAF210CC8"]  
appStudioWizard.setControlSelectedItems(collectionControlName,  
arrayOfIds);
```

getControlItem

Retrieves an item from a collection control by specifying the control name and either the relationshipId or itemId. Use relationshipId when the table represents relationships, and itemId when the table represents dashboard data or references.

```
var itemControl = appStudioWizard.getControlItem(collectionControlName,  
relationshipId/itemId);
```

getControlItems

Retrieves all items from a collection control by specifying the control name. This method returns all items currently bound to the control.

```
var items = appStudioWizard.getControlItems(collectionControlName);
```



getControlItemByRelatedId

Retrieves an item from a collection control using the relatedItemId and the control name. It works only for collection controls that display non-null relationship data.

```
var controlItems =  
appStudioWizard.getControlItemByRelatedId(collectionControlName,  
relatedItemId);
```

addControlItem

This API adds a new item to a collection control—such as a table, worksheet, or tree table—that displays relationship data. The new item is inserted immediately after the specified relationshipId within the collection. The function requires the collection control name and an ItemData object containing all relevant properties and their values. Upon successful insertion, the newly added item is returned.

```
const itemData = {  
  properties: {  
    address: "Sample Data",  
  },  
  relatedItem: {  
    properties: {  
      name: "Sample Item",  
      description: "Sample Desc",  
    },  
  },  
};  
  
var addedControlItem =  
appStudioWizard.addControlItem(collectionControlName, relationshipId,  
itemData);
```

addControlItems

Adds multiple items to a dynamic table control by passing an array of itemData objects, each containing all necessary properties and values. This method only supports dynamic table controls



that display relationship data and returns an array of all inserted items.

```
const itemdataArray = [  
  {  
    properties: {  
      name: 'Sample Item 1',  
      description: 'Sample Desc 1',  
    },  
  },  
  {  
    properties: {  
      name: 'Sample Item 2',  
      description: 'Sample Desc 2',  
    },  
  },  
]  
  
var addedControlItem =  
appStudioWizard.addedControlItem(collectionControlName, itemdataArray);
```

insertControlItem

This API inserts an item into a collection control—specifically a table, tree table, or worksheet that displays relationship data—immediately after the specified relationshipId. It requires the collection control name and returns the inserted item.

```
appStudioWizard.insertControlItem(collectionControlName, relationshipId,  
insertItemId).then((insertedItemControl) => {  
  // accessing inserted ItemControl and setting its value  
  insertedItemControl.setValue(columnPropertyName, cellValue);  
});
```

deleteControlItem

This API deletes an item from a collection control. For relationship tables, you provide the collection control name and the relationshipId of the item to delete. For tables displaying reference items or



items, you pass the itemId instead.

```
appStudioWizard.deleteControlItem(collectionControlName,  
relationshipId/itemId);
```

setCommandDropdownOptions

Each collection control includes its own toolbar with various commands. If a command in the toolbar is a dropdown, you can set the options for that dropdown by using this function, providing the collection control name and the command name.

```
var options = [  
  { label: 'optionLabel', value: 'optionValue' },  
  { label: 'optionLabel2', value: 'optionValue2' },  
];
```

```
appStudioWizard.setCommandDropdownOptions(collectionControlName,  
commandName, options);
```

filterCollectionItems

This API is applicable only to Table, Worksheet, and Report controls that support expression-based filtering. By invoking this function and passing an expressionString, the items displayed in the control will be filtered accordingly. The filtering is executed on the server, and the control's data is refreshed to reflect the updated results.

You can use this API in the OnBeforeLoad event of the control to load data dynamically based on the specified filter expression. Alternatively, it can be used in a custom toolbar command's onClick handler. However, when used this way, you must also call reloadControl immediately afterward to ensure the data in the control is refreshed according to the new filter. For Report controls, this API is only applicable if the report was created using an expression-based query.

```
appStudioWizard.filterCollectionItems(collectionControlName,  
expressionString);
```



Simple Expression: A simple expression consists of a property name, an operator, and a value. The type of operator that can be used in the expression depends on the data type of the selected property. To view the list of supported operators for a specific property, you can access the Property Filter section in the Properties flyout of the respective control.

```
" propertyName OPERATOR value "
```

Following operators can be used to create an expression based of the property's data type.

Contains, EqualTo, NotEqualTo, LessThan, GreaterThan, LessThanEqualTo, GreaterThanEqualTo, StartsWith, EndsWith, DoesNotContains, DoesNotStartsWith & DoesNotEndsWith.

Complex Expression: Complex expression is composed of many simple expressions with logical and grouping operators.

```
" expression LOGICAL_OPERATOR expression LOGICAL_OPERATOR expression "
```

Logical operators 'AND' and 'OR' can be used along with grouping operators '(' and ')' to a create complex expressions that contain simple expressions.

Sample Expression:

```
const expressionString = "id NotEqualTo
131706F6B80D47A5A68ED00BAF210CC8"
const expressionString = "name Contains app OR name EndsWith item"
const expressionString = "classification EqualTo Item AND name StartsWith
AS_"
```

setInsertFilterExpression

This API allows you to set a **default filter expression** inside the **InsertItem flyout** when it is opened to select an item for insertion. It is supported in **Table**, **Worksheet**, and **TreeTable** controls. You can optionally pass a **boolean parameter enforce**, which, if set to true, will hide the configured filter expression from the user.



// creating a custom expression string

```
let expressionString = "classification EqualTo Item AND id NotEqualTo  
8D58C2D8DC1649279E64CCF5BD2FF7A7";
```

*// Refer to **filterCollectionItems** section on how to build your expression string*

```
appStudioWizard.setInsertFilterExpression(controlName, expressionString,  
enforce, typeName);
```

setControlParameters

This API can be used on Structure, Graph, and Report controls to pass custom parameters to the underlying QueryDefinition or ServerMethod. It accepts a JavaScript object containing key-value pairs that represent the parameters to be passed. These parameters will be sent to the server and used during data retrieval. You can use this API in the OnBeforeLoad event of the control to ensure the data is loaded based on the supplied custom parameters. Alternatively, it can be used in the onClick handler of a custom toolbar command. In that case, you must call the reloadControl API immediately after, to refresh the control's data using the updated parameters.

```
appStudioWizard.setControlParameters(controlName,  
{classification:"Component"});
```

getStructureActiveTab

Gets the typeName of current active tab of structure control given the structure control name and selected node itemId.

```
appStudioWizard.getStructureActiveTab(controlName, itemId);
```

setStructureActiveTab

Sets the currently active tab in the structure control based on the structure control name, the selected node's itemId, and the tab's typeName. For regular relationships, the typeName is a combination of the relationship name and the related ItemType name, as illustrated in the example below. In the case of null relationships, the typeName consists of only the relationship name. When the tab



displays an item from an ItemProperty or a reference item, the typeName should be the ItemType name alone.

```
const typeName = "relationshipName(relatedItemTypeName)";  
appStudioWizard.setStructureActiveTab(controlName, itemId, typeName);
```



Control Item

A control item represents an item displayed in the collection control, either at the root level or as a child. It provides an API to set and retrieve property values, as well as to add or remove child items from the parent control item.

getItemId

Once you have controlItem that represents a row in collection controls, associated relationshipId/itemId can be obtained by calling this function.

```
var itemId = controlItem.getItemId();
```

getRelatedId

Once you have controlItem that represents a row in collection controls, associated relatedId can be obtained by calling this function.

```
var relationshipId = controlItem.getRelatedId();
```

isItemNew

Once you have controlItem that represents a row in collection controls, associated isNew state can be obtained by calling this function. Return true if the item is newly added or inserted.

```
var isItemNew = controlItem.isItemNew();
```

getValue

Once you have a controlItem representing a row in a collection control, you can retrieve the value of a specific cell by providing the column's property name. If the controlItem represents relationship data, and you need to access a property of the related item, the columnName should be specified in the format "related.relatedItemPropertyName".

```
var cellValue = controlItem.getValue(columnPropertyName);
```

setValue



Once you have `itemControl` that represents a row in collection controls, a particular cell value in that row can be set by passing column property name and cell value. If the `controlItem` represents relationship data, to set `relatedItem` property value, you need to pass `columnPropertyName` as “`related.relatedItemPropertyName`”.

```
itemControl.setValue(columnPropertyName, cellValue);
```

getChildItems

Once you have `controlItem` that represents a row in worksheet, `treeTable`, `structure`, `graph` controls, its child items can be retrieved by passed child `relationshipName`.

```
var childControlItems =  
controlItem.getChildItems(childRelationshipName);
```

getChildItem

Once you have `controlItem` that represents a row in worksheet, `treeTable`, `structure`, `graph` controls, its child item can be retrieved by passing child `relationshipName` and child `relationshipId`

```
var childControlItem = controlItem.getChildItem(childRelationshipName,  
childRelationshipId);
```

getChildItemByRelatedId

Once you have `controlItem` that represents a row in collection controls, `childControlItem` can be obtained given that child `relationshipName`, child `relatedId`.

```
var childControlItems = controlItem.getChildItemByRelatedId(  
childRelationshipName, childRelatedItemId);
```

addChildItem

Once you have `controlItem` that represents a row in collection controls, `childItem` can be added given that child `relationshipName`, child `relationshipId`, and `itemData`. It also returns newly added item in the form of `childControlItem`.



```
const itemData = {  
  properties: {  
    address: "Sample Data",  
  },  
  relatedItem: {  
    properties: {  
      name: "Sample Item",  
      description: "Sample Desc",  
    },  
  },  
};
```

```
var childControlItem = controlItem.addChildItem(childRelationshipName,  
childRelationshipId, itemData);
```

insertChildItem

Once you have controlItem that represents a row in collection controls, childControlItem can be inserted given that child relationshipName, child relationshipId, and insertItemId. It also returns newly added item in the form of childControlItem.

```
controlItem.insertChildItem(childRelationshipName, childRelationshipId,  
insertItemId).then((insertedChildItemControl) => {  
  // accessing inserted ItemControl and setting its value  
  insertedChildItemControl.setValue(columnPropertyName, cellValue);  
});
```

deleteChildItem

Once you have controlItem that represents a row in collection controls, childItem can be deleted from the controlItem given childRelationshipName and childRelationshipId.

```
controlItem.deleteChildItem(childRelationshipName, childRelationshipId);
```

getParentItem



Once you have `controlItem` that represents a node in `TreeTable` or `Structure` controls, parent item can be obtained by using this API.

```
var parentControlItem = controlItem.getParentItem();
```

getItemType

Once you have `controlItem` that represents a row or node in collection controls, its relationship name can be obtained by calling this function. If the `controlItem` represent item from `ItemProperty` or `reference` item, then you will obtain its `ItemType` name.

```
var itemTypeName = controlItem.getItemType();
```

getItemConfigId

Once you have `controlItem` that represents a row in collection controls, associated item configId can be obtained by calling this function.

```
var itemConfigId = controlItem.getItemConfigId();
```

getRelatedItemType

Once you have `controlItem` that represents a row or node in collection controls, its related `ItemType` name can be obtained by calling this function.

```
var relatedItemType = controlItem.getRelatedItemType();
```



Layout Controls: Group & Section

Below APIs are supported for Group & Section controls when their Associated Type setting is set to **Type**.

setLayoutControlContext

Set itemId as context for a layout control based on its control name to refresh the control with passed itemId.

```
appStudioWizard.setLayoutControlContext(sectionControlName, itemId);
```

getLayoutControlContext

Get itemId associated with a layout control based on its name.

```
var itemId =  
appStudioWizard.getLayoutControlContext(sectionControlName);
```

expandSection

By passing the section control name and a Boolean parameter as true or false, section control can be expanded or collapsed respectively.

```
appStudioWizard.expandSection(sectionControlName, true);
```



List Control

setListOptions

Set list options for a List control by its control name.

```
const options = { optionValue: "OptionLabel", optionValue2:  
"OptionLabel2" };  
appStudioWizard.setListOptions(listControlName, options);
```

setCollectionListOptions

Collection controls can have column that represents a list. Table, Worksheet, TreeTable, DashboardTable, DashboardWorksheet can have a column of type list. Following API sets list options for a collection control by passing its list column name.

```
const options = {optionValue: "OptionLabel", optionValue2:  
"OptionLabel2"};  
  
appStudioWizard.setCollectionListOptions(collectionControlName,  
columnName, options);
```

setCollectionChildListOptions

Set list options for nested table for a collection control by passing its list column name and relationship name.

```
const options = {optionValue: "OptionLabel", optionValue2:  
"OptionLabel2"};  
  
appStudioWizard.setCollectionChildListOptions(collectionControlName,  
relationshipName, columnName, options);
```



Frame Control

getElement

Gets DOM element of the control based on the control name. This is mainly used to get frame DOM element for the Frame control.

```
var htmlFrameElement = appStudioWizard.getElement(frameControlName);  
if (htmlFrameElement) htmlFrameElement.src = "siteURL";
```



CADViewer Control

setControlValue

Sets cadDocumentId on the CADViewer control by its control name to load the cad model.

```
appStudioWizard.setControlValue(viewerControlName, "cadDocumentId");
```



DynamicTable Control

setTableConfiguration

Sets the configuration for a DynamicTable control by its control name. The configuration object defines the table definition, including its columns, data types, widths, labels (with localization support), and other display settings.

Each column object should specify a unique name, a localized label object, control definition (including data type and localized label), width (in pixels), visibility, and optionally a data source if the column represents a reference or item field.

The `columnWidthByPercentage` property determines whether widths are treated as fixed pixel values (false) or percentage-based (true).

```
const columns = [  
  {  
    name: 'supplier name',  
    label: 'Supplier Name',  
    control: {  
      name: 'supplier name',  
      label: 'Supplier Name',,  
      dataType: 'String',  
    },  
    width: 500,  
    hide: false,  
  },  
  {  
    name: 'team id',  
    label: 'Team',  
    control: {  
      name: 'team id',  
      label: 'Team',  
      dataType: 'Item',
```



```
dataSource: 'Team',  
  }  
width: 400,  
hide: false,  
  }];  
configurationObject = {  
  columns: columns,  
  columnWidthByPercentage: false,  
  }  
  appStudioWizard.setTableConfiguration(tableControlName,  
  configurationObject);
```

Messages

The Message API allows you to display different types of messages, each with a corresponding background color. When any of these APIs is called, a toast notification appears in the top-right corner of the wizard and remains visible for 5 seconds.

showSuccessMessage

Show's success message as flash notification with green background.

```
appStudioWizard.showSuccessMessage(message);
```

showErrorMessage

Shows error message as flash notification with red background.

```
appStudioWizard.showErrorMessage(message);
```

showWarningMessage

Shows warning message as flash notification with orange background.

```
appStudioWizard.showWarningMessage(message);
```

showInfoMessage

Shows info message as flash notification with blue background.

```
appStudioWizard.showInfoMessage(message);
```

Progress Ring

The ProgressRing API is used to display a busy icon during time-consuming operations within a script. When the script makes a server call—whether through the IOM API or HTTPClient—the busy icon can be shown over the wizard to indicate ongoing activity.

showProgressRing

Shows busv icon on the Wizard.

```
appStudioWizard.showProgressRing()
```

hideProgressRing

Hides busv icon on the Wizard.

```
appStudioWizard.hideProgressRing()
```

ApplyMethod

It allows you to invoke any server method defined on the server side. All parameters required by the method must be packaged in JSON format when making the call.

Example 1: Calling serverMethod by passing parameters and getting item in the response

Client Code:

```
const parmObj = { partId: "FB3CBC878D5747DEBF25508B23CF6B5B" };  
const resultObj = appStudioWizard.applyMethod("methodName", parmObj);
```

Server Method:

```
string partId = this.getProperty("partId");  
if(partId == null) throw new Exception("Please provide partId.");  
  
Item partItem = this.getInnovator().newItem("Part", "get");  
partItem.setID(partId);  
partItem = partItem.apply();  
if(partItem.isError() && partItem.isEmpty()) throw new Exception("No  
items found.");  
  
return partItem;
```

Example 2: Calling serverMethod by passing json and getting json in the response

Client Code:

```
const partDetails = { classification: "Component", make_buy: "Make" };  
const body = { request: JSON.stringify(partDetails) };  
const resultObj = appStudioWizard.applyMethod("methodName", body);
```

Server Method:

```
string partDetailsString = this.getProperty("request");  
var partDetails = new Newtonsoft.Json.Linq.JObject();  
if (partDetailsString == null) {  
    throw new Exception("Please provide partDetails.");  
}
```



```
else { partDetails =  
Newtonsoft.Json.Linq.JObject.Parse(partDetailsString);}  
  
string classification =  
partDetails.SelectToken("$.classification")?.ToString();  
string make buy = partDetails.SelectToken("$.make buy")?.ToString();  
Item query = this.getInnovator().NewItem("Part", "get");  
query.setAttribute("select", "item_number, name, classification,  
make buy");  
query.setProperty("classification", classification);  
query.setProperty("make buy", make_buy);  
var partItems = query.apply();  
if(partItems.isError() && partItems.isEmpty()) throw new Exception("No  
items found");  
  
return partItems;
```

Output:

```
[  
{  
  "classification": "Component",  
  "make buy": "Make",  
  "name": "Trunk",  
  "item number": "004",  
  "itemtype": "4F1AC04A2B484F3ABA4E20DB63808A88"  
},  
...]
```



Examples

How to Set and Get Value on Checkbox Control

```
var controlName = "checkbox1";  
appStudioWizard.setControlValue(controlName, false);  
  
var controlValue = appStudioWizard.getControlValue(controlName);  
console.log ("Control Value: " + controlValue);
```

```
//Control Value: false
```

How to Set and Get Value on Text Control

```
var controlName = "itemNumber";  
appStudioWizard.setControlValue(controlName, "P00456");  
  
var controlValue = appStudioWizard.getControlValue(controlName);  
appStudioWizard.showSuccessMessage("Control Value: " + controlValue);  
  
//Control Value: P00456
```

How to Set and Get Value on RichText Control

```
var controlName = "richtext1";  
appStudioWizard.setControlValue(controlName, "<h2>Can set the value.  
</h2>");
```

```
var controlValue = appStudioWizard.getControlValue(controlName);  
appStudioWizard.showSuccessMessage(controlValue);
```

```
//Control Value: <h2>Can set the value.</h2>
```



How to Set and Get Value for a Row on Table Control

```
var controlName = "table1";  
var controlItems =  
appStudioWizard.getControlSelectedItems(controlName);  
controlItems[0].setValue("column2", "some value")  
var cellValue = controlItems[0].getValue("column1");  
appStudioWizard.showSuccessMessage(cellValue)
```

How to Configure DynamicTable Control

The DynamicTable is loaded at runtime through event handling, based on the provided configuration and data. You can define its structure in the OnBeforeLoad event, located within the Settings flyout in the Studio.



