

# IFS Customer Engagement 25R1 GA

## CE Studio Designer Guide



**Document Revision: 25R1-GA-1**  
**Publication Date: May 14, 2025**



# Contents

<b>1 Updates for the current release.....</b>	<b>9</b>
<b>2 About CE Studio apps.....</b>	<b>11</b>
What is a CE Studio app?.....	12
About providers.....	13
About components and elements.....	14
About actions and events.....	15
Creating a standard app or template.....	17
Importing CE Studio Apps.....	19
<b>3 Basic end-to-end configuration.....</b>	<b>21</b>
Introduction.....	21
Getting started.....	24
Adding providers to an app.....	25
Add the components.....	26
Adding a data grid.....	26
Formatting a data grid.....	28
Adding a form and UI elements.....	30
Adding actions and configuring events.....	32
Adding an action and loading a data grid.....	33
On selecting a row in the data grid.....	37
Finding out how the action sets are used.....	41
Sorting and paging the data grid.....	42
Expanding and collapsing UI elements.....	43
Preview in browser.....	44
Make the CE Studio app live.....	45
Test the public portal app in the public portal realm.....	45
Test the media app in Agent Desktop.....	46
<b>4 Working with templates.....</b>	<b>48</b>
Adding a template as a component to an app.....	48
Deciding when to create templates.....	49
Importing templates into a standard app.....	51
Configuring the app to use data from the template.....	53

Configuring the template to use data from the app.....	55
Converting templates and apps.....	57
Adding configuration options to templates.....	59
Public topics and templates.....	63
Using templates with placeholders.....	64
Example of using templates with placeholders.....	64
Template placeholder example for FSM.....	67
How to configure a template with placeholders.....	67
Standard CE templates for media apps.....	68
<b>5 Data providers and API services.....</b>	<b>71</b>
Provider types.....	71
Additional provider types.....	73
Creating a new provider.....	75
Creating a new provider version.....	77
Using the Metadata Viewer.....	78
IFS Applications/Cloud provider.....	81
Locating projections in IFS Cloud.....	85
Understanding IFS Cloud provider method names.....	87
Accessing IFS Cloud unbound methods.....	89
Working with child provider types in IFS Applications providers.....	89
Field Service Management provider.....	91
Report provider.....	95
IFS CE Agent Desktop provider.....	98
Realtime Statistics provider.....	99
Workflow provider.....	103
About work definition elements.....	108
Tenant oData 4 service.....	109
OpenAPI providers.....	110
System provider.....	111
Troubleshooting providers.....	112
<b>6 Components and elements.....</b>	<b>114</b>
Adding components and selecting properties.....	115
Configuring component properties.....	118
Field properties.....	120
Template properties.....	122

Adding a custom field.....	122
Adaptive cards.....	123
Configuring placeholders in adaptive cards.....	124
Example - configuring placeholders in adaptive cards.....	126
Configuring action inputs in adaptive cards.....	128
Adding an adaptive card to a CE Studio apps.....	129
Configuring actions and events for adaptive cards.....	130
Copying adaptive cards between tenants.....	132
Troubleshooting adaptive cards.....	132
Calendars and planners.....	134
Simple Calendar.....	134
Planners.....	138
Captcha.....	142
Charts.....	142
Combo boxes.....	149
Data elements.....	152
File attachments.....	156
File attachments on IFS Cloud.....	156
File attachments for media apps and FSM.....	158
List Template and Attachment Download components.....	159
Email attachments.....	159
Image elements.....	161
Label elements.....	164
List View Element.....	164
Overlay components for modals.....	167
Radio buttons.....	167
Tab groups.....	170
Using timers.....	171
Configuring a data grid for bulk updates and deletes.....	173
Display formats for dates and numbers.....	177
Troubleshooting components and elements.....	178
<b>7 Actions, rules and events.....</b>	<b>181</b>
Actions and action sets.....	182
Actions and filters.....	190
Actions and dynamic filters.....	194
Rules and rule sets.....	195

Event configuration for apps and templates.....	198
Event configuration for media apps and templates.....	199
Event configuration for components.....	201
Data transformations.....	203
Configuring data transformations.....	204
Transformation output examples.....	211
Using transformed data.....	214
Global actions.....	215
URI and Link parameters.....	217
Unit tests.....	220
Troubleshooting actions.....	223
<b>8 Display tasks.....</b>	<b>226</b>
Using display tasks.....	226
Active display tasks.....	228
Combined display tasks.....	229
Expand, Collapse, Show, Hide display tasks.....	229
Navigate display tasks.....	230
Prompt display tasks.....	231
Clicks display task.....	231
<b>9 Portals and public portals.....</b>	<b>233</b>
Portals for registered users.....	234
Configuring portal access groups.....	235
Creating portal apps.....	236
Adding links to portal apps.....	238
Creating portal apps to use as landing pages.....	240
Configuring the portal settings.....	240
Testing the portal.....	241
Public portals for anonymous users.....	242
Configuring public portal apps.....	242
Adding links to public portal apps using a Portal List Template.....	243
Public portal URLs.....	245
<b>10 Authentication for public portals.....</b>	<b>247</b>
About the example challenge apps.....	249

Configuring the example challenge apps.....	249
Protecting a public portal app.....	250
Authentication settings for protected apps.....	252
Registering new users through OTP and a challenge app.....	255
Personalizing data based on secure session info.....	256
Signing out from a public portal.....	257
Response templates.....	258
<b>11 Surveys.....</b>	<b>260</b>
Configuring a survey app.....	261
Configuring the survey element.....	261
About the candidate request ID and Survey ID.....	263
Actions for surveys.....	263
Events for surveys in public portal apps.....	265
Previewing surveys in public portal apps.....	265
Testing surveys in public portal apps.....	266
Sending survey invites from media apps in Agent Desktop.....	266
Transferring voice callers to an IVR survey.....	268
Configuring a survey as part of a public portal app.....	270
Configuring the survey element in public portal apps.....	270
Configuring the survey candidate in public portal apps.....	270
Actions and events for surveys in public portal apps.....	272
Configuring a survey as part of a portal app.....	272
Configuring the survey element in portal apps.....	272
Configuring the survey candidate in portal apps.....	273
Actions for surveys in portal apps.....	275
Events for surveys in portal apps.....	276
Testing surveys in portal apps.....	277
<b>12 Media apps.....</b>	<b>278</b>
Email.....	278
Configuring CE Studio apps for email.....	278
About the IFS CE Email Template.....	281
Using the email template.....	283
Overview of the email provider types and methods.....	284
CE Email type and methods.....	285
OutboundEmail type and methods.....	288

Email methods from the activation.....	289
Chat.....	290
About the IFS CE Chat Template.....	291
Using the chat template.....	292
Configuring the Chat component.....	293
Using application data with chat.....	294
Social Media.....	295
Overview of CE Studio apps for social media.....	295
Social Media provider types.....	296
Configuring the Social Media Element.....	297
Rules for the Social Media Element.....	299
Actions for the Social Media Element.....	300
Methods for social media conversations.....	302
Message states.....	303
Opening and closing social media activations.....	303
Sending messages using WhatsApp Content Templates.....	304
Integrating media apps with Agent Desktop.....	306
Linking CE Studio apps to Agent Desktop media types.....	306
Accessing application data.....	309
Accessing call data.....	310
Using the Toolbar Query Params - Configuration example.....	313
IFS CE Agent Desktop provider methods.....	314
General methods in the IFS CE Agent Desktop provider.....	314
Voice call methods in the IFS CE Agent Desktop provider.....	316
Email and chat methods in the IFS CE Agent Desktop provider.....	318
Agent Desktop events.....	318
Call states.....	319
Agent states.....	320
App state change event.....	322
Error and return codes.....	323

### **13 Last-Mile Customer Portal..... 324**

Last-Mile Customer Portal apps.....	325
Configure Last-Mile Customer Portal app for surveys.....	326
Configuring a custom Last-Mile Customer Portal app.....	327
Stepper elements and display tasks.....	328
Actions for Last-Mile Customer Portal apps.....	329

Event configuration for Last-Mile Customer Portal apps.....	330
<b>14 Styles and themes.....</b>	<b>331</b>
Themes.....	332
Styling a CE Studio app.....	332
Styling components.....	334
Templates and inherited styles.....	337
<b>15 Localizing strings in the user interface.....</b>	<b>339</b>
Translation providers.....	340
Translating strings in the user interface.....	340

# 1

## Updates for the current release

Summary of documentation updates for the current release.

Version	Changes
25R1 GA - (6.9.0)	<p>Event Configuration :</p> <ul style="list-style-type: none"><li>• End-users can configure an onChange event for the date picker &amp; Richtextbox element directly within the Designer interface.</li></ul>
25R1 - (6.8.5)	<ul style="list-style-type: none"><li>• Data Transformation : DataGrid does not support sorting when an action contains filters with dynamic values in Data Transformation. <a href="#">Configure_Data_Transformations</a></li><li>• Currently, identifiers of already used buttons cannot be updated</li><li>• You can apply theming to the multi-select feature in the DataGrid.</li><li>• IVR surveys - Setting up a scheduled invite</li></ul>
24R2 - (6.8.4)	<p>Template restrictions. See : <a href="#">Importing templates into a standard app</a></p> <p>List view elements : Enable to add and apply multiple format conditions to the List View.</p>
24R2 - (6.8.3)	<ul style="list-style-type: none"><li>• When the "Dynamic UI" checkbox is selected, the "Ignore Case" option is disabled. The "Ignore Case" functionality is now only available when the dynamic filter is turned off.</li><li>• When a template is converted to a standard app, the background options become available again in the App Settings.)</li></ul>
24R2 - (6.8.3)	<p>Outbound Call Status Reasons. See <a href="#">Outbound call status types</a></p> <p>Open API Providers. See <a href="#">OpenAPI providers</a></p> <p>The OnChange event is no longer activated when the user tabs into the following input types: text box, text area, decimal, number, and currency.</p>

Version	Changes
24R2 SUI (6.8.1)	Additional information on configuring the Social Media Element to show conversations. See <a href="#">Methods for social media conversations</a> .
24R2 GA (6.8.0)	How to <a href="#">configure the Social Media Element for previewing conversations</a> . How to configure an action to send messages using WhatsApp Content Templates. <a href="#">Sending messages using WhatsApp Content Templates</a> .
24R2 EA (6.8.0-rc1)	Updates to templates to reflect new button names, specifically <b>Insert into App</b> instead of Import.
24R1 GA SU3 (6.7.3)	<ul style="list-style-type: none"> <li>• You can now add multiple <a href="#">SMS From Numbers</a> when configuring authentication in public portal apps.</li> <li>• For IFS Cloud providers, information on: <ul style="list-style-type: none"> <li>• <a href="#">Locating projections in IFS Cloud</a></li> <li>• <a href="#">Accessing IFS Cloud unbound methods</a></li> </ul> </li> <li>• Information on <a href="#">Copying adaptive cards between tenants</a></li> <li>• Information on <a href="#">setting the time zone</a> for a provider endpoint that does not use UTC.</li> </ul>
24R1 GA (6.7.0)	<ul style="list-style-type: none"> <li>• <a href="#">Registering new users through OTP and a challenge app</a>.</li> <li>• Configuring the <a href="#">List View Element</a>.</li> <li>• Information on how to configure the Attachment component and List Template element for file downloads. See <a href="#">File attachments</a>.</li> </ul>
24R1 EA (6.7.0-rc1)	<ul style="list-style-type: none"> <li>• <a href="#">Using templates in version 6.7</a>.</li> <li>• You now download the sample CE Studio apps, including the Last-Mile Customer Portal apps, from the Central Template Repository which is part of CE Studio Designer. See <a href="#">Importing apps from the app store</a>.</li> <li>• Updated information on <a href="#">configuring authentication for public portal apps</a> in 6.7.</li> <li>• <a href="#">Configuring CAPTCHA</a>.</li> <li>• Information on how to <a href="#">configure the survey</a> in a Last-Mile Customer Portal app.</li> <li>• Updated information on using <a href="#">calendars and planner components</a>.</li> </ul>

# 2

## About CE Studio apps

Brief overview of CE Studio apps and data providers.

In CE Studio Designer you can configure different types of CE Studioapp.

These are the different types of app available:

- Portal apps

You can configure self-service apps where authentication is provided by CE Studio. This type of app is intended for authorized users to view and update their information. Portal users access these apps through a portal landing app that you also configure in CE Studio.

Tenant type: All types

- Public portal apps

You can configure self-service apps that don't require any authentication, for example, to provide publicly available information or contact forms. However, you can use an alternative form of authentication that you implement depending on the requirements of a third-party system.

Tenant type: All types

- Media apps (IFS Customer Engagement only)

You can configure an app for each communication channel that you handle in Agent Desktop, such as email, chat, SMS, social media, inbound and outbound voice calls. You can also configure apps for the main page that's shown when agents are, for example, idle or working offline. If required, each contact center unit can use for different apps. Templates are provided as a starting point.

Tenant type: Fully Featured (with contact center)

- Wallboard apps (IFS Customer Engagement only)

You can configure apps to show statistics and other information relevant to contact centers.

Tenant type: Fully Featured (with contact center)

You can create templates for any components that you want to reuse in a CE Studio app. Regardless of the requirements of the app, you always follow the same configuration process.

**Note** There are some sample templates available for download from the Central Template Repository that demonstrate how you can use the features of CE Studio.

## What is a CE Studio app?

A CE Studio app is a container with one or more components, such as forms, data grids, stack panels, data elements and, optionally, templates such as the IFS CE Email Template.

The outermost component (the app):

- Has a version. One version is the live version and this is the version that is used in the portal, public portal or Agent Desktop. You cannot edit or delete the live version.
- Depending on the required data sources, the app is configured with one or more *providers*. A provider determines the available data types, properties and methods. There are several system providers, such as Reports.
- Has a theme and other properties that can be inherited by its components.
- Has an ID and a locale so that it can be localized.

A template is a special type of reusable component, for example a chat component that you can embed in multiple CE Studio apps, or a header or footer that you want to add to every app.

### Properties that are inherited from the app

Option	Inheritance	Change in
Theme		Components, buttons
Header tint		Components
Headerless		Components
Display density	Inherited from app	
Background color/image	Inherited from app	Components, data elements, tab, labels
Background transparent		Components
Rounding	Inherited from app	Data elements, tabs
Drop shadow	Inherited from app	

Option	Inheritance	Change in
Typeface	Inherited from app	Data elements, stack panels, labels
Label color		Components
Label position		Form fields
Icon		Component headers, buttons, form fields, tab
Icon position		Form fields, tabs
Conditional formatting		Data grid, data element, calendar
Additional styling options		Button, chart, stack panel, tab

## About providers

CE has a provider for each supported data source. A provider stores the metadata types and methods of the data source. You can add multiple providers to a CE Studio app. Regardless of the differences between data sources, you always use providers in the same way when creating a CE Studio app.

For more detailed information, see [Data providers and API services](#).

### Choosing a provider version

In order to control the upgrade process, a provider has a version number and one of the provider versions can be the live or active version. When creating templates and standard apps in CE Studio Designer, you add the provider(s) you require and select the provider version you want to use. The version you select depends on how you intend to manage the upgrade process:

- **Manual upgrades:** To control when a template or standard app is upgraded, choose a specific, numbered version of the provider. You can then choose when to upgrade the template or CE Studio app.
- **Automatic upgrades:** To automatically upgrade a template or standard app, select the active or live version of the provider. The upgrade is applied whenever you create another version of the provider and make that version live.

### Updating a provider

When there is a new release of a supported data source, you can update the provider used in CE Studio Designer. You only need to do this if the data source extends the API to include new features in the release and you want to use those features. There is no need to create a

new provider if the integration does not require those features and the release is backwards compatible.

### Exploring the provider metadata

There is a metadata viewer for exploring the provider metadata. For details, see [Using the Metadata Viewer](#).

### Concurrency

Where the CE Studio app is planned to be multi-user then the provider can optionally perform checks for concurrent usage before sending the request to the provider endpoint.

## About components and elements

---

Components are the building blocks of CE Studio apps and templates:

- Each component contains one top-level element: such as a form, data grid, stack panel, calendar, data element. It can also contain [templates](#) if the app type is standard app.
- An element is configured with properties selected from a single data provider.
- Elements are populated and updated by running action sets (or rule sets for conditional processing).
- Elements are shown, expanded, reset and so on by display tasks. Display task sets can be run by action sets, rule sets or attached directly to an event.

For detailed information on the available types, see [Components and elements](#).

### Forms

Use forms for entering and displaying data in a flexible format. You can add any of the following to forms:

- Fields — from one or more providers
- Other elements, such as data grids (use an Overlay element to perform a lookup from a row in a data grid)
- UI elements, such as RichText, TextArea, Currency, ComboBox, DatePicker
- Button groups
- Labels
- In IFS Customer Engagement only, call data parameters, such as the caller's phone number

## Data grids

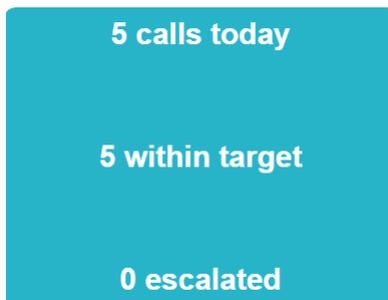
Use data grids for displaying data in a tabular format. A data grid can only contain properties from one provider and provider type only. The data grid is populated by running an action to get the data from the external data source. In this action, you can also:

- Use conditions to filter the data
- Transform the data
- Set a default sort order
- Set the page size

You can format the columns to make the data easier to understand, for example, use icons to represent Boolean values or color coding.

## Data elements

Data elements are stylable panes that can display data from a provider and/or fixed values.



*Example of a data element showing call statistics for a contact center*

## Stack panels

You can design headers and footers using stack panels. This lets you add an image and banner text. You may want to create these as templates so that you can add them to a series of apps. You can add other elements to a stack panel, such as button groups.

## About actions and events

---

An action invokes a method on the provider in order to retrieve or modify data on an external data source. It publishes the returned data if applicable. If data was returned from the provider method then the action can optionally:

- Run any attached display task sets to update the state of the user interface. For example, show, enable, expand and reset UI elements.
- Publish the data. Publishing makes the response (the *topic*) available to any components that subscribe to the action. This is the standard way of handling OnNotification events.

The action set to which the action belongs can also manipulate the data using [data transformations](#), for example, to combine values, perform calculations or aggregations.

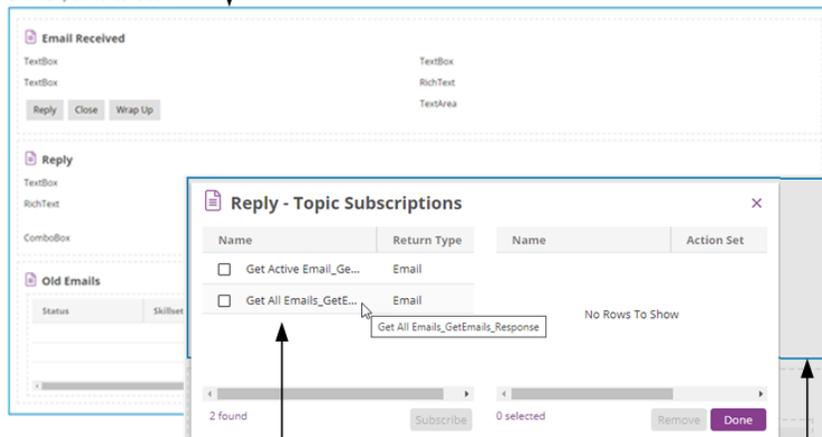
**Note** Actions can access data from a template and events can access the action sets, rule sets and display tasks from a template. See [Deciding when to create templates](#).

## OnNotification events

CE uses a publish and subscribe model to exchange data between the components in the app or template and between providers.

At runtime, the OnLoad event on the outer container runs the action set that publishes the data

Email template 18 feb version 1



Subscriptions in this action set

The OnNotification event on the inner element (or another element in the app) subscribes to the action

*Inner (child) components subscribe to data published by the outer component*

For example:

- An action in an action set is configured to publish a topic, such as get the emails in an email conversation. The action set is part of the event configuration for the app or template (the outer container). When the action set runs, it notifies its subscribers.
- An element in the app or template, such as a data grid, subscribes to the action response (referred to as the topic subscription). The topic subscription is part of the OnNotification event configuration for the inner component or element.
- If there is a data transformation configured then you can choose to subscribe either to the *raw* data (the standard topic subscription), or to the transformed data.

See also [Event configuration for apps and templates](#) and [Event configuration for components](#).

## Creating a standard app or template

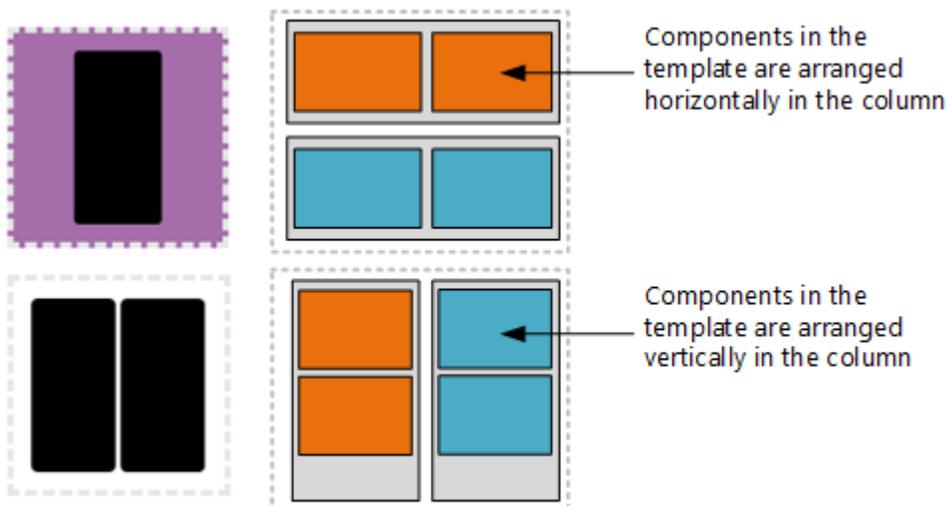
To create a standard app:

1. In CE Studio Designer, on the Apps page, click **Create App** and then fill out the details:

Type	<p>Select <b>Standard</b> or <b>Template</b> depending on what you want to create. Select <b>Standard</b>, for example, if you want to add templates (reusable components) to the app.</p> <p>You can change the type later. See <a href="#">Converting templates and apps</a>.</p> <p><b>Note</b> For an overview of templates, see <a href="#">Deciding when to create templates</a>.</p>
App Group	<p><b>Note</b> You cannot change the app group later.</p> <p>There are several types of app:</p> <ul style="list-style-type: none"> <li>• <b>Portal</b> apps give external users self-service access to CE Studio apps. Not used in Agent Desktop. Users must register on the portal and authentication is provided by CE.</li> <li>• <b>Public Portal</b> apps give anyone with the URL self-service access to CE Studio apps, however, you can implement authentication yourself.</li> </ul> <p>Fully-featured tenants on IFS Customer Engagement can also create:</p> <ul style="list-style-type: none"> <li>• <b>Media</b>: used in Agent Desktop. There may be a different app for each media channel used in Agent Desktop and a separate one for the home page that agents see when in the Idle state.</li> <li>• <b>Wallboard</b>: typically projected onto large screens and displayed to the agents working in the contact center. You can open these from the Admin Portal (on the <b>Reports &gt; Realtime Statistics</b> page).</li> </ul>
Contact Center Unit	<p>You must always set the contact center unit. This is needed, for example, to locate the configured phone numbers, email addresses and response templates.</p>
Access Group	<p>Applies to portal apps only. This is the user group that controls who has access to the portal landing page and its apps.</p>

Category	A label to help you group the apps and templates in CE Studio. You can change the category later.
Description	A description for use in CE Studio Designer. To see or edit the description: go to the Apps page and then click <b>Edit</b> .  <b>Note</b> For portal apps this becomes the description that's shown on the landing page.

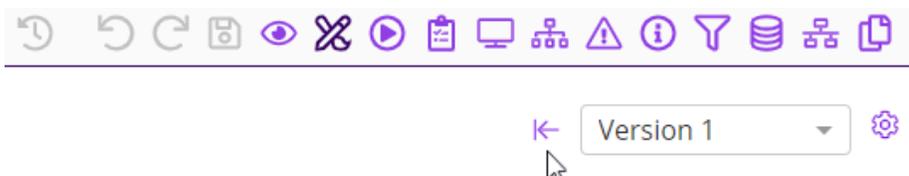
- Select the number of columns in the layout grid. This is just a starting point. For a standard app, you can change this at any time by adding and removing columns. For apps built from templates, the layout option controls how the components in each template are placed on the screen:



- Click **Continue**.

You have now created version 1 of the app.

- To see the version history, click the **Manage Versions** icon:



- To rename the app, go to **Home > Apps** and click in the **Name** column to edit the name.
- To configure the app – if you have gone back to the Home page –click **Manage Versions** and then **Designer**.

## Importing CE Studio Apps

---

There are three ways to import a CE Studio app or template:

- By uploading a ceapp file.
- By downloading from the Central Template Repository.
- By importing a template into a CE Studio app. See [Importing templates into a standard app](#).

### Importing CE Studio apps from an uploaded ceapp file

1. On the Designer page, click **Import App > File Upload**.
2. For non-system providers, select the credentials for the target environment.
3. You can then list the existing providers created for this environment, with these credentials. If there is no provider, then you can create it.
4. Click **+**, check the provider details and then click **Continue**:

Name	Enter a name for the new provider.
Service	The service type from the ceapp file you are importing. For IFS Cloud, you can either type the name, for example <code>RequestHandling.svc</code> , or search for it by typing part or all of the name.
Provider Type	The provider type from the ceapp file you are importing.
Endpoint	The endpoint URL for the target environment. This is from the credential you selected.
Credential Name	Name of the credential that you selected. The credential object has the information to authenticate with the endpoint.

5. The creation of the provider is time sensitive. Once it is created, the **Continue** button becomes available. You can see the progress of the provider creation from the list (for example **Processing, Persisting**). The **Continue** button becomes active if the provider version is successfully created.

## Importing CE Studio apps from the Central Template Repository

**Note** Providers can be created using the default credentials. You can select the credentials to use as the default credentials.

For IFS Customer Engagement, go to the **Admin Portal**, edit the required credentials and select the **Default Credential** option.

1. On the Designer page, click **Import App > Select from Store**.
2. Select an app to import.
3. Click **Create**.

The message `Your app is ready to import` appears when the process is complete.

4. You can either:
  - Import the app using the default settings - click **Advanced** to see what these are.
  - Click **Advanced** to choose the settings for importing the app.

**Advanced** lets you enter a name for the app, a note, select a different credential, provider(s) and provider versions.

5. Click **Import**.

# 3

## Basic end-to-end configuration

Start here if you are new to CE Studio Designer.

If you are new to CE Studio Designer then follow this walkthrough to build a simple CE Studio app for any of these provider types:

- IFS Field Service Management
- IFS Applications 10 and IFS Cloud
- OData 4

### Introduction

---

This section explains what's involved in the end-to-end configuration of a CE Studio app for use in either your public portal or in Agent Desktop, using any of these providers:

- IFS Field Service Management
- IFS Applications 10 and IFS Cloud
- OData 4

This section uses the following simple example.

Calls an API endpoint to get data from IFS Field Service Management

The data grid has a configurable sort order, page size, column styles

**This Week's Tasks**

Description ↑	Critical	Accepted	Add Meters	Cust Plan Start Da...	Cust Plan End Date
Onsite service visit	<input type="checkbox"/>	false	<input checked="" type="checkbox"/>		
Onsite service visit	<input type="checkbox"/>	false	<input checked="" type="checkbox"/>		
Damaged enclosure/casing	<input type="checkbox"/>	false	<input type="checkbox"/>		
Persistent paper jam at location J.	<input type="checkbox"/>	false	<input type="checkbox"/>		
Damaged cable onsite	<input type="checkbox"/>	false	<input type="checkbox"/>		
Product Repair	<input type="checkbox"/>	true	<input type="checkbox"/>		

Page 2 of 24 |< < > >|

**Task Details**

address\_id:

task\_category:

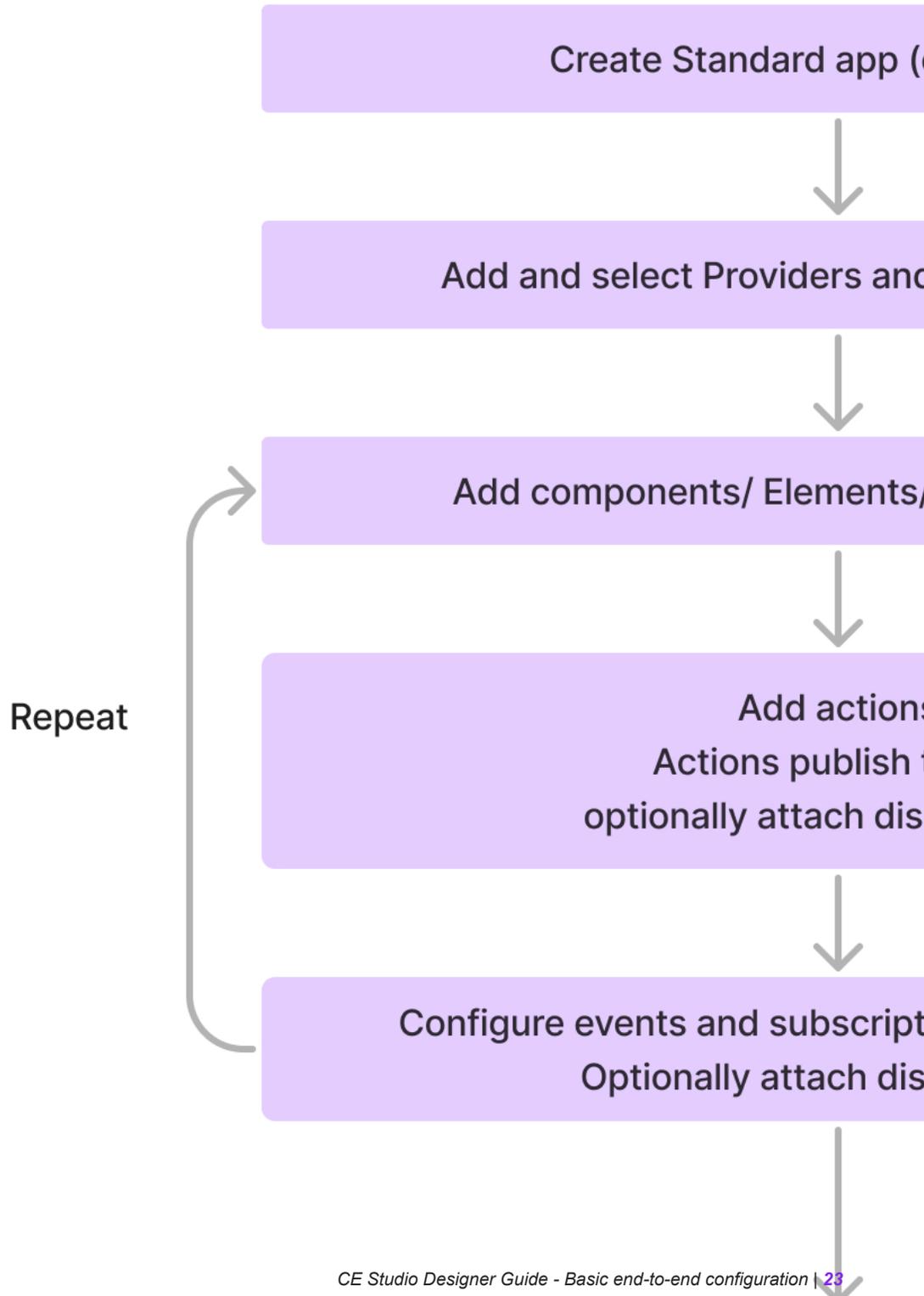
task\_class:

task\_group:

work\_duration:

Form expands and is populated when a row is selected in data grid

The following diagram shows the end-to-end configuration covered in this section:



## Getting started

To get started on a basic CE Studio app for FSM follow these steps. IFS suggest that you start by creating either a public portal app or a media app.

**Note** Only fully-featured tenants can create media apps.

### Create one or more providers

Create a provider as described in [Creating a new provider](#).

You can use:

- IFS Field Service Management
- IFS Applications 10 and IFS Cloud
- OData 4

### Create a standard app

To create a standard app:

1. In CE Studio Designer, on the Apps page, click **Create App**.
2. Fill out the details. The basic fields are described below:

Type	<p>Select <b>Standard</b> or <b>Template</b>.</p> <p><b>Note</b> For a summary of the difference between standard apps and templates, see <a href="#">Working with templates</a>.</p>
App Group	<p>Select the type of CE Studioapp. You cannot change this later. For example:</p> <ul style="list-style-type: none"> <li>• <b>Public Portal:</b> you access a public portal apps on your public portal.</li> <li>• <b>Media:</b> you use media apps in Agent Desktop. There may be a different app for each media channel used in Agent Desktop and a separate one for the home page that agents see when in the Idle state.</li> </ul> <p><b>Note</b> Only fully-featured tenants can create media apps.</p>

Contact Center Unit	Select preferred contact center.
Access Group	Leave this empty.
Category	A label to help you group the apps in CE Studio. You can change the category later.

- Select the number of columns in the layout grid. This is just a starting point. You can change this at any time by adding and removing columns.
- Click **Continue** to create version 1 of the app.
  - To see the version history, click the **Manage Versions** icon:



- To rename the app, go to **Home > Apps** and click the **Name** column to edit the name.
- To configure the app – if you have gone back to the Home page – click **Manage Versions** and then **Designer**.

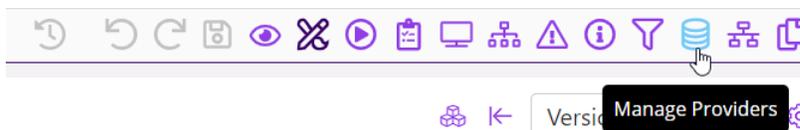
For more details about the Create App page, see [Creating a standard app or template](#).

## Adding providers to an app

- Create a provider if you do not already have one. See [Creating a new provider](#).
- Go to the Designer page for the app that you should have created already:
  - Go to **Home > Apps**.
  - Click **Manage Versions** for the app.
  - Click **Designer**:



- Select the provider and choose the active (live) version:



*Adding the provider and selecting the provider version*

**Note** Alternatively, you can select a numbered version. Whether you choose a numbered version or the active version depends on the upgrade policy you want for the CE Studio app. For details, see [Choosing a provider version](#).

4. When you have multiple providers, then you set one of these as the default by selecting the **Is Default** check box. Typically, the provider you use most often.
5. Optional – click  on the toolbar to view the metadata for the provider. For details, see [Using the Metadata Viewer](#).

**Note** You can add multiple providers to an app, including multiple providers of the same type. For example, you can configure a CE Studio app to use the active version of the provider as well as an older numbered version.

## Add the components

---

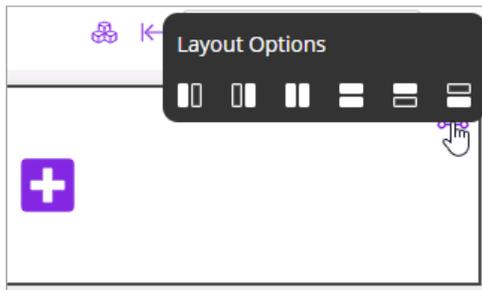
The top-level components in a standard app are, for example, forms, data grids, stack panels, charts, data elements and iFrames. Once you've added one of these, you can add further elements to the component to sit above, below, or alongside it such as a:

- Button group
- Stack panel, for example as a header to a data grid or form
- Data grid
- Rich text box (by converting the field type)
- Chart
- Data element
- Label element

## Adding a data grid

1. In CE Studio Designer, go to the Designer page .

If you chose the single column layout, then the CE Studio app contains a single cell. You can add rows and columns this by selecting from the Layout Options:



2. To add a component: click **+**, next click on **Element** and then select **Data Grid**.
3. Select the properties for the data grid. You can start by selecting the provider:

1. Select provider

2. View the properties for the provider and click to add or remove them

3. Optionally filter the list of provider types

4. Start typing to filter the properties (optional)

Note: Hidden property eg a key field

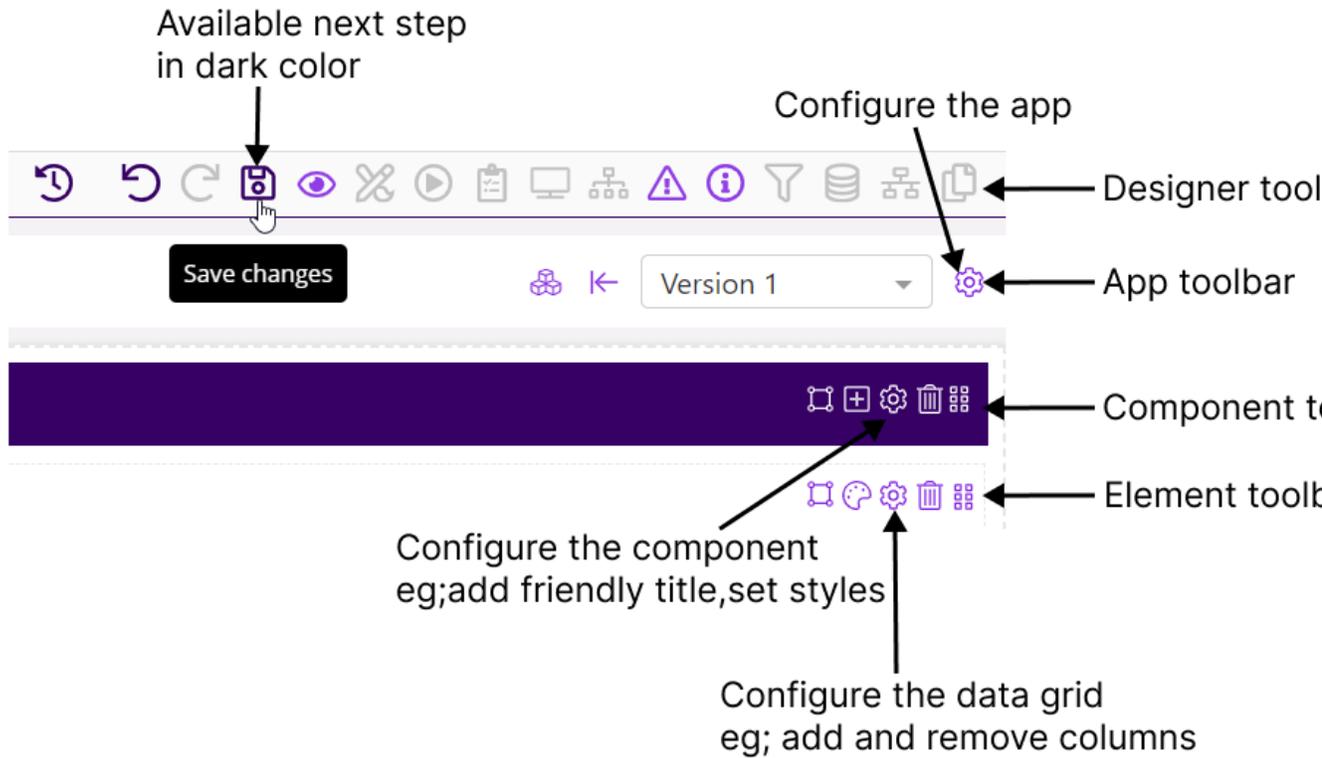
Property Name	Data...	Type	
<input checked="" type="checkbox"/> critical	boolean	task	
<input type="checkbox"/> Property Na...			
<input type="checkbox"/> accepted	boolean	task	
<input type="checkbox"/> task_id	number	task	

You can only add properties from one provider when working with data grids.

Mandatory properties, such as key and id fields, are added automatically. These properties are hidden unless you choose to show them. Scroll to the right to see which properties are hidden. Click to hide and show them.

See [Metadata Selector](#) for details of how to view and select properties.

4. You can add and remove columns from the data grid later by clicking on the data grid (element) toolbar:



#### *Toolbars for configuring the app, its components and elements*

5. To make it easier to configure actions and events edit the data grid's identifier:
  - a. On the component toolbar, click .
  - b. Go to the **Settings** tab.
  - c. The default identifier for the first data grid in an app will be *[title]*. Set the title and enter something more descriptive in the **Identifier** field.
6. Save the app if you want to preview the data grid's header row .

## Formatting a data grid

**Note** Formatting can be conditional based on the value in a data grid field. Other types of formatting such as sorting, filtering, and transforming data, are performed by actions.

### Edit the labels in the UI

To add friendly column headers:

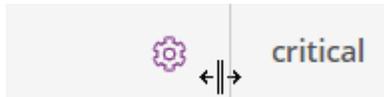
- Click to edit the label or click .

- Click  to *localize* the column header.

### Edit the columns' width and order

End users can choose to switch between displaying a data grid with the columns sized to their default width (the **Auto-Size All** option) or expanding the data grid horizontally to fill the screen space (the **Size to Fit** option). Alternatively, you can change the column widths:

- Hover over the column boundary and drag to resize it, or click  and enter the width in pixels.



Reorder the columns:

- Hover over the header (away from the label and boundary) and press to drag the header to a new position. To pin the column right or left so that the other columns expand to fit, click .

### Make expandable

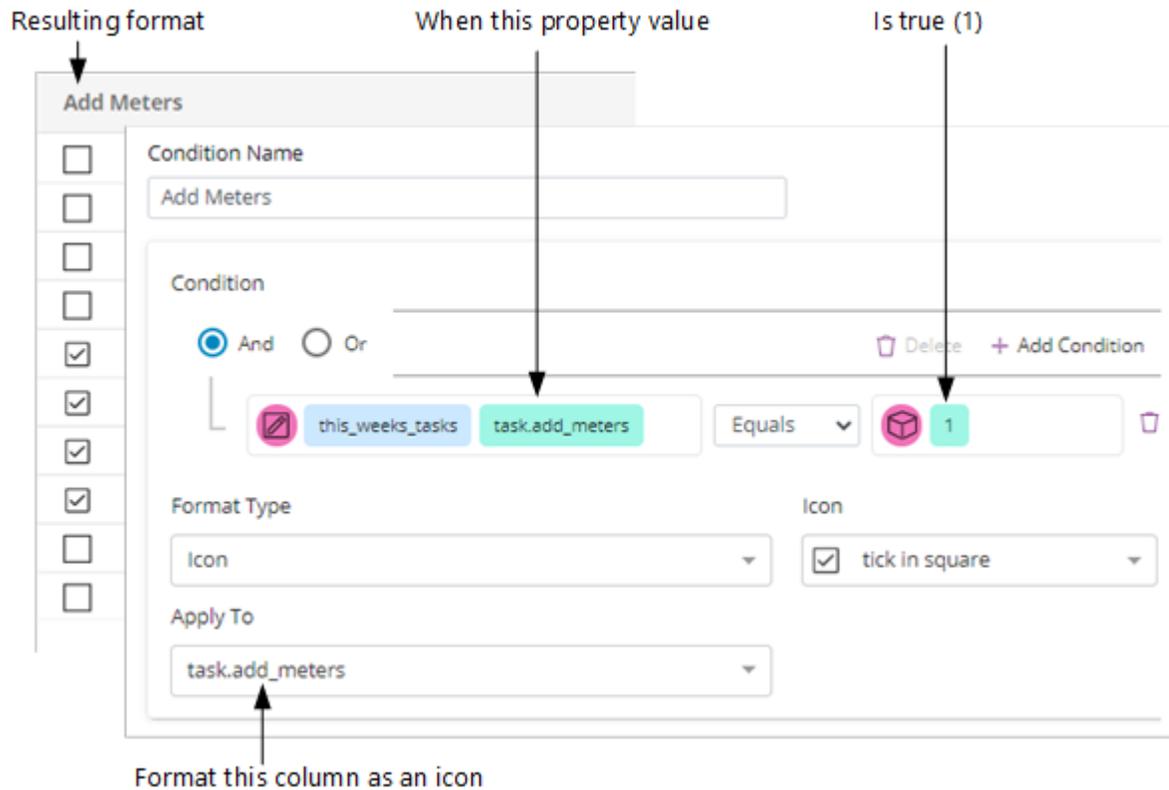
Make the component containing the data grid expandable and collapsible (you can also configure display tasks to control when an element expands and collapses):

- Click  on the component toolbar, go to **Settings** and set **Expandable** to **No**.

### Apply conditional formatting

Formatting can depend on a condition. For example, if a Boolean value is *false*, you can hide the value, show an icon or change the color. When using Booleans in conditions, you need to enter 1 or 0 not *true* or *false*.

To add a condition: save the app  and then click  on the data grid toolbar — properties added to the data grid since the last save aren't shown in the conditions dialog:



*Formatting a Boolean property to display an icon when the value is true (1)*

## Adding a form and UI elements

**Note** This example adds the form as a top-level component. However, you could also add it to a modal dialog using an *overlay component*.

1. Add another row to the app:



2. Add another component. Click **+** and then click **Form**.
3. Select the provider and then some properties. Refer, *Adding a data grid*

If you are following the example in the *Basic end-to-end configuration* section then select the same properties as the data grid.

Scroll to the right to see which properties are hidden. Click  to hide and show them.

4. You can also add UI elements to the form, for example, to get a value from the user to pass to the back end for use in a search. Click  on the component toolbar:

**[title]**

 **Fields**  **Settings**

Search UI elements...

Field Element	
<input checked="" type="checkbox"/>	TextBox
<input type="checkbox"/>	ComboBox
<input type="checkbox"/>	DatePicker
<input type="checkbox"/>	Checkbox
<input type="checkbox"/>	Number

11 found **Add**

*Example of adding a date picker to a form*

**Note** When you add UI elements like this, you must always save the app before attempting to configure the UI elements.

### Converting UI types

On the property toolbar, you can convert a field to a different UI type . For example, you can convert a text box to any of these:

**Ui Types**

 RichText

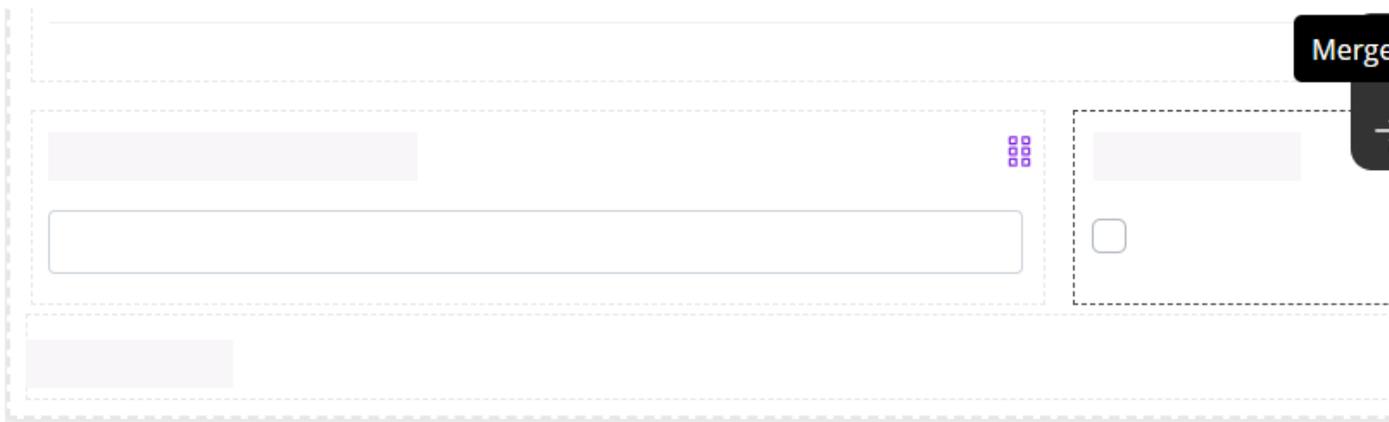
 TextArea

 ChatInput

 ComboBox

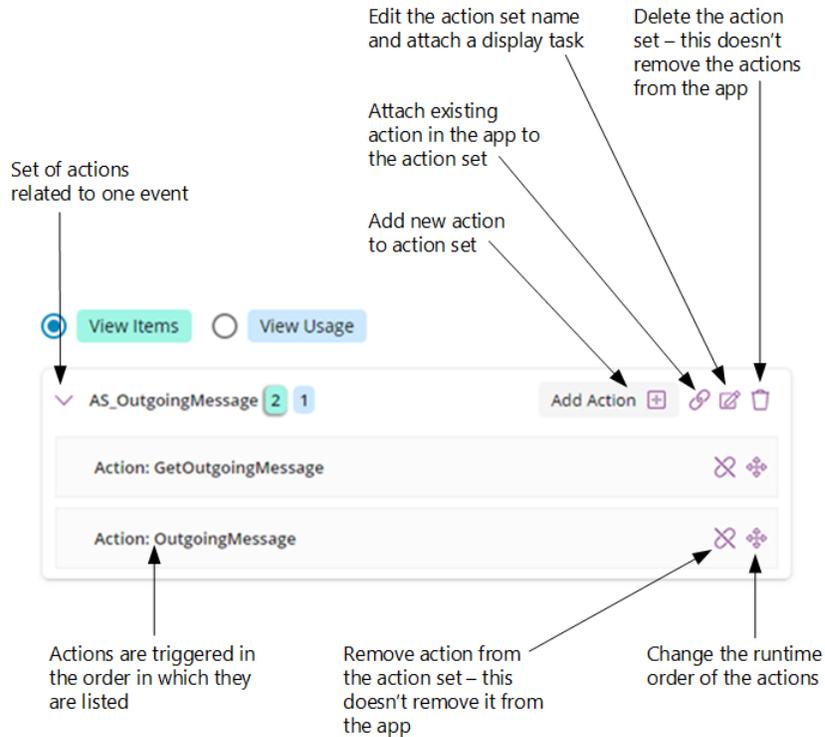
**Tip on merging cells in the layout grid**

If the adjacent cell is empty, you can make the cell full width by clicking  and then choosing **Merge Right** or **Merge Left**:



## Adding actions and configuring events

Actions invoke operations on external data sources. For API calls that return data, the action can then filter, sort and transform the data. Actions run when an event occurs such as the app loading or the selection of a row in a data grid.



*How actions are configured*

Actions can also be used with display tasks to change the user interface and added to rule sets for more complex behavior.

## Adding an action and loading a data grid

Add an action that runs when an OnLoad event occurs for the CE Studio app. The action invokes a get operation on the data source:

1. In CE Studio Designer, go to the Action Sets  page.
2. Create the action:
  - a. Click **Create New Action Set**. The action set groups related actions that need to run at the same time. You can also configure transformations on the data that is returned by the actions in the set.
  - b. Click **Add Action** to add an action to the set:

1. Select provider



Action Name

Get Data\_3

FSM 6.4.5



### Parameter Mapping

#### Suggested Parameters

There are currently no required parameters to show.

### Filters

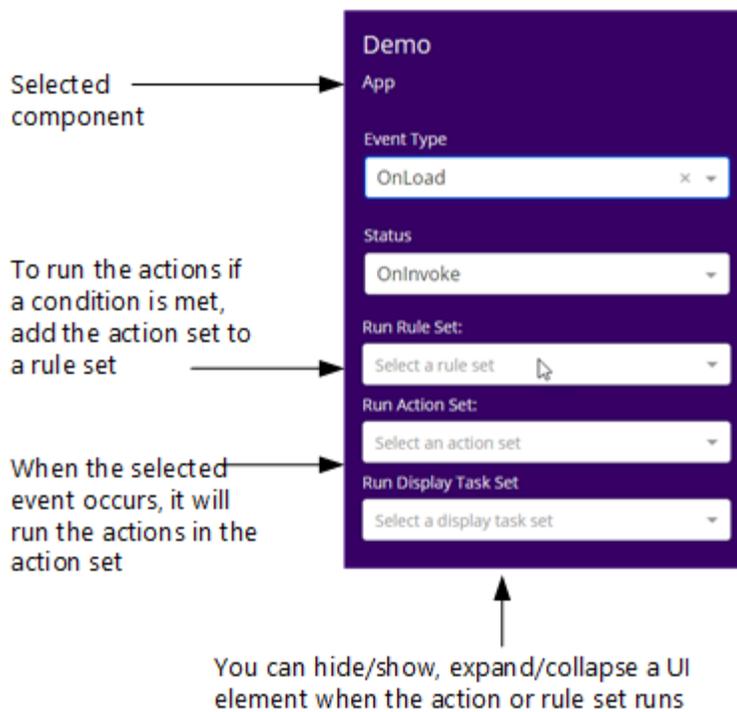
### Transformations

### Sorting

- c. Select **Publish Topic**. Actions that provide data for other components in the CE Studio app must publish the topic. Any component that consumes the data provided by this action must subscribe to the topic.

**Note** For example, it is not necessary for an action that updates a record to publish a topic as this type of action runs in the background.

- d. Save the action.
3. Configure events on the data grid to use the new action. For example:
    - a. In CE Studio Designer, go to the Event Configuration  page.
    - b. Click the outer container (**App** is shown at the top of the event pane):



*Event configuration on the app – the outer container*

- c. Select OnLoad as the event type and OnInvoke as the status.
- d. In **Run Action Set**, select the action set.
- e. Configure the data grid to consume the data returned by the action:
  1. Select the data grid—**Element** is shown at the top of the event pane.
  2. Select OnNotification as the event type.
  3. In **Topic Subscriptions**, select the topic associated with the action set. If the action is not shown then go back to the action and check that you selected the **Publish Topic** check box:

 [title]

 **DataGrid - Topic Subscriptions**

Name	Topic Type	Return
<input checked="" type="checkbox"/> Add_get_Respons...	Standard	address

No Rows

1 found

[Subscribe](#)

0 selected

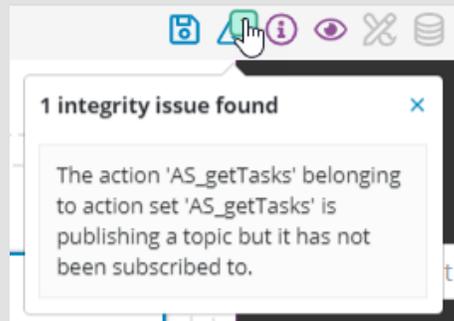
This type of action is referred to as a standard topic.

4. Click **Subscribe**.

**Note** If you later need to modify the action then IFS recommend that you remove the subscription here and reselect it.

- f. Save the event by clicking  on the main toolbar.

**Note** Turn on the integrity checker to verify that for every published action there is an element that subscribes to it.



4. Click  to preview the app.

## On selecting a row in the data grid

This action populates the form when a row is selected in the data grid. A filter on the action only returns the data with a unique ID that matches the ID of the selected data grid row.

### To add an action with a filter

1. Add a new action set.
2. *Add an action* to get the data for the data grid.
3. Select **Publish Topic**.
4. In **Filters**, click **Add Condition** to add an AND condition:

And     Or

 Mapping Field Field Reference

Source	Entity
Filter	Filter
Field Value >	address
<b>Field Reference &gt;</b>	address.contact
Static >	address.part_need
Global Variable >	address.payment
System Variable >	address.place_address
Toolbar Query Param >	address.product
<b>11 found</b>	<b>9 found</b>

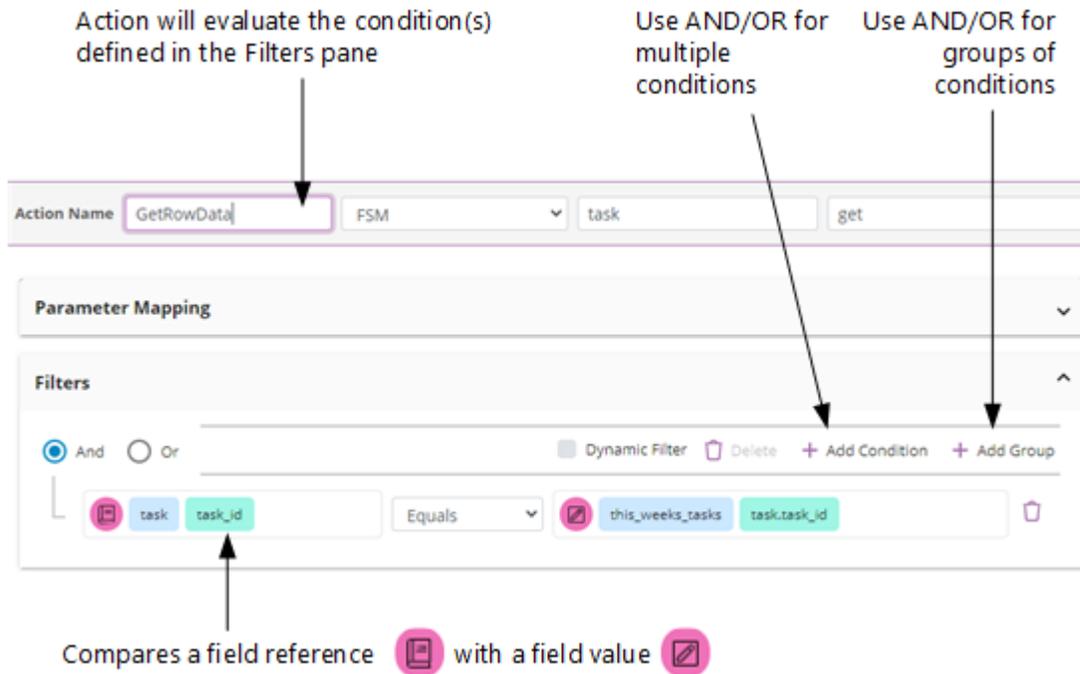
*Example of mapping the left side of the condition (the field reference)*

 Mapping Field Field Value

Source	Component	Field
Filter	Filter	Filter
Field Value >	All Components >	Company
Field Reference >	[title] >	T Company
Static >		T Company
Global Variable >		T TextBox-
System Variable >		≠ address.a
Toolbar Query Param >		📅 address.a
<b>11 found</b>	<b>2 found</b>	<b>7 found</b>

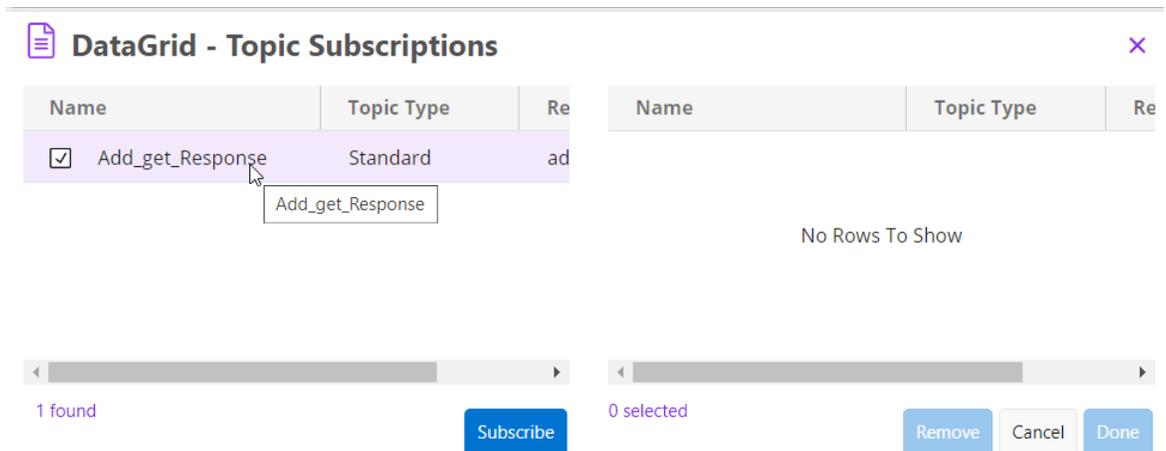
*Example of mapping the right side of the condition (the field value)*

The resulting condition:



Example of the resulting condition

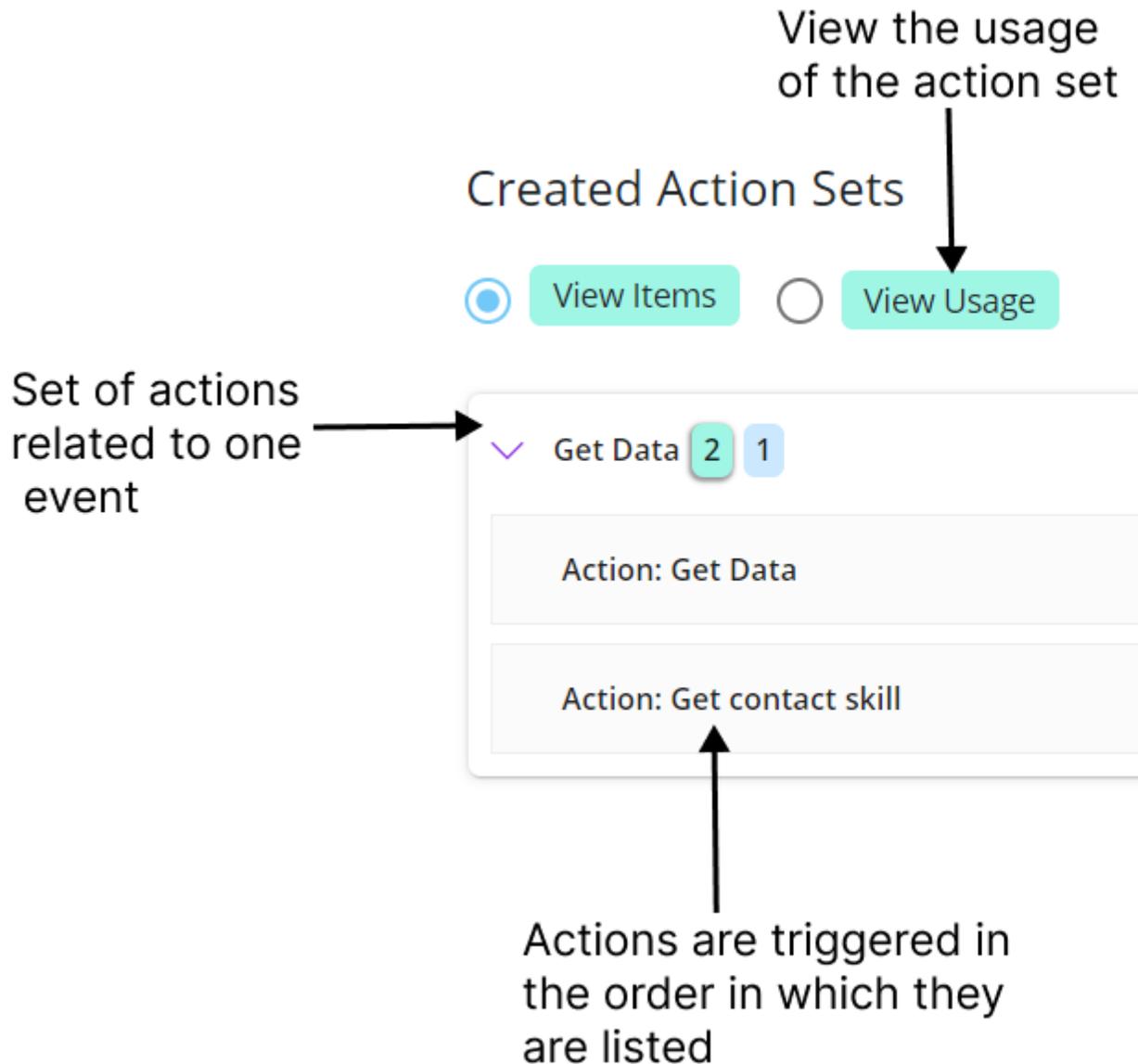
5. Configure the data grid for the row selected event:
  - a. Go to Event Configuration .
  - b. Click the data grid.
  - c. Select the OnRowSelected event type.
  - d. Select the action set you added.
  - e. Set the form to subscribe to the topic that contains the response:
    1. Click on the form.
    2. Select OnNotification as the event type.
    3. Select the subscription and click **Subscribe**:



6. Save and preview the app. In Preview, when you click a row in the data grid, the form updates to show the data for that row.

## Finding out how the action sets are used

You can find out how the action sets are used on the View Usage tab:



1. In CE Studio Designer, go to the Event Configuration  page.

2. Click the outer container (**App** is shown at the top of the event pane).
3. Select the **OnLoad** event type.

The **Run Action Set** fields show the action set that runs when this event occurs.

4. Click the action set name.

This takes you to the Actions page.

5. Click **View Usage**.
6. You can now expand each of the action sets to see how these are used in the CE Studio app.

## Sorting and paging the data grid

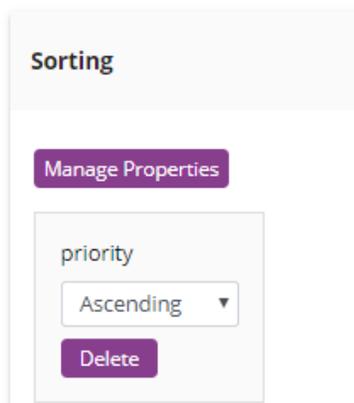
Actions can set the page size and how the rows are sorted. To configure this, edit the get action for the data grid.

### Set the number of rows per page

To set the page size, edit the get action and then go to **Paging**. The default page size is 10.

### Sorting the results

The property you choose to sort on becomes the default sorting method for the data grid. Users can click a column header to change the sorting. By default the data grid is sorted by its first column.

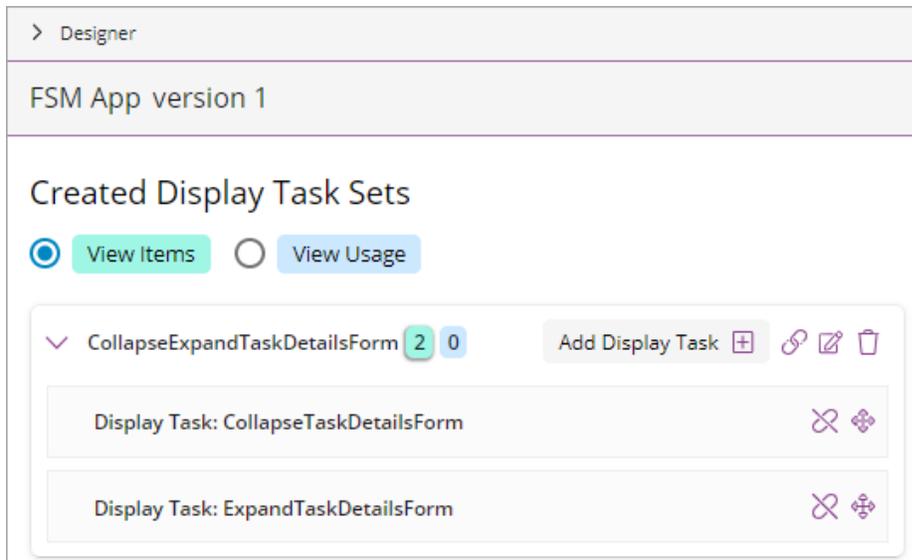


To set the default column to sort on:

1. Edit the action and then go to **Sorting**.
2. Click **Manage Properties** and choose the property to sort by.
3. Select an ascending or descending sort order.

## Expanding and collapsing UI elements

You use display tasks to update UI elements when an event occurs and an action runs.



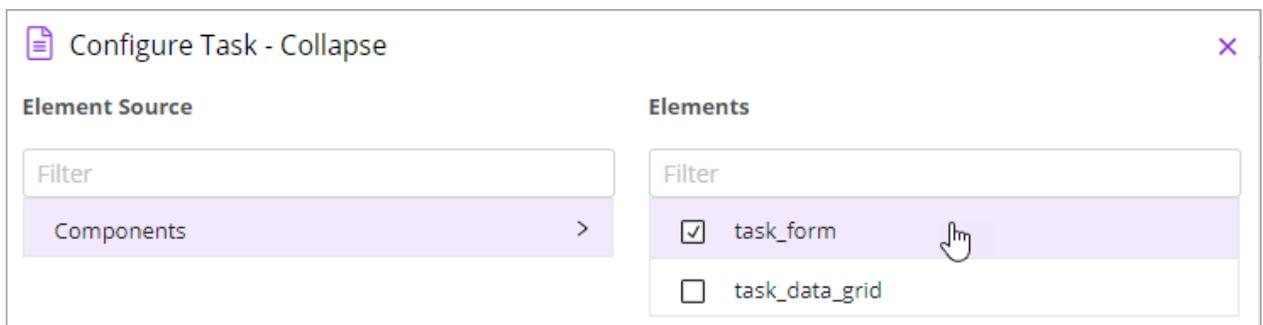
*Display tasks used in this example*

To configure a display task to collapse the form when the app loads:

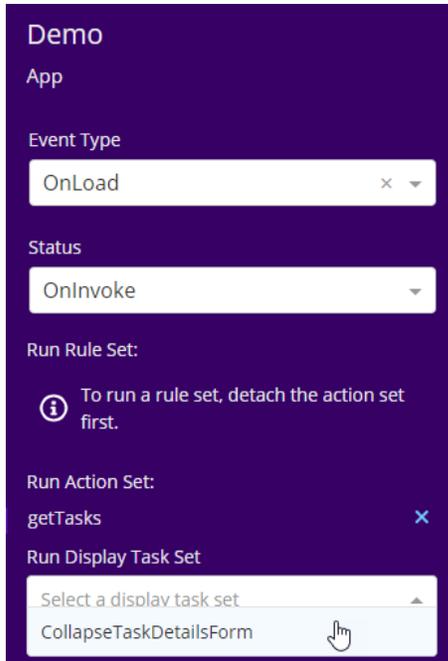
1. Go to the **Display Task Sets**  page.
2. Create a new display task set.



3. Add a **Collapse** display task to the set:



4. Save the display task.
5. In Event Configuration, modify the app's OnLoad event to also run the display task:



6. Save and preview the app.
7. To complete this example, you could also configure a display task to expand the form when a row is selected in the data grid. You can add it to the same display task set.

### Usage on display tasks

To use a display task, you must attach it to something. You can attach them:

- To events as in this example.
- To action sets—edit the action set and select a display task.
- To rule sets to run one action set and display the task if a condition evaluates to *true* and a different action set and display task if it evaluates to *false*.

## Preview in browser

To check the appearance of the CE Studio app, click  to preview it in a browser. To fully test a CE Studio app, you must always test a public portal app in the public portal realm and a media app in Agent Desktop.

Preview is useful to check layout and styling, and that display tasks and conditions (in actions and rule sets) are working as you expect.

## Previewing media apps

There are some limitations on what you can preview if the media app is intended for handling activations in Agent Desktop, such as inbound email and voice calls, chat because any expected events from Agent Desktop are ignored.

For example:

- Actions that rely on an incoming call ID or activation ID will be ignored as there are no live activations in preview mode – instead you will see the message `Agent Desktop events are ignored in preview mode.`
- Display tasks that are triggered to run by these actions will also not run, and features will not hide, show, expand, and so on, as they would in Agent Desktop.
- Template options that use display tasks to hide and show specific features in the template will also not run. Again, you need to test this in Agent Desktop.

## Make the CE Studio app live

---

You make the version of the app live when you are ready to use the app, such as:

- Use it as a template in another app
- Use it as the default public portal app in your public portal
- Use it as a default media app in Agent Desktop

To make the CE Studio app live:

1. You can either:
  - Go to the Designer page and click **K** (this will navigate to the **Manage Versions** page).
  - Go to **Home > Apps** and click **Manage Versions**.
2. For the version that you want to test, select the **Live** check box.

You can no longer make any changes to the version live.

**Note** To make changes to the live version, you need to copy a new version. There are two ways to do this:

- On the Manage Versions page, click **Copy new version**.
- Click **Manage Versions** to open the CE Studio app and then on the toolbar, click **Copy as new version**.

Depending on the app, you may also be able clear the **Live** check box.

## Test the public portal app in the public portal realm

---

Each tenant has a public portal. To test the public portal app in the public portal realm:

1. In CE Studio Designer, go to the side bar and click **Portal Settings**:



2. From the Default Public Portal Page, select your CE Studio app. If your app is not shown then you need to make it live. See [Make the CE Studio app live](#).
3. In your browser, using incognito mode, go to your public portal.

