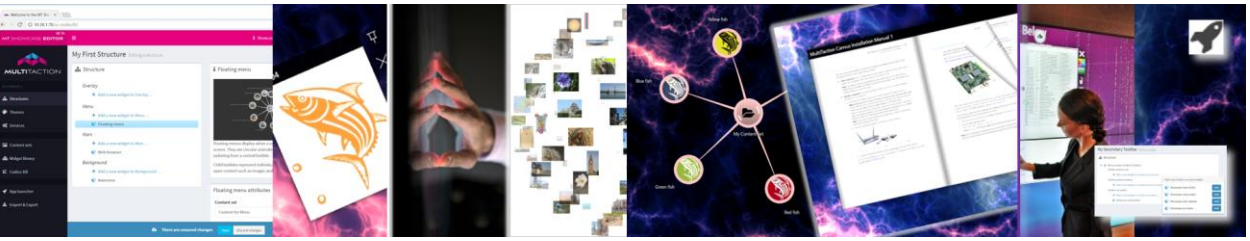
MULTITACTION



MT SHOWCASE 1.8 EDITOR MANUAL

Revision 4
April 2018

Contents

1 Introduction 9
1.1 About MT Showcase 9
1.2 About the MT Showcase Editor 10

2 Concepts 11
2.1 Apps 11
2.2 Widgets and attributes 12
2.2.1 Content widgets 12
2.2.2 Menu and hotspot widgets 13
2.2.3 Attributes 14
2.2.4 Widget library 15
2.3 Structures 16
2.3.1 Overlay 16
2.3.2 Menu layer 17
2.3.3 Main layer 17
2.3.4 Background layer 18
2.4 Themes 19
2.5 Service sets 20
2.6 Content sets 21
2.7 Media Library 22
2.8 Codice database 23
2.9 Welcome screen 24

3 Getting started 25
3.1 Using the touch screen 25
3.1.1 Hand and finger gestures 25
3.1.2 Infrared pens and erasers 25
3.1.3 Codice cards 26
3.2 Start the Editor 26
3.3 Create a simple app 29
3.4 Edit your app 32
3.5 Add a toolbar 33
3.5.1 Create a new toolbar 33
3.5.2 Test the toolbar 35
3.5.3 About pin 35

4 Create a structure 37

5 Create a finger menu 38
5.1 Set up a content set 39
5.2 Add a finger menu to your app's structure 41
5.3 Edit the content set 42
5.4 Configure the central bubble 42
5.5 Configure the child bubbles 44

5.6 Test the finger menu 46
5.7 Next steps 47
5.7.1 Explore the available bubble and connector attributes 47
5.7.2 Use a theme to streamline menu setup 47
5.7.3 Twitter menus 48

6 Create a theme 49

6.1 Why use a theme? 49
6.2 Theme FAQs 49
6.3 Create a new theme 50
6.3.1 Title bar example 51
6.3.2 Finger menu example 52
6.4 Add a theme to your app 55
6.5 Edit an existing theme 55
6.6 Set up theme defaults 56
6.6.1 Edit the theme defaults 56
6.6.2 How to override a theme default 57

7 Content widgets 58

7.1 Image Viewers 58
7.2 Video viewers 59
7.2.1 Video streams 60
7.2.2 Video preview 61
7.3 Web browsers 62
7.3.1 Web browser cache 63
7.4 PDF books and flows 64
7.5 Image movies 65

8 Create a rotating menu 66

8.1 Set up a content set 67
8.2 Set up rotating menus 71
8.3 Configure the rotating menu to use nodes 72
8.4 Add a rotating menu to your app's structure 74
8.5 Add a theme to your app 75
8.6 Test the rotating menu 75

9 Create a teaser 76

9.1 Set up a content set 77
9.2 Add a teaser to your app's structure 79
9.3 Set up the teaser 80
9.4 Test the teaser 82
9.5 Next steps 82
9.5.1 Customize the general teaser attributes 82
9.5.2 Customize the teaser menu behavior 83

10 Create a cloud 84

10.1 Cloud size 85

10.2 Set up a content set 85
10.3 Add a theme to your app 86
10.4 Add a cloud to your app's structure 86
10.5 Test the cloud 87
10.6 Twitter clouds 87

11 Create a content hotspot 88

11.1 Add a content hotspot directly to the Main layer 89
11.1.1 Add a content hotspot to your app 89
11.1.2 Set up the content hotspot 90
11.2 Add a hotspot image directly to the Main layer 92
11.2.1 Add an image to your app 92
11.2.2 Add hotspots to the image 93
11.2.3 Set up the hotspot image 93
11.3 Add hotspot images to a menu 94
11.3.1 Add images to your content set: Recommended method 94
11.3.2 Add images to your content set: Alternative method 95
11.3.3 Add hotspots to the image 96
11.3.4 Set up the hotspot image 96
11.3.5 Add a hotspot menu to your app 96
11.4 Add hotspot images to Codice content 97
11.4.1 Add an image to Codice content 97
11.4.2 Add hotspots to the image 98
11.4.3 Set up the hotspot image 98
11.5 Add hotspots to the image 99
11.6 Set up the hotspot image 101
11.7 Add multiple hotspots to an image 104
11.8 Add multiple content hotspots 105
11.9 Test the content hotspot 105