```
const tableObject = {
  columns: [
    {
      name: 'supplier_name',
      label: 'Supplier Name',
      control: {
        name: 'supplier_name',
        label: 'Supplier Name',
        dataType: 'String'
      },
      width: 500,
      hide: false
    },
    {
      name: 'team_id',
      label: 'Team',
      control: {
        name: 'team_id',
        label: 'Team',
        dataType: 'Item',
        dataSource: 'Team'
      },
      width: 400,
      hide: false
    },
    {
      name: 'effective_date',
      label: 'Effective Date',
      control: {
        name: 'effective_date',
        label: 'Effective Date',
        dataType: 'Date'
      },
      width: 300,
      hide: false
    },
    {
      name: 'contact_number',
      label: 'Contact Number',
      control: {
        name: 'contact_number',
        label: 'Contact Number',
        dataType: 'String'
      },
    },
  ],
}
```



```
        width: 200,  
        hide: false  
    }  
],  
columnWidthByPercentage: false  
};  
  
// Apply configuration to the table  
  
appStudiowizard.setTableConfiguration('dynamicTable', tableObject);
```

During the OnLoad event, actual data can be loaded into the table. The following code snippet should be placed in the OnLoad event handler within the Settings flyout in the Studio.

```
const dataArray = [  
  {  
    id: 'SUPPLIER_ID_001',  
    properties: {  
      supplier_name: 'Supplier One',  
      team_id: {  
        data_source: { name: 'Team' },  
        id: 'TEAM_ID_001',  
        keyed_name: 'Engineering Team A'  
      },  
      effective_date: '2025-06-01T00:00:00',  
      contact_number: '+1-555-100-1000'  
    }  
  },  
  {  
    id: 'SUPPLIER_ID_002',  
    properties: {  
      supplier_name: 'Supplier Two',  
      team_id: {  
        data_source: { name: 'Team' },  
        id: 'TEAM_ID_002',  
        keyed_name: 'Production Team B'  
      },  
      effective_date: '2025-06-03T00:00:00',  
      contact_number: '+1-555-200-2000'  
    }  
  },  
  {  
    id: 'SUPPLIER_ID_003',  
    properties: {  
      supplier_name: 'Supplier Three',  
      team_id: {  
        data_source: { name: 'Team' },  
        id: 'TEAM_ID_003',  
        keyed_name: 'Quality Team C'  
      },  
      effective_date: '2025-06-05T00:00:00',  
      contact_number: '+1-555-300-3000'  
    }  
  },  
  {  
    id: 'SUPPLIER_ID_004',  
    properties: {  
      supplier_name: 'Supplier Four',
```