If the URL of CE Studio Designer is:

```
https://<test>/<tenant>/ce/0/studiodesigner/
```

then the URL of the default CE Studio app for the public portal is:

```
https://<test>/<tenant>/ce/0/public/
```

**Note** There is also a way of testing a public portal app without making it live. See [Public portal URLs](#) for details.

## Test the media app in Agent Desktop

---

Follow these steps to test a media app in Agent Desktop.

**Note** You cannot test a public portal app in Agent Desktop.

### Associate the app with a media type

1. In CE Studio Designer, go to **Home > Default Media Apps**.
2. If you are following the steps to create an example app then you can set the example app as the **Default Home Page**.

This means the example app will load when you first sign on to Agent Desktop. If you can't see your CE Studioapp then check that you made it live.

**Note** Apps configured for inbound email, voice calls and so on, aren't suitable for the home page because they require an activation (email, call, chat) to be present. Select the appropriate media type for these. See [Linking CE Studio apps to Agent Desktop media types](#).

## Sign on to Agent Desktop

The Agent Desktop URL depends on the tenant, for example, if the URL of CE Studio Designer is:

```
https://<test>/<tenant>/ce/0/studiodesigner/
```

then the URL of Agent Desktop is:

```
https://<test>/<tenant>/ce/0/consolelite/
```

Once you have signed on, you should see the app selected as the Home page. If you selected a different media type, then you need to create the activation or go to the state that triggers the app to load.

For details of how to use Agent Desktop, see *IFS Customer Engagement Agent Desktop User Guide*.

# 4

## Working with templates

Working with templates, sharing data between apps and templates.

In CE Studio Designer you can choose to create either standard apps or templates. This section explains how to create templates and how to share and access data between apps and templates.

### Adding a template as a component to an app

A template is a reusable component that is intended for use in different CE Studio apps. Templates are self-contained components, and a template has limited knowledge of the standard apps which embed it. This ensures that you can update the template whenever you need to and, when you make that version of the template live, automatically push the updates to all the apps that embed the template.

When you use a template in a standard CE Studio app, you always use the Live version of the template. Whenever you make a new version of the template live, you will immediately upgrade all the apps that embed the template. When importing a CE Studio app, you can only map to a template that has a live version.

#### Embedding a template in an app

To use a template in an app, you:

1. Add an empty cell to the app.
2. Click  and select the template.
3. Depending on the template further configuration may be needed:

<b>Options tab</b>	On the <b>Options</b> tab, select the options configured for the template (if any) or skip this step if you want to use the default options. For example, the standard IFS CE Email template has options to hide or show the Follow up emails data grid.  For details, see <a href="#">Adding configuration options to templates</a> .
<b>Placeholders tab</b>	On the <b>Placeholders</b> tab, select the template that you want to use in the placeholders. A template can have several placeholders for other templates or there may not be any at all.  For details, see <a href="#">Using templates with placeholders</a> .
<b>Outer App Mapping tab</b>	On the <b>Outer App Mapping</b> tab, map the placeholders configured in the template. For example, if there is a placeholder for an email address then map to the field value that will provide the email address.  For details, see <a href="#">Configuring the template to use data from the app</a> .

#### 4. Click **Embed**.

This will embed the template in the app — you automatically take the live version of the template.

To go back later and change the options, placeholders and outer app mapping, click  on the component toolbar.

From the standard app, you can:

- Access the action sets, display tasks and rule sets in the template, for example on the Event Configuration page where you choose between **App** or **Template** as the source for action sets, display tasks and rule sets.
- Use data in the template when configuring action sets, rule sets and display tasks, such as field values from forms or currently selected row in data grids. See [Configuring the app to use data from the template](#).

## Deciding when to create templates

The primary purpose of a template is as a reusable component that you can use in different CE Studio apps.

By design, templates are self-contained. This ensures that you can update the template whenever you need to and, when you make that version of the template live, automatically push the updates to all the apps that embed the template.

**Note** You can break the link between a template as used in an app and the template file by importing the template into the app.

### Considerations when creating apps that will use templates

When developing standard apps, consider these possibilities:

Standard App	See
Convert app to template to make the 'app' reusable.	<a href="#">Converting templates and apps</a>
Add template to app and convert the template to standard components by inserting it.	<a href="#">Importing templates into a standard app</a>
App runs actions, rules and display tasks in the template.	<a href="#">Configuring the app to use data from the template</a>
App sets fields using values from the template.	<a href="#">Configuring the app to use data from the template</a>

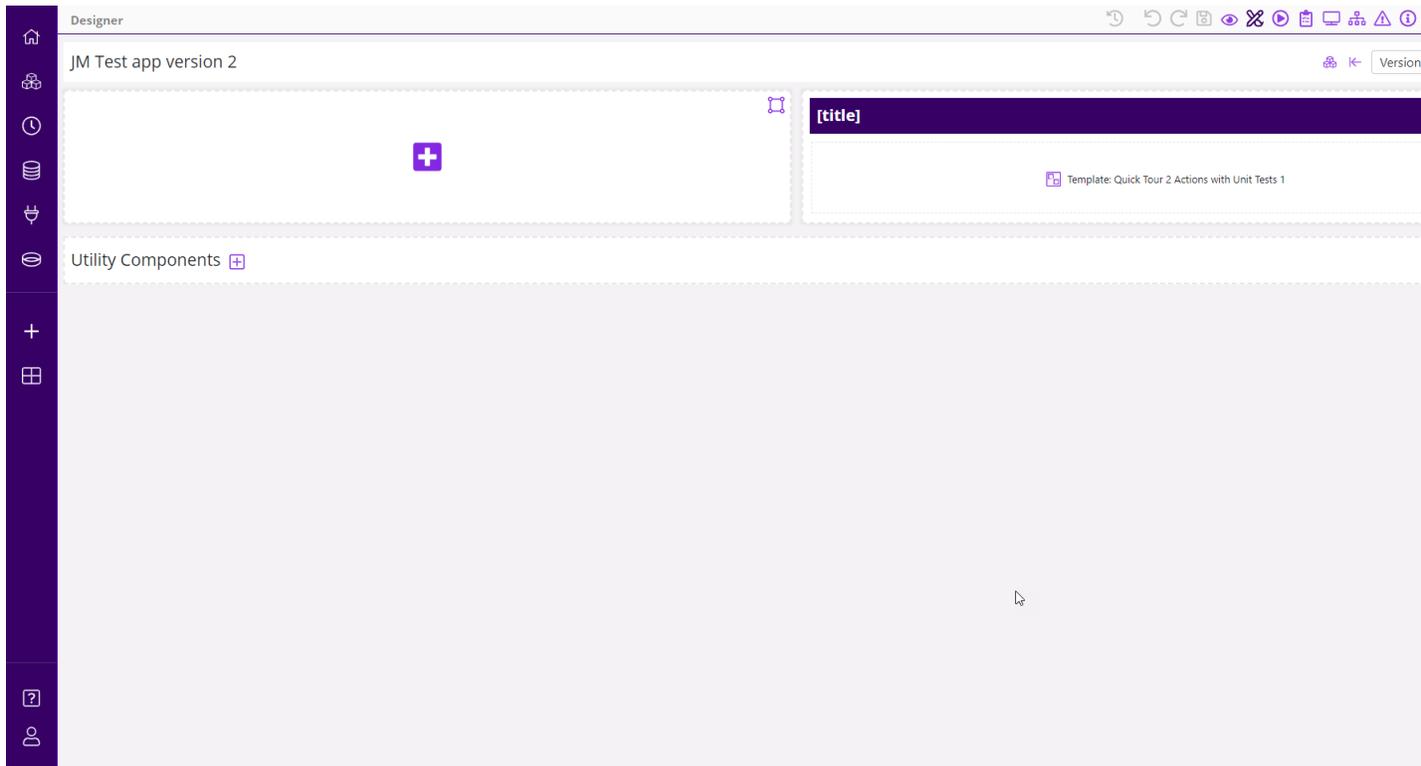
### Considerations when creating templates

When developing templates as reusable components, consider these possibilities:

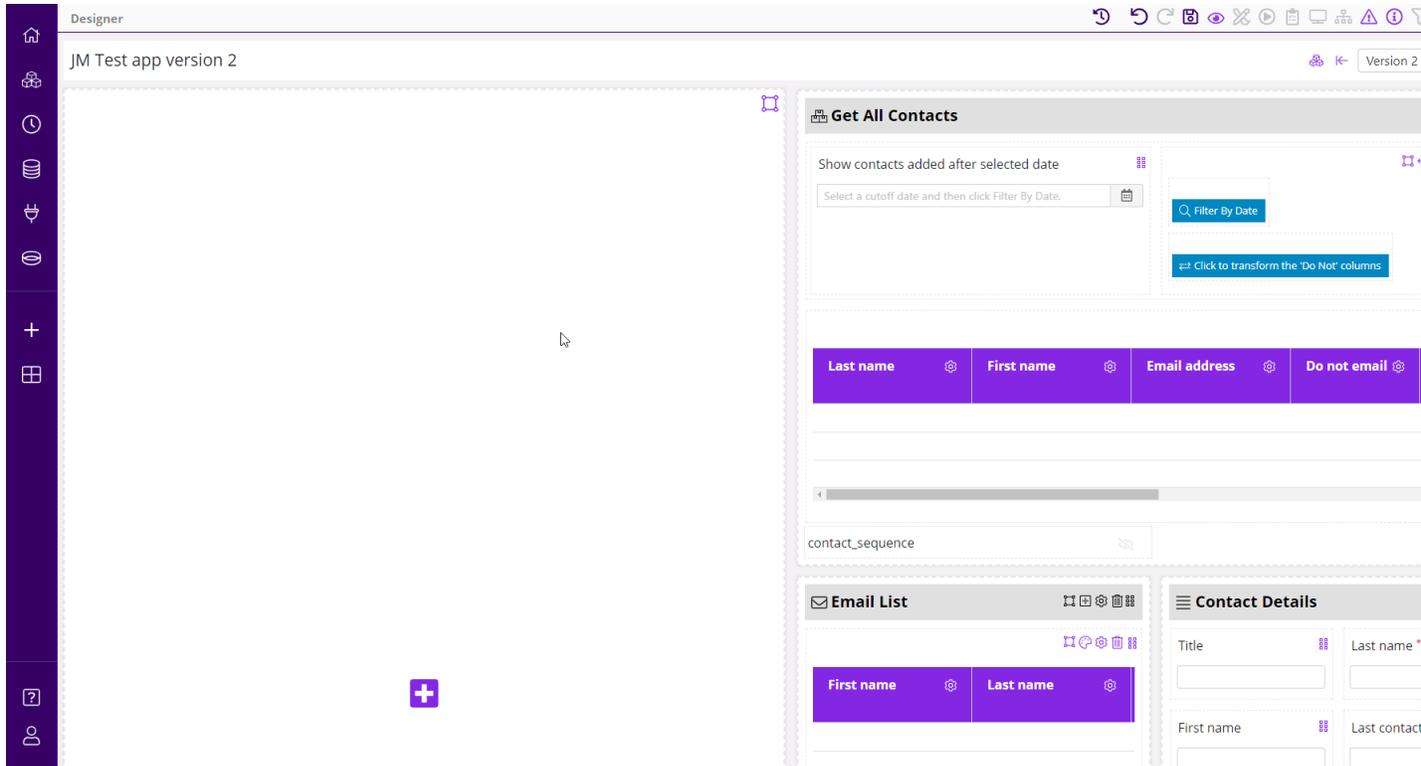
Requirement	See
Convert template to app, for example, to add another template to it.	<a href="#">Converting templates and apps</a>
Template can set fields using values from the app.	<a href="#">Configuring the template to use data from the app</a>
Template can pass data to the standard app.	<a href="#">Public topics and templates.</a>
There are several more advanced options for when you need to configure large numbers of apps with slightly different requirements.	<ul style="list-style-type: none"> <li>• <a href="#">templates with placeholders</a></li> <li>• <a href="#">Adding configuration options to templates</a></li> </ul>

## Importing templates into a standard app

When you insert a template into a standard app, you copy the components in the template into the app, and remove the link to the original template. This means that changes to the template will not reflect in the standard app.



*An embedded template in app*



*After inserting the template - the components in the template are copied into the app*

When inserting templates to apps, system includes following restrictions :

- You cannot use media type templates inside Public Portal and Portal Apps.
- The Public Portal and Portal templates are allowed only within Public Portal and Portal Apps.
- With wallboard and media type apps , you need to use wallboard and media type templates only.

You can insert template into an app (public portal, portal, media) provided that the template:

- Does not use providers that are not supported in the app. For example, you cannot insert a media template into a public portal app if the media template uses the IFS CE Agent Desktop provider.
- Does not contain template placeholders - see [Using templates with placeholders](#).
- If it contains template options then these will be converted to static values - see [Adding configuration options to templates](#).

**Note** On insert the action sets, display task sets and rule sets from the template are renamed so that the origin of the item is clear. The name of the template is added as a prefix to the original name.

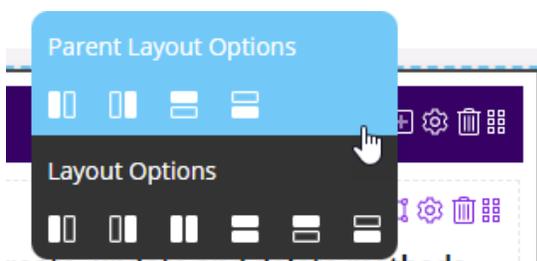
## How to insert templates into apps

There are two ways of inserting a template into a CE Studio app:

- Either, select the template to insert in the usual way and then click **Insert into App**.
- Or, if you have already embedded the template, click  on the template toolbar and then click **Insert into App**.

## Layouts and inserted templates

The embedded template occupies a single cell in the layout grid. When you insert the template, then the template remains in that layout cell (the parent layout) and each component is copied to a child cell. This is why when you add rows and columns you see this toolbar:



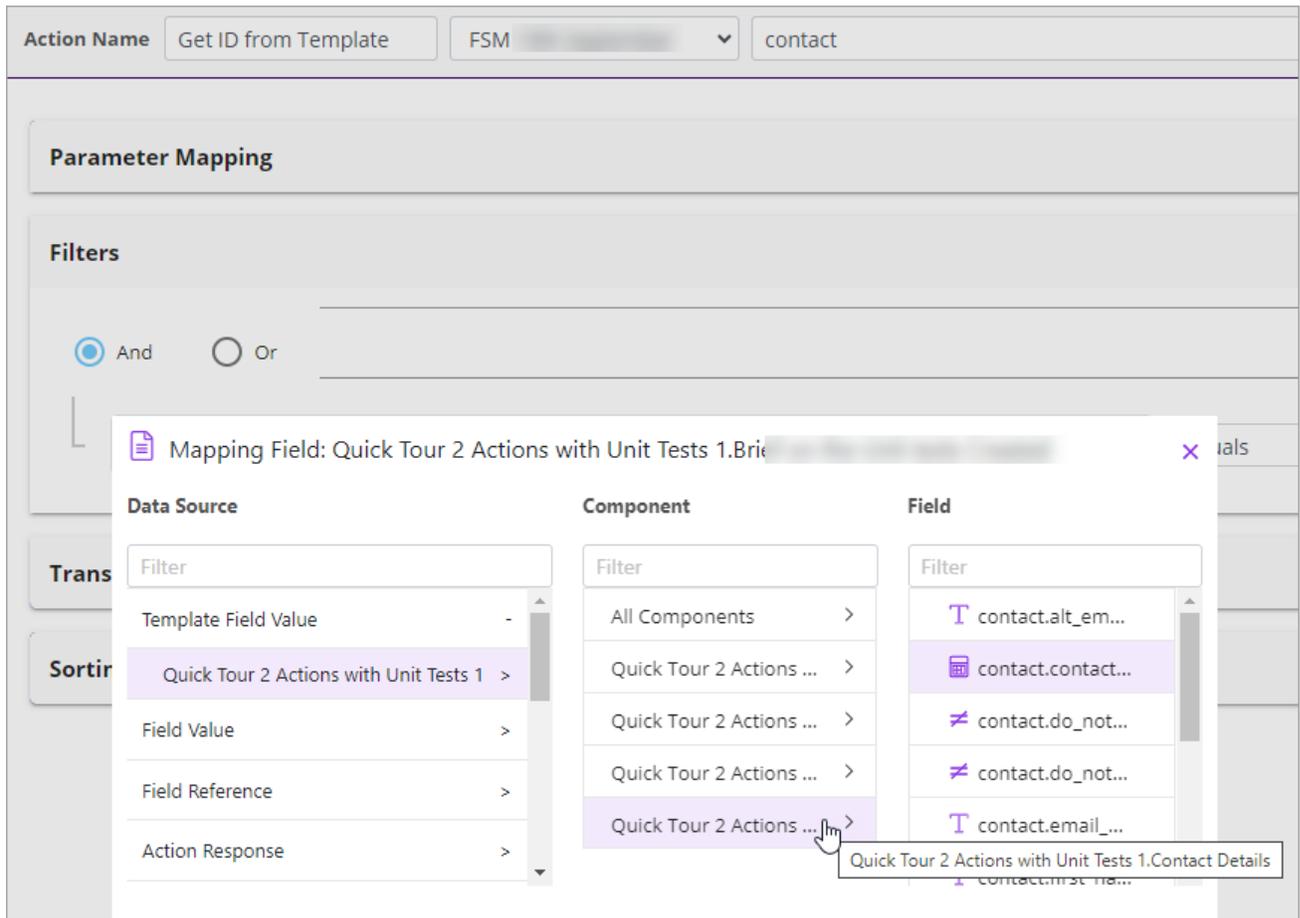
Select from the **Parent Layout Options** to add a row above or below the original template, or the **Layout Options** to add a row above or below the current component.

## Configuring the app to use data from the template

The outer app – the standard app that uses the template – can access the field values in the template. You can therefore configure actions, display tasks and rules to make use of field values set in the template.

**Note** In some cases you need to add the providers in the template to the standard app before you can access certain field values. Specifically, this applies to static, predefined values (enums) which are a property of the template's provider.

## Action or rule in an app gets field value from a template



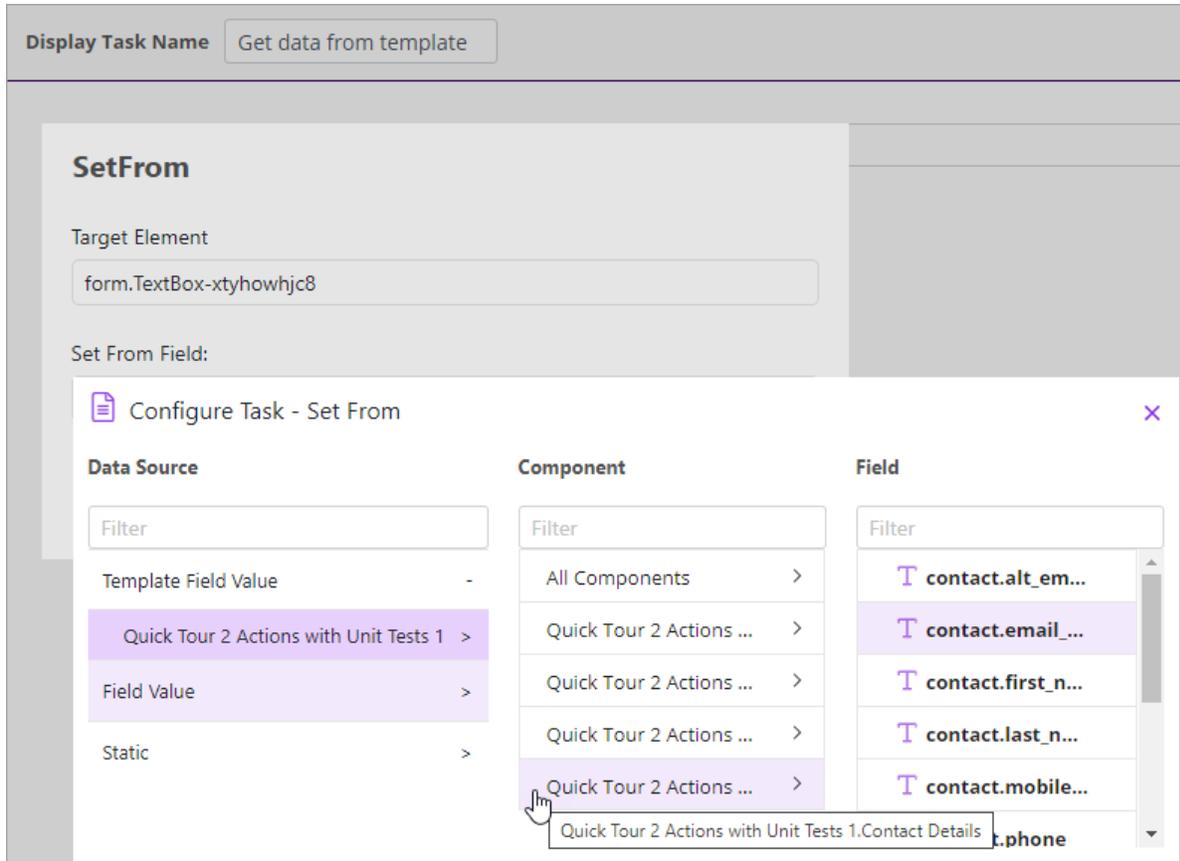
*Actions configured for a template field value*

To use a field value from a template in an app:

1. In the standard app, embed the template that you want to use.
2. In the action or rule add a condition.
3. On the left or right side of the condition or rule, select **Template Field Value**. All the components in the template are listed.
4. Select the required component and field.

## Display task in app gets field value from template

For this you need to use the Set From display task.



To use a field value from a template in Set From display task:

1. In the standard app, embed the template that you want to use.
2. Add a Set From display task.
3. In **Target Element**, select an element in the app.
4. In **Set From Field**, select **Template Field Value** and the template to use.
5. Select the component in the template and then the required field.

This field will effectively act as a placeholder.

## Configuring the template to use data from the app

You can configure actions, display tasks and rules in a template to make use of field values set in the outer app. The outer app is the standard app that uses the template.

### Display task in template gets field value from app

For this you need to use Set From display task.

Display Task Name

Please select a sub task...

### SetFrom

Target Element

Set From Field:

Data Source	OuterAppData
<input type="text" value="Filter"/>	Name
Field Value >	<input type="text" value="email"/>
Static >	Data Type
<b>Outer App Data &gt;</b>	<input type="text" value="string"/>

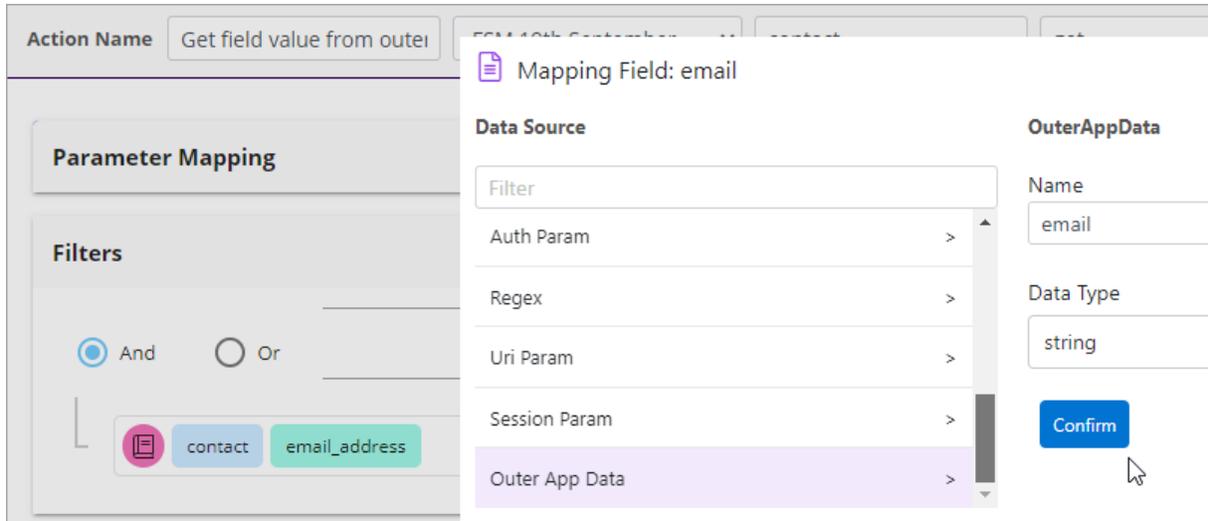
*A Set From display task configured to get data from the outer app*

To use a field value from an app in a Set From display task:

1. In the Set From display task.
2. In **Target Element**, select an element in the template.
3. In **Set From Field**, select **Outer App Data**.
4. Enter the name of the field and select the data type.

#### Template gets field value from outer app

You can design the template to use field values from the standard apps that use the template. This app is referred to as the outer app. For example, the outer app provides the email address and the template has the action to send emails.



*Action configured to use Outer App Data as the data source*

1. In the template, add a condition to the action or to a rule.
2. On the left or right side of the condition or rule, select **Outer App Data** as the data source.
3. In **Name**, enter a name to use as a placeholder.
4. Select the data type.
5. Save the action.
6. Still in the template, configure a display task to populate the placeholder. The outer app will use this display task.
  - Use a type such as Set To or Combined.
  - For the data source, select **Outer App Data** and select the placeholder you configured.
7. In the outer app:
  - a. Embed the template and, on the **Outer App Mapping** tab, map the placeholder to something in the app.
  - b. Configure the app to use the display task in the template.

## Converting templates and apps

---

### Converting a template into a standard app

You can convert a template to a standard app. You might want to do this in order to embed or insert another template into it. You cannot embed or insert templates into templates only into standard apps.

1. In CE Studio Designer, on the Apps page, locate the template and click **Edit**.
2. From the **Type** list, select **Standard**.
3. Click **Update**.

An error is displayed if there are reasons why the template cannot be converted.

There are some restrictions on converting a template to a standard app:

Template is used by other apps	Clone the template to continue.
Public resources are configured for a version of the template	<p>Clone the template and on the <b>App Settings &gt; Public Resources</b> tab delete the resources listed.</p> <p><b>Note</b> Public resources are created automatically on the <b>App Settings &gt; Public Resources</b> page of the template, when an app that uses the template configures the app to use a display task in the template.</p>
Options are configured for a version of the template	<p>Clone the template and on the <b>App Settings &gt; Options</b> tab delete the listed options (there may be associated rules to remove first)</p> <p><b>Note</b> Options are configured as a rule in the template. See IFS CE Email template for an example and the Hide Show Follow Up Grid rule.</p>
Template placeholder is configured for version of the template	Remove the template placeholder.

### Converting a standard app to a template

You can convert a standard app to a template to create a reusable component that you add or insert into another standard app:

1. On the Apps page in CE Studio Designer, locate the standard app and click **Edit**.
2. From the **Type** list, select **Template**.
3. Click **Update**. An error is displayed if the app contains a template.

You cannot convert a standard App into a template if contains a template or an inserted template. To work round this either delete the template or insert it.

## Adding configuration options to templates

When designing templates, you can build in some configuration options for the apps that use the template. For example, to:

- Set a hidden or readonly field to a specific value
- Set a value to be used in conditional logic (as below)

**Note** Options are converted to static values when the template is imported into the CE Studio app.

In the following example, the template has a Reply button and a Contact History data grid that can be hidden or shown depending on the requirements of the app that embeds the template.

The screenshot displays two side-by-side panels. The left panel, titled 'Email Form', contains two input fields and two buttons labeled 'Reply' and 'Cancel'. The right panel, titled 'Contact History', shows a data grid with columns for 'To' and 'From'. Below the grid, it indicates 'No Rows To Show'.

*The example that's used in this section*

This example template is configured like this:

### App Settings

In the template, each option that can be set in the app is added as a key-value pair.

**App Settings**
✕

Styles
Options

---

Key

Value

 ✕

Use the enter key to add

true
false

Key

Value

 ✕

Use the enter key to add

true
false

Add Option
Delete Selected Option(s)

Cancel
Continue

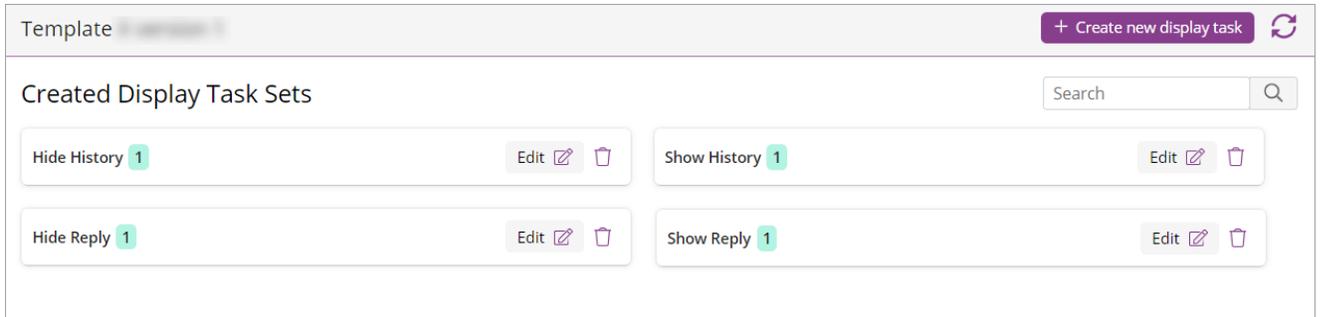
*Example of configuring the template options*

1. On the app toolbar, click  and then go to the **Options** tab.
2. Add two options where **Key** is the name of a rule set in the template and **Value** is the allowed settings.
3. Click to select the value that will be the default for each option.

**Note** If you use a template directly as an app in Agent Desktop then you automatically use the default set for each option.

### Display tasks

In the template, there are separate display tasks to show and hide the Reply button and data grid.

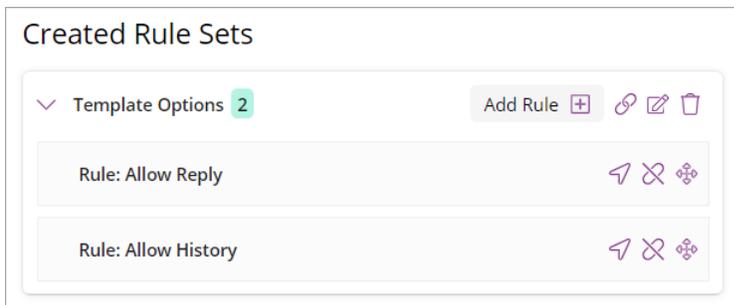


*Example of display tasks to hide and show the button and data grid*

The display tasks are attached to a rule set.

### The rule set

In the template, there is a single rule set, with one rule for the button and another rule for the data grid. Depending on how each option is set in the app, the rule set runs the corresponding display task.



*Example rule set that runs the hide and show display tasks*

For example, this is the rule that shows the Reply button:



For each option configured in App Settings, the rule set applies one rule if the app sets the option to *true* and the other rule if the app sets the option to *false*:

The screenshot shows a configuration panel for a rule set. It is divided into two main sections: 'When True (Default)' and 'Otherwise'. Each section has a 'Run Display Task' field with a selected option ('Show Reply' and 'Hide Reply' respectively) and a 'Run Action Set' dropdown menu set to 'Select an action set'. Below each section, there is a note: 'No filters configured. Create filters'.

*Example conditional logic used by the rule set*

In the template, the rule set is run when the OnLoad event is invoked.

**Important** If the invoked rule set (or the attached action set or display task set) depend on an incoming activation ID then the current configuration of the option cannot be applied when there is no activation ID and the default for the option is used instead. This applies, for example, when the template is used as a home page app or when previewed in the browser.

### How this is used in apps that embed the template

In the app, the template is added in the usual way. To set the options to the required values, you edit the component that contains the embedded template:



And then set the template options as required or accept the default:

**App Template** [Close]

Options [Placeholder]

Key	Value
Allow Reply	true
Show History	true

[Cancel] [Continue]

## Public topics and templates

When you configure actions to publish a topic, the topic is private. This means that a standard app that embeds a template *cannot* subscribe to any of the topics published by actions inside the template. To make topics available outside the template, for example to the standard apps that embed the template, you need to configure these topics as *public*.

**Important** The template and the apps that use the public resources must be configured with *exactly* the same provider. Providers created for the same version of a provider are not identical because they have a different internal identifier.

**Note** You cannot make transformation topics public.

### Configuring a public topic in a template

To configure a public topic in a template:

1. Click **App Settings** on the app toolbar.



2. On the **Public Resources** tab, click **Add Public Resource**.
3. Enter a name for the topic – you cannot rename this later.
4. In the **Topic** field, click to list the Standard topics that are published by this template and then select the topic.

You cannot make transformation topics public.

## 5. Click **Continue**.

### Subscribing to a public topic in an app

Components and elements subscribe to public topics in the same way that they subscribe to other topic types, for example:

1. Create a standard app. The app must use exactly the same provider as the template.
 

This means that the app and the template cannot use different providers even if the providers are for the same service type.
2. Add a template that is publishing public topics.
3. Create the app in the usual way. For example, add providers to the app, and add components and elements, such as data grids and data elements.
4. Set up the subscription:
  - a. On the event configuration page , click the component or element.
  - b. From the **Event Type** list, select **OnNotification**.
 

The Topic Subscriptions dialog is displayed.
  - c. In the Topic Subscriptions dialog, subscribe to the topic. Its topic type is Public.

## Using templates with placeholders

---

Templates with placeholders are an advanced feature for CE Studio administrators who need to configure and maintain large numbers of apps where some of the apps have slightly different requirements. For example, where the apps perhaps have similar functionality but different requirements for what data is shown.

### Example of using templates with placeholders

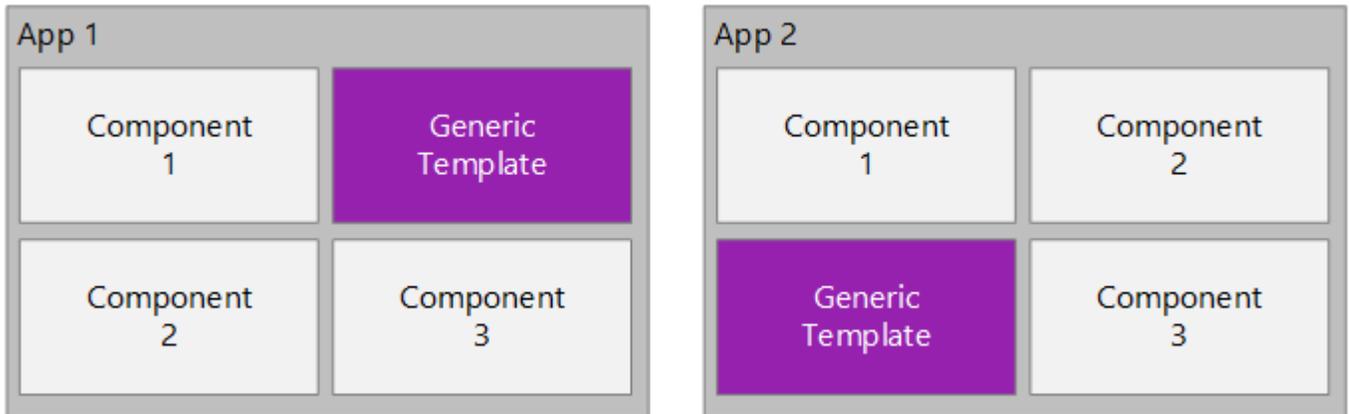
In the following example, a series of apps share many of the same business requirements but one of the required components varies slightly. This difference is managed by using a generic template for the common features with a placeholder for the separate templates that define the custom features. You select which template to use in the placeholder when you configure the app. This use of templates allows you to build just one version of each component, which will simplify future maintenance.

The rest of this topic gives more a more detailed explanation for why you might need to use templates with placeholders.

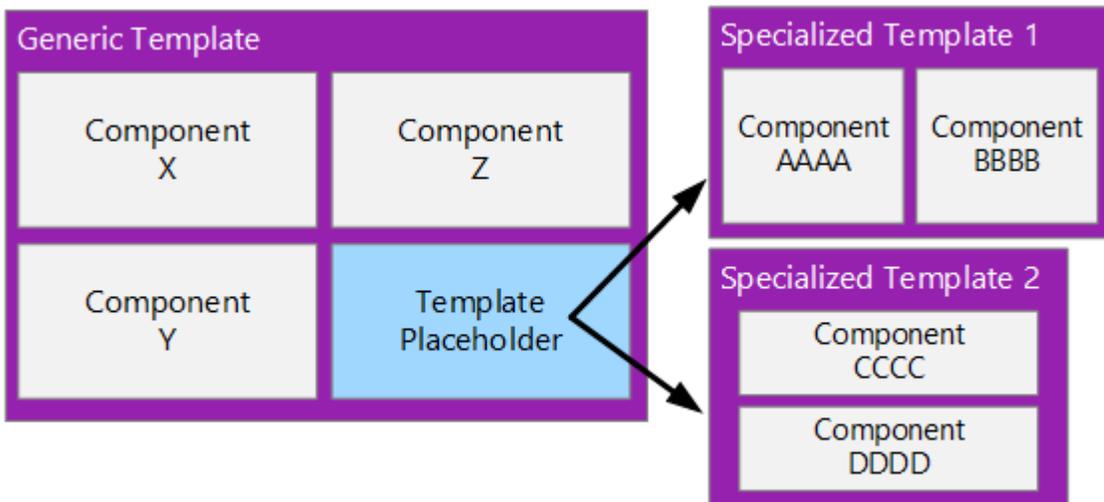
See also [How to configure a template with placeholders](#).

Example

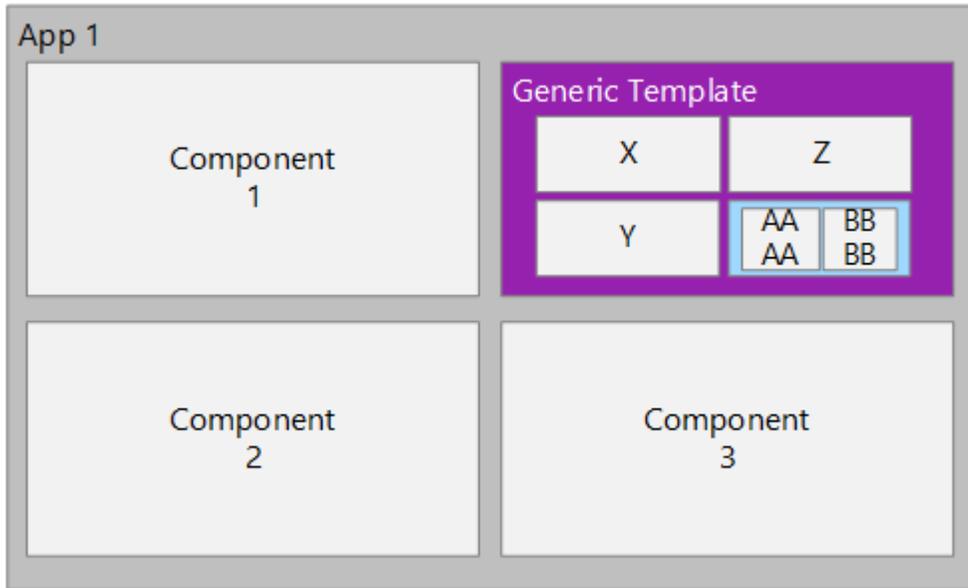
In this example there are two apps that require a component that has some features that are common to all the apps and some features that are different in each app:



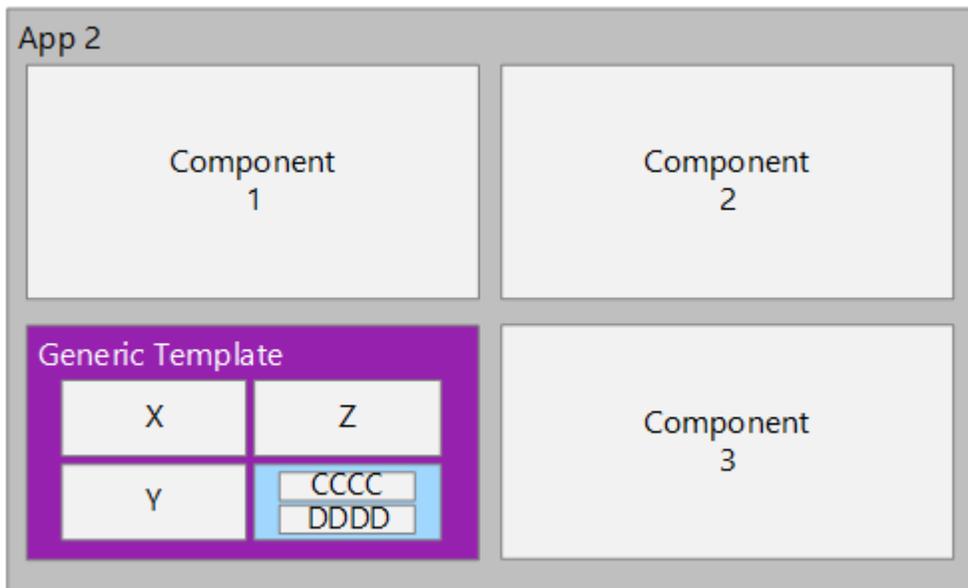
The Generic Template has components for the common features and a placeholder for the component that will be different in each app. The specialized components are defined in separate templates:



When App 1 is configured, the placeholder in the Generic Template is set to Specialized Template 1:



When App 2 is configured, the placeholder in the Generic Template is set to Specialized Template 2:

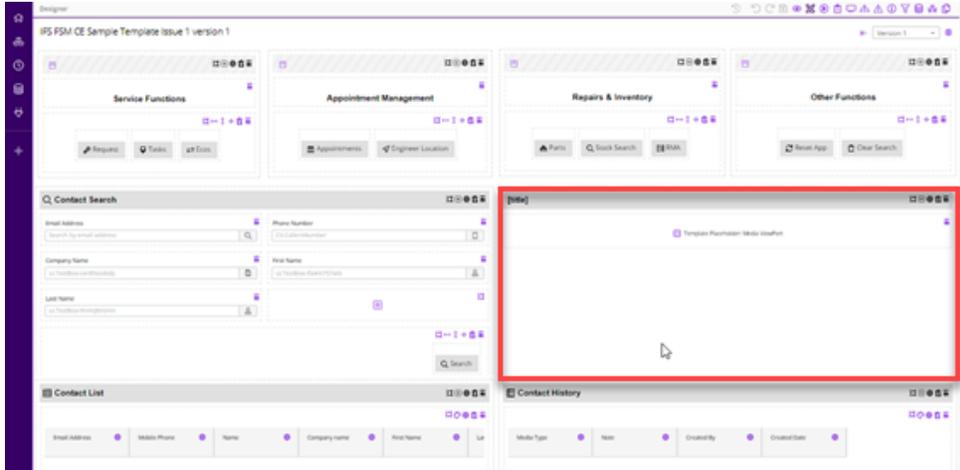


**Note** It is possible for a nested template to itself have a placeholder for another template. So in the above example, Specialized Template 1 and 2 could themselves have template placeholders.

See also [How to configure a template with placeholders.](#)

## Template placeholder example for FSM

There is a sample FSM template that demonstrates the use of placeholders. When a Media app is created from the following template, you can add one of the system media templates (Voice call, Email or Chat) to the placeholder indicated in red:



You can download this template from the Central Template Repository.

## How to configure a template with placeholders

This explains how to configure a template with two placeholders. For an explanation of why you might need to use templates with placeholders, see [Using templates with placeholders](#).

The following figure shows how an app looks when configured with the example template:

*Previewing the app with the example template*

The example app is configured from three templates:

### How the example template is configured

You create a template in the usual way and then add a **Template Placeholder** for each placeholder you need in the template:

There are two placeholders in this example:

*Template created with two placeholders*

You can preview the template but there's nothing to see. You select the template to display in each placeholder when you configure the app.

In the template's App Settings, you can also set the *style* of the template and add *configurable options*.

### Configuring the example app

You create a standard app in the usual way and add the template with placeholders to it—on the Designer page it looks like any other template:

#### *Example template after adding it to an app*

However, when you add a template that contains placeholders, you are prompted to select a template for each of the placeholders—you can select any template:

#### *Selecting a template for each placeholder*

## Standard CE templates for media apps

---

**Note** Not applicable to self-service only tenants.

IFS Customer Engagement provides standard templates for commonly-used components in media apps, such as voice, email, social media, and chat, as well as sample templates for integration purposes.

**Important** You will not automatically upgrade to new versions of providers or templates. You must select the version(s) yourself. In new tenants, the standard CE templates default to the latest provider version.

The following standard templates are available in this release. For details of the providers required for these templates. see the release notes.

Name	Version	Description
IFS CE Callback Template	3	Provides a simple component for callbacks or ringbacks. The service agent can use it to place the call and reschedule the call for later if the caller doesn't answer. Requires the Call Backs workflow definition that can be downloaded from the IFS Knowledge Base.
IFS CE Chat Template	2	Where clients have a chat client, embedded in a web page or mobile application, you can add a ready-made chat component to any CE Studio app. This lets agents reply to callers, end the chat with the caller and end the Wrap Up phase of the activation freeing the agent to accept another activation. A caller can also be transferred to an agent with a different skill set.
IFS CE Email Template	5	<p>This template provides a ready-built email component for any CE Studio app. This lets service agents reply, forward or close incoming emails, see the conversation history, and create and send outbound emails with attachments.</p> <ul style="list-style-type: none"> <li>• Version 5: full conversation history</li> <li>• Version 4: embedded attachments (max. 25)</li> <li>• Version 3: cc and bcc</li> </ul>
IFS CE Inbound Voice Template	2	Provides a simple component for inbound calls, with buttons to hang up the call and wrap up the activation.
IFS CE Outbound Voice Template	1	Provides a simple component for use when making outbound calls, with buttons to hang up the call and wrap up the activation.

Name	Version	Description
IFS CE Social Media Template	4	<p>Provides a ready-built component for each social media type. The supported types are grouped in a tab group. The correct tab is automatically displayed when the social media activation is presented to an agent in Agent Desktop. The tab group is hidden when viewed in Agent Desktop.</p> <p><b>Note</b> Use version 4 for SMS. Use version 3 for WhatsApp.</p>
IFS CE Progressive and Predictive Dialing Template	1	<p>Provides a simple component for making outbound campaign calls. Requires the Campaigns workflow definition that can be downloaded from the Admin Portal help.</p>

**Note** You can modify and extend any of the standard templates by cloning a template as a new template, and then modifying the new template as required.

### Using the IFS CE system templates

IFS CE system templates are ready to use. You do not need to make them live. However, if you need to modify the template then you first need to make a copy of the template:

1. On the Apps page, locate the template and then click **Manage Versions**.
2. Click **Designer**. If you see a warning that the template is live, click **OK** to continue.
3. On the Designer toolbar, click .
4. Select the **Create as new app** option and then click **Continue**.

The template opens automatically so you can start editing it. You can rename the template on the Home page. Click the name of the template to edit it.

# 5

## Data providers and API services

CE has a provider for each data source that it supports. A provider stores the metadata types and methods of the data source. To view the methods, properties and relationships for a data source, use the [Metadata Viewer](#).

Credentials needed to access data at the provider endpoint are created separately.

**Note** There are some sample templates available for download from the Central Template Repository that demonstrate how you can use the different providers in CE Studio apps.

You can access the Swagger documentation for the Studio Services API. The URL of the index page for your tenant is: `https://<environment>/<tenant-path>/ce/0/studioapi/swagger/index.html`.

**Note** On some environments the tenant name (such as Dev Preview) and tenant path (devpreview) are different. The index page URL uses the tenant path.

### Provider types

---

The following provider types are available to all types of tenant.

**Note** CE providers do not currently support webhook requests.

Provider	Type	Description
IFS Applications/Cloud provider for both IFS Cloud and IFS Applications 10		<p>For details, see <a href="#">IFS Applications/Cloud provider</a>.</p> <p><b>Note</b> Minimum supported versions are IFS Applications 10 Update 12, and IFS Cloud 21R1.</p>
IFS CE oData provider		<p>For any oData version 4 service.</p> <p><b>Note</b> The web service must adhere to the standards specified in the version 4 protocol for CE to be able to consume it. Depending on the web service, bearer tokens may expire and it is your responsibility to manage this.</p>
IFS CE OpenAPI provider		<p>For any OpenAPI service, version 2 or 3.</p> <p>The OpenAPI service fully supports any endpoint that complies with the OpenAPI 2 or 3 specification. See <a href="#">OpenAPI providers</a> for details.</p> <p><b>Note</b> The web service must adhere to the standards specified in the version 2 or 3 protocol for CE to be able to consume it. Depending on the web service, bearer tokens may expire and it is your responsibility to manage this.</p>
IFS CE System provider	System	<p>Provides generic functions such as access to translation providers, appdata, support for authentication, portal apps, sending SMS. See <a href="#">System provider</a> for details.</p>

You need to create providers for third-parties, such as IFS Applications/Cloud provider and oData, as these depend on the deployment.

## Additional provider types

The following providers are also available - the media providers are not available to self-service tenants.

**Note** CE providers do not currently support webhook requests.

Provider	Type	Description
IFS Field Service Management provider		<p>Lets clients integrate IFS Customer Engagement with an IFS Field Service Management 6+ deployment. The provider retrieves all the properties and methods exposed by that version of the FSM service. For further details, see <a href="#">Field Service Management provider</a></p> <p><b>Important</b> In version 6.0.0, an FSM provider worked even when there was no integration user and password configured. This issue is now fixed so the FSM deployment must have either have SSO enabled or an integration user configured.</p> <p><b>Note</b> To use FSM's Metrix Perform Methods, use the IFS CE oData provider. For details, see <a href="#">Metrix Perform Methods (unbound methods in FSM)</a>.</p> <p><b>Note</b> You can access IFS Planning and Scheduling Optimization data through the IFS Field Service Management provider.</p>
IFS CE Last-Mile Technician Portals Provider	System	Use when configuring public portal apps for Last-Mile Customer Portal.

Provider	Type	Description
IFS CE Reports provider	System	Provides access to the <i>reports</i> designed and scheduled in the Admin Portal.
IFS CE Survey provider	System	Use when configuring public portal, portal and media apps for surveys.
IFS CE Agent Desktop provider	Media	Provides access to <i>Agent Desktop toolbar, events and agent state</i> . Provides functions for integrating CE Studio apps with the Agent Desktop toolbar. For example, the tools for accepting, closing and wrapping up activations. <ul style="list-style-type: none"> <li>• In version 6, you can select call states when configuring rules.</li> <li>• In version 5, you can select agent state names from a list rather than enter the state number.</li> <li>• Version 4 supports WhatsApp.</li> <li>• Version 3 supports only Facebook and Twitter.</li> </ul>
IFS CE Chat provider	Media	Provider for <i>chat</i>
IFS CE Email provider	Media	Provider for <i>email</i> . Use this when designing apps or templates for receiving, sending and retrieving emails in media apps. The provider is a layer on top of the tenant's email service and handles backend functions such as sending email. <ul style="list-style-type: none"> <li>• Version 4 supports embedded images.</li> <li>• Version 3 supports cc and bcc.</li> </ul>
IFS CE Real Time Statistics provider	Media	Provides <i>real-time statistics on the contact center</i> . For example, gives access to real-time data on the Agent Desktop queue. Used for wallboards.
IFS CE Social Media provider	Media	Give service agents the ability to manage public Facebook and Twitter messages, Facebook and Twitter direct messages, WhatsApp, SMS.

Provider	Type	Description
IFS CE Workflow provider	Media	Provider for <i>custom workflows</i> configured in the Admin Portal. For example, the ability to manage work objects created from work definitions, such as campaign dialing, and call backs or ring backs.

You need to create providers for third-parties, such as FSM, IFS Applications/Cloud provider and oData, as these depend on the deployment.

**Note** You do not need a provider for call data (fully-featured tenants only) or user data. This information is always available to CE Studio media apps. You add these properties as described in *Accessing call data*.

## Creating a new provider

**Note** This describes how to create a provider for an external data source, such as IFS Field Service Management or IFS Applications/IFS Cloud. Only IFS technical users can create the system providers.

There are two ways of creating providers. You can either create the providers and then add them to the CE Studio apps you create, or you can import an existing CE Studio app and automatically create the providers during the import.

Once you have created the providers, there are several ways of accessing the provider versions. You can access:

- The Active (live) version of the provider.
- A numbered version (version 1), but note that the version might not be live.
- The Latest version of the provider. This is the latest successfully created version, but it might not be live.

Select the Active version or Latest version if you want an automatic upgrade policy for your app or template, or the numbered version for manual upgrades.

### Create providers to add to CE Studio apps

To create a new provider which you will then add to a CE Studio app:

1. Go to **Home > Providers**.
2. Click **Create Provider**.

3. On the **Provider Details** tab:

Field	Description
Provider Name	Enter a descriptive name that will make it easier to identify the provider when configuring CE Studio apps and templates. You can edit this later.
Service type	Select the service type. The <b>IFS Applications</b> type covers both IFS Applications 10 and IFS Cloud.
Time Zone	<p>By default, data containing datetimes is saved as UTC, and returned without any offset for time zones.</p> <p>If data is saved by the data source (such as Field Service Management, IFS Cloud) with a time zone offset then you need to specify this offset when creating the provider. The default time zone for all providers is UTC.</p> <p>When the timezone offset is calculated CE Studio takes into account both the provider's time zone and the user's time zone:</p> <ul style="list-style-type: none"> <li>• For public portal apps, it takes the user's time zone from the browser.</li> <li>• For all other types of app, it adjusts the display of times by allowing for the time zones from the CE user and also from the browser .</li> </ul>
Credentials Object	<p>Select the credentials for the target environment. This contains both the endpoint URL and the credentials needed to connect to that environment.</p> <p>You need to create credentials first:</p> <ul style="list-style-type: none"> <li>• For IFS Customer Engagement, go to the Admin Portal and the <b>Credentials Manager &gt; Manage Credentials</b> page.</li> </ul> <p><b>Note</b> You cannot create the provider if there is an issue with the endpoint or a network problem.</p>

Field	Description
Service	<p>The service name without the URL of the endpoint.</p> <ul style="list-style-type: none"> <li>For IFS Cloud, <code>MainCaseHandling.svc</code> is an example of a service name. You can search for a service by typing the 3 or more characters from the name.</li> <li>For an endpoint where there is no service name, enter a forward slash <code>/</code>.</li> <li>IFS Customer Engagement only: for IFS Field Service Management, <code>ProcessXML</code> is an example of a service name.</li> </ul>
Description	Additional information about the provider. This is an optional field, and you can edit this later.
Concurrency Checks	Use this where the deployment does not enforce concurrency on operations such as update, upsert and patch. When selected, the provider will attempt to handle concurrency for these types of operation.

4. Click **Continue**. The button is unavailable if any mandatory information is missing.

**Note** You cannot create the provider if there is an issue with the endpoint or a network problem. In this situation, the status of the provider will show as **!**. Try again later to create the provider.

5. Version 1 is created. Make the provider version live. You can only use a provider which has a live version.

## Creating a new provider version

You must create a new provider version after changing the credentials for accessing the provider.

You may also need to create a new version of a provider when a product modifies its API or schema.

To create a new version of a provider:

1. In CE Studio, go to **Home > Providers**.
2. For the provider you want to update, click **Manage Versions**.
3. Click **Create New Version**.
4. Depending on the provider type, enter the endpoint URL. This should be the same endpoint or closely related to it, for example, a different version of the endpoint.

5. Click **Create**. It may take a few minutes to create the new version.
6. If required, click **Live** to make this the active version of the provider. This means that any apps and templates that are configured to use the active version of the provider will immediately update to use this version.

**Note** If you want to prevent any future app or templates from using an earlier version of a provider, select **Decommissioned**. You cannot decommission a version if it is in use. *Decommissioning is permanent — you cannot undo this.*

### Changing an app to use a different version

**Note** This only applies to apps and templates that do *not* use the active version of the provider.

1. In CE Studio, go to **Home > Apps**.
2. For the app or template that you want to update, click **Designer**.
3. Click **Manage Providers** .
4. Select the provider version from the **Version** list.

### Locking an app to a numbered version

You can lock an app or template to a specific version of a provider so that they do not automatically upgrade to the next provider version. To do this, select a numbered version rather than the live version when you add the provider to the app or template.

## Using the Metadata Viewer

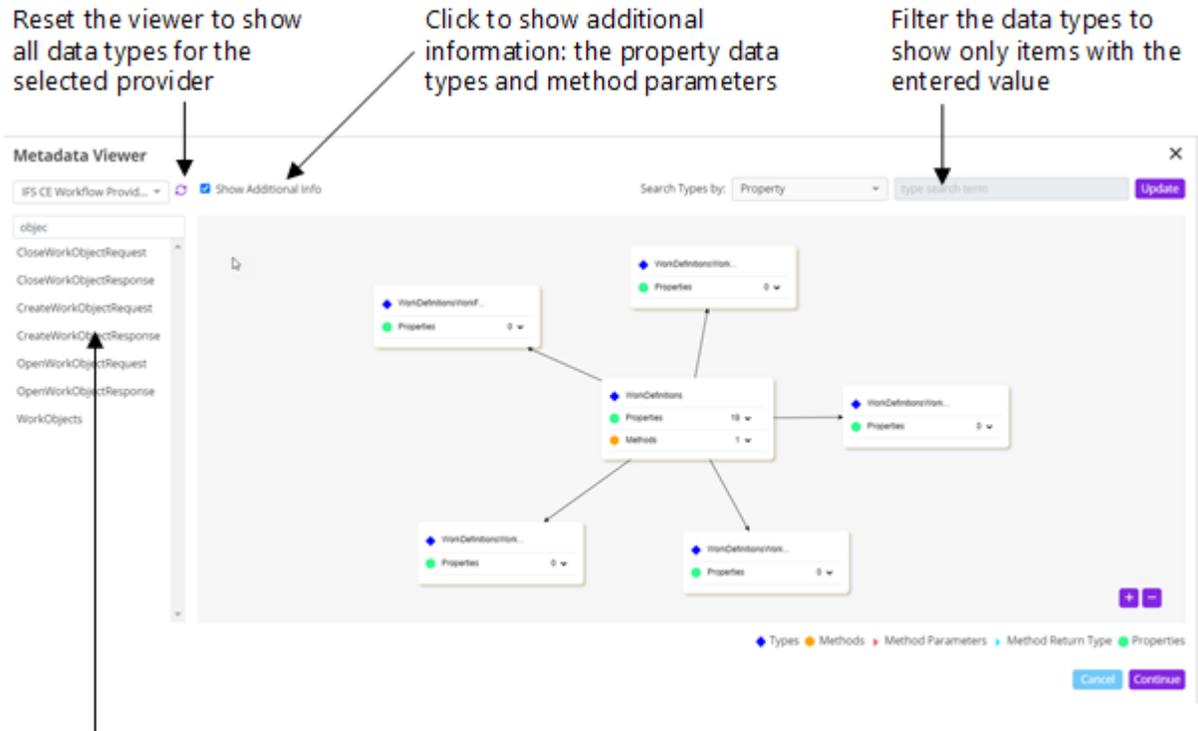
---

You can inspect the properties, methods and relationships exposed by any provider in the Metadata Viewer and then add and remove properties from components and actions using the Metadata Selector.

## View the metadata for the provider

Once you have added a provider to your app or template, you can inspect the metadata for that provider, or any other provider in the app or template, by clicking  **Metadata Viewer** on the toolbar.

### Features of the Metadata



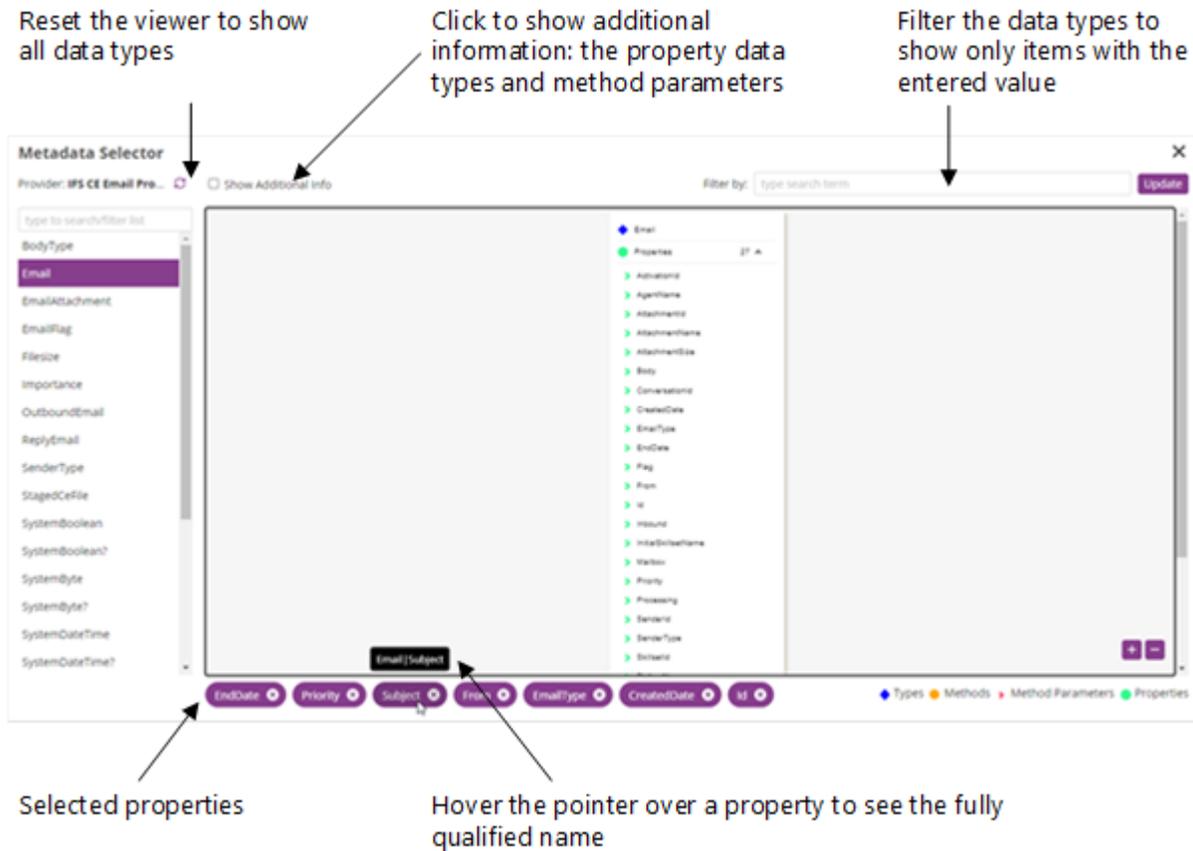
**Viewer** Filter the list of data types and then click to view the details

You can search for a data type, property or method in the Metadata Viewer using the **Filter** box. The data types in the main area of the viewer are then filtered along with its associated parent information. For example, searching for a property called description will filter the main view to show just the data types that contain a description property.

To reset the Metadata Viewer to show all data types, click  **Grid View** in the top left.

## Using the Metadata Selector to add properties to components

You can open the Metadata Selector and use it to find the properties you need to add to a component, such as a form or data grid.

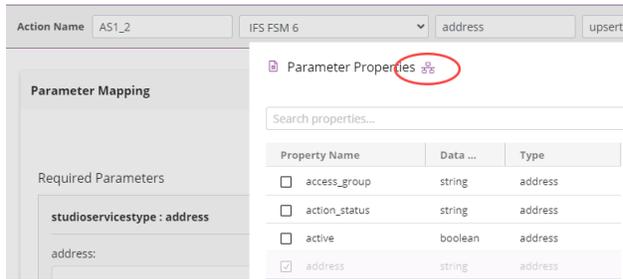


### Features of the Metadata Selector

1. In a component, such as a form or data grid, click and select the provider.
2. Click to open the Metadata Selector.
3. Select the data type and expand the list of properties. For details of the data type of each property, click the **Show Additional Info** check box.
4. Click each property that you want to add to the component. As you do so the selected properties are listed at the bottom of the dialog.
5. Click **Continue** to add the selected properties to the component. Any required properties are automatically added.

### Using the viewer when adding actions

You can also use the Metadata Selector when adding optional properties to the action that you are configuring.



*Opening the Metadata Selector in Manage Properties*

## IFS Applications/Cloud provider

**Important** For a successful integration, you require a knowledge of the target system and its metadata. For details of the entity types, properties, and methods, refer to the IFS Cloud documentation.

The IFS Applications/Cloud provider stores the data types (OData entity types), properties and OData methods for integrating with both IFS Applications 10/IFS Cloud and IFS Cloud deployments. The translation service must be deployed in the environment – you will not be able to create a provider without this.

### Authentication for providers in IFS Cloud

Providers for an IFS Cloud deployment can either use Azure Active Directory single sign-on (SSO) or an integration user account for authentication.

- For SSO, all users in Agent Desktop and CE Studio making requests through the provider are authenticated using their existing SSO credentials. All users must have a user account in the client's Azure AD deployment. Any backend services that send requests to the target system use a default integration username and password. The default integration user needs to be set up in the IFS Cloud deployment.

**Note** The tenant requires an SSO deployment to be configured for it. You can request this from IFS Support using the appropriate ServiceNow form.

**Note** In IFS Customer Engagement, to view the SSO deployment or to configure a user group for the SSO deployment, go the Admin Portal and the **Studio > SSO Deployment** page. If a user group is associated with the SSO deployment then only users belonging to the user group must sign on with their SSO credentials.

- If the tenant is not configured for Azure Active Directory single sign-on then all users making requests through the provider are authenticated using the default integration username and password as configured in the IFS Cloud deployment. If the deployment supports multiple locales, then each locale can have a separate integration user configured. The locale that is used is taken from the locale of the signed in user.

All versions of a provider will use the same username/password or SSO credentials.

### Creating IFS Applications providers

For both IFS Cloud and IFS Applications 10, you need to create a provider for each projection that you want to use.

**Note** You cannot create a provider if the translation service is not deployed in the environment.

You can obtain the URI of the endpoint by using the API Explorer or the Developer Tools:

1. Start by obtaining the URI of the projection:
  - a. Go to the IFS Applications 10 or IFS Cloud deployment.
  - b. Identify the entity you are interested in, such as Work Orders.
  - c. Find an existing work order.
  - d. Press **F12** to show the developer tools pane.
  - e. On the Preview tab, examine one of the requests to find the uri of the endpoint.
  - f. Copy the uri starting `https:` and ending with `.svc`.
2. Create the provider as described in [Creating a new provider](#).

### Concurrency checks, etags and IFS Applications/Cloud providers

Depending on the implementation, you may need to select the **Concurrency Checks** option for the IFS Applications/Cloud provider. This is necessary, for example, if there are any database constraints implemented in IFS Cloud.

**Note** The error `Error with external service: Reason Phrase = Precondition Required` is shown if concurrency checks are enabled in IFS Cloud but not in the provider.

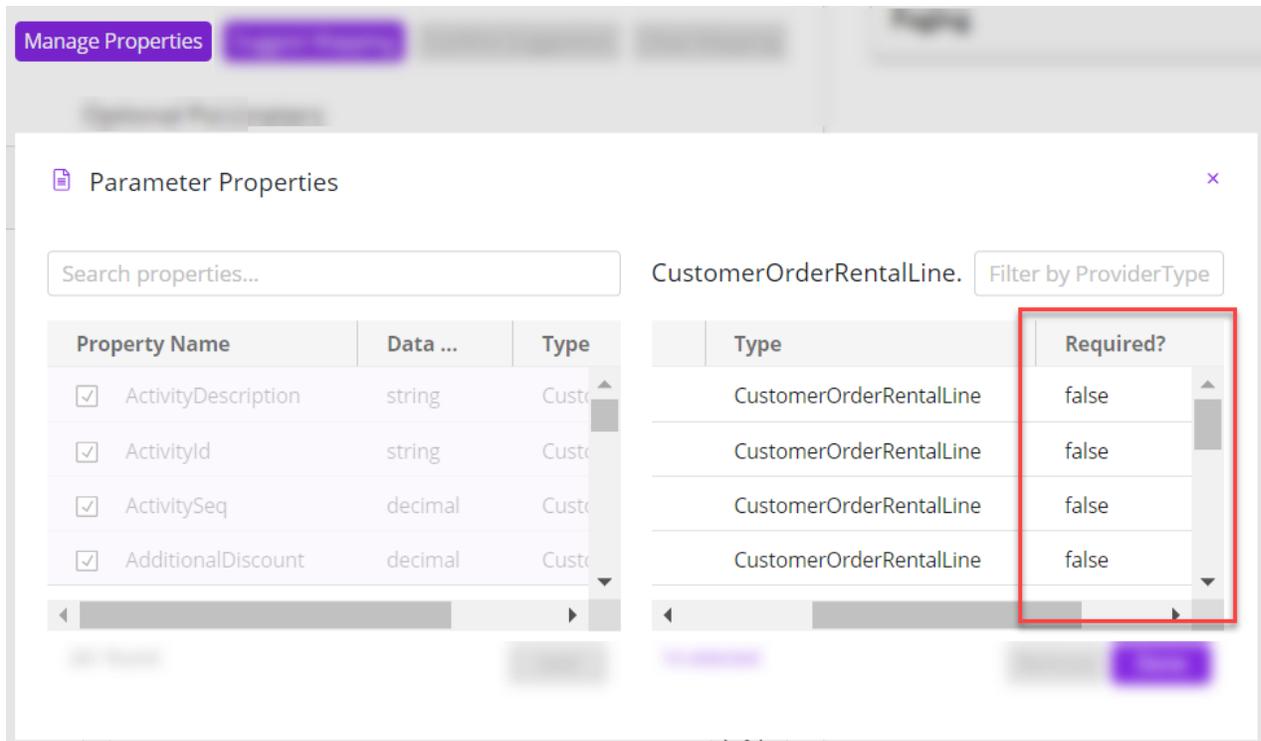
**Note** The IFS etag is automatically included in provider responses when data is read from any odata provider and, when Concurrency Checks is enabled, is included in update requests to the provider.

### IFS Applications provider data types and properties

The provider stores all the entity types and properties exposed by the projection used when you created the provider. You can add any of these entity types and properties to CE Studio apps. Once you have created a CE Studio app, you can explore the provider metadata using the [Using the Metadata Viewer](#).

When configuring actions, there are two parameter types, suggested and optional:

Suggested parameters	IFS Cloud or oData properties that are <i>not nullable</i> are defined by CE Studio as suggested parameters. For these properties the Parameter Properties dialog shows <b>Required?</b> as true. These parameters are required for the CE Studio action to succeed.
Optional parameters	IFS Cloud or oData properties that are <i>nullable</i> are defined as optional parameters. For these properties the Parameter Properties dialog shows <b>Required?</b> as false. These parameters are not necessarily required for the CE Studio action to succeed, however at the endpoint there may be some underlying relationship between properties which is not known to the CE Studio provider. You therefore need to check the oData or IFS Cloud metadata to verify this.



*Nullable and non-nullable properties*

### Working with IFS Applications provider methods

IFS Applications 10/IFS Cloud and IFS Cloud provider methods are available when creating actions in CE Studio.

To create, update or delete an entity instance, you must include all the key properties required by the target system. For updates and deletes, this includes the key(s) to uniquely identify the instance of the entity. Note that the properties you see when configuring actions are studioservicetype properties and whether they are required will depend on the context:

- To add properties to the action that you are configuring, click **Manage Properties** and select the required and optional properties needed by the method. Then map each property to the fields in the CE Studioapp that hold the new values.
- To exclude unwanted properties, select the **Ignore** check box.
- To pass null for any property, select the **Use Null** check box.

For details of how to add actions, see [Creating an action and mapping the parameters](#).

### Using lookup codes

When testing a CE Studio app that uses a lookup code from an IFS Applications provider be aware that it can take a long time for the values to load. Lookup will be much quicker once the values are cached.

## Limitations

In CE Studio, an action can sort the data in the response. The property to sort by can be configured in the action. If it is not configured then sorting is on the first column in the data grid. However, IFS Cloud and IFS Applications 10 do not allow sorting on fields that are the CLOB data type. CE Studio typically handles these properties as very long strings.

This means that when you configure an action for either IFS Cloud or IFS Applications 10, then sorting by one of these properties results in the error: "ORA-00932: inconsistent datatypes: expected - got CLOB". If this occurs then you should change the property that is used for sorting.