12 Create a service set 106

12.1 Email Sending service 106
12.1.1 What do emails sent from MT Showcase contain? 106
12.1.2 Set up the SMTP server 107
12.1.3 Set up an email template 108
12.1.4 Set the personal space collection mode 109
12.1.5 Define URLs for large items in media library 110
12.2 Data Gathering service 111
12.2.1 Enable data gathering 111
12.3 Twitter Connection service 113
12.4 Media Server service 114
12.4.1 Prepare the media server 114
12.4.2 Set up the Media Server service 114
12.5 Snippets service 116
12.6 Add a service set to your app 117

13 Codice cards 118
13.1 Identify Codice codes with the Input Visualizer 119
13.2 Add a Codice detector 120
13.3 Personal space 121
13.3.1 Register personal markers 121
13.3.2 Configure the Personal Space widget 122
13.3.3 Title and info text examples 123
13.3.4 Drag items into a personal space 124
13.3.5 Send email attachments (or URLs) 125
13.3.6 Save items to a personal web page 126
13.4 Codice content 127
13.5 Markers that ban tweets 130
13.5.1 Are tweets permanently removed? 130
13.6 Erasers 131
13.7 Create your own Codice cards 131

14 Toolbars 132
14.1 Adding a toolbar 132
14.2 Toolbar buttons 133
14.3 Primary and secondary toolbars 134

15 Annotations 135
15.1 Background annotation or widget annotation? 135
15.2 Enable annotations 136
15.2.1 Enable background annotations 136
15.2.2 Enable annotations on a widget 137
15.3 Remove annotations 138

16 Sounds 139
16.1 Guidelines 139
16.2 Add sounds to your media library 140
16.3 Add sounds to an individual widget 140
16.4 Loop sounds continuously 142
16.4.1 Loop sounds while a widget is open 142
16.4.2 Loop sounds while an app is running 144
16.5 Set default sounds 145
16.5.1 Set custom sounds for different types of widget 145
16.5.2 Set global sounds for all widgets 146
16.6 Turn off sounds for an individual widget 147

17 Add a Twitter feed 148
17.1 Setup overview 148
17.2 Create a Twitter app 149
17.3 Set up the Twitter connection service 149
17.3.1 Retrieve the consumer key and consumer secret 149
17.3.2 Add the Twitter Connection service to a service set 150
17.3.3 Add the consumer key and consumer secret 150

17.3.4 Authorize the app to use a Twitter account 151
17.3.5 Add the Twitter Connection service to your app 153
17.4 Add a Twitter Feed to your app 154
17.4.1 Create a Twitter feed content set 154
17.4.2 Set up a Twitter cloud 156
17.4.3 Set up a Twitter finger menu 157
17.5 Filter the Twitter feed 158
17.5.1 Add a search filter to the Twitter feed content set 158
17.5.2 Search filters 158
17.5.3 Remove tweets with a Codice card 159

18 Table mode 160
18.1 Enable table mode 160
18.2 Enable Codice support for table mode 161

19 Backgrounds and animation effects 162
19.1 Add an animated background 163
19.2 Add animation effects 163
19.3 Customize colors for Awesome background 164

20 Screensaver 165
20.1.1 Add a screensaver to your app 165
20.1.2 Set up content for your screensaver 167
20.2 Lock screen 168
20.2.1 Add locking behaviour to a screensaver 168
20.2.2 Add a Screensaver trigger widget to your app 169
20.2.3 Test lock screen 170

21 Widget customization 171
21.1 Drop shadows 171
21.2 Size limits 173
21.2.1 Maximum widget size 173
21.2.2 Minimum widget size 173
21.2.3 Set the maximum and minimum sizes 174
21.3 Snap to angle 175
21.4 Colors 177

22 Schedules for apps and widgets 178
22.1 Create an app event 178
22.2 Create a widget event 180
22.3 Assign a widget event 180
22.4 Manage app and widget events 181

23 Import & export 182
23.1 Zip files 182
23.2 Export data from MT Showcase 182
23.3 Import data into MT Showcase 184

24 Keyboard 185
24.1 Keyboard focus 186
24.2 Snippets 186
24.2.1 Add snippets to your app 186
24.2.2 Insert snippets to text fields 188

25 Other features 189
25.1 Advanced content sets 189
25.1.1 Folders and subfolders 189
25.1.2 Add a URL 190
25.1.3 Add a Twitter feed 191
25.1.4 Add a widget 191
25.1.5 Widget example: Add an Input Visualizer to a content set 191
25.2 Animated content 192
25.2.1 Add animated content to your app 192
25.2.2 Set up the animated content 194
25.3 App switcher widget 195
25.3.1 Create an App switcher content hotspot 195
25.3.2 Add the App switcher widget to a finger menu 196
25.3.3 Add the App switcher widget to Codice content 198
25.3.4 Create a collection of linked apps 198
25.3.5 Set up the app switcher 199
25.4 Exit Showcase widget 200
25.4.1 Create an Exit Showcase content hotspot 200
25.4.2 Add the Exit Showcase widget to a finger menu 201
25.5 Maximization areas 204
25.5.1 How to maximize and unmaximize a widget 204
25.5.2 Add a maximization area 205
25.6 Opening animations 206
25.6.1 Finger menu animations 206
25.6.2 Content hotspots 208
25.6.3 Codice content 208
25.7 Password protection 210
25.8 Welcome screen with app list 211
25.8.1 Provide a method to access the welcome screen 212
25.8.2 Remove an app from the welcome screen 212
25.9 Closing content 213