```
    team_id: {
      data_source: { name: 'Team' },
      id: 'TEAM_ID_004',
      keyed_name: 'Logistics Team D'
    },
    effective_date: '2025-06-07T00:00:00',
    contact_number: '+1-555-400-4000'
  }
},
{
  id: 'SUPPLIER_ID_005',
  properties: {
    supplier_name: 'Supplier Five',
    team_id: {
      data_source: { name: 'Team' },
      id: 'TEAM_ID_005',
      keyed_name: 'Maintenance Team E'
    },
    effective_date: '2025-06-09T00:00:00',
    contact_number: '+1-555-500-5000'
  }
},
{
  id: 'SUPPLIER_ID_006',
  properties: {
    supplier_name: 'Supplier Six',
    team_id: {
      data_source: { name: 'Team' },
      id: 'TEAM_ID_006',
      keyed_name: 'Procurement Team F'
    },
    effective_date: '2025-06-11T00:00:00',
    contact_number: '+1-555-600-6000'
  }
}
];
// Add sample records to the table
dataArray.forEach((data) => {
  appStudioWizard.addItem('dynamicTable', data.id, data);
});
```



Aras ProApp Designer - Administration Guide



Introduction

Purpose and Target Audience

This guide describes how to use Aras ProAppDesigner for building complex UIs and configurations. It is addressed for administrators handling configuration of complex UIs and end users of those UIs.



Feature Terms and Definitions

The following Feature terms and definitions are used in this document:

Term	Definition



Content overview

This document describes how to use the Aras ProAppDesigner. This User Guide is organized into the following sections:

SECTION 1:	This section outlines the purpose and scope of this Guide, as well as its intended target audience. It supplies a brief overview of each section to help users in navigating the guide.
Introduction	
SECTION 2:	This section provides Aras ProAppDesigner overview
Overview	
SECTION 3:	This section outlines the architecture of Aras ProAppDesigner.
Architecture	
SECTION 4:	This section provides how Aras ProAppDesigner provides better performance
Performance	
SECTION 5:	Aras ProAppDesigner TOC
TOC	
SECTION 6:	This section provides how template can be designed using studio.
Template Studio	
SECTION 7:	This section provides detailed information about the wizard.
Wizard	
SECTION 8:	This section provides information about different dashboards.
Dashboards	
SECTION 9:	This section provides information about Aras ProAppDesigner customization.
Template Customization	

Aras ProAppDesigner Overview

Introduction

Built on Aras Innovator as an add-on solution. It is a low code solution to generate complex, responsive, mobile friendly UI. It has a rich set of controls for building complex and modern UI. Any multi-page application can be built rapidly in studio environment with drag-drop capabilities.

Organizes complex UI in to structured containers like Pages, Dialogs, Sections, Groups which intern enable UI to adapt any screen size. It enables creating guided UI for showing long data entry forms with conditional display on UI elements based on the configuration. It has rich features like conditional display, validation rules, default values, conditional formatting all with configuration. You don't have to be a programmer to build application with Aras ProAppDesigner.

It was built using Angular SPA framework with web services as backend. It works both with Aras standard on-premise and enterprise versions.

Aras ProAppDesigner also comes with a couple of backend components, to perform certain server-side functions. These components are explained in detail in later sections of the document.



Roles

When you import Aras ProAppDesigner package, by default it deploys Template Admin group identity. This identity needs to be assigned to all users who are going to work with the templates in Template Studio.



Typical Workflow

To create any application using Aras ProAppDesigner you need to follow these steps:

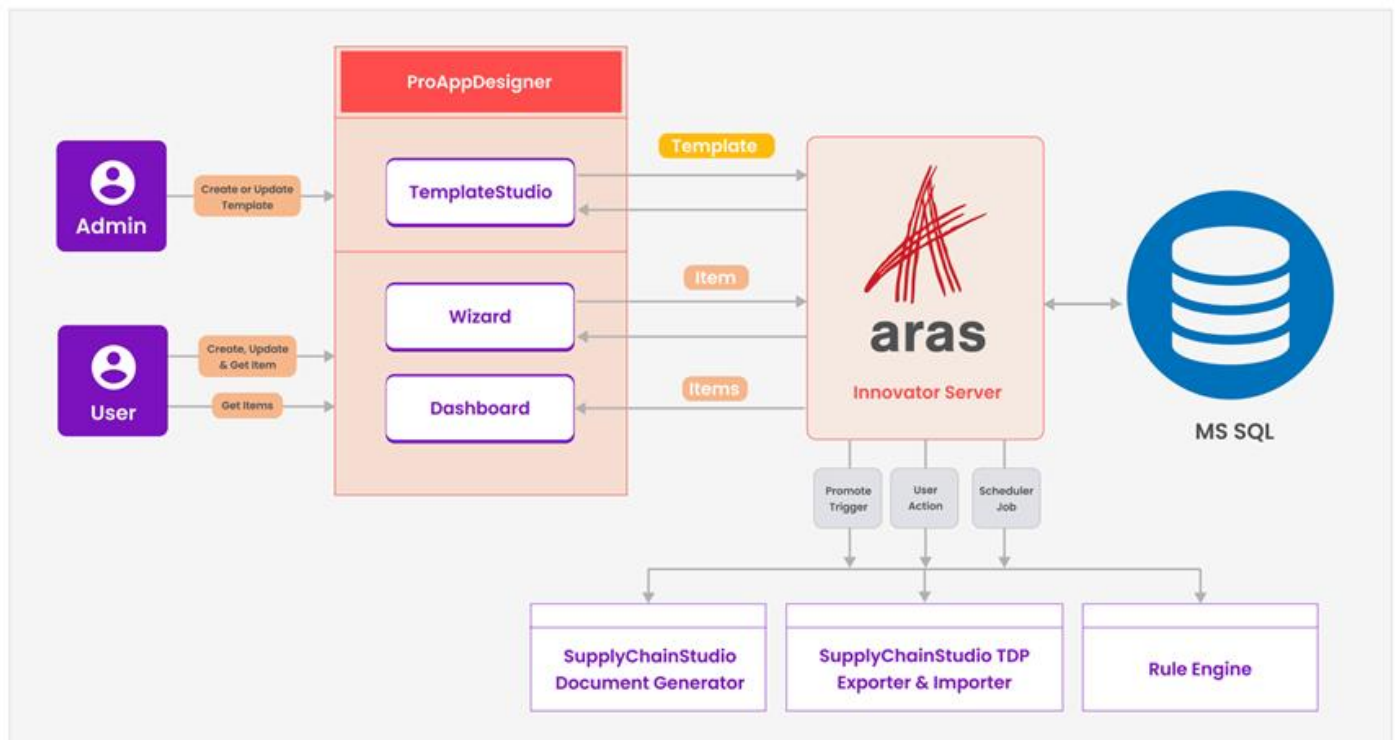
- Create DataModel (ItemTypes & RelationshipTypes) for the application.
- Create UI Template for the root ItemType and bind it to the DataModel.
- Publish the template through Lifecycle.
- Create, view and edit actions from Innovator.



Architecture

Aras ProAppDesigner comes with two bundled SPAs, one for TemplateStudio and the other for Wizard. TemplateStudio is for creating UI template for a specific ItemType. Wizard provides a runtime view of the item that is being created or modified. Both TemplateStudio and Wizard communicate with InnovatorServer by calling ServerMethods for various actions. All the UI rendered on the client side is generated on the fly based on the data fetched from the server. Communication between client UI components and server is always lightweight (only JSON) and less chatty.

Aras ProAppDesigner Architecture:



Aras ProAppDesigner Components:

ProAppDesigner

Backend Components

- Word/PDF Generator
- Technical Data Package
- Presentation Generator
- Subscriptions & Notifications
- Background & Cron Jobs
- Rule Engine
- Integration Framework

UI Components

- Template Studio (ItemType)
- Wizard (Item)
- Dashboard (Tables & Reports)
- Portal

Template Types

- Advanced View (Replaces Form)
- Enhanced View (Augments Form)
- Portal View (Portal Form)
- Type Dashboard (Type Specific)
- Global Dashboard (All Types)

Controls

Text	List	Checkbox	Image	File
Document	RichText	Item	Items	xClassification
ButtonGroup	Table	Worksheet	TreeTable	Structure
Graph	Tab	Report	Banner	Frame
HTML	CAD Viewer	PDF Viewer	Image Viewer	Widget
Signature	ProjectPlan	Calendar	Section	Group

Features

- Responsive Layout
- Computed Values
- Display Conditions
- Conditional Formatting
- Validation Rules
- Dynamic ReadOnly



Performance

Aras ProAppDesigner code is optimized to get best possible performance by using various techniques. Since Aras ProAppDesigner is written using Angular, it will never get html from server. It generates html on the client based on data received from the server. It also uses very few roundtrips to server to get the data. Once Angular app is loaded on the client side, it will be cached on the client across browser sessions. It will be updated only when a newer version of app is available on the server.



Caching

Aras ProAppDesigner gives very good performance while rendering Studio, Wizard, and Dashboards because of the caching implemented at various levels. For showing any item using Aras ProAppDesigner, it needs Template, ItemType definition, and Item data. Template is cached in Local Storage on the client, it can persist across browser sessions. It has logic implemented to update the cache if the template is modified on the server. ItemType definition is cached at Session level, it also has logic implemented to update the cache if ItemType definition has changed on server side.

Item data is cached at session level as long as is being used in the Wizard for the showing UI. In case, same item is being shown in two different Wizards, for example one in the table other one in the form, it will have only one entry in the cache. If the item is being updated in one place, updated values will be shown in other places where it is being referred. As soon as all the references for the item in the cache are cleared from the UI, item will be removed from the cache.



Progressive Loading

While loading any Item in Aras ProAppDesigner, it gets the data for only for those controls which are visible. If any control is hidden because of display condition, its data won't be fetched until it is displayed.



Table and Worksheet Benchmarking

This benchmarking was done on a QA machine with limited capacity. You will see better performance on a production server, since majority of the load time is taken by the server in fetching the data from the database.

Table Control

Item Count	Column Count	Pagination	Load Time
2000	8	Yes	Around 1 second
2000	8	No	Around 5 seconds
5000	8	Yes	Around 1 second
5000	8	No	Around 10 seconds

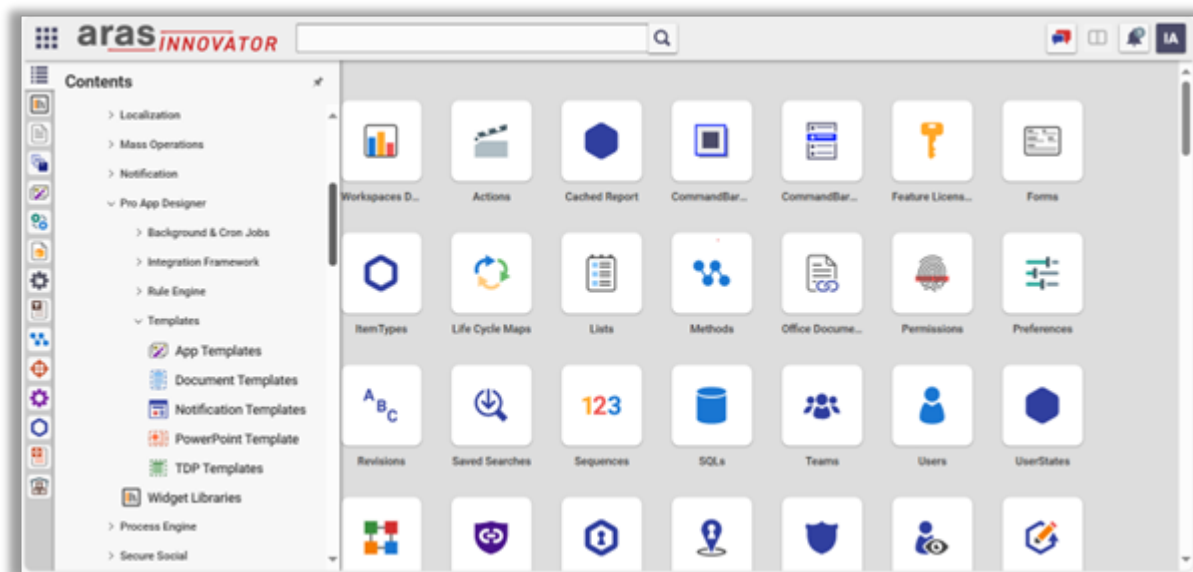
Worksheet Control

Item Count	Column Count	Load Time
2000	8	Around 4 second
5000	8	Around 8 seconds



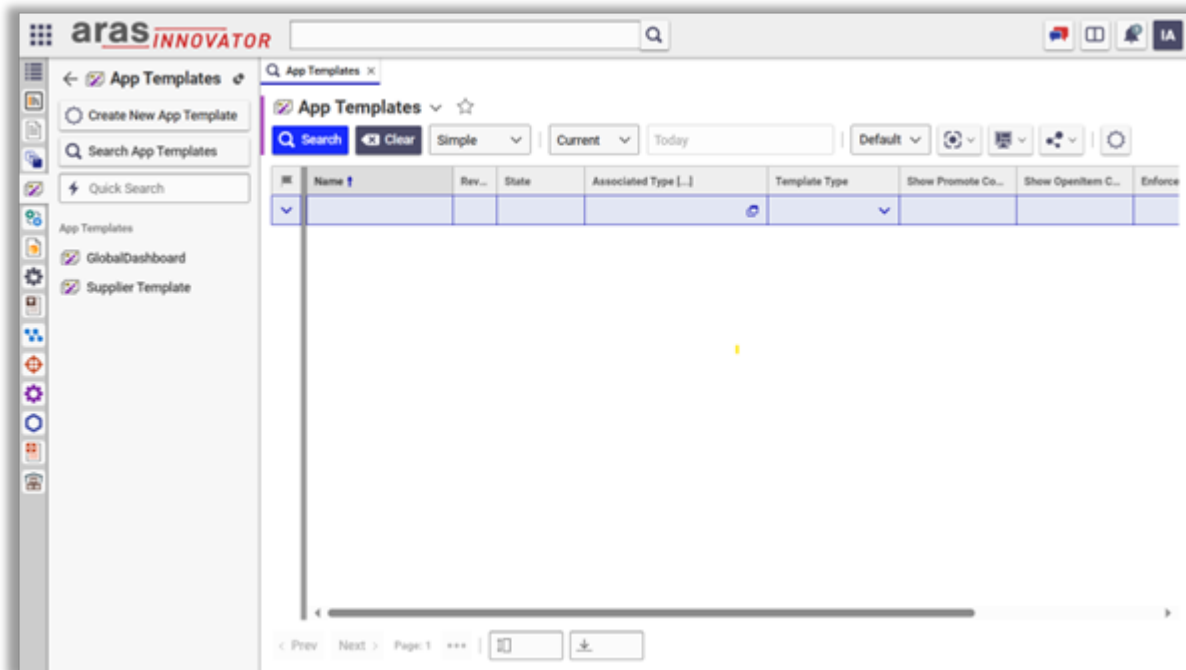
TOC

Once Aras ProAppDesigner package is installed, you will get to see a category called “Aras ProAppDesigner” under TOC. This category shows underneath AppTemplate type, upon clicking that node you will get to see all the templates existing in the system.

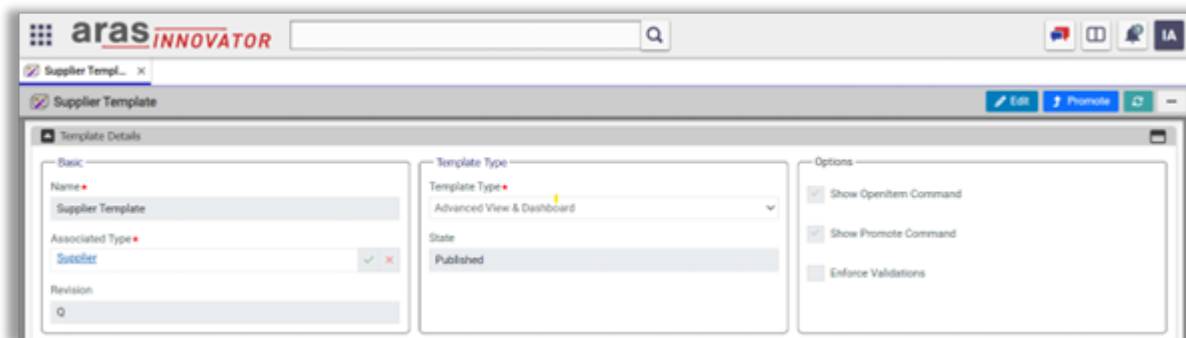


You can also create new template from the search grid:





For creating new template, you need provide name and select ItemType for which you want to create UI:



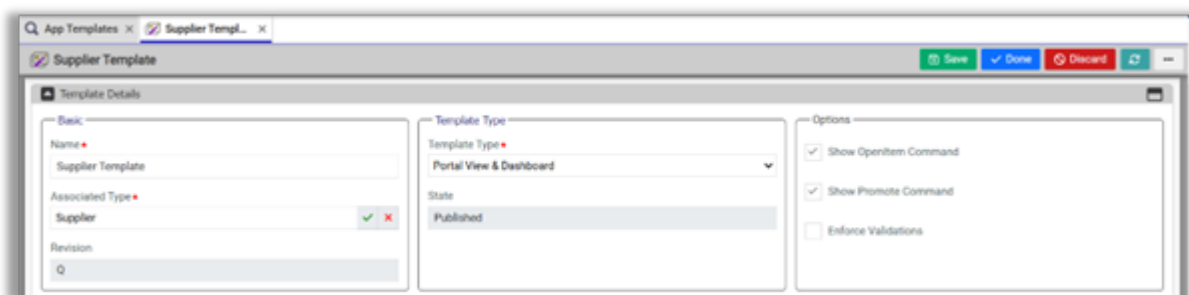
For creating template, you also need to provide TemplateType. You can select one of the following options from the dropdown based on the type of the template you want to create:

Template Type	Template Views	Description
Advanced View	Wizard, Portal Wizard	Advanced View template created for an ItemType will replace default Innovator Form.



Enhanced View	Wizard, Portal Wizard	Enhanced View template will augment existing Innovator Form of an ItemType by adding Sidebar Item.
Advanced View & Dashboard	Wizard, Dashboard, Portal Wizard, Portal Dashboard	Advanced View & Dashboard template created for an ItemType will replace default Innovator Form and adds new TOC item for Type Dashboard.
Enhanced View & Dashboard	Wizard, Dashboard, Portal Wizard, Portal Dashboard	Enhanced View & Dashboard template will augment existing Innovator Form of an ItemType by adding Sidebar Item and adds new TOC item for Type Dashboard.
Type Dashboard	Dashboard, Portal Dashboard	Type Dashboard adds new TOC item for Type Dashboard that shows dashboard tables and reports of a specific ItemType.
Global Dashboard	Dashboard	Global Dashboard adds new TOC item for Global Dashboard that shows dashboard tables and reports of various ItemTypes.
Portal View	Portal Wizard	Portal View template created for an ItemType will be used to show details of the item in the ArasPortal.
Portal View & Dashboard	Portal Wizard, Portal Dashboard	Portal View & Dashboard template created for an ItemType will be used to show details of the item adds new TOC item for Type Dashboard in the ArasPortal.
Global Theme	Theme	Global Theme template allows to create styles that can be used across all templates.

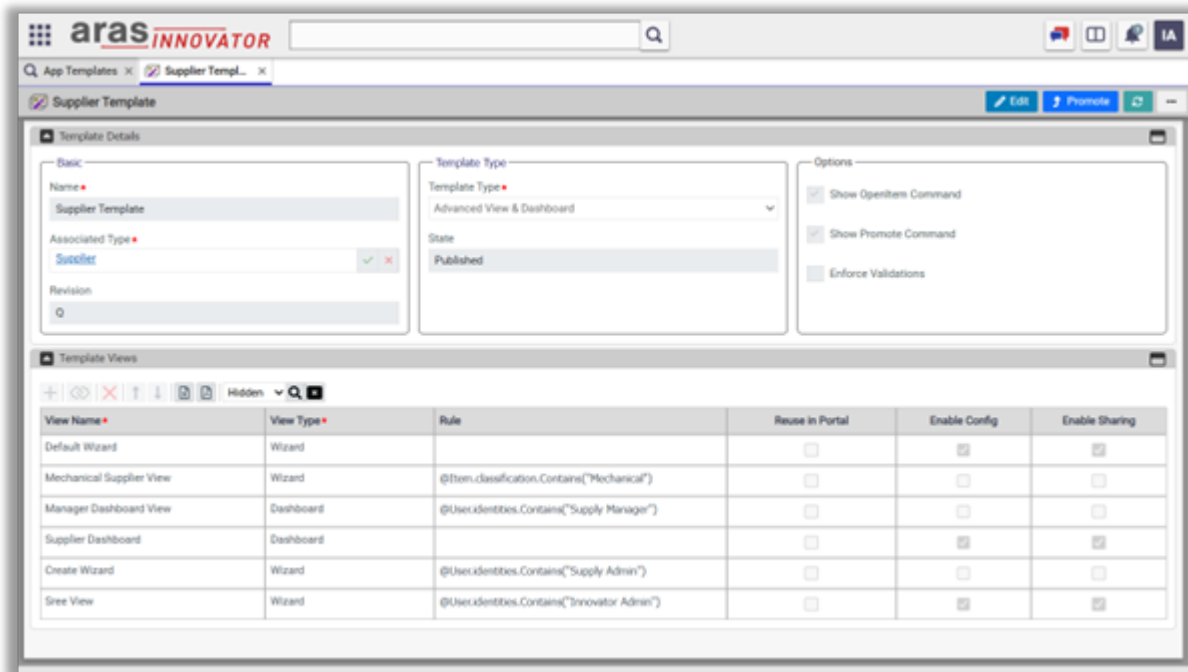
Template Type Portal View & Dashboard allows to create portal view for the ItemType, that can show limited information of the items to the external users. This view is used by Aras ProAppDesigner Portal application.



By selecting the options 'Show OpenItem Command' and 'Show Promote Command', you can control whether to show these actions in the Wizard at runtime. By selecting the option 'Enforce Validations', you can enforce, you can't navigate away from the page until all validations are fixed in the current page.

After successfully creating the template, you will be able to create views for the template using "Template Views" section. By adding a row to the table and providing "View Name", a new view will be created. For each view, you will be able to select "View Type" from the available options from the dropdown. Options will vary based on the selected "Template Type". Please refer to the above Template Type table for the allowed View Types for each Template Type. You can provide optionally "Rule" that will define the condition when this view is used when you open or edit item from the Search Grid. Rule can be defined with a Boolean Expression which contains context ItemType properties, User properties and User roles. At runtime when you use the item, these rules across multiple views will be evaluated and first matched view will be returned to show item data. In order to allow you to create, there must be at least one view with no Rule or the Rule that doesn't contain any ItemType properties in the expression. By selecting "Reuse in Portal" option for a view will allow reuse of the view in the Portal. By selecting "Enable Config" option, admin can allow end user to config his view, based on the default view shown to him at runtime. By selecting "Enable Sharing" option, admin can allow end user to share his view with other users in the system at runtime.

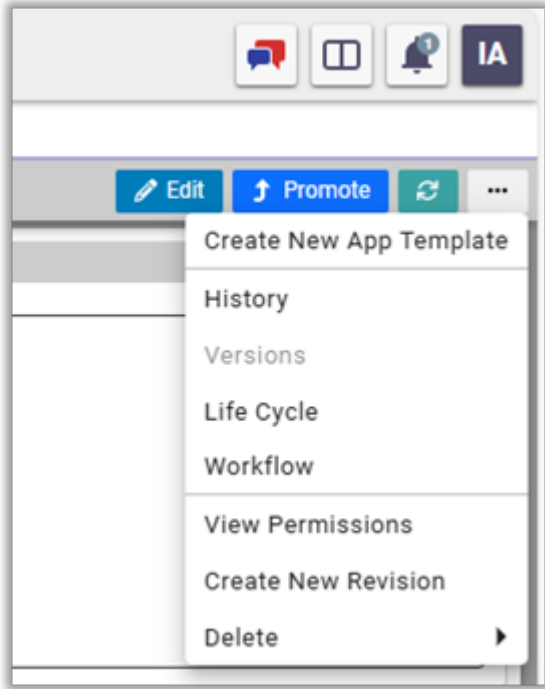




Double-clicking row or selecting it and clicking the “Open Relationship Item” command will open the view in the studio. By selecting Promote action, you will publish the template making it available for other users to use when an item is created or opened.

There are additional actions that can be performed on the template item. For example, if the template is already published, in order to update the template a new revision has to be created. You can select ‘Create New Revision’ action to create a new revision for the current template item.





Template Studio

Template studio enables creating template for a specific ItemType. ItemType properties and relationships can be used to create UI of the template. Upon publishing the template, ItemType default form will be overridden with the template.



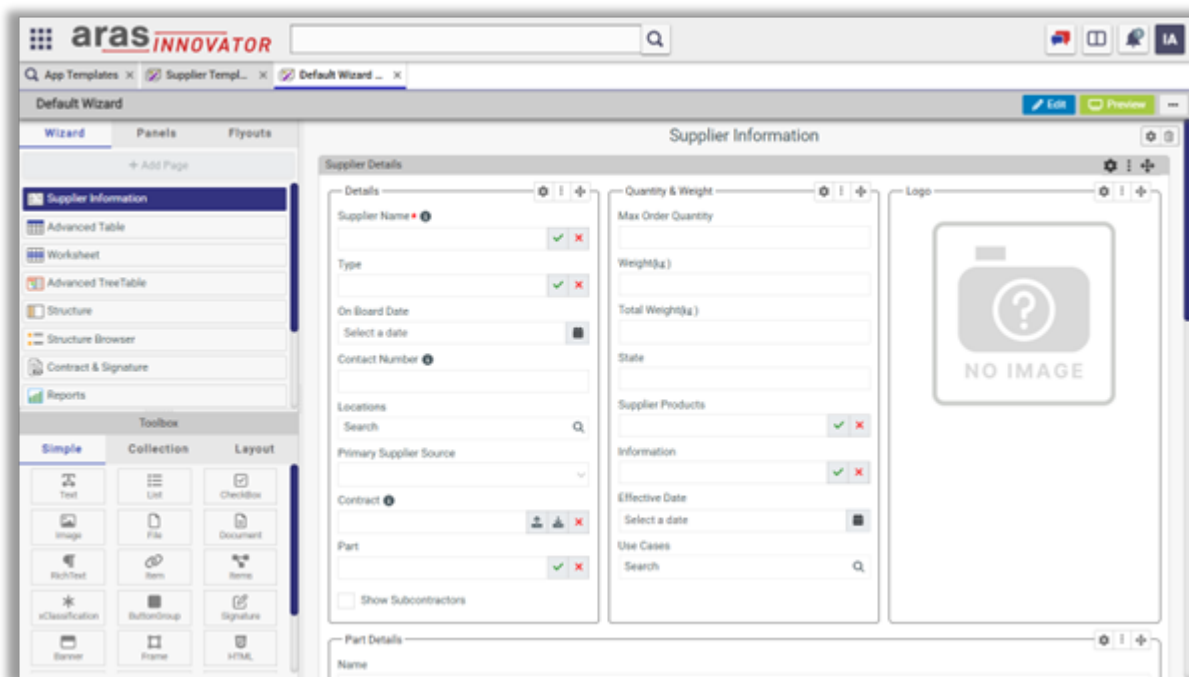
Features

- Pages and Dialogs for building single-page and multi-page applications.
- Layout controls (Sections and Groups) in Toolbox to organize controls on the pages and dialogs.
- Basic controls in Toolbox for presenting all property types from ItemType definition.
- Table control to show Related Item data in tabular form with drag & drop, conditional formatting, computed columns capabilities.
- Preview feature for reviewing changes quickly while building the template without creating Item from ItemType.
- Display Condition on all UI controls can be defined with expression to dynamically render or hide controls based on property values.
- Default Value on Text control to show computed value based on the expression built from other properties.
- Validation Rules on all controls to validate entered data before submitted request to server. Rules can be built based on the expression from the properties of the current and previous pages.