## See also

[Understanding IFS Cloud provider method names](#)

[Locating projections in IFS Cloud](#) which explains how to find the name of the projection you need in order to create a provider.

[Accessing IFS Cloud unbound methods](#) when creating actions.

[Working with child provider types in IFS Applications providers](#)

## Locating projections in IFS Cloud

This topic explains how to find the information you need in IFS Cloud for creating IFS Cloud providers, specifically:

- The base URL for the credential
- The name of the service (the IFS Cloud projection)
- The full provider URL if you want to view the provider metadata as XML

Start by accessing the required version of IFS Cloud and navigate to the page from which you want to retrieve the projection.

### Base URL

For the credentials that you will create, you need the base URL from the address bar, that is the first part of the URL terminating with:

```
/main/ifsapplications
```

For example `https://ifs23r1.ifsdemo.com/main/ifsapplications`

### Projection name

You can:

- Press **F12** on your keyboard to open the development tool bar
- Click **page info**.

**Note** For convenience, it's easier to understand and retrieve the projection through the debug log.

For older versions of IFS Cloud, select the debug log window from the quick navigator.

**For new IFS Cloud versions:** On the page info window, locate the projection name as shown below, and add `.svc` to the name:

## Page info

### Main page:

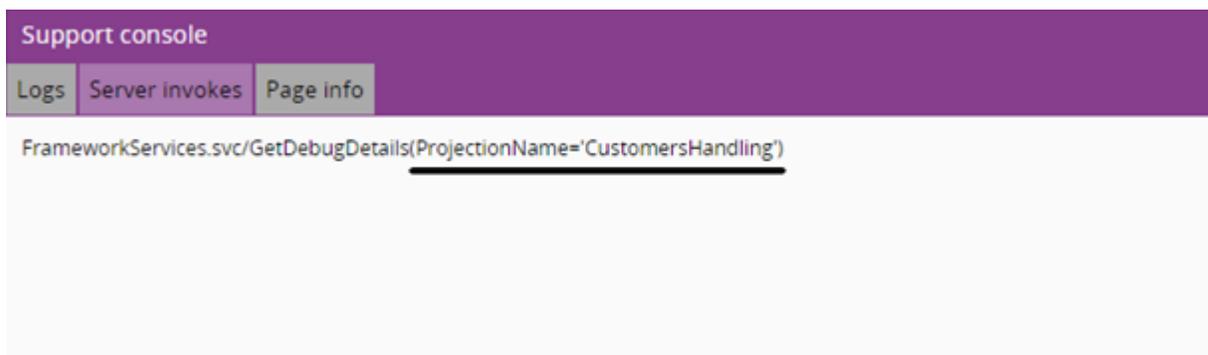
Entity (LU Name): CustomerInfo  
Projection: CustomersHandling  
 Projection Entityset: CustomerInfoSet  
 Projection Entity: CustomerInfo  
 View: CUSTOMER\_INFO  
 Component: ENTERP

### Page elements

Entity (LU name): CustomerInfo  
 Element name: EnterpCustomersList  
 Projection Entityset:  
 Projection Entity: CustomerInfo  
 View: CUSTOMER\_INFO

### *Using the Page Info window*

**For older IFS Cloud versions:** In the debug log window, switch to the **Server Invokes** tab. Obtain the projection name as shown below, and add `.svc` to the name:



### *Using the Server Invokes tab*

## Constructing the provider URL

To construct the full provider URL, take the base URL + /projection/v1/ + projection-name.svc

Using the above example:

```
https://ifs23r1.ifsdemo.com/main/ifsapplications/projection/v1/customerhandling.svc
```

## Viewing the metadata as XML

You can use the full provider URL to view the metadata as an XML file:

1. Paste the provider URL into a browser tab followed by /\$metadata.

For example:

```
https://ifs23r1.ifsdemo.com/main/ifsapplications/projection/v1/customerhandling.svc/$metadata
```

2. Enter the credentials used when creating the provider.

You will then see the metadata as XML.

## Understanding IFS Cloud provider method names

The metadata that is used by CE Studio when generating IFS Applications/Cloud or oData providers determines the method names of each provider type, including the child types.

You can view the metadata for any oData 4 type provider like this:

1. Paste the provider URL into a browser tab followed by /\$metadata.

For example, `https://<tenant>/main/ifsapplications/projection/v1/BusinessLeadHandling.svc/$metadata`

2. Enter the credentials used when creating the provider.

You will then see the metadata as an XML file.

Each oData 4 type is an entity set. An entity set may have:

- One or more navigation properties - these are the relationship or child types
- Get, patch, create and delete methods - the syntax is `get@<entity set name>`, `create@<entity set name>` and so on
- Additional methods for each relationship - syntax is `get@<entity set name>@<relationship>`
- Other methods, such as DefaultCopy, dissociate, associate and so on which has been referenced as actions in the metadata

Consider these examples from the IFS Applications provider BusinessLeadHandling.

For the `AddressCity` provider type, this is the metadata:

```
<EntitySet Name="Reference_AddressCity"
  EntityType="IfsApp.BusinessLeadHandling.AddressCity"/>
```

The entity set name is `Reference_AddressCity` and the entity set type is `IfsApp.BusinessLeadHandling.AddressCity`

This entity set has no relationships so it only has these four methods:

- `delete@Reference_AddressCity`
- `patch@Reference_AddressCity`
- `create@Reference_AddressCity`
- `get@Reference_AddressCity`

`BusinessActivity` is a slightly more complex provider type - it has one relationship. This is its metadata:

```
<EntitySet Name="Reference_BusinessActivityRef"
  EntityType="IfsApp.BusinessLeadHandling.BusinessActivityRef">
  <NavigationPropertyBinding Path="ActivityRef"
    Target="Reference_BusinessActivity"/>
</EntitySet>
```

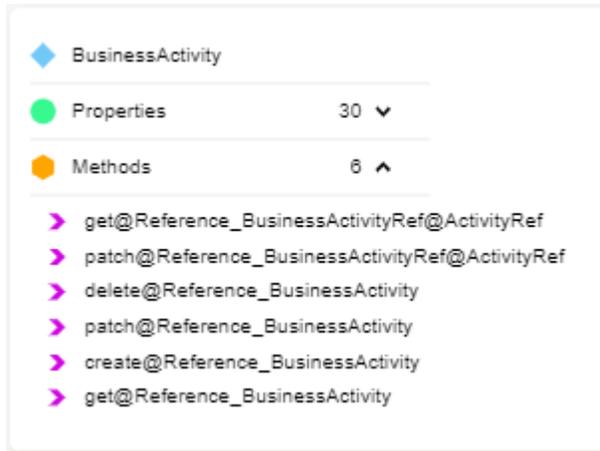
The relationship is referenced by the path name "ActivityRef".

This entity set therefore has the basic four methods as well as get and patch methods for the relationship:

- `delete@Reference_BusinessActivityRef`
- `patch@Reference_BusinessActivityRef`
- `create@Reference_BusinessActivityRef`
- `get@Reference_BusinessActivityRef`
- `get@Reference_BusinessActivityRef@ActivityRef`
- `patch@Reference_BusinessActivityRef@ActivityRef`

The `get@Reference_BusinessActivityRef@ActivityRef` method, for example, gets the `ActivityRef` children of each `BusinessActivity`.

This is how the metadata for `BusinessActivity` looks in the Metadata Viewer:



## Accessing IFS Cloud unbound methods

Some IFS Cloud providers have methods that are not bound to any of the entity types in the provider. These are referred to as unbound methods.

To access the unbound methods:

1. In CE Studio Designer, create an action and select `UnboundMethodClient` in the provider type filter.
2. In the Method field, you can now view all available unbound methods in the provider.

## Working with child provider types in IFS Applications providers

The Metadata Viewer shows all the provider types in the provider and their relationships.

Typically, there are several main or parent types with relationships to one or more child types. The properties of the child types in parent types are referred to as navigation properties in CE Studio Designer.

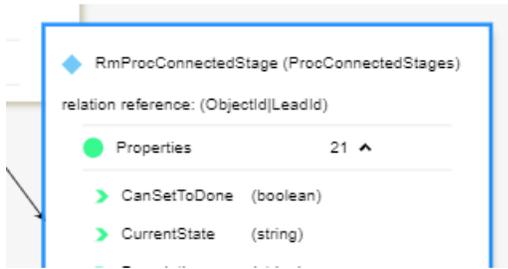
When you include navigation properties in data grids and forms, the actions you configure for the data grid or form automatically return all the associated navigation properties. However, to invoke patch (update), delete, Inbound methods (Inbound actions, functions) at entity level requires an additional read action before invoking them:

1. First, read the entity data to which the method belongs.
2. Show the entity data in a grid or form.
3. When you configure the above actions, use the entity data in the action parameters.

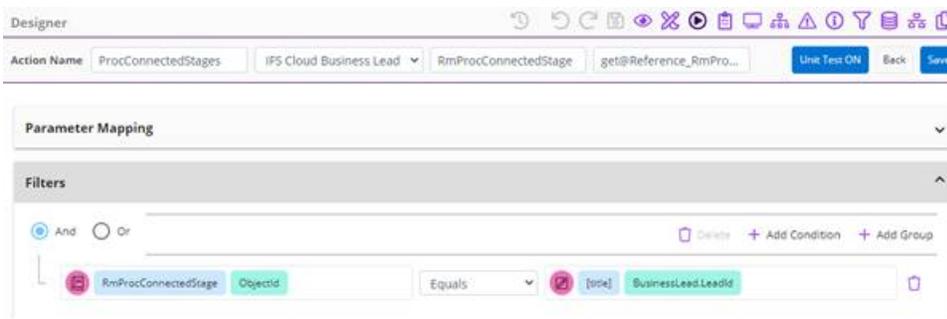
For example, consider the projection: `/ifsapplications/projection/v1/BusinessLeadHandling.svc`

One of the main provider types is `BusinessLead`. The current stage of a `BusinessLead` is defined by the child type `RmProcConnectedStage`. You can view the referential constraint

between `BusinessLead` and `RmProcConnectedStage` in the Metadata Viewer by selecting the **Show Additional Info** option:



The referential constraint between a `BusinessLead` object and its stage is `(ObjectId|LeadId)`. So, in CE Studio Designer, to get the stage details for a `BusinessLead` object, you filter on this relation reference:



`RmProcConnectedStage` has several methods for managing the stage. For example, `RmProcConnectedStage` type has Inbound method at entity level called `SetDone`. If you want to invoke this Inbound method in CE, you should read list of `RmProcConnectedStages` for given business Lead and display the result in grid or form and then you can use these values to configure a `SetDone` method/action to identify the key values of the item.

The `RmProcConnectedStage` key consists of the properties:

- `ObjectId`
- `ProcessId`
- `StageId`

You can guess what the key properties might be from their names but otherwise the key properties are not identified in the Metadata Viewer. To get this information, you need to view the metadata directly:

1. In a browser, enter the provider URL appended with `/$metadata`.
2. Enter the credentials used to create the provider.
3. Search for the child type.

```

<EntityType Name="RmProcConnectedStage">
  <Key>
    <PropertyRef Name="ObjectId"/>
    <PropertyRef Name="ProcessId"/>
    <PropertyRef Name="StageId"/>
  </Key>
  <Property Name="luname" Type="Edm.String" MaxLength="80"/>
  <Property Name="keyref" Type="Edm.String" MaxLength="4000"/>
  <Property Name="Objgrants" Type="Edm.String" MaxLength="2000"/>
  <Property Name="ObjectId" Type="Edm.String" MaxLength="128"/>
  <Property Name="ProcessId" Type="Edm.String" MaxLength="20"/>
  <Property Name="StageId" Type="Edm.String" MaxLength="20"/>

```

You can then configure the action with the required properties.

#### Suggested Parameters

**studioservicetype : RmProcConnectedStage**

---

ObjectId : string

 [title] RmProcConnectedStage.ObjectId

---

ProcessId : string

 [title] RmProcConnectedStage.ProcessId

---

StageId : string

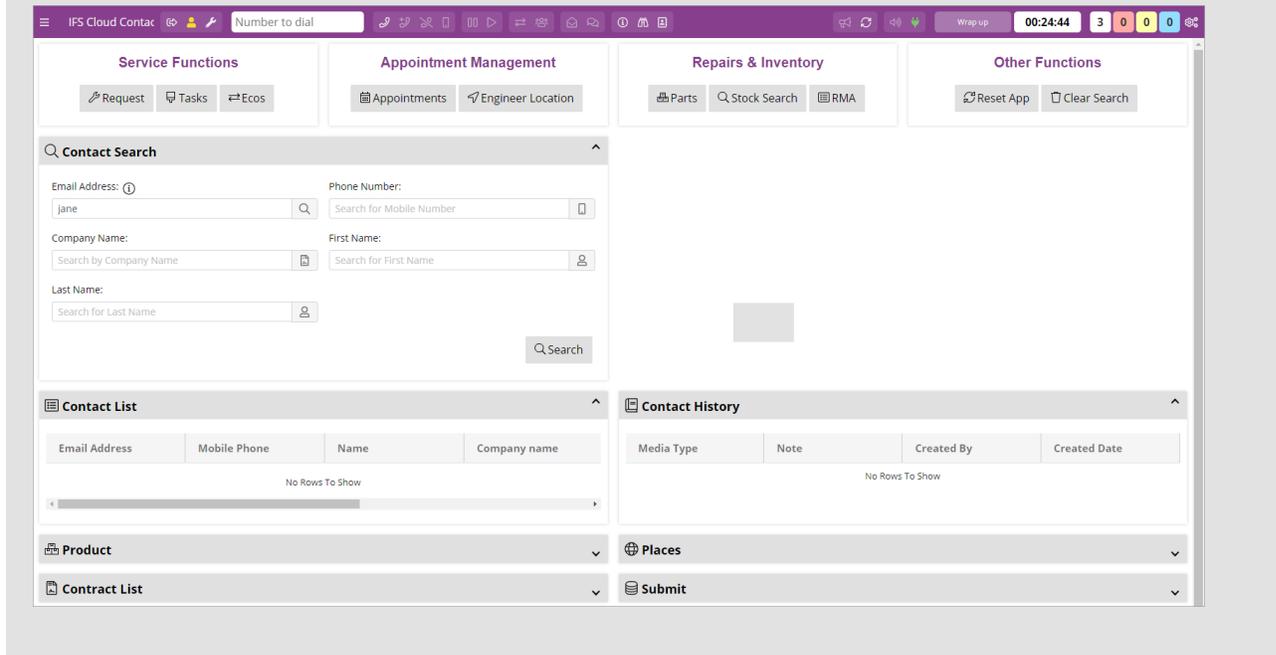
 [title] RmProcConnectedStage.StageId

## Field Service Management provider

**Important** For a successful integration, you require a knowledge of the target system and its metadata. For details of the entity types, properties and methods, refer to the FSM documentation.

The IFS Field Service Management provider stores the data types, properties and value-added, bound methods for integrating with an FSM deployment. Use the IFS CE oData provider for the Metrix Perform Methods; these are the unbound methods that are not associated with any tables.

**Note** You can download an example template for FSM from the Central Template Repository:



## Authentication

A deployment can either use Azure Active Directory single sign-on (SSO) where all users in Agent Desktop and CE Studio sign on using their existing SSO credentials. All users must have a user account in the client's Azure AD deployment. Any backend services that send requests to the target system use the integration user name and password. The integration user account needs to be set up in the FSM deployment.

**Note** The tenant requires an SSO deployment to be configured for it. You can request this from IFS Support using the appropriate ServiceNow form.

**Note** To view the SSO deployment or to configure a user group for the SSO deployment, go the Admin Portal and the **Studio > SSO Deployment** page. If a user group is associated with the SSO deployment then only users belonging to the user group must sign on with their SSO credentials.

If the tenant is not configured for Azure Active Directory single sign-on then all users can sign on using the integration user name and password as configured in the FSM deployment. If the deployment supports multiple locales then each locale can have a default user. The locale of the signed-on user determines which locale is used.

## Creating FSM providers

CE supports the FSM Metrix integration service only. The endpoint URL for all FSM providers take this form:

`https://<FSM-instance-hostname>/MetrixIntegrationService/M5WebService.asmx/ProcessXML`

Where `<FSM-instance-hostname>` is the name of your FSM instance.

**Note** FSM does not enforce concurrency on update and upsert operations — it depends on the FSM deployment whether this is enforced. However, the FSM provider will attempt to handle concurrency for update and upsert operations if you select the **Concurrency Checks** option when creating the provider for your FSM deployment. This will also be applied to nested types.

**Note** If you planning to use FSM's Metrix Perform Methods then you also need to create an IFS CE oData provider. See below for details.

**Note** If you need access to additional tables in FSM then you can raise a support ticket to request that the required tables are added to the FSM whitelist for your tenant. You can request that access is restricted to a specific contact center or release version. Create a new provider version once the tables are added.

See also [Creating a new provider](#).

## Creating a new version for an FSM provider

You may need to create a new version of a provider when the product modifies its API or schema. For example, you need to create a new version after adding a custom table to FSM.

## FSM provider data types and properties

The FSM provider stores all the data types and properties for the FSM API version selected when building the provider. You can add any of these types and properties to CE Studio apps.

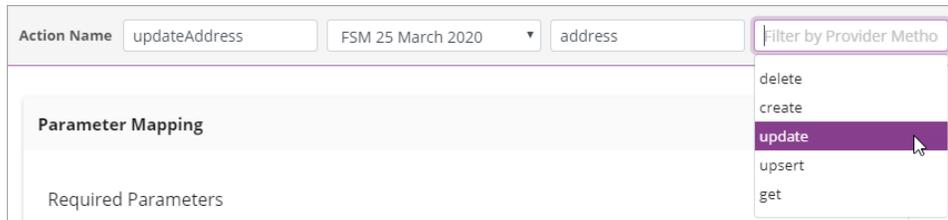
When you select a data type and one or more of its properties, the ID property is automatically selected for you.

**Note** For Booleans, the provider handles any of the following values as *true*: yes, YES, y, Y, 1. It handles these values as *false*: no, NO, n, N, 0, null, empty string.

## FSM provider methods

The FSM provider exposes value-added methods for each property, built on the underlying entities (types) in the data source. For example:

- The IFS Field Service Management provider contains value-added methods (`delete`, `create`, `update`, `upsert` and `get`) that hide the underlying SQL/XML markup queries required by FSM. Value-added methods such as these make it much easier for you to manage the data in the backend.



*Helper methods for IFS Field Service Management data sources*

## FSM provider methods (bound methods)

FSM provider methods are available when creating actions in CE Studio. Methods for FSM types, such as `contact`, `task`, `agreement`, hide the underlying SQL/XML markup queries required by FSM. These are:

### **get**

Gets all records for the selected data type. No required parameters. Add a filter to the action to filter out unwanted records.

### **create**

Creates a new record in FSM. Nested types are upserted.

### **update**

Updates the record in FSM. The required parameters are `studioservicetype` properties. You need to map these to the fields that hold the updated values. You can add also optional properties to the update. Nested types are upserted.

### **upsert**

Creates a new record if a record with the given ID(s) doesn't exist or updates an existing record if it does.

### **delete**

Deletes the record with the given ID(s).

There are also some additional types for:

- Accessing FSM code tables, such as `IFS-CE-StudioServices-Models-CodeModel`
- Saving attachments in FSM: `TemporaryFile`
- Getting attachments: `UploadedFile`

Where there are required parameters for a provider method, there is a **Manage Properties** button.

### Metrix Perform Methods (unbound methods in FSM)

FSM has a large number of methods (Metrix Perform Methods) that are not bound to any tables, and which are not accessible through the IFS Field Service Management provider. To use these methods you need to create an IFS CE oData provider:

1. Create a provider for the oData service type. The provider URL is the URL of the oData service for your FSM deployment. For the basic steps, see [Creating a new provider](#).
2. Typically, you will add both providers to your CE Studio app, that is both the IFS Field Service Management provider and the IFS CE oData provider.
3. To access one of the Metrix Perform Methods:
  - a. Create an action set and action.
  - b. In the action, select your IFS CE oData provider and the `UnboundMethodClient` as the provider type.

All the methods belong to this single provider type.

4. Select the required method from the list.

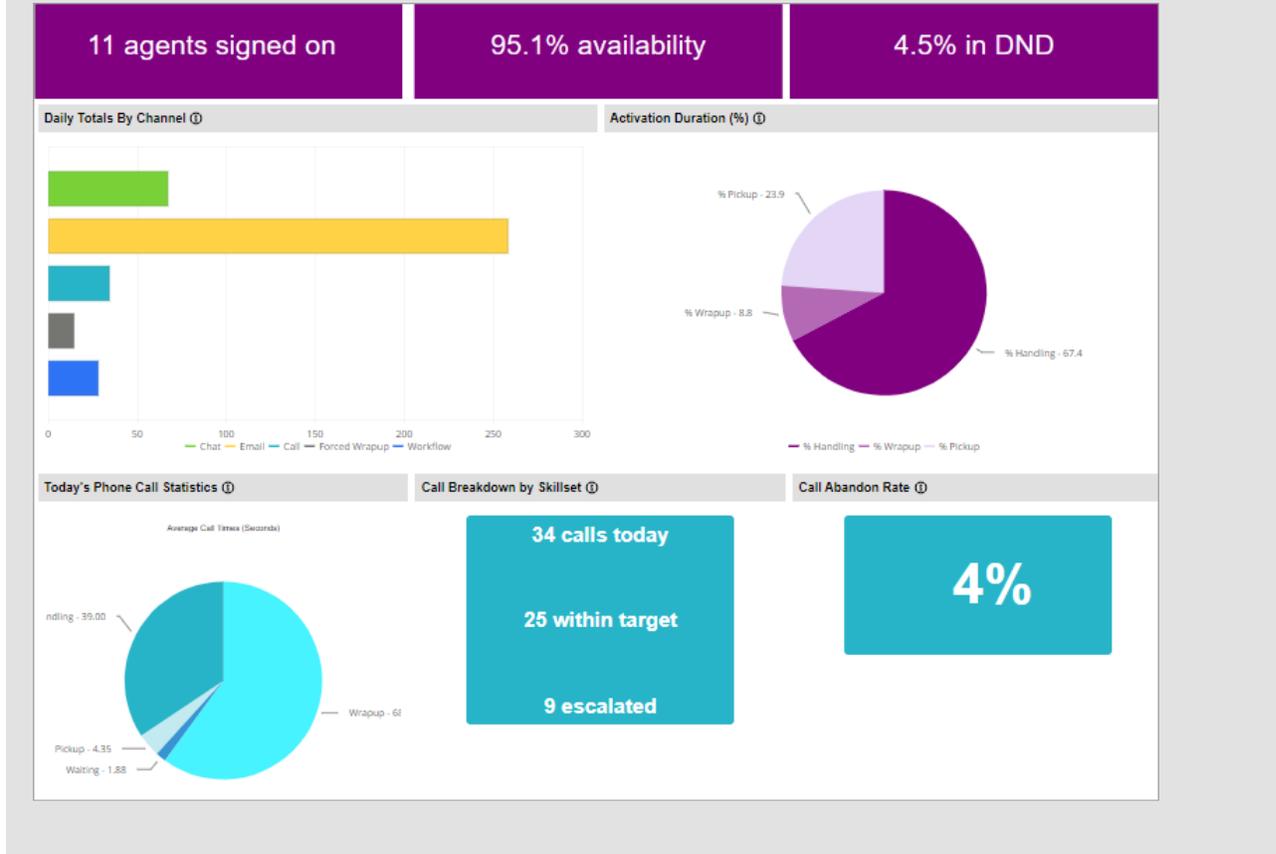
**Note** The methods are slow to load because there are a large number of Metrix Perform Methods.

## Report provider

---

The IFS CE Reports provider gets report data that is configured and generated using the Report Designer, in the Admin Portal, and makes the data available to CE Studio apps. You can then transform the data if required, and configure components, such as data grids, charts, and data elements to show the data.

**Note** You can download an example that demonstrates the use of the IFS CE Reports provider, from the Central Template Repository. All the data in the following example wallboard is provided by the Report provider:



There are some differences between the data provided through the IFS CE Reports provider and other providers:

- The data is readonly so you can't perform operations on it (apart from data transformations configured in action sets).
- You can't enable multi-row select on a data grid that shows report data.
- The action that gets the data can't filter the data or sort it. You need to do this either in Report Designer or by using a data transformation.
- A data grid that shows report data can't be used with events such as `onRowSelected`. (It is not possible to configure the action for the selected row.)

### Creating a report definition

You need a report definition.

1. Create a report definition in the Admin Portal for the data you want to access. Refer to the Admin Portal help for details.
2. In the Admin Portal, schedule the report to generate as an in-memory report, at the required frequency. Note that the report provider uses the schedule name rather than the report definition name.

**Important** We strongly recommend that you set up a schedule to generate the report at the required frequency, such as once a day, every 5 minutes and so on. This ensures that the report is generated *once* for all users in Agent Desktop. If the schedule ends or fails to run then users will see a Report Not Found error in Agent Desktop.

### Adding the IFS CE Reports provider

You need to add the IFS CE Reports provider to your app or template if you want to configure data grids or charts to show report data generated in the Admin Portal.

### Adding an action to the report provider

Add an action to the app or template to get report data for a specific report definition. In the action:

1. Select the report provider.
2. Enter `Table` as the provider type.
3. Enter `GetReport` as the method.
4. In Required Parameters, click `ReportScheduleName`.
5. In Mapping Field, click **Static** and then enter the name of the report schedule. You can copy the schedule name from the Admin Portal.

**Note** The generated report is cached in memory using the report's unique schedule name.

6. Select **Publish Topic**.

### Adding a data grid for report data

You need to configure a component to display the report data, such as a chart or data grid:

1. Add a data grid, for example, and then click  to configure it.
2. Click  and select the report provider.
3. Enter `Table` as the data type.

**Note** This is the only data type and there are no properties to search on. The properties are the formatted fields from the report definition.

4. Enter each field name and its data type:

Property name	The properties are the formatted fields from the report definition. Copy the field names from the <b>Heading</b> column on the Format page of the report definition.
Data type	Select the data type if known. Otherwise, use <b>Date</b> for datetimes and <b>String</b> for everything else.

## IFS CE Agent Desktop provider

The IFS CE Agent Desktop provider enables you to configure CE Studio apps that interact with the Agent Desktop toolbar, by running JavaScript functions that you invoke as actions attached to rules in rule sets. All the work is done by the CE Studio Runtime:

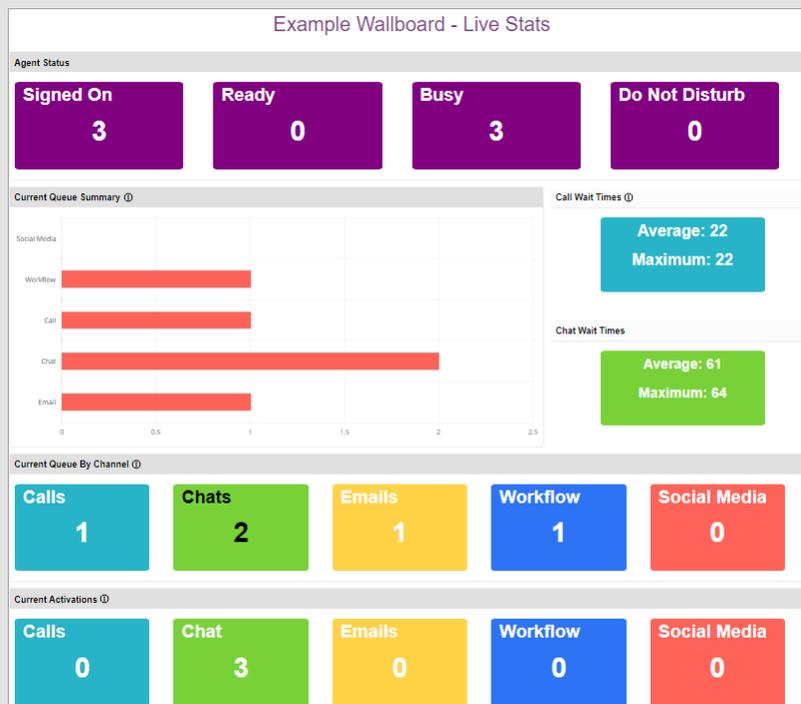
Function type	
Toolbar	<p>Functions for incoming events from Agent Desktop (and its media services) to the runtime. There is a corresponding event for each function.</p> <p>You can configure a rule set to run action sets and/or display task sets when one of these events occur. You attach them to events of type <i>OnToolbarEventReceived</i>.</p>
PostMessage	<p>Functions for outgoing events from the runtime to Agent Desktop (and its media services). Agent Desktop responds with a callback such as <i>closeEmail</i>, <i>hangUp</i>, <i>wrapUp</i>.</p> <p>You can configure a rule set to run action sets and/or display task sets after receiving a callback. You attach them to <i>OnPostMessageCallbackReceived</i> events of type.</p>

For further details, see [Event configuration for media apps and templates](#).

## Realtime Statistics provider

The IFS CE Real Time Statistics provider exposes the data types and methods for getting real-time statistics on agents signed on to Agent Desktop and activations queuing at any of the contact center's units. Data is either current data or up to 30 seconds old. The provider obtains the data by querying the contact center's media hubs.

**Note** You can download an example that demonstrates the use of the IFS CE Real Time Statistics provider, from the Central Template Repository. All the data in this example wallboard is provided by the Real Time Statistics provider:



### Using the Realtime Statistics provider types and methods

To use this provider, add actions to return data from the media hubs:

1. Select a provider type, for example `AgentStatus` or `Statistic`.
2. Select the method, such as `GetFlattenedAgentStatii` or `GetStatisticsAsync`.
3. Click **Manage Properties** and set the following parameters:

Dedupe / GroupByOcu	<p>Contact centers have multiple media hubs. You therefore need to set the parameter to <i>true</i> to remove any duplicate rows. For example, when set to <i>false</i> and used with <code>QueuedActivation</code>, each activation will be listed twice (or more depending on the number of media hubs) as each activation queues at each media hub. Setting this to <i>true</i> insures that each activation is listed once.</p> <p>To set this, select <b>Static</b> and then enter <code>true</code>.</p> <p>Default: <i>true</i></p>
OcuReference	<p>The contact center unit defaults to the agent's user contact center unit (as set in the Admin Portal).</p> <p>To set this, you can do one of the following:</p> <ul style="list-style-type: none"> <li>• Map it as a <b>Toolbar Query Param</b> and then select the <code>UserContactCenter</code> parameter.</li> <li>• Select <b>Static</b> and then enter the name of the contact center. You could also map it to a global variable that holds the name of the contact center.</li> </ul>
QueryGlobally	<p><b>Not all provider types.</b> If present, you must always map this parameter to <b>Static</b> and then enter <code>true</code>. This means that all the media hubs are queried.</p>

#### 4. Publish the topic.

### AgentStatus

The method `GetFlattenedAgentStatus` gets data about agents who are currently signed on to Agent Desktop. The properties relevant to the configuration of wallboards are summarized below:

Property	Description
ActivationId	The activation ID
ActivationTime	The time the email, call etc was accepted by the agent.
ActivationTypeName	Activation types. See <a href="#">here</a> for a list.
Agent	Agent's user name.
AgentInWaiting	Agent's user name.
AgentStateName	Current state. See <a href="#">Agent states</a> .

Property	Description
ClientAccount, ClientId, ClientName	The contact center unit, its ID or name. This may also represent a client account and for this reason these details are only shown when there is a current activation.
DndCode	The reason for the agent being unavailable (in Do Not Disturb mode). These are the default reasons: <ul style="list-style-type: none"> <li>• 0 Not in Do Not Disturb mode</li> <li>• 1 Bathroom break</li> <li>• 2 Short break</li> <li>• 3 Long break</li> <li>• 4 Meeting</li> <li>• 5 Training</li> <li>• 6 Coaching</li> <li>• 7 Admin</li> <li>• 8 Escalations</li> </ul>
DndCodeName	
LogonTime	The time the agent signed onto Agent Desktop.
MonitoredStation	The number of the station that is being monitored.
MonitorTypeName	The type of monitoring: Monitor, Coach, Assist
OcuName	The contact center unit that the agent is signed onto.
Offline	An agent is offline when they are in the Idle state. An agent is offline when in DND mode or when they are using a CE Studio app that is accessed by clicking  on the Agent Desktop toolbar. This parameter is a Boolean where offline ( <i>false</i> ) and online ( <i>true</i> ).
OwningOcuName	The contact center that the agent is assigned to. This is set in the Admin Portal on the <b>User Management &gt; User Contact Centers</b> page.
SentTime	When the information in <code>AgentStatus</code> was sent from the mediahub.
ServerName	The name of the media hub that the agent is logged into. Will be empty if <i>Dedupe</i> is set to <i>true</i> .

Property	Description
SkillsetId	The skillset of both the agent and the activation. 0 when the agent is idle.
StateChangeTime	When the agent's state changed.
Station	The station that the agent is logged into.

### MonitoredOperator

The method `GetFlattenedMonitoredOperators` gets data about agents who are being monitored. The properties are identical to `AgentStatus`. Note the following:

Property	Description
SupervisorStation	Is the person who is doing the monitoring.
OperatorStation	Is the person who is being monitored.

### QueuedActivation

The method `GetFlattenedQueuedActivations` gets data about activations in the queue. The properties are identical to `AgentStatus`. Note the following:

Property	Description
MatchedStation	The item is queuing for a specific station (agent). For example, a callback for a specific agent. This is set to 0 if the item is queuing for the next available agent.
OcuId, OcuName	Contact center unit that the activation is queuing on.
Priority	Priority of the activation in queue (a number 0–9).
TimeDue	The time that the activation is due to be presented to the agent.

### Statistic

The method `GetStatisticsAsync` gets aggregated data about active activations and what's in the queue.

Property	Description
AgentsAvailable	Total number of agents currently in the Idle state.
AgentsDnd	Total number of agents currently in DND mode.
AgentsInWrap	Total number of agents who have completed their current activation and are wrapping up their work on this activation.
AgentsLoggedOn	Total number of agents signed on to Agent Desktop
CallsInbound	Total number of callers currently talking to agents.
CallsInProgress	
OcuId	ID of the contact center unit
OcuName	Name of the contact center unit
QueuedCalls	Total number of calls in the queue
QueuedChats	Total number of live chat callers in the queue
QueuedEmails	Total number of emails in the queue
QueuedItems	Total number of items in the queue
QueuedSocialMedia	Total number of Facebook and Twitter messages and posts in the queue
QueuedWorkObjects	Total number of work objects in the queue, such as call backs

## Workflow provider

The IFS CE Workflow provider exposes the data types and methods for creating and processing work objects belonging to a work definition.

A work definition defines the data in the work objects as work elements. Work elements are dynamic properties and you need to add the work element names to the CE Studio app before you can configure actions that filter by work elements. Typically, you add the work element names as columns in a data grid.

A work definition also defines the business process as a series of events, actions and states. For example, a work object might be the details needed by an agent in order to call a client at a scheduled time. The configured events, actions and states allow the agent to cancel, reschedule and complete the call back. Work definitions are configured in the Admin Portal.

**Note** We provide a simple example of a work definition for call backs (IFS CE Demo Callback Workflow Definition and a template that uses it (the IFS CE Callback Template). For further details of work definitions, including work definitions for call backs, see the Admin Portal help.

You need to configure CE Studio actions for all the custom events defined in a work definition. The actions use these Workflow provider data types and methods:

Data Type	Description
WorkObjects	<p>A work object, with data as defined by the work elements in the work definition.</p> <p>A useful method on this type is <code>WorkObjects.GetTabularResponse</code>. Using this you can then filter by the work elements that you have already added to the CE Studio app, typically, columns in a data grid.</p>
CreateWorkObjectResponse	<p>The work object to be created.</p> <p>Data for the work elements defined in the work definition are passed using <code>CreateWorkObjectRequest.Elements</code></p>
ExecuteWorkEventResponse	<p>Runs an event on a work object, including updating any data passed using <code>ExecuteWorkEventRequest.Elements</code></p> <div style="background-color: #f0f0f0; padding: 5px; margin-top: 10px;"> <p><b>Note</b> Events are uniquely identified either by an event ID or by the event and work definition names.</p> </div>
OpenWorkObjectResponse	<p>Opens an existing work object, obtaining a lock on it.</p>
QueueWorkEventResponse	<p>Runs an event on a work object that is either currently locked by another user or which doesn't require a lock.</p> <p>Data to be updated can be passed using <code>ExecuteWorkEventRequest.Elements</code></p>
CloseWorkObjectResponse	<p>Clears the lock on a work object once processing is completed.</p>
WorkDefinitionIndexedElements	<p>Gets indexed work elements for a work definition.</p>

Data Type	Description
WorkObjectIndexedElements	Get work object values for a work definition with indexed work elements.

### CloseWorkObjectResponse

Use the `CloseWorkObject` method to clear the lock on a work object, for example, after the agent clicks a cancel button. There are two required parameters:

- *LockId* – map the lock ID to the toolbar query parameter *WorkLockId*
- *WorkObjectId* – map the work object ID to the toolbar query parameter *WorkObjectId*

### CreateWorkObjectResponse

Use the `CreateWorkObject` method to create a new work object with the properties (elements) defined by the work definition. This method has the following required parameters:

- *ClientReference* – select **Use Null**.
- *InitialStateReference* – select **Ignore**.
- *LinkId* – what the work object relates to, such as the call ID, chat ID, activation ID. For a home page app, use a Toolbar Query Param such as `UserUniqueGuid`.
- *LockType* – select **Agent App (7)**.
- *ObjectReference* – the value that's used as the object reference for the work object. For example, any numeric value such as a reference number, a phone number, and so on.
- *WorkDefinitionReference* – the work definition reference as used in the app. The form of the reference depends how you configure the methods `ExecuteWorkEvent` or `QueueWorkEvent`. For example, it could be the work definition name.

Optional parameters are the work elements in the work definition. This is the data type `CreateWorkObjectRequest.Elements`.

### ExecuteWorkEventResponse

Use the `ExecuteWorkEvent` method to run an event on the work object. The data which the event needs to process is stored as work elements in the work object and is passed using the child data type `ExecuteWorkEventRequest.Elements`. The user must have a lock on the work object. Work events are uniquely identified either by an event ID or by the event name and work definition.

This method has these required parameters:

- *EventId* – the event ID. You will need to use this if there are several work definitions with the same event name. Select **Use Null** for *EventName* and *WorkDefinition*.
- *EventName* – the name of the work event to run. You must also enter the work definition.

- *LockId* – map this to the toolbar query parameter *WorkLockId*.
- *WorkDefinition* – map this to the appropriate component in your app. You must also enter the event name.

The method also has these optional parameters:

- `ExecuteWorkEventRequest.Elements` – add any work elements defined in the work definition that you want to process.
- *Flags* – determines whether the lock on the work object is retained or released after the event runs. Enter 1 to clear the lock and 0 to keep the lock.
- *WorkEventRequestId* – select **Use Null** for this. (This is the ID you would like the work event to queue as.)

### OpenWorkObjectResponse

Use the `OpenWorkObject` method to open a work object. This locks the work object for the user and therefore allows them to update the work object. The required parameters are:

- *LinkId* – what the work object relates to, such as the ID of another work object, the call ID, chat ID, activation ID.
- *WorkObjectId* – the ID of the work object to open.

Click **Manage Properties** to see the available optional properties, such as *Reason* and *Timeout*.

### QueueWorkEventResponse

Use the `QueueWorkEvent` method to run an event on a work object that either does not require a lock or that is locked by another user, in which case the event will run once the lock is cleared. Work events are uniquely identified either by an event ID or by the event name and work definition.

Required parameters are:

- *EventId* – the event ID. You will need to use this if there are several work definitions with the same event name. Select **Use Null** for *WorkDefinitionReference*.
- *ObjectId* – the ID of the work object
- *ReferenceId* – a guid of an associated work object. If you set this then you also need to set *ReferenceType*.
- *ReferenceType* is one of the following numerical values:
  - 0 – Agent Desktop activation attempt
  - 1 – Agent Desktop activation
  - 2 – System activation
  - 3 – Admin
  - 4 – Work timer
  - 5 – Script open

- 6 – iResults
- 7 – Agent app
- 8 – Client service
- 9 – Inbound call
- 10 – Campaign manager
- 11 – Chat
- 12 – Email
- *WorkDefinitionReference* – optional if you have mapped the event ID. Map this to the appropriate component in your app. You must also enter the event name.

Click **Manage Properties** to see the other optional properties which include:

- `ExecuteWorkEventRequest.Elements` – add any work elements defined in the work definition that you want to process.
- *Flags* – whether the lock on the work object is retained or released after the event runs. Enter 1 to clear the lock and 0 to keep it.
- *EventName* – the name of the work event to run. Required if you use *WorkDefinitionReference*.
- *DueTime*

## WorkObjects

Use the `GetTabularResponse` method with a filter to get the work object to display in the app when it loads in Agent Desktop. For example, in the filter, map the *Id* property on *WorkObjects* to the Toolbar Query Param property *WorkObjectId*.

You also need the work definition Id. For example, map the Field Reference `WorkObjects > WorkDefinitionId` property to `WorkDefinitions.Id`.

When working with data grids, for example, you can filter by the element values:

**Filters**

And     Or     Dynamic Filter    Delete    Add Condition    Add Group

	<div style="border: 1px solid #ccc; padding: 2px; display: inline-block; background-color: #e0e0e0;">  WorkDefinition    Id         </div>	Equals ▼	<div style="border: 1px solid #ccc; padding: 2px; display: inline-block; background-color: #e0e0e0;">  workdefinitions    WorkDefinitions.Id_1         </div>	
	<div style="border: 1px solid #ccc; padding: 2px; display: inline-block; background-color: #e0e0e0;">  Elements    W_EMAIL_SUBJECT         </div>	Like ▼	<div style="border: 1px solid #ccc; padding: 2px; display: inline-block; background-color: #e0e0e0;">  workObjectsTable    searchByEmailSubject         </div>	

**Note** Work elements are dynamic properties and you need to add the work element names to the CE Studio app before you can configure actions that filter by work elements. Typically, you add the work element names as columns in a data grid.

### WorkDefinitionIndexedElements and WorkObjectIndexedElements

To get a list of indexed work elements, use the `WorkDefinitionIndexedElements` method `GetTabularResponse`. In the action, filter by the work definition id and by the name of the required work elements. These elements must be set as indexed in the work definition.

To then filter work objects by specific values in an indexed work element, use the `WorkObjectIndexedElements` method `GetTabularResponse`. In the action filter by these two properties:

- Work definition element id, ie `WorkObjectIndexedElements > WorkDefinitionIndexedElementId`
- Value, that is `WorkObjectIndexedElement > Value`

## About work definition elements

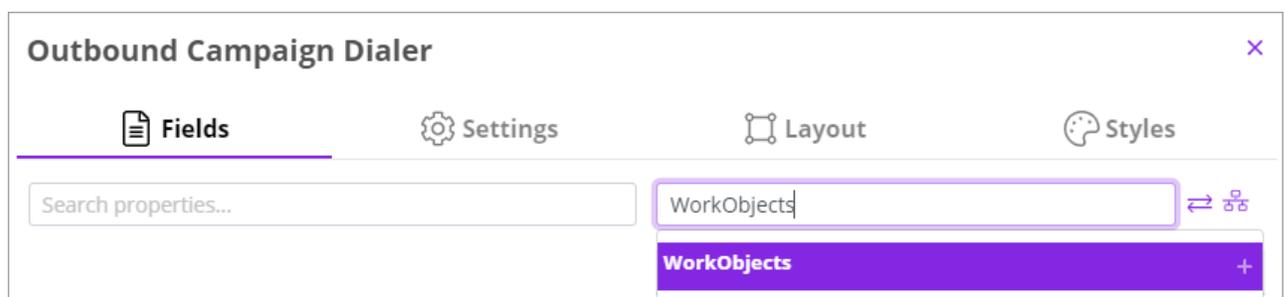
When building a CE Studio app for workflows, you can use the Metadata Selector to select the work definition elements from the `Elements` type. See [Using the Metadata Viewer](#).

The following steps describe an alternative way of adding the elements defined in the work definition as properties of the workflow provider.

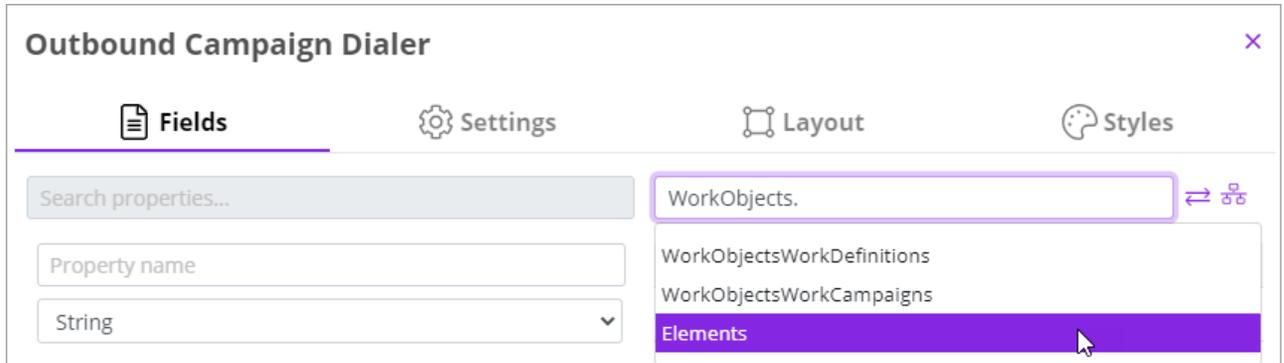
To select elements as workflow provider properties for a component or element:

1. Add or edit a component or element.
2. Click  and select the workflow provider.
3. In the Filter types field, type `WorkObjects`.

The filter shows **WorkObjects+** indicating that there are child types.



4. To see child types, type a period ( . ).



5. Select **Elements**.
6. On the right, you can now enter the name of an element defined in your work definition.
7. Repeat this for each element required in the CE Studio app.

## Tenant oData 4 service

Third-party systems can use the Tenant APIs to query IFS Customer Engagement, send requests and pull data in real time for consumption in third-party systems, such as workforce management or Customer Relationship Management systems.

The APIs enable a third party to:

- Query entities in CE such as users, skillsets, contact centers, workflow definitions, work objects. For example, to query what's currently waiting in the queue.
- Run create, patch, and delete operations to create work objects and other entities in their tenant, such as users. For example, to create call data for outbound campaign calls.
- Download call recordings and request the download of archived call recordings.

All actions performed through the Tenant APIs are audited. You can view the audit trail in the Admin Portal on the **Standard Reports > Audit History** page. The user that is shown in the audit history is the account used by the Tenant oData 4 service.

### Metadata

To view the metadata for the Tenant oData 4 service, go to the base URL, in your tenant. For example, if the URL for CE Studio Designer is:

```
https:// ce-uk.testing.com/<tenant-name>/ce/studiodesigner/
```

then the base URL is:

```
https://ce-uk.testing.com/<tenant-name>/ce/api/odata/v1/$metadata
```

**Note** To view the metadata, you need to sign on to the tenant, for example, to CE Studio Designer. Your user profile needs the Supervisor role or higher.

Alternatively, you can view the metadata by adding the provider to a CE Studio app and then using the [Metadata Viewer](#).

### Authorization

Backend services that use the APIs require an API key. You can request this through IFS Service Center. The key must be passed in a bearer token, in the authorization header of all requests to the Tenant oData 4 service.

### Creating a provider

You can create an oData provider to test the Tenant APIs. The URL will be similar to this:

```
https://ce-uk.testing.com/<tenant-name>/ce/api/odata/v1/
```

## OpenAPI providers

---

OpenAPI providers support:

- OpenAPI version 2 or 3
- Metadata structure that follows the common structure defined in the Swagger documentation for the version.

**Note** OpenAPI providers do not support file download.

Credentials for OpenAPI providers:

- Authentication scheme is API Key or Basic Authentication.
- Environment depends on the version:
  - Version 2 – {baseUrl}/swagger.json
  - Version 3 – {baseUrl}/openapi.json

For example,

```
https://petstore.swagger.io/v2
```

```
https://petstore3.swagger.io/api/v3
```

```
https://api.boomi.com/api/rest/v1
```

Creating OpenAPI providers:

1. Select the correct credentials object as this supplies the environment path.
2. Enter / as the service name.

The environment path, for example, will be `https://petstore3.swagger.io/api/v3/`.

## System provider

The IFS CE System provider provides miscellaneous provider types and methods required by IFS Customer Engagement, for example for public portal authentication, message translation and SMS sending.

### GenerateSecuredLink

See [URI and Link parameters](#).

### GetAccessiblePublicPortalApps, GetAccessiblePortalApps

See [Adding links to portal apps](#) or [Adding links to public portal apps using a Portal List Template](#).

### Methods for public portal authentication

See [Authentication for public portals](#) for details.

### Send SMS

To send SMS messages from a CE Studio app, use the `SendSmsResponse` method `SendSms`. It has the following parameters:

Parameter	Description
appRef	Optional parameter included for backwards compatibility. Used to store any required data. Select <b>Use Null</b> .
appTag	Optional parameter included for backwards compatibility. Used to store any required data. Select <b>Use Null</b> .
body	The message to send.
clientid	Optional parameter for the contact center unit guid. Select <b>Use Null</b> . Note: This parameter is required if the <i>from</i> parameter is not set. You can only obtain the contact center unit guid from IFS Support.
from	The phone number to use for SMS sending. This parameter is required if the <i>clientid</i> parameter is set to Use Null.

Parameter	Description
messageld	This is required if the SMS message is part of a conversation. For outbound SMS sending, select <b>Use Null</b> .
MessageType	Enter 1 for standard SMS. Other message types are not supported.
SenderType	Enter 5 for CE Studio apps.
to	The phone number to send to.

## Troubleshooting providers

It is possible for an import of a CE Studio app to fail if the target environment has:

- Different live versions of the providers
- A newer version of a provider that contains changes that are not backwardly compatible
- Does not have an updated version of the provider

### Error with external service

Error messages prefixed with the phrase `Error with external service` are responses from the endpoint, such as IFS Cloud. These are not IFS Customer Engagement errors. You need to check for the cause in IFS Cloud.

### Before importing

1. Check that the providers and versions are correct for the CE Studio app you need to import.
2. If you are planning to map the providers in the ceapp file to the live version of each provider then make sure the correct version is live.
3. Check the providers, such as the System provider. For example, to import a portal app you need to map to:
  - Either version 5 or later of the IFS CE System provider
  - Or make version 5 of the IFS CE System provider live and then map to the live version

### Mapping to the live provider version

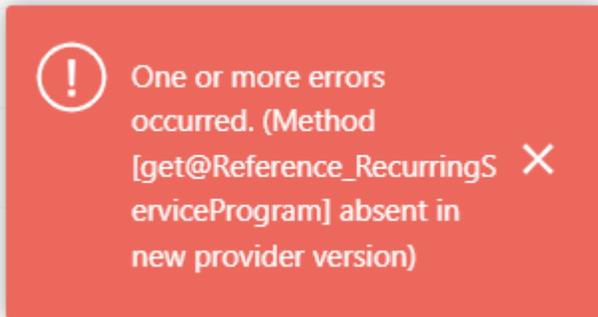
An import may fail if the wrong provider version is live (see above).

For example, if a CE Studio app requires version 5 of the IFS CE System provider but version 1 is the live version, then the import will fail if you map the IFS CE System provider in the ceapp file to the live version. Any actions in the app that use this provider may be missing from the imported app.

To fix this, you need to delete the CE Studio app, change the live version of the provider to the correct one and re-import the app.

### Finding out why an import or clone is failing

During an import or clone, you may occasionally see warning messages similar to the following:



This message tells you that there is a difference between the metadata for the old and new provider versions. The provider type is named in the method, that is `RecurringServiceProgram`.

In this example, the CE Studio app is configured to use the `RecurringServiceProgram` type, for example, an action gets data for a data grid or form that uses properties returned by the method in question. However, in the new provider version, the method does not return the same properties, and one or more properties might be missing. This prevents the import or clone from succeeding.

To fix this, begin by checking which properties are involved:

1. In the original CE Studio app, locate the action that uses the method named in the warning message.
2. On the Actions page, find out where the action is used (the Usage tab).
3. On the Events page, check the OnNotification events to find out which component or element subscribes to this action.
4. On the Designer page, check the component or element in question, and get a list of the configured properties.

These properties are potentially preventing the import or clone from succeeding. The next step is compare the old and new provider versions.

5. Still in the original CE Studio app, in the Metadata Viewer, select the provider, find the provider type, and check the list of properties.
6. Create or open a CE Studio app that uses the new version of the provider, and repeat the previous step. Look for differences between the two versions.

Once you know what the difference is between the provider versions, you can update the CE Studio app, for example update or remove the properties in question.

## 6

## Components and elements

Details of working with the different component and element types.

**Note** There are some sample templates available for download from the Sample Templates page (in the help if you are reading this in a PDF). These demonstrate the range of components available in CE Studio Designer.

The screenshot displays a stack of components in the CE Studio Designer interface. The components are:

- Data Grid (Single Row Selection):** A panel with a purple header containing the text: "You can map a data element to one or more properties in your data source (like this {0}, {1}) or use it to show static text." Below the header are two data rows:
 

{0} agents are ready	{0} agents online
	{0} are busy with callers and {1} are away from their desk
- Data Grid (Multi-Row Selection):** An empty data grid component.
- Forms and UI Elements:** A panel for configuring forms and text boxes.
- Forms and Text Boxes:** A panel for configuring forms and text boxes.
- Chart:** A line chart with a red line and circular markers. The y-axis ranges from 0 to 1.2. The data points are approximately: (1, 0.7), (2, 0.45), (3, 0.95), (4, 0.55), (5, 0.7), (6, 0.35).
- iFrame:** A video player showing a YouTube video titled "AGENT DESKTOP OVERVIEW IN IFS CUSTOMER ENGAGEMENT".
- Overlay:** A panel with a green button labeled "Click Me".

Below the stack, a caption reads: "This is a stack panel, used as a footer for this page."

**Note** You can download an example that demonstrates the range of components available in CE Studio Designer. Available from the Store which is part of CE Studio Designer.

The screenshot shows the components palette on the left and a stack panel preview on the right. The palette includes:

- Data Grid (Single Row Selection)
- Data Grid (Multi-Row Selection)
- Forms and UI Elements
- Forms and Text Boxes
- Data Elements (see tooltip)
- Chart (see tooltip)
- Iframe
- Overlay

The stack panel preview shows a 2x2 grid of data elements and a chart. The data elements are:

- Top-left: You can map a data element to one or more properties in your data source (like this {0}, {1}) or use it to show static text.
- Top-right: {0} agents online
- Bottom-left: {0} agents are ready
- Bottom-right: {0} are busy with callers and {1} are away from their desk

The chart shows a line graph with a red line and a green area under the line. The y-axis ranges from 0 to 1.2. The x-axis has 10 ticks. The data points are approximately: (1, 0.7), (2, 0.4), (3, 0.9), (4, 0.5), (5, 0.7), (6, 0.3).

The iframe shows a video player with the title "AGENT DESKTOP OVERVIEW IN IFS CUSTOMER ENGAGEMENT".

The overlay shows a green button with the text "Click Me".

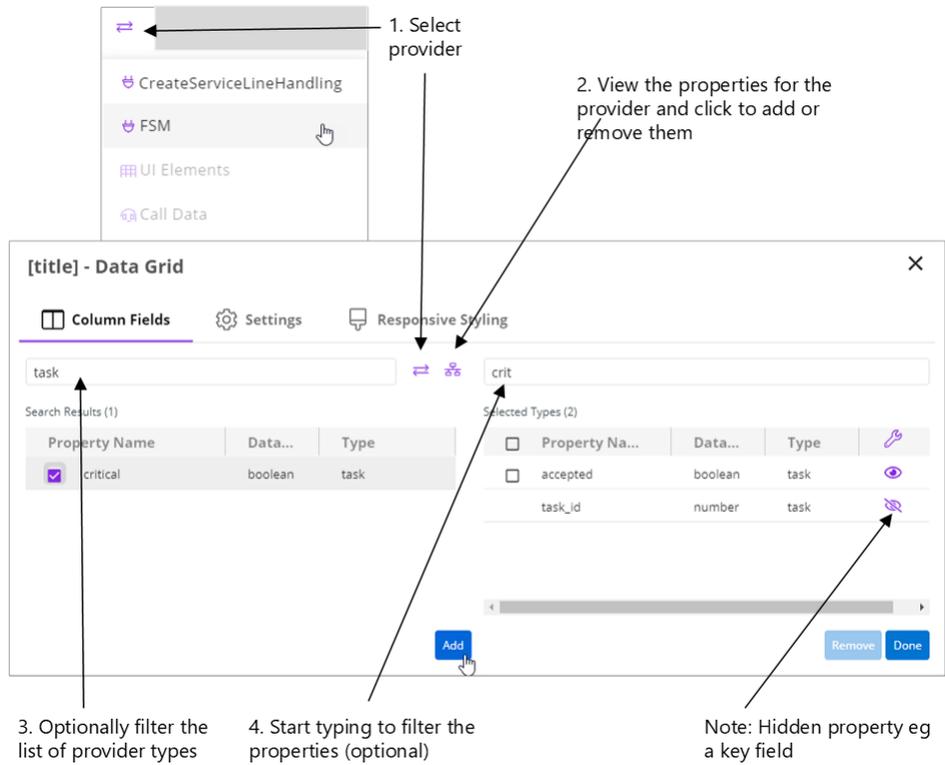
Below the stack panel preview, it says: "This is a stack panel, used as a footer for this page."

## Adding components and selecting properties

Follow these steps to add a component, such as a data grid or form, and select the properties to display:

1. In CE Studio Designer, go to the Designer page .
2. In an empty layout cell, click  and select **Element > Data Grid** or **Element > Form**.

You are then prompted to select the properties.



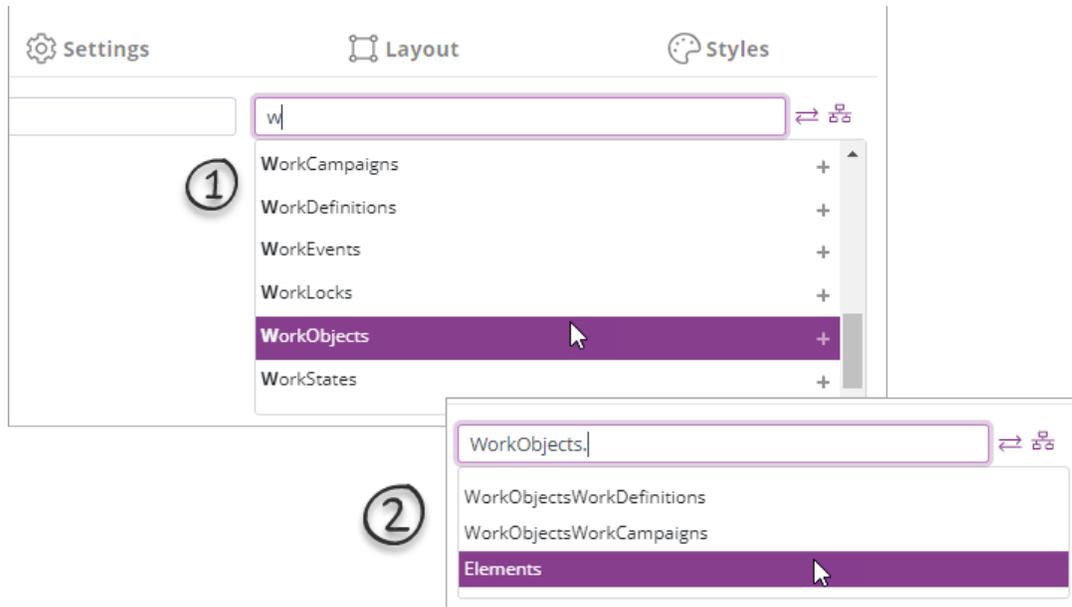
### Selecting properties for a data grid

- Click to select the provider. Here are some of the options that will be displayed:

<i>provider</i>	Lists the providers that you have added to the app or template.
Transformed Data	Data that has been manipulated using data transformations. For example, a full name property that's derived from first and last name properties. See <a href="#">Data transformations</a> .
UI Elements	Elements such as check boxes, text fields, that you can add to forms. If added to a data grid, these appear below the data grid.
Call Data	IFS Customer Engagement only: Any call data is automatically available to the app or template. You don't need a provider for this.

- Click to view the properties of the provider. You can also select properties in the [Metadata Selector](#).
- Type the first few letters of the type name to list matching types. The properties are listed after you select a type.

If a provider type has subtypes then a + appears against the name. Select the type and then type a period (.) to list the subtypes.



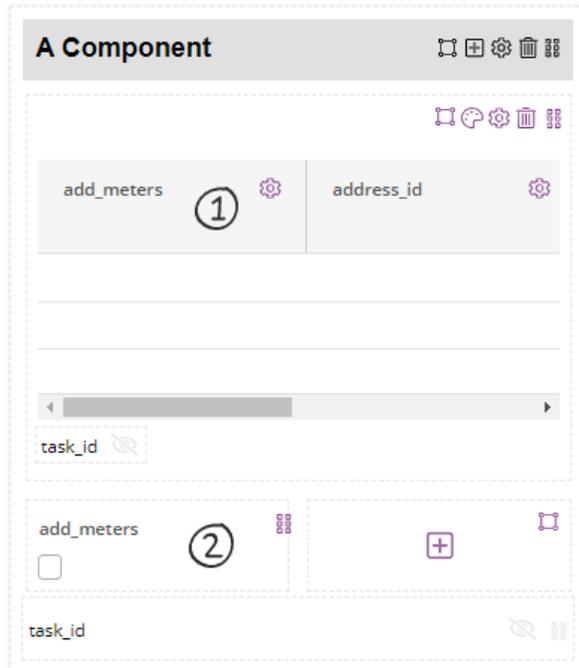
*Selecting a provider subtype (child)*

Note that:

- A component or element can only contain properties from one provider (forms are an exception - you can add properties from multiple providers).
- Mandatory properties, such as primary keys, are added automatically and are hidden by default.
- You can only add a property once to the same element.

### Adding and removing properties from a data grid

Once you have created a data grid, click  on the element toolbar to add and remove properties from the data grid. Clicking  on the component toolbar will add the properties as form fields below the data grid.



Two instances of the `add_meters` property – one in the data grid and one in the form

## Configuring component properties

Click  on the component toolbar to configure the component properties: Fields, Settings, Layout and Styles.

### Fields

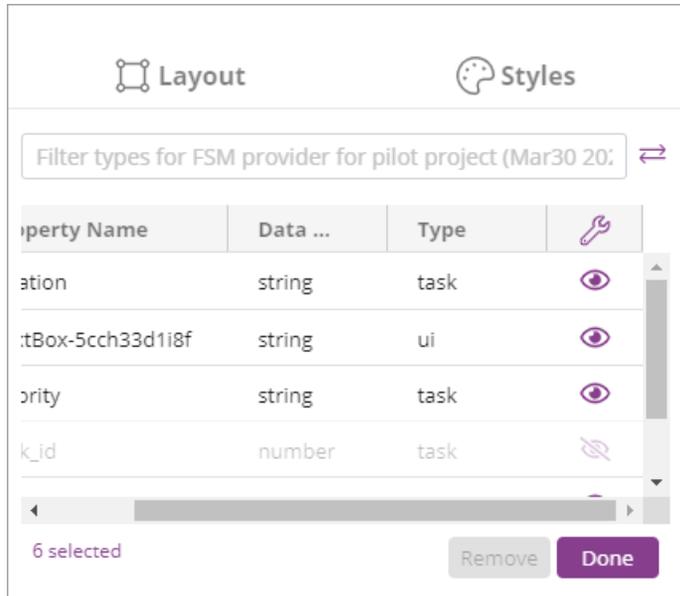
In the component properties dialog , use the **Fields** tab to add fields to the component or remove fields that are not referenced in any actions and rules.

A component can only contain properties from a single provider, as well as UI elements.



Example of a custom field created from a Number field (the default title is the qualified name)

- Click  and select the provider. The list shows the providers added to the app or template in addition to UI Elements and Call Data.
- After adding a property, scroll to the right and click  to hide or show the property:



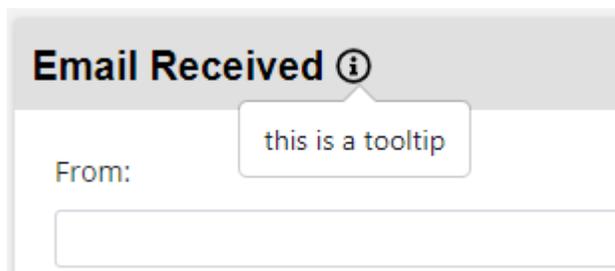
### Hiding and showing properties

- After adding a UI element and closing the dialog, click  on the element toolbar to configure the field. This type of field is a custom field.

### Settings

In the component properties dialog , use the **Settings** tab to:

- Add an identifier to make the component easier to identify when adding actions and display tasks.
- Select a message ID to localize the app or template.
- Add a tooltip to tell your users how to use a particular field. Position the tooltip relative to the field using the **Tooltip Popover** setting. For example:



- Turn off **Expandable** — components are expandable by default.
- Add a *timer* to trigger an OnTimerElapsed event.

## Layout

In the component properties dialog , use the **Layout** tab to:

- Set the number of columns in the component. Individual elements in the component can still split and merge the columns.
- Set the tab order for the fields. To see the tab index of a field, click  on the CE Studio Designer toolbar and then click the info icon next to the field.

## Styles

In the component properties dialog , use the **Styles** tab to change the formatting of the component:

- Override the theme inherited from the app settings or change the tint of the component header.
- Hide the header on the component — this also hides the white component background.
- Add an icon to the header.

## Field properties

**Important** Always save the app or template after adding UI elements. This enables you to fully configure them.

Click  on the element toolbar to configure the field properties: Data source, Display and Validation.

### Data source

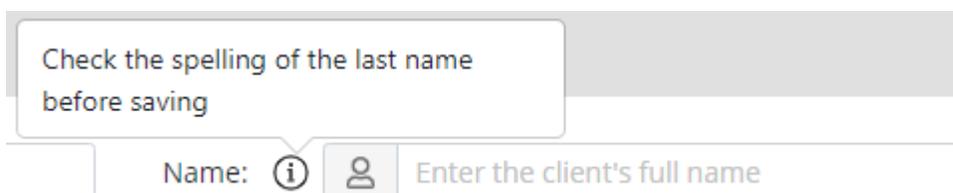
The data source properties for provider fields are set for you when you add a field to a component. You can edit some of the properties in the Field Properties dialog. You need to configure all the properties when you add a UI element as a custom field.

Title	The label for the field. Click  if you want to select a message ID for use when localizing the app or template.
Placeholder	Descriptive text, such as a hint to the user, on what to enter in the field.
Is Custom Field	This is automatically selected when you add a UI element to a component. For an example of a custom field, see <a href="#">Combo boxes</a> .

Provider Id	Depends on the field type: <ul style="list-style-type: none"> <li>Shows the provider selected when you added a provider property.</li> <li>Empty for UI elements — use <b>Lookup Provider Id</b> for fields such as combo boxes.</li> <li>Empty for call data IFS Customer Engagement only.</li> </ul>
Qualified Name	Only shown once you save the app or template.
Field Identifier	For custom fields created from UI elements. You must add an ID before you can save the field, for example, you can copy the default title. <div style="background-color: #f0f0f0; padding: 10px; margin-top: 10px;"> <p><b>Important</b> Do not change the identifier once you start to add actions that rely on this identifier. If you change the identifier then you must update any actions that rely on the identifier.</p> </div>
Lookup Provider Id	Provider that provides the lookup values for combo boxes. See <a href="#">Combo boxes</a> .
Lookup Code Name	The provider property that provides the lookup values for a combo box.
Depends on Qualified Name	Makes the combo box into the child of another combo box in the app or template. See <a href="#">Combo boxes</a> .

## Display

In the Field Properties dialog, use the **Display** settings to format the field, for example to add an icon and a tooltip. You can also use this to include the time in the DatePicker UI element:



*Example of some display options for text boxes*

Note that:

- Description** on the Data Source tab provides the hint that's displayed in the field.
- The **Inline** label alignment is shown only when previewing the app and not on the Designer page.

## Validation

In the Field Properties dialog, use the **Validation** settings to verify user's input. The data will be validated at the point when the user saves the entry.

You can also use a regular expression:

- Either enter a regular expression in **Adhoc Regex**
- Or select an existing regular expression. Add these by clicking **Global Configuration** on the left:



**Note** There is a sample template (available for download from the Central Template Repository) that demonstrates the use of regular expressions in CE Studio Designer.

## Template properties

**Note** These apply to components in CE Studio apps that embed a template.

Click  on the template placeholder toolbar to set the options for the embedded template. The App Template dialog is empty if the embedded template has no options:

Key	Value
Allow Reply	true
Show History	true

For details of configuring options when designing templates, see [Adding configuration options to templates](#).

## Adding a custom field

You can add a UI element as a custom field:

1. On the Designer page, click  on the component toolbar.
2. Add the UI element to an existing component, such as a form or data grid:
  - a. On the **Fields** tab, click  and select **UI Elements**.
  - b. Select an element.
 

Depending on the element type, you may be able to convert it later to a different type. See [Combo boxes](#).
  - c. Click **Add** and then **Done**.
3. Configure the UI element:
  - a. Go to the new field and click  on the element toolbar.
  - b. Update the field properties as needed. See [Field properties](#).
  - c. In **Field Identifier**, enter an ID. You won't be able to save the field without this.

## Adaptive cards

---

Adaptive cards for use in CE Studio apps and queue progression scripts are created using the Microsoft Adaptive Card editor.

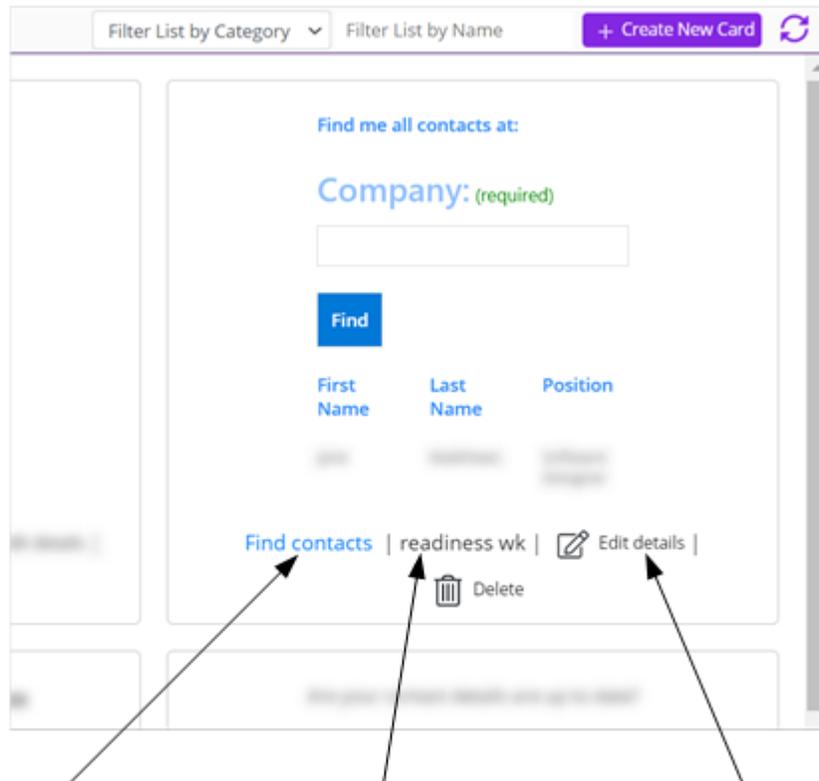
You can use both the Microsoft editor which is available at <https://adaptivecards.io/designer/> or you can use the same editor embedded in Studio Designer. The version of the Microsoft editor embedded in CE Studio has additional features to support placeholder and action input mapping.