1 Introduction

1.1 About MT Showcase

MT Showcase is the perfect solution for presenting rich interactive media content. It can present a wide range of media content in an innovative and intuitive way. It also allows designers to create custom interactive apps that can run on MultiTaction Cells, making use of all their advanced features.

For example, MT Showcase allows your users to:

- View and zoom high-resolution photos, graphics, and charts
- View and zoom vector graphics, such as maps
- View and zoom videos, even with 4K resolution
- Browse PDF documents such as brochures
- Annotate documents and images with infrared pens
- Use Codice markers (2D barcodes) to send screen content to an email account or to display specific content on the screen

A range of ready-to-use content for demonstration purposes is available to registered users on the MT Showcase downloads page. For example, MultiTaction resellers can use MT Showcase to demonstrate the multi-touch high-resolution display capabilities of MultiTaction Cells.

MT Showcase on an interactive video wall

But the real power of MT Showcase is provided through the Editor. The Editor is a web-based tool that lets you create MT Showcase apps. You can run these apps on a video wall to give your users a compelling interactive experience. See section 1.2 for details about the Editor.

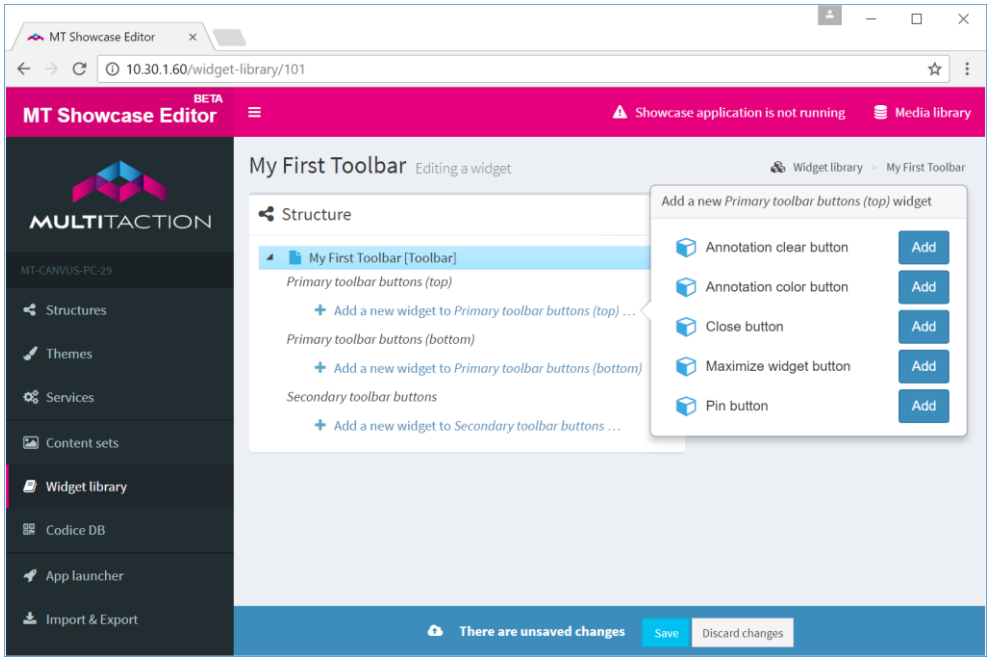
1.2 About the MT Showcase Editor

The MT Showcase Editor is a web-based tool for creating custom MT Showcase apps. No coding is required. Anyone can use the Editor to create a compelling interactive experience for their users.

The Editor allows you to choose the content for your app (including images, videos, web sites, PDFs and a background). You can specify how screen items behave. For example, you can control how finger menus look and behave.

Finally, you also use the Editor to perform various administrative tasks in MT Showcase. For example, you can export and import apps; you can manage the media library; and you can set up services. For example, the *Email Sending* service allows users to send screen items to an email address while the *Twitter Connection* service allows you to add a Twitter feed to your app.

And best of all, you can edit an app on the fly, while it is running on your video wall. As soon as you click Save, your app is updated. No need to close the app, no waiting while it recompiles. You can see the effects of your changes immediately!



MT Showcase Editor. Example 'Editing a widget' page. This example shows how to add buttons to a new toolbar that can be used by any widgets in your app.