- Conditional Formatting can be defined on Text control, List control and Table control column to highlight the fields or cells with defined styles. Rules are built with expression based on thresholds set on property value.
- ButtonGroup control to define actions that can show dialogs or standard actions from Innovator like WhereUsed, StructureBrowser, DPN View etc.
- RichText control will allow you to create rich-text that can be stored as formatted text property on the item.
- Signature control can capture your signature as picture and store it as image property on the item.
- Report control can show table or chart-based reports based on QueryDefintion for local and global data.
- Structure control can show any structured data with all relationships with the help of defined QueryDefintion.
- Item control can show item property from the item. Items control can show related items from a RelationshipType with keyed names.
- Navigation panel is shown based on the defined pages and their order, provides guided UI with next and previous buttons.
- Responsive UI: Controls rendered on the page will get adjusted to the available page width to avoid showing horizontal scrollbars.



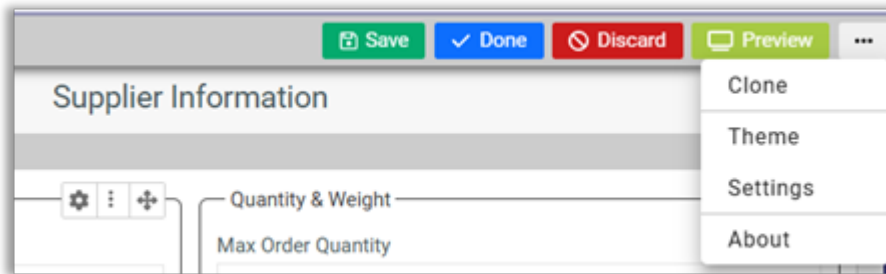
Studio Layout

When you create a new template or open an existing one, it will be shown with the following layout in the Studio. You will see at the top banner, with template name on the left and template actions on the right. On the left side, you will see sidebar with Wizard and Dialogs, where you can create and organize pages and dialogs. Beside the left sidebar, you will see Toolbox with vertically stacked controls. On the right side in the workspace, you will see selected page or dialog content.



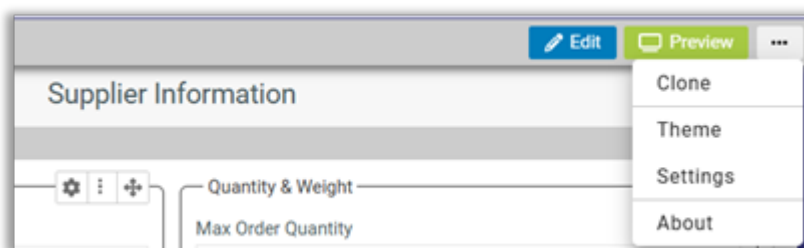
TemplateView Actions

TemplateView actions are shown in the studio in the top-right corner. When you open the template in studio with edit mode, you will see actions like Save, Done, Discard, Preview, Theme, and Help.



- **Save:** By selecting Save action, you can save the changes done so far on the template. Once changes are saved, you can't discard them using Discard action.
- **Done:** By selecting Done action, you can save the changes done so far on the template and put the template in read-only mode.
- **Discard:** By selecting Discard action, you can discard all the changes done so far on the template after the last save and put the template in read-only mode.
- **Preview:** By selecting Preview action, you can test your template in preview quickly without creating real item.
- **Clone:** By selecting Clone action, you can create a new template view based on the current view. For cloning, you need to provide "View Name" and "View Type" in the flyout.
- **Theme:** By selecting Theme action, you can set various control type styles at the template level.
- **Settings:** By selecting Settings action, you get to see template level settings in the flyout.
- **About:** By selecting About action, you get to see current Aras ProAppDesigner version and the build date.

When you open template in studio with read-only mode, the following actions will be shown.



- **Edit:** By selecting Edit action, you can apply the lock on the template and get exclusive access to make changes on the template.

TemplateView Settings

TemplateView settings are shown in the flyout when you click on Settings action. Flyout shows TemplateView name, commands that can be shown at runtime on top of item details page. It also allows you to configure events like OnLoad and OnSave.

Name	Visibility	On Click	Can Show	Actions
Edit	Show In ButtonGroup			✕ ⬇
Save	Show In ButtonGroup			✕ ⬇
Done	Show In ButtonGroup	✓		✕ ⬇
Promote	Show In ButtonGroup	⬆		✕ ⬇
Discard	Show In ButtonGroup	🗑		✕ ⬇
Open Item	Show In ButtonGroup	📄		✕ ⬇
Discussion Panel	Show In ButtonGroup	💬		✕ ⬇
Refresh	Show In ButtonGroup	🔄		✕ ⬇
Create New	Show In Dropdown	➕		✕ ⬇
Separator	Show In Dropdown			✕ ⬇
History	Show In Dropdown	📅		✕ ⬇
Versions	Show In Dropdown	📅		✕ ⬇
Life Cycle	Show In Dropdown	📅		✕ ⬇
Workflow	Show In Dropdown	📅		✕ ⬇
Separator	Show In Dropdown			✕ ⬇
View Permissions	Show In Dropdown	🔒		✕ ⬇
Create New Revision	Show In Dropdown	➕		✕ ⬇
Delete	Show In Dropdown	🗑		✕ ⬇
Separator	Show In Dropdown			✕ ⬇
Config	Show In Dropdown	🔧		✕ ⬇
Share	Show In Dropdown	🔗		✕ ⬇

- **Name:** It shows the name of the current TemplateView. It is shown as read-only.
- **Commands:** It allows you to configure actions shown on the top of item details page. By default, it shows standard commands for which you can change visibility and icons. With visibility setting, you can configure the action to be shown as part of ButtonGroup or Dropdown. You can hide the action by selecting visibility as Hide. You can also add custom commands by defining “On Click” and “Can Show” scripts.



- **On Load:** OnLoad event allows you to write custom logic that will be executed after the item data is loaded. Please refer to the Template Customization section for more details.
- **On Save:** OnSave event allows you to write custom logic that will be executed after the item is saved. Please refer to the Template Customization section for more details.



Control Actions

Every control has following action icons shown upon selecting the control. These actions are available on the control only when the template is in the edit mode:



- **Settings:** By clicking the settings icon, settings flyout will open on the right side from where you can change various settings of the selected control.
- **Miscellaneous:** By clicking three-dot icon, two actions will show in the dropdown. Using delete action, you can delete the selected control from the page. Using MoveTo action, you can move the control in the current page or to any other pages under different parent.
- **DragDrop:** By holding drag-drop icon, you can reposition the selected control within its immediate parent.

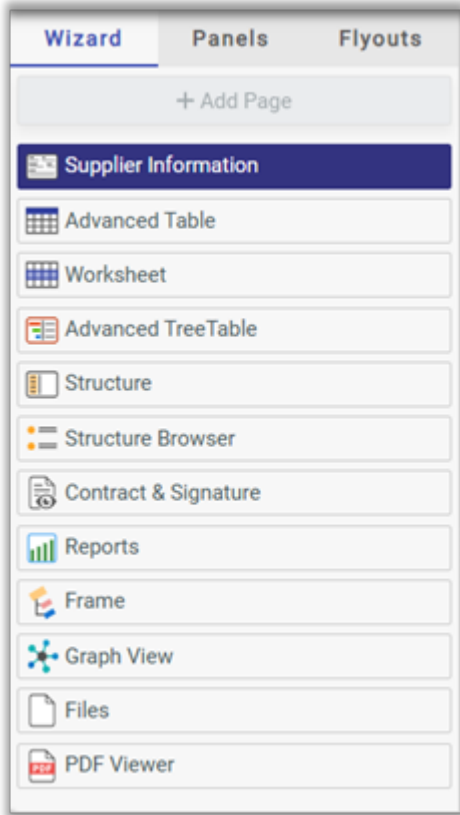
Template Studio Sidebar

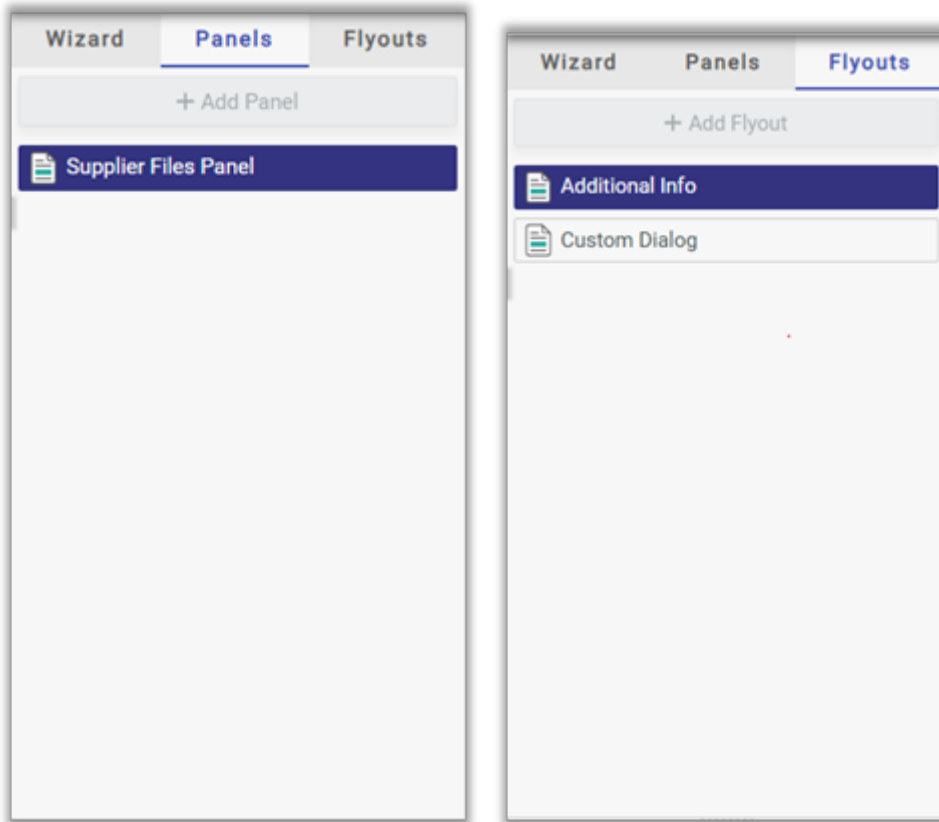
Template Studio Sidebar shows Wizard, Panels & Flyouts in the upper section and Toolbox in the lower section. Wizard tab shows a list of pages that can be shown at runtime when you open the item. In Aras ProAppDesigner you can create single page or multi page applications. When you create only one page, at runtime when the page is shown in the Wizard, it shows item's keyed name as page title. In case of multipage application, all the pages are shown in the Navigation bar with their page titles.

Panels tab shows list of panels that can be used while adding panel control in any group control on the page. Panel looks like a page in all aspects, and it can be created just like a page with only exception that it can be used under a group control in the page to create complex layouts. Same panel can be used on many pages, for showing same header and footer across all pages of the Wizard.

Flyouts tab shows a list of flyouts that can be used at many places in the Wizard by associating them to click handlers of various controls like buttons and toolbar commands. Flyout looks like a page in all aspects, and it can be created just like a page with only exception that it can be shown as a popup with submit, cancel and reset buttons.







You can create new page by clicking on 'Add Page' button in the sidebar. You can drag-drop pages to reorder created pages. Each page on the right side is shown with actions in the top-right corner with settings and delete icon.

- **Name:** It is used to uniquely identify page inside the wizard.
- **Label:** It is used to show title on the page. It can be localized by defined Languages and Locales inside innovator. Label also allows you to set the icon for the page.
- **Access:** It allows you to define roles which can access or edit the page. Roles (group or alias identities) can be at Show and Edit level. When page is hidden because you do not have defined role, properties used inside this page are not even fetched from server.
- **Read Only:** Based on the option selected for Read Only, whole page can become read-only even if you can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the page read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If the returned value is true, the whole page will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the page will be shown or hidden from the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the page if expression is evaluated to true. Expression will be evaluated based on the cached property



values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is false, whole page will be hidden.

- **Save On Navigate:** If this option is selected as Yes, all updates made so far are saved to server when you navigate away from the page.
- **OnLoad:** OnLoad event allows you to show custom UI in the page if the page has HTML or Frame controls. Please refer to the Template Customization section for more details.

Panels can be created or managed in the same way as wizard pages. Panel can be placed just like any other control under the group through panel control available in the Toolbox. Panel can also be used as a tab inside Tab control.

Flyouts can be created or managed in the same way as wizard pages. The difference is, flyout can be shown as a popup at runtime based on the button click event. In order to show flyout as a popup,

you need to place a button on the page and associate its OnClick handler with the flyout. Flyout has one additional setting to set the Flyout Width in percentage with respect to the page width.

- **OnSubmit:** OnSubmit event allows you to write custom logic that will be executed when the page is submitted. It can be mainly used if the dialog has any custom UI and if you want to save the changes to page or any external system. Please refer to the Template Customization section for more details.

At runtime, flyout is shown as popup, it shows Reset, Ok and Cancel buttons at the bottom. Reset button will reset all values to their original values if you made any changes, Ok button will commit the changes to the client cache and Cancel button will close the flyout without committing the change to the cache.

Additional Info ✕

Generation

Created On
Select a date 📅

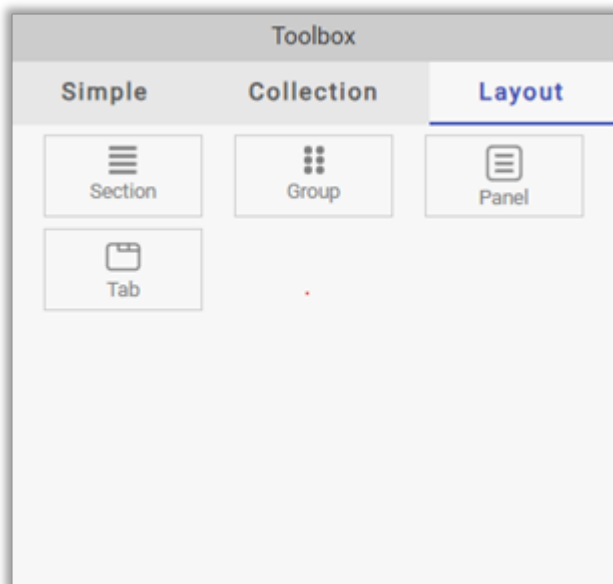
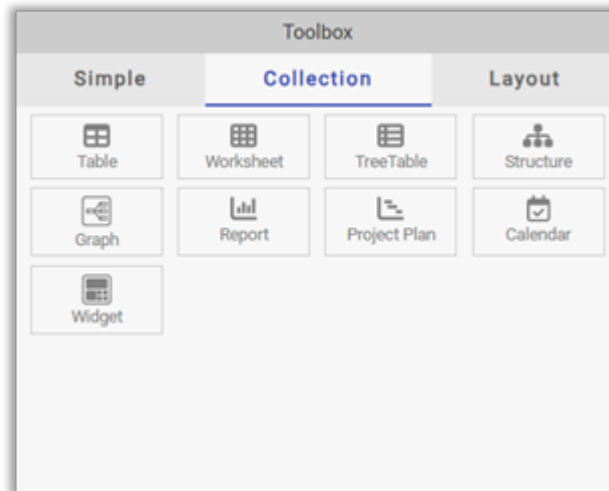
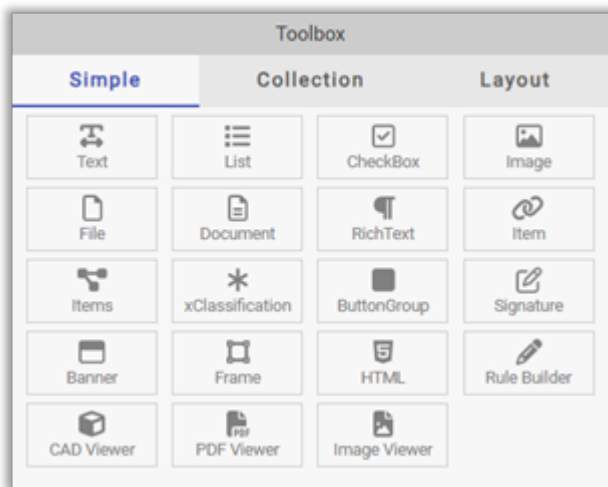
Modified On
Select a date 📅

Release Date
Select a date 📅



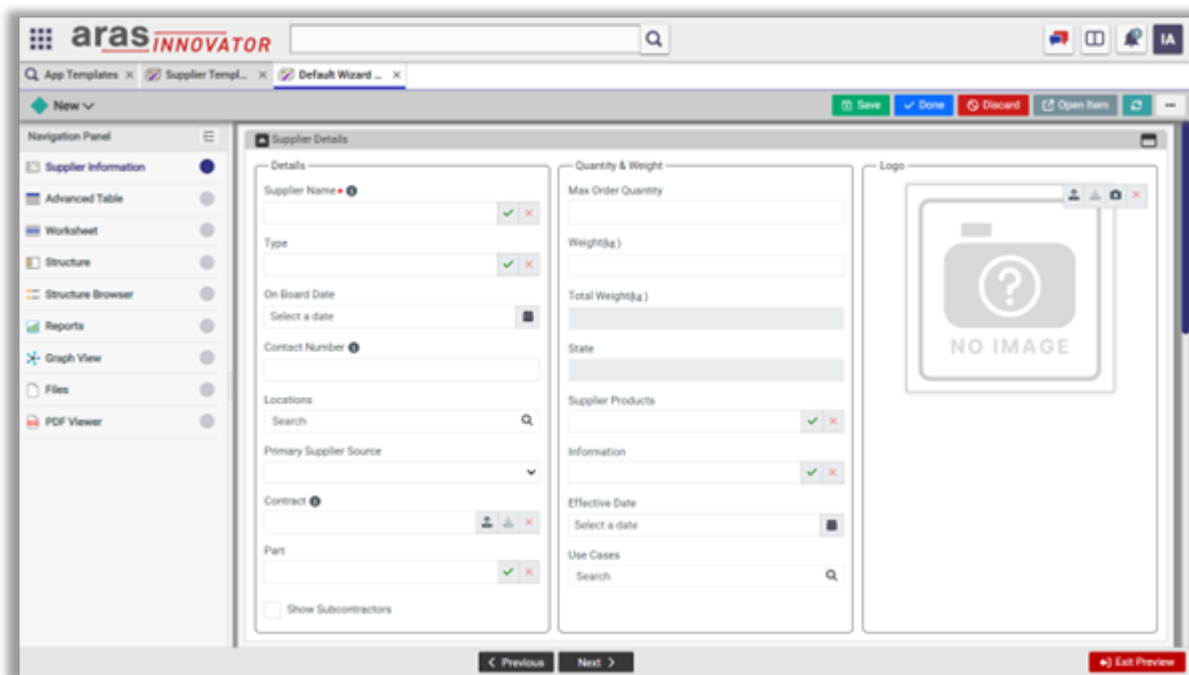
Toolbox

Toolbox is shown vertically in the studio with all available controls. In order to place any control in the page, you need to select container where you need to place the control. When container or control is selected from the page, only those controls which can be added to the page with respect to selected control will be enabled in the Toolbox. Each control from the Toolbox will be explained in detail later in the following sections.



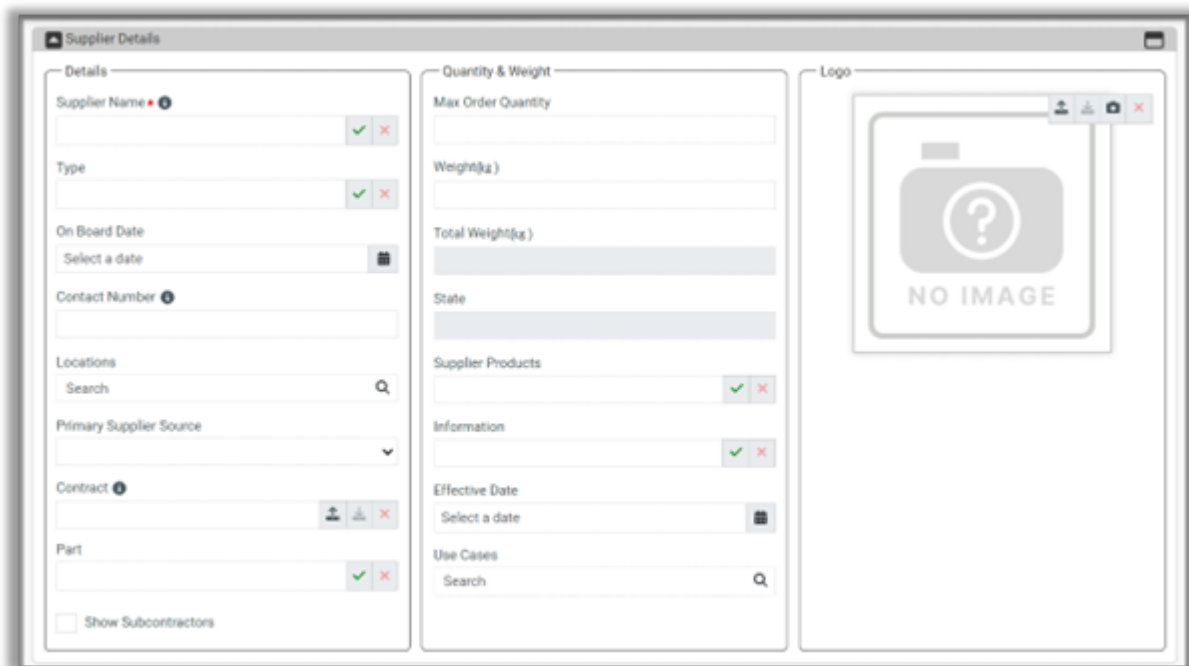
Template Preview

When you select Preview action, it shows how wizard looks at runtime when you really work with the item without creating the real item. It is very handy to test all your changes quickly without creating the item before publishing the template. In preview mode, you can check how your pages are shown with their UI with all display conditions, validation rules, default value (computed fields), and conditional formatting. In case any specific control can't be rendered because of missing context item, it shows the message indicating the same.



Section Control

Section control is a layout control to organize group controls placed inside it. Section controls are vertically stacked one over the other. Section controls are the top-level container controls that exist in pages and dialogs. By defining display condition or access permissions, section control can be shown or hidden based on the evaluation criteria. When section control is created on the page, by default a group control is placed inside it.



Section control behavior or characteristics can be configured through its settings flyout by selecting gear-icon from top-right corner. You can place the section anywhere on the page among other sections using drag-drop icon. By selecting 3-dot icon, it shows Delete and MoveTo options to delete or move the section to other pages.

- **Name:** It is used to uniquely identify control inside the template.
- **Label:** It is used to show the title of the section. It can be localized by defined Languages and Locales inside innovator.
- **Associated Type:** Following options can be selected for the Associated Type. By default, Associated Type is set as Context Type.

- **Associated Type:** Context Type

When Context Type is selected, while adding control to a group of sections, properties from the template ItemType will be shown for binding with the control.

- **Associated Type:** Reference Type

When Reference Type is selected, a dropdown with a list of all item properties from the contextual ItemType will be shown. By selecting a value for the dropdown, you will be shown with properties of the selected ItemType inside the groups of the section for binding with controls.

- **Associated Type:** Type

When Type is selected, while adding a control to a group of sections, properties from the selected ItemType will be shown for binding with the control. In this case, the section will be shown in the UI only when the item of the selected ItemType is set as context item on the section.

- **Associated Type:** Type (UI)

When Type (UI) is selected, while adding control to a group of sections, properties from the selected ItemType will be shown for binding with the control. In this case, the section will always be shown in the UI with empty values. Selected ItemType will only be used to show custom form with properties defined on the selected ItemType. You must write your own custom script to save these values to the server.