To open the Microsoft Adaptive Card editor in CE Studio:

1. Go to **Global Configuration** (on the sidebar) > **Card Catalog**.
2. Click **Create New Card**.

The Card Catalog is initially empty.

3. Create the adaptive card in the usual way. If you are new to adaptive cards, then refer to the Microsoft documentation and samples at <https://adaptivecards.io/samples/>.
4. Map the placeholders and the submit actions using the **Configure Mapping** option. This step is required to make the cards functional in CE.



Card name – click to modify the card

Category – use this to filter what's shown in the catalog

Change the card name or category

### The Card Catalog

See also:

- Create new adaptive cards, please refer to the resources available at <https://adaptivecards.io/designer/>.
- [Configuring placeholders in adaptive cards](#)
- [Configuring action inputs in adaptive cards](#)

## Configuring placeholders in adaptive cards

You need to configure the placeholders in the adaptive card so that you map them to your data sources. Mapping is done when the card is used in the Media Script Editor or in CE Studio.

It is also helpful at this stage to add sample data to the adaptive card if you have not already done so (see [Example - configuring placeholders in adaptive cards](#)). The sample data is also used later to preview the adaptive card when adding the card to CE Studio apps.

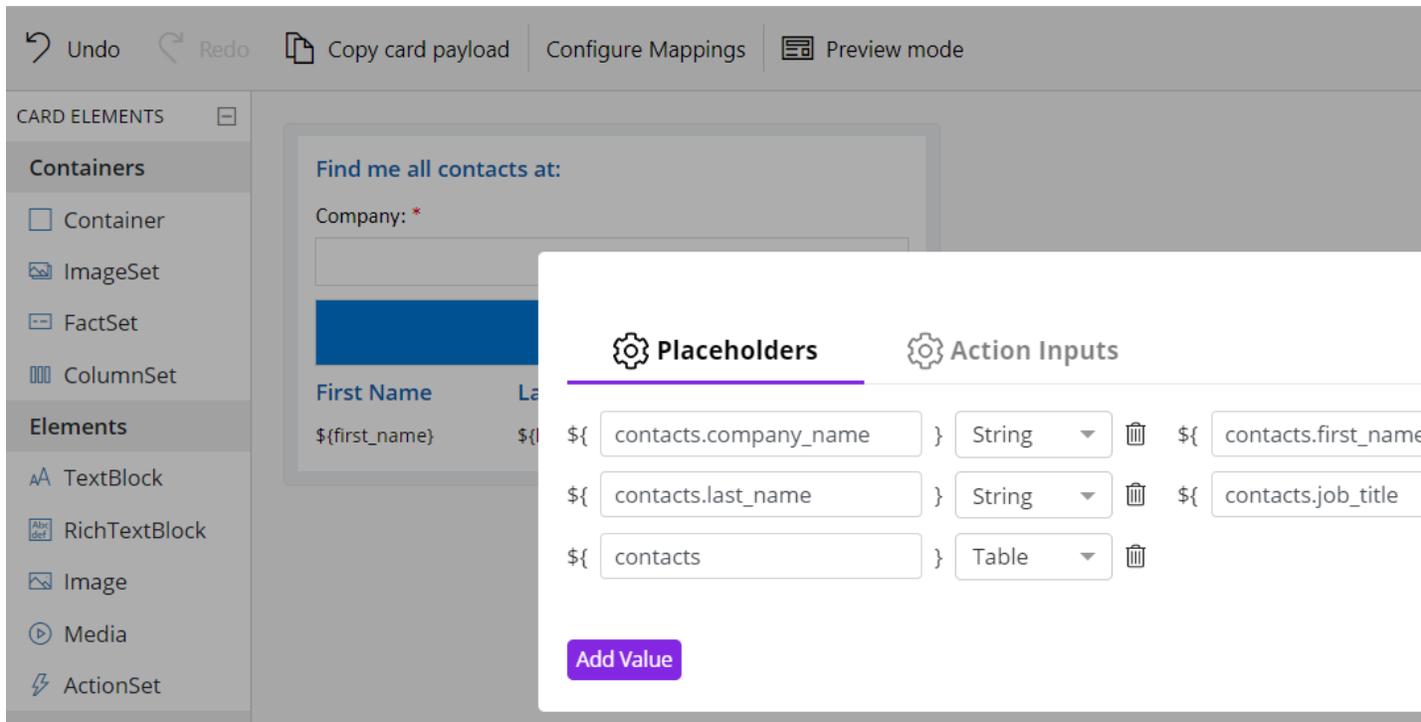
**Example data for an adaptive card with the placeholders shown below**

```
{
  "contacts": [
```

```

    {
      "first_name": "John",
      "last_name": "Smith",
      "company_name": "IFS",
      "job_title": "Software Designer",
      "contact_sequence": "123"
    }
  ]
}

```



*Example placeholders for the above sample data*

To configure the placeholders in an adaptive card:

1. Click **Configure Mappings** on the adaptive card menu bar.
2. Go to the **Placeholders** tab.
3. Map the key of the JSON object literal to the Table type. This is `contacts` in the example given above.
4. Map each placeholder in the adaptive card to a key pair in your sample data.
5. The dialog may suggest placeholders for the top level of the card (`$root`). Delete these.
 

For example, you need the placeholder `contacts.first_name` but you don't need the placeholder `first_name`.
6. Click **Continue** to save the mapping or **Close** to cancel your changes.

You can also click **Restore** to revert to the original placeholders from the payload. Note that this does not restore your edits to these placeholders.

## Example - configuring placeholders in adaptive cards

Consider this sample data (a JSON string):

```
{
  "contacts": [
    {
      "first_name": "John",
      "last_name": "Smith",
      "company_name": "IFS",
      "job_title": "Software Designer",
      "contact_sequence": "123"
    }
  ]
}
```

Based on the above example then:

- You use `contacts` as the data context that you set on containers, ColumnSets, FactSets and so on. When you configure a placeholder for this in the **Configure Mappings** tab, you select the Table type. This is equivalent to the Topic Subscription in CE Studio.
- When configuring element properties, you use `first_name` as the placeholder for first names. When used as a placeholder for an input field or TextBlock then you enter `${first_name}` and when you configure this in the **Configure Mappings** tab, you enter `${contacts.first_name}` and select the String type.

The image displays two screenshots of the IFS Studio Designer interface, illustrating how data context is configured for a parent element and its child.

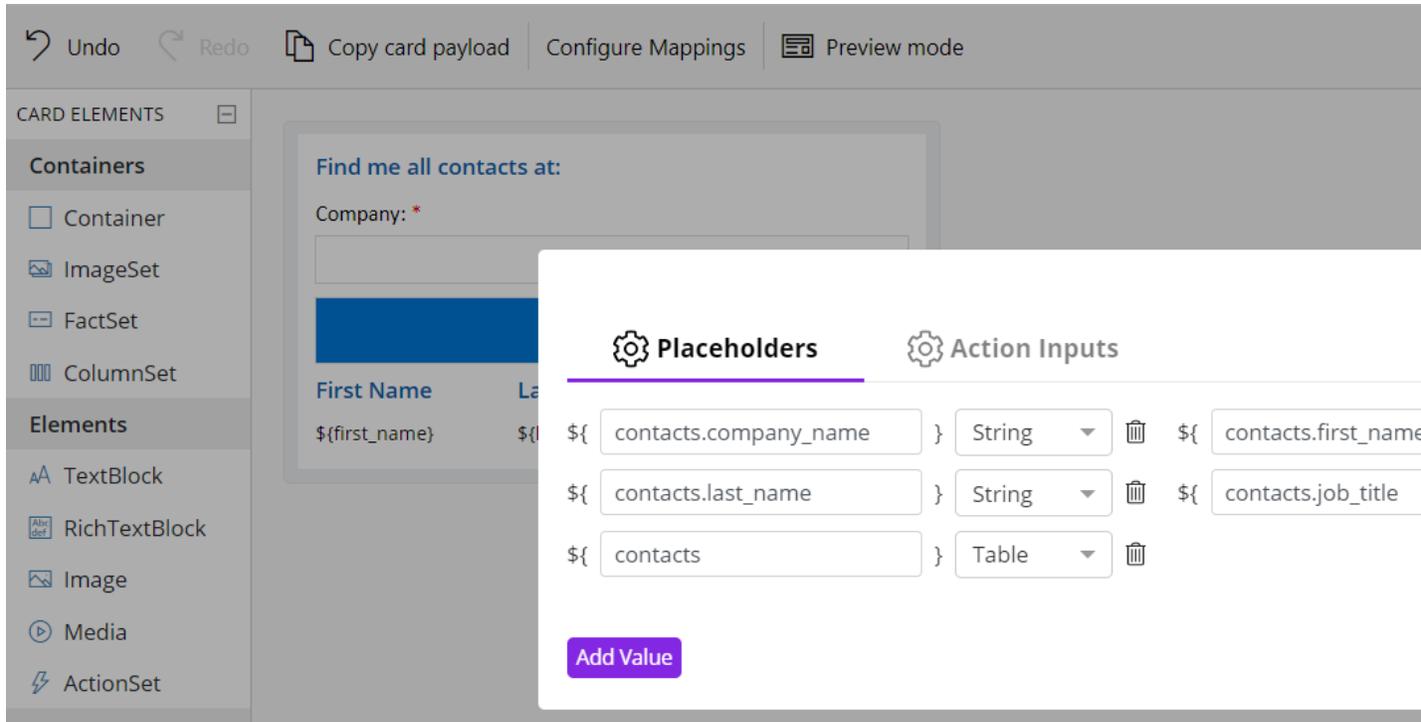
**Top Screenshot: ColumnSet Properties**

- CARD STRUCTURE:** A tree view on the left shows a hierarchy starting with 'ColumnSet [3 columns]', which is selected and highlighted in blue. Below it are 'Column [Stretch]' and 'TextBlock [\${first\_name}]'.
- ELEMENT PROPERTIES:** The right panel shows the configuration for the selected 'ColumnSet'. The 'Data context' field is highlighted with a red box and contains the value `${contacts}`. Other properties include 'Only show when' (not set), 'Id' (not set), 'Initially visible' (checked), 'Minimum height in pixels' (not set), and 'Style' (not set).

**Bottom Screenshot: TextBlock Properties**

- CARD STRUCTURE:** The tree view shows the 'TextBlock [\${first\_name}]' element selected and highlighted in blue.
- ELEMENT PROPERTIES:** The right panel shows the configuration for the selected 'TextBlock'. The 'Text' field is highlighted with a red box and contains the placeholder `${first_name}`. Other properties include 'Data context' (not set), 'Only show when' (not set), 'Id' (not set), and 'Initially visible' (checked).

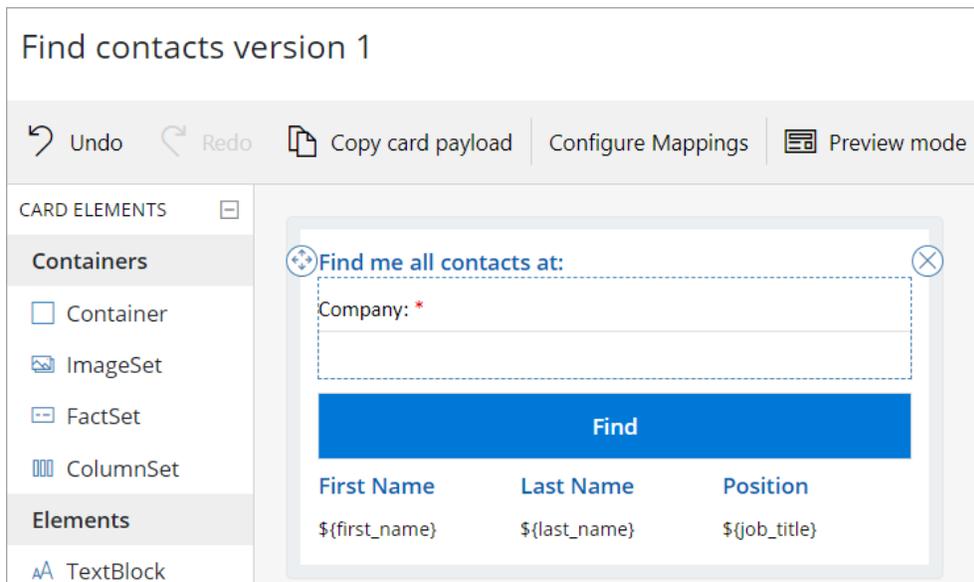
*Example showing data context set on the parent, and placeholder added to the child*



Example placeholders for the above sample data

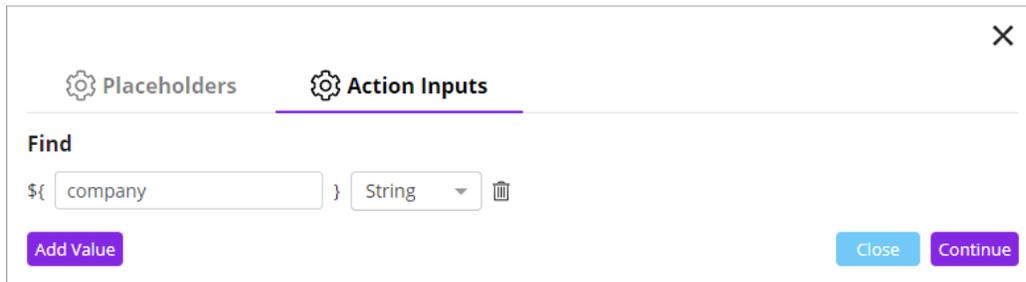
## Configuring action inputs in adaptive cards

You need to configure input elements in the adaptive card as action inputs using the **Configure Mappings** option.



Example adaptive card

In the example card shown above, there is an input element, `Input.Text`, to let end users enter a company name that will be used in a search. The input must be passed to the action that is invoked by the card's Find button (configured in the card as a submit action). You must configure this input as an *action input*.



*Configuring the action input for the Find button*

To define an input element as an action input for use in CE Studio:

1. Click **Configure Mappings** on the adaptive card menu bar.
- 2) Go to the **Action Inputs** tab.
- 3) Click **Add Value**.
- 4) Enter the Id of the input element and select the data type.
- 5) Click **Continue**.

This makes the input available to CE Studio when configuring CE Studio actions.

## Adding an adaptive card to a CE Studio apps

You can add adaptive cards as elements inside any top-level component.

To add an adaptive card to a CE Studio app:

1. In a form or other top-level component, go to an empty layout cell. Adaptive cards are added as elements to a form or other top-level component.
2. Click **+** in the empty cell and select **Adaptive Card**.
3. Select a card from the list.
4. Select the version - you can select any of the card versions. One of the versions is the default - the default is set in the Card Catalog.

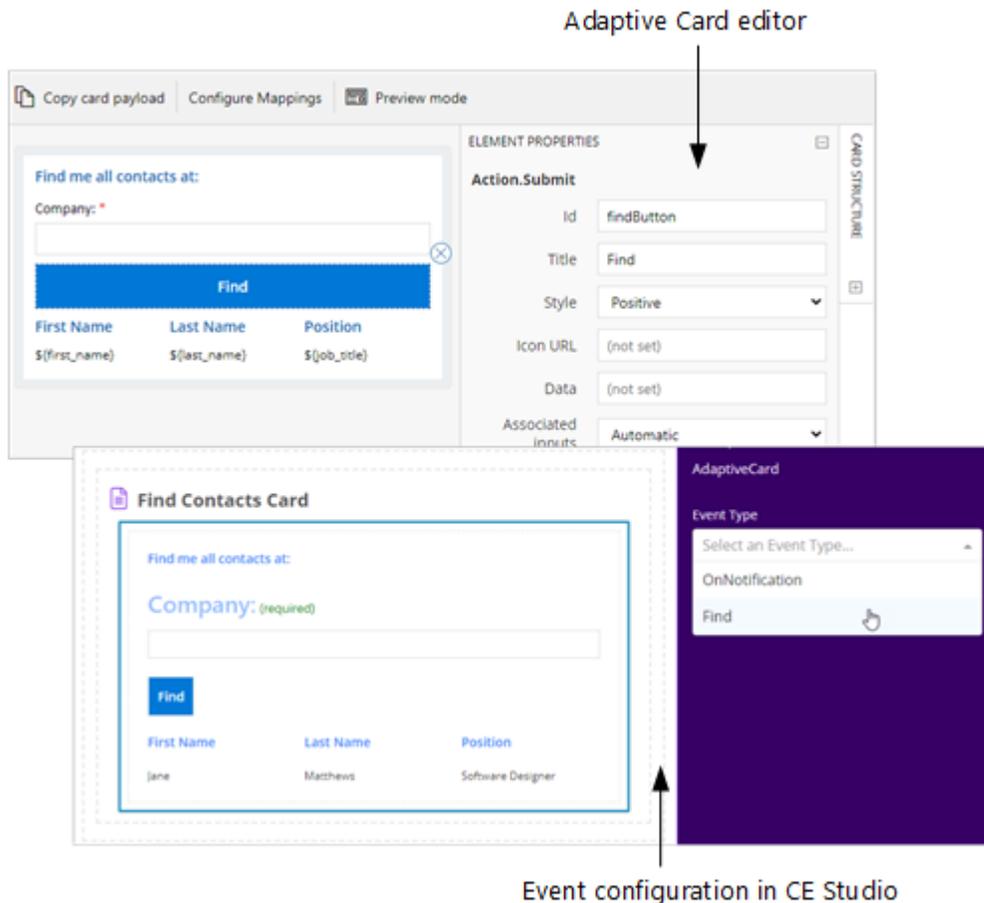
5. The dialog lists all the placeholders in the card. You can map the placeholders to:
- A provider property
  - A message text that contains a placeholder, with the form {0}
  - A value from a field configured in the CE Studio app
  - A static value
  - A toolbar query parameter

## Configuring actions and events for adaptive cards

You configure actions for adaptive cards in the usual way. For adaptive cards that contain action inputs, you need to add a filter to the action. The condition in the filter maps the action input as an event context. For details of action inputs, see [Configuring action inputs in adaptive cards](#).

*Example of a filter for an action input*

Each submit action in an adaptive card will generate a custom event type. For example, a submit action with the title *Find* generates a *Find* event type. You can configure the custom event type to invoke the operations required by the adaptive card. The custom event type can run an action set or rule set, with or without a display task set.



*Example of a submit action and its custom event type*

### Creating an action for an action input in an adaptive card

1. Create an action set and add an action to the set.
2. Enter the service type, provider type and provider method.
3. If this operation returns response data then select the option to publish the topic.
4. Go to the **Filters** tab and click **Add Condition**.
5. Click the left side of the condition and, for example, select **Field Reference**.
6. On the right side of the condition, select **Event Context**. Select **[title].AdaptiveCard** as the event context element, where **[title]** is the Id of the CE Studio component containing the adaptive card.

The action input(s) are then listed in the **Properties** column.

7. Select the action input.

### Configuring events for the submit action in an adaptive card

You can use OnNotification and generated, custom event types for adaptive cards. Other event types, such as OnLoad, can also be configured on the component that contains the card.

1. Go to the Event Configuration page.
2. Click the adaptive card and select the custom event type. The event type name is the title that you gave to the submit action in the Microsoft Adaptive Card editor. You can configure the custom event type to run an action set or a rule set, with or without a display task set.
3. Click the adaptive card, select the OnNotification event type if you need to subscribe the card to topic data.

### Copying adaptive cards between tenants

You can copy adaptive cards between tenants but some additional configuration is needed:

1. In the Card Catalog, edit the card you want to copy.
2. Click **Copy card payload**.
3. Create a new card on the target tenant.
4. Click **Clear Card**.
5. In the **Card Payload Editor** tab, paste in the card payload.
6. Reconfigure the mappings.
7. For all the elements in the card, check that the element properties are set correctly. Note that not all properties are copied as part of the card payload, for example, the Data Context.

### Troubleshooting adaptive cards

We recommend that you always develop and test your adaptive card in the adaptive card editor using sample data. Only then should you add the adaptive card to a CE Studio app.

#### Field title is displayed when there is no data in the adaptive card

Adaptive Cards template language provides built-in functions for conditional layout. To hide an element when there is no data available use the `$when` conditional property. CE topic data is always an array so the `$when` condition can be used in combination with the built-in `count()` function to check the length of the data array is greater than 0. This will ensure any data placeholders will be hidden while no data is available: (see [Adaptive Cards Template Language - Adaptive Cards](#)).

For example:

```
{
  "type": "FactSet",
  "$data": "${contacts}",
  "$when": "${count($root.contacts) > 0}",
```

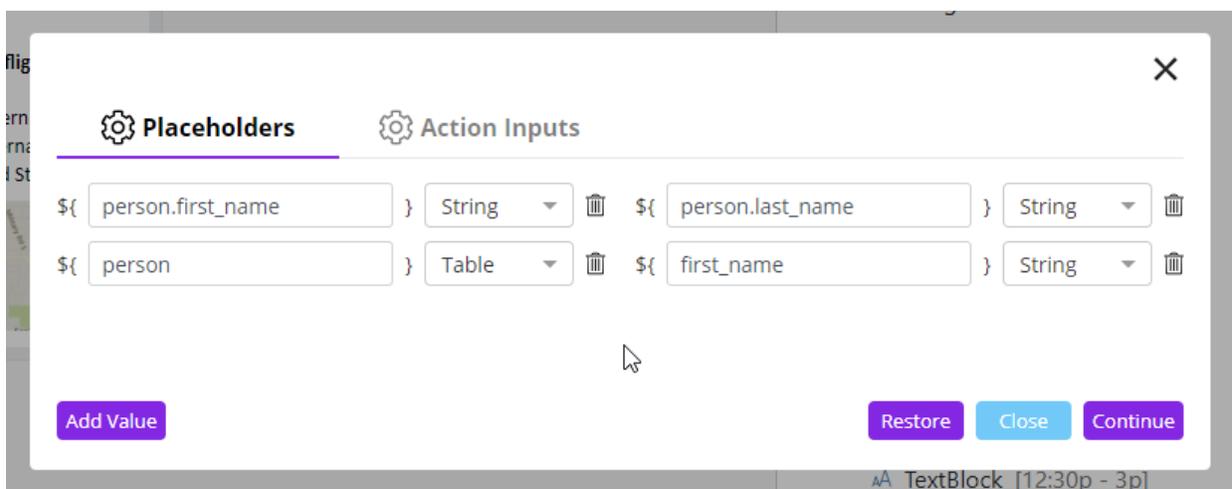
```

    "facts": [
      {
        "title": "First Name",
        "value": "${first_name}"
      },
      {
        "title": "Last Name",
        "value": "${last_name}"
      }
    ]
  }

```

When I map the placeholders in the card to provider properties I see both root and my table name

This is because one or more of the placeholders in the adaptive card are at root level in the card structure. You can see this in the Configure Mappings dialog:



CE topic data is always an array. In the example given above, `person.first_name` is the correct mapping for the adaptive card and `first_name` should be deleted as this placeholder is at root level. In CE Studio Designer, the latter will appear at root level (**\$root**) when configuring the adaptive card.

### Images and styles do not render on the Preview page

Images and styles in the adaptive card that require data when rendered are only shown when you preview the adaptive card.

### Adaptive card fails to render on the Preview page

When an adaptive card is not displayed on the Preview page, you can use the Developer tools for the browser to understand why. For example, in Chrome, go to the Console tab in the Developer tools and look for any error messages.

## Calendars and planners

There are two components for selecting dates:

Simple Calendar	Use Simple Calendar for selecting a single date. You can then configure the click event to run an action or rule to populate another component such as a List View element. The Last-Mile Customer Portal app uses Simple Calendar in this way to show available appointments for a given date.
Planner	<p>Use Planner when you want to show available appointments for a week or month, and let users select and book an appointment. You can configure Planner to show additional information for appointments.</p> <div style="background-color: #f0f0f0; padding: 5px;"> <p><b>Note</b> Business hours are preset to 8.00am to 5.00pm - this will be changed in a future release.</p> </div>

You can use both of these elements with IFS Planning and Scheduling Optimization.

### Simple Calendar

The simple calendar component displays pre-defined dates which can be viewed through a calendar picker. The first available date is automatically selected for the user.

You can configure the active view to either month or year. It also allows you to integrate scheduling information by mapping calendar dates to provider properties. Based on the requirement, you can enable or disable specific dates by either a condition or manually. You can style the foreground and background colors of disabled and enabled dates. The number of active dates shown in the calendar depend on the page size of the action.

#### Adding a Simple Calendar

1. Add a simple calendar as a top-level component or add it as an element within an existing component.
2. On the **Calendar Settings** tab, select the features needed for the calendar.

**Calendar-Picker**

**Calendar Settings** Style

[Configure Calendar Picker Element](#)

Identifier:  Active View:

Calendar Types:  Width (px):

Mapping:     Navigate  Visible Date Range

Min Date:  Max Date:

Preview:

APR	September 2023						Today
MAY	SU	MO	TU	WE	TH	FR	SA
JUN							
JUL						1	2
AUG	3	4	5	6	7	8	9
Sep	10	11	12	13	14	15	16
Oct	17	18	19	20	21	22	23
Nov	24	25	26	27	28	29	30
Dec 2024							

## Mapping the Providers

Calendar data can be supplied by any provider with date properties:

1. Click the **Configure Calendar Picker Element** button.

**Calendar-Picker**

**Calendar Settings** Style

[Configure Calendar Picker Element](#)

Identifier:  Active View:

Calendar Types:  Width (px):

Mapping:     Navigate  Visible Date Range

Disable & Enable Dates

By Condition

By Manual

Preview:

APR	January 2024						Today
MAY	SU	MO	TU	WE	TH	FR	SA
JUN							
JUL	1	2	3	4	5	6	
AUG	7	8	9	10	11	12	13
SEP	14	15	16	17	18	19	20
OCT	21	22	23	24	25	26	27
NOV	28	29	30	31	1	2	3
DEC	4	5	6	7	8	9	10

2. In the field selector, select the provider and then select the **provider properties** in the usual way. These must be date properties.
3. On the **Calendar Settings** tab, select the provider property from the Mapping list. Click the **+** icon if you need to add a second property. For example, to configure an action that uses a click event on a simple calendar (event context), you need an ID field.

**Calendar-Picker**

**Calendar Settings** Style

[Configure Calendar Picker Element](#)

Identifier:  Active View:

Calendar Types:  Width (px):

Preview:

APR	January 2024						Today
MAY	SU	MO	TU	WE	TH	FR	SA
JUN							
JUL	1	2	3	4	5	6	

## Enabling Dates using the By Condition Option

You can enable specific dates by configuring one or more conditions. For example, enable dates in the current month and show other dates as disabled.

1. To add a condition, tick the **By Condition** check box.
2. Click the **Add Condition** button and create a new

Condition Name	Format Type	Text Color	Background
Add Condition			

condition.

3. Map the right side of the condition to a static value, a system variable or to the `CalendarPickerElementProperty`.

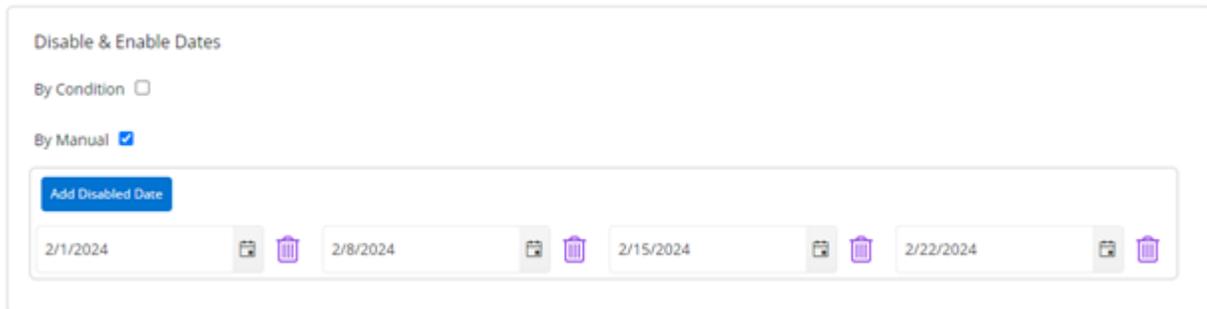
**Note** `CalendarPickerElementProperty` allows you to populate data from any provider to a list view element. Example: If there is a requirement to display the respective time slots for a specific booking date. Once you click a specific date on the calendar picker, the respective times available will be displayed.

4. Style the Text and background colors to show the enabled dates.
5. Save the condition.

## Disabling dates using the By Manual option

You can prevent users from selecting specific dates by adding them manually.

1. Select the **By Manual** check box and then click **Add Disabled Date**.



2. Select each disabled date. These dates will display in a slightly lighter color in the Preview pane. You can style this further on the **Style** tab.

### Styling Active and Inactive Dates in the Simple Calendar

Dates in a *Simple Calendar* have two states, active and inactive. You can style the background and text color for each state.

1. On the **Style** tab.
2. Navigate to the **Style** section.
3. Select the Calendar State you want to style.
4. Select suitable formatting by changing the background and text colors.

### Adding Actions and Events To the App

Add an action to provide the required calendar data for Simple Calendar, selecting the Publish Topic check box. The number of active dates shown in the calendar depends on the page size of the action, which is 10 by default.

On the Events Configuration page, configure an event to run the action. The Simple Calendar must subscribe to the OnNotification event type and this action.

### Configuring OnDateSelectedEvents for Simple Calendar

Use the **OnDateSelected** event to control what happens when a user selects an active date in the calendar.

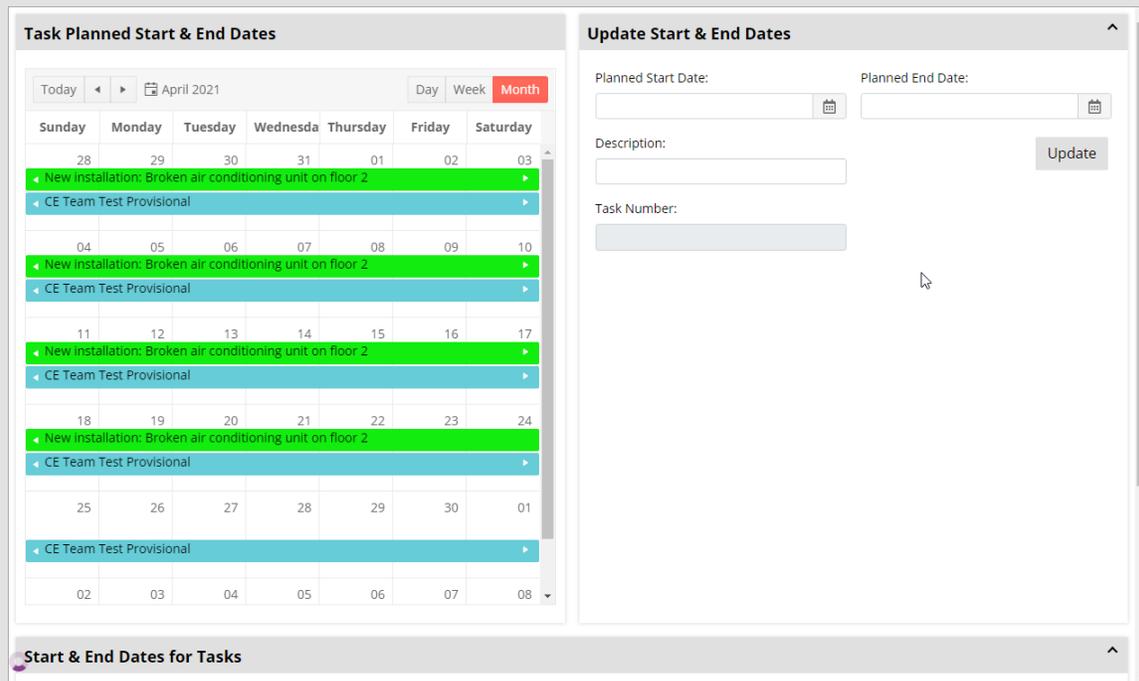
1. Configure an action that uses a click event on a calendar. For example, in the condition in the action, map the left side to a Field Reference and the right side to the Event Context.
2. On the **Events Configuration** page, click the Simple Calendar and select the **OnDateSelected** event type.
3. Select the rule set or action set to run.

## Planners

**Note** Previously called the Calendar component and renamed as Planner after the introduction of the Simple Calendar component, which is an alternative calendar.

The Planner component displays calendar events in a day, week or month view. The calendar events can be the datetime properties from a provider type or appointment data from IFS Planning and Scheduling Optimization. The style of the events can depend on one or more string properties, for example, the event type, event status and so on. Users can easily navigate between dates and switch between showing the full day or just business hours.

**Note** You can download an example that demonstrates two different ways of using Calendars and Planners in CE Studio and how to configure them. This is available from the Central Template Repository which is part of CE Studio Designer. These demonstrate.



Calendar data can be supplied by any provider with date and time properties:

Data	Mandatory	Description
start datetime	Yes	Provider must have a date property that supplies values for the start datetime.

Data	Mandatory	Description
end datetime	Yes	Provider must have a date property that supplies values for the end datetime.
detail text	No	Any string that provides additional information that you want to display in the event.  <b>Note</b> You can set a default for this when you configure the rules for styling the events.

You require a separate component or element for creating the calendar events.

### Adding a planner

1. Add a planner as a top-level component or add it as an element within an existing component.
2. Click **Configure Planner**.
3. Click  to select a provider and select the properties in the usual way.

You must add properties that will provide start and end datetimes.

4. In the Calendar dialog, map the properties that you selected from the provider to the fields in the Calendar dialog. You must map **Start Time** and **End Time** to date properties. A description property is optional.
5. Select which view buttons to show in the app, and which view is the default.
6. Save the app.

### Styling the events

**Note** You can add conditions to style events based on the description property but not on date properties.

1. Click  to open the planner component.
2. Go to the **Format** tab.
3. Configure a condition for each rule as shown in the following example. If there is no detail text for an event, then you can enter default text to be shown instead.

### Creating components for adding calendar events

You can use almost any component to create calendar events, for example, users could select a datetime from a data grid or enter the details in a form.

**Note** You can configure datetime fields in forms to either show or hide the time. To enable users to set the time in a date field, click  on the toolbar for the field and, on the **Display** tab, select the **Include Time** option.

You need to add actions to load calendar events into the calendar component and to create an event from the entered details. You may also want to configure an OnSuccess event to update the calendar component with the new event.

**Note** Users must enter all date parts using 2-digits, that is DD, MM. The create or update will fail if users enter 1 instead of 01.

### Refreshing planner components

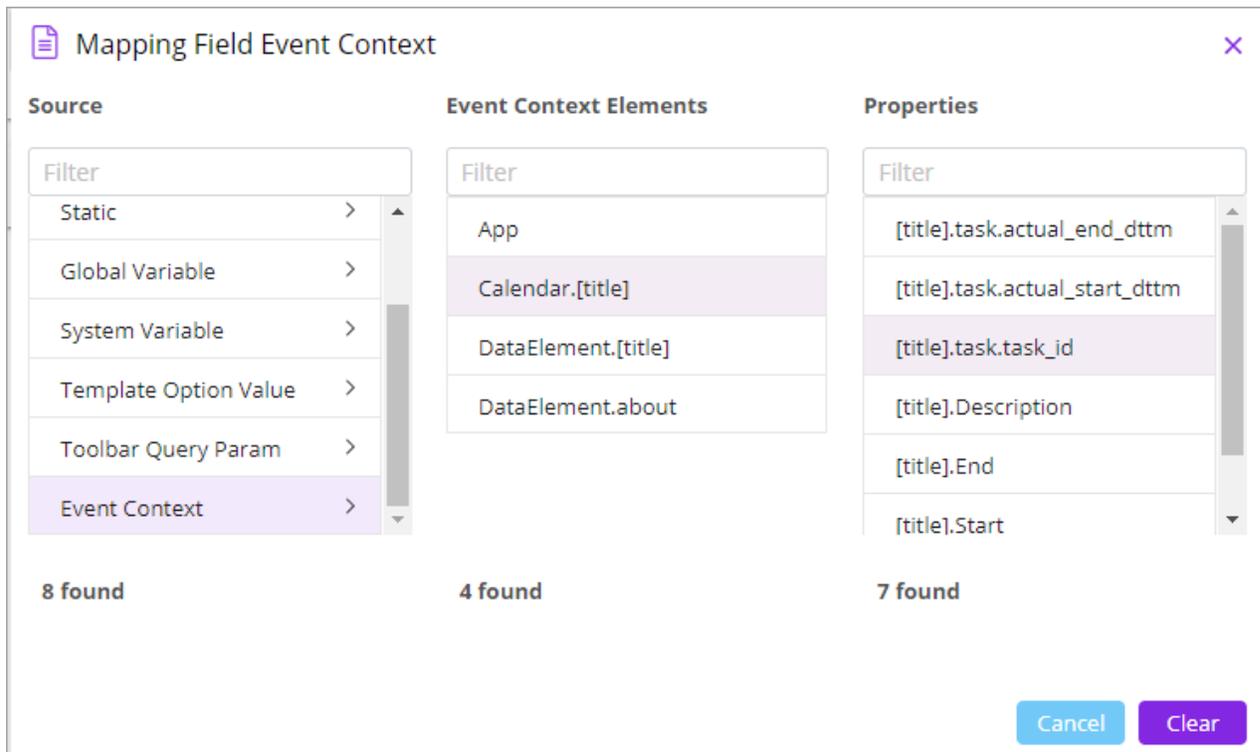
For details, see [Using timers](#).

## Selecting calendar events

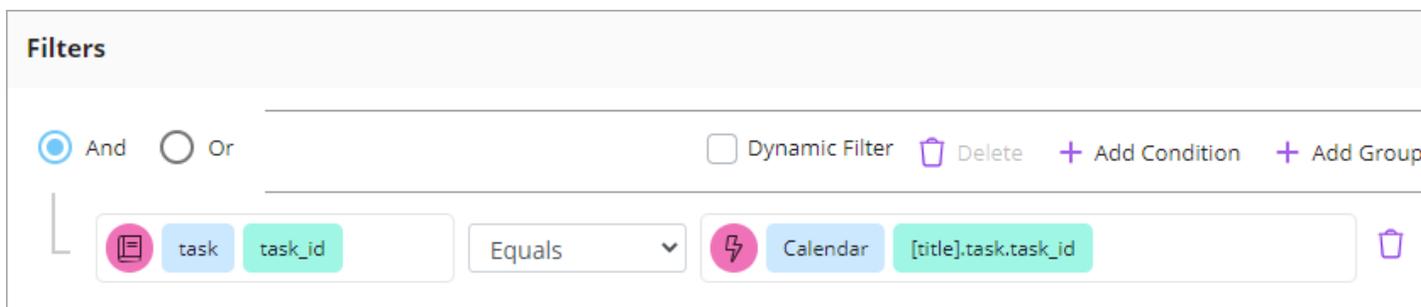
You can configure an action to use the calendar event selected by the user. For this you use the OnEventClick event type. The event will then run the selected action set and the action will be filtered using the data from the event context.

You need to set up the filter on the action:

1. In the **Filters** pane of the action, add a new condition.
2. Map the left side of the condition to the field reference.
3. Map the right side of the condition to the event context, for example, like this:



The condition should look similar to this:



## Captcha

---

Customer Engagement uses Google reCAPTCHA. reCAPTCHA uses an advanced risk analysis engine and adaptive challenges to keep malicious software from engaging in abusive activities on your website. The Captcha element protects only the component that you add it to and the actions invoked by onClick events for that component. It does not protect all the components in the CE Studio app.

You need to register a site on Google reCAPTCHA, add the site details and then configure Captcha element in CE Studio:

- In IFS Customer Engagement add the site in the Admin Portal (on the **Studio> Manage CAPTCHA** page).

### Adding a CAPTCHA element to a CE Studio app

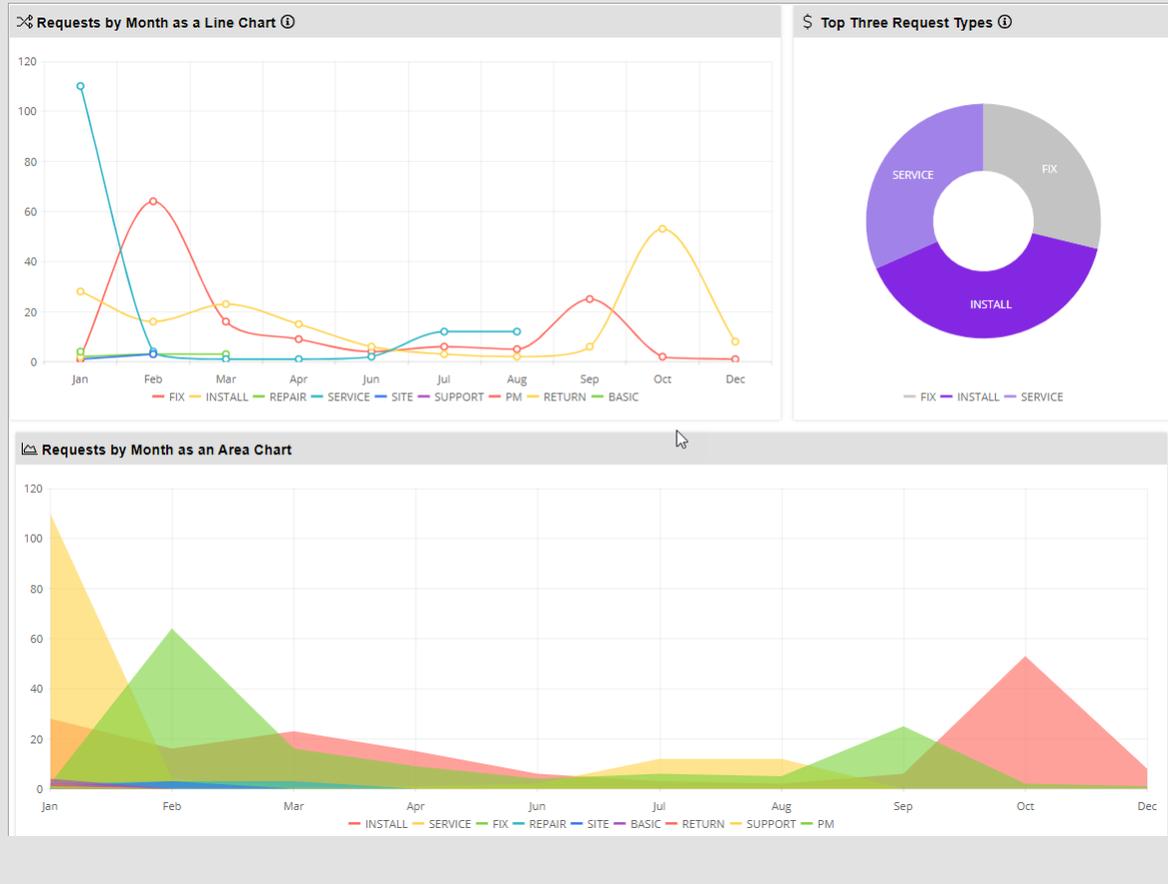
1. Add a Captcha element to a component. The Captcha element protects only the component that contains it. You can add:
  - One scorebased element
  - A checkbox or badge element
  - Multiple checkbox and badge elements
2. Fill in the necessary details. Ensure to select the suitable Captcha type and site. If adding both checkbox and badge elements then you need a site key that is configured for both the element types.
3. Click **Continue**.
4. Save the CE Studio app to confirm the configuration settings.

## Charts

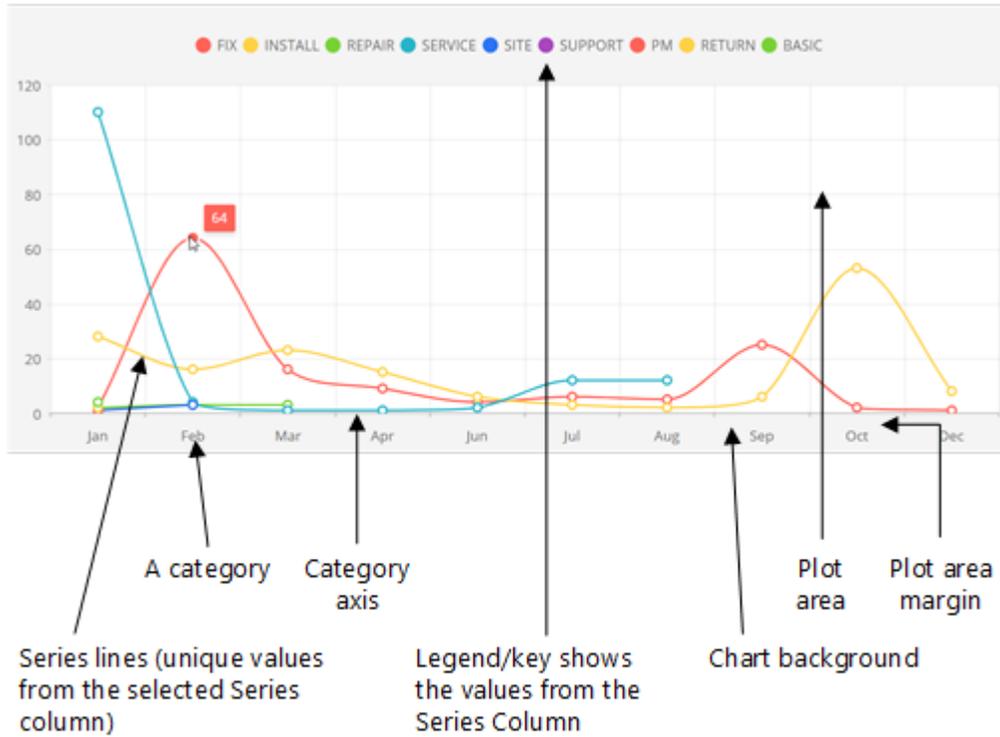
---

You can add charts to CE Studio apps to help users visualize their data. All the features of the chart and how the features are displayed are configurable.

**Note** You can download an example that demonstrates the chart types available in CE Studio and how to configure them. This is available from the Central Template Repository which is part of CE Studio Designer.

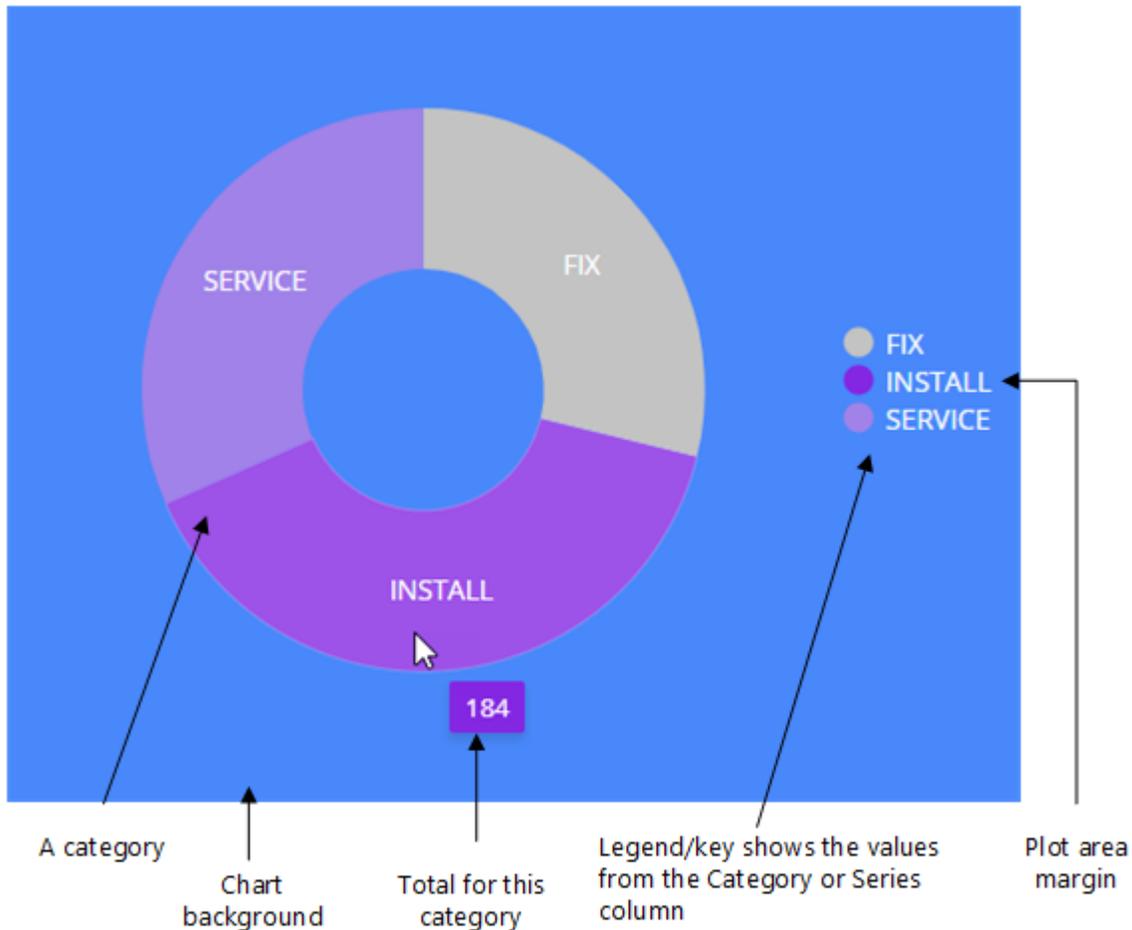


All areas of the charts are highly configurable.



Configurable features of charts in CE Studio

In addition to line charts and area charts, you can also use pie and donut charts:



*Configurable features of pie and donut charts*

You can experiment with the chart configuration using the chart preview area. The chart preview area shows dummy data, which is represented by the red line. You use the preview area to experiment with the styling, and try out dynamic features such as animation, panning and the crosshairs.

**Important** You may need to prepare the data before you can chart it. For example, clean the data to remove rows with missing values by using filters on actions, and group or aggregate the data by using [data transformations](#).

### About series and categories

A *category* corresponds to a column in a spreadsheet or database table. The categories form the category axis on line and area charts, and the segments on pie and donut charts.

The *series* corresponds to the row in a spreadsheet or database table. Depending on the shape of your data, the series column is one of the following:

- A property representing a data series (grouping). Each unique value from this column is plotted as a separate line or area on the chart. You do not require properties for the **Category** or **Value** columns.
- A property to be plotted (y-axis) that you want to plot against a category (x-axis) using values from a third property. You require properties for both the **Category** and **Value** columns.

### Actions for chart components

You will need to add an action to get the chart data.

**Note** Set a suitable page size on actions intended for use with charts. The page size configured for the action will set the maximum number of records displayed in the chart.

### Step 1 - Adding a chart component

**Note** Before starting, it can be helpful to verify the data that you want to chart. To do this you can display it in a data grid. This will show you the range of values in your data, whether there are missing values and whether your data transformations are working as intended.

IFS Customer Engagement only: if using the IFS CE Reports provider then check in the Admin Portal that the report is scheduled and when it last ran.

You can either add a chart as a top-level component or you can add it as an element inside an existing component. Choose the chart type before starting to configure the chart.

To add a chart:

1. In an empty cell of the layout grid, click  and then select **Chart**.
2. Select the chart type from the left toolbar:



The configuration varies slightly depending on the chosen chart type. However at any point you can switch chart type by selecting from this toolbar.

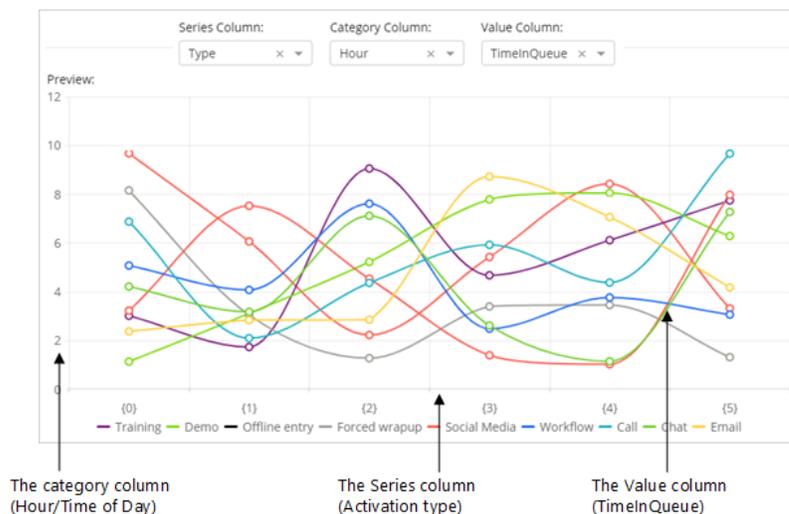
## Step 2 - Adding the categories and data series

**Note** All the chart types use categories. Only line and area charts use a Series column.

1. Click **Configure Chart** to add the properties that you want to the chart. Repeat this step if you need to add and remove categories later.
  - a. Click  to select the data provider.
  - b. Filter by the type that will provide the data for the chart.
  - c. Add the properties.

For line and area charts, it is can be easier to add the properties in the reverse order. For example, if you want the category axis to read 2000, 2001, 2002 then add the properties in the order 2002, 2001, 2000. Alternatively, you can reorder a small number of categories by dragging and dropping.

2. Depending on the chart type, select the series, category and value columns:



### *The Category, Series and Value Columns in a Line Chart*

3. By default, the categories and series are represented as {0}, {1} and so on. In order to style these, enter the values that you expect them to have. Entering the values lets you select a color and add default text (or a message ID) for each one of them on the **Category Axis** and **Series** tabs.

**Note** IFS Customer Engagement only: It is not currently possible to style data returned by a series that is a single property. This applies in particular to properties of the IFS CE Real Time Statistics provider.

Depending on the chart type, each category or series value has a default style and is added to the legend. This is not shown in the chart preview but is shown when you preview the app.

### Adding a chart title and changing the chart style

On the **Chart Settings** tab of the chart component, you can set the following options:

- A chart title — press the `Enter` key to add the title to the chart
- Enable Animations — whether users can pan across the chart or zoom in to see the detail (but note that the animations will run each time the chart is refreshed)
- Format the area covered by the chart and the surrounding background behind the legend and chart title

### Configure the category axis for line and area charts

On the **Category Axis** tab, you can configure the horizontal axis of line and area charts or make the axis invisible if you do not want to use this.

**Note** As part of the category axis configuration, you can choose to display a vertical line (crosshair) that appears when you move the pointer over the chart.

### Refreshing chart components

For details, see [Using timers](#).

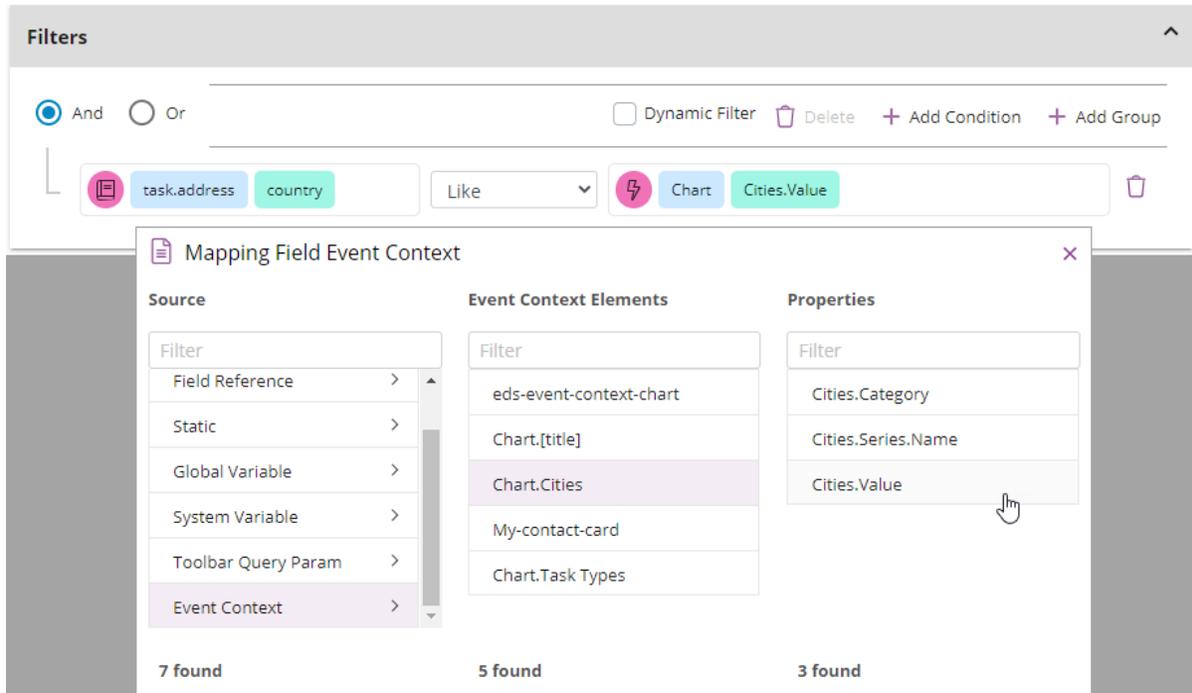
**Note** If you have configured animations then the animations run each time the chart is refreshed.

### About the event context and the `OnSeriesClicked` event

You can also configure the `OnSeriesClicked` event for charts.

Charts (like data elements) can generate an event context. For example, clicking a line in a line graph or a segment in a pie chart could update a data grid to show details based on what you click. This click is the event context. In the example below, the righthand side of the action filter must map to an event context which provides the value (city name) required in the condition. The action runs when the chart is clicked (the `OnSeriesClicked` event type):

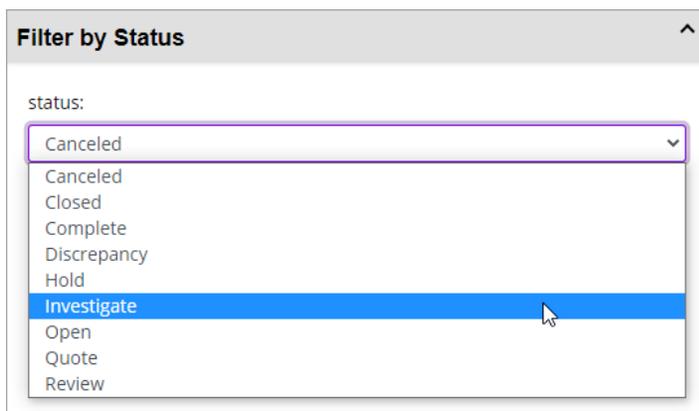
*Example of an action configured to use the Event Context source type*



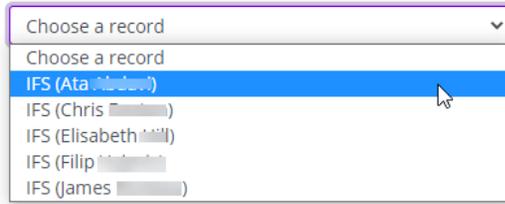
**Note** The form, in the above example, must be configured for an OnNotification event and must subscribe to the above action.

## Combo boxes

Combo boxes display a list of values from a provider lookup code or provider properties – values are always supplied by the provider. Combo boxes that display a list of provider property values require an action to get the values and an OnNotification event to subscribe to them.



*Displaying values from a lookup code*



*Displaying values from a record (company name, first name, last name)*

**Note** Use the OnNotification event to subscribe to the action that provides the data. Use the OnValueChanged event to configure what happens when users select from the list.

### Create a combo box to show a lookup code

1. Add a combo box UI element to a form (or convert a text box into a combo box).
2. Configure the **Provider Id** field to get the current value, for example from the selected row in a data grid.

The element containing the combo box must subscribe to an action that provides this value.

3. Configure the lookup for the combo box. The lookup provides the range of values displayed in the combo box:
  - a. Click **Configure**  on the combo box.
  - b. On the **Data Source** tab, in **Select Source**, select **Lookup**.
  - c. In **Lookup Provider Id** click **+** and then select the provider that will provide the values.
 

The selected lookup code is shown in the **Lookup Code Name** field.
  - d. If the values in the combo box depend on a value selected in another combo box, then enter the qualified name of that combo box in **Depends on Qualified Name**.
  - e. Click **Save**.

This type of combo box does not need to subscribe in order to get the values in the list.

### Configure a combo box to show one or more provider properties

1. Create a combo box as explained above.
2. Configure the **Provider Id** field to get the current value, for example from the selected row in a data grid. You can configure a second provider id that will be used when the first one doesn't return a value.

The element containing the combo box must subscribe to an action that provides this value.

3. Click **Configure**  on the combo box.
4. On the **Data Source** tab, in **Select Source**, select **Provider**.

You can only select one provider.

- Click **Configure** and then select one or more provider properties to display in the combo box. These can be child or navigation properties.

After clicking **Done**, the selected properties are added to the **Value** list. You can now configure these as display text. You will map each to a placeholder {0}, {1}, and so on:

- In **Display text data**, select the first value.
- Click + and then select the next value(s).

**Note** To delete the provider configuration, delete these mappings.

- In **Display Text**, enter the placeholders. The first selected value is mapped to the first placeholder which is {0}. You can also enter additional text, such as punctuation. To enter a default that is used when there is no value, enter ?? and then the default to use, such as {??None}.

The screenshot shows a configuration window for a combo box. It has the following sections:

- Select Source:** A dropdown menu currently showing 'Provider'.
- Configure:** A purple button.
- Value:** A dropdown menu showing 'contact\_sequence'.
- Display text Data:** A list of two items:
  - 'last\_name' with a delete icon (x) and an add icon (+).
  - 'company\_name' with a delete icon (x) and an add icon (+).
- Display Text:** A text input field containing the placeholder string '{??None} {{0} {1}}'.

- On the **Events** page, configure the **OnNotification** event for the combo box.

### Configuring the **OnValueChanged** event type

Configure events for the combo box:

- On the **Event Configuration** page, click the combo box.
- From the **Event Type** list, select the **OnNotification** event.
 

The combo box must subscribe to the action that gets the provider providers.
- Click the combo box and select the **OnValueChanged** event type.
- Select the action set, rule set or display task set that will run when users select from the combo box.

### Disable or Enable Drop down in combo boxes

The drop down functionality of combo boxes can be configured to either enable or disable as and when it is required through a display task.

**Note** To permanently disable the combo box drop down navigate to **Configure > Validation** and check the Read-only option. You can enable this option back again by removing the check on the Read-only option.

## Data elements

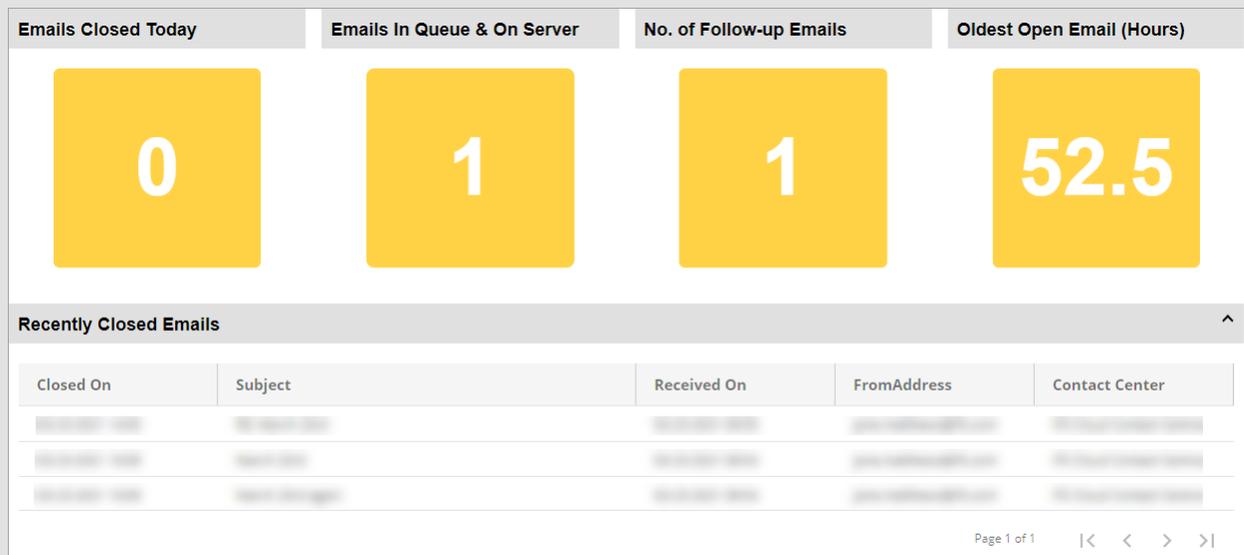
---

Use data elements to create wallboards and IFS Cloud-style lobby pages. Each data element can show:

- Text, either fixed or mapped to a property from a provider.
- Optional header and footer text, either fixed or mapped to a property from a provider.
- Optional, fixed background image that displays behind the text.
- Optional foreground and background colors, or background image, that changes depending on the mapped property.

You can add data elements as top-level components. This means that you can configure the data elements for events such as OnLoad, OnNotification. Data elements added as elements of a component support OnClick events only.

**Note** You can download an example that demonstrates how you can use data elements in CE Studio. This is available from the Central Template Repository which is part of CE Studio Designer.



### Adding a data element

#### 1. To add a data element:

- As a top-level component, click  in an empty cell of an app or template.
- As an element inside an existing component, add a row or column to the component and then click  in an empty cell.

#### 2. Click **Data Element**.

#### 3. To make use of data from a provider, click **Configure Data Element** and then select the properties and key(s):

- Click  to select a provider and search for the properties. For details, see [Adding a data grid](#).
- Click  to use the [Metadata Selector](#).

The keys are required if you intend to configure an event context on this data. This is when a click on the data element runs an action that is filtered by the data in the data element.

#### 4. Enter the text to show in the data element:

- You can type in fixed text or click  and select the ID of the message that you want to use.
- If you do not want a header or footer, delete the placeholder text.

- Optionally, map the header, content and footer to one or more of the properties that you configured earlier:
  - Use {0} as a placeholder for the first value from the mapped property.
  - You can insert placeholders anywhere in the header, content or footer text.
  - If you map multiple properties then the subsequent placeholders will be {1}, {2} and so on.
  - You can surround the placeholders with text, such as Created by {0} on {1}.
  - If you want to set a default value then prefix the value you want to use with :??

Note that the data element always displays the first match that it finds.

- Go to the **Styles** tab to set the font size, corner rounding, default background and foreground colors.

**Note** When entering a color, you can switch between HSLA and Hex by clicking the format selector button.

- Save the data element.

After saving the data element, you can configure conditional formatting.

### Using conditional formatting

You can add multiple conditions to change the foreground and background color of the data element based on a property value. Currently it is not possible to conditionally set the background color if you also have a background image.

**Configure Data Element** ×

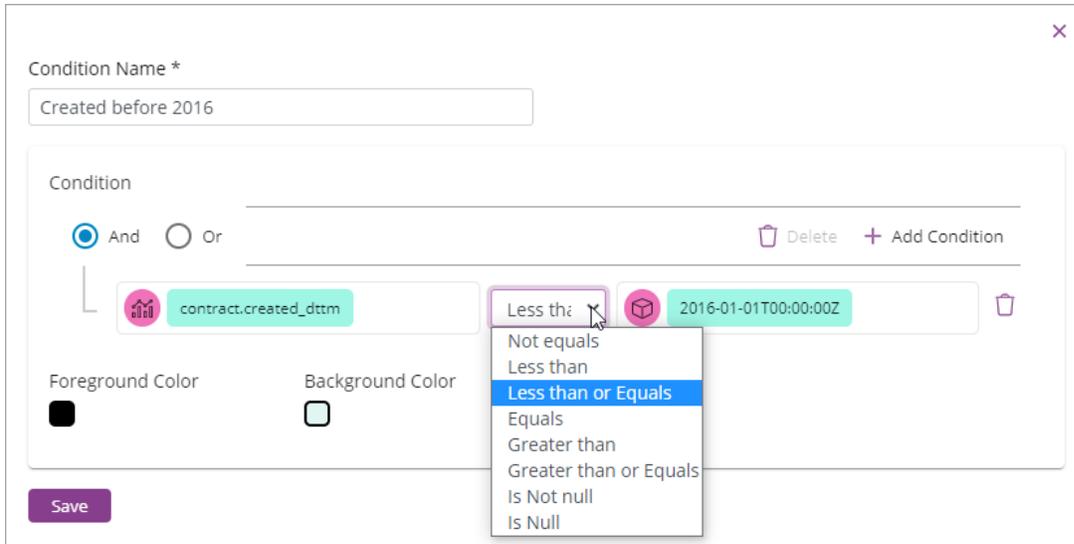
⚙ Settings
🗨 Styles
🗨 Format

Condition	Apply To	Foreground	Background	
Pre-2016 contract	Data Element	■	□	🗑 ✎
Post-2016 contract	Data Element	■	□	🗑 ✎

New Condition

Cancel
Continue

*Example of conditional formatting based on created date*



*Adding a condition to format a data element based on the date*

### Event configuration for data elements

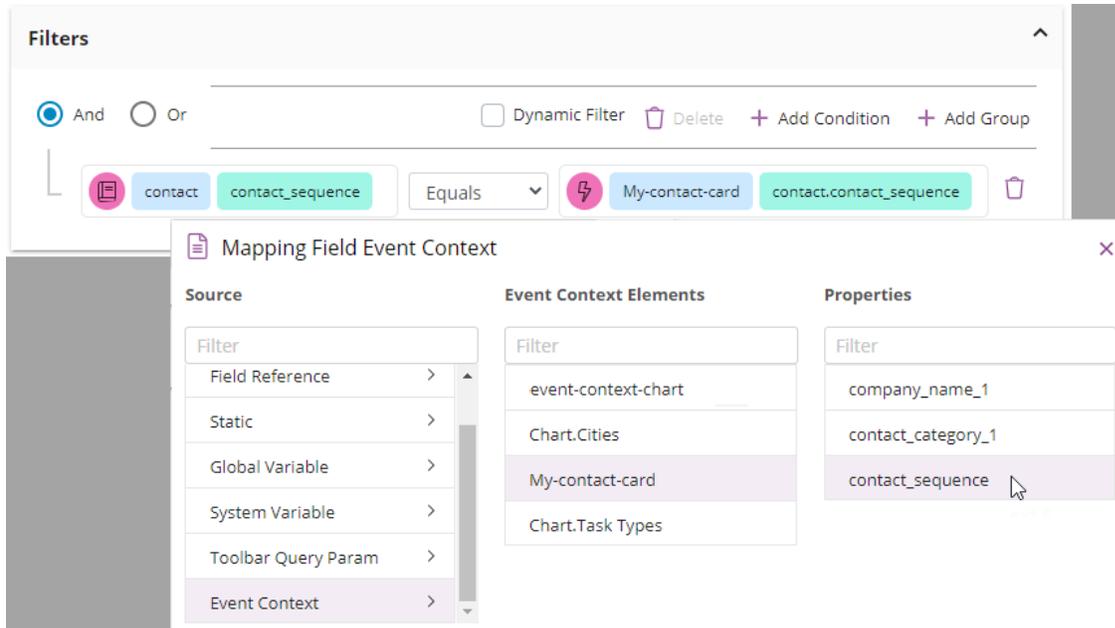
Configuration	Event type
Data elements added as components	Any event type that can be configured for components, such as OnLoad, OnNotification, OnTimerElapsed.  A data element can subscribe to any number of topic subscriptions. The subscription will populate all the data elements within the component.
Data elements added as elements	You can only configure OnClick events. For example, to run a rule set, action set or display task set.

### About the event context

Data elements (like charts) can generate an event context. For example:

1. Clicking a row in a data grid could update a data element.
2. Clicking the data element could then update a form to show more of the data from the data element.

This second click is the event context. In this example, the righthand side of the action filter must map to an event context which provides the ID or key to get the required data, as shown below:



*Example of an action using the Event Context source type*

**Note** The form, in the above example, must be configured for an OnNotification event and must subscribe to the above action.

## File attachments

You can use the File Upload component to upload files. The way in which you can handle file attachments depends on the data type.