2 Concepts

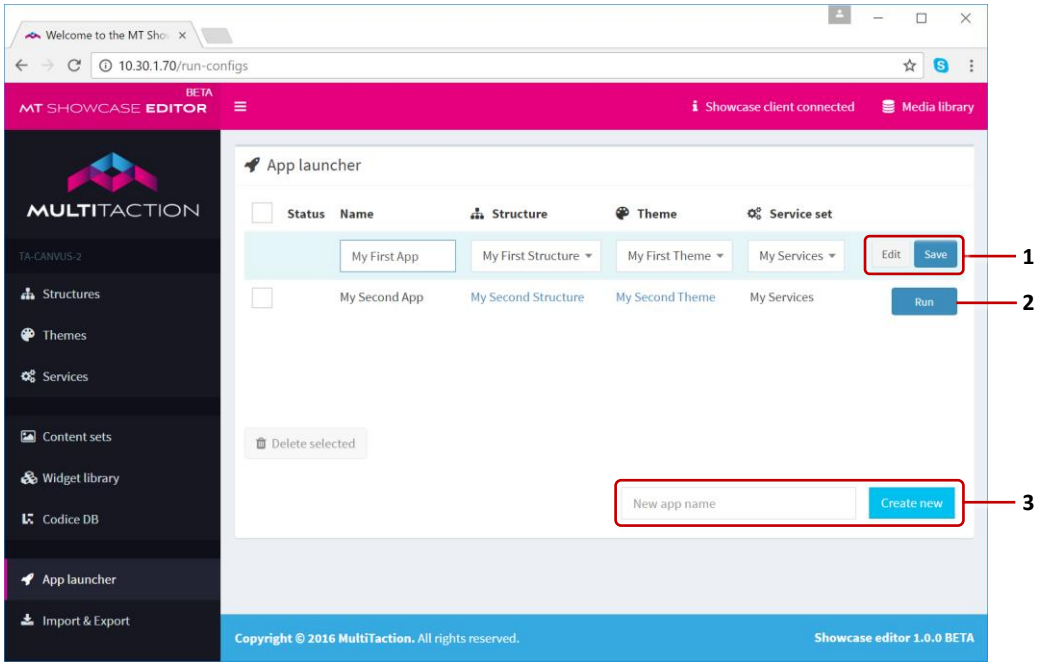
2.1 Apps

An *app* is an individual MT Showcase application. An app defines the actual content available to users on your video wall (images, videos, PDFs, and so on), plus the appearance and behavior of screen items such as finger menus, content hotspots, and the screen background.

To create an app, you must define a *structure*. Typically, you also assign a *theme* and *service set* to your app, although these are not mandatory.

- The *structure* defines the combination of *widgets* used in each layer of the app; see section 2.3.
- A *theme* is a collection of attribute values for MT Showcase widgets. If your app uses a theme, widgets inherit their attribute values from the theme; see section 2.4.
- A *service set* defines a specific administrative setup for an MT Showcase app. See section 2.5.

A single MT Showcase installation can support multiple apps. You run apps and create new apps in the *App launcher* screen. You can also edit and save apps in this screen, assigning a name, structure, theme, and service set.



App launcher screen. 1 *Edit and Save buttons.* 2 *Run button.* 3*. 'New app name' input box and 'Create new app' button.*

2.2 Widgets and attributes

A widget is simply a component in an MT Showcase app. A widget's *attributes*, such as its size, behavior and appearance, can be defined in the Showcase Editor.

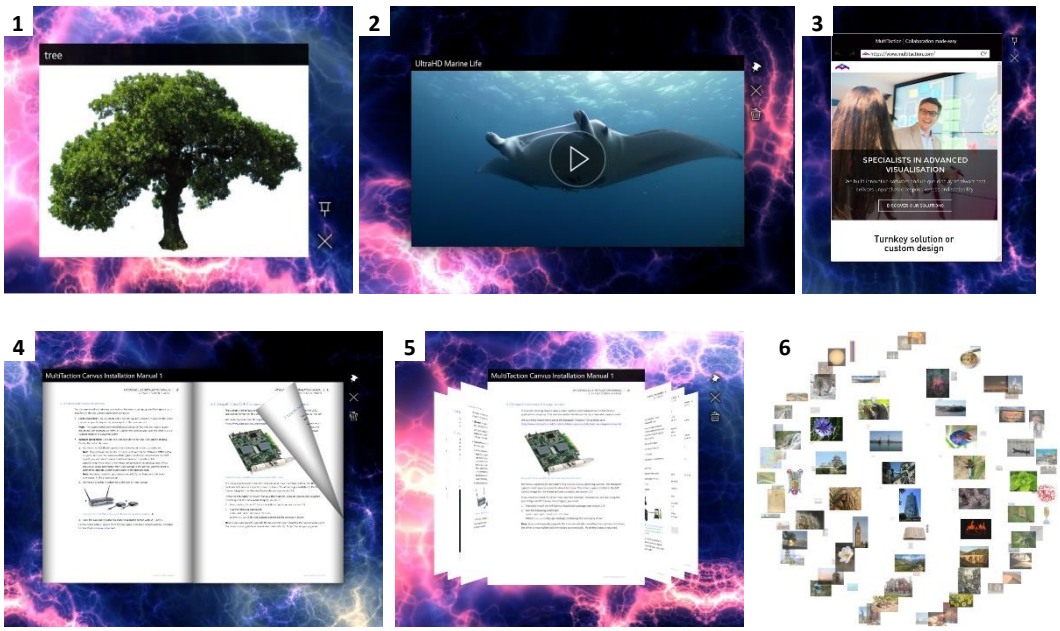
2.2.1 Content widgets

There are several types of widget in MT Showcase. But as an MT Showcase app designer, you will mainly focus on widgets that display content directly (such as the image viewer) and widgets that open or launch content (such as menus and hotspots).