- **Access:** It allows you to define roles which can access the section. Roles (in the form group identities) can be at Show and Edit level. When section is hidden because you do not have defined role, properties used inside this section are not even fetched from server.
- **Read Only:** Based on the option selected for Read Only, section can become read-only even if you can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the section read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If the returned value is true, the whole section will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the section will be shown or hidden from the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the section if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If the returned value is false, the whole section will be hidden.
- **Border:** It allows you to show or hide the border and legend of the section. In case if you choose to show the border, you can select the border style using Color, Background, Type and Thickness settings. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **Font:** It allows you to set the required font, size, color for showing the legend of the section. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **Margin:** It allows you to set required top, right, bottom, left margins with respect to adjacent controls. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **Padding:** It allows you to set the required top, right, bottom, left padding with respect to parent control. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.



Group Control

Group control is a layout control used to organize child controls placed inside it. Group controls are horizontally stacked beside each other inside a section control. When multiple group controls are placed inside section control, they equally take the available width. When you resize the page on desktop, based on the available width they can be shown in single row or multiple rows inside a section control. In mobile view, they will be vertically stacked inside a section control. By defining display condition or access permissions, group control can be shown or hidden based on the evaluation criteria. When a section control is created on the page, by default a group control is placed inside it.

The screenshot displays two side-by-side group controls within a larger container. The left group control is titled 'Details' and contains several input fields and controls:

- Supplier Name:** A text field containing 'Aras'.
- Type:** A dropdown menu showing 'Electrical/Assembly' with a green checkmark and a red 'X' button.
- On Board Date:** A date picker field showing '12/7/2023'.
- Contact Number:** A text field containing '444-444-4237'.
- Locations:** A dropdown menu showing 'India' with a red 'X' button and a search field below it.
- Primary Supplier Source:** Radio buttons for 'Make' (selected) and 'Buy'.
- Contract:** A text field containing 'Aras Innovator 33 - Publishing Service Setup Guide.pdf' with upload, download, and delete icons.
- Part:** A dropdown menu showing 'bicycle' with a green checkmark and a red 'X' button.
- Show Subcontractors:** A checked checkbox.

The right group control is titled 'Quantity & Weight' and contains:

- Max Order Quantity:** A text field containing '2'.
- Weight(kg):** A text field containing '500'.
- Total Weight(kg):** A text field containing '1,000.00', which is highlighted in yellow.
- State:** A greyed-out text field.
- Supplier Products:** A dropdown menu showing 'P100' with a red 'X' button and a green checkmark.
- Information:** A dropdown menu showing 'Replicator System Schematic.pdf' with a green checkmark and a red 'X' button.
- Effective Date:** A date picker field showing 'Select a date'.
- Use Cases:** A search field with a magnifying glass icon.



Group control behavior or characteristics can be configured through its settings flyout by selecting gear-icon from top-right corner. You can place the group anywhere inside the section among other groups using drag-drop icon. By selecting 3-dot icon, it shows Delete and MoveTo options to delete or move the group within current page or to other pages.

- **Name:** It is used to uniquely identify control inside the template.
- **Label:** It is used to show the title of the group. It can be localized by defined Languages and Locales inside innovator.
- **Associated Type:** Following options can be selected for the Associated Type. By default, Associated Type is set as Context Type.
 - **Associated Type:** Context Type
When Context Type is selected, while adding control to a group of sections, properties from the template ItemType will be shown for binding with the control.

- **Associated Type: Reference Type**
When Reference Type is selected, a dropdown with a list of all item properties from the contextual ItemType will be shown. By selecting a value for the dropdown, you will be shown with properties of the selected ItemType inside the groups of the section for binding with controls.
- **Associated Type: Type**
When Type is selected, while adding a control to a group of sections, properties from the selected ItemType will be shown for binding with the control. In this case, the section will be shown in the UI only when the item of the selected ItemType is set as context item on the section.
- **Associated Type: Type (UI)**
When Type (UI) is selected, while adding a control to a group of sections, properties from the selected ItemType will be shown for binding with the control. In this case, the section will always be shown in the UI with empty values. Selected ItemType will only be used to show custom form with properties defined on the selected ItemType. You must write your own custom script to save these values to the server.
- **Access:** It allows you to define roles which can access the group. Roles (in the form group identities) can be at Show and Edit level. When group is hidden because you do not have defined role, properties used inside this group are not even fetched from server.
- **Read Only:** Based on the option selected for Read Only, group can become read-only even if user can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the group read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is true, the whole group will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the group will be shown or hidden in the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the group if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is false, whole group will be hidden.
- **Width:** It allows you to set the width for the group. Width can set as 'Auto' or 'Explicit'. If it is set as 'Auto', the parent section will give equal widths for all the groups defined with the 'Auto' option from the available width. If it is set as 'Explicit', it gives you the option to set the width in percentage from the available width of the parent section. All the groups defined with 'Explicit' option under the same section will first be given requested widths from the available width, and remaining groups with 'Auto' option will get equal portions from the remaining width.
- **Border:** It allows you to show or hide the border and legend of the group. In case if you choose to show the border, you can select the border style using Color, Background, Type and Thickness settings. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **Font:** It allows you to set the required font and size for showing the legend of the group. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **Margin:** It allows you to set required top, right, bottom, left margins with respect to adjacent controls. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **Padding:** It allows you to set the required top, right, bottom, left padding with respect to parent control. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **On Save:** OnSave event allows you to write custom logic that will be executed when the data shown in the group is saved. It can be mainly used if the group shows the data related to an item that is different than the item shown on the page. Please



refer to the Template Customization section for more details.



Text Control

Text control displays properties of datatypes String, Text, Integer, Float, Decimal, Date, Sequence and Classification. Text control honors Format Specifier set on the ItemType definition. Text control honors the default value set through ItemType definition. You can also set default value for Text control by defining text expression with properties from other controls from the current page or previous pages. Text control has additional settings like Display Condition, Validation Rules, Conditional Formatting etc. that are covered in the subsequent sections.

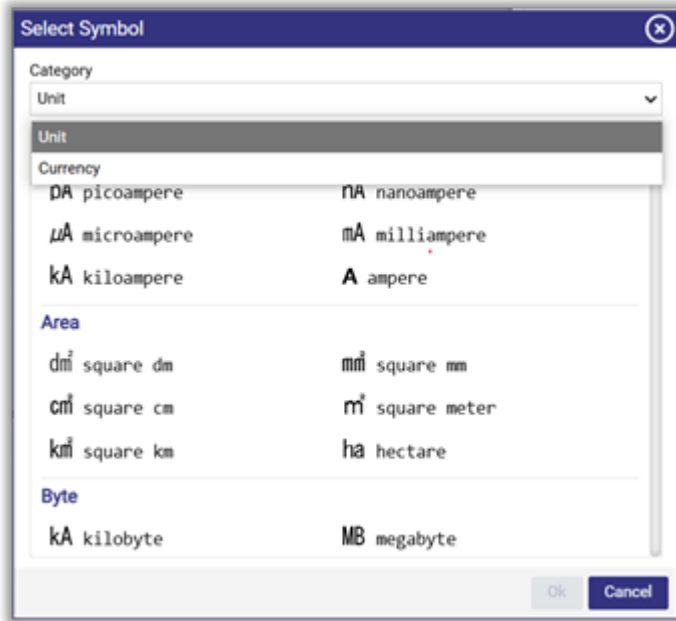
- **Name:** It is used to uniquely identify control inside the template. For text control, name is derived from the property bound to the control.
- **Label:** It is used for showing title on the text control. For text control, label is derived from the property bound to the control. If property label is localized inside innovator, at runtime localized label is pulled from the property definition.

- **Data Type:** It allows type of data that you can entry in the text control. It is derived from the property bound to the control.
- **Tooltip:** Tooltip will be shown when you hover the mouse on the information icon on the form field. It is derived from the property bound to the control.
- **Help Text:** Help Text flyout will be shown when you click on the information icon on the form field label. It is derived from the property bound to the control.
- **Format Specifier:** It shows the format in which data should be rendered in the control. It is derived from the property bound to the control.
- **Unit:** Unit setting will be shown only for the numeric properties associated with the control. Using this setting, you should be able to set units for Weight, Length, Area etc. through the Select Symbol dialog.

The screenshot shows the 'Text Settings' dialog box with the following fields and values:

- Name (maximum 32 characters): total_weight
- Label: Total Weight
- Data Type: Decimal
- Tooltip: (empty)
- Help Text: (empty)
- Format Specifier: (empty)
- Unit: kg (highlighted with a red border)
- Access: Show
- Computed Value: Max Order Quantity*Weight
- Read Only: (radio buttons for Yes, No, Expression, Script)

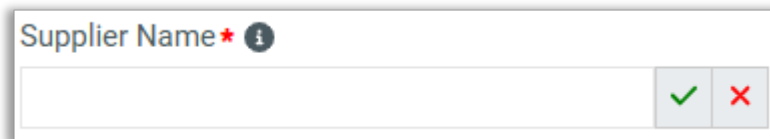
Buttons for 'Ok' and 'Cancel' are located at the bottom right of the dialog.



- **Control Style:** Text control can be rendered as textbox, text area, rule by setting its style as single line, multi-line, masked, expression, or rule.



Control Style: Single Line



Control Style: Multi Line

- **Access:** It allows you to define roles which can access control. Roles (in the form group identities) can be at Show and Edit level. When control is hidden because you do not have defined role, property associated with this control is not even fetched from server.
- **Computed Value:** It allows you to show default value when control is rendered. Default value can be computed based on the expression; in the expression you can reference other properties from the current page or previous pages. If you create an expression for default value and make control as read-only, it can dynamically change its value based on what you enter in other controls on the page.
- **Read Only:** Based on the option selected for Read Only, control can become read-only even if you can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the control read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is true, the control will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the control will be shown or hidden in the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the control if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If the returned value is false, the control will be hidden.
- **Validations:** By default, a row is displayed under the Validations section. By clicking the cell under Rule, a condition or script can be set to validate the data entered in the control. Normally validations are executed when you click on Save, Done, Next buttons from the wizard. While

you are defining the rule, you need to set the message that you want to display to user when rule is not satisfied. Message can also be a text expression; you can use current control or other control values to display dynamic text to the user. You can also set locale specific messages for the languages you configured. While defining the rule, you also need to specify Type of the rule as Error or Warning. Additional rules can be added by clicking on Add (+) action on the previous row after which you need to add the rule. If you define multiple rules, all rules should be satisfied to move forward. You can remove or reposition the rules using the respective icons from actions column.

The following example shows validation rule of type Error. When validation rule fails, error will be shown in the popup dialog with options to ignore and move forward or fix the issues before moving forward. If you move forward without fixing validation errors for that page, in the navigation panel, page title will be shown with red circle with tick icon to indicate there are validation errors in the page.

Validations			
Rule	Message	Type	Actions
Supplier Name...	*Supplier name can't b...	Error ▾	+ ✎ ✕ ⛶



Validation Rule ✕

Rule

Expression Script ?

Control Search 🔍	Operator Contains ▼	Value <div style="border: 1px solid #ccc; height: 20px;"></div> +
--	---	---

Expression AND OR () 🗑️

Supplier Name NotEqualTo ""

Message

Language English ▼

Expression Control + 🗑️

*Supplier name can't be empty"

Ok
Cancel

Validations Failed ✕

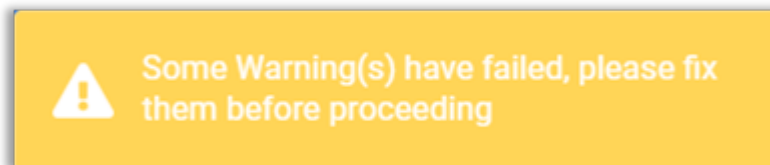
Validation rules defined on the controls have failed. Do you want to fix them before proceeding?

Yes
No