### File attachments on IFS Cloud

You can use the File Upload component in media or portal apps to upload files. The way in which you can handle file attachments depends on the data type. You use the ceStream data type for IFS Applications 10/IFS Cloud

For example:

	CeStagedFile data type	ceStream data type
File upload	Not applicable	<p>Add the Upload File element to a component.</p> <p><b>Select:</b></p> <ul style="list-style-type: none"> <li>• Allow Multiple Files (optional)</li> <li>• Use Stream Upload (required)</li> </ul>
File download	Use the List Template element with the utility component Attachment Download. See below for details of how to configure these.	
View downloaded document	Not applicable	<p>End users click the View Document button.</p> <p>This is automatically configured when you add a property with the ceStream data type.</p> <p>For example, adding the FileData property of the EdmFileForFileOperation provider type to a form will automatically configure the file download for you.</p> <p>Note: This is part of the DocumentRevisionsHandling projection.</p>

**Note** There is also a UI Element called FileDownload. You can add this to forms, and map it to a property of the ceStream data type.

For further information on file attachments, see the example file upload template which is available from the Central Template Repository.

### Configuring the List Template element and the Attachment Download component

You use the List Template element and the Attachment Download component to download files. See [List Template and Attachment Download components](#).

## File attachments for media apps and FSM

You can use the File Upload component in media or portal apps to upload files. The way in which you can handle file attachments depends on the data type. You use the `CeStagedFile` data type for these providers:

- IFS CE Email provider
- IFS Field Service Management

**Note** To test file upload with the `CeStagedFile` data type, you need to load the app in Agent Desktop as an ocuid is required.

For example:

	<code>CeStagedFile</code> data type	<code>ceStream</code> data type
File upload	<p>Add the Upload File element to a component.</p> <p>Do <b>NOT</b> select:</p> <ul style="list-style-type: none"> <li>• Use Stream Upload</li> </ul> <p>Whether you can select <b>Allow Multiple Files</b> depends on the provider. For example, the IFS CE Email provider supports multiple attachments but the IFS Field Service Management provider does not.</p> <p><b>Important</b> Test file upload in Agent Desktop not in the Designer preview.</p>	Not applicable
File download	Use the List Template element with the utility component Attachment Download. See below for details of how to configure these.	
View downloaded document	End users open files from the download folder.	Not applicable

**Note** There is also a UI Element called `FileDownload`. You can add this to forms, and map it to a property of the `ceStream` data type. You cannot map this to a property of the `CeStagedFile` data type.

For further information on file attachments, see:

- The standard email template that is available on all tenants
- The example attachments template for IFS Field Service Management. This is available from the Central Template Repository.

### Configuring the List Template element and the Attachment Download component

You use the List Template element and the Attachment Download component to download files. See [List Template and Attachment Download components](#) for details.

## List Template and Attachment Download components

**Note** For background information [File attachments](#).

You use the List Template element and the Attachment Download component to download files:

1. Add a List Template element and configure it:
  - a. In **Templates**, select **Attachment Template**.
  - b. Select the properties that define the mime type, filename and file size.
2. Add an Attachment Download component from the Utility Components area of the screen:
  - a. Click **Configure**.
  - b. Map the **Base64 Content** field to `provider > UploadedFile > BodyBase64`.
  - c. Similarly, map the **File Name**, **Mime Type** and **Size** fields to suitable `UploadedFile` properties.
3. Configure an action to get the record with the file attachment.
4. Both the List Template element and the Attachment Download component subscribe to this action.

## Email attachments

This topic explains how the standard IFS CE Email template is configured for email attachments. You use the email provider type `UploadedFile`.

Attachments are automatically saved once the email is successfully sent.

### Add an element to list the attachments

On the Designer page:

1. Click **Add**  and select **List Template**.

You need to add the List Template as the inner element of a component. For example, you can add a list template to a form component.

2. From the **Templates** list, select **Attachment Template**.
3. Map the fields you require — for example:
  - a. Set the Attachment ID to the `AttachmentId` property on the `Email` type.
  - b. Set the File Name, Mime Type and File Size to the corresponding properties on the `InboundFile` type.

The maximum file size is 10mb.

### Add an element to upload attachments

1. In the Upload Files dialog, specify the file extensions that you want to support and the maximum file size of each attachment.
2. For an email app, you must select **Allow Multiple Files**.

### Add an action to download email attachments

This action lets the agent view the contents of the file attachment. The downloaded file is an instance of the property type `UploadedFile`.

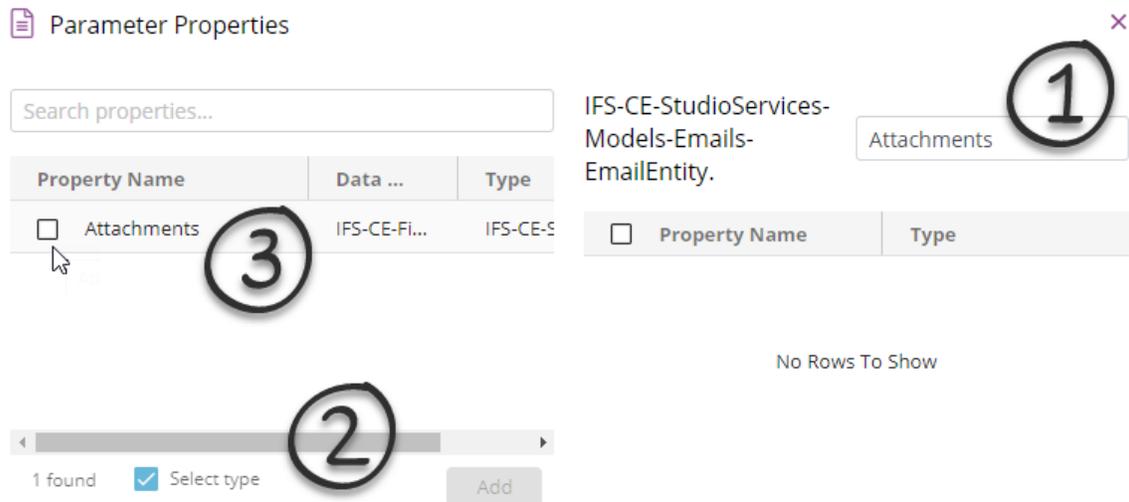
On the Action Sets page, add an action:

1. Select the property type `UploadedFile`.
2. Select the `get` method.
3. There is one required parameter for the email attachment ID. Map this to the field in the incoming email that has the attachments.
4. Publish the topic.

### Add an action to send email with an attachment

On the Action Sets page, add an action:

1. Select the property type `OutboundEmail`.
2. Select the `sendemail` method.
3. Click **Manage Properties**.
4. Map all the fields that are in the component for email replies. Map the file attachment to the `IFS_CE_StudioServices-Models-Emails-EmailEntity` property *Attachments*:



## Image elements

The image element is a container for one or more images. These can be any of the following:

Web images	Supports web URLs using https not http. You can enter either a static URL or the image element can subscribe to OnNotification events that return the image.
Base64 images	Supported base64 image types are: jpeg, png, gif and webp. An example of an IFS Cloud provider that uses Base64 images is the projection <code>EquipmentObjectStructureNavigationHandling</code> .
Binary images	A binary image is an image stream (the CE data type <code>CeStream</code> ). At this release, CE only supports image streams from IFS Cloud providers. When the image source is mapped to a field that has the data type <code>CeStream</code> then the CE Studio app fetches the image binary and converts it to display the image. An example of an IFS Cloud provider that uses binary images is the projection <code>DocumentRevisionsHandling</code> .

You add an Image element in the usual way, and then add one or more images and configure each image separately. For details of how to add a component, see [Adding components and selecting properties](#).

## Configuring image elements

Field	Description
Image Group Identifier	Uniquely identifies all the images as a group. You can change this when configuring any image within the group. You use this to identify images from the group when configuring the click event for the image.
Image Identifier	Uniquely identifies an image within the group. You can change this when configuring the image. Use this to identify images from the group when configuring the click event for the image.
Width (px), Height (px)	Enter the image width and height as a numeric value.
Source	<p>The default image to display when there is no dynamic image supplied by a provider. Enter:</p> <ul style="list-style-type: none"> <li>An https web URL. Image URLs using http will not load.</li> </ul> <p>A preview is shown if the URL is valid. If the URL is not reachable or invalid, then the preview panel shows the default 'broken image' icon. The Alt Text (if configured) is shown beneath.</p> <p>The dynamic image to display. Enter:</p> <ul style="list-style-type: none"> <li>A placeholder, such as {0} for the first placeholder, {1} for the second placeholder and so on. You then map each placeholder to a provider property in the <b>Source Mapping</b> field.</li> </ul> <p>The default image is shown until the placeholders are substituted with data values. If the resulting image URL is invalid, the broken image icon and any alternative text is shown.</p>

Field	Description
Source Mapping	<p>If required, you can map the image to a provider property.</p> <ol style="list-style-type: none"> <li>1. Click the <b>Configure Image Element</b> button and use the property picker to select the properties that will provide the image.</li> <li>2. The selected properties are listed in <b>Source Mapping</b>. The first property you select is mapped to the first placeholder {0}.</li> <li>3. If there are multiple placeholders and properties then click the + to configure the next placeholder.</li> </ol>
Alt Text, Alt Text Mapping	<p>Optionally, enter the alternative (Alt) text for the image. This is displayed in the CE Studio app if the image fails to load. You can also specify placeholder mappings for the alternative text.</p>
Keys for Event Context	<p>Images in the image element group can have OnClick and OnNotification events attached. The data mapped to the Image element becomes available, for example, becoming the value of an action filter or an input parameter for an action or rule.</p> <p>To configure the keys:</p> <ol style="list-style-type: none"> <li>1. Click the <b>Configure Image Element</b> button and use the property picker to select the properties required.</li> <li>2. The <b>Keys for Event Context</b> list shows all the selected fields. You can add one or many of these as keys. Click the + to add each additional key.</li> </ol>

## Event configuration for image elements

**Important** You must configure the OnNotification event for the image element itself and not on the component that contains the image element. This insures that the image element displays the dynamic image rather than the default image.

You can use the OnClick events to run actions, rules and display tasks. To pass data to the action, rule or display task, you need to configure one or more keys as explained above. This then becomes available when configuring the event context in the action filter. For details, see [Actions and action sets](#) and the **Event Context** source type.

You can configure OnClick and OnNotification events for the images in an image element on the Event Configuration page.

## Label elements

---

Configuring label elements are similar to configuring [data elements](#) but in addition, you can control how the date and time values are formatted in the placeholder. For example:

Your appointment is on {0:MM-dd-yyyy} at {0:HH:mm}.

In this example, the placeholder is {0}. The first use displays and formats the date and the second use displays and formats the time.

## List View Element

---

The List View Element is a structured display of data entries presented in a list format. It can be configured manually or dynamically and can be linked with other elements to show information derived from interactions with the list view.

### Adding a List View Element to a CE Studio App

1. As a first step, make sure to create a provider for the list view element as per the requirement.
2. Add a list view element as a top-level component or add it as an element within an existing component.
3. The **Configure List View Element** window will then be displayed as soon as you add the element.

**Configure List View Element**

**Settings**

**Configure List View Element**

Type: Manual

Orientation: Horizontal

**Add List Item**

### Configuring the List View Element Manually

A List View Element configured manually is a fixed list of items defined by yourself.

1. Select the manual option from the Type drop down.
2. Click on the **Add List Item** button to populate each list item.
3. Add the necessary static text you wish to display on the list item.
4. Optionally, you can navigate to the Styles tab and format your list items as you preferred. Click **Continue** to save.

### Configuring the List View Element Dynamically

A List View Element configured dynamically, shows items from a provider.

1. Select the dynamic option from the Type drop down.
2. Click on the **Configure List View Element** button.
3. Select the relevant provider properties and fields to display in the list view element.
4. Select the mapping fields from the mapping drop down.
5. Provide the list item text and ensure that the value mapped is in curly brackets Ex. First Name : {0}, {1}.
6. On the Event Context drop down, select the relevant provider properties to display in the List View Element. You also need to select the key(s) for this provider type.

- Optionally, you can navigate to the Styles tab and format your list items as you preferred.
- Click **Continue** to save.

### Actions and Events List View Element

You need to configure actions and events for List View Element. Create an action to get the data for the list view element. Optionally, you can also add an element to display the data reflected by the click event on a list view. On the Events page, configure the necessary events to run the action. The list view element must subscribe to this.

**Note** For actions that run when users click on a list view item, map the right side of the action filter as follows:

- From Data Source, select Event Context.
- Select the list view element.
- Select the property GetSelectedListViewItem

dynamic type (based on the key selected) it will be `List.ViewElement.table name.field name`

### Disable the list items on list view element

You can refer to this section, if there is a requirement of disabling a list item (non-clickable).

**Note** Only item from the list can be configured for this requirement

- Open the **Configure List View Element**.
- On the Format tab, navigate to the **Add Condition** button.
- Give a suitable condition name.
- Provide the required condition to disable the list items matching the condition.
- Select the Format Type as Disable from the drop down.

6. You can also format disabled list view items by selecting a text and background color.
7. This overrides the Disabled style set on the Styles tab.

## Overlay components for modals

---

You can use overlay components for modal windows and dialogs. Overlay components can contain any combination of elements and UI elements. You can add as many overlay components as needed.

To configure an overlay component:

1. In **Designer**, under Utility Components, click  and then select **Overlay**. Utility components have a default name that can't be edited.
2. Overlays are added as forms.
  - Click  on the component toolbar to select provider properties and add UI elements.
  - Click  on the component toolbar to add other element types, such as calendars, buttons groups, iFrames and so on.
3. Add a display task that toggles the overlay component on or off. See [Display tasks](#).
4. In **Event Configuration**, set the event type(s) that run the display task to show and hide the overlay component.

The display task can also be attached to an action set or rule set. See [Event configuration for components](#).

5. When the overlay component is toggled on, it must be populated with data (for example). It must therefore listen for an [OnNotification event](#) and subscribe to an action that will provide the data. The action that gets the data must have **Publish Topic** set.

## Radio buttons

---

Radio buttons are elements that allow users to select a single option from a set of mutually exclusive choices. You can either configure the radio buttons manually, by lookup (requires a provider) or with provider properties. You can also configure radio buttons with dynamic label data if you require to get data from a provider.

### Adding a Radio Button to a CE Studio App

1. As a first step, make sure to create a **provider** for the radio button element as per the requirement.
2. Add a radio button as a top-level component or add it as an element within an existing component.
3. As a default setting, you are provided with three radio buttons.

## 4. Click



### Configure.

5. Add the necessary details and click the **Save** button.
6. Radio buttons can either be configured through **Lookup**, **Provider**, or **Manual** source.

### Configure a Radio Button Manually

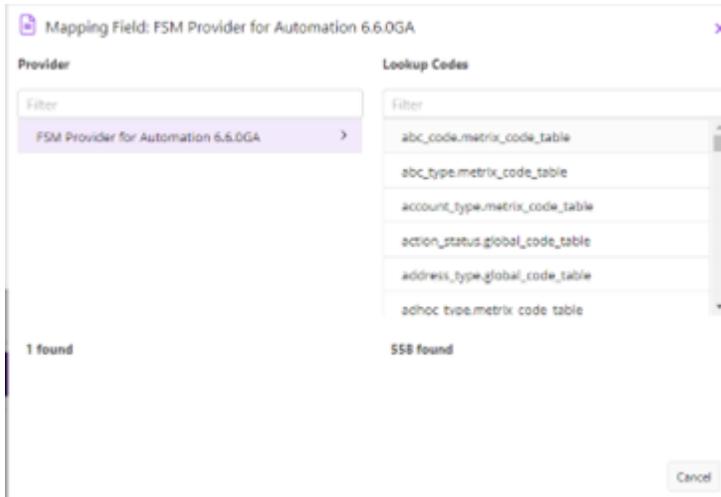
1. On the **Data Source** tab, in Select Source, select **Manual**.

2. Click the **Add Radio Options** button.
3. Provide the Label and Value for each radio button . The Label is the name of the option appearing on the form and the Value is the reference for internal use.(For Example: Label = Highly Satisfactory and its reference is denoted by its value and hence Value = 1).

### Configure a Radio Button to display a Lookup Code

Lookup codes are provided by a provider. You do not need to configure an action for these.

1. Create a radio button as shown above.
2. On the **Data Source** tab, in **Select Source**, select **Lookup**.
3. In the **Lookup Provider ID** field, click +.
4. Select the required **Lookup Codes**, and save.



**Note** This type of radio button does not need to be subscribed to get the values.

### Configure a Radio Button to show one or more Provider Properties.

1. Create a radio button as explained above.
2. On the **Data Source** tab, in **Select Source**, select **Provider**.
3. Click the **Configure** button and then select one or more provider properties to display as radio button options.
4. In the **Value** list, select the Identifier for the provider type.
5. In **Display Text Data** list, select the properties to display next to the radio button. Click + to add additional properties.
6. In the **Display Text field**, map each display text to a placeholder {0}, {1}, and so on.

### Actions and Events for Radio Buttons

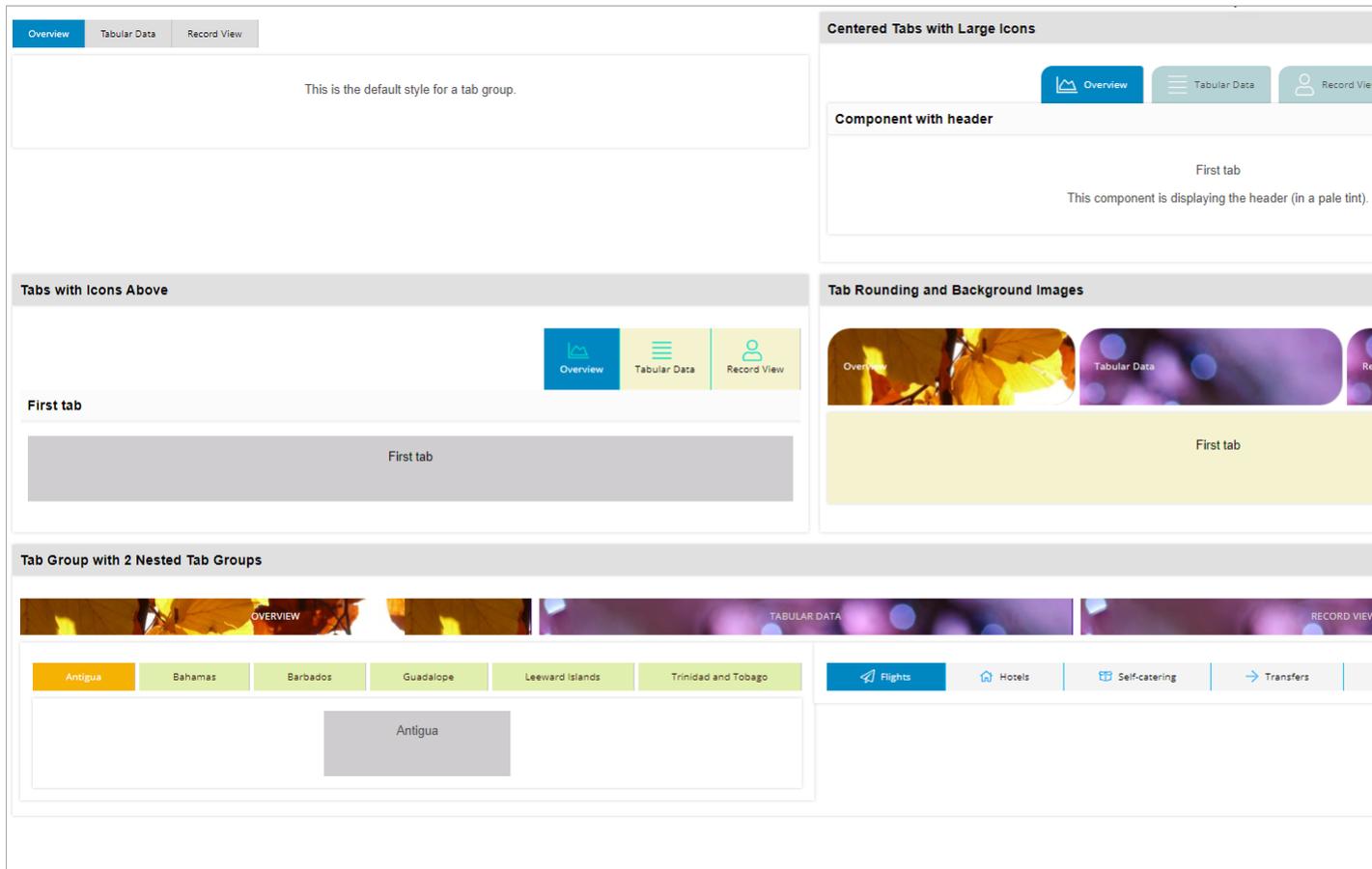
You need to configure an **action and event** for radio buttons that use provider properties.

1. Create an action to get the data for the radio buttons.
2. On the Events page, configure the necessary events to run the action. The radio button element must subscribe to this.

**Note** Use the **OnValueChanged** event to configure what happens when users select a radio button.

## Tab groups

You can add tab groups as top-level components to CE Studio apps. The example template shown below shows the default styling for tab groups, how you can style tab groups and nest one tab group inside another.



*Example template for tab groups*

**Note** You can download the sample template from the Central Template Repository, which is part of CE Studio Designer.

### Display tasks for tab groups

Use the **Active** display type to:

- Bring the target tab to the front of the tab group
- For the default tab, when used with an OnLoad event, brings the data into the tab.

## Events for tab groups

Use the OnTabSelected event for a tab in a tab group. The event is triggered when users click the tab or when a display task makes the tab active.

## Using timers

---

Use a timer when you need to:

- Refresh an app and its components to display the latest data.
- Leave a CE Studio app open for an extended period of time, such as 30 minutes.

You can configure multiple timers with different timer settings. For example, you can configure the timer interval as part of the:

- App settings
- Component settings 
- Element settings 

You can then start the timer using either of the following:

- **Auto Enable Refresh Timer** option in the app, component or element settings.
- A display task set containing one or more Enable Timer display tasks.

The OnTimerElapsed event determines what happens when the timer reaches the timer interval.

## Performance implications of timers

Using a timer has a performance cost. For this reason only configure the minimum number of timers required by the CE Studio app.

Optimize all queries (actions with filters) that run when a timer expires. This might mean configuration at the queue progression stage. For example, when using timers in an email app, group emails by conversation at the queue progression stage. Don't do this in the CE Studio app.

Take care to avoid duplicate timers. For example, if you decide that you don't need a component that has a configured timer then be sure to disable the timer. Timers continue run even when the component is hidden.

## Adding timers

You can configure multiple timers as part of the app, component or element settings. For example, to configure a timer for a data grid, click  on the element toolbar and then go to the **Settings** tab.

Option	Description
Timer Interval (in secs)	<p>By default, the timer runs for 60 seconds before it triggers the OnTimerElapsed event. The timer interval can be as small as 5 seconds but up to 1 hour.</p> <p><b>Note</b> When setting the timer interval, consider the number of timers in the app and the number of Agent Desktop users who will use the app.</p>
Timer Position	<p>Displays a small spinner if there is any delay while the action runs (optional). The position of the spinner is relative to the item on which the timer is configured (that is the app, component or element).</p> <p>Select <b>None</b> if you do not want to use a spinner.</p> <p><b>Note</b> This applies to actions only. You will not see a timer for display tasks.</p>

### Starting the timers

You can start the timers in one of two ways:

- In the app, component or element settings, select the **Enable Auto Refresh Timer** check box to automatically start the timer.
- Add a display task set containing one or more Enable Timer display tasks. You then configure an event, typically the OnLoad event for the app, to run the display task set which will start the timer(s).

**Note** You cannot start the timer as part of its OnTimerElapsed event.

Timers run continuously until they are paused. At each timer interval, OnTimerElapsed events are triggered.

### What happens when the timer elapses

Regardless of how a timer was started, a timer will trigger an OnTimerElapsed event at the configured timer interval. You can then configure the OnTimerElapsed event to run a rule, an action and/or display task set.

## Pausing timers

To pause a timer, for example to prevent a chart from updating, you can configure a Disable Timer display task. The timer remains disabled until you enable it again.

### Example

In this example, an app contains two data grids. One grid refreshes every five seconds and the other grid every 60 seconds.

1. On the Settings tab of each data grid () , the timer interval is set to the required number of seconds.
2. On the Display Task Sets page, there is a *display task set* to start all the timers in the app. There is an Enable Timer display task configured for each data grid.
3. On the Event Configuration page:
  - The OnLoad event for the app is configured to run:
    - Display task set to start all the timers.
    - Action set to get the data for both data grids.
  - For each data grid, the OnTimerElapsed event is configured to run an action to get the data again.

## Configuring a data grid for bulk updates and deletes

---

You can enable multi-row select on a data grid so that users can select multiple items to work on.

**Note** You cannot enable multi-row select on readonly data grids, such as a data grid configured with the IFS CE Reports provider.

Customers					
<input type="checkbox"/>	Name ↓	Customer Category	B2B Customer	Country	Language
<input type="checkbox"/>	██████████	Customer	<input type="checkbox"/>	DK	nl
<input type="checkbox"/>	██████████	Customer	<input type="checkbox"/>	DE	de
<input checked="" type="checkbox"/>	██████████	Prospect	<input type="checkbox"/>	DE	de
<input checked="" type="checkbox"/>	██████████	Prospect	<input type="checkbox"/>	DE	de
<input type="checkbox"/>	██████████	Prospect	<input type="checkbox"/>	DE	de
<input type="checkbox"/>	██████████	Prospect	<input type="checkbox"/>	CA	en
<input type="checkbox"/>	██████████	Customer	<input type="checkbox"/>	DE	de
<input type="checkbox"/>	██████████	Prospect	<input type="checkbox"/>	SE	en
<input checked="" type="checkbox"/>	██████████	Prospect	<input type="checkbox"/>	AT	nl
<input type="checkbox"/>	██████████	Customer	<input type="checkbox"/>	DE	de

3 selected
 Page 4 of 7 |< < > >|

Update Customer Category: 
Update B2B Customer

*Example of a data grid designed for bulk updates*

### Adding input fields for bulk updates on rows in a data grid

To add input fields that can be used to enter the values for bulk updates on rows in a data grid follow these steps. Following these steps ensures that the action(s) have the right context for bulk updates:

1. Click  on the toolbar of the component containing the data grid.
2. Click  to select the provider.
3. Enter the data type to filter on.
4. Select the properties that you want to update and then click **Done**.

The properties are then added as separate elements within the component containing the data grid as shown in the above example.

### Enabling multi-select for a data grid

Multi-select is part of the data grid properties. To enable multi-select:

1. Click  on the toolbar of the data grid.
2. Go to the **Settings** tab.
  - a. Select the **Enabled** check box to enable multi-row select. This has the following options:

Option	Description
Show Selection Count	By default, a count of the number of selected rows is displayed below the data grid. This is the number of selected rows in the data grid not the number of rows on the current page.
Allow Clear All	Displays a <b>Clear All</b> button so that users can deselected selected rows. Useful for data grids with multiple pages.
Select All By Default	Select all rows from the first page of the data grid when it loads or the maximum number of rows as set in the <b>Maximum Row Selection</b> field.
Maximum Row Selection	Enter the total number of rows that can be selected at once (1000 by default) across all the pages of the data grid. No more rows can be selected once the user reaches this limit.  <b>Note</b> You can configure the actions that perform bulk updates and bulk deletes to run in batches.
Key Columns	Use this field to select one or more key properties. This is important for multi-select data grids where selecting a single row may require multiple keys to ensure uniqueness.

**Note** Once you enable multi-row select, you cannot use the OnRowSelected event.

### Configuring display tasks for on success and failure events

**Note** At this release, there is no feedback to tell the user which rows failed to update or were not deleted when an operation fails. The output of the whole batch will be returned.

You should configure display tasks to notify the user about the outcome of a bulk update or bulk delete. Configure:

- A success message when the bulk update or bulk delete succeeds - configure this for the event onSuccess status
- An error message when the bulk update or bulk delete fails - you *must* configure this for the event onFailure status. See below for details.

You can configure the actions that perform bulk updates and bulk deletes to run in batches. You specify the number of batches to use as part of the action set configuration. Configure:

- A success message when a batch operation succeeds - configure this in the action set (see below)
- An error message when a batch update fails - configure this in the action set (see below)

Use the Prompt display type and select the appropriate message type.

### Configuring action sets for bulk update and bulk delete

Configuration options for bulk update and bulk delete are part of the action set properties – you create the actions in the usual way.

1. Create an action set for a bulk update or bulk delete.

If required, select a display task set from the **Attach Display Task Set** list. The selected display task set will run when the action is invoked.

2. The request that's made for a bulk operation can be divided into batches, such as 4 requests of 250 rows each. You configure this on the **Batch Settings** tab:

Field	Description
Continue On Error	Whether the bulk update or bulk delete will continue if it encounters an error: <ul style="list-style-type: none"> <li>• Select this option if you want to ignore any errors and continue the bulk operation. At the end of the operation, the final status is the status of the last batch.</li> <li>• Deselect this option if you want to stop the bulk operation when an error occurs or when an error occurs in one of the batch operations. If any batch or any single row in a batch fails then the status is OnFailure.</li> </ul>
Batch Size	The maximum number of rows in each batch passed to the server (not the number of selected items in the data grid). For example, if <b>Batch Size</b> is set to 250 and there are 1000 selected rows then there will be 4 requests with 250 rows. Defaults to 1.
Batch Success Display Task Set	Select the display task that runs when the bulk update or delete operation succeeds.

Field	Description
Batch Failure Display Task Set	Select the display task that runs when the bulk update or delete fails.

3. Add the action(s) to the action set as needed.

### Event configuration for OnFailure when using the batch settings

If you configure the action set for multiple batches then you *must* attach a display task to the event OnFailure status. The app will error if this is missing and a bulk update or delete stops on error.

1. In Event Configuration , select the outermost component (the app).
2. In **Status**, select OnFailure.
3. Select the display task that you configured for this purpose.

## Display formats for dates and numbers

Studio designers can configure formatting for data grid columns, charts labels, data elements and read only form fields when working with dates, numbers, and decimals. The value to format is represented by the placeholder {0}. Anything entered outside the angle brackets is fixed text that will be applied to all values.

**created\_dttm**

Header



Format (date)

Format Preview: **1/27/2022**

*Display format for a data grid column - the values will display as short dates*

### Field Properties

 Data Source
 Validation
 Display

Is Hidden

Display Format (when read only)



Format Preview: **001234**

Preview

*Display format for a number field to display leading zeros*

Here are some examples of display formats:

Date	{0:d}	Short date (02/21/2022)
	{0:D}	Long date (Monday, 21 February 2022)
	{0:m}	February 21
	{0:MMM}	Feb (as a 3-character month)
Decimal on a decimal property	{0:d}	"1234"
	{0:D6}	"001235"
	{0:n2}	"1,234.00"

**Note** Numbers and dates are localized. You can test their appearance using the URL parameter `_locale`. For example, `_locale=ja-JP`.

For the full range of formatting possibilities, see this github.com page: [https://github.com/clr-format/clr-format/blob/master/test/core/String/formatInvariant\\_should.ts](https://github.com/clr-format/clr-format/blob/master/test/core/String/formatInvariant_should.ts)

## Troubleshooting components and elements

### Invalid entities reported on saving the CE Studio app

When you try to save the CE Studio app, you get an error similar to the following:

The following properties for the given entities are invalid: 'IfsApp-ServiceContractHandling-RecurringServiceProgram|TriggerType'

A component is configured to use the property named in the error message (`TriggerType` in this example). Using the Metadata Viewer, check whether this property exists in the provider version that you are using.

### Data grid or form fails to display topic data

If a component is not displaying any data but the component is subscribed to an action, and you know that the action is successfully returning data (*for example, by looking in Developer Tools*) then it is likely that CE Studio Runtime is unable to match the response to the component.

For example, the action returns data for the IFS Applications entity set `BusinessLeadHandling-BusinessLeadContact` but you have by mistake configured a data grid for `BusinessLeadContactHandling-BusinessLeadContact`.

You can check the fully-qualified name in Developer Tools:

1. In Chrome, open the Developer Tools.
2. In the **Name** column, click the last invoke.
3. Go to the **Preview** tab.
4. Expand `components` and find the component.
5. Expand `properties > 0`.
6. Expand `metadata > columns`.
7. Expand one of the columns. This show you the fully-qualified name.

Name	× Headers	Preview	Response	Initiator	Timing	Cookies
<input type="checkbox"/> ProviderTypes						
<input type="checkbox"/> SsoDetails						
<input type="checkbox"/> Locales						
<input type="checkbox"/> InitialConfig						
<input type="checkbox"/> styles						
<input type="checkbox"/> Regex?page=1&pageSize=100						
<input type="checkbox"/> 279						
<input type="checkbox"/> 9						
<input type="checkbox"/> 279						
<input type="checkbox"/> Types?appld=607&includePro...						
<input type="checkbox"/> versions?page=1&pageSize=5...						

```

▼ properties: [{id: 13726, layoutReference: "d03f0354-86bf-487b-90ef-b...
  ▼ 0: {id: 13726, layoutReference: "d03f0354-86bf-487b-90ef-ba9fd3179...
    id: 13726
    layoutReference: "d03f0354-86bf-487b-90ef-ba9fd3179e43"
  ▼ metadata: {columns: [,...], rowRendering: [,...]}
    ▶ autoRefreshConfiguration: {isTimerEnabled: false, timeIntervalI...
    ▼ columns: [,...]
      ▼ 0: {identifier: "BusinessLeadContact.Name", uiType: "TextBox"
        alias: null
        cellRendering: null
        clrFormat: null
        dataType: "string"
        header: "Name"
        identifier: "BusinessLeadContact.Name"
        isEnum: false
        isHidden: false
        isKey: false
        orderIndex: 0
        pinned: null
        providerId: 45
        qualifiedName: "IfsApp-BusinessLeadContactHandling-Business...
        translation: null

```

*Finding the fully-qualified name*

# 7

## Actions, rules and events

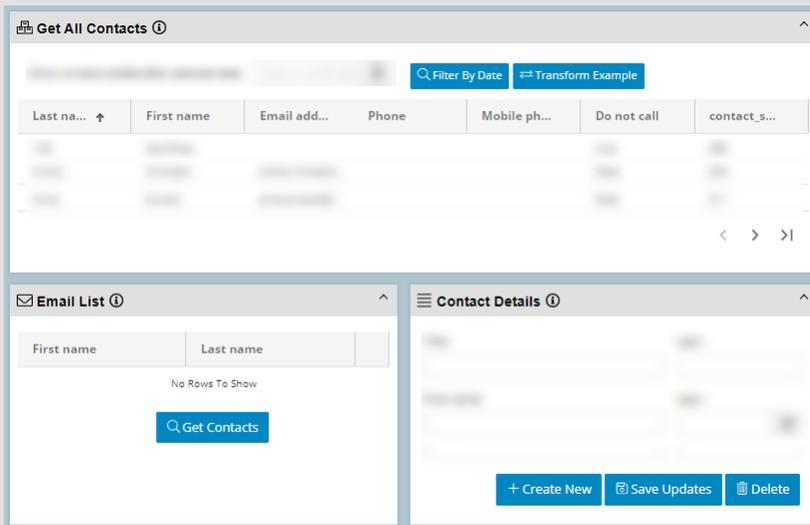
Action sets contain one or more related actions. Each action invokes one of the provider methods. As part of the action, you can configure conditional logic (filters), input and output transformations, a sort order and a page size. Actions are run when specific events occur. You can chain actions so that the response from one action is the input to another action in the same action set.

Rules and rule sets let you define more complex behavior for the template or app. For example, depending on conditional logic, you can update the user interface, run actions and filter the data in the response. At runtime, a rule set is triggered when a specific event and event status occurs. Each event status can trigger a different rule set.

**Note** There are some sample templates available for download from the Central Template Repository that demonstrate the use of actions, rules and events in CE Studio.

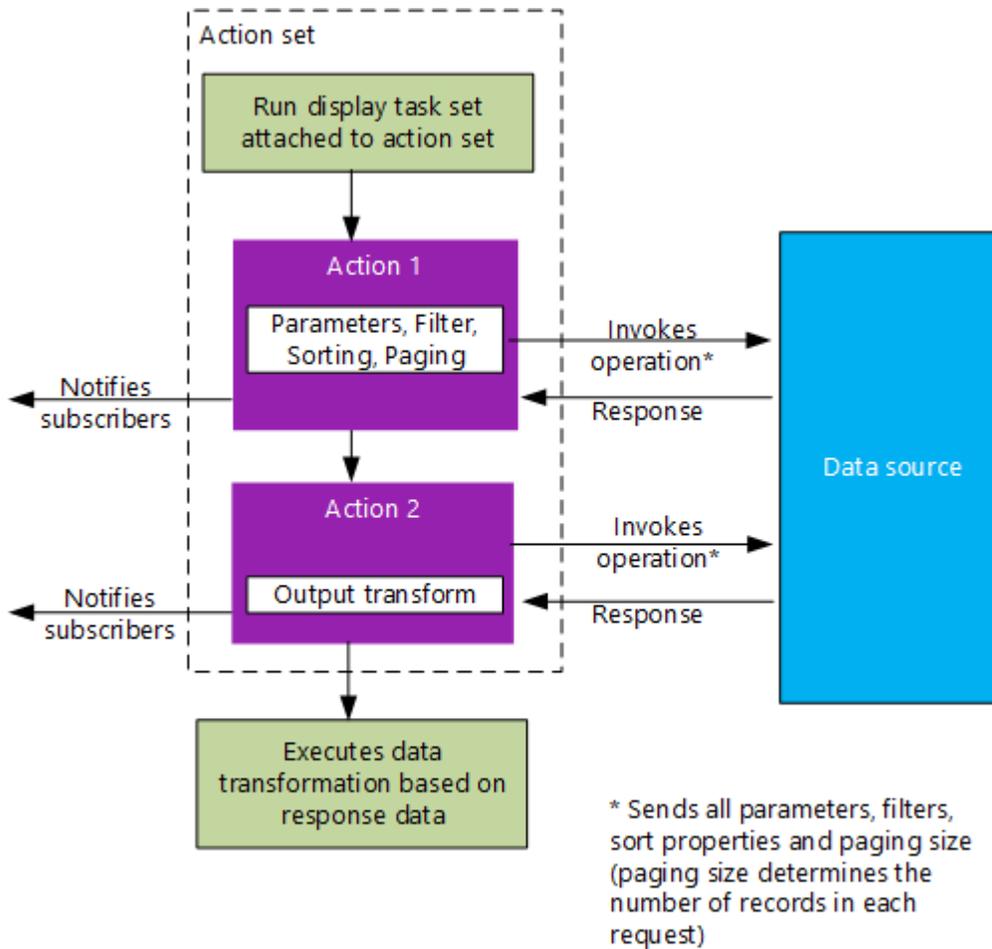
## Actions and action sets

**Note** You can download an example that demonstrates the use of actions from the Central Template Repository which is part of CE Studio Designer:



An action set (see figure below) contains one or more actions that are triggered in the order in which they are listed in the action set. An action invokes one of the provider methods. As part of the action, you can configure input parameters, conditional logic (filters), simple transformations, a sort order and a page size. Note that:

- The display task set attached to the action set is run first, in order to pass in any data required by the actions.
- Input transformations are applied before invoking the provider method.
- Filters, sorting and page size are passed to the endpoint as part of the request
- If the action publishes a topic then the OnNotification event is triggered before the next action in the action set is run.
- The page size determines the amount of data requested, for example, the number of rows to display in a data grid or chart. Going to the next page in a data grid will send another request (invoke) to the endpoint.
- Data transformations configured for the *action set* are applied once all the actions in the action set have run.
- If the provider takes too long to respond to the request (invoke) then the request times out. The default timeout is 30 seconds and this returns an error `Studio Service Rpc 500 error. the operation was canceled`. You can ask IFS Support to increase the timeout to allow more time for the endpoint to respond.



*How the actions in an action set work*

**Note** If one action in an action set fails then all the actions in the action set fail, including any data transformations. This is because the actions and data transformations in the action set are dependent on each other.

### Attaching action sets to events and rules

You can attach an action set to:

- An event and status, such as OnLoad – OnSuccess, OnClick – OnInvoke, OnRowSelected – OnInvoke. We recommend that you invoke the action set as few times as possible. For example, it is better to attach the action set to the OnLoad event for the app rather than attach it to individual components, which will cause the same action set to be invoked multiple times.

**Note** OnNotification events are handled differently. For details of the publish/subscribe model, see [OnNotification events](#).

- A rule. Each rule in a [rule set](#) can trigger one action set, and you can use the rule set to group the action sets that you want to invoke at runtime. You can also use rules to apply conditions to determine when the action sets are run. You then attach the rule set to an event.

You can optionally attach a display task to an action set. This means that the display task will run once the action set has is invoked.

*Example showing how action sets are used in events and*

Created Action Sets

View Items  View Usage

GetActiveEmail 1 1 Add Action + [link](#) [edit](#) [delete](#)

Events:

App - JM Email Template - OnLoad - OnInvoke

CloseEmail 1 1 Add Action + [link](#) [edit](#) [delete](#)

Rules:

Close Email Rule

*rules*

**Note** To understand how action sets are used by the app or template, click **View Usage** on the Actions page. Click to go to the event or rule configuration.

### Creating an action and mapping the parameters

When you add a new action, the first step is to map the parameters to properties in the provider. Once you've configured the parameters and saved the action, the action is attached to the action set.

**Note** You can attach the same action to multiple action sets. You can also detach an action from an action set and attach it to another one. This means that when you delete an action set, you do not delete any of the actions attached to it. Once created, the actions remain in the app. To list all the actions click **All Actions** on the Actions page. You can delete an action from the All Actions page.

To create an action:

1. Click **Add Action** to add a new action to the action set.

**Note** Give the action set and action meaningful names. It is not possible to rename them later once you've configured a transformation on the data set and used the transformed data to configure components.

2. Select the provider, the provider type and the method — types used by the app are in blue because you added these properties to a component:

3. The required parameters for the method are typically displayed in the **Parameter Mapping** area, although for some providers and methods you may need to click **Manage Properties** to add them.
4. For each required parameter, you can:
  - Select the **Use Null** option for properties that are *not* primary keys. This means that you want to set the value in the new or updated record to null (does not exist). However, for an operation to succeed, some parameters may require a value to be set (for example, primary keys) or the external system may be configured to set a default value if one isn't provided. Where this is the case, selecting **Use Null** will cause the operation on the external data source to fail. A typical error message is `Error calling Invoke Studio Service Rpc 500: CEContext required`.
  - Select the **Ignore** option. This means that you do not want to set or update the value. For example, you might use this when updating a record and you only want to update just one of the values.
  - Click the parameter to map it to a value:

Source type	Parameter mapping
Field Value	Map the parameter to a field in one of the app's components. The response from the provider will only return data for the fields configured in the component.

Source type	Parameter mapping
Action Response	<p>Map the parameter to the response from another action in the same action set. The action providing the response must run first. For example, the first action gets an identifier from one provider type and the second action (the chained action) uses the identifier in the response to get data from a different provider type.</p> <p>Note that:</p> <ul style="list-style-type: none"> <li>• The action response always returns a single record.</li> <li>• The action that provides the input for the chained action must publish a topic which must have a subscriber.</li> <li>• The <code>ce context</code> is not passed in a chained action. The action that returns this error must be invoked separately. For example, there is an action set with 4 actions. Actions 2, 3 and 4 depend on action 1. Actions 2 and 3 are invoked successfully but action 4 fails with a <code>ce context</code> error. The solution is to run actions 1-3 for the <code>OnInvoke</code> status and run action 4 separately for the <code>OnSuccess</code> status.</li> </ul>
Static	Map the parameter to a fixed value.
Global Variable	Map the parameter to a user-defined variable. You define these in <b>Global Configuration</b>  .
System Variable	Map the parameter to a predefined system variable. To view these go to <b>Global Configuration</b>  .
Toolbar Query Param (IFS Customer Engagement only)	<p>Map the parameter to an Agent Desktop runtime parameter. For example, to the activation ID of the current email, chat message or voice call.</p> <div data-bbox="678 1507 1471 1644" style="background-color: #f0f0f0; padding: 10px;"> <p><b>Note</b> Not applicable to self-service only tenants or IFS Services.</p> </div>

Source type	Parameter mapping
Event Context	<p>Map the parameter to a click event on a wide variety of component types including chart, data element and label element. For example:</p> <ul style="list-style-type: none"> <li>Clicking a row in a data grid updates a data element with that data. The data element must subscribe to the OnNotification event.</li> <li>Clicking a data element updates a form to show that data. The form must subscribe to the OnNotification event.</li> <li>Clicking a segment in a pie chart, updates a data grid to show all the data represented in the segment. The data grid must subscribe to the OnNotification event.</li> </ul>
Session > AppData	<p>Map to data passed into the CE Studio app, for example:</p> <ul style="list-style-type: none"> <li>From script elements used in queue progression (IFS Customer Engagement only)</li> <li>From link parameters defined in App Settings &gt; Parameters.</li> </ul>
Session > Auth	<p>Map the parameter to an auth property, such as UserFirstName, UserFullName.</p>

- Click **Manage Properties** to add and remove parameters to the action. You can then map their parameters.

Click  to view the properties of the data type and add additional optional properties to the action using the [Metadata Viewer](#).

- Select **Publish Topic** for actions that need to notify other components that response data is available.

Other components will need to subscribe to this. See [About actions and events](#).

**Note** This subscription is local to the app or template. When working with templates, you can make the topic public so that an app can subscribe to it, see [Public topics and templates](#).

- Click **Save**. This saves the action and *attaches* it to the action set.

**Note** If you see the message `The action cannot invoke Client functions when sibling actions invoke Server functions`, then the action has been saved (it is listed on the All Actions page). To use the action, you will need to attach it to an action set that does not contain actions from other types of provider.

### Attaching and detaching actions to action sets

In the **Action Sets** view, click  to add an existing action to an action set. You can add the same action to several action sets.

**Note** You cannot add an IFS CE Agent Desktop provider action to an action set that contains actions for server-side providers such as the IFS CE Email provider. You will see the message `The action cannot invoke Client functions when sibling actions invoke Server functions`.

Click  to remove an action from an action set. The action is not deleted and is listed in the **All Actions** view. Any existing subscriptions are preserved.

**Note** You cannot detach an action that is the data source for a data transformation or provides a response for a chained action. To do this, you would first need to remove the data transformation or detach the chained action.

### Deleting actions and action sets

You can only delete actions and actions sets that are not used.

To delete an unwanted action set, you can detach the actions in the action set and then delete the action set. This does not affect the components that subscribe to the actions that were originally in the action set. To continue to use the actions, add them to another action set.

To delete an unwanted action, you must first reconfigure or delete anything that is using the action. For example, reconfigure the component that is subscribing to the action: go to the Event Configuration page and remove the OnNotification event for the component.

**Note** You must delete actions all actions. Deleting a single action set is not sufficient, as there may be relationships with other action sets.

### Running the actions in a set in a particular order

If you want the actions in an action set to execute in a particular order then use change the order of the actions in the set by dragging and dropping.

### Filtering results returned by an action

Use one or more conditions to filter the results returned by the action, for example, to discard data that is not required by the app. Use *Groups* to switch between logical operators. For details, see [Actions and filters](#).

### Transformations on actions

**Note** These simple transformations apply to the action and not to the action set. For details of complex transformations that apply to the action set, see [Data transformations](#).

Use an input transformation to change the property type that will be used to invoke the operation on the external data source. Use an output transformation for simple transformations on string properties. For example, to change a simple field value (that is not a Boolean) into something more user-friendly. Output transformations are applied after the results have been filtered.

The screenshot shows the 'Transformations' panel in a software interface. It is divided into 'Input Transformation' and 'Output Transformation' sections. The 'Input Transformation' section has a dropdown menu with the text 'Please select a property' and a plus icon. The 'Output Transformation' section has a purple 'Manage Properties' button. Below this, a table shows the transformation for the 'company\_name' property of a 'contact' entity. The table has two columns: 'Source' and 'Transform'. There are two rows of transformations, each with a trash icon to its right. A plus icon is at the bottom left of the table.

Source	Transform
ipswich	IFS (Ipswich)
staines	IFS (Staines)

*Example of an output transformation*

**Note** You cannot apply transformations to Boolean values.

### Sorting the results returned by an action

Set a default column and sort order for ordering the rows in a data grid. Agents can click to sort on a different column.

## Paging

The paging will depend on how you intend to use the action. By default, the page size is 10 rows. This default is intended for use with UI elements such as data grids. You must change the page size if you intend to use the action with charts or data transformations because the page size also sets the maximum number of records available for the chart or data transformation.

## Actions and filters

Use one or more conditions to filter the results returned by the action, for example, to discard data that is not required by the app.

There are several ways of setting up the condition:

Left side of condition	Right side of condition	Description
Field Reference	Field Value	Filter a provider property by a field value entered by the user (for example).
App Template Option	App Template Option	Filter by one of the template options. See <a href="#">Adding configuration options to templates</a> .
ToolbarQueryParam	ToolbarQueryParam	Filter by a setting in Agent Desktop.
Field Value	Field Value, static value or global variable (but not a Field Reference).	Filter by the values entered by the user in one or more fields, or using a static value, or using the value stored in a variable.  Note that you cannot create a condition where the right side of the condition is a field reference. If using a field reference, then this must be on the left side of the condition.
Regex	Regex	Filter by a regular expression configured on the Global Configuration tab.

**Note** You can create reusable filters on the  Filters page. To use a filter of this type, you need to use rule sets.

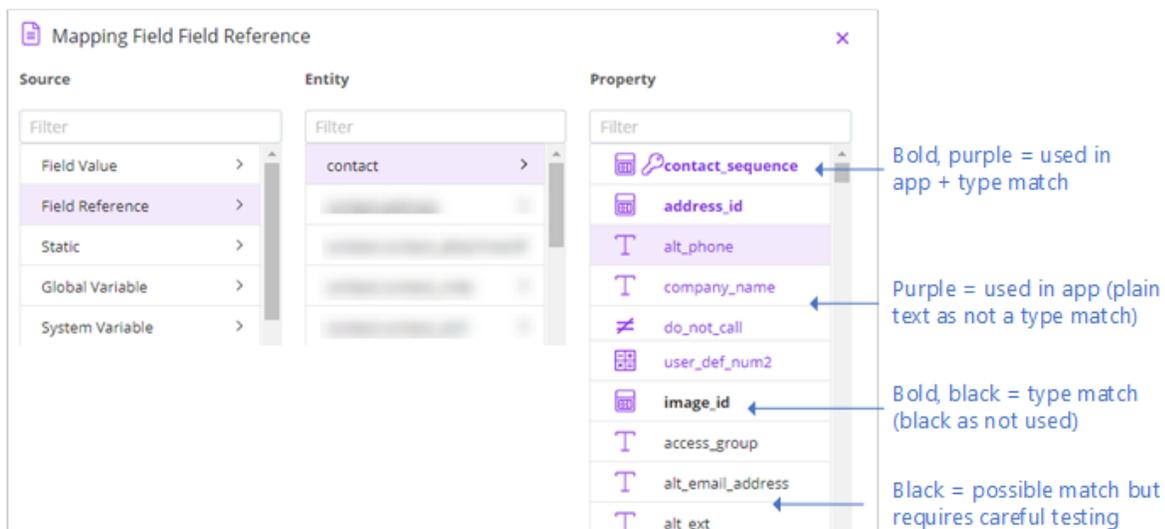
**Note** If the action depends on call data values, then the action cannot be previewed. The action will only run in Agent Desktop.

### Mapping the left and right sides of a condition

When mapping the left and right sides of the condition, you can map:

- A date to a date, string or decimal
- A number and a Boolean to any data type other than a date
- A decimal to any data type

To make it easier to set up the conditions, there is a symbol for each data type. Also, the following conventions are now applied when configuring conditions as shown in the following figure:



The screenshot shows the 'Mapping Field Reference' dialog box with three columns: Source, Entity, and Property. The Source column lists 'Field Value', 'Field Reference', 'Static', 'Global Variable', and 'System Variable'. The Entity column lists 'contact'. The Property column lists various fields: 'contact\_sequence', 'address\_id', 'alt\_phone', 'company\_name', 'do\_not\_call', 'user\_def\_num2', 'image\_id', 'access\_group', 'alt\_email\_address', and 'alt\_ext'. Annotations on the right side explain the styling: 'Bold, purple = used in app + type match' (pointing to 'contact\_sequence'), 'Purple = used in app (plain text as not a type match)' (pointing to 'company\_name'), 'Bold, black = type match (black as not used)' (pointing to 'image\_id'), and 'Black = possible match but requires careful testing' (pointing to 'alt\_ext').

If the data types match then the property on the right side of the condition is in **bold** text. This means that if you select a bold type then the condition will work.

If, however, the data types are not a direct match then the property on the right side of the condition is in plain text. For example, for a date property, a date value is in **bold**, but string or decimal values are in plain text. If you select a non-bold type then the condition may not work, for example, it may depend on the values in that property. *You will therefore need to test the action carefully to make sure that it works in all required cases.*

## Selecting operators

The operators available to the condition depend on the data type, for example strings and numbers use different operators.

The available operators also depend on the provider - not all endpoints support all the operators. Although an operator is listed, such as **Like**, **Not Like**, **Matches Pattern**, **Not Matches Pattern**, it does not mean that the provider will support it. This applies, in particular, to projections configured as a IFS Applications/Cloud provider. When the operator is not supported then you will see an error when you run the action. You will see errors similar to the following:

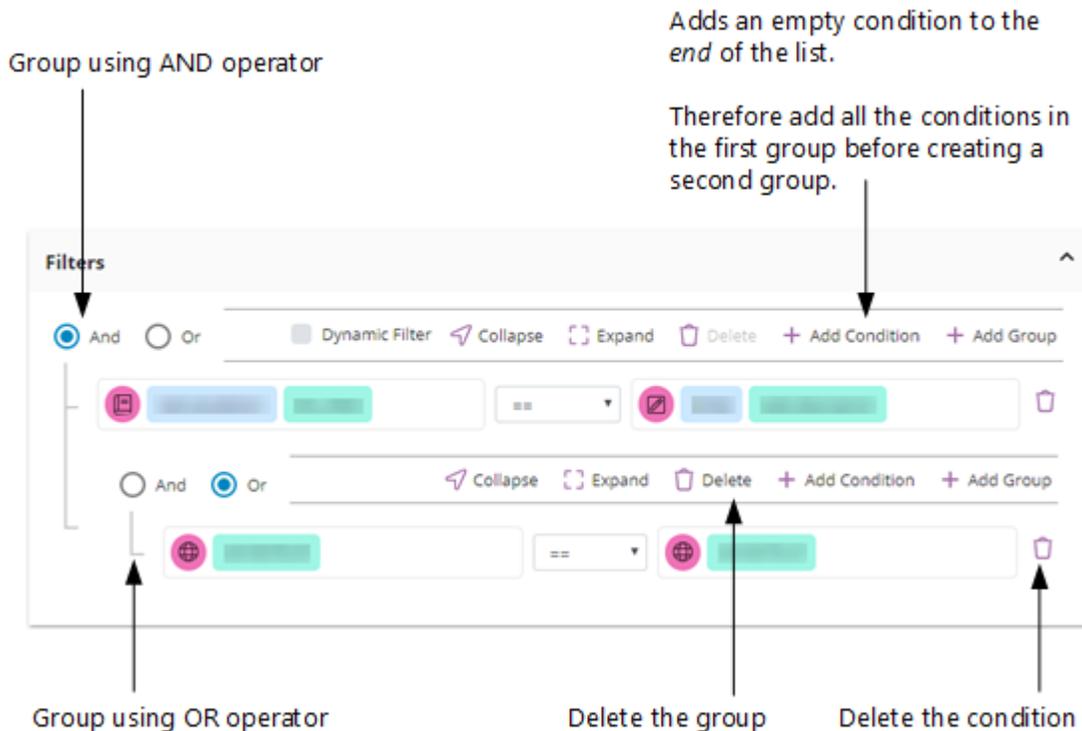
- The given key 'notStartsWith' was not present in the dictionary.
- EXPRESSION\_PROPERTY\_NOT\_IN\_TYPE

If you are using multiple conditions and the fields are different data types then you can use **Matches Pattern**, **Not Matches Pattern**. Regex is a pattern which can generalize any pattern of string. You can use the predefined regex patterns from the Global Configuration tab, and add your own regex patterns.

Regex is used when you are adding a condition to a rule set or an action set to map a value. By using this feature, you are able to search multiple columns for a filter and these multiple columns can have different data types.

## Using groups to switch between logical operators

Use *Groups* to switch between logical operators.



Example of adding conditions to a filter

### Logical operators AND OR

The logical operators AND OR determine the relationship between the conditions in the action filter, both within the group and the relationship between groups. The default operator is AND.

The conditions in the group are evaluated when the action runs:

AND	<p>If any one of the conditions in the group evaluates as <i>false</i> then the whole group evaluates as <i>false</i>, and the provider returns empty rows.</p> <p>When all the conditions in the group evaluate to <i>true</i> then the provider returns the rows (if any) matching the action filter.</p>
OR	<p>If any one of the conditions in the group evaluates as <i>true</i> then the whole group evaluates as <i>true</i>. The provider returns the rows (if any) matching the action filter.</p> <p>If all the conditions in the group evaluate as <i>false</i> then the whole group evaluates as <i>false</i>, and the provider returns empty rows.</p>

You can add additional groups. When you add another group, you add it as the child of the group above it.

When evaluating the conditions in the filter, you start with the child group. For example, if the child group evaluates to *false* and the parent group operator is AND then the overall outcome is *false*

regardless of how the conditions evaluate in the parent group. The provider will therefore return empty rows.

**Note** Conditions that contain field references are always sent to the provider - the provider will evaluate them. Conditions that map to field values or variables are evaluated by CE Studio as the provider has no knowledge of these. This means that:

- Pre-evaluated conditions that evaluate as *true* are included in the action filter
- Pre-evaluated conditions that evaluate as *false* are excluded

## Actions and dynamic filters

Use a dynamic filter when you do not know which field in a form, for example, will have data. The dynamic filter will take the value(s) from any populated field(s) and ignore the rest. Select the **Dynamic Filter** check box when configuring the filter on the Actions page.

The screenshot shows the 'Filters' configuration page. At the top, there are radio buttons for 'And' (selected) and 'Or'. To the right, there is a checked 'Dynamic Filter' checkbox, and buttons for 'Delete', '+ Add Condition', and '+ Add Group'. Below these are five filter rows, each with a document icon, a field reference, an operator, a search action, and a field value:

Field Reference	Operator	Action	Field Value
contact email_address	Equals	Contact Search	Email Address
contact last_name	Equals	Contact Search	Last Name
contact first_name	Equals	Contact Search	First Name
contact company_name	Equals	Contact Search	Company Name
contact mobile_phone	Equals	Contact Search	CallersNumber

*Example of a dynamic filter used in a search on any combination of email address, name, phone number*

## Rules and rule sets

**Note** You can download an example that demonstrates the use of rule sets. This is available from the Central Template Repository which is part of CE Studio Designer.

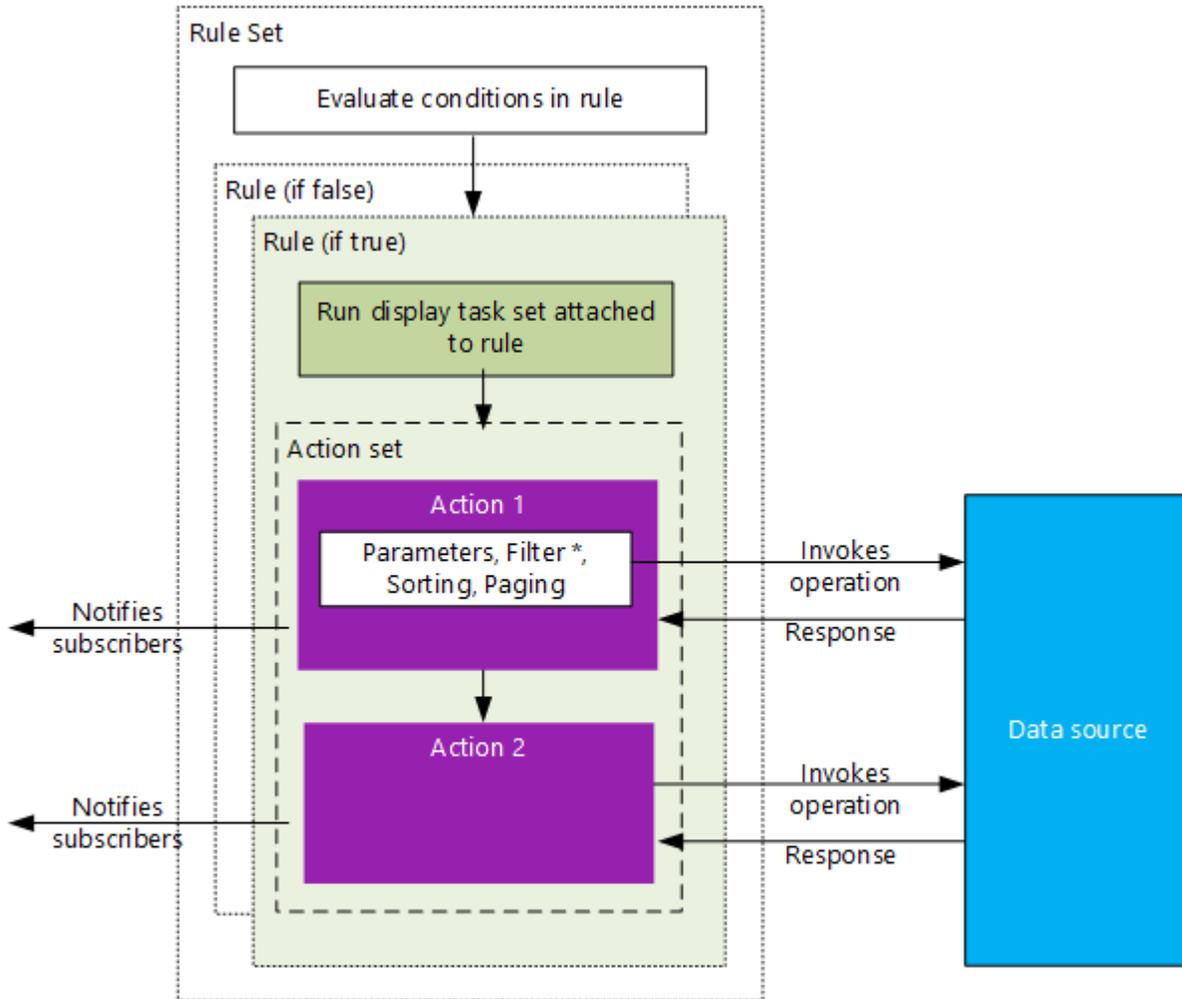
R..	S..	S..	Created By	↑
SITE	02	OP	CSR	27 J...
FIX	01	CL	ADMIN	13 ...
FIX	01	CL	ADMIN	13 ...
FIX	01	CL	ADMIN	14 ...
FIX	01	OP	ADMIN	14 ...
SE...	01	OP	STUDIO	5 F...
SE...	01	OP	STUDIO	5 F...
SE...	01	OP	STUDIO	5 F...

Request T... ↑	Severity	Status
FIX	01	CL
FIX	01	CL
FIX	01	CL
FIX	01	OP

You chose severity 1 requests.

Rules and rule sets (see figure below) let you define more complex behavior for the template or CE Studio app. For example, depending on conditional logic, you can update the user interface, run actions and filter the data in the response. At runtime, a rule set is triggered when a specific event and event status occurs. Each event status can trigger a different rule set.



\* Any standalone filters added to the rule are combined with any filters in the action.

*How rules and rule sets work*



*How the event processes the rule set*

### Rules without any conditions

A rule does not have to have any conditions if its only purpose is to run an action set or display task set. This enables you to group actions sets or display task sets together and run them in sequence.

**Note** Rule sets are required when using the IFS CE Agent Desktop provider. For this provider, you must add action sets to rule sets. This provider cannot directly call action sets.

### Rules with conditions

A rule can be defined by one or more conditions. This rule contains a single condition:

Depending whether this condition evaluates as *true* or *false* then the rule can do one or more of the following:

- Run a display task set to update the user interface or show messages. The UI state is changed once all the rules in the rule set are evaluated. Similarly, the message(s) are shown once all the rules in the set have been evaluated.
- Run an action set to perform a task on the data used or required by the CE Studio app.

A rule can have multiple conditions that will be evaluated in the order in which they are defined in the rule.

**Note** Strings can be either empty (contains no characters) or null (string doesn't exist). Where a condition does not seem to return the correct value, use the **Is Not Null or Empty** condition.

## Rule sets

A rule always belongs to a rule set. These are used to group related rules. These are the rules that relate to a specific event.

At runtime, the rules run in the order in which they are listed in the rule set. If a condition in a rule evaluates as *false* then you can stop further processing of rules in the set.

**Note** The values you enter are case sensitive.

## Adding a rule set

1. Click **Create New Rule Set**. Use this to group the rules for an event. When you configure the events for the app or template, you work with the rule set and not with the individual rules in the set.
2. Click **Add Rule**. This must have a unique name as you can detach a rule from one rule set and add it to another.
3. Configure the rule.
  - If you want a series of rules using both AND and OR operators then add a group for each operator type.
  - Add all the conditions to the first group before adding a second group.
4. In **When True (Default)**, optionally select the display task and/or action set that you want to run when the condition(s) in the rule evaluate as *true*.
5. In **Otherwise**, repeat this step for when the condition(s) in the rule evaluate as *false*. This is optional.
 

For example, you might use display tasks to show toast messages when a rule fails.
6. If there is more than one rule in the rule set and the condition(s) in the rule evaluate as *false*, you can stop further processing of the rules in the step: select the **Stop Further Rules if False** check box.

## Event configuration for apps and templates

---

Click  on the CE Studio Designer toolbar to configure the following event types for apps and templates. For events on inner components and elements, including the OnNotification event, see [Event configuration for components](#).

## OnLoad

The OnLoad event occurs when the app loads or a template is loaded by the app. Loading occurs in parallel, and display tasks such as toast messages may display in any order.

You can attach a different action set, rule set or display task set to each OnLoad status:

- OnInvoke
- OnSuccess
- OnFailure

## OnTimerElapsed

The OnTimerElapsed event occurs when a *timer* elapses. Timers are set at the app, component or element level as part of the settings. You can configure any action or display task to run when this event type occurs.

## Event configuration for media apps and templates

---

Click  on the CE Studio Designer toolbar to configure the following event types for media apps and templates. For events on inner components and elements, including the OnNotification event, see [Event configuration for components](#).

### OnToolbarEventReceived

These are toolbar events from Agent Desktop such as agent state, voice call state and chat session. They can be one of the following types:

- EVT\_AGENT\_STATE\_CHANGE: the agent's state changes as they accept an activation, work on the activation, wrap up and then close it. See [Agent Desktop events](#).
- EVT\_OUTBOUND\_STATE\_CHANGE: for outbound voice calls, for example, call starts to ring, call is answered, disconnected. See [Agent Desktop events](#).
- EVT\_START\_CHAT: a new chat session starts.
- EVT\_END\_CHAT: either the caller or the agent ends the chat session.
- EVT\_CHAT\_MESSAGE: either the caller or the agent sends a new message.

To respond to these types of event, you need to configure a rule set. For background information, see [IFS CE Agent Desktop provider](#).

### OnPostMessageCallbackReceived

These are callbacks from Agent Desktop, such as after sending a chat message or ending a chat session. You can attach rule sets to these callbacks so that action sets and/or display task sets to run in response to the callback.

Callback	Notes
sendChat	After a chat message is sent.
sendChatTyping	When the agent or caller starts typing in a chat session.
sendChatStarted	After a chat session starts.
dialOut	After dialing an outbound call.
dialOutEx	After details of the outbound call are returned.
hangUp	After hanging up the current call.
wrapUp	After wrapping up the current activation.
holdCall	After putting a call on hold.
transferCall	After connecting the primary call to the enquiry call. (Once the agent leaves the call, the agent will be in a wrap up state.)
pauseRecording	If call recording is in use, after pausing the recording. For example, while the agent is handling sensitive information.
resumeRecording	When call recording is in use, after restarting the recording.
connectToPrimary	After connecting or reconnecting the agent to the primary voice call.
connectToEnquiry	After connecting the agent to a second call (the enquiry call).
conferenceCallers	After connecting the calls to start a conference call.
getAgentState	After getting the agent's current state. For a list of states, see <a href="#">Agent states</a> .
getActivationType	After getting the type of the current activation.
getCallId	After getting the activation ID.
getEmailInfo	After details of the current email are returned.
closeEmail	After closing the current email.

## Event configuration for components

Click  on the CE Studio Designer toolbar to configure the following events for the components and elements belonging to an app or template. This includes the OnNotification event. When any of these occur with status OnInvoke, OnSuccess or OnFailure, you can:

- Attach a rule set that runs any combination of action set, display task set and filter
- Run an action set and any attached display task set
- Run a display task set

For OnNotification events, you need to set up the subscription (see below).

### Supported event types

Event type	Event target
OnRowSelected, OnItemSelected	<p>When a single row is selected in a data grid, for example to populate the fields in a form, or an attachment selected for download or single item is selected from a list view.</p> <p><b>Note</b> Do not use OnRowSelected for data grids where <i>multi-row select</i> is enabled.</p>
OnLoad	<p>When a component is loaded by the app or template. Loading occurs in parallel, and display tasks such as toast messages may display in any order.</p>
OnChange	<p>Specifically for UI elements, such as text boxes, number fields, currency fields. For example, use this event to run an action (for example) to update what's shown in a data grid.</p>
OnChanged, OnEnter	<p>Specifically for chat and the typing notification.</p>
OnChange Delay	<p>Allows to set the number of milliseconds before the event is triggered. It is set to 1000, ie 1 second, by default.</p> <p><b>Note</b> Must utilize the OnChange event.</p>
OnValueChanged	<p>Specifically for combo boxes to run an action (for example) in response to the new selection.</p>
OnClick	<p>Specifically for buttons.</p>

Event type	Event target
OnSeriesClicked	For use with chart components.
OnSlotClick	When the user clicks a slot in a calendar component.
OnTimerElapsed	When a <i>timer</i> elapses. Timers are set at the app, component or element level as part of the settings and elapse after a configured time interval. You can configure any action or display task to run when this event type occurs.
OnSurveySubmit	Specifically for survey elements (not available in IFS Services).

### OnNotification events

OnNotification events occur when an action invokes a provider method and then notifies its subscribers. The subscribers are any components that will consume the response from the provider method. You either configure the OnNotification event on the parent component or on the element itself:

Data grid	Configure the OnNotification event and subscription on the data grid itself.
Form, data element, label	Configure the OnNotification event and subscription on the component that contains the: <ul style="list-style-type: none"> <li>• form</li> <li>• data element</li> <li>• label</li> </ul>
Image element	Configure the OnNotification event and subscription on the image element itself.

For an example of an action that publishes a topic (topic description), see [Adding an action and loading a data grid](#).

### About the event context

An event context passes information to the action that is configured to use the event context so that it can use the data from elements such as adaptive card, data element, survey element.