The following widgets display content directly. They contain the text, images and video content that you want to show on your video wall.

- **Image viewer:** Displays image files, including PNG, JPEG, BMP and TIF files.
- **Image movie:** Plays a sequence of images as an animation or movie.
- **Video viewer:** Plays a video. The video source can be a video file (eg, an MP4 movie), a video stream, or a video capture device (eg, a web cam).
- **Web browser:** Displays web pages and HTML files, but can also display images, videos, text files and PDFs.
- **PDF Book:** Displays PDF documents as open books. It uses a realistic page-flip animation when paging through a document.
- **PDF Flow:** Displays PDF documents as a stack of pages. It uses a carousel-like animation when paging through a document.
- **Cloud:** Displays a rotating collection of items (images, movies, browsers, and so on), clustered together in a ball. Users rotate the cloud to find the item they want.

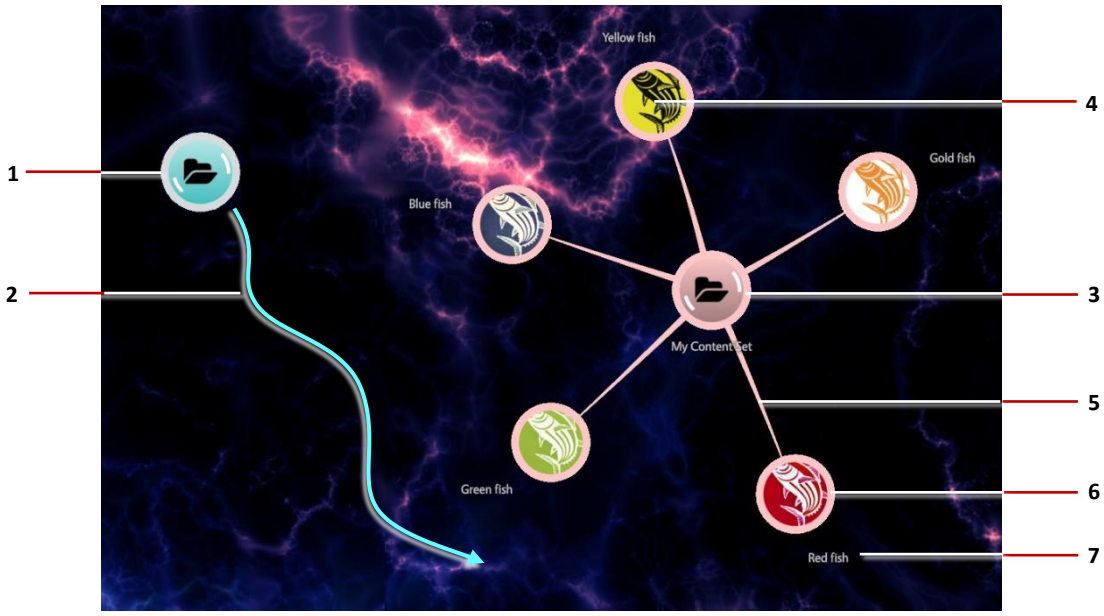
See section 7 for more details about content widgets.



Example content widgets. 1 *Image viewer.* 2 *Video viewer.* 3 *Web browser.* 4 *PDF Book.* 5 *PDF Flow.* 6 *Cloud widget.*

2.2.2 Menu and hotspot widgets

These widgets display menus or activate screen hotspots.



Example finger menu and teaser.

Teaser: 1 *Teaser bubble. Tap the bubble to open a finger menu.* 2 *Teaser movement.*

Finger menu: 3 *Central bubble.* 4 *Child bubble. In this example, the bubbles display thumbnails (or 'preview images') of the menu items. Tap any child bubble to display the menu item.* 5 *Bubble connector.* 6 *Bubble background color.* 7 *Bubble visible name.*

- **Finger menus:** These display when a user taps and holds an empty area of the screen. They can also be launched from content hotspots or with a Codice card. A finger menu is a circular animated menu, with child bubbles radiating from a central bubble. Child bubbles represent individual menu items and can display thumbnail images.

  Separate widgets control the appearance and behavior of the individual bubbles and the connectors that radiate out from the central bubble to the child bubbles. You can also replace bubbles with *nodes* (either cubes or custom images).

- **Rotating menus:** These display when a user taps and holds an empty area of the screen. They can also be launched from content hotspots or with a Codice card. A rotating menu is an oval animated menu, with items orbiting a central node. The effect is similar to satellites orbiting a planet (see the screenshot on page 66). Users can tap an orbiting item to open a submenu of items such as images, videos and PDFs. Rotating menus are a direct alternative to finger menus.

  A separate widget controls the appearance of the menu items. By default, the central node, orbiting items and submenu items are all shown as bubbles, like the ones in finger menus, but you can display them as *nodes* (cubes or custom images).