The following example shows validation rule of type Warning. When validation rule fails, warning will be shown in the flash notification with orange background color. Warnings can be ignored by clicking on Save, Done, Next buttons second time to move forward with the warning.

Validations			
Rule	Message	Type	Actions
Contact Number...	*Contact number can't...	Error ▼	+ ✎ ✖ ⚡
Contact Number...	*Contact number shou...	Warning ▼	+ ✎ ✖ ⚡



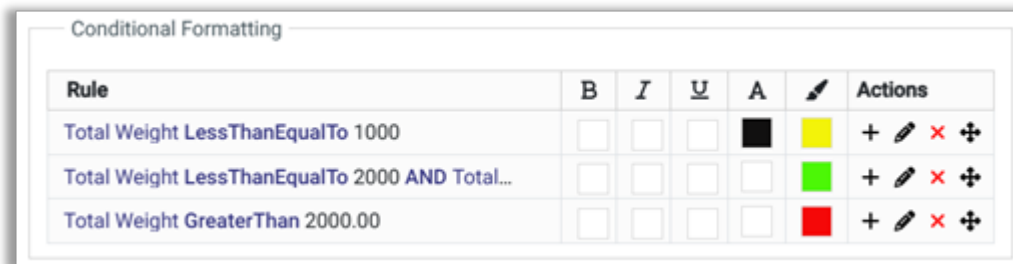
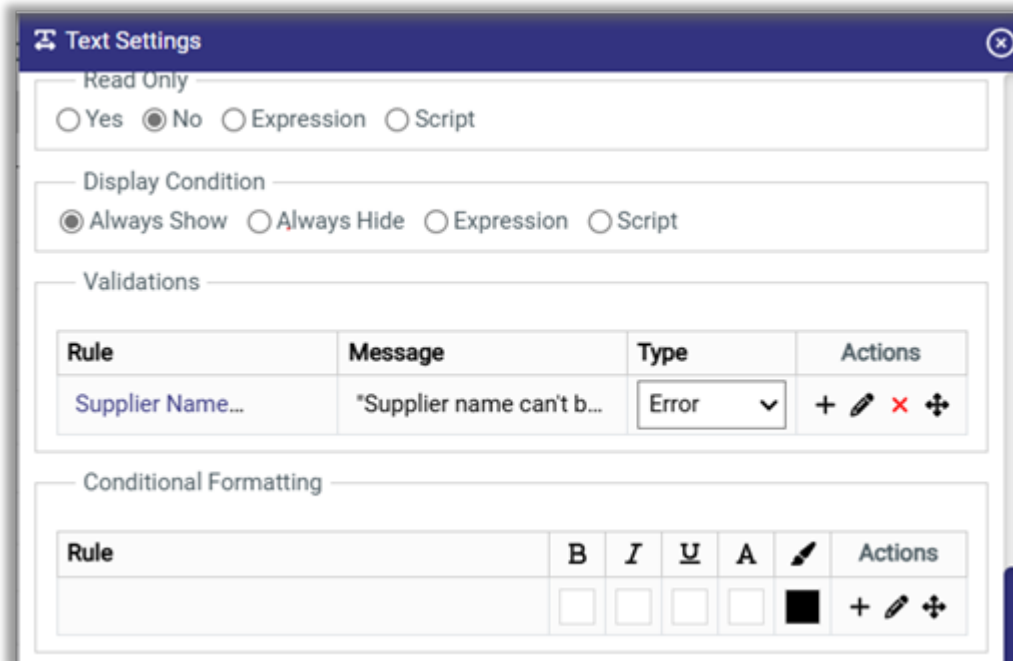


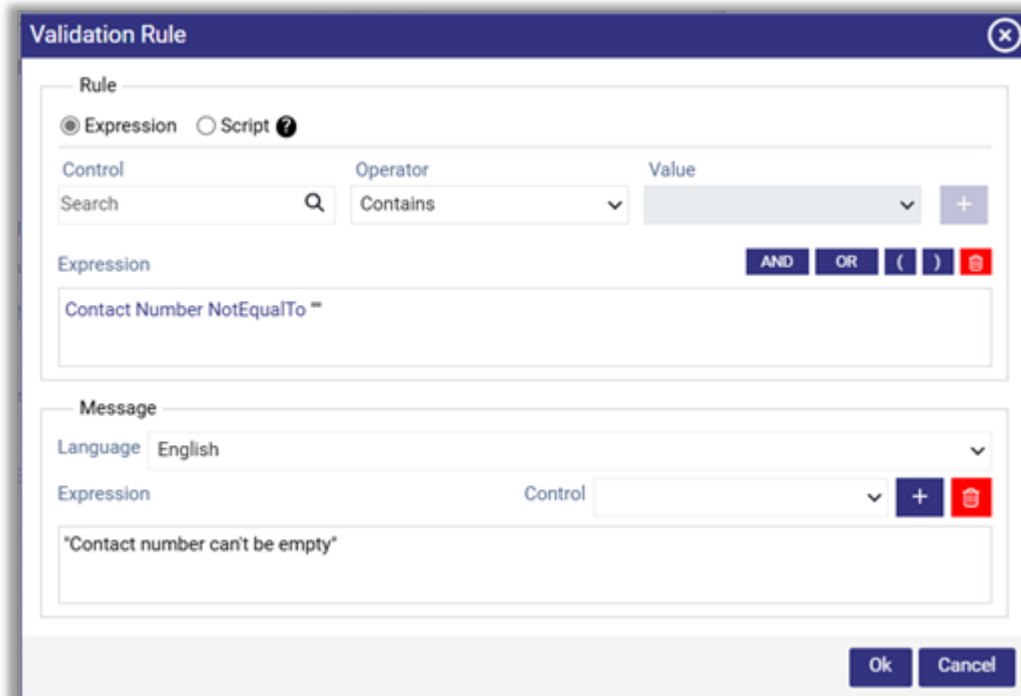
Validation rule can be defined using Script option. This will be useful for defining validation based on some complex logic rather than simple expression. Script can be written in JavaScript by accessing control values using API. Script can contain the logic to access data from the server to perform complex validation. Script should return a Boolean value to indicate whether the validation is successful.

- **Conditional Formatting:** By default, a row is displayed under Conditional Formatting section. By clicking the cell under Rule, a condition or script can be set to display specific styles for the data displayed in the control if the condition is evaluated to true. Additional rules can be added



by clicking on Add (+) action on the previous row after which you need to add the rule. Condition formatting rules are evaluated at runtime in the order in which they are defined from first to last, whenever the first rule is satisfied it stops and won't execute remaining rules. Because of that, extra care should be taken while defining the order of the rules. You can edit the existing rule by clicking on action icon null . You can remove or reposition the rules using the respective icons from actions column.





Conditional Formatting rule can be defined using Script option. This will be useful for defining the rule based on some complex logic rather than simple expression. Script can be written in JavaScript by accessing control values using API. Script can contain the logic to access data from the server to execute complex rule. Script should return a boolean value to indicate whether the rule is successful.

- **Reset Dependency:** It allows you to reset value of the control to the default if any of the controls defined in “Reset Dependency” have their value changed at runtime.
- **Font:** It allows you to set the required font and size for the value in the control. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **Margin:** It allows you to set required top, right, bottom, left margins with respect to adjacent controls. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **On Load:** OnLoad event allows you to write custom logic that will be executed after the value in the control is loaded. Please refer to the Template Customization section for more details.
- **On Change:** OnChange event allows you to write custom logic that will be executed after the value in the control is updated. Please refer to the Template Customization section for more details.

List Control

List control can be used to display properties of datatypes List, Filter List and Multi Value List. While displaying list, Aras ProAppDesigner uses same underlining list with options defined in the Innovator. List control comes with Control Style setting, with this, options can be shown as RadioButtonGroup in case of single selection list or CheckboxGroup in case of multi selection list. If you don't set the Control Style, default Dropdown will be used to show the options. List control honors the default value set through ItemType definition.

- **Name:** It is used to uniquely identify control inside the template.
- **Label:** It is used to show label on the control. It can be localized by defined Languages and Locales inside innovator.
- **Data Type:** It shows the type of the property as defined in the ItemType definition. It can be List, Filter List or Multi Value List, and it is shown as disabled.



- **Tooltip:** Tooltip will be shown when you hover the mouse on the information icon on the form field. It is derived from the property bound to the control.
- **Help Text:** Help Text flyout will be shown when you click on the information icon on the form field label. It is derived from the property bound to the control.
- **Control Style:** You can configure List control style as Radio Group in case of List or Check Group in case of Multi Value List.

When Radio Group or Check Group is selected as a style, Orientation setting will be shown to arrange options either horizontally or vertically.

List control with Radio Group Control Style:

Multi List control with Dropdown Control Style:

- **Selection Mode:** It shows whether the current List control is a Single Value or Multi Value List. Selection mode is shown as disabled; it shows the list type as defined in the ItemType definition.
- **Document Associations:** It allows you to associate a document with each option of the List. These associations will be used while generating word file based on what options have been selected from the List. If this List property is used as ContentControl in the word template, ContentControl will be replaced with the documents selected through List options.

Document Associations

Show Document Associations

Option	Item	File Name	Actions
Albania			
Argentina			
Armenia			
Australia			
Austria			
Bahamas			
Belgium			
Bolivia			
Brazil			
Bulgaria			
Canada			
Chile			
China			
Colombia			
Congo			
Costa Rica			

Select Document

Document Type
Document

Property Relationship

Select Relationship Type
Document File

Select Property of Relationship
Related File

Search Keyword (Displays only top 100 matches)
Search by document name

<input checked="" type="checkbox"/>	Name	Related File
<input type="radio"/>	PLM System Migration	PLM System Migration.pptx
<input type="radio"/>	SM0014	MP0101CCCC Geometry.wcax
<input type="radio"/>	SM2000002	Convective Heat transfer within 3D printer STUDY Report.pdf
<input type="radio"/>	0006	Costing Sheet.txt



- **Access:** It allows you to define roles which can access control. Roles (in the form group identities) can be at Show and Edit level. When control is hidden because you do not have defined role, property associated with this control is not even fetched from server.
- **Read Only:** Based on the option selected for Read Only, control can become read-only even if you can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the control read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is true, the control will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the control will be shown or hidden in the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the control if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If the returned value is false, the control will be hidden.
- **Validations:** Validation rules work on List control similar to Text control. Please refer to Validation section of the Text control for complete details.
- **Conditional Formatting:** Conditional Formatting works on List control similar to Text control. Please refer to the Conditional Formatting section of the Text control for complete details.
- **Reset Dependency:** It allows you reset value of the control to the default if any of the controls defined in "Reset Dependency" have their value changed at runtime.
- **Font:** It allows you to set the required font and size for showing list options and the selected value. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **Margin:** It allows you to set required top, right, bottom, left margins with respect to adjacent controls. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **On Load:** OnLoad event allows you to write custom logic that will be executed after the value in the control is loaded. Please refer to the Template Customization section for more details.
- **On Change:** OnChange event allows you to write custom logic that will be executed after the value in the control is updated. Please refer to the Template Customization section for more details.



Checkbox Control

Checkbox control can be used to display properties of datatype Boolean from the associated ItemType.

- **Name:** It is used to uniquely identify control inside the template.
- **Label:** It is used to show label on the control. It can be localized by defined Languages and Locales inside innovator.
- **Data Type:** It shows the type of the property as defined in the ItemType definition. In case of Checkbox control it will always be Boolean, and it is shown as disabled.
- **Tooltip:** Tooltip will be shown when you hover the mouse on the information icon on the form field. It is derived from the property bound to the control.
- **Help Text:** Help Text flyout will be shown when you click on the information icon on the form field label. It is derived from the property bound to the control.
- **Associated Document:** It allows you to associate a document with the Checkbox control. This associated document will be used while generating word file if this Checkbox is selected. If this Boolean property is used as ContentControl in the word



template, ContentControl will be replaced with the document associated with the CheckBox Control.

- **Access:** It allows you to define roles which can access the control. Roles (in the form group identities) can be at Show and Edit level. When control is hidden because you do not have defined role, property associated with this control is not even fetched from server.
- **Read Only:** Based on the option selected for Read Only, control can become read-only even if you can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the control read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is true, the control will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the control will be shown or hidden in the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the control if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If the returned value is false, the control will be hidden.
- **Validations:** Validation rules work on Checkbox control similar to Text control. Please refer to Validation section of the Text control for complete details.
- **Reset Dependency:** It allows you reset value of the control to the default if any of the controls defined in "Reset Dependency" have their value changed at runtime.
- **Font:** It allows you to set the required font and size for the label of the checkbox.
- **Margin:** It allows you to set required top, right, bottom, left margins with respect to adjacent controls.
- **On Load:** OnLoad event allows you to write custom logic that will be executed after the value in the control is loaded. Please refer to the Template Customization section for more details.
- **On Change:** OnChange event allows you to write custom logic that will be executed after the value in the control is updated. Please refer to the Template Customization section for more details.



Image Control

Image control is used to display properties of datatype Image from the associated ItemType.

Image control will display the picture stored in the vault. It can also load the picture stored on the server folder by setting default value of image property as “../images/ImageName.svg” on ItemType definition.

- **Name:** It is used to uniquely identify control inside the template.
- **Label:** It is used to show label on the control. It can be localized by defined Languages and Locales inside innovator.
- **Data Type:** It shows the type of the property as defined in the ItemType definition. In case of Image control, it will always be Image, and it is shown as disabled.
- **Tooltip:** Tooltip will be shown when you hover the mouse on the information icon on the form field. It is derived from the property bound to the control.

- **Help Text:** Help Text flyout will be shown when you click on the information icon on the form field label. It is derived from the property bound to the control.
- **Show Label:** If show label is selected as Yes, image will be shown with a label at top-left corner.

The image shows a configuration panel for the 'Show Label' property. It has a title bar with the text 'Show Label'. Below the title bar, there are two radio buttons: 'Yes' (which is selected) and 'No'.

- **Placement:** It allows you to place the picture in the available space. By selecting placement as Automatic, picture will be scaled to available space without compromising aspect ratio. With Manual option, by setting width and alignment of the picture, height is computed with the help of aspect ratio.

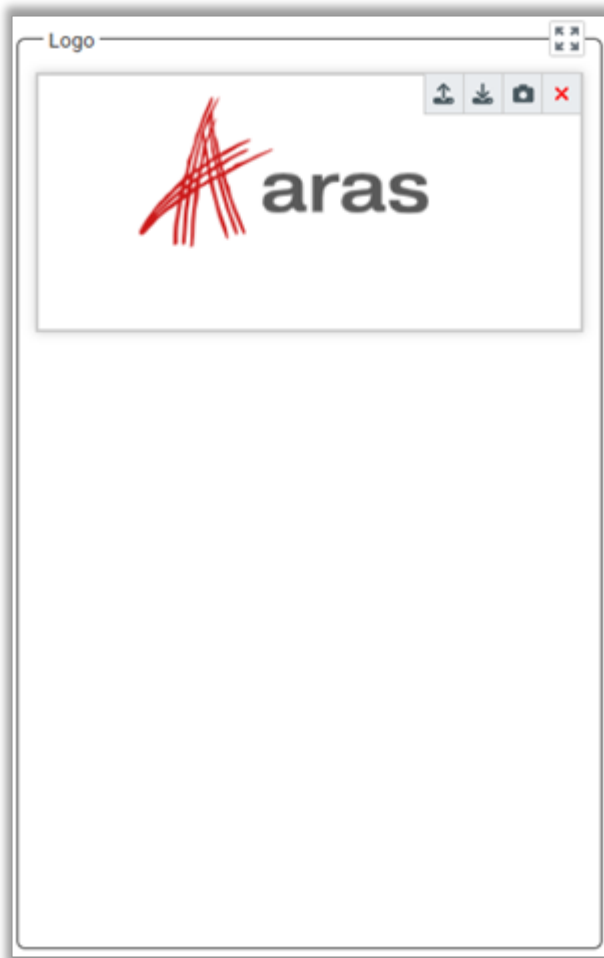
The image shows a configuration panel for the 'Placement' property. It has a title bar with the text 'Placement'. Below the title bar, there are two radio buttons: 'Automatic' and 'Manual' (which is selected). Below the radio buttons, there are two input fields: 'Width (pixels)' with the value '250' and 'Alignment' with a dropdown menu showing 'Left'.

- **Access:** It allows you to define roles which can access the control. Roles (in the form group identities) can be at Show and Edit level. When control is hidden because you do not have defined role, property associated with this control is not even fetched from server.
- **Read Only:** Based on the option selected for Read Only, control can become read-only even if you can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the control read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is true, the control will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the control will be shown or hidden in the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the control if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If the returned value is false, the control will be hidden.
- **Reset Dependency:** It allows you reset value of the control to the default if any of the controls defined in "Reset Dependency" have their value changed at runtime.

- **Validations:** Validation rules work on Image control similar to Text control. Please refer to Validations section of the Text control for complete details.