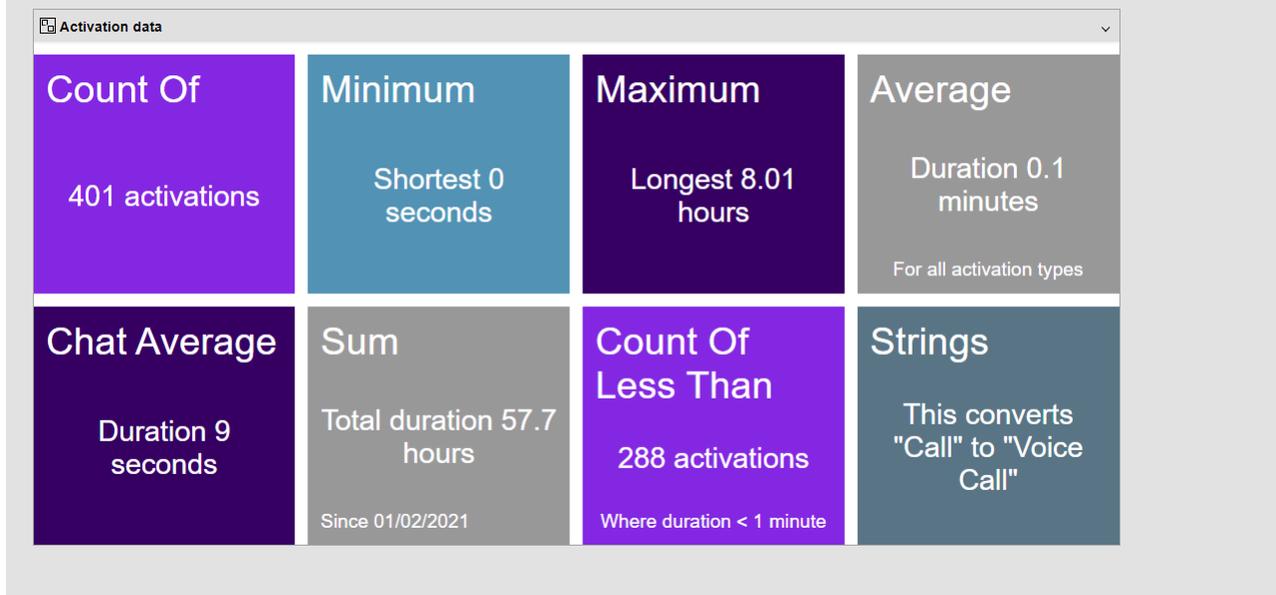
When configuring the element, such as a data element, you may need to provide a key. This means that if you subsequently replace the element, you may need to remap the action so the event context is correctly defined with the correct element identifier.

Not all elements need a key, for example, manually-configure list view elements. A manually-configured list view element has a click event that you can use to run an action set, rule set and/or display task set. The value passed by the click event is the name of the list view item. In an action,

for example, configure the left side of the condition as a field reference and the right side to **Event context > List View > GetSelectedListItem**.

## Data transformations

**Note** You can download an example that demonstrates the use and configuration of data transformations from the Central Template Repository which is part of CE Studio Designer.



You use data transformations to modify data from a provider, for example, to aggregate data for use in data grids, charts and data elements.

You configure these as part of the action set. The data that you want to transform is provided by the actions in the action set. Where necessary, use the action to filter the data. The action can use any provider method that returns data, such as `get`, `patch`.

You can add multiple queries to the data transformation. Once a query is created in an action set and saved, a virtual provider, Transformed Data, is created for the app. Each query configures a new provider type for that virtual provider and each output in the query configures provider properties for that query type. The new provider type can also be used in a subsequent query in the same data transformation as a data source. You need to publish the query to use it in the app.

Each query in a data transformation will publish a topic with the topic type Transformation whereas the topics published by actions will be marked as Standard.

For full details, see [Configuring data transformations](#).

## Configuring data transformations

To configure a data transformation, you first create an action set with an action that will provide the primary data. You can then configure the data transformation.

**Note** For background information, see [Data transformations](#).

### Creating the action set

1. Create an action set for the data transformation.

Choose a descriptive name because the name of the action set will become the name of the transformed data type, both in the Metadata Selector and in Designer. The name of the new data type will be <action set>-<query name>.

2. Add the actions to the action set that will return the data.

The data returned by the action becomes the primary data source for the transformation. You can add multiple actions if the data transformation requires data from more than one data source.

You can use any provider methods that return suitable data.

**Note** Do not set a page size on the action. This will limit the number of rows used in the data transformation.

**Note** The Data Transformation in Data Grid does not support sorting when an action contains filters with dynamic values. [Configuring data transformations](#)

3. To make the data available for data transformations, these actions must publish a topic.

### Creating the queries

1. Edit the action set .
2. Go to the **Data Transformations** tab and click **Add Query**.
3. Create the query:

Field	Description
Query Name	<p>This is the name that you are giving to the transformed data type. Query names can contain alphanumeric characters, but not spaces or special characters.</p> <p>The name of the new data type will be &lt;action set&gt;-&lt;query name&gt;.</p> <p><b>Note</b> You cannot change the query name once you have used in an app.</p>
Primary Data Source	<p>Select the data source you want to use. This is the data type that's returned by the action(s) you configured for this action set. If there are multiple actions returning the same data type then choose an action name that differentiates between them.</p>
Primary Data Source Alias	<p>A unique short name that you will use to refer to the primary data source later in the data transformation. Aliases are case sensitive and can contain alphanumeric characters and underscores, but not spaces or special characters. They must not start with a number.</p> <p><b>Note</b> These naming conventions apply to all aliases created on the Data Transformations tab.</p>
Max. No. of Rows	<p>The maximum number of rows that you want to output. For example, the top 100 rows in the data source. Enter 0 to return all the rows.</p> <p><b>Note</b> You can change this later after creating the queries. This page size does not override the page size set in the action.</p>

Field	Description
Publish	<ul style="list-style-type: none"> <li>• Select the <b>Publish</b> option if you want the transformed data to be available in Designer. The transformed data can also be used by another query.</li> <li>• Clear the <b>Publish</b> option if the data transformed by this query will not be available in Designer. The transformed data is only for use by another query.</li> <li>• The last or only query in the transformation must be published.</li> </ul>

### Sorting the query

Optionally, you can sort the data or grouped data that is output by the query. To sort by data that is grouped, you must enter the alias used on the **Grouping** tab.

**Note** Use the sorting to control the order of categories and series on the X and Y axis of charts.

### Filtering with a Where clause

On the **Joins** tab, you can filter the data in the query, for example:

```
<alias>.Call_Type == "Caller rang off"
(<alias>.Type=="Call") && (<alias>.Initial_Skillset_Name=="ABCD")
```

**Note** You do not need to add a join to use the Where clause.

### Configuring a join

You can configure one or more joins between tables on the **Joins** tab. For example, to create data that combines an employee's personal details with their address details. You can use an optional Where statement to filter the data that's used in the join.

**Note** The join expression follows SQL join syntax. Use aliases when referencing properties.

Field	Description
Data Source	<p>Select the data source that is the target of the join (the right side of the expression). The action set needs to contain an action that returns this data.</p> <div style="background-color: #f0f0f0; padding: 10px; border: 1px solid #ccc;"> <p><b>Note</b> The data for the left side of the expression is provided by the primary data source set on the <b>Configuration</b> tab.</p> </div>
Alias	<p>A shortname for the data source that is the target of the join (the right side of the expression). You will use this alias later in the data transformation to refer to fields in the join.</p>
Source Expression	<p>For the first join this is the primary data source.</p> <p>To reference the fields in the primary data source, use the alias you entered on the <b>Configuration</b> tab.</p> <p>For example: <code>&lt;primary_alias&gt;.addressId</code></p>
Target Expression	<p>Applies to the target data source.</p> <p>To reference the fields in this data source, use the alias you entered in the <b>Alias</b> field on this tab.</p> <p>For example: <code>&lt;alias&gt;.add_id</code> giving an expression:  <code>&lt;primary_alias&gt;.addressId=&lt;alias&gt;.add_id</code></p> <p>You do not need to enter the = between the left and right sides of the expression as this is implied.</p>

Field	Description
Join Type	<p>How the expression is evaluated:</p> <ul style="list-style-type: none"> <li>• <b>Inner</b>: Returns rows that have a matching value from both sides of the expression.</li> <li>• <b>LeftOuter</b>: Returns all rows from the left side of the expression and matching rows from the right side. Where there is no expression match, data will be NULL from the target specified on the join.</li> <li>• <b>Cross</b>: Returns every combination of rows from the left and right sides of the expression.</li> <li>• <b>RightOuter</b>: returns all rows from the right side of the expression and matching rows from the left side. Where there is no expression match, data will be NULL from the source data source (that is the query you are building).</li> <li>• <b>FullOuter</b>: returns all rows when there is a match from the right or left side of the expression.</li> <li>• <b>FullOuterDistinct</b>: Same as FullOuter but removes duplicates.</li> </ul>

If required, you can filter the data returned by the join using a Where clause. The Where clause accepts static values only. For example:

```
<alias>.last_name != "Shaw"
```

where

- == is the operator equal-to
- != is the operator not-equal-to

### Configuring grouping

On the **Grouping** tab, you can group rows that have the same values in one or more fields in order to create summary rows or as input to aggregation operations such as `Count ( )` and `Avg ( )`. The Having clause enables you to filter the data in the grouping.

**Important** If you configure a grouping then you can only use the aggregate functions on the **Output** tab. `Count ( )` will act on the grouped field.

Field	Description
Expression	<p>Enter the field that you want to group by.</p> <p>For example, to group by a last_name field from the primary data source, enter:</p> <pre>&lt;primary alias&gt;.last_name</pre> <p>To group by a field that is the result of a join and not in the primary data source, enter:</p> <pre>&lt;join alias&gt;.job_title</pre> <p>You can also group by a property of a field, for example, if the field type is timespan. In this example, the field is grouped by the hour part of the timespan:</p> <pre>&lt;primary alias&gt;.Activationtime.Hour</pre> <div style="background-color: #f0f0f0; padding: 5px; margin-top: 10px;"> <p><b>Note</b> If there is no suitable field to group by then you can enter a static value as the expression. This means that all rows will be in the same group.</p> </div>
Alias	<p>A unique shortname to identify this grouping on the <b>Output</b> tab. (See Primary Data Source Alias for details.)</p>
Having	<p>Enter an expression to filter the data returned by the Grouping operation.</p> <p>For example, to exclude data where there is only one instance of the value, enter:</p> <pre>Count() &gt; 1</pre> <p>To exclude people under 18 years of age, enter:</p> <pre>Count(&lt;alias&gt;.age &gt; 18)</pre>

### Configuring the output from the query

The final step is to configure the data you want to return on the **Output** tab. The query must return all the data that you intend to use, for example, all the properties required for a data grid or chart.

1. For each property that you want to output, click **Add Output** and then enter the details:

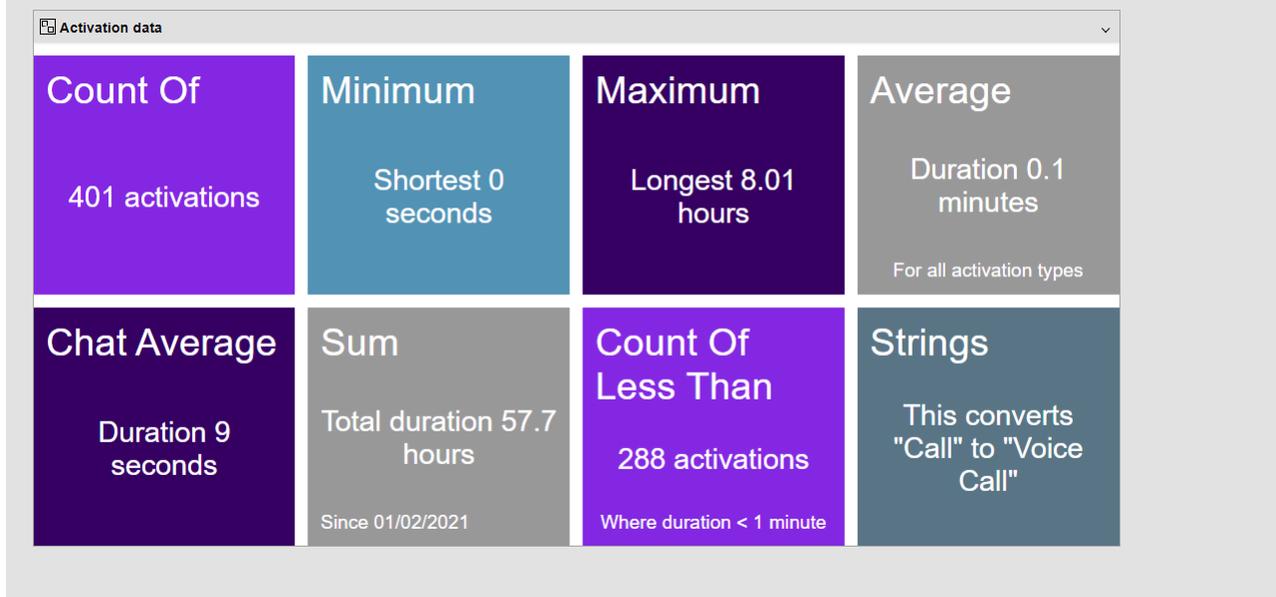
Field	Description
Field/Expression	<p>Is one of the following:</p> <ul style="list-style-type: none"> <li>• A property from the primary data source prefixed with the primary data source alias, such as <code>&lt;primary_alias&gt;.address</code>.</li> <li>• A property from the joined data sources prefixed with the join alias, such as <code>&lt;join_alias&gt;.fullAddress</code></li> <li>• An alias from the <b>Grouping</b> tab. For example, if you group by a <code>last_name</code> property then the alias for this grouping will give you the grouped last names.</li> <li>• An aggregate function (see <a href="#">Transformation output examples</a>).</li> <li>• A standard <code>c#</code> expression (see <a href="#">Transformation output examples</a>).</li> </ul>
Alias	<p>The name of the new property that is output by this query. You will use this name in Designer when selecting this property.</p> <p><b>Note</b> Although you can add multiple output expressions, you cannot use an alias declared in one output expression in another output expression in the same query. You need to fully declare the field name.</p>
Type	<p>You do not need to select a data type unless you want to change the data type as part of the transformation. The query assumes the type is a string.</p> <p><b>Note</b> If you change the type after the property has been used in a component then you will need to remove that property (for example from a data grid) and add it again.</p>

2. Click **Continue** to validate the query and save it. A green tick appears if the query is valid.
3. You can now add other queries to the data transformation. The first query you added can be used as a data source in the new query.

**Note** Aliases declared in the first query are not available in subsequent queries, therefore each query can use the same aliases if you wish.

## Transformation output examples

**Note** You can download an example that demonstrates the use and configuration of data transformations from the Central Template Repository which is part of CE Studio Designer.



There are three types of output:

- A field referenced by its alias (primary or join)
- An aggregate function
- A *c#* expression

### Fields

For example consider the alias *CityGroup*:

- If *CityGroup* is a previously declared primary alias then this will pass through the data from the primary data source without any transformation.
- If *CityGroup* is the alias for a grouping then this will pass through the result of grouping the data. Note that if you have a grouping then you *cannot* also pass through ungrouped fields from the primary data.

This example is valid as *hour* is a grouped field and *Max(a.Waiting)* is an aggregate function:

Output	
Field/Expression* ⓘ	Alias*
Max(a.Waiting)	Waiting
hour	Time

But this example is not because *a.Waiting* is simply a field from the primary data source and neither grouped nor part of an aggregate expression:

Output	
Field/Expression* ⓘ	Alias*
a.Waiting	Waiting
Alias/parameter 'a' not found	
hour	Time

### Aggregate functions

If you configured a grouping then you must use one of the aggregate functions:

Function	Returns...
Count()	The number of values in the grouped field that evaluate to <i>true</i> .
Count( <i>predicate</i> )	<p>The number of grouped rows that match the expression you enter. This must return a Boolean — only rows where the expression returns <i>true</i> are counted.</p> <p>For example:</p> <pre>Count(&lt;alias&gt;.waiting_time &gt; 300) Count(&lt;alias&gt;.age == 21) (Count(&lt;alias&gt;.Call_Type == "Caller rang off") * 100) / Count()</pre>
Sum( <i>expression</i> )	<p>The total sum of a numeric field. For example:</p> <pre>Sum(&lt;alias&gt;.line_price) Sum(&lt;alias&gt;.line_price &gt; 10)</pre>

Function	Returns...
<code>Avg(expression)</code>	The average value of a numeric field. For example, this averages the activation wrapup times and rounds the result to two decimal places:  <pre>Avg(&lt;alias&gt;.Wrapup).ToString("0.00")</pre> For a timespan, use an expression such as <pre>Avg(&lt;alias&gt;.TimeSpanObject.TotalSeconds)</pre>
<code>Min(expression)</code>	The row with the lowest value in a numeric field.
<code>Max(expression)</code>	The row with the highest value in a numeric field.

### c# expressions

You can enter any valid c# expression that returns a data type recognized by the CE data transformation engine. Before using the expression, we recommend testing the expression using an online code editor.

**Note** The data transformation engine returns the message `Cannot invoke a non-delegate type` if it does not recognize the data type. For example, although the following is a valid c# expression: `d.zipcode.Substring(d.zipcode.IndexOf("")+1)`, it does not return a string. As a workaround, you would use this instead: `d.zipcode.ToString().Substring(d.zipcode.ToString().IndexOf("")+1)`.

Some examples are given below.

Type	Example
String concatenation	There are two ways of concatenating strings: <ul style="list-style-type: none"> <li><code>String.Concat(c.fname, " ", c.mid_name, " ", c.lname)</code></li> <li><code>c.fname + " " + c.mid_name + " " + c.l_name</code></li> </ul>
Convert datetime to string	To convert a datetime to a string and then format it. For example, to get just the date or just the time: <pre>p.ActivationTime.ToString("hh:mm")</pre> <pre>p.ActivationTime.ToString("yyyy:MM:dd")</pre>

Type	Example
Extract month from datetime	To extract a date part (month in this case) from a datetime field, use <code>Datetime.Month</code> :  <code>c.date.Month</code>  where <code>c.date</code> is the datetime field.
Calculate a percentage	<code>(Count(&lt;alias&gt;.Total_Operator_Duration &lt; 6) * 100) / Count()</code>
Round a floating point number	To round a floating point to 2 decimal places, use <code>ToString()</code> :  <code>&lt;alias&gt;.WaitingTime.ToString("0.00")</code>  <b>Note</b> If the transformation is on an integer then you may need to cast the number to a suitable type first, for example,  <code>((decimal)a.TotalAgentDuration)/3600).ToString("0.00")</code>

## Using transformed data

**Note** For background information, see [Data transformations](#).

### Viewing the transformed data types

Click  on the main toolbar to open the Metadata Viewer and then select the transformed data type. The name is a combination of the action name followed by the query name: `<action>-<query name>`.

### Using the transformed data types

1. Add a component, such as a data grid, chart or data element.
2. Click  to select the provider.
3. Select **Transformed Data**.
4. Type in its name to list the new data types or click  to open the Metadata Selector.
5. Add the data types in the usual way.

## Event configuration

When an action set has a data transformation then the actions used in the data transformation will publish a Transformation topic where the return type is the <action>-<query> data type:

### DataGrid - Topic Subscriptions

	Topic Type	Return Type
ataForSegment...	Standard	task
lectedContact...	Standard	contact
isks_get_Respo...	Standard	task
hartData_taskT...	Transformation	getChartData-...

To use the transformed data, components or elements need to subscribe to this topic rather than to the Standard topic.

## Global actions

A global action makes data from a provider available in the CE Admin Portal so that you can:

- In Report Designer, design reports that include external data.
- In the Media Script Editor, configure channels, such as chat and email, to make use of external data.

You can add placeholders to global actions which will be substituted later when you use the global action in the Report Designer or Media Script Editor. For example, the running script can substitute a placeholder with a value obtained from the caller. All the provider properties will be available for selection in the Report Designer and Media Script Editor.

Global actions are not associated with any app or template. Global actions will always use the active version of the provider — you cannot select a specific provider version. When working with global actions, note that:

- You can create a global action for any of the providers in the system not just for the providers in the app or template.
- You always use the active version of the provider – you can't select a specific version.
- You can filter the data that's returned by the provider or you can set up a placeholder and then configure the filter in Report Designer or the Media Script Editor.
- The Paging option is not used in global actions. A global action will always return all the records.
- You can't delete a global action once it is used in Report Designer or the Media Script Editor – you first need to remove the global action from the report definition or media script.

## Creating a global action

1. In CE Studio, click  in the sidebar and then click **Global Action Sets**.
2. Click **Create New Global Action Set**.
3. Enter a descriptive name for the global action set to make it easy to find this global action later in Report Designer or the Media Script Editor.

Each global action set will contain one global action.

4. Click **Add Action**:
  - a. Select one of the providers – you can choose from any of the providers in the system.  
You will always use the active version of the provider.
  - b. Select the data type.
  - c. Select the provider method.
5. If required, expand **Filters** and add a condition to filter the data that is returned by the provider. You can filter on:
  - Field reference
  - Static value
  - Global or system variable
  - A placeholder: you will then choose the property or value to filter on when setting up the report in Report Designer or working on the media script. Using a placeholder means that the global action can be used in multiple places as the person configuring the report or media script can choose which property to filter on.

For details of adding conditions, see [On selecting a row in the data grid](#).

6. Save the global action.

## Finding out where the global action is used

1. In CE Studio, click  in the sidebar and then click **Global Action Sets**.
2. On the Created Global Action Sets page, click **View Usage**.
3. Expand the global action set to see how it is used. The usage lists:
  - Name of the media configuration, such as voice or email queue progression script and the name of the version in which it is used
  - Name of the report definition

## Deleting a global action

Before you can delete a global action or the global action set, you need to remove the global action from the report definition or media script that uses it.

## URI and Link parameters

You can use both URI parameters and Link parameters in most types of CE Studio app to pass values:

- Into a CE Studio app from an external data source
- From a CE Studio app to an iFrame component
- Between CE Studio apps

**Note** Link parameters are not supported in public portal apps that use Public Portal Authentication.

	Dynamic URI parameters	Link parameters
Parameters and values are encrypted	No	Yes
Configuration	Component or App Settings	App Settings
Use as input in actions	Yes	Yes
Use in Navigate display tasks	Yes	Yes
Data source for action mapping	Field Value or UriParam	Session > App Data
Generating the URL	Navigate display task or generated by external system	Use methods in the IFS CE System provider to generate and revoke the links

### URI parameters

You can configure a CE Studio app to use the values passed as parameters in a query string that is appended to the CE Studio URL. Parameters and values are not encrypted and are therefore not secure.

You add URI parameters as properties in a component, such as a form. You can then use these as input to an action, rule and so on by mapping to them as field values. These are also available to map as placeholders in iFrame components.

To configure an app with dynamic URI parameters:

1. Add a component, such as a form and, in the field selector, select **Dynamic URI**.
2. Enter the parameter name. The lettercase is important, for example `Param1` is not the same parameter as `param1`.

3. Click **Done** to add the parameter as a property to the component.
4. Save the app. The name of the URI parameter is also the field identifier.
5. Configure the action(s) to use the URI parameters, for example in a filter:
  - a. In the Filters section of the action, add a new condition.
  - b. On the left side of the condition, for example, select **Field Reference**.
  - c. On the right side of the condition, select **Field Value** and map to the dynamic URI property. (Currently not possible to map using the **UriParams** source.)

You pass the parameter values in a query string appended to the URL of the app. For example, this is a public portal URL with a single URI parameter (?param1=m):

```
https://<tenant-front-end-path>/ce/0/public/?param1=m
```

This is an example with two URI parameters, separated by &:

```
https://<tenant-front-end-path>/ce/0/public/?param1=m&param2=true
```

## Link parameters

**Note** You can download an example that demonstrates link parameters and another app that generates the links. This is available from the Central Template Repository which is part of CE Studio Designer.

The screenshot shows two panels side-by-side. The left panel, titled 'Form', contains the following fields and controls:

- App Name:** \* (required) with a value of 'IFS CE Example Personalized Link App'.
- Expiry Time In Minutes:** \* (required) with a value of '5'.
- User Parameter:** \* (required) with a value of '307' and an information icon.
- Is Reusable:** A checkbox that is checked.
- Generate:** A blue button.

The right panel, titled 'Link details', contains:

- Link:** A text area containing a long, encrypted URL: `https://ce-uksouth1.ifsce-test.com/jm2/ce/0/public?appName=IFS%20CE%20Example%20Personalized%20Link%20App&_C=AAAbuBP5jqS8maZVGvM7+tlavW2ENHyO21X7VotcV1k+I9UHL0wriuEtmRO+dfOycMk/hmidOr4MFnyw2Al9FyIlg`.
- Link Reference:** A text area containing the value '1ee507e8-4292-4836-a9a5-be5' and a 'Delete' button.

Use Link parameters when you want to securely pass values from one CE Studio app to another, for example to personalize the data shown when a user accesses the CE Studio app. Values passed into the app using link parameters are encrypted and are part of the session data available to the app. You can use the values passed into an app as input to actions.

**Note** Link parameters are not supported in public portal apps that use Public Portal Authentication.

To configure link parameters:

1. On the Designer page, click **App Settings**.  
For an example, see the *IFS CE Example Personalized Link App* in the above download.
2. Go to the **Parameters > Links Parameters** tab.
3. Enter the parameter name. The lettercase is important, for example `Param1` is not the same parameter as `param1`.
4. Click **Done** to add the parameter(s) as a property of the app's session data.
5. Save the app.
6. Configure the action(s) to use the Link parameters, for example:
  - a. In the Filters section of the action, add a new condition.
  - b. On the left side of the condition, for example, select **Field Reference**.
  - c. On the right side of the condition, select **Session > AppData** and map to the Link parameter.

To generate the URLs that use the parameters, you need to generate links that encrypt the parameters and their values. You do this using the IFS CE System provider (version 6 and later) and the `SecuredLink` method `GenerateSecuredLink`.

### Generating links that use Link parameters

**Note** For an example, see the *IFS CE Example Generate Secured Link App* in the above download.

To generate links containing the link parameters, you need to configure a CE Studio app to generate the links. The `SecuredLink` method `GenerateSecuredLink` has the following optional parameters and you can also add custom parameters:

Parameter	Description
<code>AppName</code>	Name of the target CE Studio app
<code>AuthenticationMode</code>	Always set to 0
<code>ExpiryInMinutes</code>	Minutes before the link expires
<code>IsReusable</code>	A Boolean. When <i>true</i> , the link can be reused up until it expires. When <i>false</i> , the link can be used once only.

To add additional custom parameters:

1. Edit the action with the `GenerateSecuredLink` method.
2. Click **Manage Properties**.

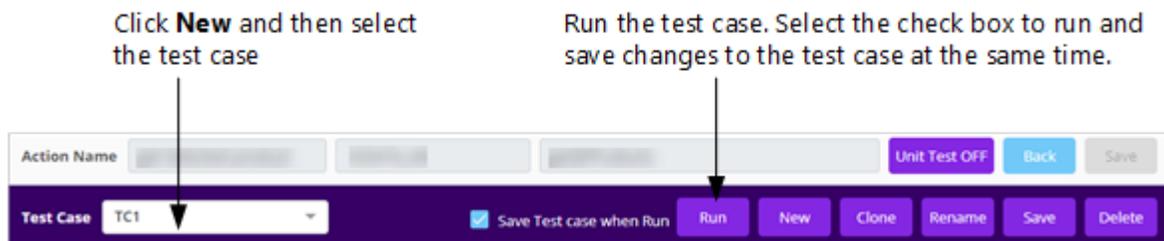
- Click the field next to GenerateLinkRequest and then select **Parameters**.  
Do not select **User Credentials**.
- Enter the parameter name as configured in the target app.
- Select the correct data type for the parameter.
- Map these parameters in the action or to fields in a form that you want to use for generating multiple links.

## Unit tests

You can mock the execution and responses for all the actions, global actions, and rules in a CE Studio app. You can then use these on the Actions page to test the behavior of actions and action sets.

For an example, download the Actions sample template from the Central Template Repository in CE Studio Designer.

There is a unit test available for each action to which you can add any number of test cases. You can run each test case individually or, from the action set, you can run all the test cases in the unit test with the same name. This will also run the data transformations configured on the action set.



*Toolbar for creating, managing, and running the test cases in a unit test*

How you configure the test cases will depend on the type of action and the provider type.

**Note** For example, unit tests are not available in IFS Cloud for write operations such as PATCH, UPDATE and DELETE.

### Mocking create and update actions

For actions with parameters that are mapped, you can add mock data for each parameter. You need to supply mock data for all the parameters configured in the action apart from those mapped to static values. If you leave parameters with static values empty, then the test case will run with the configured values. When you run the test case, you will see the response in a JSON viewer.

You can mock the parameter mapping for create actions

Toggle **Show Mock Data** to see the parameters or enter and edit mock data for each mapped parameter

The screenshot shows the 'Parameter Mapping' window. It is divided into two main sections: 'Suggested Parameters' and 'Optional Parameters'. Both sections have a 'Show Mock Data' toggle switch. In the 'Suggested Parameters' section, the toggle is currently off. Below it, there are four parameter rows for 'studioservicetype : ODataDemo-Product': 'ID : number' with a 'selectedProduct' dropdown and 'ID (as number)' input; 'Price : decimal' with a 'selectedProduct' dropdown and 'Decimal-turbuqrsu' input; 'Rating : number' with a 'selectedProduct' dropdown and 'Rating (as number)' input; and 'ReleaseDate : date' with a 'Now' button. In the 'Optional Parameters' section, the 'Show Mock Data' toggle is turned on, and there are three empty input fields for 'Description : string', 'DiscontinuedDate : date', and 'Name : string'.

*Mocking suggested and optional parameters for a create or update action*

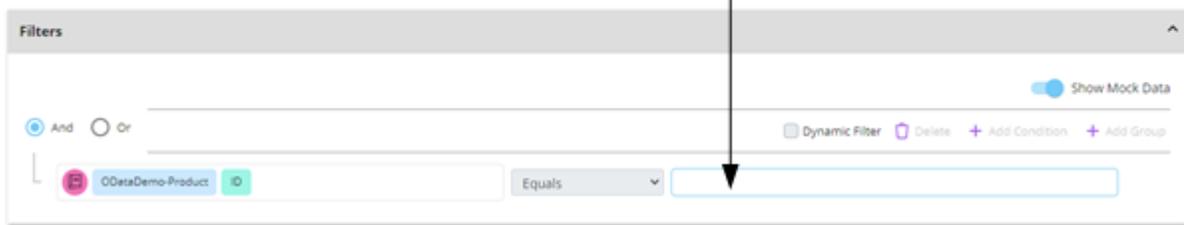
**Important** When you run the test case, you will send a request to the endpoint and will therefore create a new record or modify an existing one with the mock data.

**Note** You cannot run unit tests for update and patch operations for IFS Applications 10/IFS Cloud and oData providers. These actions require an entity cache (CEContext) along with the request which is provided from the server with the corresponding get response. The unit test will therefore return the error `Object reference not set to an instance of an object`.

### Mocking read actions

For read actions that request specific data, rather than all the data for the provider type, you can mock the conditions or dynamic conditions used in the request. If a condition is mapped to a static value, then you can mock the static value or you can leave this empty to use the configured value. When you run the test case, you will see the response from the provider, in a tabular format.

For read actions, you can mock the data used in conditions and dynamic filters



*Mocking the condition(s) in a read action*

**Note** If there is no condition in the action then there is nothing to mock.

Read actions publish a topic and will therefore have subscribers. You use the Topic Mock Data pane to mock the response from the endpoint.

### Mocking the response

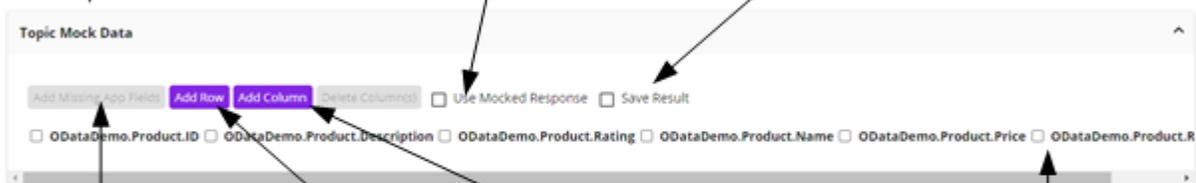
You can mock the response when:

- No provider is available
- Suitable data isn't available, for example, when configuring data transformations

For actions that publish a topic and therefore have subscribers, you can mock the response data

Use a mocked response rather than invoke the provider

Saves the response as part of the test case for use by data transformations



Click **Add Missing App Fields** if any properties are not listed in Topic Mock Data (for example, you edit the app after creating the test case)

Click **Add Row** to enter the mock data

Click **Add Column** if you want to test the action with additional properties

Properties used in the CE Studio app

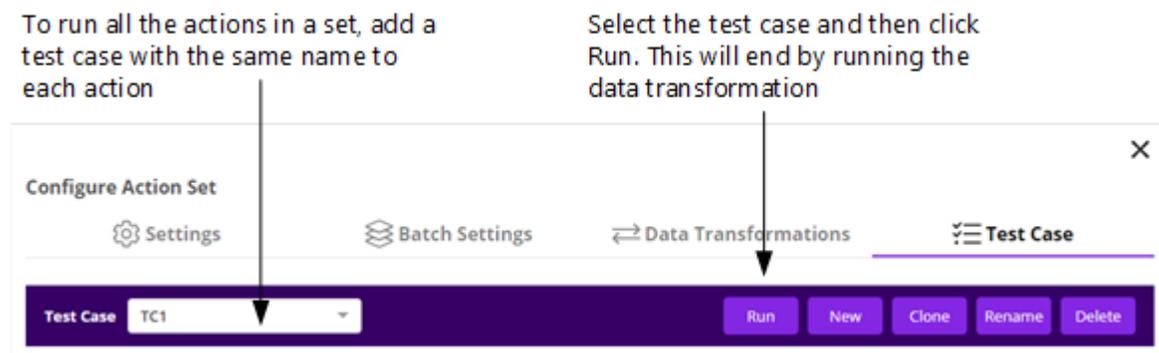
*Mocking the response*

There is one mocked response for each test case. You can decide how to use the mocked response:

- Deselect **Use Mocked Response** to use the data returned from the provider. After running the test case, you can save this data as input to a data transformation by selecting **Save as Mocked Data**.
- Select **Use Mocked Response** to use the mock topic data as the response. After running the test case, the response will contain the mock data. Select **Save as Mocked Data** to save this data as input to a data transformation.

### Mocking the actions in the action set and running the data transformation

You can run all the actions in the action set if each action contains a test case with the same name. After running the test case, the data transformation will run. The data transformation will use the response from the provider or the mocked response depending on how you chose to configure the action upon which the data transformation depends.



*Running all the actions in an action set*

### Mocking a global action

When mocking a global action, you need to select the properties you want to use in the topic mock data.

### Mocking rules and rule sets

You can also mock rules and rule sets. A test case for a rule will always return either *true* or *false*. Similarly, running all the test cases in the rule set will evaluate all the rules and then return either *true* or *false*. The result is shown in a JSON viewer.

## Troubleshooting actions

This topic describes how to use the Developer Tools provider by the browser to troubleshoot issues with CE Studio apps.

### Finding out which action in an action set is failing

When there are multiple actions in an action set then an OnFailure response from the provider will prevent all the actions in the set from running. Use the Developer Tools in your browser to work out which action is failing.

1. In Chrome, open the Developer Tools.
2. In the **Name** column, click the last invoke.
3. Go to the **Preview** tab.
4. Expand `responsesInOrder` and then expand each response (action).
5. The `outcomeStatus` is either `OnSuccess` or `OnFailure`. For `OnFailure`, there will be an error message.

Name	×	Headers	Payload	Preview	Response	Initiator	Timing	Cookies
<input type="checkbox"/> RuntimeAppVers...	▼	{displayTaskIdsInOrder: [],...}						
<input type="checkbox"/> Invoke?localeCo...	▼	{displayTaskIdsInOrder: [] entityCache: null hashedEventHistory: null outcomeStatus: "OnFailure"						
	▼	responsesInOrder: [{providerId: null, topicId: 0, topicData: null, topicPagination: null, actionInfo: null, entityCache: null, errorMessage: "Error calling Invoke Studio Service Rpc\n500: {\"error\":{\"code\" outcomeStatus: "OnFailure" providerId: null ruleInfo: null specificTypePrefix: null topicData: null topicId: 0 topicPagination: null						
	▼	0: {providerId: null, topicId: 0, topicData: null, topicPagination: null, actionInfo: null, entityCache: null, errorMessage: "Error calling Invoke Studio Service Rpc\n500: {\"error\":{\"code\" outcomeStatus: "OnFailure" providerId: null ruleInfo: null specificTypePrefix: null topicData: null topicId: 0 topicPagination: null						
	▼	1: {providerId: 44, topicId: 2826,...} actionInfo: {actionId: 2826, requiredContext: [], injectedFilterIds: [],...} actionId: 2826 injectedFilterIds: [] requiredContext: [] sorts: [{qualifiedName: "IfsApp.BusinessActivityHandling.BusinessActivityCampa entityCache: {18877 107: {,...}, 18877 106: {,...}, 18877 105: {,...}, 18877 104: {,...} errorMessage: null outcomeStatus: "OnSuccess" providerId: 44						

### Identifying failing actions

### How to view the topic data in the response from the provider

You can use the Developer Tools in your browser to verify whether an action is returning any topic data.

1. In Chrome, open the Developer Tools.

2. In the **Name** column, click the last invoke.
3. Go to the **Preview** tab.
4. Expand `responsesInOrder` and then find the response (action).
5. Expand `topicData`.

For example, for a data grid there will be an entry for each row on the page (default page size is 10) and one entry for a form.

Name	× Headers	Payload	Preview	Response	Initiator	Timing	Cookies
<input type="checkbox"/> RuntimeAppVersion?loc...	▼ {displayTaskIdsInOrder: [],...}						
<input type="checkbox"/> Invoke?localeCode=en-...	<pre> displayTaskIdsInOrder: [] entityCache: null hashedEventHistory: null outcomeStatus: null ▼ responsesInOrder: [{providerId: 44, topicId: 2795,...}, {providerId: 46, topicId: 2804,...}]   ► 0: {providerId: 44, topicId: 2795,...}   ▼ 1: {providerId: 46, topicId: 2804,...}     ► actionInfo: {actionId: 2804, requiredContext: [], injectedFilterIds: [],...}     ► entityCache: {19679 0202: {...}, 19679 25: {...}, 19679 24: {...}, 19679 32: {...}}     errorMessage: null     outcomeStatus: "OnSuccess"     providerId: 46     ruleInfo: null     specificTypePrefix: null     ▼ topicData: [{IfsApp-BusinessLeadHandling-BusinessLead SourceId: null,...},...]       ► 0: {IfsApp-BusinessLeadHandling-BusinessLead SourceId: null,...}       ► 1: {IfsApp-BusinessLeadHandling-BusinessLead SourceId: null,...}       ► 2: {IfsApp-BusinessLeadHandling-BusinessLead SourceId: "Id1",...}       ► 3: {IfsApp-BusinessLeadHandling-BusinessLead SourceId: "Id3",...}       ► 4: {IfsApp-BusinessLeadHandling-BusinessLead SourceId: "Id3",...}       ► 5: {IfsApp-BusinessLeadHandling-BusinessLead SourceId: "Id1",...}       ► 6: {IfsApp-BusinessLeadHandling-BusinessLead SourceId: null,...}       ► 7: {IfsApp-BusinessLeadHandling-BusinessLead SourceId: "Id2",...}       ► 8: {IfsApp-BusinessLeadHandling-BusinessLead SourceId: "Id3",...}       ► 9: {IfsApp-BusinessLeadHandling-BusinessLead SourceId: "Id4",...} </pre>						

*Viewing the topic data in the response*

### Subscription issues: No properties have been requested

Any component, element or UI element that subscribes to an action must be configured with an appropriate data type matching the action response.

For example, if a component containing a Text Area needs to subscribe to an action that returns a message text then the data source of the Text Area must be set to the appropriate provider type.

If no provider type is set on the Text Area then this error is displayed: No properties have been requested.

# 8

## Display tasks

You configure display tasks to change the state of the user interface. Display tasks are always run before any action sets or rule sets.

A display task always belongs to a display task set, and you can use this to group related display tasks.

You can run display tasks on any event status: OnInvoke, OnSuccess, OnFailure. This means that the action set or rule set for the OnInvoke status must return a success or failure response before the display tasks for the OnSuccess or OnFailure event can run.

**Note** There are some sample templates available for download from the Central Template Repository that demonstrate the use of display tasks.

### Using display tasks

---

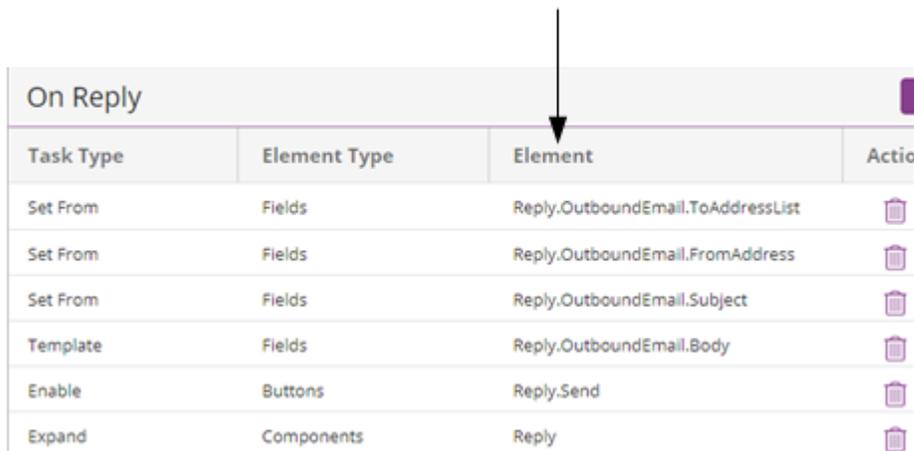
Display tasks can run:

- After an event is invoked, succeeds or fails. For example, an information message on success and an error message on failure.
- After a rule in a rule set evaluates to *true* or *false*. For example, on success, a display tasks shows or expands a component and updates all the fields.
- Before an action set is run (OnInvoke status) and all the subscribers have been notified. A display task attached to an action set will always run before the actions in that action set.

You can configure display tasks for components, fields, buttons and toast messages. A display task is completely self-contained and independent of the context in which it is run.

The display tasks that must run when a particular event occurs are grouped in a display task set. The first display task in the set runs first. To change the order in which they run, use drag and drop.

Display tasks apply to the components, fields and buttons in this column



On Reply			
Task Type	Element Type	Element	Action
Set From	Fields	Reply.OutboundEmail.ToAddressList	
Set From	Fields	Reply.OutboundEmail.FromAddress	
Set From	Fields	Reply.OutboundEmail.Subject	
Template	Fields	Reply.OutboundEmail.Body	
Enable	Buttons	Reply.Send	
Expand	Components	Reply	

*Example of how a display task set groups the display tasks for a component*

**Note** To create the display tasks for an event, click **Create new display task set** and name the set. To add the first task to the set, click **Edit** and then **Add new task**.

### Task types

The following table summarizes the available types of display task, and how you apply them.

Task	Element Type	Notes
Set From	Fields	Updates a field with the value from another field.
Set To	Fields	Updates a field to display a default or initial value.
Combined	Fields	See <a href="#">Combined display tasks</a> .
Clear Selection	Fields	For batch-enabled data grids. Clears the selection made by the user.
Template	Fields	For setting a field to a value in an embedded template.

Task	Element Type	Notes
Enable, Disable	Buttons, Fields	For making buttons and fields available or unavailable depending on the state of the CE Studio app.
Clicks	Buttons	For generating an automatic button click.
Reset All Fields	App, Components	Resets all fields with all the components in the app or within a single component.
Enable Timer, Disable Timer	App, Components, Elements (Fields)	For use with <i>timers</i> .
Show, Hide	Components (not overlays), Tabs, Buttons, Fields	See <i>Expand, Collapse, Show, Hide display tasks</i> .
Expand, Collapse	Components	See <i>Expand, Collapse, Show, Hide display tasks</i> .
Toggle Component	Overlays	<i>Overlay components</i> only
Active	Tabs	See <i>Active display tasks</i> .
Goto Step	Stepper	Specifically used with stepper components to go to the current step.
Prompt	App	For toast messages. See <i>Prompt display tasks</i> .
Refresh Window	App	To refresh the browser tab.
Navigate	iFrame components	See <i>Navigate display tasks</i> .
Reset Control	File upload and data grids	To reset or clear the upload of files and data grids.

## Active display tasks

Use the Active display type to:

- Make a tab in a tab group into the active tab.

- For the default tab of a tab group, bring the data into the tab as part of an OnLoad event.

## Combined display tasks

---

Use the **Combined** display type to update a field with a concatenated value derived from other fields in the app.

### Combined

Target Element

ProductCode

Add Field

Field: selectedProduct  + Field: selectedColor 

When you add a Combined display task the values are combined without any separator.

## Expand, Collapse, Show, Hide display tasks

---

### Expand, Collapse

If the component is already expanded, it collapses the component leaving just the title visible and all other content hidden.

The component must be configured as expandable (the default setting). To make a component expandable, on the Designer page, click  and go to the **Settings** tab.

### Show, Hide

Hidden elements are still available in any rules or actions that need them, and OnLoad events are still triggered.

**Note** Not intended for use with overlays. Use the Toggle Component display task instead.

## Navigate display tasks

You can configure navigate display tasks for URLs, including URLs to other CE Studio apps. The page can open in the same tab, a different tab or an iFrame.

**Note** For display tasks that perform a sign out, as with portal apps, then the redirect URL should open in the same tab.

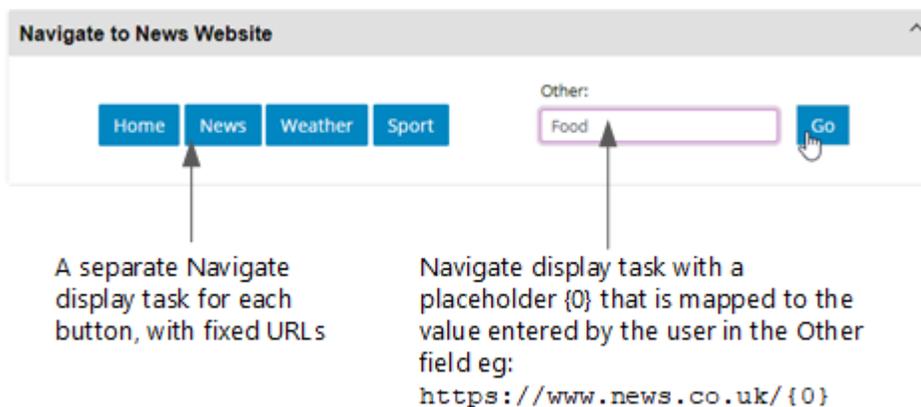
### Navigate display task and iFrames

You can configure an iFrame to embed a webpage, play a video and so on. You can either add the URL as a property of the iFrame component or configure a Navigate display task for the URL. Not all URLs can be loaded in an iFrame:

- For URLs that can be loaded in an iFrame, you can map the URL to an iFrame component.
- For URLs that cannot be loaded, you open them in the same browser tab or in a separate tab.

### Navigate display task with placeholders

You can configure the Navigate display task with a placeholder so that the user can select the URL to go to, as shown in the following example:



*Example of Using the Navigate Display Task with a Placeholder*

### Limitations

When using Navigate display tasks with iFrames:

- The embedded page cannot pass commands to the CE Studio app. The embedded app is fully functional but is independent of the parent app.

## Prompt display tasks

---

### Prompt

Define any messages needed by the top-level container, whether an app or a template. These are displayed as toast messages.

For example, if you are creating an app that embeds a template then you can add messages for the app. At runtime, the app may display messages from both the app and the embedded template.

You can configure the following types of message — you decide how to use the different message types:

Message type	For example, attach display task to...
Information	OnLoad event with status OnSuccess
Warning	OnLoad event with status OnFailure
Error	OnLoad event with status OnFailure  <b>Note</b> Will display in addition to the error message sent from the endpoint.
Success	OnLoad event with status OnSuccess

When you configure a message, you can either enter the text as a hardcoded string or you can click  and enter a message ID from any of the providers in the app or template.

**Note** In Agent Desktop, prompt display tasks are not visible when the Monitoring page is open.

## Clicks display task

---

**Note** You can use this display task as an alternative to configuring an App State Change event. See [App state change event](#) for details.

Use the Clicks display type to automatically run the action configured in an OnClick event for a button.

Running an action in this way is useful in a media app which is the default home page app in Agent Desktop. When an agent wraps up an activation, they go back to the home page app. The home page app is not reloaded so there is no OnLoad event to run an action, however, you may need to update the app in some way.

This is how to run an action in the above scenario:

1. Add a button - this can be a hidden button.
2. Configure a display task set with a Clicks display task. The target of this display task is the button.
3. In Event Configuration, configure an OnClick event to run the required action.
4. Add a rule set to run the Clicks display task. Do not select the action here as it will not run.
5. In Event Configuration, configure an event to run the rule set when the agent goes from an activation back to the home page app. For example:
  - Select the OnToolbarEventReceived type.
  - Select the EVT\_AGENT\_STATE\_CHANGE event.
  - Select the rule set.

# 9

## Portals and public portals

Each tenant has a portal for authenticated users and a public portal for anonymous (unauthenticated) users.

**Note** For public portals and the *default* public portal app, you can configure a custom domain. For IFS Customer Engagement only, you upload the certificate and configure the domain in the Admin Portal.

### Portals

A portal lets end-users, such as customers, suppliers, partners, sign on to view and update their information. This type of portal can:

- Allow users to create, modify and delete data in external data sources.
- Provide links to other pages in the same portal. Each page is a separate CE Studio app.
- Embed other third-party webpages (if the webpages allow themselves to be embedded in an iFrame)

Each tenant has one portal and users must register with the portal before they can sign on to it.

For full details, see [Portals for registered users](#).

### Public portals

Anyone can access a public portal - users do not sign on to the portal and the system provides no way of controlling who has access to it. However, you can configure two-factor authentication, one-time tokens and so on, in order to control access to the site.

See [Authentication for public portals](#) for details.

You can use unauthenticated public portals to, for example:

- Gather information, such as feedback

- Let users submit requests, such as quotations, appointments, brochures
- Provide information to users

You can embed the public portal URL in emails, on web pages and so on. Anyone who has the URL can access the public portal so you should carefully consider the type of information you show in public portal apps.

For full details, see [Public portals for anonymous users](#).

## Portals for registered users

---

Each tenant is preconfigured with a single portal that is ready to use. The portal is accessed through its own URL and authorization is managed separately from access to Customer Engagement applications (Admin Portal, CE Studio Designer).

You need to set up:

- The CE Studio apps that allow registered portal users to view and update their information.
- The landing page to which portal users are redirected after signing on. You can set up a different landing page for each group of users.
- The user groups for controlling who has access to the portal and which CE Studio apps they can see.

Portal setup involves the following steps in CE Studio Designer:

1. Create a user group for each type of user. All users who sign on to the portal must belong to one of these user groups.
2. Create the landing page for the portal. You create it as a standard app. If you require different landing pages for the different user groups then you may prefer to create the landing page as a template.

See [Creating portal apps to use as landing pages](#).

3. Depending on the complexity of the portal, you may need to create separate portal apps. However, if the portal is very simple then you may be able to add the features you require to the landing page itself.

Create any portal apps as templates. You can use any of the features of CE Studio, except for those that depend on current activations.

See [Creating portal apps](#).

4. If you created multiple portal apps then add links to let users navigate between them.

See [Adding links to portal apps](#).

5. On the Portal Settings page, select the app that will be used as the landing page for the portal. You can add multiple landing pages. All portal users will use the default landing page unless a different landing page is set in their user profile.

See [Configuring the portal settings](#).

6. Test the portal. You can preview the landing page(s) in CE Studio Designer, but you must test everything in the tenant's portal realm.

See [Testing the portal](#).

7. Optional. You can change the theme of the signup page seen by portal users.

Please contact IFS Support if you want to change the theme of the signup page. They will send you a zip file containing the resources that need to be updated.

## Configuring portal access groups

The access group determines which apps and landing page a user sees when they sign on to the portal. Different groups of users can see a different landing page and different apps. You can move both apps and people from one access group to another.

You need to create the access groups as user groups in the Admin Portal. You can set up more than one user group. Whoever administers the portal will be responsible for adding and managing the users in these groups.

### Adding an access group

To add an access group:

1. Go to the Admin Portal for the tenant.
2. In the side menu, go to **User Management > User Group**.
3. In the User Group Detail pane, enter the details of the new access group.

<b>Group Name</b>	This becomes the name of the access group in CE Studio.
<b>Contact Center</b>	Select any contact center if there is more than one. The access group does not use this.
<b>User</b>	Select the individual users who belong to the access group.
<b>Sub Group</b>	You can also add a group of users to this access group.

4. Refresh CE Studio Designer if it is open.

You can now create a portal app and select this user group as the access group for that portal app.

### Changing the access group

To change the access group for a portal app:

1. In CE Studio Designer, go to the Apps page.
2. Click **Edit** for the portal app you want to update.

### 3. Select a different access group from the **Access Group** list.

You may need to refresh CE Studio Designer if you have just added a new user group in the Admin Portal.

## Creating portal apps

You can add CE Studio apps to the portal. Before you start there are several things to remember:

- Create the portal apps as templates. Only templates can be added to the app selected as the portal landing page.
- Navigation from the portal landing page to the portal apps in the same access group as the signed-in user is provided by a *Portal List Template*. This means that you do not need to add any links yourself. However, you do need to add navigation from each portal app to the landing page (see below).
- Portal apps require a sign out button. You can either add this to each portal app or to a template that you use as the banner on all your portal pages (see below).
- Portal users will not have an Azure Active Directory user account so they cannot sign on using their SSO credentials. If the provider is SSO enabled then an *integration user account* must exist at the external data source.

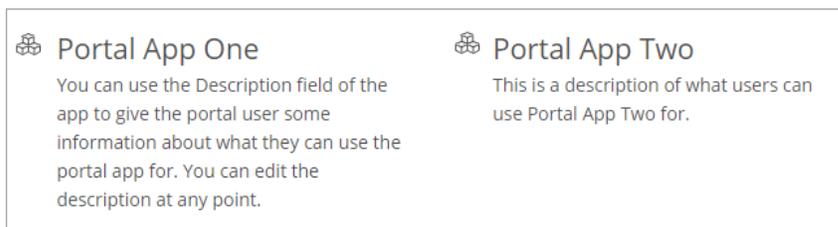
You can create an unlimited number of portal apps for a portal.

### Creating portal apps

To create a portal app in CE Studio Designer:

#### 1. Create the portal app as a template.

The access group determines who can access this app, and the name and description will be shown like this:



For details of how to create a template, see *Creating a standard app or template*.

#### 2. Build the portal app in the usual way. You can use any of the components and elements available in CE Studio Designer.

### Adding navigation from the portal app back to the landing page

Use a Navigate display task to let users navigate from portal apps to the landing page of the portal:

1. In CE Studio Designer, click  to go to the Display Task Sets page.
2. Click to create a new display task set.
3. Add a new display task to the set.
4. Select **Navigate** as the task type.
5. Enter the relative URL of the landing page. This should take the user back to the *same* tab.

For example, if the URL of the landing page is:

```
https://<environment-name>/<tenant-name>/ce/portal/
```

then you can enter the URL as:

```
/<tenant-name>/portal?appName=<app-name>
```

6. You can then add a button group, and configure an OnClick event to run the display task.  
See [About components and elements](#) and [Event configuration for components](#).

### Adding a logout button

**Important** Logout URLs using this format: `auth/logoff/?redirecturi=` are no longer supported in 6.1.3. You must now update this to `${logouturl}?redirecturi=`. See below for details.

Use a Navigate display task to send the user to an app belonging to the tenant when they click the logout button, for example, to an unsecured public portal app. The display task must redirect the portal user back to the same browser tab. The portal logout URL is a relative URL. It has a fixed part `${logouturl}?redirecturi=` and then the path to go to:

```
${logouturl}?redirecturi=<redirect-url>
```

For example, if you wanted to use `www.ifs.com` (the default for all tenants) as the page to go to on logout then you would:

- In the Navigate display task, enter the URL like this:

```
${logouturl}?redirecturi=https://www.ifs.com/
```

- Or like this for the tenant's default public portal app:

```
${logouturl}?redirecturi=https://<test>/<tenant>/ce/0/public/
```

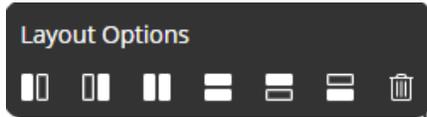
See also [Public portal URLs](#).

You may prefer to create a [template](#) with a logout button and then add the template to the landing page and the portal apps.

### A note about page layout

When you add a component to a CE Studio app, the component will be displayed at the full width of the app. If it is the first component in the app then it will be positioned at the top of the page.

To create space around the component, you can add empty rows or cells to the layout using the Layout Options toolbar:



### Adding links to portal apps

To link to the other portal apps in the same access groups as the current signed on user, you use a Portal List Template. You can use this element in any portal app. Each portal app in the same access group appears like this at runtime:



*How Portal Apps are Displayed at Runtime*

### Add an action to get the accessible portal apps

1. Add the System provider to the portal app. You need to use version 2 or later.

If you do not know how to add providers follow the steps in [Adding providers to an app](#).

2. Add the action to get the portal apps that are in the same access group as the current signed on user:

**Note** See [Adding actions and configuring events](#) if you do not know how to configure action sets and actions.

Provider	Select the System provider.
Type	Select PortalApp.
Method	Select the method GetAccessiblePortalApps.

3. Select the **Publish Topic** option.

The action must publish the topic (the Portal List Template will subscribe to this using the OnNotification event).

No filtering, sorting or paging is required in the action.

### Configure the portal links on the page

To configure the portal links on the page:

1. On the Designer page , add a Portal List Template. Add this as an element inside an existing component.

There are various ways of doing this, for example, you could add it to a headerless stack panel (the stack panel could provide a styled heading):

- a. Add a row below or above the stack panel.
  - b. Add the Portal List Template.
2. Configure the Portal List Template:
    - a. Select the option **Portal App** from the Templates list. This option will provide links to all the portal apps in the access groups to which the signed-in user belongs.
    - b. Fill out the details:

Name	Click in the <b>Name</b> field. Select the <i>System</i> provider. Select <code>PortalApp</code> as the type. Select <code>Name</code> as the property.
Description	In the same way, map the <b>Description</b> to <code>System &gt; PortalApp &gt; description</code> .
Guid	In the same way, map the <b>Guid</b> field to <code>System &gt; PortalApp &gt; AppVersionGuid</code> .

- c. Click **Done**.

The next step is to configure the OnLoad and OnNotification events.

### Configure the OnLoad and OnNotification events

1. On the Event Configuration page , configure the event that will load the accessible portal apps:
  - a. On the left, click the outermost container. **App** is shown as selected on the right.
  - b. From the **Event Type** list, select **OnLoad**.
  - c. From the **Run Action Set** list, select the action you configured above.
2. The Portal List Template must subscribe to this topic:
  - a. On the left, click the box representing the Portal List Template.
  - b. From the Event Type list, select **OnNotification**.

- c. Select the action you configured and click **Subscribe** and then click **Done**.
3. Save the portal landing app.

## Creating portal apps to use as landing pages

In a portal, the landing page is the first CE Studio app that users see after signing onto the portal. If you have multiple access groups then you will need to create a landing page for each access group.

We recommend that you create the app for the portal landing page as a standard app rather than as a template.

**Note** If you need to re-use the landing page multiple times then you could consider creating the app as a template. For background information, see [Working with templates](#).

Apart from creating the landing page as a standard app, there are no other differences between the CE Studio apps that you create for portal apps and the CE Studio apps that you create for portal landing pages – the portal landing page is simply the CE Studio app that you choose to use as the landing page. You set this on the Portal Settings page in CE Studio Designer.

**Note** For information on how to create portal apps, see [Creating portal apps](#).

## Configuring the portal settings

Once you have created the portal landing page (or pages if there is more than one access group), you need to go to the Portal Settings page in CE Studio Designer and select the portal app designed as the landing page for the portal. **Only live portal apps are listed here.**

**Note** The Portal Settings page only lists the configured portal landing pages for the tenant's portal. A full list of portal apps is available on the Apps page in CE Studio Designer.

**Note** Ignore the **Default Public Portal Page** list. You use this when configuring public portals.

To configure the portal landing page(s):

1. In CE Studio Designer, go to the Portal Settings page.

When fully configured, there will be one row for each access group.

2. Click **Add Portal Category** and then enter its details:

Category Name	A descriptive name for internal use. This will not be displayed in the portal.
Portal App	Select the portal app that you want to use as the landing page for this category. If any portal apps are missing from the list, then check whether you made these live. You can change this later by selecting a different app from the Portal App list on the Portal Settings page.
Description	A descriptive name for internal use. This will not be displayed in the portal.

- If you only have one access group, then the app you set as the landing page will be the default landing page. If you have several access groups, then you need to select one of the landing pages as the default.

This landing page will be seen by portal users who do not have a default landing page configured in their user profile. However, they must belong to the same access group as the selected portal app.

## Testing the portal

**Important** Always test the portal in incognito mode. Always start a new session when troubleshooting access problems - do not reuse the same browser tab.

You can preview the portal in CE Studio Designer, but you must test it in the portal realm. The URL will be similar to this:

`https://<environment-name>/<tenant-name>/ce/portal/`

- Make live the version of the landing page and the portal app(s) that you want to use.
- Make sure that one of the portal apps is set as the landing page on the Portal Settings page. See [Configuring the portal settings](#).
- Create test users for yourself – you need one user for each access group that you want to test:
  - The **UserID** is your email address. You need a separate email address for each access group. You can't use the email address that you are use to sign on to CE Studio Designer.
  - The user role is **Portal User**.
  - Set one of the landing pages as the default landing page. If you leave this empty, the user will automatically use the default landing page as set on the CE Studio Designer Portal Settings page.
- In incognito mode, go to the URL for the tenant's portal.

The sign up page is displayed.

5. Enter the email address of the user you created and then click **Sign Up**.

A verification email is sent to the email address you used. This is only valid for 5 minutes.

6. In the email you receive, click the link to verify yourself to the system. You can request another verification email if the first one times out.
7. Go to the portal URL for the tenant, sign in and you should see the landing page that you set as the default for this user. Test that:
  - All the apps in the same access group as the signed-in user are accessible.
  - You can navigate between the apps.
  - Apps work as intended.
  - You can sign out correctly from anywhere in the portal.

## Public portals for anonymous users

---

Each tenant has a single public portal that is ready to use.

**Note** However, the maximum number of public portal sessions must be configured for the tenant. This is 0 by default. You request this through IFS Service Center. No more users can access the public portal once the maximum number of sessions is reached. Sessions end when the user closes the browser window, or when the tab is idle for more than 5 minutes. It takes approximately one minute for a user's session to be cleared from the cache.

In this release, a new session is started each time a user clicks a quick link to go to another app within the public portal. This means that the same person may have several sessions. This will be addressed in a future release.

You can design one or more CE Studio apps for use as public portal apps. See [Configuring public portal apps](#).

The public portal is accessed through its own URL. See [Public portal URLs](#). The default app for the portal is configured on the Portal Settings page. This type of portal is for unauthenticated users so to test a public portal you must use your browser in incognito mode.

Access is for anonymous users and, by default, anyone who has the public portal URL can access the portal. You can implement your own solution for authenticating users, for example, you can send secure links or one-time passwords.

See [Authentication for public portals](#) for details.

## Configuring public portal apps

To configure a public portal app for your tenant:

1. Create a CE Studio app, selecting **Public Portal** as the app group.

2. Add the IFS CE System provider (version 7 or later) and any other providers that you need.

You cannot add the Agent Desktop provider.

3. Design the CE Studio app. You can use any combination of templates, components, elements, and UI elements.

You must create the templates using the **Public Portal** app group. You cannot use templates created for other types of app.

4. Make the CE Studio app live.

5. Go to the Portal Settings page and select the public portal app from the **Default Public Portal Page** list.

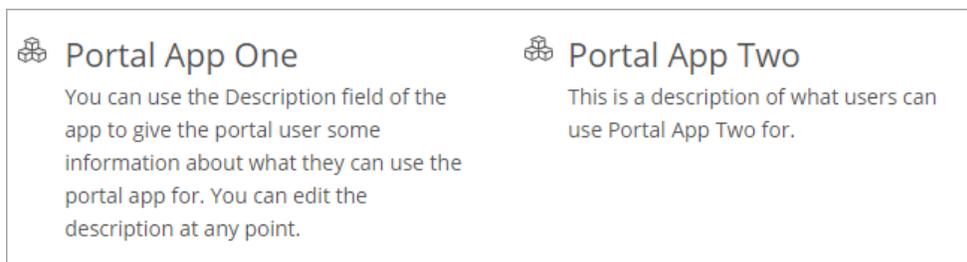
This app is used if the public portal URL doesn't include the name of the app (see below).

### Configuring additional public portal apps

Each page in the public portal is configured as a public portal app. To add further pages, you create additional public portal apps. To link between pages in the public portal app, you use the Portal List Template element. See [Adding links to public portal apps using a Portal List Template](#).

## Adding links to public portal apps using a Portal List Template

To link to the other apps in the public portal, you use a Portal List Template element. Each app in the public portal appears like this at runtime:



### *How Portal Apps are Displayed at Runtime*

**Important** In this release, a new session is started each time a user clicks a quick link to go to another app within the public portal. This means that the same person may have several sessions. This will be addressed in a future release.

### Create a template

If you have several public portal apps then it is quickest to add and configure the Portal List Template element once in a template and then insert that template into each public portal app. However, you can also add the Portal List Template element directly to a public portal app.

To create a template containing a Portal List Template element:

1. Create a new template, selecting **Public Portal** as the app group.
2. Add the IFS CE System provider. You need to use version 7 or later.

If you do not know how to add providers follow the steps in [Adding providers to an app](#).

### Add an action to get the accessible public portal apps

1. Add the action to get the public portal apps:

**Note** See [Adding actions and configuring events](#) if you do not know how to configure action sets and actions.

Provider	Select the IFS CE System provider.
Type	Select PortalApp.
Method	Select the method GetAccessiblePublicPortalApps.

2. Select the **Publish Topic** option.

The action must publish the topic (the Portal List Template will subscribe to this using the OnNotification event).

No filtering, sorting or paging is required in the action.

### Configure the Portal List element

To configure links between all the live public portal apps:

1. On the Designer page , add a Portal List Template element.
2. Configure the Portal List Template:
  - a. Select the option **Public App** from the Templates list. This option will provide links to all the live public portal apps on the tenant.
  - b. Fill out the details:

Name	Click in the <b>Name</b> field. Select the System provider. Select PortalApp as the type. Select Name as the property.
Description	In the same way, map the <b>Description</b> to System > PortalApp > description.

- c. Click **Done**.

The next step is to configure the OnLoad and OnNotification events.

### Configure the OnLoad and OnNotification events

1. On the Event Configuration page , configure the event that will load the live public portal apps:
  - a. On the left, click the outermost container. **App** is shown as selected on the right.
  - b. From the **Event Type** list, select **OnLoad**.
  - c. From the **Run Action Set** list, select the action you configured above.
2. The Portal List Template must subscribe to this topic:
  - a. On the left, click the Portal List Template.
  - b. From the Event Type list, select **OnNotification**.
  - c. Select the action you configured and click **Subscribe** and then click **Done**.
3. Save the public portal app.

### Using the template

Make the template live and then add the template containing the portal list template to each public portal app.

## Public portal URLs

**Important** You need to use an incognito session in the browser to fully test your public portal apps. If you are unable to start a session, then check whether the maximum number of sessions has been reached.

The maximum number of public portal sessions must be configured for the tenant. This is 0 by default. You request this through IFS Service Center. No more users can access the public portal once the maximum number of sessions is reached. Sessions end when the user closes the tab or browser window, or when the tab is idle for more than 10 minutes. It takes approximately one minute for a user's session to be cleared from the cache.

The server returns a 401 when a session expires and a 503 when the session cannot be started because the maximum number of sessions is exceeded.

If you use the URL without a named app, that is: `https://<test>/<tenant>/ce/0/public/` then the user will go to the app selected as the default public portal app on the Portal Settings page. Users will always go to the default app if the URL is invalid.

### URL for the default public portal app

If the URL of the public portal app in CE Studio Designer is:

`https://<test>/<tenant>/ce/0/studiodesigner/`

then the URL of the default app for the public portal is:

```
https://<test>/<tenant>/ce/0/public/
```

### URL for a named public portal app

And the URL of a named app for the public portal is:

```
https://<test>/<tenant>/ce/0/public/?appName=<name of the app>
```

**Note** Users go to the default app if the URL is invalid.

### Versions and public portal apps

Users are always taken to the live version of the public portal app.

### URL for testing a non-live public portal app

When developing a public portal app, you can test it using the browser's incognito mode. You do not need to make the app live in order to test it - you can append the version number of the app like this:

```
https://<test>/<tenant>/ce/0/public/?appName=MyApp:1
```

Where `MyApp` is the name of the CE Studio app and `1` is the version number.

In 6.4 and later, you need to select the option **Allow Non-live Versions to be Accessible**. By default, non-live versions are not accessible. To change this, go to the Apps page and click **Edit** for the app.

### Custom domain for the default public portal app

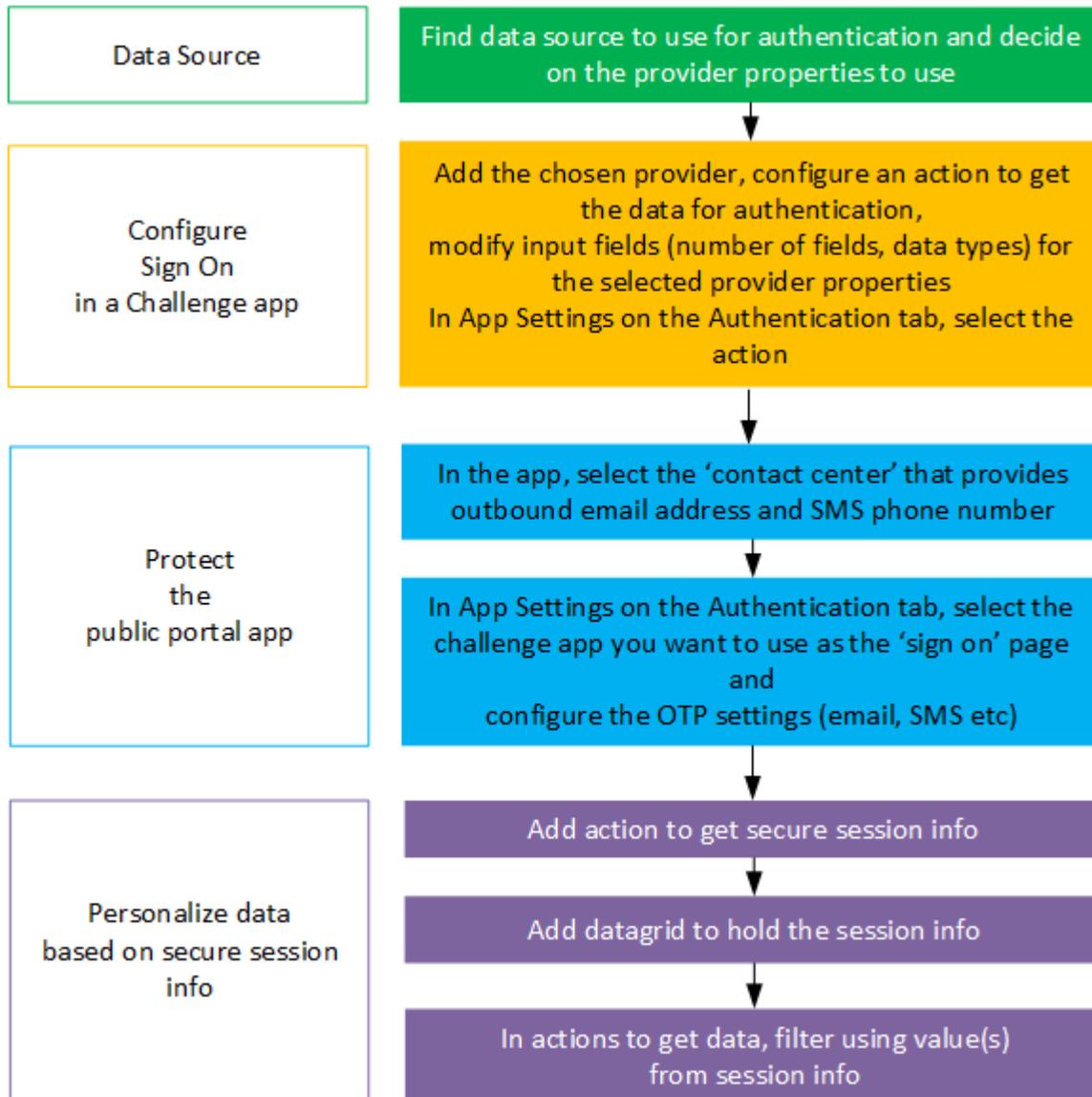
A custom domain enhances branding for your public portal by providing a customized URL, applying only to the default public portal. For IFS Customer Engagement only, you do this by uploading the certificate for the domain in the Admin Portal, thereafter the URL for the domain covered by the certificate can be configured in the Admin Portal.