- **Teasers:** These are menu launchers. They can be a single bubble drifting slowly across the screen, 'wobbling' briefly at regular intervals to attract users' attention, or a single bubble that fades in and out, reappearing at random screen locations. In both cases, users can tap the teaser to open a regular finger menu or rotating menu. For greater

impact, you can specify multiple instances of a teaser drifting across the screen at the same time, and you can use a custom image or video instead of a bubble.

(A separate widget controls the behavior of the teaser menu, including its speed, direction and animation properties.)

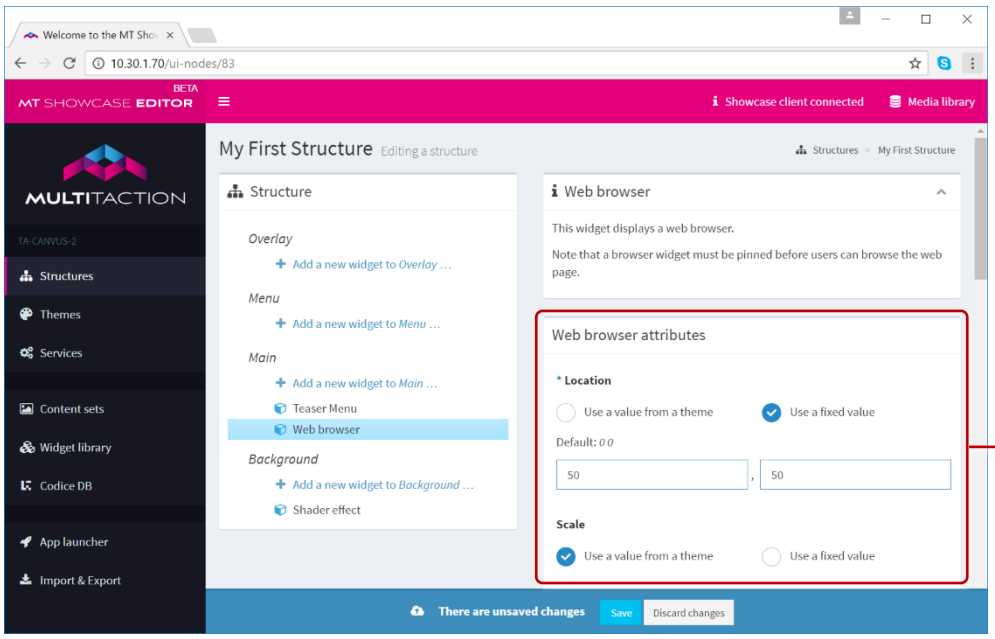
- **Content hotspots:** These are special areas of the screen that support custom interactions. When users tap a hotspot, menus or other widgets such images or videos are displayed. The app designer specifies the number of hotspots, their size and location. They can also assign images or videos as hotspot background.

Other types of widget include games (Pong and Missile Command) and Codice detectors. (A Codice is a 2D barcode or *marker* printed on paper or card.)

Most widgets mentioned above are interactive. Users can move, resize, rotate, pin and close these widgets, if permitted. As the app designer, you determine what actions are permitted for each widget in the app. For example, you may include an image with a fixed size and position in your app, but superimpose a hotspot above this image so that, when a user taps the image, MT Showcase opens a menu of secondary items (such as images, videos, or PDFs) that users can move and resize as they like.

2.2.3 Attributes

A widget has *attributes* that define its size, location, appearance or behavior. Some widgets have many attributes. Others may only have one or two attributes. And some widgets have none.



Editing a structure screen. 1 *Example widget attributes pane.*

Most widget attributes either have numeric values (such as height and width, screen coordinates, or snap to angle increments) or an Enabled check box (the attribute can be enabled or not enabled).

But some attributes take different values. For some, you must specify the path and filename of an *asset* in the media library (such as a Book widget's PDF file). For others, you choose from a drop-down list of values (for example, the **toolbar** attribute lets you choose from a list of existing toolbars or create a new one).

2.2.4 Widget library

The widget library is a repository of *shared widgets* that you can re-use in any MT Showcase apps on your MT Showcase server.

What is a shared widget? These are specific types of widgets that you have created and named. As soon as you create a shared widget, it appears in the widget library. You can quickly edit or rename any shared widgets in the widget library.

The most common types of shared widget include:

- **Menu bubbles:** Items in finger menus are shown as bubbles, radiating from a central bubble. This widget sets the bubble size, color, image and label.

  Before you can display a finger menu in your MT Showcase app, you must create at least one named bubble widget (such as 'Red bubbles').

  Note: *By default, finger menus render menu items as circular bubbles. But if you prefer, you can use any image you want. For example, you can set up a finger menu to use your corporate logo instead of bubbles.*

- **Bubble connectors:** In a finger menu, connectors attach child bubbles to the central bubble. This widget sets the connector color and length and how much they bend when a user drags the menu.

  Before you can display a finger menu in your MT Showcase app, you must create at least one named bubble connector widget (such as 'Short red connector').

- **Toolbars**: Before you can assign a toolbar to, say, an image viewer, you must create and name a toolbar widget.