- **Margin:** It allows you to set required top, right, bottom, left margins with respect to adjacent controls.
- **On Load:** OnLoad event allows you to write custom logic that will be executed after the value in the control is loaded. Please refer to the Template Customization section for more details.
- **On Change:** OnChange event allows you to write custom logic that will be executed after the value in the control is updated. Please refer to the Template Customization section for more details.



Runtime View:



- **Upload:** By selecting this icon, you can upload picture from the local directory. Upload icon is enabled only when the item is in edit mode.
- **Download:** By selecting this icon, you can download the picture to a local directory. Download icon is always enabled if the image control has picture.
- **Capture:** By selecting this icon, you can capture the picture using device's camera. If multiple cameras are present, it will show you options to select specific camera.
- **Delete:** By selecting this icon, you can delete the picture shown in the image control (browser cache). Picture will be deleted from server only when the item is saved. Delete icon is enabled only when the item is in edit mode.



File Control

File control is used to display properties of datatype File from the associated ItemType. It can also show the file stored on the server by setting default value of file property with id of the file item on ItemType definition.

The screenshot shows a 'File Settings' dialog box with the following fields and options:

- Name (maximum 32 characters):** contract
- Label:** Contract
- Data Type:** Item
- Tooltip:** Select the file for Contract
- Help Text:** A contract is a legally binding agreement between two or more parties that creates obliga
- Data Source:** File
- Control Style:** File Video
- Show Label:** Yes No
- File Filter:** Search
- Access:** Show

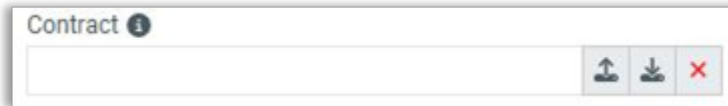
- **Name:** It is used to uniquely identify control inside the template.
- **Label:** It is used to show label on the control. It can be localized by defined Languages and Locales inside innovator.
- **Data Type:** It shows the type of the property as defined in the ItemType definition. In case of File control, it will always be Item, and it is shown as disabled.
- **Tooltip:** Tooltip will be shown when you hover the mouse on the information icon on the form field. It is derived from the property bound to the control.

- **Help Text:** Help Text flyout will be shown when you click on the information icon on the form field label. It is derived from the property bound to the control.
- **Data Source:** It shows the data source of the property as defined in the ItemType definition. In case of File control, it will always be File, and it is shown as disabled.
- **Control Style:** File control can be rendered as File control for selecting file from the folder or Video Player to capture and play the video.
- **Show Label:** If show label is selected as Yes, video player will be shown with a label at top-left corner.
- **File Filter:** It allows you to set an allowed file format from the predefined list of options.
- **Access:** It allows you to define roles which can access the control. Roles (in the form group identities) can be at Show and Edit level. When control is hidden because you do not have defined role, property associated with this control is not even fetched from server.
- **Read Only:** Based on the option selected for Read Only, control can become read-only even if you can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the control read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is true, the control will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the control will be shown or hidden in the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the control if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If the returned value is false, the control will be hidden.
- **Validations:** Validation rules work on File control similar to Text control. Please refer to Validations section of the Text control for complete details.
- **Reset Dependency:** It allows you reset value of the control to the default if any of the controls defined in "Reset Dependency" have their value changed at runtime.
- **Font:** It allows you to set the required font and size for the selected file of File control.
- **Margin:** It allows you to set required top, right, bottom, left margins with respect to adjacent controls.
- **On Load:** OnLoad event allows you to write custom logic that will be executed after the value in the control is loaded. Please refer to the Template Customization section for more details.
- **On Change:** OnChange event allows you to write custom logic that will be executed after the value in the control is updated. Please refer to the Template Customization section for more details.



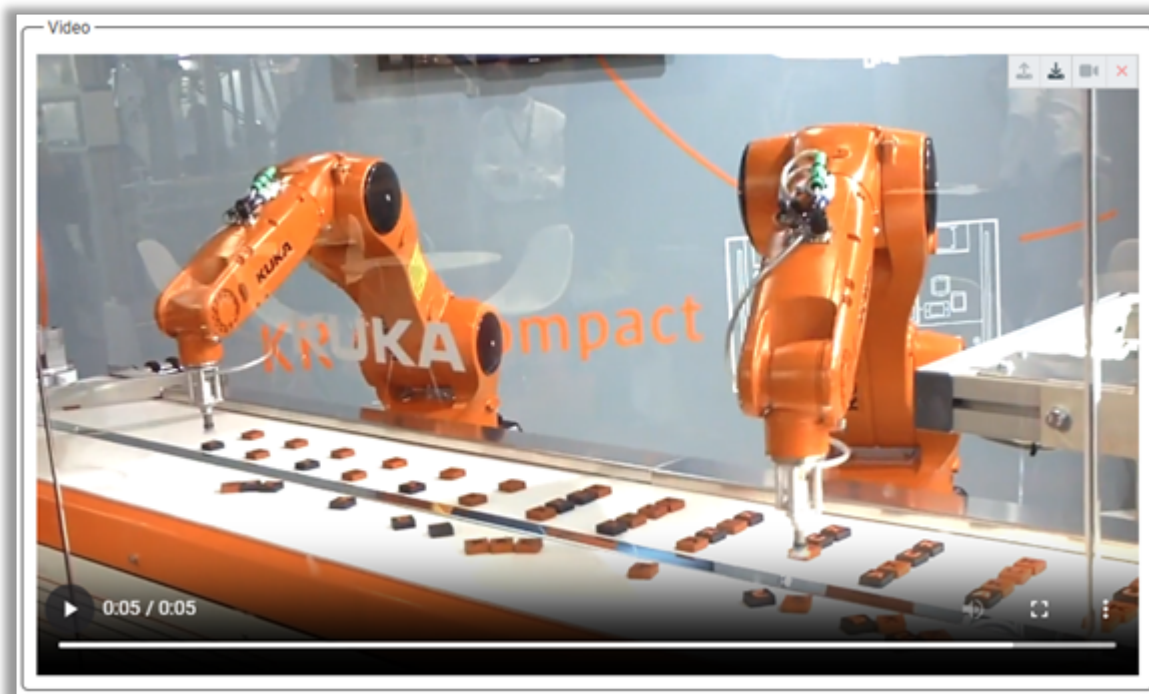
Runtime View:

Control Type: File



- **Upload:** By selecting this icon, you can upload file from the local directory. Upload icon is enabled only when the item is in edit mode.
- **Download:** By selecting this icon, you can download the file to a local directory. Download icon is always enabled if the file control has value.
- **Delete:** By selecting this icon, you can delete the file shown in the file control (browser cache). File will be deleted from server only when the item is saved. Delete icon is enabled only when the item is in edit mode.

Control Type: Video



- **Capture:** By selecting this icon, you can capture the video using device's camera. If multiple cameras are present, it will show you options to select specific camera.

Document Control

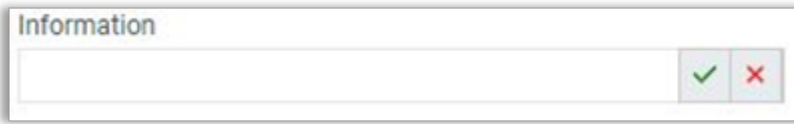
Document control is used to allow you to select the document which is stored in Innovator. Upon clicking on tick icon of the control, Select Document dialog will be shown. After entering search criteria, Select Document dialog shows all the files of type Word, PDF and Image. Selected document is stored on the context item as file property. Document control is useful in case you want to generate Word/PDF document from the context item.

- **Name:** It is used to uniquely identify control inside the template.
- **Label:** It is used to show label on the control. It can be localized by defined Languages and Locales inside innovator.
- **Data Type:** It shows the type of the property as defined in the ItemType definition. In case of Document control, it will always be Item, and it is shown as disabled.

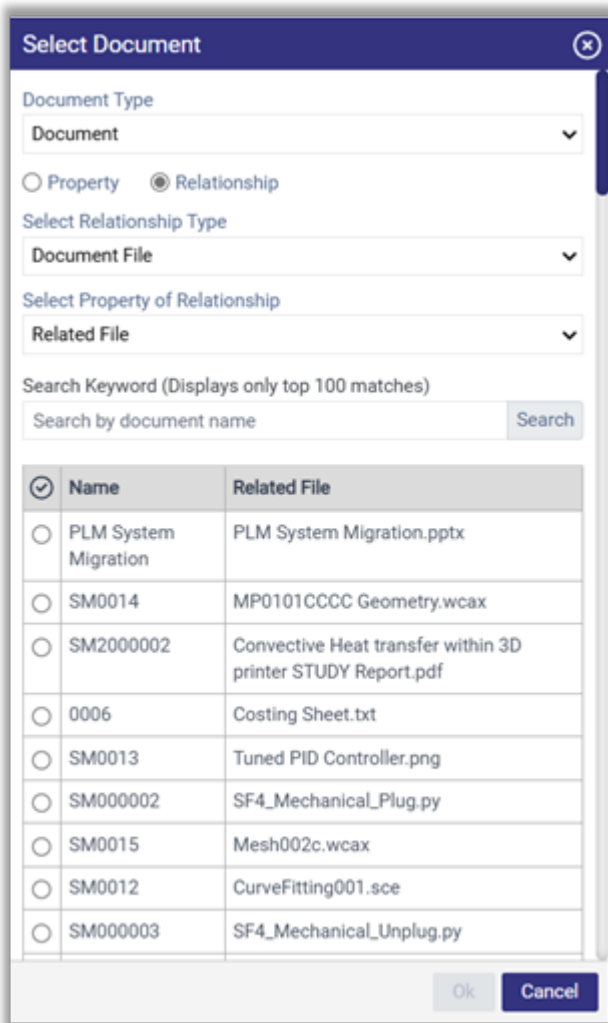
- **Tooltip:** Tooltip will be shown when you hover the mouse on the information icon on the form field. It is derived from the property bound to the control.
- **Help Text:** Help Text flyout will be shown when you click on the information icon on the form field label. It is derived from the property bound to the control.
- **Data Source:** It shows source type of the property as defined in the ItemType definition. In case of Document control, it will always be File, and it is shown as disabled.
- **Document Filter:** It allows you to select the document types that you want to see in the search results. Allowed document types are Image, PDF, Word, Presentation.
- **Access:** It allows you to define roles which can access control. Roles (in the form group identities) can be at Show and Edit level. When control is hidden because you do not have defined role, property associated with this control is not even fetched from server.
- **Read Only:** Based on the option selected for Read Only, control can become read-only even if you can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the control read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is true, the control will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the control will be shown or hidden in the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the control if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If the returned value is false, the control will be hidden.
- **Validations:** Validation rules work on Document control similar to Text control. Please refer to Validations section of the Text control for complete details.
- **Reset Dependency:** It allows you reset value of the control to the default if any of the controls defined in "Reset Dependency" have their value changed at runtime.
- **Font:** It allows you to set the required font and size for the selected document of Document control.
- **Margin:** It allows you to set required top, right, bottom, left margins with respect to adjacent controls.
- **On Load:** OnLoad event allows you to write custom logic that will be executed after the value in the control is loaded. Please refer to the Template Customization section for more details.
- **On Change:** OnChange event allows you to write custom logic that will be executed after the value in the control is updated. Please refer to the Template Customization section for more details.



Runtime View:



Select Document Flyout:



RichText Control

RichText control is used to display properties of datatype FormattedText from the associated ItemType. In edit mode, RichText control shows toolbar, using it you can format the text with bold, italic, font, color etc. It also allows you to insert items like images, hyperlinks and videos, upon saving all these items are embedded inside FormattedText on the property.

- **Name:** It is used to uniquely identify control inside the template.
- **Label:** It is used to show label on the control. It can be localized by defined Languages and Locales inside innovator.
- **Data Type:** It shows the type of the property as defined in the ItemType definition. In case of RichText control, it will always be FormattedText, and it is shown as disabled.
- **Tooltip:** Tooltip will be shown when you hover the mouse on the information icon on the form field. It is derived from the property bound to the control.

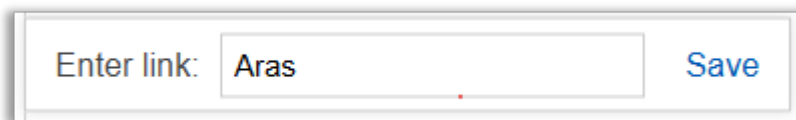
- **Help Text:** Help Text flyout will be shown when you click on the information icon on the form field label. It is derived from the property bound to the control.
- **Show label:** If show label is selected as Yes, RichText control will be shown with a label at top-left corner.
- **Inline Toolbar:** If an inline toolbar is selected as Yes, toolbar will be shown within RichText control. Otherwise it will be shown in the popup after selecting the tick icon.
- **Max Height:** It is an optional setting, if you don't set any value for this, it will be shown with the height that fits the content to avoid scrollbar. By setting max height explicitly, RichText control will auto grow up to this height based on the content, if the content goes out of bounds, then it shows scrollbar.
- **Access:** It allows you to define roles which can access the control. Roles (in the form group identities) can be at Show and Edit level. When control is hidden because you do not have defined role, property associated with this control is not even fetched from server.
- **Read Only:** Based on the option selected for Read Only, control can become read-only even if you can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the control read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is true, the control will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the control will be shown or hidden in the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the control if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is false, the control will be hidden.
- **Reset Dependency:** It allows you reset value of the control to the default if any of the controls defined in "Reset Dependency" have their value changed at runtime.
- **Margin:** It allows you to set required top, right, bottom, left margins with respect to adjacent controls.
- **On Load:** OnLoad event allows you to write custom logic that will be executed after the value in the control is loaded. Please refer to the Template Customization section for more details.
- **On Change:** OnChange event allows you to write custom logic that will be executed after the value in the control is updated. Please refer to the Template Customization section for more details.



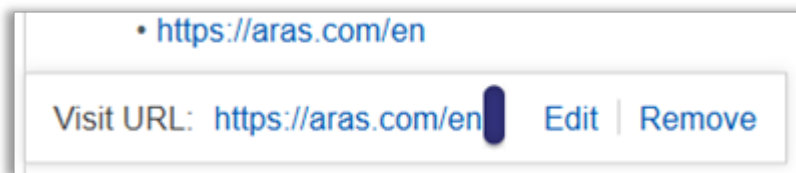
RichText Control Toolbar:



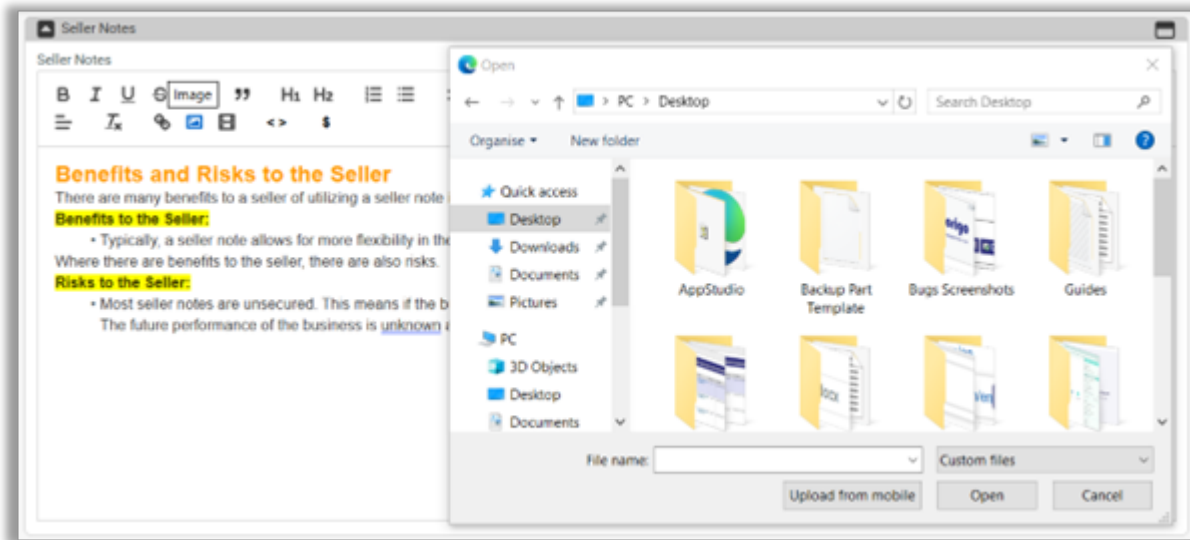
Hyperlink can be inserted in RichText control by highlighting the text and clicking on the null icon from toolbar. Then it shows pop-up as shown below to set link for the selected text.



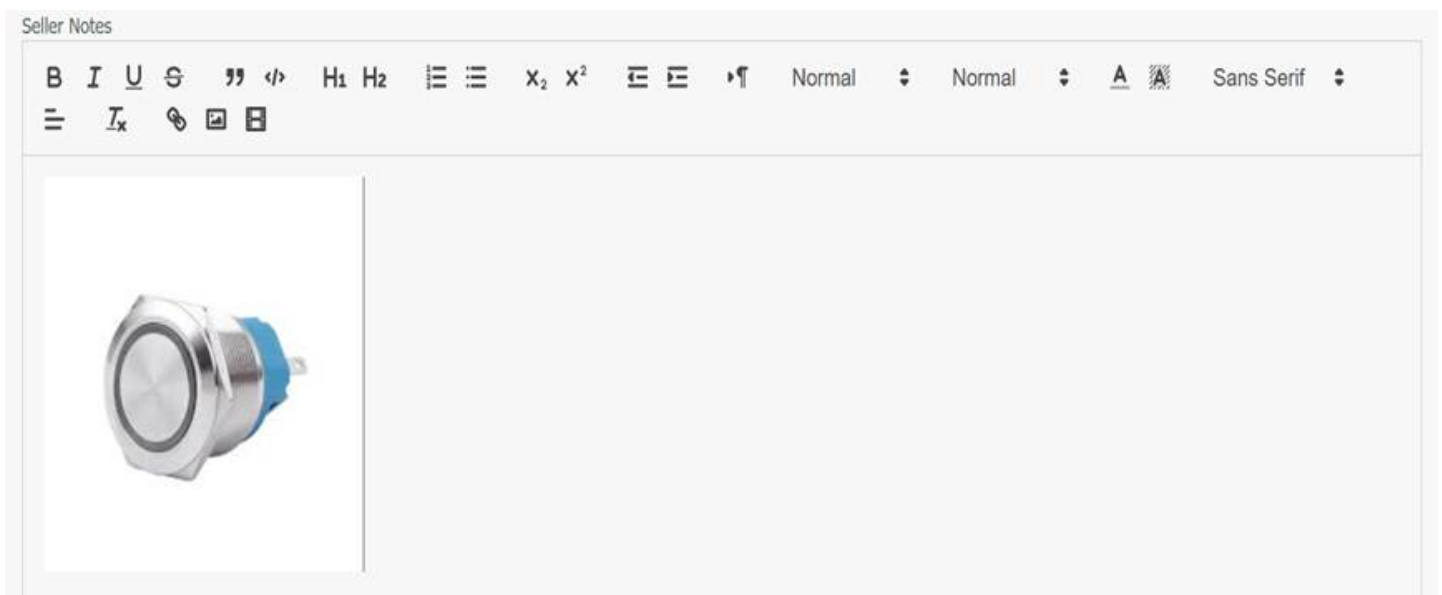
By clicking on the linked text, it shows the options to edit the current link or to remove link all together.



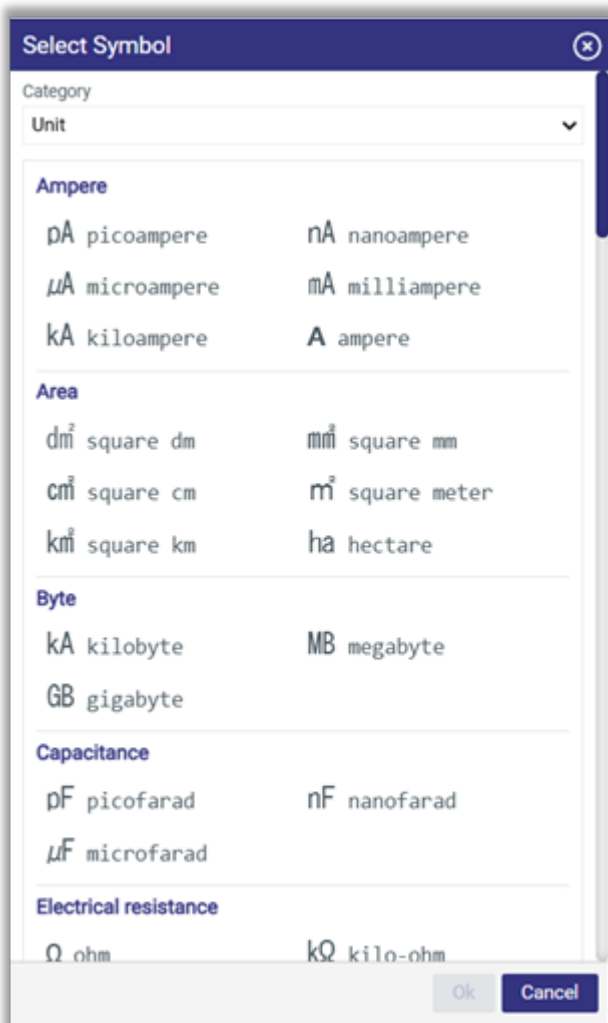
Picture can be inserted in RichText control at the current position of the cursor by selecting null icon from the toolbar. Then file browser dialog comes up in order to select the picture from the local directory.



The selected picture is displayed in the RichText control at the current cursor position.



Raw HTML command from the toolbar will allow you to set any custom HTML as content for the RichText control. Selecting Insert Symbol command will show Select Symbol flyout, for inserting a symbol at the cursor position.



Item Control

Item control is used to display properties of datatype Item from the associated ItemType. Using this control, you can select existing item of the property's Data Source ItemType through Select Dialog.

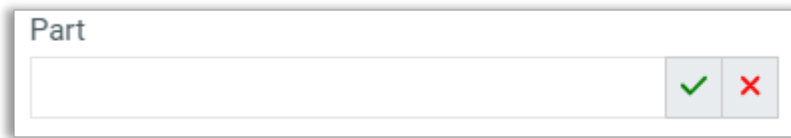
- **Name:** It is used to uniquely identify control inside the template.
- **Label:** It is used to show label on the control. It can be localized by defined Languages and Locales inside innovator.
- **Data Type:** It shows the type of the property as defined in the ItemType definition. In case of Item control, it will always be Item, and it is shown as disabled.