# 10

## Authentication for public portals

Access to public portal apps is for anonymous users and therefore anyone who has the public portal URL can access the portal. If you need to control access then you can implement your own solution for authenticating users, for example, you can send secured links or one-time passwords. This section describes how to do this in IFS Customer Engagement 6.7 and later, using the standard IFS CE Example Challenge App as the sign on page. You can download these from the Central Template Repository.

The following diagram summarizes the setup steps for protecting a public portal app using the example challenge apps.



*Process of configuring authentication for a public portal app*

When configuring this type of app, you must also:

- Only show the data that belongs to the signed-on user. See [Personalizing data based on secure session info](#).
- Configure sign out. See [Signing out from a public portal](#).

## About the example challenge apps

---

**Note** There are several example challenge apps that are configured for a string input, such as an email address. You can download these from the Central Template Repository.

A challenge app is any unprotected, public portal app that requires users to enter their credentials, such as a username and password, email address, account number. The challenge app functions as a sign-on page. When users go to the public portal, they are first presented with the challenge app and must enter the required details for authentication. The challenge app then redirects the user to the protected app.

The challenge app performs the authentication, and the protected app specifies what form the challenge takes. Whether the user does a one-time passcode (OTP) challenge before they gain access to the protected public portal app or whether they are sent a secure link.

In IFS Customer Engagement 6.7, you can use the same challenge app with multiple protected public portal apps *provided that*:

- The protected apps all share the same provider for the authentication details.
- The data type of the input field(s) for the credentials is the same. The data types must match, for example email address is a string whereas account number might be an integer.

If you require different providers for authentication, then you need to configure a separate challenge app for each provider.

## Configuring the example challenge apps

---

The example challenge apps that you download from the Central Template Repository are partially configured. You need to complete the following steps and then make the app live:

### Step 1 - add a single action to get the authentication details

**Note** For some providers, you may require two actions to get the authentication details. For example, one action to verify that an account number is valid and a second action to get the contact details for that account.

1. Add the provider that will be used to verify the user's credentials.

For example, in IFS Field Service Management the `contact` type has properties for email address and phone numbers as does the IFS Cloud provider, `CrMContactCreation`.

2. Create a new action set for verifying the credentials, using your chosen provider. When creating the action set, select the **Authentication Action** check box.

Selecting the **Authentication Action** check box makes the Authentication Response Properties available in the action.

3. Add an action to the action set. In the Filters setting of the action, add a condition for each of the challenge app input parameters (email address, account number and so on).
4. Select the **Publish Topic** check box. This is needed for the rules and display tasks to work correctly.
5. In **Authentication Response Properties**, add the provider properties that you want to use in the OTP challenge or to send a secure link.
6. Save the action.

### Step 2 - if using a challenge app with Captcha

In the Admin Portal, configure the site that provides the reCaptcha service and then configure the Captcha element to use this site.

### Step 3 - configure the challenge app to use the action

**Note** If you require two actions to get the authentication details. For example, one action to verify that an account number is valid and a second action to get the contact details for that account. Enter the first action in **Setup** and the second action in **Get Authentication Details**.

1. On the Designer page, click **App Settings** .
2. Go to the **Authentication** tab.
3. Add your action in two places:
  - In **Setup**, select your action from the **Initiate Authentication Action Set** list.
  - In **Get Authentication Details**, select your action from the **Select Authentication Details Action Set** list.
4. In **Get Authentication Details**, select the provider properties to use in the OTP challenge or to send the secure link.

When ready to test the challenge app, make it live.

## Protecting a public portal app

A protected app is any public portal app that requires users to confirm their identity through a challenge app before they gain access to the public portal app. When users go to the public portal, they are first presented with a sign in page (the challenge app) and enter the required details. In the background, the challenge app initiates authentication with the provider endpoint that has the

data for authentication. If successful, the challenge app then redirects the user to the protected app.

To make a public portal app into a protected app, you select a challenge app and then configure the type of OTP challenge or secure link.

### Prerequisites

Before you can complete the configuration of a public portal app as a protected app, you need to configure the challenge app and make it live. See [Configuring the example challenge apps](#) for details.

### Creating a public portal app and configuring it for authentication

1. Create a new CE Studio app:

<b>Type</b>	Public Portal
<b>App Group</b>	Either Standard or Template
<b>Contact Center Unit</b>	Select the contact center unit for this app. This filters the response templates, phone numbers and email addresses in the Admin Portal to those belonging to the contact center unit. These will be used by the app to send the OTP challenge.

2. On the Designer page, click **App Settings** .
3. Go to the **Authentication** tab.
4. In **Setup**, select the challenge app that you want to use.
5. In **OTP Configuration**, configure the verification method (email, SMS), the timeout period, the mode of authentication and so on. For details, see [Authentication settings for protected apps](#).

### Testing the protected public portal app

When you are ready to test the protected app:

1. Optional. Make the CE Studio app live.  
If you are not ready to make the app live, then you can specify the name of the app in the preview URL. For details of how to do this, see [Public portal URLs](#).
2. If your app is live, then make the CE Studio app into the default public portal app. To do this, go to the Portal Settings page and select your live portal app from the **Default Public Portal Page** list.
3. Test the protected app using your browser in incognito mode. You cannot test the app properly if you are logged in as a CE user in the browser.

## Authentication settings for protected apps

There are different OTP options available for protected public portal apps:

- Send a one-time password/code to the user's email address or mobile phone
- Allow the user to choose where to receive the password
- Verify that the user owns the email address or mobile phone number by requiring them to type it in

To configure the OTP options for a public portal app:

1. On the Designer page, click **App Settings** .
2. Go to the **Authentication** tab.
3. In the **Setup** section, select the CE Studio app configured as the challenge app as described in [Protecting a public portal app](#).
4. In the **OTP Configuration** section, select the required options:

Field	Description
Verification Methods and Precedence	<p>Select the verification method for the OTP challenge and then drag to change the precedence:</p> <ul style="list-style-type: none"> <li>• <b>SMS</b>: requires the contact center unit to have an SMS-enabled phone number for sending outbound SMS messages. Select the number to use below.</li> <li>• <b>Email</b>: defaults to the outbound email address defined for the contact center unit.</li> </ul> <p><b>Note</b> You need to select the verification methods <i>before</i> dragging to change the order.</p>
Timeout	<p>The OTP challenge expires after the timeout period. Maximum timeout is 60 minutes.</p> <p><b>Note</b> The expiry time is calculated from when the email or text is sent and not from when the user receives it.</p>

Field	Description
Maximum failed attempts allowed	<p>Sets a limit on the number of attempts to enter the OTP code. For example, if you enter 1 then the user can only make one mistake before their session expires and they must start again.</p> <p>The minimum is 1 and the maximum is 5.</p>
SMS Numbers	<div data-bbox="873 569 1469 741" style="background-color: #f0f0f0; padding: 5px;"> <p><b>Note</b> Important: to send SMS messages to the UK requires the use of a UK phone number.</p> </div> <p>Lists the SMS-enabled phone numbers for sending outbound SMS messages. By default, the number displayed to the user is masked using asterisks.</p> <p>You can add multiple SMS numbers but for each number, you need to specify the country or countries that you send SMS messages to:</p> <ol style="list-style-type: none"> <li><b>a.</b> In the <b>SMS Numbers</b> field, select the number to use. There must be one default number present.</li> <li><b>b.</b> In the <b>Country Codes</b> field, enter the country code(s) separated by commas. Country codes should only be used once. You must use the E.164 number format to specify the country codes. This format is a + followed by the international country code for the destination (e.g. +1, +44, +351).</li> </ol>

Field	Description
Phone Number Mask	<p>You can:</p> <ul style="list-style-type: none"> <li>• Enter a different character in the adjacent field.</li> <li>• Select a different way of masking the number by configuring an alternate regex pattern. To view or edit the phone number mask, go to the side menu, and select <b>Global Configuration &gt; Regex</b> dialog.</li> </ul>
Email From Address, Email Mask	<p>Lists the system email address available for sending emails in this contact center unit. Like <b>SMS From Number</b>, you can change the character used in the mask and/or select a different mask.</p>
Mode of Authentication	<p>The OTP challenge sent to the user contains one of the following:</p> <ul style="list-style-type: none"> <li>• One Time Code</li> <li>• Secure Link</li> </ul>
Require User to Confirm OTP Address	<p>This is an additional layer of security. It prompts the user to type in their full email address or mobile number.</p>
SMS Content, Email Content	<p>Select the SMS and/or email response template to use.</p>
Whether Users Can Select the Verification Method	<p>Select the verification method to use:</p> <ul style="list-style-type: none"> <li>• <b>Always:</b> lets the user choose between receiving an SMS or an email even if there is no choice because the user has a mobile phone number but no email address (for example).</li> <li>• <b>Never (Configured Precedence):</b> you send the challenge using your preferred verification method. If that is unavailable, then you use the alternative method.</li> <li>• <b>Only When User Has Multiple Methods:</b> lets the user choose between receiving an SMS or an email only if both are available for the user.</li> </ul>

Field	Description
Code Length	Length of the password that is sent to the user. Minimum 6, maximum 9 characters in length.
Code Type	By default, the code is numbers only but you can choose to use a code that is a mix of numbers and letters.

- Click **Continue** to save the authentication settings.

## Registering new users through OTP and a challenge app

Elevate user registration security and streamline the process with OTP implementation for challenge apps. This topic outlines a process for implementing OTP registration for new users in challenge apps via email or SMS.

### Developing the challenge app for registering new users

- Create a new or edit an existing app in public portal.
- Create two Forms as a top-level component or add it as an element within an existing component. One Form has elements to obtain the user's details (Ex. Email/ SMS etc) and buttons to:
  - Register the new user.
  - Submit the OTP code (The second form has a field for entering the OTP code sent by the system and a button to submit the code).

### Actions for registering new users

There are two main types of actions which must be configured for the new user registration OTP to work correctly.

**Initiate Registration** – The purpose of this action is to take the email address (for example), and then send an OTP to the email address. Ensure the action is configured with the below:

Provider	IFS CE System Provider, Version 9
Type	PublicPortalAuthentication
Method	InitialAuthenticationEx

This action must have the following parameter mappings:

Name of the Parameter	
otpAddress	Map this to a field with a user's email address or mobile number.
otpVerificationMethod	Map this to a static value which must either be email or sms.
responseTemplateName	Map this to a static value which must be the name of the response template created in the admin portal.

**Invoke Registration action set** – The purpose of the actions in this action set is that it takes the OTP entered by the user, matches it with the sent OTP for validation, and upon successful validation, registers the email address almost instantaneously. This action set has two actions under it:

**Note** Mandatory to use chained action so that the system uses the verified address. It checks from validated data.

**Validate Registration** – The purpose of this action is to take the OTP from the user and matching it with what it has sent the user. Once it matches it sends it back to the system. Ensure the action is configured with the below:

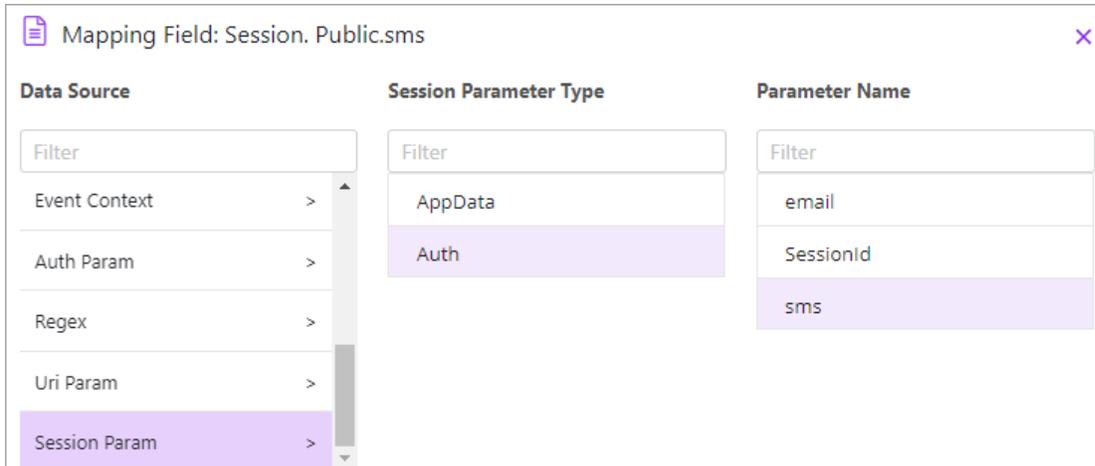
Provider	IFS CE System Provider, Version 9
Type	PublicPortalAuthentication
Method	InvokeAuthentication

**Register User** - This action creates the user details. This action must be a chained action and take the email address or phone number from the first action in the action set.

## Personalizing data based on secure session info

A public portal app must only show data belonging to the signed-on user.

To make sure of this use the auth data available in the session when getting data from the provider.



*Mapping to auth data in the session*

To add an action that makes use of secure session information:

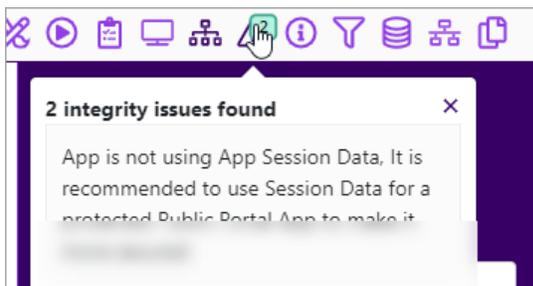
1. Create an action set and select the **Authentication Action** check box.



2. Add the action with a filter similar to this:

Condition	Map to...
Right side	For example, a field reference
Left side	<b>Session Param &gt; Auth &gt; one of the authentication response properties</b> For example, email, phone

If this is not configured then a warning is displayed in the Integrity checker, on the Events page:



## Signing out from a public portal

You can let users sign out securely from a public portal app. This requires a method to invalidate the session and a display task to refresh the window:

1. Insure that the public portal app is using version 7 or later of the IFS CE System provider.
2. Create an action to invalidate the current session:
  - Provider type: `PublicPortalAuthentication`
  - Method: `InvalidateSession`
3. Create a display task to refresh the browser tab:
  - Task: Refresh Window
  - Select the app
4. Add a sign out button to the public portal app.
5. On the Event Configuration page, configure the OnClick event of the button to run both the action and the display task.

## Response templates

---

You can configure custom emails and SMS messages for sending the OTP challenge. You do this by configuring response templates in the Admin Portal. You need to create response templates for each browser locale used by your users. The default locale is US English.

The default message is:

```
Please use CE authentication code %OTPCode% to access [%AppName%]. This code will expire in %OTPCodeExpiry% minutes.
```

The default message for a secure link is:

```
Please use the following link to access [%AppName%] %SecuredLink%
```

When configuring the response templates use the following placeholders for the OTP code and so on:

<code>%OTPCode%</code>	Placeholder for the automatically-generated OTP code.
<code>%SecuredLink%</code>	Placeholder for the automatically-generated link.
<code>%AppName%</code>	The name of the protected CE Studio app that the user is trying to access.
<code>%OTPCodeExpiry%</code>	The configured expiry period for the OTP code or secure link.

To configure a new response template:

1. In the Admin Portal, go to the **Media Management > Response Template** page.
2. Click **New**.
3. Enter the name of the template.
4. Select the type as **SMS Response** or **Email Response**.

5. Select the language. The language used for the OTP challenge depends on the locale of the user's browser.
6. Complete the details. The content must include the above placeholders.
7. Save the template.

To use the response template:

1. In CE Studio Designer, open the protected app.
2. Click **App Settings** and go to the Authentication tab.
3. On the OTP Configuration section, select the response template from the **SMS Content** and/or **Email Content** lists.

# 11

## Surveys

You create App surveys in the Admin Portal. Here you:

- Add questions, select the type of question, add validation and change the order of the questions.
- Set up the distribution settings which control how the survey invites are sent out.
- Save the questions in a survey version and make this version live when you are ready to test the CE Studio app.
- Use Report Designer to view the results of the survey.

You display and style the questions using the survey element, in a public portal or portal app. The Admin Portal help explains how to create the survey definitions.

The survey element shows the questions from the live version of the survey. If no version is live then it shows the questions from the latest version. However, to use the survey, both the survey and the survey app must have a live version.

**Note** It is not possible to generate the survey invite link if the CE Studio app is not live.

There are two types of survey:

Stepped survey	Users click to move between the questions in the survey. Their answer is saved on each button click.
List survey	Users scroll vertically through the questions in the survey. There is only one button at the end of the survey which saves all the answers to the survey.

**Note** Once users click the final button, the survey is completed and they cannot re-use the link to repeat the survey. Links expire on the configured number of days after the survey ends.

The survey element lets you check the look and feel of the survey as you style the public portal app. To test how the questions are validated, you need to view the app on the Preview page. See [Previewing surveys in public portal apps](#) for details.

For further information, see:

- [Configuring a survey app](#)
- [Configuring a survey as part of a public portal app.](#)
- [Configuring a survey as part of a portal app](#)

## Configuring a survey app

This topic describes how to configure the public portal app in which an app survey is displayed. Users open the survey in the public portal app by clicking the survey link that they receive via email or SMS.

You create the survey (App survey type) and its questions in the Admin Portal and then configure an app in CE Studio Designer. Add a survey element to display the questions to the user, and configure the survey element to set the type and style of the survey.

For details of adding a survey as one part of a public portal app or portal app, see:

- [Configuring a survey as part of a public portal app](#)
- [Configuring a survey as part of a portal app.](#)

## Configuring the survey element

Use the survey element to display the questions in the survey.

### Select the survey

Once you have configured a survey and its questions, you can configure the survey:

1. Create a public portal app.
2. Add the IFS CE Survey provider.
3. Add the survey element.

Field	Description
Survey Category	Select the survey category.

Field	Description
Survey	<p>Select the survey. The start text, end text and questions are displayed. You can review them here. To make changes to the text, you need to use the Admin Portal.</p> <p>You can choose to include this text in the survey or hide them.</p> <p><b>Note</b> You use the latest version of the survey if the survey has no live version.</p>
Identifier	<p>Enter the identifier for this element. You use this when configuring the survey actions.</p> <p><b>Note</b> If you see RPC errors when clicking the buttons in the survey then check that the identifier is set and that the actions are mapped correctly to use this identifier.</p>

4. You can either click **Continue** and save the CE Studio app or you can go to the **Styles** tab and start to configure the survey.

#### Set the survey flow, progress bar, button style and content styles

Once you have added the survey element, you can style the survey:

1. Click **Configure** on the Survey element toolbar.
2. Go to the **Styles** tab.
3. Go to the **Survey** tab:

Field	Description
Survey Flow	<p>Select <b>Stepped</b> for a survey style where the user clicks to move through the survey or <b>List</b> for a style where the user scrolls vertically through the questions and then clicks once at the end to submit their responses.</p>
Height	<p>In a stepped survey flow, use this to set the minimum height of each page in the survey.</p> <p><b>Note</b> Enter the height as a number of pixels - do not append <code>px</code> to the value.</p>

Field	Description
Show Start Screen, Show End Screen	Shows any start screen text or end text if configured in the survey. On the Content tab you can add images and background images.
Show Progress Bar	For stepped surveys only, show a bar indicating the user's progress through the survey.  <b>Note</b> Does not apply to surveys with rules.
Button Config	Configure a style for all the buttons in the survey. For list surveys, this is used by the button that appears on the last question or end text screen only.

- Go to the **Content** tab to style the splash text and questions.

### Configure actions and events

See:

- [Actions for surveys](#) and [Events for surveys in public portal apps](#).
- [Configuring a survey as part of a portal app](#)

## About the candidate request ID and Survey ID

Any CE Studio app that uses surveys requires these two identifiers:

- *CandidateRequestId* to identify the person doing the survey
- *SurveyId* to identify the actual survey

If either of these are missing then:

- Survey invites aren't generated
- Responses to the survey aren't saved

Depending on the type of CE Studio app, you can pass the IDs to the app by using secure links or configure the CE Studio app to generate them.

See also [Actions for surveys](#).

## Actions for surveys

For the survey element, you need two actions. One to get the questions and another to save the responses to the survey. The person taking the survey (the candidate) and a survey id is passed in the secure link.

### Link parameters for secure links

Customers access the survey in the public portal app using a secure link. This feature requires link parameters for the values that are passed in the secure link. There are two link parameters:

- *CandidateRequestId* - string
- *SurveyId* - string

Add these on the **Link Parameters** tab:

1. On the Designer page, click **App Settings**.
2. Go to the **Parameters** tab.
3. Go to the **Link Parameters** tab.

### Configure the action to get the questions

You need an action to get the questions for the survey passed in the secure link. This action is for the IFS CE Survey provider:

- Provider type: `SurveyQuestionsResponse`
- method: `GetQuestions`

Click **Manage Properties** button and add the following optional parameters:

- *CandidateRequestId*
- *SurveyId*

Map both parameters to the candidate request id and survey id from the session: go to **Session > AppData**.

Select the **Publish Topic** check box. The action must publish the topic because the survey element needs to subscribe to it.

### Configure the action to save the survey responses

This action saves the person's answers to the survey. This action is for the IFS CE Survey provider:

- Provider type: `SavedSurveyResponse`
- Method: `SaveSurveyResponses`

Click **Manage Properties** and add the following optional parameters:

- `CandidateRequestId` - map this to **Session > AppData**
- `SaveSurveyResponseString` - map this to **Event Context > [SurveyElement] > SaveSurveyResponseString**

**Note** You must map this parameter after adding the survey element. If you remove the survey element and add another one then you must remap this parameter.

## Events for surveys in public portal apps

Configure your public portal app for these events:

Event type	Purpose
OnLoad OnInvoke	When the public portal app containing the survey loads, it runs the <i>action</i> to get the questions. Use the OnLoad OnInvoke event for this.
OnNotification	The survey element subscribes to the above action.

When the user clicks any of the buttons in a stepped survey or the final button in a list survey, the OnSurveySubmit event is triggered:

Event type	Purpose
OnSurveySubmit	Select the survey element and then choose the <b>OnSurveySubmit</b> event type. This must run the action to save the answers.

## Previewing surveys in public portal apps

You cannot preview a fully-configured survey in a public portal app because it requires a secure link to pass the candidate request id and survey id. However, you can clone a version of the CE Studio app and modify the action that gets the questions. You can then preview the cloned version, for example, to check the styling and test the validation in the questions.

To modify the get questions action so that you can preview the CE Studio app:

1. Edit the action that gets the questions.
2. For the **CandidateRequestId** parameter, select **Use Null**.
3. For the **SurveyId** parameter, map it the event context like this:

## Optional Parameters

**request : SurveyQuestionsRequest**

---

CandidateRequestId : string  Use Null

SurveyId : string  Use Null

[title] [title].SurveyElement.SurveyId

## Testing surveys in public portal apps

You need to make the app live - for scheduled surveys, survey links are only generated for live apps.

You must test public portal apps and surveys in incognito mode.

**Note** You cannot test a survey link in a browser browser where you are already signed into the CE Studio Designer or the Admin Portal. You will be unable to submit the survey.

## Sending survey invites from media apps in Agent Desktop

You can configure a media app to send survey invites to customers.

**Note** You need to configure response templates for sending the link to the survey, either email and/or SMS.

1. Add the IFS CE Survey provider to the media app.
2. Add an action to send the survey invite:
  - Provider type: `SurveyCandidateResponse`
  - Method: `AddSurveyCandidate`
3. Click **Manage Properties** to add the following provider properties. Map these as explained below.

Parameter	Description
CandidateId	The recipient of the survey invite (candidate request ID). Map this to a unique identifier, such as: <ul style="list-style-type: none"> <li>• Activation ID if you want to send a survey invite whenever they contact the contact center.</li> <li>• Email address or phone number if you want them to receive the invite once for each survey distribution period.</li> </ul>
ContactCenterUnitId	For media apps, map the contact center unit to <b>Toolbar Query Param &gt; AccountId</b> .
ContactId	Map this to <b>Toolbar Query Param &gt; ActivationId</b> .
ContactType	Map this to <b>Static &gt; MediaApp</b> .
LanguageCode	Map this to a field that will provide the language code, such as en-GB or to a static value.
RequestMode	There are three request modes: <ul style="list-style-type: none"> <li>• <b>PercentageBased</b>: uses the distribution settings configured in the survey.</li> <li>• <b>Immediate</b>: sends the survey invite immediately using the current media channel.</li> <li>• <b>Scheduled</b>: sends the survey invite after the scheduled interval using either the email address or mobile phone number configured above.</li> </ul> For details of the distribution settings, see the Admin Portal help.
SurveyAppIdOrName	Enter the name of the CE Studioapp that you will use to display the questions.
SurveyIdorName	Enter the name of the survey as configured in the Admin Portal. <div style="background-color: #f0f0f0; padding: 5px; margin-top: 10px;"> <p><b>Note</b> <i>Not</i> the name of the survey version.</p> </div>
EmailAddress	Map this to a field that provides the email address. The email address is used to send the survey invite.
EmailTemplateName	The email will use this template. Enter the name of the email response template as configured in the Admin Portal.

Parameter	Description
PhoneNumber	Map this to a field that provides the mobile phone number. This is used to send an SMS message containing the link to the survey invite.
SMSTemplateName	The SMS message will use this template. Enter the name of the SMS response template as configured in the Admin Portal.

- On the Event Configuration page, configure an event to run the action, for example, you can use a button click or an OnLoad event to run the action.
- The next step is to transfer the voice caller to the IVR survey. For details, see [Transferring voice callers to an IVR survey](#).

## Transferring voice callers to an IVR survey

**Note** You can download an example app that demonstrates how to transfer an inbound call to the IVR survey. Available from the store which is part of CE Studio Designer.

To transfer a caller on an inbound call to an IVR survey, the CE Studio media app needs to:

- Dial the IVR survey and if answered...
- Transfer the caller to the survey.

### Requirements

- You need to configure an IVR survey in the Admin Portal. For details, see the Admin Portal help.
- Make sure that there is a live survey version.

You reference the IVR survey by the name of survey and *not* by the name of the survey version. For this reason, you don't need to associate the survey with a phone number or queue progression script.

- You must configure an action to generate the survey request. You need version 7 or later of the IFS CE Agent Desktop provider. For details of the action parameters, see [Sending survey invites from media apps in Agent Desktop](#).

### Configuring the transfer to an IVR survey

You can configure the transfer to the IVR survey like this. This uses a button click:

- An OnClick event triggers a rule set. The rule set runs an action configured for the Agent Desktop provider `dialInfo` method `dialOutEx`.

- This dials the IVR survey and not a specific phone number. The format is described below.
- The CE Studio app verifies that the call is answered. You can do this using a rule set configured for:
    - The `OnToolbarEventReceived` event type
    - The `EVT_OUTBOUNDCALL_STATE_CHANGE` event (`ENQUIRYANSWERED`).
  - When the rule succeeds, it runs an action for the Agent Desktop provider `actions` method `transferCall`. If the rule fails then it runs a different action to retrieve the call: the Agent Desktop provider `actions` method `connectToPrimary`.

**Important** The CE Studio app must generate a survey request (`invite`) before it can transfer the caller to the IVR survey. The action that generates the `invite` runs on the `OnInvoke` status and the transfer rule set runs on the `OnSuccess` status.

### Dialing a survey

Configure an action to dial a survey and pass the candidate request ID and survey ID:

- Use the `dialOutEx` method discussed above.
- Click **Manage Properties** and add these properties:

appData	<p>Configure this dynamic property like this:</p> <ol style="list-style-type: none"> <li>Click the <b>Filter by Provider Type</b> field and select <b>appData</b>.</li> <li>In the <b>Property name</b> field, enter <code>candidateRequestId</code> (the name of a predefined script element).</li> </ol> <div data-bbox="509 1234 1472 1404" style="background-color: #f0f0f0; padding: 10px; margin-top: 10px;"> <p><b>Note</b> You use the <code>appData</code> parameter to access script elements defined in queue progression scripts. A dynamic property name must always match the script element name.</p> </div>
number	<p>Map this to the name of the survey (not the name of the survey version), for example like this:</p> <pre>survey:name of IVR survey</pre>

### Automatically transferring the caller

To automatically transfer the caller to the survey before the agent hangs up, modify the configuration of the hang up button to:

- On the `OnInvoke` status, dial the IVR survey and transfer the caller.

2. On the OnSuccess status, hang up the call.

## Configuring a survey as part of a public portal app

---

This topic describes how to add surveys to public portal apps so that the user can take part in a survey after using the public portal app (for example). They do not receive a survey invite.

You can configure a public portal app to let your users complete a survey. You do not send any survey invites externally - the option to do the survey is part of the public portal app.

### Configuring the survey element in public portal apps

You use the survey element to display the questions in the survey and style the survey.

The survey requires a survey candidate to identify the survey instance. This is used when getting the survey questions and submitting the answers. When you generate the survey candidate depends on the requirements for the public portal app. For example, you can generate it when the public portal app is loaded in the browser.

#### Add the survey element and the IFS CE Survey provider

1. Create a new public portal app or modify an existing one.
2. Add the IFS CE Survey provider.
3. Add the survey element. For details of how to configure the survey element, see [Configuring the survey element](#).

### Configuring the survey candidate in public portal apps

In a public portal app, the survey requires a survey candidate to identify the survey instance. This is used when getting the survey questions and submitting the answers. When you generate the survey candidate depends on the requirements for the public portal app. For example, you can generate it when the public portal app is loaded in the browser.

#### Configure the survey candidate

To configure the survey candidate:

1. Add a form to the public portal app.
2. Add the following fields to the form:

Provider	IFS CE Survey provider
Provider type	SurveyCandidateResponse

Fields	<ul style="list-style-type: none"> <li>• CandidateRequestId</li> <li>• SurveyLink</li> </ul>
--------	--

3. Create a new action to generate the survey candidate:

Provider	IFS CE Survey provider
Provider type	SurveyCandidateResponse
Method	AddSurveyCandidate

4. Click **Manage Properties** and add the following properties. Map these as described below.

Parameter	Mapping
CandidateId	Map to the user's name or a similar unique value.
ContactCenterUnitId	Map to a system variable: <b>System Variable &gt; ContactCentreUnit</b> .
ContactId	Map to a unique identifier for the user, such as <b>Session &gt; Auth &gt; SessionId</b> .
ContactType	Map to the contact type: <b>Static &gt; PublicPortal</b> .
LanguageCode	Map to <b>Session &gt; Auth &gt; User Locale</b> or to a static value, such as en-GB.
RequestMode	Map to <b>Static &gt; Immediate</b> . You cannot use either <b>PercentageBased</b> or <b>Scheduled</b> .  For details of the distribution settings, see the Admin Portal help.
ReturnSurveyLink	Map to <b>Static &gt; True</b> . This means that the survey link is used internally by the app and is therefore not available to send via email or SMS.
SurveyAppIdorName	Map to the name of the current portal app.
SurveyIdorName	The name of the survey as defined in the Admin Portal.
SurveyCandidateAddress.EmailAddress, SurveyCandidateAddress.PhoneNumber	Select the <b>Use Null</b> check box for this type of app.

5. Select the **Publish Topic** check box. The form you added earlier needs to subscribe to this action.

## Actions and events for surveys in public portal apps

For the survey element, you need to create two action sets. One with an action to get the questions and another with an action to save the responses to the survey. The configuration is the same for both public portal apps and portal apps.

See [Actions for surveys in portal apps](#).

See [Events for surveys in portal apps](#).

## Configuring a survey as part of a portal app

---

This topic describes how to add surveys to portal apps so that the user can take part in a survey after using the portal app (for example). They do not receive a survey invite.

You can configure a portal app to let your users complete a survey. For portal apps, you do not send any survey invites - the option to do the survey is part of the portal app.

You use the survey element to display the questions in the survey. You create the survey (App survey type) and its questions in the Admin Portal. The survey is then displayed to the end-user as part of the portal app. Users register and sign onto the portal in the usual way.

The survey requires a survey candidate to identify the survey instance. This is used when getting the survey questions and submitting the answers. When you generate the survey candidate depends on the requirements for the portal app. For example, you can generate it when the portal app is loaded in the browser.

## Configuring the survey element in portal apps

You use the survey element to display the questions in the survey and style the survey.

The survey requires a survey candidate to identify the survey instance. This is used when getting the survey questions and submitting the answers. When you generate the survey candidate depends on the requirements for the portal app. For example, you can generate it when the portal app is loaded in the browser.

### Add the survey element and the IFS CE Survey provider

1. Create a new portal app or modify an existing one.
2. Add the IFS CE Survey provider.
3. Add the survey element. For details of how to configure the survey element, see [Configuring the survey element](#).

## Configuring the survey candidate in portal apps

In a portal app, the survey requires a survey candidate to identify the survey instance. This is used when getting the survey questions and submitting the answers. When you generate the survey candidate depends on the requirements for the portal app. For example, you can generate it when the portal app is loaded in the browser.

**Note** You can only generate the candidate request Id once for any request. For testing, you need to change the candidate details before you can generate a new candidate request Id.

### Configure the survey candidate

To configure the survey candidate:

1. Add a form to the portal app.
2. Add the following fields to the form:

Provider	IFS CE Survey provider
Provider type	SurveyCandidateResponse
Fields	<ul style="list-style-type: none"> <li>• CandidateRequestId</li> <li>• Status</li> </ul>

3. Create a new action to generate the survey candidate:

Provider	IFS CE Survey provider
Provider type	SurveyCandidateResponse
Method	AddSurveyCandidate

4. Click **Manage Properties** and add the following properties. Map these as described below.

Parameter	Mapping
CandidateId	<p>Map to the user's name, such as <b>Session &gt; Auth &gt; UserFullName</b>.</p> <p><b>Note</b> When testing you need to change the value each time as a candidate id is only generated once for each of the survey's distribution periods. For example, you could set it to a system variable such as <b>System Variable &gt; Now</b>.</p>
ContactCenterUnitId	<p>Map to a system variable: <b>System Variable &gt; ContactCentreUnit</b>.</p> <p><b>Note</b> Do not map to any of the <b>ToolbarQueryParam</b> options.</p>
ContactId	<p>Map to a unique identifier for the user, such as <b>Session &gt; Auth &gt; UserId</b>.</p> <p><b>Note</b> The property you select must always be a GUID, such as a user or session ID.</p>
ContactType	Map to the contact type: <b>Static &gt; Portal</b> .
LanguageCode	Map to <b>Session &gt; Auth &gt; User Locale</b> or to a static value, such as <code>en-GB</code> .
RequestMode	<p>Map to <b>Static &gt; Immediate</b>. You cannot use either <b>PercentageBased</b> or <b>Scheduled</b>.</p> <p>For details of the distribution settings, see the Admin Portal help.</p>
SurveyAppIdorName	Map to the name of the current portal app.
SurveyIdOrName	The name of the survey as defined in the Admin Portal.

Parameter	Mapping
SurveyCandidateAddress.EmailAddress, SurveyCandidateAddress.PhoneNumber	<p>Either the email address and/or the phone number is required to generate a survey candidate.</p> <p>For example, for email address, you can map to <b>Session &gt; Auth &gt; IDPUpn</b>.</p> <p>Select the <b>Use Null</b> check box if you do not have both values.</p>

5. Select the **Publish Topic** check box. The form you added earlier needs to subscribe to this action.

## Actions for surveys in portal apps

For the survey element, you need to create two action sets. One with an action to get the questions and another with an action to save the responses to the survey.

### Configure the action to get the questions

1. Create the action to get the questions for the survey. This action is for the IFS CE Survey provider:
  - Provider type: `SurveyQuestionsResponse`
  - method: `GetQuestions`
2. Click **Manage Properties** button and add the following parameter:

*CandidateRequestId*
3. Map the parameter to the `CandidateRequestId` field in the form (see [Configuring the survey candidate in portal apps](#)).
4. Select the **Publish Topic** check box. The action must publish the topic because the survey element needs to subscribe to it.

### Configure the action to save the survey responses

1. Create the action to save the answers to the survey. This action is for the IFS CE Survey provider:
  - Provider type: `SavedSurveyResponse`
  - Method: `SaveSurveyResponses`
2. Click **Manage Properties** and add the following parameter:
  - `CandidateRequestId`
  - `SurveyResponseString`
3. Map these as follows:

CandidateRequestId	Map this to the CandidateRequestId field in the form (see <a href="#">Configuring the survey candidate in portal apps</a> ).
SurveyResponseString	Map this to the button click, that is to <b>Event Context &gt; [SurveyElement] &gt; SaveSurveyResponseString</b>

**Note** You must map these parameters after adding the survey element. If you remove the survey element and add another one then you must remap them.

## Events for surveys in portal apps

On the Event Configuration page, configure your portal app for these events:

### Generate the survey candidate

Event type	Purpose
OnLoad	When the portal app loads, it runs the <i>action</i> to generate the survey candidate. Use the OnLoad OnInvoke event for this.
OnNotification	The form containing the CandidateRequestId field subscribes to the above action.  See <a href="#">Configuring the survey candidate in portal apps</a> .

### Get the survey questions

Event type	Purpose
OnLoad, OnClick	Before users can start the survey, you need to run the action to get the survey questions. For example, if there is a button that users click to take the survey then you can use the OnClick event to run the action.  See <a href="#">Configure the action to get the questions</a> .
OnNotification	The survey element subscribes to the above action.

### Submit the survey answers

When the user clicks any of the buttons in a stepped survey or the final button in a list survey, the OnSurveySubmit event is triggered:

Event type	Purpose
OnSurveySubmit	<p>Select the survey element and then choose the <b>OnSurveySubmit</b> event type. This must run the action to save the results.</p> <p>See <a href="#">Configure the action to save the survey responses</a>.</p>

## Testing surveys in portal apps

Testing a portal app that is configured for surveys is the same as testing any portal app.

When testing the survey, it is useful to show the form that contains the **CandidateRequestId** field. If this field is empty then it was not possible to generate a survey candidate which indicates a problem with the configuration.

For further details of testing portal apps, see [Testing the portal](#).

# 12

## Media apps

This section describes how to configure media apps for handling email, chat and social media messages in Agent Desktop.

### Email

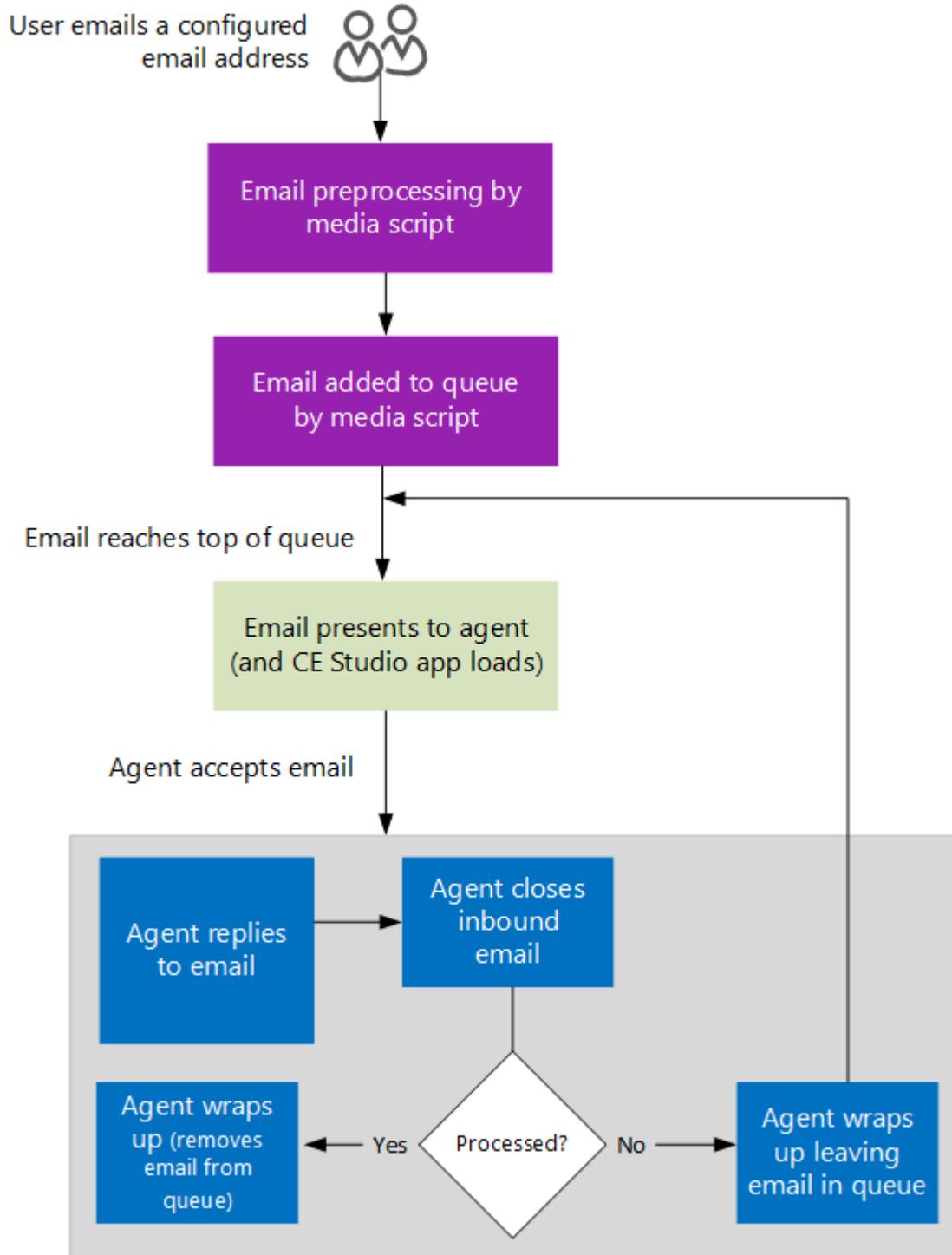
---

You can configure a CE Studio app for handling emails in Agent Desktop or add email handling to any app that needs it. You can configure multiple email apps if different contact center units have different requirements for email. To speed up and simplify configuration, IFS Customer Engagement provides an email template that you can use as a starting point.

**Note** For background information on templates, see [Working with templates](#).

### Configuring CE Studio apps for email

The following diagram summarizes the life cycle of an inbound email.



*Life cycle of inbound email*

When an email is sent to one of the tenant's configured email addresses, it can be processed by a script before it is presented to an agent in Agent Desktop. Emails that need the attention of an agent are then placed on the queue for the next available (and suitably skilled) agent. When the

email reaches the top of the queue and a suitably skilled agent goes into the idle state, the email is presented to that agent and the associated CE Studio app loads.

The agent deals with the email, closes it and wraps up. If they are unable to deal with the email, they may need to transfer it to another agent.

It is useful to consider the following when designing apps for email:

- Emails are delivered to agents in Agent Desktop from the queue. Each email has an ID and is automatically tracked with a conversation ID (by default based on the subject).
- When an email is accepted by an agent, it is handled as an email activation and has an activation ID. Some methods needed for handling email, such as close and wrap up, are provided by the *activation* and the IFS CE Agent Desktop provider.
- How emails are handled in the queue is based on the email media script. You configure scripts for queue progression in the Admin Portal. The script may send automatic replies and perform other processing based on the results of keyword searches and natural language processing. For details of media scripts, see the *IFS Customer Engagement Admin Portal Guide*.
- Emails remain in the queue until an agent closes them. The email stays in the queue if it isn't closed — by default, the email keeps its position in the queue unless its priority is changed.
- Other emails in the thread or conversation are held in a follow up queue.
- To provide additional data for use with emails, you can design entities and import data into entities. For example, you might need to define categories for use when wrapping up a closed email. You do this in the Admin Portal.

### Email conversations

By default, inbound emails are automatically grouped into conversations based on the subject line and are identified by a conversation ID. Only the latest email in the conversation is presented to an agent — any other open emails in the conversation are held in a follow up queue. When the active email is closed and wrapped up by an agent, the next email from the follow up queue is presented to the same or a different agent.

Depending on business requirements, you can design the email app to:

- Show all the emails in the conversation
- Let the agent deal with all the open emails in the conversation at the same time

You can change the way that a conversation is defined. For example, a conversation could be:

- All emails from the same client
- All emails with the same reference

When handling emails, you can:

- Configure the app to save any data that might be needed later as part of the application data for a conversation.

- Add a reference, such as a case number, to email replies. You can then configure the media script to detect and make use of these AppRef values.
- Configure the media script to look at the email history and assign the email to the agent that handled previous emails in the conversation.

## About the IFS CE Email Template

The screenshot displays the IFS CE Email Template interface in Agent Desktop. It features a purple header bar with navigation icons and a search field. The main content area is divided into three sections: 'Incoming Email', 'Email Follow-up', and 'Reply or Forward'. The 'Incoming Email' section includes fields for 'From:', 'To:', and 'Subject:', with the subject being 'RE: Your annual boiler service'. Below the subject is a text area for inserting text. Action buttons for 'Reply', 'Forward', 'Close & Wrap Up', 'Requeue', and 'Transfer' are present. A dropdown menu for 'Transfer to agent with skill set:' is also visible. The 'Email Follow-up' section contains a table with columns for 'Subject', 'To', 'From', and 'Email Type', which is currently empty. There are 'Refresh' and 'Close Follow-up' buttons below the table. The 'Reply or Forward' section is at the bottom.

*The email base template as it appears in Agent Desktop*

**Note** Clicking **Reply** or **Forward** expands the Reply or Forward pane. To discard the reply, collapse the pane. Note that you can copy the email template and configure different behavior if required.

You can inspect the IFS CE Email Template:

1. In CE Studio Designer, go to **Home > Apps**.
2. Locate IFS CE Email Template in the list and click **Manage Versions**.
3. Click **Designer**.
4. You cannot make changes to the live version of the template but you can inspect it. Click **OK** to continue.

The template is configured as follows:

## Components

The top-level component is the template. It contains three components and a utility component.

The Follow-up Emails component show all the emails in the same conversation. These are held in a follow-up queue. The **Flag** column indicates the type of email, such as 1 for inbound emails waiting in the queue and 4 for outbound emails.

The Incoming Email component has a

- A list container for displaying multiple file attachments.
- A label element for the email body. (The label element is used here to improve the loading speed of long emails. We do not recommend using the Rich Text Editor.)
- A button group.
- Hidden property fields for the mandatory ID, the conversation ID and the start date. (The start date will be used as the received date. This is the date the email is put on the queue.)

The Reply component has:

- Send buttons for handling emails with and without attachments.

A hidden utility component for downloading inbound email attachments.

## Providers

The template uses the IFS CE Email provider and the IFS CE Agent Desktop provider. The email provider has different types for inbound and outbound email. Information about email activations in Agent Desktop is obtained through the client provider, which also provides close and wrap up actions.

The template uses the Live version of the providers. This means that the template will update automatically whenever IFS update the provider.

## Actions

The template has an action set for each function performed by the app.

**Note** You may need to modify the configuration of the send action based on how you have configured email sending in your tenant. For example, if there is no EWS Mailbox for the from address.

## Display tasks

The template has display tasks, for example, to update the UI when the app loads and to populate fields in the Reply component from fields in the Incoming Email component. There is a specialized display task for formatting the email in a reply to.

## Rule sets

Has rule sets for invoking the close and wrap up actions.

**Note** Actions that use IFS CE Agent Desktop provider types and methods must be in a rule set. It is not currently possible to invoke actions directly.

## Events

The outer container – the template – has an OnLoad event that runs an action set to:

- Get the email for the current email activation in Agent Desktop
- Run the App Loading display task to disable fields in the inbound email

The Incoming Email component has an OnNotification event and subscribes to a topic published by the Get Active Email action set.

The Close and Wrap Up buttons both have onClick events and run rule sets. For example, the Wrap Up button runs the Wrap Up Email rule set.

## Using the email template

You can embed the IFS CE Email Template in an email app or you can import the template and then modify and extend it to meet your own requirements.

**Note** For background information on templates, see [Working with templates](#).

### Basic workflow for email

In Agent Desktop, the basic workflow is to accept the email activation, close the email and then close (wrap up) the activation. For an app based on the email template, agents will:

1. Accept inbound emails by clicking **Accept Email**  on the Agent Desktop toolbar.
2. Close the email by clicking the **Close** button in the app.
3. Close the activation either by clicking the **Wrap Up** button in the app or  on the toolbar.

For additional information, see *IFS Customer Engagement Agent Desktop User Guide*.

### Creating an app from the email template

1. *Creating a standard app or template.*

You do not need to add any providers for use with the template.

2. Add the template to the app:
  - a. Click  to add a component.

- b. Select the template IFS CE Email Template.
  - c. Select the template options.
  - d. Click **Continue**.
3. Click  to save the app. The app is automatically saved as version 1.  
You have now created a standard email app. You can preview it by clicking  but you will not be able to test the functionality as you require Agent Desktop for this.
  4. On the Manage Versions page, make the app live.
  5. On the **Default Media Apps** page, select your email app as the standard app for inbound emails.

You can now test the email app in Agent Desktop by sending an email to one of the email addresses configured for the tenant.

**Note** You also need a media script that adds the inbound email to the queue. For details of media scripts, see the *IFS Customer Engagement Admin Portal Guide*.

## Overview of the email provider types and methods

The IFS CE Email Template uses the IFS CE Email provider. This provider defines the following data types:

*Summary of the IFS CE Email provider data types*

Data Type	Description
Email	The email record in the tenant database. The email record includes both details of the original email and how it was processed.
OutboundEmail	The outbound email that will be sent by the email activation.
OutboundEmail.Attachments	An email attachment that is attached to an unsent email. The attachment is held in temporary storage until the email is sent successfully.
EmailAttachment	An email attachment that has been saved.
ReplyEmail	The email that will be sent in reply by the email activation.
EmailFlag	Type of email record to get: inbound, outbound, email in the queue, <i>email in the follow up queue</i> or any combination of these.

Data Type	Description
UploadedFile	An email attachment that is opened in a browser. This data type can also be used for other purposes than email, such as by IFS Field Service Management.
BodyType	For outbound email, whether the email body is text or HTML. Provides a lookup for the BodyType property: HTML (0),Text (1).
Importance	For outbound email, the Importance tag. Provides a lookup for the importance property: Low (0), Normal (1), High (2) and None (3).

## CE Email type and methods

The IFS CE Email provider type, `Email`, models the email record in the tenant database. The email record includes both details of the original email and how it was processed. `Email` has several methods for getting email records.

**Note** In an action, the `Email` methods always run before any filters defined on the same action. For example, you might get all the emails from a specific sender and then apply a filter to the records returned by the method.

### EmailFlag flag

The `EmailFlag` flag specifies the type of email record to be processed by the `Email` methods. Typically you map these as static values:

Enum	Description
1	Inbound emails waiting in the queue
2	Emails received
4	Outbound emails sent from Agent Desktop by agents or automated replies sent by the email script.
8	Inbound, open emails in the same thread or conversation waiting in the follow up queue (see <a href="#">Configuring CE Studio apps for email</a> ).

For example, enter 15 for all email types (1 + 2 + 4 + 8). You can also set any combination of the above.

## Email methods

### GetEmails

Gets all the emails of the type specified by *flag* where *flag* is an `EmailFlag` enum.

### GetEmailsByActivationID

Gets the email where the ID of the activation matches the given *activationid*. Map:

- *activationid* to the `Activationid` in the activation context (`ToolbarQueryParams`).
- *flag* to an `EmailFlag` enum.

### GetEmailsByConversationID

Gets all the emails of the type specified by *flag* in the same conversation. Map:

- *conversationid* to a field that provides the *conversationid* of an email.
- *flag* to an `EmailFlag` enum.

### GetEmailsByDates

Gets all the emails of the type specified by *flag* for a single date or a range of dates. Map:

- *startdatefrom*, *startdateto* to a single start date or range of start dates to get emails based on when the email is put on the queue (this therefore includes any emails currently in the queue). The start date is set when the script adds the email to the queue.
- *enddatefrom*, *enddateto* to a single end date or range of end dates to get emails based on when they were closed. The end date is set when the agent wraps up the email or the script closes the email.
- *flag* to an `EmailFlag` enum.

For any unused parameters, select **Use Null**.

### GetEmailsByDirection

Gets either inbound or outbound emails where:

- *inbound* should be *true* for inbound emails and *false* for outbound emails. Outbound emails will include both emails sent by the agent and auto replies sent by the email script. This parameter overrides *flag*.
- *flag* is an `EmailFlag` enum.

### GetEmailsById

Gets all the emails of the type specified by *flag* with an ID that matches the given email ID.

Map:

- *id* to a field value that provides the ID of another email selected in the app.
- *flag* to an `EmailFlag` enum.

### GetEmailsByMailbox

Gets all the emails of the type specified by *flag* that are in the given mailbox or in any of the mailboxes that match the Contains operator. Map:

- *contains* is a Boolean. The value in *mailbox* is processed as a Contains operator when *true* and as a literal value when *false*.
- *mailbox* is the name of the mailbox or the exact string to use as the Contains operator. For example, entering *ifs* finds any mailbox containing 'ifs' in its name.
- *flag* to an `EmailFlag` enum.

### GetEmailsByPriority

Gets all the emails of the type specified by *flag* that match the given priority. The priority derives from either the email address configuration or as modified by the email script. Map:

- *priority* to a number in the range 0 (low) through 9 (high).
- *flag* to an `EmailFlag` enum.

### GetEmailsBySender

Gets all the emails of the type specified by *flag* for the given sender email address or the addresses that match the given Contains operator. Map:

- *contains* is a Boolean. The value in *sender* is processed as a Contains operator when *true* and as a literal value when *false*.
- *sender* is the email address of the sender or the exact string to use as the Contains operator. For example, entering *ifs* finds any email address containing 'ifs'.
- *flag* to an `EmailFlag` enum.

### GetEmailsBySkillsetId

Gets all the emails of the type specified by *flag* that match the given skill set ID. You can get the skill set ID from the activation context (ToolbarQuery Param). Map:

- *skillsetid*, for example, to a field value that provides the skill set ID of another email selected in the app.
- *flag* to an `EmailFlag` enum.

### GetEmailsBySkillsetName

Gets all the emails of the type specified by *flag* that match the given skill set name or names that match the given Contains operator. Skill sets are set in the email address configuration or modified by the media script for the queue. Map:

- *contains* is a Boolean. The value in the *skillsetname* is processed as a Contains operator when *true* and as a literal value when *false*.
- *skillsetname* is a string giving the skill set name or the exact string to use as the Contains operator. For example, entering *ifs* finds any skill set name containing 'ifs'.
- *flag* to an `EmailFlag` enum.

### GetEmailsBySubject

Gets all the emails of the type specified by the *flag*, where the subject line matches the given subject or Contains operator. Map:

- *contains* is a Boolean. The value in the *subject* is processed as a Contains operator when *true* and as a literal value when *false*.
- *subject* is a string giving the subject or the exact string to use as the Contains operator. For example, entering *ifs* finds any email with 'ifs' in the subject line.
- *flag* to an `EmailFlag` enum.

## OutboundEmail type and methods

The IFS CE Email provider type, `OutboundEmail`, is an unsent, outbound email. It has:

- A single `sendemail` method for sending the email.
- An additional provider type `Attachments` for any files attached to the email.

### sendemail

Sends an email for the current activation. Specify what you want to include in the email, including any attachments and how it should be processed. The following properties are required and specific to CE:

- *ClientId* — you must map this to the account code of the contact center unit. In the `ToolbarQueryParams`, this is *AccountId*.
- *SenderId* — you must map this to the activation ID in order to associate the outbound email with the activation. In the `ToolbarQueryParams`, this is the *ActivationId*.

The following optional property is also specific to CE:

- *AppRef*: an additional reference that can be set by a CE Studio app and used by the email script to process the email in the queue if the recipient replies to the email. For further details, see the *IFS Customer Engagement Admin Portal Guide*.

To add parameters for an attachment,

1. click **Manage Properties**.
2. Click in the **Filter by Provider Type** field adjacent to the **OutboundEmail**. field and select **Attachments**. You can then add parameters to set the filename, for example.

**Note** For outbound email, the `email` property *SenderType* should be set to 1 (the default value). *SenderType* is an enum with these values:

- AutoReply: 0
- Agent: 1
- WorkObject: 2
- TextChat: 3
- IVR: 4
- Report: 5

## Email methods from the activation

The activation context of the current email provides the methods for closing the email and wrapping up. You can also get information about an email.

### The CloseEmail method

An agent can either close the email or they can transfer the email to another agent. Both these cases are handled by the `emailCloseData` method `closeEmail`. It has four required parameters:

Required parameter	Description
<i>appData</i>	A <code>SystemString</code> that stores the state of a group of emails (identified by a conversation ID).  For example, an agent may need to store notes relating to an email in the conversation that they have closed and refer to these notes later when handling other emails in the same conversation.
<i>category</i>	A <code>SystemString</code> for a custom property that you may want the agent to set when they wrap up the email.  For example, you may need to categorize emails for reporting purposes.
<i>processed</i>	Set this to: <ul style="list-style-type: none"> <li>• <i>true</i> to close the email and remove it from the queue.</li> <li>• <i>false</i> to leave the email on the queue, at the same priority. The email will then be presented to another agent.</li> </ul>
<i>newSkillset</i>	A <code>SystemString</code> that is set to the name of the skill set to which to transfer the email.  <b>Note</b> If <i>processed</i> is <i>false</i> then you may also need to update the skill set. If you do not then the email is likely to be presented to the same group of agents again.

To use the `closeEmail` method in an action:

1. Add an action to an action set.
2. As provider, select IFS CE Agent Desktop provider.
3. As the provider type, select `emailCloseData`.

4. As the method, select `closeEmail` and set the required parameters.

You do not need to publish this action.

### The `wrapUp` method

The `actions` method `wrapUp` does one of the following based on the outcome of the `closeEmail` method:

- Removes a *processed* email from the queue and puts the agent back into the idle state ready for the next activation.
- Puts the agent back into the Idle state but leaves the email in the queue so it can be picked up by another agent.

`wrapUp` has no required or optional parameters.

To use the `wrapUp` method in an action:

1. Add an action to an action set.
2. As provider, select IFS CE Agent Desktop provider.
3. As the provider type, select `actions`.
4. As the method, select `wrapUp`.

You do not need to publish this action.

### The `getEmailInfoCallback` method

The `emailInfo` method `getEmailInfoCallback` gets information about the current email or email entity.

## Chat

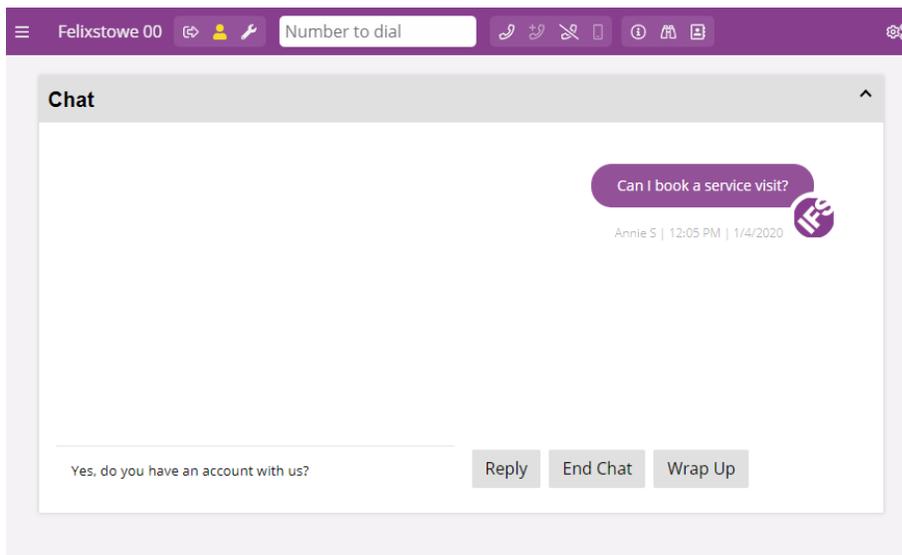
---

IFS Customer Engagement provides a chat template that you can use in any of your apps. It is unlikely that you will need to modify this template.

You can access data from the chat session if the media script was configured to saved the data in script elements.

**Note** For background information on templates, see [Working with templates](#).

## About the IFS CE Chat Template



*The base template as it appears in Agent Desktop*

You can inspect the IFS CE Chat Template:

1. In CE Studio Designer, go to **Home > Apps**.
2. Locate IFS CE Chat Template in the list and click **Manage Versions**.
3. Click **Designer**.
4. You cannot make changes to the live version of the template but you can inspect it. Click **OK** to continue.

The template is configured as follows:

### Components

The top level component is the template. It has one component, a form, that contains three elements:

- A Chat element to show the initial question and then the follow up messages from the caller. The Chat element is configured with the mandatory `InitialQuestion` property from the Chat client. See [Configuring the Chat component](#).
- A ChatInput UI element for entering the message to send.
- A button group. The **End Chat** and **Wrap Up** buttons duplicate tools available on the Agent Desktop toolbar.

### Providers

The template uses the IFS CE Agent Desktop provider and the IFS CE Chat provider. The client provider is required to obtain information from the chat session in Agent Desktop (the activation), and perform reply, close and wrap up actions.

The template uses the Live version of the providers associated with the template. This means that the template will update automatically whenever IFS updates either of these providers.

### Actions

The template has an action set for each function performed by the app. Actions for components and elements in the GUI need to publish topics. For example:

- IncomingMessage action uses the client provider `incomingMessage` data type and the `addChatMessageWithoutDate` method. The method parameters are mapped to properties on `chatEvent`.
- OutgoingMessage action set contains two methods. The methods are run in the order in which they are listed in the action set.
- OnChatChanged action sends an indicator that the agent is typing.

### Display tasks

The template has a single Message Sent display task to update the UI whenever a chat message is sent.

### Rule sets

Has a rule set for invoking each of the action sets.

**Note** Actions that use IFS CE Agent Desktop provider types and methods must be in a rule set. It is not currently possible to invoke actions directly.

### Events

The outer container – the template – has an OnLoad event that runs an action set that:

- Gets the chat message
- Runs the Message Sent display task

The inner Chat component has an OnNotification event and is configured with subscriptions. These notify the app of the incoming message, the end chat session event and the call ID of the current chat session.

Each button in the button group has an onClick event which when invoked runs the appropriate rule set.

## Using the chat template

You can embed the IFS CE Chat Template in a chat app or you can import the template and then modify and extend it to meet your own requirements.

**Note** For background information on templates, see [Working with templates](#).

## Basic workflow for chat

In Agent Desktop, the basic workflow is to accept the chat activation, end the chat session and then close (wrap up) the activation. For an app based on the chat template, agents will:

1. Accept a new chat session by clicking **Accept Chat**  on the Agent Desktop toolbar.
2. End the chat session by clicking the **Close** button in the app or  on the toolbar.
3. Close the activation either by clicking the **Wrap Up** button in the app or  on the toolbar.

For additional information, see *IFS Customer Engagement Agent Desktop User Guide*.

## Creating an app from the chat template

1. *Creating a standard app or template.*

You do not need to add any providers for use with the template.

2. Add the template to the app:

- a. Click  to add a component.
- b. Select the template IFS CE Chat Template.
- c. Click **Continue**.

3. Click  to save the app. The app is automatically saved as version 1.

You have now created a standard chat app. You can preview it by clicking  but you will not be able to test the functionality as you require Agent Desktop for this.

4. On the Manage Versions page, make the app live.
5. On the **Default Media Apps** page, set your app as the default app for chat.

You can now test the app in Agent Desktop by starting a chat session using, for example, the demo Chat client. For details of the chat client, see the *IFS Customer Engagement Admin Portal Guide*.

**Note** You also need a media script that adds chat to the queue. For details of media scripts, see the *IFS Customer Engagement Admin Portal Guide*.

## Configuring the Chat component

**Note** The chat client defines a JSON-formatted, dictionary-like object with a mandatory key-value pair, where the key is `InitialQuestion` and the value is a string entered by the caller. You need to add this property to the Chat element. For details of how to configure chat, see the *IFS Customer Engagement Admin Portal Guide*.

This section describes how to add a component with the mandatory `InitialQuestion` property, and the ChatInput box:

1. Click  to add a form.
2. Click  and select **UI Elements**.
3. Add a ChatInput box to the form.
4. Click  on the component toolbar.
5. Add the Chat element:

**Configure Chat** ✕

**Add Property** 

<p>Chat End Direction</p> <input style="width: 90%; border: 1px solid #ccc;" type="text" value="chatEvent   message   direction"/>	<p>Message</p> <input style="width: 90%; border: 1px solid #ccc;" type="text" value="messageData   message"/>
<p>Sender</p> <input style="width: 90%; border: 1px solid #ccc;" type="text" value="messageData   senderName"/>	<p>Timestamp</p> <input style="width: 90%; border: 1px solid #ccc;" type="text" value="messageData   sentTime"/>
<p>Type</p> <input style="width: 90%; border: 1px solid #ccc;" type="text" value="messageData   etype"/>	

Cancel
Done

6. Click **Add Property** and add the property called `InitialQuestion`.
7. Click **Done**.

## Using application data with chat

See also [Accessing application data](#)

To access data saved in script elements in a CE Studio app:

1. Add a form to store the data.
2. Add a field for each script element:
  - a. In the field selector, select the IFS CE Chat provider.
  - b. In **Filter Types** box, enter `AppData.Elements`.
  - c. Enter the name of the script element - the lettercase must match.
3. Add an action to get the data. Use the chat provider method `AppData.GetTabularResponse` and publish the topic.
4. Attach the action set to the `OnLoad` event of the app.

## Social Media

---

You can configure a CE Studio app for handling social media messages in Agent Desktop. You can configure multiple social media apps if different contact center units have different requirements. To speed up and simplify configuration, IFS Customer Engagement provides a social media template that you can use as a starting point.

Before you can configure a CE Studio app to send WhatsApp messages, you need to complete these steps in the Admin Portal:

- Enable WhatsApp on the tenant
- If contacting the caller outside of the 24-hour message window, or initiating the conversation, then you need to create and get approval for the WhatsApp Content Templates you plan to use.

**Note** The standard template is not configured to send messages using WhatsApp Content Templates but only to reply to WhatsApp messages sent by the caller.

### Overview of CE Studio apps for social media

You can display social media conversations and send replies in any CE Studio app that's configured for social media.

The best way to display a conversation in a readable format is to use the Social Media Element. It can show any type of social media conversation, such as WhatsApp and SMS. When the element is fully configured, it lets agents view previous messages, see new ones, reply to messages and close them.

You can add the Social Media Element to any media app, for example, to the:

- Default Social Media app – this app loads when a social media activation presents to an agent. Use this when replying to queued messages and closing messages.
- Home Page app – you could use this to view conversations. Queued messages could then be answered and closed or, if you want to avoid multiple users answering the same message, opened in the default social media app and then answered and closed.
- Forced Wrapup app – this app loads when the agent goes into Forced Wrap mode in Agent Desktop. The options for answering and closing messages are similar to Home Page apps.

These CE Studio apps require version 6 or later of IFS CE Social Media provider.

#### Example IFS CE Social Media template

The example Social Media template demonstrates how to configure social media actions and events. The template is intended for use as the default Social Media app with social

media activations. It is not configured for other uses, for example, it does not let you preview conversations, and this message is displayed:

WhatsApp Preview Unavailable

You can complete the configuration if required by first cloning the template. For details, see [Configuring the Social Media Element](#).

## Social Media provider types

When building CE Studio apps for social media you need to use these providers:

- IFS CE Social Media Provider
- IFS CE Agent Desktop provider

The Social Media provider defines the types, properties and methods for handling conversations and messages.

SocialMediaMessage	Messages belong to a conversation. New messages have queued state and closed messages have a processed state.
SocialMediaFollowup	Followup messages sent by the sender before the original message is answered.
SocialMediaConversation	<p>Series of social media messages from the same caller or messages grouped by a custom requirement. To access a message, you always need the conversation Id.</p> <p><b>Note</b> You establish whether it's a new or existing conversation through the media script using the script option Is New Conversation.</p> <p>By default, the system determines whether a WhatsApp or SMS message is part of an existing conversation based on the sender's phone number. It relies on the individual sending from the same mobile device. However, you can determine this based on other criteria such as a reference in the message. For further information, see the Admin Portal help.</p>

SocialMediaActivation	Wrapper round the social media message that allows a queued message to be processed in Agent Desktop.
SocialMediaActivationHistory	Provider type for processed messages (closed social media activations).
SocialMediaDefinition	How social media messages are processed, such as status, coverage plan. Used when configuring the Social Media Element. See <a href="#">Methods for social media conversations</a> .
SocialMediaIdentity	Identity of senders and receivers in a social media conversation. Used when configuring the Social Media Element. See <a href="#">Methods for social media conversations</a> .

The IFS CE Agent Desktop provider manages opening, closing and wrapping up activations, and integration with the Agent Desktop toolbar.

## Configuring the Social Media Element

The Social Media Element displays a conversation in a readable format. When fully configured, it lets agents answer and close messages, view previous messages and see new ones as they arrive.

### Configuring the data source for the Social Media Element

1. Create a new media app.
2. Add the IFS CE Social Media provider. Select version 6 or later.
3. Add the Social Media element.
4. In the Social Media Element, click the **Configure Social Media** button to configure the data source:
  - a. Select the IFS CE Social Media provider.
  - b. Select and add all the `SocialMediaMessage` properties. Some additional properties are needed, and these are described below.

**Note** The Social Media Element will display a Preview not available message if following From, To, and Message details are not fully configured.

## From Details

Field	Mapping
From	FromId
Name	Name To find this property, click <b>Configure Social Media</b> and filter on <code>SocialMediaMessage.FromIdentity</code>
Profile Image	For media channels that support an image, such as X, get the ImageUrl from <code>SocialMediaMessage.FromIdentity</code> .
Facebook Handle, Twitter Handle	Facebook, X only This is a dynamic property. The example IFS CE Social Media template defines this as <code>screen_name</code> . To define this property, filter on <code>SocialMediaMessage.AppData</code> .

## To Details

Field	Mapping
To	Told Note: Guid identifying the sender of the message.
Name	Name To find this property, click <b>Configure Social Media</b> and filter on <code>socialmediamessage.ToIdentity</code> .

## Message Details

Field	Mapping
Message Type	ClassDescription
Message ID	Id (SocialMediaMessage)

Field	Mapping
Followup Request ID	Id (SocialMediaFollowup) To find this, click <b>Configure Social Media</b> , filter on <code>SocialMediaMessage.Followup</code> and select the Id.
WhatsApp ID, SMS ID etc	PlatformMessageId
Parent ID	ParentId
Original Message ID	TopParentId
Message state	State
Delivery Status	DeliveryStatus
From	FromId
Text	Text
Created at	CreationTime
Replies	Facebook, Facebook DM, Twitter, Twitter DM only This is a dynamic property. The example IFS CE Social Media template defines this as <code>reply_count</code> . To define this property, filter on <code>SocialMediaMessage.AppData</code> .
Retweets	See Replies
Favorites	See Replies
Is Outbound	IsOutbound NOTE: This is true for outbound messages and false for inbound ones.
Definition Id	Social media definition

## Rules for the Social Media Element

Actions configured for the Social Media provider can be invoked directly by events on the Social Media Element but actions that use methods from the IFS CE Agent Desktop provider must

be invoked inside a rule set. This includes the actions to close the message and wrap up the activation.

You can also configure rules when you require conditional logic, for example, when an action applies to specific social media platforms only. Use the following values to refer to specific platforms:

- Facebook = 0
- Facebook Direct Message = 1
- X = 2
- X Direct Message = 3
- WhatsApp = 4
- SMS = 5

### Actions for the Social Media Element

You need to configure actions to get the messages to display in the Social Media Element and for all its UI controls. The IFS CE Social Media template provides simple examples of these actions.

The event context provides many of the required values. This is the event that occurs when users click the Social Media Element. Useful values supplied by the event context are:

- AppData
- FirstMessageTime
- LastMessageTime
- MessageId
- Reaction
- ReplyText

### Getting social media messages

Use the `SocialMediaMessage` method `GetTabularResponse`.