  For example, you may want to create a toolbar named 'Pin and Close Toolbar'. You can then assign this toolbar to any image viewer in your app. In fact, you can assign this toolbar to any content widget in your app (for example, video viewers, PDF viewers and browsers).

- **Teaser operator:** Before you can add a teaser to your MT Showcase app, you must create a *teaser operator* widget. This widget controls how a teaser behaves. Two behaviors are supported:
  - **Floating content behavior:** The teaser floats slowly across the screen, animating briefly at regular intervals.
  - **Fading content behavior:** The teaser pulses rhythmically, fading in and out at random screen locations.

- **Opening animations:** Many widgets can play an animation when they open. For example, finger menus can open with a circular progress bar or a spinning spiral of small starts. You can create and name widgets to control the properties of these animations, such as their color and speed (such as 'White star spiral').

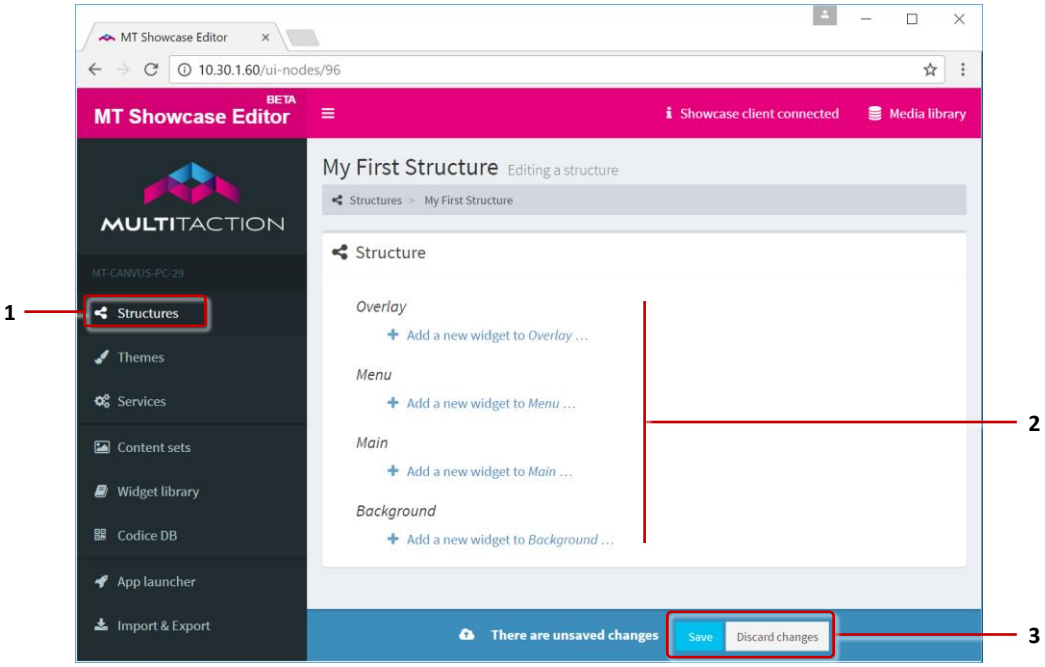
- **Close after idle:** If a user stops interacting with a widget (for example, a menu item or an item opened from a hotspot), the widget fades and closes automatically. 'Close after idle' specifies the timeout and fade duration for idle widgets.

  You can create separate versions of this widget. For example, you may want two versions of this widget ('30 second timeout' and '60 second timeout') to enable you to assign different timeouts to, say, videos and PDFs.

- **Snap to angle:** You can constrain the incremental rotation of widgets so they automatically re-adjust themselves to a fixed angle of rotation. The effect is similar to a boat righting itself after keeling over. The snap to angle widget sets the incremental rotation for other widgets.

2.3 Structures

A *structure* defines the framework of an MT Showcase app. Specifically, it defines the combination of widgets—their types, size and location on screen—used in each layer of the app. Instructions for creating a structure are in section 4.



*MT Showcase Editor, 'Editing a structure' screen. **1** Structures menu. **2** Overlay, Menu, Main and Background layers. **3** Save button and Discard Changes button.*

2.3.1 Overlay

This is the topmost layer in an MT Showcase app. Widgets in this layer operate above any screen content in the main layer. There are six overlay widgets:

- **Animated content:** The animated content widget applies animation to image and video widgets. There are two types of animation. You can configure images or videos to float slowly across the screen. Or you can configure them to pulse rhythmically, fading in and out at random screen locations. For details, see section 25.2.

Note: *Animated content is also available in the background layer (see section 2.3.4). If added to the overlay layer, content appears* above *items in the main layer. If added to the background layer, content appears* underneath *items in the main layer.*

- **Champagne, Particles** and **Ripple**: These widgets are purely cosmetic and superimpose animation effects above the main app layer (streams of rising bubbles, floating circles and wavelets emanating outwards from a finger, respectively).

  Note: *These effects are also available in the background layer (see section 2.3.4). When added to the overlay layer, the effects appear to be* above *items in the main layer. When are added to the background layer, they appear to be* underneath *items in the main layer.*

- **Input Visualizer:** This widget is provided for administrative purposes. It shows visualizations of touch events for fingers, hands, Codice cards, and infrared pens. It also provides a quick mechanism for identifying Codice code numbers. For an example use of the Input Visualizer, see section 13.1.