- **Tooltip:** Tooltip will be shown when you hover the mouse on the information icon on the form field. It is derived from the property bound to the control.
- **Help Text:** Help Text flyout will be shown when you click on the information icon on the form field label. It is derived from the property bound to the control.
- **Data Source:** It shows the type of the Item that can be selected as defined in the ItemType definition. It is shown as disabled.



- **Select Filter:** It allows you to select filter criteria based on the properties of the Related ItemType. Defined filter is always applied with 'And' condition on the Selected Items dialog along with the Search Keyword. If FilterType is Static, for the expression you need to provide constant value. If FilterType is Dynamic, for the expression you need to provide other property name as value, with this at runtime it will pull the value from the selected property and send it in the filter request to the server. As an admin, you can also enforce the filter, in that case end-user can see the applied filter on the Select Dialog but can't change the filter criteria. If enforce filter is not set, end-user can change the filter criteria at the runtime.
- **Select Columns:** It allows you to define columns that you want to show in the Select Dialog. Column widths can be given in percentage or pixels modes. In percentage mode, you never see horizontal scrollbar, column widths will be adjusted to the available dialog width. But in pixels mode, based on the given column widths you may see horizontal scrollbar in the dialog. Flyout width in percentage, allows you to set width of the flyout shown for selecting the item. By selecting Show Add command 'Yes', Select Item flyout with show toolbar with Add Item command on the table. Using this command, new item can be created on the fly as part of the select item operation.
- **Access:** It allows you to define roles which can access control. Roles (in the form group identities) can be at Show and Edit level. When control is hidden because you do not have defined role, property associated with this control is not even fetched from server.
- **Read Only:** Based on the option selected for Read Only, control can become read-only even if you can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the control read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is true, the control will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the control will be shown or hidden in the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the control if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is false, the control will be hidden.
- **Validations:** Validation rules work on Item control similar to Text control. Please refer to Validations section of the Text control for complete details.
- **Reset Dependency:** It allows you reset value of the control to the default if any of the controls defined in "Reset Dependency" have their value changed at runtime.
- **Font:** It allows you to set the required font, size and color for the value of Item control.
- **Margin:** It allows you to set required top, right, bottom, left margins with respect to adjacent controls.
- **On Load:** OnLoad event allows you to write custom logic that will be executed after the value in the control is loaded. Please refer to the Template Customization section for more details.
- **On Change:** OnChange event allows you to write custom logic that will be executed after the value in the control is updated. Please refer to the Template Customization section for more details.
- **On Select Filter:** On Select Filter event will be launched after hitting the search button in item selection flyout. Admin can configure a custom script to modify entered filter expression or assign entirely new filter expression string to the eventData. eventData object is available in the scope of the event handler code. Please refer to the Template Customization section for more details.



Runtime View:



- **Select:** By selecting tick icon, you will get 'Select Item' flyout from which you can search based on keyword/filter and select a specific item. By clicking on search after entering keyword or filter criteria, you will get a list of matched items, and you can select one of the items. Item control supports typeahead; by entering first three characters, it shows all matched items in the dropdown.

Select Part
✕

Search Keyword (Displays only top 100 matches)

Property Filter

Property	Operator	Value
State	EqualTo	Preliminary ✓ ✕

Expression AND OR () ✕

State EqualTo Preliminary

+

	Name	Part Number	State
<input checked="" type="checkbox"/>			
<input type="checkbox"/>	Polyurethane Cover	100013	Preliminary
<input type="checkbox"/>	Velcro Belt	10012	Preliminary
<input type="checkbox"/>	Acetate Filter	Acetate Filter	Preliminary
<input type="checkbox"/>	Aluminum Foil	Aluminum Foil	Preliminary
<input type="checkbox"/>	Axle	AXL	Preliminary
<input type="checkbox"/>	Axle coming from GM V	AXL-GMV	Preliminary
<input type="checkbox"/>	Axle coming from SVC and light weight	AXL-SVC	Preliminary
<input type="checkbox"/>	Axle coming from TVS - Heavy Duty	AXL-TVS	Preliminary
<input type="checkbox"/>	Belt	Belt	Preliminary
<input type="checkbox"/>	Bicycle	bicycle	Preliminary

- **Remove:** By selecting remove icon, you can remove existing item. Item reference will be removed only when you save the context item.



Items Control

Items control is used to select and display list of items with their keyed-names from the associated ItemType. Using this control, you can select existing items of the property's Data Source ItemType through Select Dialog. Selected items will be saved as relationships upon saving the context item.

Items Settings

Name (maximum 32 characters)
SupplierToProduct

Label
Supplier Products

Data Source
Product

Select Filter

Property	Operator	Filter Type	Value
		Static	

Expression
AND OR ()

Enforce Filter

Select Columns

Column Width Percentage Pixels

Name Width (%)
0

Name	Width	Actions
Product Number	30	✕ ⛶
Name	30	✕ ⛶

Overflow Width(%) 30

Ok Cancel

- **Name:** It is used to uniquely identify control inside the template. It is always the selected RelationshipType name.
- **Label:** It is used to show label on the control. It can be localized by defined Languages and Locales inside innovator. It always shows selected RelationshipType label.
- **Data Source:** It shows the type of the Item that can be selected as defined in the ItemType definition. It is shown as disabled.



- **Select Filter:** It allows you to select filter criteria based on the properties of the Related ItemType. Defined filter is always applied with 'And' condition on the Selected Items dialog along with the Search Keyword. If FilterType is Static, for the expression you need to provide constant value. If FilterType is Dynamic, for the expression you need to provide other property name as value, with this at runtime it will pull the value from the selected property and send it in the filter request to the server. As an admin, you can also enforce the filter, in that case end-user can see the applied filter on the Select Dialog but can't change the filter criteria. If enforce filter is not set, end-user can change the filter criteria at the runtime.
- **Select Columns:** It allows you to define columns that you want to show in the Select Dialog. Column widths can be given in percentage or pixels modes. In percentage mode, you never see horizontal scrollbar, column widths will be adjusted to the available dialog width. But in pixels mode, based on the given column widths you may see horizontal scrollbar in the dialog. Flyout width in percentage, allows you to set width of the flyout shown for selecting the items. By selecting Show Add command 'Yes', Select Items flyout with show toolbar with Add Item command on the table. Using this command, new item can be created on the fly as part of the select items operation.
- **Access:** It allows you to define roles which can access the control. Roles (in the form group identities) can be at Show and Edit level. When control is hidden because you do not have defined role, property associated with this control is not even fetched from server.
- **Read Only:** Based on the option selected for Read Only, control can become read-only even if you can edit the properties used inside the page with lock. You can select one of the options like 'Yes', 'No', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally make the control read-only if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is true, the control will be read-only.
- **Display Condition:** Based on the option selected for Display Condition, the control will be shown or hidden in the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the control if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is false, the control will be hidden.
- **Validations:** Validation rules work on Items control similar to Text control. Please refer to Validations section of the Text control for complete details.
- **Reset Dependency:** It allows you reset value of the control to the default if any of the controls defined in "Reset Dependency" have their value changed at runtime.
- **Font:** It allows you to set the required font, size and color for the value of Items control.
- **Margin:** It allows you to set the required top, right, bottom, left margins with respect to adjacent controls.
- **On Load:** OnLoad event allows you to write custom logic that will be executed after the value in the control is loaded. Please refer to the Template Customization section for more details.
- **On Change:** OnChange event allows you to write custom logic that will be executed after the value in the control is updated. Please refer to the Template Customization section for more details.
- **On Select Filter:** On Select Filter event will be launched after hitting the search button in item selection flyout. Admin can configure a custom script to modify entered filter expression or assign entirely new filter expression string to the eventData. eventData object is available in the scope of the event handler code. Please refer to the Template Customization section for more details.



Runtime View:

Supplier Products

✓
✗

- **Select:** By selecting tick icon, you will get 'Select Items' flyout from which you can search based on keyword/filter and select list of items. By clicking on search after entering keyword or filter criteria, you will get a list of matched items, and you can select multiple items.

✕

Select Product

Search Keyword (Displays only top 100 matches)

Search

Property Filter

Property	Operator	Value	
state	Contains	Released	✓ ✗

Expression AND OR () ✕

state Contains Released

<input checked="" type="checkbox"/>	Product Number	Name
<input type="checkbox"/>	P100	Toolbox
<input type="checkbox"/>	P100	engine2stroke screw
<input type="checkbox"/>	P101	ToolboxStand
<input type="checkbox"/>	P78	windshaft
<input type="checkbox"/>	P98	cableline 98

Ok
Cancel



xClassification Control

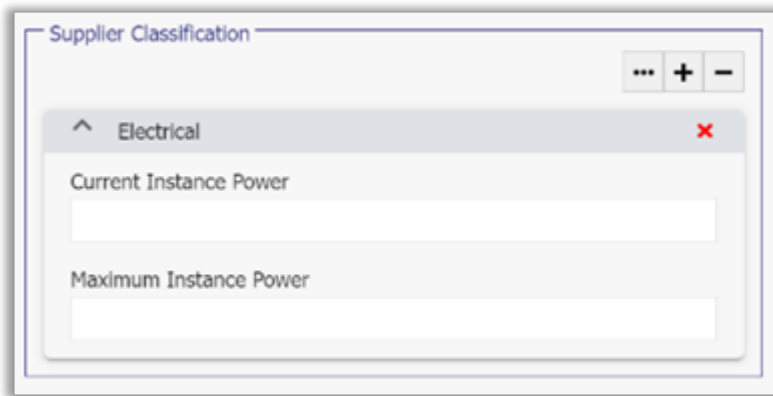
xClassification control is used to associate xClassification tree nodes and xProperties with the item. It works exactly same way as you work with regular Innovator forms.

- **Name:** It is used to uniquely identify control inside the template.
- **Label:** It is used to show title of the group. It can be localized by defined Languages and Locales inside innovator.
- **Access:** It allows you to define roles which can access control. Roles (in the form group identities) can be at Show and Edit level. When group is hidden because you do not have defined role, properties used inside this group are not even fetched from server.

- **Display Condition:** Based on the option selected for Display Condition, the control will be shown or hidden in the wizard at runtime. You can select one of the options like 'Always Show', 'Always Hide', 'Expression', or 'Script'. Expression allows you to define a Boolean expression using properties from the current and previous pages (in case of multi-page application) to conditionally display the control if expression is evaluated to true. Expression will be evaluated based on the cached property values on the client. Script allows you to write custom JavaScript code that should return Boolean value. If returned value is false, the control will be hidden.
- **Border:** It allows you to show or hide the border and legend around selected xClasses. In case if you choose to show the border, you can select the border style using Color, Background, Type and Thickness settings. If any of the settings are changed, it shows Reset icon on the border, using which you can reset the settings.
- **Font:** It allows you to set the required font and size on the individual fields of the classification properties.
- **Margin:** It allows you to set required top, right, bottom, left margins with respect to adjacent controls.
- **On Load:** OnLoad event allows you to write custom logic that will be executed after the value in the control is loaded. Please refer to the Template Customization section for more details.
- **On Change:** OnChange event allows you to write custom logic that will be executed after the value in the control is updated. Please refer to the Template Customization section for more details.



Runtime View:



When you click on “...” from xClassification control runtime view, you will see xClassification tree selection dialog. Select specific subclass and click on Ok, you will see selected subclass along with its xProperties in the control.

- **Expand/Collapse:** Displays or hides all the subclasses in a multiple level hierarchy.
- **Search Filter:** Displays the classes and subclasses containing the characters typed into the filter box. For example, to only show subclasses that start with the keyword "EI", enter EI into the search field, and only those subclasses starting with EI will be displayed.