You need to get the social media messages for a specific conversation Id. Where you get the conversation Id from depends on the type of CE Studio app you are creating. See [Methods for social media conversations](#) for details.

For all actions that get messages, note that:

- You need to sort the action by `CreationTime` in descending order.
- In the action, the page size sets the number of messages displayed in the Social Media Element.

**Note** You can only use the Toolbar Query Param data source **SocialMediaConversationId** when working with activations.

### View previous messages, View more messages

Actions are needed when users click **View more messages** and **View previous messages** in the Social Media Element. There is a specific event type for each of these controls.

For example, clicking **View more messages** triggers the OnViewMoreNext event. The action for this needs to get all the messages for the specified conversation with a creation date later than the last message time.

- Social Media Provider
- Provider type: SocialMediaMessage
- Method GetTabularResponse - use these conditions:

Last message time: data source is the event context

Conversation Id: Depending on the use case, this is provided by one of the following:

- SocialMediaConversationId from the Toolbar Query Param data source when Social Media Element is used inside an activation.
- ConversationId configured on the Social Media Element when Social Media Element is used outside an activation

You need to sort the action by CreationTime in descending order.

The Social Media Element must subscribe to this action.

### Send Reply

Use the Social Media provider. The methods depend on the platform, for example:

Facebook	FbCommentResponse	FbComment
SMS	SmsSendReplyResponse	SmsSendReply
X	TwitterReplyResponse	TwitterReply
WhatsApp	WhatsAppSendReplyResponse	WhatsAppSendReply

Map all the parameters to values returned by the event context (user clicking the Social Media Element).

**Note** You can only send a reply if there is text in the Message box of the Social Media Element. If you want to make sure that no one else replies to a queued message, then open it in an activation. See [Opening and closing social media activations](#).

### Close Reply

Use the Agent Desktop provider and the `followUpSocialMedia` method `closeFollowupSocialMedia`. Map the `MessageId` to the id returned by the event context.

**Note** If you want to make sure that no one else replies to a queued message, then open it in an activation. See [Opening and closing social media activations](#).

## Methods for social media conversations

Use the following provider types and methods when creating actions to get the conversation to display in the Social Media Element.

### SocialMediaIdentity GetTabularResponse

Returns the GUID of the social media identity. Social media identity is the combination of the caller (such as mobile phone number) and the social media definition.

Add and map these parameters in the action:

Name	For example, map <code>Name</code> to the caller's mobile phone number.
DefinitionId	Map this to the social media definition ( <code>Id</code> ) in the configuration of the Social Media Element.

### SocialMediaMessage GetTabularResponse

Returns all social media conversations. To get the full conversation for a single caller (client), you need to use a filter that gets the:

- Social media definition `Id` and the client `Id` of the sender.
- Social media definition `Id` and the client `Id` of the recipient.

You can configure the filter like this:

The screenshot displays the 'Filters' configuration window. At the top, there are radio buttons for 'And' (selected) and 'Or'. To the right, there are options for 'Dynamic Filter', 'Delete', and '+ Add'. Below this, a filter is defined for 'SocialMediaMessage' with the field 'DefinitionId' set to 'Equals'. This filter is linked to a sub-section with 'Or' logic. This sub-section contains two filters: 'SocialMediaMessage' with 'FromId' set to 'Equals', and 'SocialMediaMessage' with 'Told' set to 'Equals'. Each filter includes a 'Load By Details' button and a 'Delete' icon.

## Message states

The `state` property of the inbound message takes these values:

- Queued (1) – new message, waiting for an agent to reply and close it.
- Followup (2) – caller's reply received in the CE Studio app before the previous message is closed.
- Processed (3) – the inbound message has been closed.

This also covers outbound messages:

- Processing (0) – Outbound reply sent either from the CE Studio app or from the media script.

## Opening and closing social media activations

If you want to make sure that the user has an exclusive lock on a message, for example to reply to it or close it, then you need to open the message in an activation before replying or closing. This prevents the system from presenting the queued message to another agent.

## Opening activations

This is handled automatically if the CE Studio app is used as the default social media app. If it is used for a different purpose, such as a Home Page app, then you need two additional actions. For example:

### 1. Get the social media activation id

- Social Media Provider
- Provider type: `SocialMediaActivation`
- Method `GetTabularResponse`

Add conditions to filter by the current conversation Id. To get the activation Id, try a filter such as `SocialMediaActivation.Id is not null`.

### 2. Open the social media activation

- Agent Desktop provider
- Provider type: `openSocialMedia`
- Method `openSocialMedia`

This takes a single parameter `idRequest`. Map this to the `Id` property of the `SocialMediaActivation` obtained in the previous action.

You need to run these actions in a rule set. For example, `OnInvoke` run an action or rule set to get the activation Id. `OnSuccess`, run a rule set to open the activation.

## Closing activations

Closing a social media activation, changes the activation status so it will not be presented to any other agents in Agent Desktop. You also need a wrapup action to dismiss the CE Studio app.

To close a social media activation:

- Agent Desktop provider
- Provider type: `socialMediaCloseData`
- Method: `closeSocialMedia`

To close an activation and set the status to `Processed`, set the `processed` property to `true`.

To put the message back on the queue instead of closing the activation, set `processed` to `false`. Other parameters are optional.

## Sending messages using WhatsApp Content Templates

To set up an action to send a message using a WhatsApp Content Template:

1. Add the IFS CE Social Media provider, version 5, to the CE Studio media app.

**Note** You need to use version 5 or later for WhatsApp Content Templates.

2. Create an action:
  - Provider type: `WhatsAppSendTemplatedContentResponse`
  - Provider method: `WhatsAppSendTemplatedContent`
3. Click **Manage Properties** and select all the properties. If the WhatsApp Content Template has parameters then add them as well:
  - a. Click the field next to **WhatsAppSendTemplatedContentRequest** and select **Parameters**.
  - b. Enter the name of each parameters and click **Add**.
4. Configure the properties as follows:

ApprovedTemplateNameOrId	The name of the WhatsApp Content Template as created in the Admin Portal, and approved by Meta.
DefaultLanguage	The ISO language code of the WhatsApp Content Template. If the template is available in multiple languages then this is the language you want to use if you don't support the caller's language. You need to use the languages supported by Twilio as listed here: <a href="https://www.twilio.com/docs/verify/supported-languages">https://www.twilio.com/docs/verify/supported-languages</a>
DefinitionNameOrId	The name of the social media definition created for WhatsApp in the Admin Portal.
FromWhatsAppNumber	The WhatsApp-enabled phone number as configured in the Admin Portal for sending WhatsApp messages.
Language	The Twilio code for the caller's language. If the WhatsApp Content Template has a version in this language then this version is used. If it doesn't then it will send the template using the default language configured above.
1, 2, 3 etc	Optional. These are the parameters configured in the WhatsApp Content Template.

ToWhatsAppNumber	The caller's WhatsApp phone number.
------------------	-------------------------------------

5. Configure an event to invoke the action.
6. Test the configuration on the CE Studio Preview page.

## Integrating media apps with Agent Desktop

**Note** For details of how agents use Agent Desktop, see the *IFS Customer Engagement Agent Desktop User Guide*.

### Linking CE Studio apps to Agent Desktop media types

To associate a CE Studio app with a media (activation) type in Agent Desktop you can set a default app for each media type. The default apps will be used by all the tenant's contact centers and contact center units.

#### Home page apps and apps for offline working

Any component that has an activation ID as a property can only be loaded when an activation is presented to the agent. This means that you cannot use any of the system templates as the home page app. For example, the IFS CE Email Template will look for an email activation when it loads and will display an error when it can't find one. You can only configure this template as the app for the email media type.

**Note** This also means that an app can only contain one component that has an activation ID as a property. For example, an app that embeds both the IFS CE Email Template and the IFS CE Inbound Voice Template cannot load without erroring as one of the those components will not have the activation ID set.

**Note** If you set a default home page app and it doesn't display in Agent Desktop then check whether a different app is configured in the Admin Portal on the **Media Management > App Home Page**.

#### Linking apps to the media types

To link a CE Studio app to each media type:

1. In CE Studio Designer, go to **Home > Default Media Apps**.
2. Set a default for each media type. The media types are described below.

Media type	Description
Default Home page	This app is shown when agents are in the idle state.
Inbound call	Use this media type for an app that loads when an inbound call is presented to the agent. The app will be used by all the tenant's inbound numbers.
Forced wrap-up	<p>An app that loads when agents:</p> <ul style="list-style-type: none"> <li>• Make an outbound call</li> <li>• Dial a number in the Address Book</li> <li>• Click <b>Forced Wrap Up</b> and select the <b>Normal</b> type.</li> </ul>
Training	An app that loads when agents click <b>Forced Wrap Up</b> and select the <b>Training</b> type.
Demo	An app that loads when agents click <b>Forced Wrap Up</b> and select the <b>Demo</b> type.
Offline entry	An app that loads when agents click <b>Forced Wrap Up</b> and select the <b>Offline entry</b> type.
Workflow	<p>An app that loads when a work object is presented to the agent. Design the app to cover all the different types of workflow handled by the contact center unit.</p> <div data-bbox="708 1146 1471 1360" style="background-color: #f0f0f0; padding: 10px; margin: 10px 0;"> <p><b>Note</b> If there is a significance difference between workflows then you need to design apps for each workflow and assign them to different contact center units (see above for details).</p> </div> <div data-bbox="708 1388 1471 1524" style="background-color: #f0f0f0; padding: 10px; margin: 10px 0;"> <p><b>Note</b> Outbound campaign calls that use Preview dialing will present as workflow activations.</p> </div>
Chat	An app that loads when a chat message is presented to the agent. The app will be used by all the tenant's chat definitions.
Email	An app that loads when an email is presented to the agent. The app will be used by all the email addresses configured for the tenant.

Media type	Description
Outbound campaign call	<p>An app that loads when agents are working on outbound campaign calls which use either Predictive or Progressive dialing.</p> <p><b>Note</b> For details, see the <i>IFS Customer Engagement Admin Portal Guide</i>.</p>
Social Media	<p>You can either use a single app for all social media messages or you can configure a separate CE Studio app for each social media type:</p> <ul style="list-style-type: none"> <li>• Select <b>Social Media - Default</b> if you have a single app for all social media messages regardless of the social media type.</li> <li>• If you have a separate app for each social media type then select the social media type, such as <b>Social Media - SMS</b>, <b>Social Media - WhatsApp</b>.</li> </ul>

### Overriding the default app for the media type

You can override the default media apps in the Admin Portal if one of the contact center units has different requirements to the other units in the contact center.

For example if a contact center that has several inbound numbers:

- By default, all inbound voice calls use the inbound call app that's set in CE Studio on the **Home > Default Media Apps** page.
- For one of the inbound numbers, the contact center unit is configured to use a different app. This is set in the Admin Portal on the **Media Management > Media Apps** page.

For example if a contact center that handles call backs and also runs outbound dialer campaigns - both are types of workflow:

- Campaign calls configured for *preview mode* present to agents as workflows and use the CE Studio app configured as the default for the workflow media type. This is the app that's set on the **Home > Default Media Apps** page.
- Callbacks are configured for a different contact center unit. This is set in the Admin Portal on the **Media Management > Media Apps** page.

To configure a contact center unit to use a different version of an app:

1. Go to the Admin Portal.
  - To set a different home page for a contact center unit, go to **Media Management > Home Page**.

- To set different apps for each media type used by a contact center unit, go to **Media Management > Media Apps**.
2. Select the required contact center unit, the media type and then the CE Studio app to use at this contact center unit.

**Note** Select **All** when agents are able to use a single CE Studio app for all their work in Agent Desktop.

## Accessing application data

You can access application data, for example data passed through from a running queue progression script that is configured with script elements. This requires the IFS CE System provider:

Chat	IFS CE System provider version 5 or later (or IFS CE Chat provider)
Email	IFS CE System provider version 8 or later

### AppData and the chat provider

See [Using application data with chat](#) for details.

### AppData for other providers

1. Add the IFS CE System provider to the CE Studio app.
2. Add a component, such as a form.
3. In the field selector:
  - a. Select the IFS CE System provider.
  - b. Enter `GetApplicationDataResponse.Value` as the data type.
 

This is the only data type and there are no properties to search on. The properties are the key-value pairs from the application data (JSON blob). For example, if accessing data from a Script Element in a queue progression script, then the key is the name of the script element. The name is case sensitive.
  - c. Enter each property and its data type.
4. Add an action to get the data. Use the `GetApplicationDataResponse` method `GetApplicationData`.

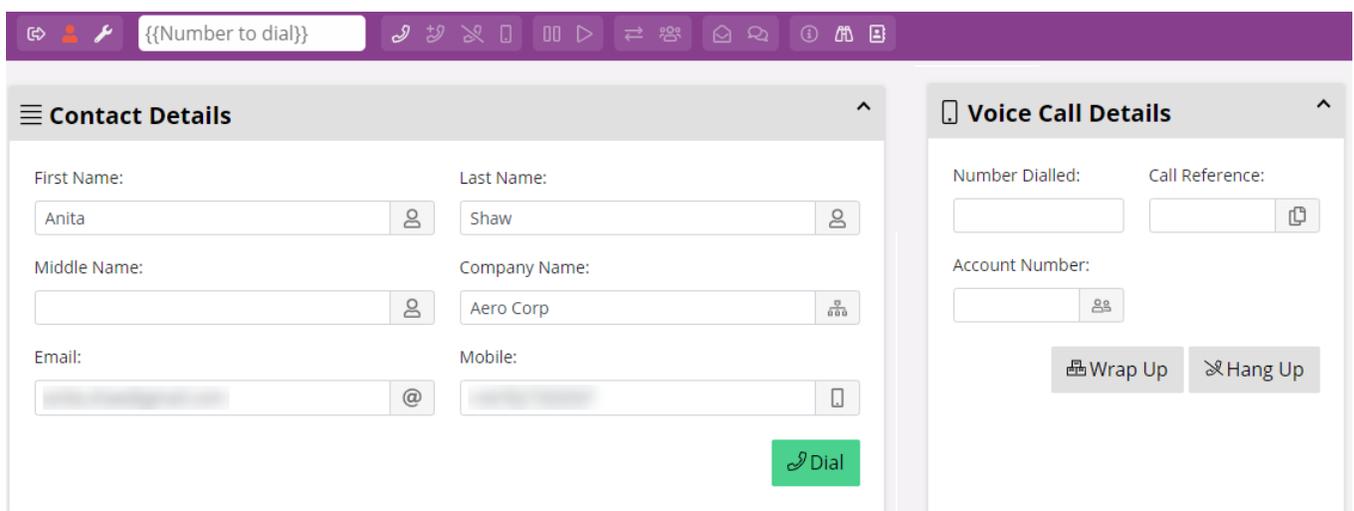
Media type	Parameter mapping
Chat	Map <i>id</i> to the Toolbar Query Param <i>incomingCallid</i>

Media type	Parameter mapping
Email	Map: <ul style="list-style-type: none"> <li><i>id</i> to the email conversation id</li> <li><i>mediaType</i> to the static value <code>Email</code></li> </ul>

## Accessing call data

You do not need a provider for call data. Current call data is always available to CE Studio apps and automatically populates the call data fields in the app. Call data is generated by all media types. The CE Studio app must be used as the home page, for example, call data is not available if the app is set as the Forced Wrap Up media type.

When configuring actions, you access call data through the Toolbar Query Param source.



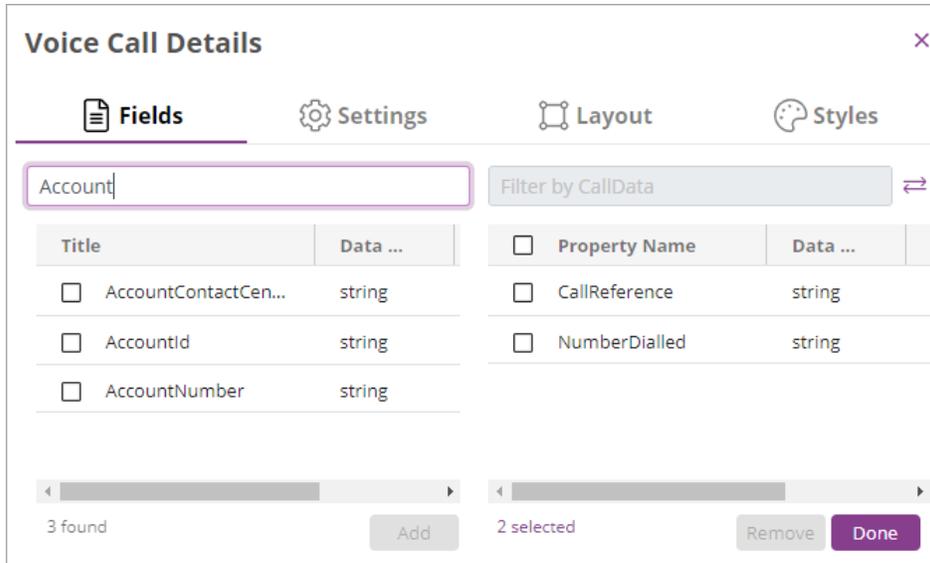
In the above app, clicking **Dial** makes an outbound call to the phone number from the contact details. This creates an activation and the call details are shown in the app. The agent can hang up and wrap up the call by using either the CE Studio app or the Agent Desktop toolbar.

*Figure: A CE Studio app that shows the call details when agents dial out*

## Adding call data properties to components

You add call data properties to components from the **Call Data** menu:

1. On the Designer page, add a new component or click  on the component toolbar to add call data to an existing component.
2. Click .
3. Select **Call Data** and then add the required properties:



### Accessing call data in methods and conditions

The Toolbar Query Param source provides access to the call data captured by Agent Desktop. You use this data source in two ways:

- When using the Client provider types and methods. For example, you map the `emailInfo` type method `getEmailInfoCallback` to the `ActivationId` field provided by the Toolbar Query Param source.
- By creating conditions in rules and action filters that make use of information from the current activation in Agent Desktop.

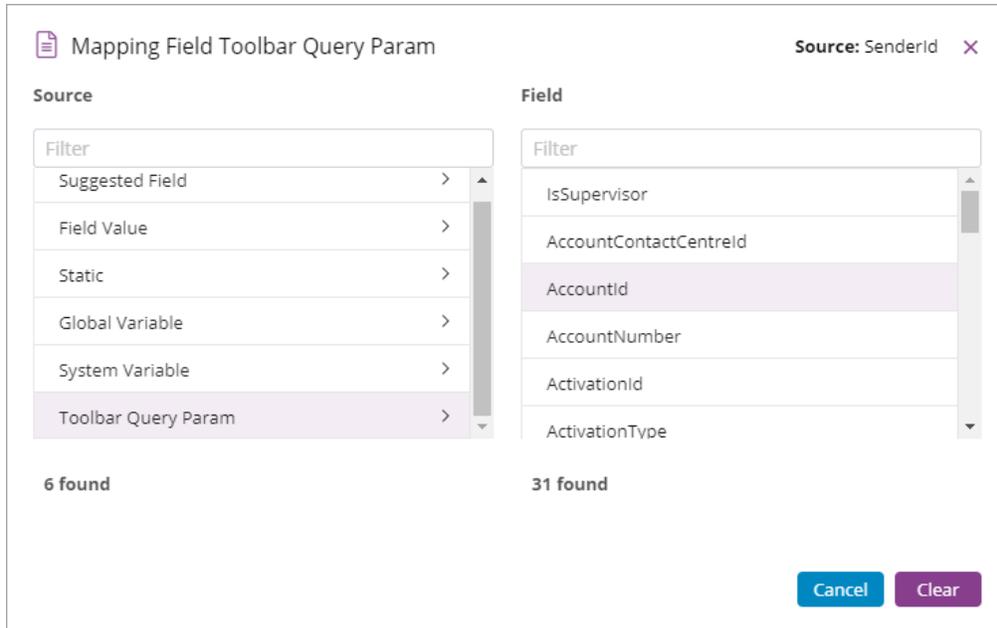


Figure: Mapping the outbound email SenderId to the account ID of the current activation in Agent Desktop

### Call data properties

The following table summarizes some of the properties used to identify activations and calls:

Call data property	Notes
Accountid	The internal code of the contact center unit. Use this when you need to associate something, such as an email SenderId, with the contact center unit that handled the activation.
AccountNumber	The unique code of the contact center unit as set in the Admin Portal.
ActivationId	The unique ID of the activation which identifies an interaction with an agent. Any task performed in Agent Desktop will have an activation ID. For example, use this when you need to get the call, email and so on associated with an activation or when you need to associate an outbound call or email with the current activation.
ActivationType	For a list, see <a href="#">General methods in the IFS CE Agent Desktop provider</a> .
ApplicationData	The DDI name or reference given to the DDI number as configured in the Admin Portal on the <b>Media Management &gt; Voice</b> page.

Call data property	Notes
CallReference	An 8-digit reference that's given to the voice call, email, work object or chat session. It is not guaranteed to be unique but serves as a reference that can be given to callers or used when searching.
CurrentOutboundNumber	Outbound phone number dialed by the agent, either through the toolbar or through a CE Studio app, Also for outbound calls dialed by the system, such as campaign calls.
IncomingCallId	A unique ID that's given to the voice call, email, work object or chat. An <i>IncomingCallId</i> may be associated with more than one activation ID.  For example, when an agent works on the email and then puts the email back on the queue, the email will be presented to another agent. The <i>IncomingCallId</i> and <i>CallReference</i> will be the same but the <i>ActivationId</i> will be different.
NumberDialled	Outbound phone number dialed by the system, for example, for campaign calls.  <b>Note</b> In the case of campaign calls where multiple phone numbers are requested, <i>NumberDialled</i> shows all the numbers that were successfully dialed. The first number in the list is the number that connected.
SkillsetName	The name of the skillset assigned to the activation.
InitialSkillsetName	The name of the first skillset assigned to the activation if the skillset was changed, for example by the queue progression script.

## Using the Toolbar Query Params - Configuration example

You use the Toolbar Query Params when configuring actions, for example, to access events and data from the Agent Desktop toolbar.

The following example configures an action that uses the Toolbar Query Params to get the skillset name from the activation and display it in a CE Studio app.

**Note** To try out this example, you could clone the example chat template.

1. Add a text box that will display the skillset name from the activation.

2. Click **Config** on the text box and then map the text box to the provider (the Chat provider in this example). Select **Skillset** as the provider type and then **Name**.
3. Add an action. The provider type will be `Skillset` and the method will be `GetTabularResponse`.
4. Add a filter to the action:
  - Left side: Map the **Field Reference** to **Skillset > Id**
  - Right side: Map **Toolbar query param** > **InitialSkillsetId**
5. Attach the action to the rule set that is run on the OnLoad event.
6. Configure the OnNotification event for the component that contains the text box.
7. Save the CE Studio app.
8. Test it in Agent Desktop.

Now when the Chat app loads in Agent Desktop, the skillset name will be displayed.

## IFS CE Agent Desktop provider methods

All IFS CE Agent Desktop provider methods must be called by a rule.

### General methods in the IFS CE Agent Desktop provider

When an agent accepts an email, chat or voice call, the communication pops to the agent as an activation. The activation provides the context for many of the actions that the app needs to perform.

#### actions

The following general methods are available on the IFS CE Agent Desktop provider `actions` type.

#### **getActivationType**

Returns the activation type:

- -1, None
- 0, Inbound call
- 1, Forced wrap-up
- 2, Training
- 5, Demo
- 6, Offline entry
- 7, Inbound call transferred internally to agent on another imedia server
- 8, Workflow
- 9, Chat
- 10, Email

- 11, IVR result
- 12, Work definition result
- 13, Outbound campaign call
- 14, Chat result
- 16, Social media
- 255, No activation — agent is in DND mode

### getAgentState

Returns the agent's current state, such as Idle, Ringing. For a list of states, see [Agent states](#).

### getCallId

Returns the ID of the current activation.

### pauseRecording

If call recording is in use, this method pauses the recording. For example, use this while the agent is handling sensitive information. Returns:

Parameter	
success	Returns 0 on success
returnCode	The error returned by the function
agentState	Agent state
agentStateText	Agent state description. See <a href="#">Agent states</a> .

### resumeRecording

When call recording is in use, this method restarts the recording. Parameters are the same as `pauseRecording`.

### wrapUp

Closes the current activation, for example, after ending a call or closing an email. This puts the agent back into the idle state ready for the next activation. When in the idle state, the CE Studio app for the home page is shown. The agent must be in a valid state for wrapping up. Invalid states are: on an active call, in an open email, making a voice recording.

### forceWrapUp

The following method is available on the IFS CE Agent Desktop provider `forceWrapUp` type.

### forcedWrap

When an agent clicks **Forced Wrapup** on the Agent Desktop toolbar, they go to the CE Studio app associated with the Normal, Offline Entry, Demo or Training media types. The default apps for these are set on the Default Media App page.

For example, you can use this approach to select an email on the Agent Desktop home page and then open it in the Offline Entry app (for example). You can then work on the email, such as close it. The email is not in an activation.

To do this this you use the forcedWrap method.

The following parameters are optional:

- `account`: The ID of the contact center unit, entered as a static value.
- `options`: A string, for any additional information you want to record or the identifier for something that you want to open in the CE Studio app associated with the Normal, Offline Entry, Training or Demo media apps, such as an email ID.
- `type`: Any string that identifies this activity. The Forced Wrapup option in Agent Desktop is configured with the types. These are lowercase:
  - normal
  - offline entry
  - training
  - demo

**Important** The above types are case sensitive and must be entered in lowercase.

## Voice call methods in the IFS CE Agent Desktop provider

When an agent accepts a voice call, the call pops to the agent as a voice call activation. This activation provides the context for many of the actions that the app needs to perform. The CE Studio app should contain all the controls need for answering calls, hanging up, and dialing phone calls.

**Important** Ideally, agents should always make outbound calls and accept inbound calls using controls in the CE Studio apps used for voice calls. This gives the best control and the most flexibility. When working on voice calls, agents should be discouraged from using the toolbar. The toolbar should be seen as a troubleshooting tool.

The following provider types and methods are provided by the IFS CE Agent Desktop provider for voice calls:

Provider type	Method	Description
actions	conferenceCallers	Connects the calls to start the conference call. No parameters.

Provider type	Method	Description
actions	connectToEnquiry	Connect the agent to a second call. The second call is termed the enquiry call. No parameters.
actions	connectToPrimary	Connect the agent to the voice call that they have accepted in Agent Desktop. This call is termed the primary call. You also use this method when you need to reconnect the agent to the primary call. No parameters.
actions	holdCall	Puts a call on hold. To retrieve the call use <code>connectToPrimary</code> . No parameters.
actions	transferCall	Connects the primary call to the enquiry call. Once the agent leaves the call, the agent will be in a wrap up state because they are still in the activation. No parameters.
chatEvent	EVT_OUTBOUND_STATE_CHANGE	Notifies the app when an event occurs on an outbound call, such as ringing, call answered, disconnected. See <a href="#">Agent Desktop events</a> .
dialInfo	dialOutEx	Places an outbound call to the number entered in or obtained from a form field or entered in the field on the Agent Desktop toolbar. The parameters are: <ul style="list-style-type: none"> <li>• <i>cli</i> – caller id may be required by the trunk provider. You can map this to the <code>ToolbarQueryParam CallersNumber</code>.</li> <li>• <i>flags</i> – map this to: <ul style="list-style-type: none"> <li>• 0 – System will retry the number for the period set in the <b>Outbound Call Ring Timeout</b> field on the Advanced Settings page in the Admin Portal.</li> <li>• 1 – System will dial the number once only.</li> </ul> </li> <li>• <i>number</i> – the number to dial</li> <li>• <i>timeout</i> – map this to the number of seconds otherwise 0</li> </ul>
dialInfo	hangUp	Hangs up the current inbound or outbound call.

## Email and chat methods in the IFS CE Agent Desktop provider

For details of the IFS CE Agent Desktop provider types and methods for email and chat, see the following templates that are provided as part of the tenant deployment:

- IFS CE Email Template — for details, see [Email](#)
- IFS CE Chat Template — for details, see [Chat](#)

## Agent Desktop events

You can configure actions to respond to specific events in Agent Desktop, such as when there is an agent state change, when a call ends or when the app changes state. Agent Desktop packages the event and sends the data to the CE Studio app. The app container can be configured to listen for these events using the toolbar event type `EVT_OUTBOUNDCALL_STATE_CHANGE`.

You can configure a CE Studio app to respond to these events using the IFS CE Agent Desktop provider. This has a `chatEvent` provider type with the following methods that can be used for more than chat:

- `EVT_OUTBOUNDCALL_STATE_CHANGE`
- `EVT_AGENT_STATE_CHANGE`

### Configuring the action

To configure an action that responds to a Agent Desktop event:

1. In a new action, select the provider IFS CE Agent Desktop provider.
2. Select **chatEvent** as the provider type.
3. Select the **EVT\_OUTBOUNDCALL\_STATE\_CHANGE** or **EVT\_AGENT\_STATE\_CHANGE** method.
4. Click **Manage Properties**.

There are no required parameters. You can add the optional properties that are needed for your action.

5. In the Parameter Properties dialog, click in the **chatEvent.** field and select **message** from the list.

This displays all the event properties. Many of these properties are specific to chat. The IFS CE Chat Template demonstrates how they are used.

6. Add the properties you require for your method and then click **Done**.

### General event data properties

The following table lists the general properties in `chatEvent`:

callID	The activation ID
--------	-------------------

data	The event data as sent by Agent Desktop
eventText	The name of the event. Use this rather than the <code>eventCode</code> when configuring conditions. For a list of these, see <a href="#">Call states</a> .
newState	The agent's current state. For a list of these, see <a href="#">Agent states</a> .
oldState	The agent's previous state.

### Running rule sets and display task sets from the OnSuccess status of an event

You can now run rule sets and display task sets from the OnSuccess status of events for Agent Desktop provider actions (currently chat only). In the action that is run, you must publish the topic and the top-level component must subscribe to the action.

For example, in a chat template you could run an action after a reply is sent to the chat caller. To do this:

1. Go to the Outgoing Message action set.
2. In the Outgoing Message action, select the **Publish Topic** check box.
3. On the Event Configuration page, click the top-level component and then select the OnNotification event. Subscribe the component to the Outgoing Message action.
4. On the Event Configuration page, click the Reply button and then select the OnClick event and its OnSuccess status. Select the action, rule or display task set that you want to run.

### Running rule sets after an app state change

See [App state change event](#).

## Call states

The call state tells you whether an inbound or outbound call is ringing, answered, or ended. It determines the active state of the Agent Desktop toolbar buttons. When the call state changes, you can run a rule set.

**Note** Configuring call states requires version 6 of the IFS CE Agent Desktop provider.

To run a rule set when the call state changes:

1. Add a rule set and a rule. See [Rules and rule sets](#).
2. Add a condition.
  - a. On the left side of the condition, select **Event Context > App** and then select a property, such as `Toolbar.CallState.NewState`.
  - b. Select an operator, such as **Equals**.

- c. On the right side of the condition, select **Static** and then select the required state (see below).
3. On the Event Configuration page, configure the event that will run the rule set:
  - a. Select the outermost container (the app).
  - b. In **Event Type**, select **OnToolbarEventReceived**.
  - c. In **Toolbar Event**, select **EVT\_OUTBOUNDCALL\_STATE\_CHANGE**. This event covers both outbound and inbound calls.
  - d. In **Run Rule Set**, select your rule set.

The complete list of call states is listed below. An enquiry call is a second call, for example, you put the inbound or outbound call on hold while you make another call.

- NONE
- INBOUND ENDED
- OUTBOUND FAILED
- OUTBOUND RINGING
- OUTBOUND ANSWERED
- OUTBOUND ENDED
- ENQUIRY FAILED
- ENQUIRY RINGING
- ENQUIRY ANSWERED
- ENQUIRY ENDED

## Agent states

The agent state tells you what the agent is currently doing or what they are connected to. It determines the active state of the Agent Desktop toolbar buttons. When the agent state changes, you can run a rule set. Both the IFS CE Inbound Voice Template and the IFS CE Chat Template provide examples of disabling a button when the agent goes into the wrap up state.

To run a rule set when the agent state changes:

1. Add a rule set and a rule. See [Rules and rule sets](#).
2. Add a condition.
  - a. On the left side of the condition, select **Event Context > App** and then select a property, such as `Toolbar.AgentState.New`.
  - b. Select an operator, such as **Equals**.
  - c. On the right side of the condition, select **Static** and then enter the number of the state (see below).
3. On the Event Configuration page, configure the event that will run the rule set:
  - a. Select the outermost container (the app).
  - b. In **Event Type**, select **OnToolbarEventReceived**.

c. In **Toolbar Event**, select **EVT\_AGENT\_STATE\_CHANGE**.

You cannot use this event to run an action directly. You need to run a rule set or use the Clicks display task to trigger the action to run. For details, see [Clicks display task](#).

d. In **Run Rule Set**, select your rule set.

The complete list of agent states is listed below. For notifications of agent state changes, use the Agent Desktop provider type `chatEvent` and its method `EVT_AGENT_STATE_CHANGE`.

Agent state name	Number	Description
IDLE	1	Agent is ready for the next activation or in DND mode.
RINGING	2	An activation (any type) is presented to the agent but they haven't accepted it yet.
WAITING_FOR_CONN	4	Applies mainly to voice calls where the agent has accepted the call but there might be a very short delay before the call is connected.
INBOUND_CALL	8	Agent is connected to an inbound call.
ON_HOLD	16	Agent is connected to an active inbound or outbound call that's on hold.
WRAP_UP	32	<p>Agent has ended the call, email or chat, and is now in the wrap up state. When they leave the wrap up state they go back to IDLE.</p> <p><b>Note</b> For work objects and offline working, agents start in the wrap up state. For example, for callbacks, agents start in the wrap up state and their status updates when they place the call.</p> <p><b>Note</b> For apps accessed by clicking  on the Agent Desktop toolbar, the agent will be shown as being in the wrap up state.</p>
CALLER_RING_OFF	64	Caller rings off before the agent is able to pick up the call. Agent goes back to IDLE when this occurs.
OUTBOUND_CALL	128	Agent has made an outbound call and is talking to the caller.
ENQUIRY_CALL		Agent has two active voice calls (primary and enquiry) and is talking to the enquiry caller. For example, before transferring the call.

Agent state name	Number	Description
CALL_RETRIEVED	1624	Occurs when the agent is in the RINGING state and doesn't pick up a call quickly enough within the timeout. The agent is put into the Do Not Disturb state and the call is presented to someone else.
INBOUND_CALL_ENQUIRY_ON_HOLD	1625	Occurs when the agent has two active calls and is talking to the primary caller (an inbound call). The enquiry call is on hold.
CONFERENCED		All parties in a call are talking to each other.
OUTBOUND_CALL_ENQUIRY_ON_HOLD	1626	Occurs when the agent has two active calls and is talking to the primary caller (an outbound call). The enquiry call is on hold.
CHAT	16384	Inbound chat answered by the agent.
EMAIL	32768	Inbound email accepted by the agent.
SOCIAL_MEDIA	65536	Inbound social media message accepted by the agent.

## App state change event

**Note** You can use this event as an alternative to using a Clicks display task. See [Clicks display task](#).

As standard, the home page app is not reloaded when you return to the home page app from an activation. This means there is no OnLoad event to run a display task or an action, however, you may need to update the app in some way.

You can update the home page app by using the EVT\_APP\_STATE\_CHANGE toolbar event to run a rule set after, for example:

- Returning from an activation to the home page app in Agent Desktop.
- Leaving the monitoring page and returning to the home page app

The EVT\_APP\_STATE\_CHANGE toolbar event has two states, Activate or Deactivate. You can decide what to run for each state. For example, in the Activate state you could run a timer display task that invokes an action and in the Deactivate state you could disable the timer and prevent the action from invoking.

To configure this event:

1. Create a rule set:
  - a. Add a rule with a condition. In the condition, map the:
    - Left side of the condition to an event context **Event Context > App > Toolbar.AppState**

- Right side to either **Activate** or **Deactivate**
- b.** Select the display task to run when the rule evaluates to *true*.  
For example, when the state is Activate then run a display task to start a timer to refresh the home page and when the state is Deactivate then disable the timer.
  - 2.** Go to the Event Configuration page.
  - 3.** Select the app.
  - 4.** In **Event Type**, select OnToolbarEventReceived.
  - 5.** In **Toolbar Event**, select EVT\_APP\_STATE\_CHANGE.
  - 6.** Select the rule set that you want to run.

## Error and return codes

- 0, Success
- 1, Unknown error
- 6, Invalid arguments
- 13, Busy already
- 14, Idle already
- 15, Find failed
- 16, Check failed
- 2005, Command ignored

# 13

## Last-Mile Customer Portal

**Note** Previously called Last-Mile Technician Portal.

CE Studio apps for Last-Mile Customer Portal can:

- Provide all necessary information about the appointment, such as technician's name, planned time, and scope.
- Provide convenience to the customer by making sure the service appointment minimizes disruption to their plans for the day.
- Allow the customer to easily contact the service provider to ask questions or change the appointment.
- Allow the organization to improve efficiency in field service by making the customer part of the digital end-to-end-process.
- Optionally, ask the customer to complete a survey.

Any app for Last-Mile Customer Portal requires a backend system which is either Field Service Management or IFS Cloud.

Example apps are provided for several different providers, and is ready to use, once you have completed these steps:

1. Complete the configuration in the Admin Portal: specifically credentials, dataset, and rules.

How to configure these are described in the Admin Portal help.

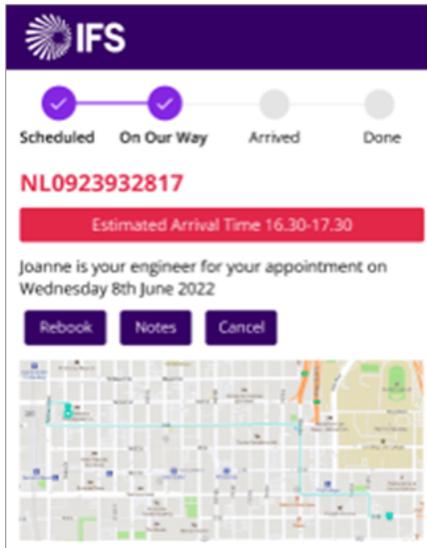
2. Download the CE Studio apps for Request Handling and Field Service Management from the Central Template Repository.

**Note** Existing customers already using Work Order Handling can raise a request to obtain the CE Studio app for this endpoint.

3. Make the app live.

For background information on the example app, see [Last-Mile Customer Portal apps](#).

## Last-Mile Customer Portal apps



*The main elements in a Last-Mile Customer Portal app*

You can download Last-Mile Customer Portal apps from the Central Template Repository.

### Stepper

A Stepper element shows the progress. Each step is configured for a specific activity status.

The GoTo Step display tasks makes each step active and rules control when each step is made active. See [Stepper elements and display tasks](#).

### Labels

Label elements show the activity details. The information is obtained by the IFS CE Last-Mile Customer Portal provider. See [Actions for Last-Mile Customer Portal apps](#). The action publishes a topic so that the stepper, labels and map can subscribe to it.

### Map

A Map element shows the location of the resource and their route to the place where the activity will be carried out.

## Configure Last-Mile Customer Portal app for surveys

Users can take a survey once the engineer has completed the task. This is an optional step that you can configure if you require it. The survey is partially configured in the Last-Mile Customer Portal and is disabled by default.

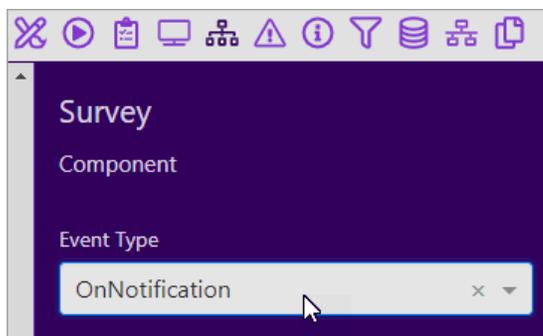
To enable surveys in the Last-Mile Customer Portal app

**Note** First configure a survey with survey questions in the Admin Portal. Make the survey version live. This version contains the questions that you want to include in the survey. You also need to configure a rule on the **Admin Portal > Last Mile Customer Portals > Rule Configuration** page to send the customer a link to the survey once the appointment is completed.

1. In CE Studio Designer, on the Designer page, go to the overlay called **Overlay - Survey**.
2. Add a Survey element and configure it for the survey that you have set up in the Admin Portal.

For details of this step, see [Configuring the survey element](#).

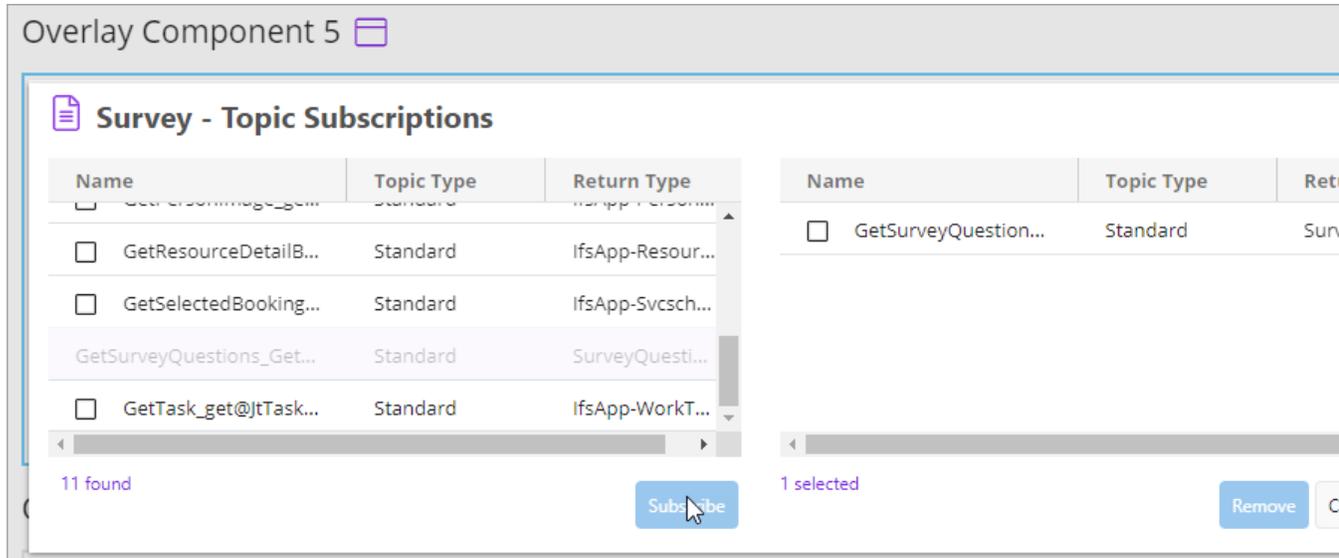
3. The actions are already created. Update the actions for the name of the survey and the name of your Last-Mile Customer Portal app:
  - AddSurveyCandidateRequest
  - SubmitSurvey
4. Go to the Event Configuration page, and in Overlay Component 5 (Survey):
  - a. Click the overlay - Survey Component is displayed in the righthand pane:



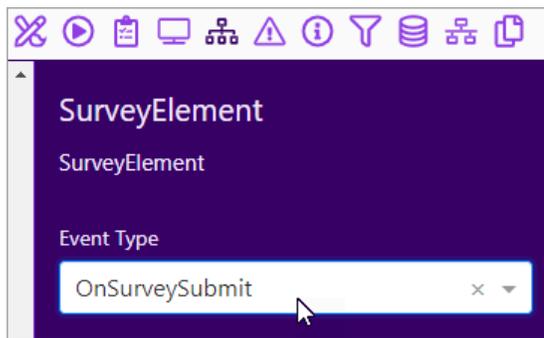
- b. From the **Event Type** list, select **OnNotification**.

The Survey - Topic Subscriptions dialog opens.

- c. In the Survey - Topic Subscriptions dialog, select `GetSurveyQuestions_GetQuestions_Response` and then click **Subscribe** and **Done**.



d. Click the survey element - SurveyElement is displayed in the righthand pane:



e. From the **Event Type** list, select **OnSurveySubmit**.

f. In **Run Action Set**, select the **SubmitSurvey** action.

5. Go to the Rules page.

6. In the StatusCheck rule, attach the rules **AddSurveyCandidateRequest** and **ShowSurveyWhenJobCompleted**.

6) Save the app.

7) Make the app live in order to test it.

## Configuring a custom Last-Mile Customer Portal app

A preconfigured Last-Mile Customer Portal app is available for download from the Samples in this help. You can clone and then modify the app to suit different business requirements.

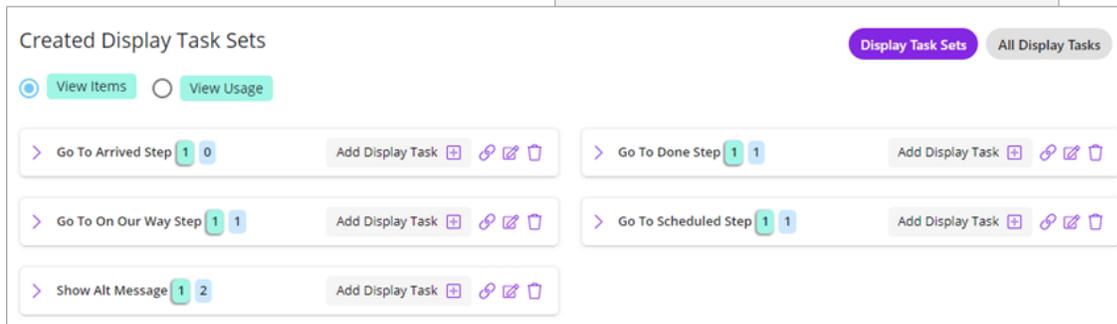
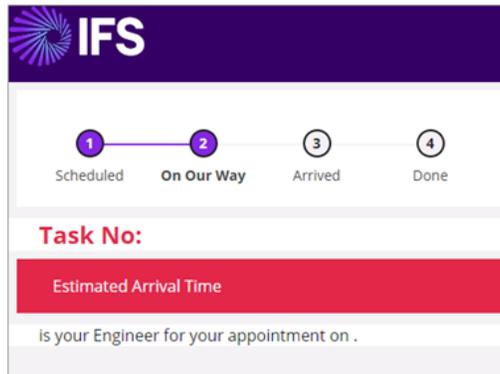
The following topics provide some information about how the example Last-Mile Customer Portal app is configured:

- [Actions for Last-Mile Customer Portal apps](#)

- *Event configuration for Last-Mile Customer Portal apps*

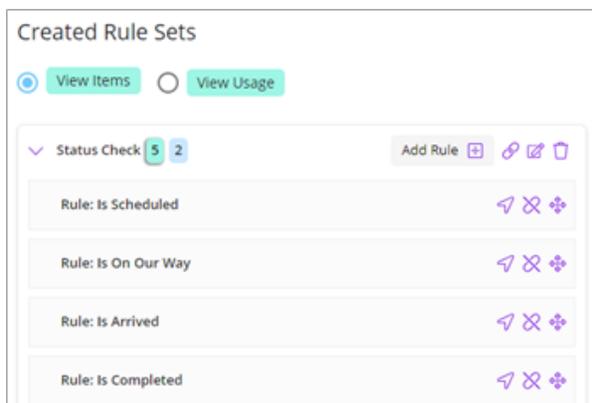
## Stepper elements and display tasks

You use the Goto Step display task to move between the steps in the stepper element used in the example Last-Mile Customer Portal app.



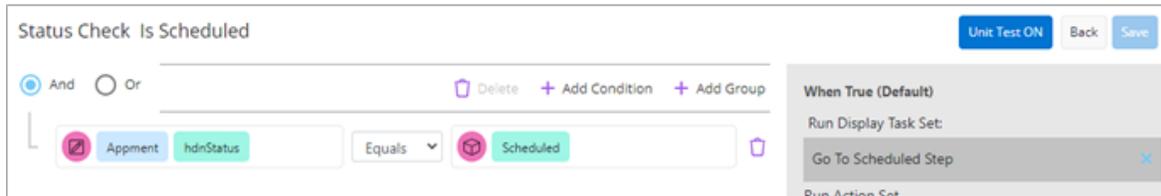
*A GoTo Step display task is configured for each step in the Stepper element*

Use a rule to check what the activity status is before running the display tasks. In the following example, when the activity status changes to **Scheduled**, the rules in the rule set are evaluated. The **Is Scheduled** rule matches the activity status and this rule then runs a display task to make the Scheduled step into the active step.



*A rule set with a rule for each step in the Stepper element*

Each rule contains a condition that checks the activity status. If the condition evaluates to *true* then the GoTo Step display task in the rule is run.



*Example of a rule that checks the activity status*

## Actions for Last-Mile Customer Portal apps

The example app has a single action to get the details for the activity Id passed in the secure link.

### Link parameters for secure links

Customers access the Last-Mile Customer Portal app using a secure link. This feature requires link parameters for the values that are passed in the secure link. There are three link parameters:

- ActivityId - string
- BackendSystem - string
- DataSetId - string

Add these on the Link Parameters tab:

1. On the Designer page, click **App Settings**.
2. Go to the **Parameters** tab.
3. Go to the **Link Parameters** tab.

### Configure the action to get the activity details

You need an action to get the activity details for the activity passed in the secure link. This action is for the Last-Mile Customer Portal provider:

- Provider type: ActivityResponse
- method: GetActivity

Click **Manage Properties** button and add the following optional parameters:

- ActivityId
- BackendSystem
- DataSetId

Map these parameters to values in the Session, specifically to Session > AppData.

The action must publish the topic because all the components in the app will subscribe to it. The stepper element, the labels and the map all use data from the activity response.

## Event configuration for Last-Mile Customer Portal apps

The example Last-Mile Customer Portal app is configured with these events:

Event type	Purpose
OnLoad OnInvoke	When the example Last-Mile Customer Portal app loads, it runs the <i>action</i> to get the activity details. Use the OnLoad OnInvoke event for this.
OnTimerElapsed OnInvoke	The app refreshes whenever the details are updated. Use the OnTimerElapsed OnInvoke event for this. This event runs the <i>action</i> to get the activity details again.

The app contains a stepper element that updates to show the current status of the activity. This is driven by a rule set that contains a rule for each of the statuses. If the app is successful in getting the activity details, it runs the rules to update the stepper. These events trigger this:

Event type	Purpose
OnLoad OnSuccess	When the example Last-Mile Customer Portal app succeeds in getting the activity details, it runs the <i>rule set</i> to check the activity status. Use the OnLoad OnSuccess event for this.
OnTimerElapsed OnSuccess	When the app refreshes, re-run the rule set to check the activity status.

All the components in the app subscribe to the activity response - the OnNotification event:

- The stepper element
- The label elements
- The map element

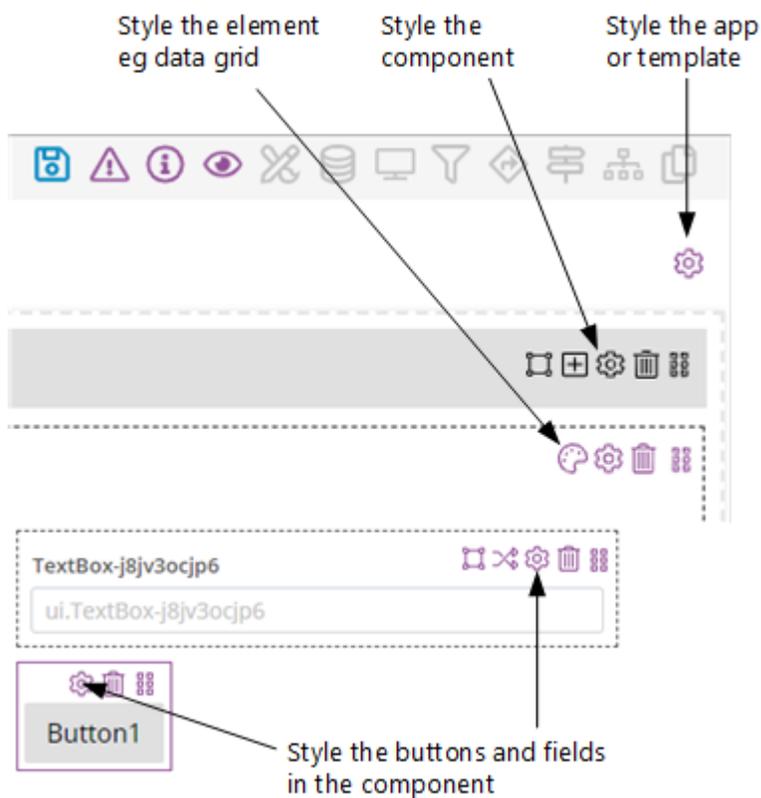
# 14

## Styles and themes

Styling CE Studio apps and individual components.

You can style the CE Studio app to apply an organization's branding and improve usability for users. To make it easier to apply the style consistently use themes.

All the style options are available on the following toolbars on the Designer page:



To configure the themes, on the side menu, click **Global Configuration > Themes**.

## Themes

---

There are system themes, for example **IFS Theme** is the default theme, and you can create your own themes to style most of the components in a CE Studio app. Not all components support themes - check the release notes for details of current limitations.

When you select a theme for a CE Studio app, any styles already applied in the app are preserved. Currently it is not possible to revert the override changes.

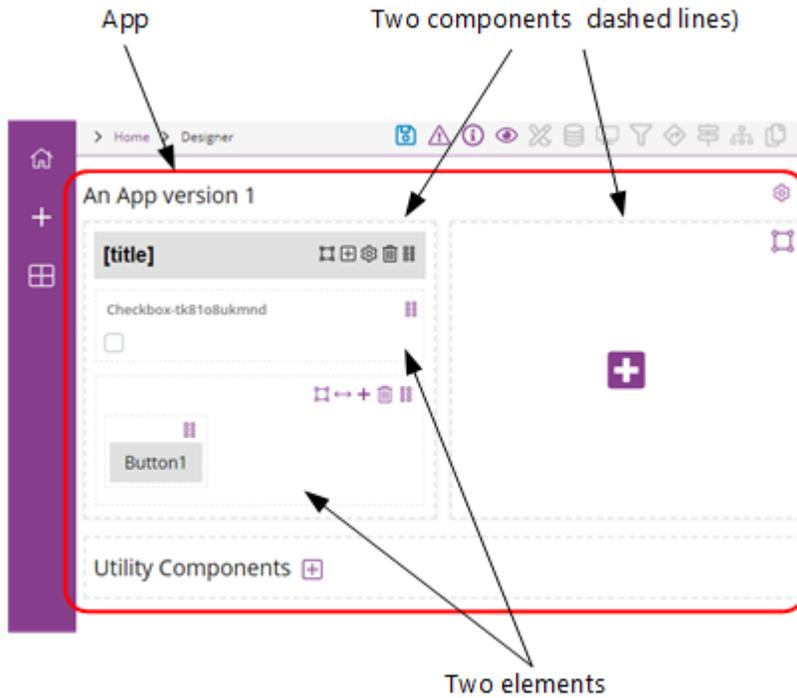
## Styling a CE Studio app

---

You can change the look and feel of a CE Studio app by setting, for example, colors, spacing, rounded corners, dropped shadows, icons. You can apply styles to:

- The whole app
- Individual components in the app (in Designer, these are the areas in dashed lines)
- Elements within components (provided that the component is set to use the default style option **Inherit from App**)
- Button groups
- Any templates embedded in the app

By default, styles are set at the top level, that is at the app level, and can then be inherited by the components, elements and templates in the app. If you do not want this then you style the individual components and templates.



*Apps, components and elements*

To change the style of the app, go to the Designer page and click  **App Settings**. You can set:

Field	Description
Theme	You can select a theme as the starting point for the app. In particular, this sets the default color of headers and buttons. <i>Default:</i> IFS Theme <i>Scope:</i> Everything that is set to <b>Inherit from App</b>
Typeface	You can change the typeface but this will only apply to headers. <i>Default:</i> Open Sans. This is an Open Source font and is used for headers, labels and buttons.
Display Density	Set the spacing around components, elements and labels. This is only visible when the app is run in Agent Desktop. <i>Scope:</i> App level only
Translation Provider	Used when localizing the labels. In IFS Services these are supplied by the providers in the app or template. In IFS Customer Engagement these can also be supplied from the Message Store.

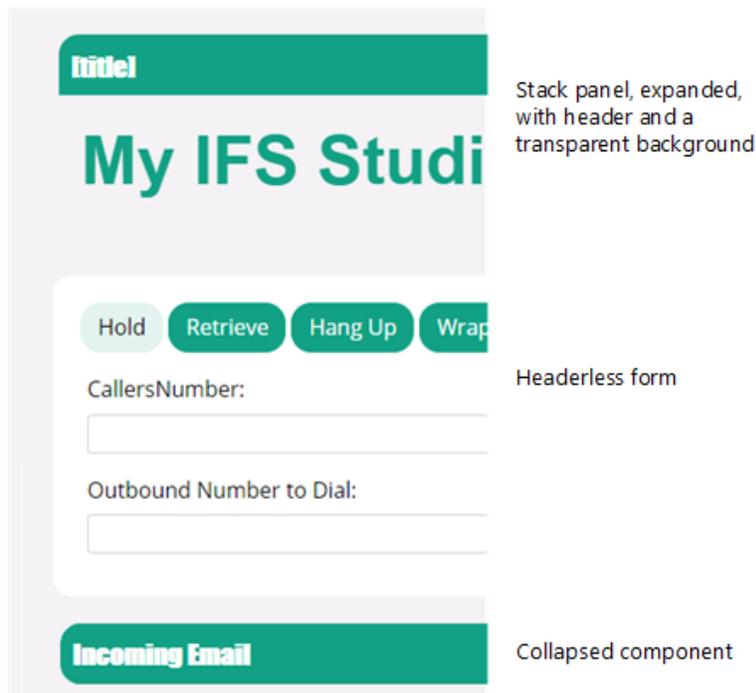
Field	Description
Rounded	Whether you want rounded corners on components, headers, fields and buttons. You can enter any value in the range 0 (square corners) through 15 (rounded corners).  <i>Scope: App level only</i>
Drop Shadow	By default, there is a narrow drop shadow around components. This is visible in both Designer and Preview.

To ensure consistency you can't change some styles:

- The app always has a pale gray background
- Components and elements always have a white background (unless you make it transparent)
- The default font, Open Sans, is always used for labels and buttons

## Styling components

In Designer, a component is the set of elements inside the dashed rectangle. There are several styling options for components.



*Three components when previewed*

**Header**

Template Placeholder: Standard Header

Preview the app to see how the styling is applied to the template

---

**Headerless**

Hold Retrieve Hang Up Wrap

Callers Number  
Cti.CallersNumber

Outbound Number to Dial  
ui.TextBox-76wjxtr6lov

The header of headerless components is grayed out

---

**Header**

Collapsed components are expanded

*How the same components are shown in Designer*

To change the style of a component, click  on the component toolbar and then go to the **Styles** tab. You can change:

Field	Description
Headerless	Whether the shaded banner above the component is visible. See above for an example. A version of the banner is always shown in Designer so click  to see the result.

Field	Description
Background Transparent	<p>The app or template has a pale gray background. Select:</p> <ul style="list-style-type: none"> <li>• <b>No</b> if you want the component to have a white background</li> <li>• <b>Yes</b> if you want the component to be the same color as the app/template background.</li> </ul> <p>Useful for Stack Panels for example.</p>
Header Theme	<p>By default, the color of the component header is the same as the app – the <b>Inherit from App</b> option.</p> <p>To override the app, select a color for the component header from the list. This becomes the default color of all of the button groups in the app.</p>
Header Tint/Shade	<p>You can set the tint of the header. The dropdown list shows you the options and how the font color changes with the tint.</p> <p>Choose <b>None</b> if you want the header to be the same color as the background.</p>
Add an Icon	<p>You can add an icon to the header to make the element more identifiable.</p> <p>Click  to display the icon before the label</p> <p>Click  to display it after the label</p>
Inherited Properties	<p>The roundness of the corners is inherited from the app or template and cannot be modified.</p>

**Note** To set Tooltips, position of the tooltip when open and message IDs for localizing the app, click  on the component toolbar and then go to the **Settings** tab.

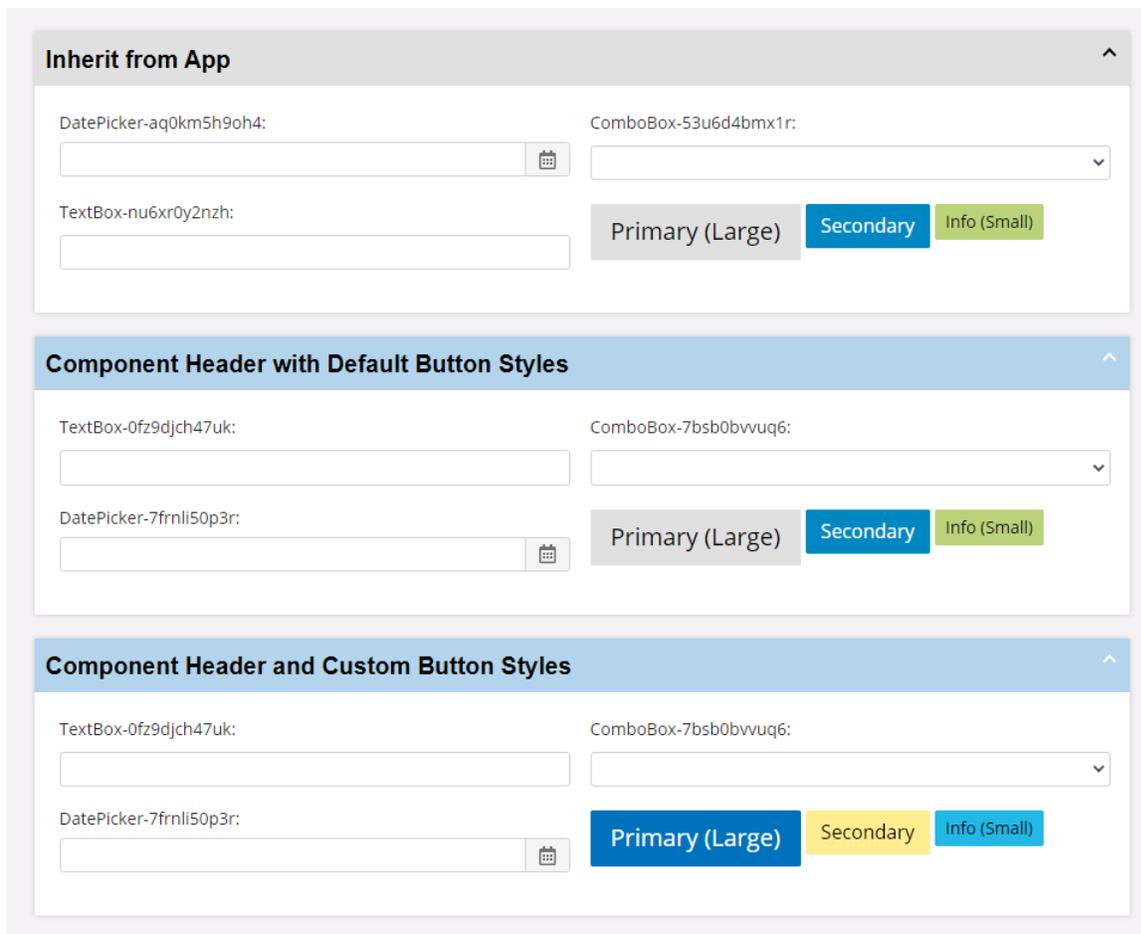
## Templates and inherited styles

Styling a template is similar to *styling a component* with the exception of button groups. To style a button, click  on the button toolbar.

The following button properties are independent of the selected theme. These are:

- Base style: primary, warning, success, secondary
- Size
- Solid or Outline
- Background tint
- Hover color tint
- Border tint

The button color however is determined by the selected theme. Consider this example of an app that embeds three templates:



The screenshot displays three distinct templates within a single application window, each with its own header and a set of form controls and buttons.

- Template 1: Inherit from App**
  - Form controls: DatePicker-aq0km5h9oh4, TextBox-nu6xr0y2nzh, and ComboBox-53u6d4bmx1r.
  - Buttons: Primary (Large) (grey), Secondary (blue), and Info (Small) (green).
- Template 2: Component Header with Default Button Styles**
  - Form controls: TextBox-0fz9djch47uk, DatePicker-7frnli50p3r, and ComboBox-7bsb0bvuuq6.
  - Buttons: Primary (Large) (grey), Secondary (blue), and Info (Small) (green).
- Template 3: Component Header and Custom Button Styles**
  - Form controls: TextBox-0fz9djch47uk, DatePicker-7frnli50p3r, and ComboBox-7bsb0bvuuq6.
  - Buttons: Primary (Large) (blue), Secondary (yellow), and Info (Small) (blue).

### Example – the Inherit from App template

The first template in the above example inherits its styles from the app. The app theme determines the color of the header and the buttons in the button group.

### Example – the Component Header with Default Button Styles

In the second template in the above example:

- The header is set to the Blue theme
- The buttons in the button group inherit their theme from the app. This means that their color doesn't change when you change the header theme.

### Example – the Component Header with Custom Button Styles

In the third template in the above example:

- The header is set to the Blue theme
- Each button in the button group is individually set to use the Blue theme. This determines the color of the button.

# 15

## Localizing strings in the user interface

Using the Message Store and Translation providers.

The Message Store, in the Admin Portal, provides a single page for managing localization. Using the Message Store, you can customize or localize the strings used in Agent Desktop and by the chat client. After configuring messages for a component, you can view localized strings on the Preview page.

You manage the list of supported languages in the Admin Portal on the **Contact Center > Manage Languages** page.

It is important to note that CE Studio Designer only loads the messages for the language set in your user profile (as configured in the Admin Portal). This means that either the messages must exist for both your default language and the target language, or you must change your user profile to the target language.

The language used by a CE Studio app depends on the type of app:

Media and portal apps	Depends on the language selected in the user profile in the Admin Portal under <b>User Management</b> .
Public portal apps	Depends on the browser locale.

**Note** It may take an hour or longer for the CE Studio message cache to be reset depending on the most frequently accessed items in the cache.

**Note** CE Studio Designer is provided in English only.

## Translation providers

---

You can choose the source for the translations of the user interface strings. The translation provider can be the CE message store, for which you require the IFS CE System provider, or from one or more of the data providers used in the app or template.

To set the translation provider for the app or template:

1. Click **App Settings** .
2. In **Translation Provider**, you can:
  - Either select one of the data providers as the single translation provider for all the user interface strings.
  - Or select **Multiple** which allows you to select the provider when you set the message ID for a field, button or header.

**Note** Translations from the Message Store are provided by the IFS CE System provider.

## Translating strings in the user interface

---

Agent Desktop uses the strings for the agent's default language and displays any missing strings in curly brackets. An agent's default language is set in their user profile. This also applies to portal apps. The language of a public portal app is determined by the user's browser locale.

**Note** Message IDs and localized strings are managed in the Admin Portal in the message store, and then assigned to properties, buttons and prompts (display tasks) by clicking .

### Adding languages

By default, only US English, UK English and English are supported but you can add additional languages in the Admin Portal, on the **Contact Center > Manage Languages** page.

### Adding message IDs and messages to the CE message store

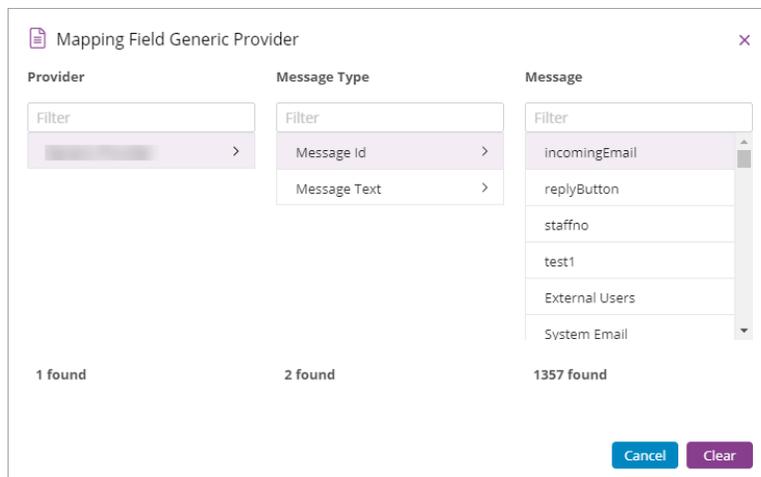
1. In the Admin Portal, on the Message Store page, click **New**.
2. Select the language.
3. In **Message Type**, select **Label**.
4. Enter the message ID and the string to use.

## Assigning message IDs in CE Studio Designer

For example, to localize the title of a component:

1. In CE Studio Designer, edit the element you want to localize. For example
  - Click  on the component toolbar and go to the **Settings** tab.
  - Click  on the button toolbar.
2. Click :
  - a. In **Default text**, enter the default string. This defaults to the title if you leave it empty and there is no message in the message store.
  - b. Click in **Message Id**.

The Mapping Field dialog opens.

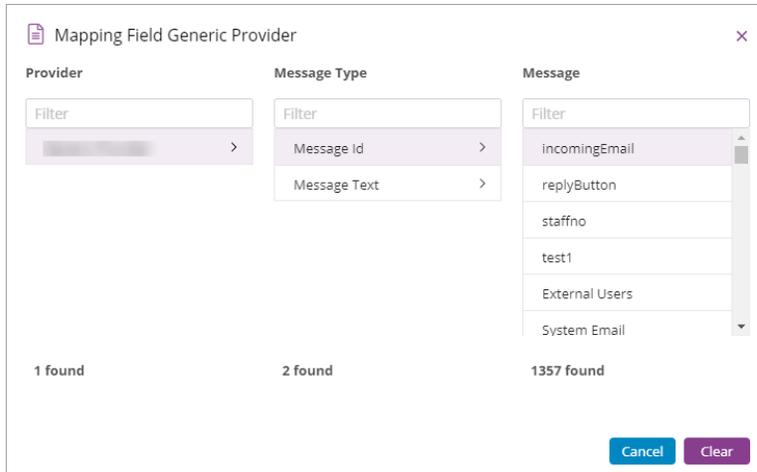


- c. Select the provider that is the source of the messages for this element.
- d. Select the message.
- e. Preview the app or template to see the message.

## Removing a message ID

1. Click .
2. Click in **Message Id**.

The Mapping Field dialog opens.



3. Click **Clear**.

## ABOUT IFS

IFS develops and delivers enterprise software for customers around the world who manufacture and distribute goods, maintain assets, and manage service-focused operations. The industry expertise of our people and solutions, together with commitment to our customers, has made us a recognized leader and the most recommended supplier in our sector. Our team of 3,500 employees supports more than 10,000 customers world-wide from a network of local offices and through our growing ecosystem of partners.

[#forthechallengers](#)

[ifs.com](https://ifs.com)

## WHERE WE ARE

### AMERICAS

+1 888 437 4968

### ASIA PACIFIC

+65 63 33 33 00

### EUROPE EAST

+48 22 577 45 00

### EUROPE CENTRAL

+49 9131 77 340

### UK & IRELAND

+44 1494 428 900

### FRANCE, BENELUX AND IBERICA

+33 3 89 50 72 72

### MIDDLE EAST AND AFRICA

+971 4390 0888

### NORDICS

+46 13 460 4000

COPYRIGHT © 2025 INDUSTRIAL AND FINANCIAL SYSTEMS, IFS AB. IFS AND ALL IFS PRODUCTS AND SERVICES NAMES ARE TRADEMARKS OF IFS. ALL RIGHTS RESERVED. THIS DOCUMENT MAY CONTAIN STATEMENTS OF POSSIBLE FUTURE FUNCTIONALITY FOR IFS'S PRODUCTS AND TECHNOLOGY. SUCH STATEMENTS ARE FOR INFORMATION PURPOSES ONLY AND SHOULD NOT BE INTERPRETED AS ANY COMMITMENT OR REPRESENTATION. THE NAMES OF ACTUAL COMPANIES AND PRODUCTS MENTIONED HEREIN MAY BE THE TRADEMARKS OF THEIR RESPECTIVE OWNERS.