- **Maximization area:** This widget defines an area (or *areas*) on the screen where maximized widgets are displayed. When a widget is maximized (for example, when a user taps the Maximize toolbar button), it floats to the nearest maximization area and scales up to fill the area. For details, see section 25.5.

- **Screensaver:** A screensaver widget can display a background such as Awesome or Shader effect, an image or movie, or animated content. The screensaver is displayed when the app has been idle for a specified time. To clear the screensaver and return to the app, the user taps the screen. For details, see section 20.

2.3.2 Menu layer

This layer is where you define a finger menu or rotating menu; see section 2.2.2. To open a finger menu or rotating menu, users tap and hold any area of empty screen.

Instructions for creating a finger menu are in section 5. Instructions for creating a rotating menu are in section 8.

Note: *These menus can also be launched from content hotspots (defined in the Main layer) or by a user holding a Codice card against the screen.*

2.3.3 Main layer

This layer includes the content widgets that users see when they use the app. It can include any combination of image, video, browser, PDF and cloud widgets.

This layer can also contain content hotspots, teasers, games (Pong and Missile Command) and Codice detectors.

See section 2.2.1 for an overview of the main widgets.

2.3.4 Background layer

This layer defines the screen background. It can include:

- **Animated content.** This widget applies animation to image and video widgets. You can configure images or videos to float slowly across the screen. Or you can configure them to pulse rhythmically, fading in and out at random screen locations. For details, see section 25.2.

  Note: *Animated content is also available in the overlay layer (see section 2.3.1).*

- **An animated background.** The following backgrounds are available:
  - **Awesome:** A background of drifting clouds, similar in effect to the Aurora Borealis. You can customize the colors and animation speed.
  - **Shader Effect:** A background that resembles the night sky, with dramatic purple and magenta nebulae drifting across the screen, punctuated with cores of bright white light. However, you can use a custom shader file if required.

- **Animation effects** superimposed on the screen background. These widgets are interactive and respond to touches, emanating outwards from the user's fingers.
  - **Champagne:** Displays streams of rising bubbles, like in a glass of sparkling wine.
  - **Particles:** Displays small floating circles, randomly connecting to nearby circles.
  - **Ripple:** Displays wavelets, like ripples on a pond.
  - **Sparkles:** Displays swirls of particles. Its effect is like dragging your hand through a sparkling fluid.

  Notes:
  - *For more about animated backgrounds and animation effects, see section 19.*
  - *These effects (not Sparkles) are also available in the overlay layer; see section 2.3.1.*

- **Images and movies.** These widgets occupy fixed positions on the screen and are not interactive. For example, you may want a video looping continuously in the background or you may want to use a background image as wallpaper.

  However, you can position hotspots in the Main layer directly above areas in your background image to make the image appear to be interactive; see section 11.

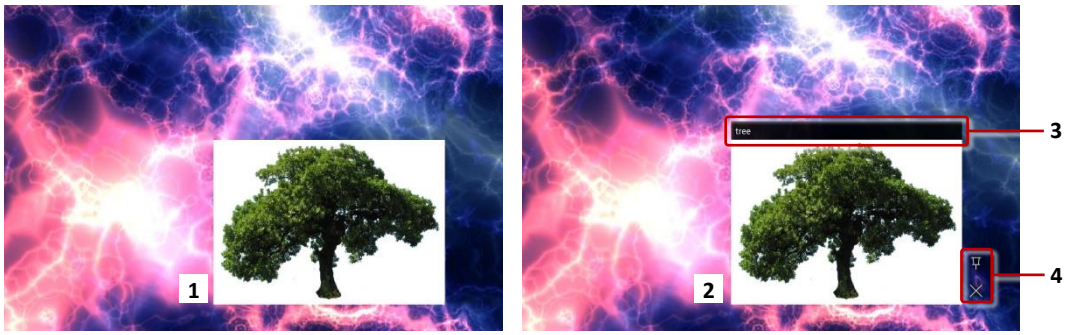
- **An annotation widget.** This defines the color and timeout for annotations drawn on the screen with an infrared pen.

2.4 Themes

A *theme* is a collection of attribute values for MT Showcase widgets. If your app uses a theme, all widgets in the app will inherit their attribute values from this theme by default. Instructions for creating a theme are in section 6.

Why use a theme? Themes provide consistent widget appearance and behavior across an app. They also simplify the process of creating apps, because the same theme can be used by one or more apps.

Consider an *image viewer* widget. By default, an image viewer has no toolbar or title bar. But if you want the images in your app to have title bars and toolbars, you can create a theme that includes an image viewer widget with these features enabled. If you then assign this theme to your app, all image viewer widgets in your app will automatically inherit a toolbar and title bar. This approach is far quicker than individually configuring each image viewer.



*Example inheritance from theme. **1** Before a theme is applied to the app, this Image Viewer widget has no title bar or toolbar. **2** After a theme is applied, the widget inherits the title bar (**3**) and toolbar (**4**) defined in the theme.*

Of course, there may be occasions when you do not want a widget to use a theme. (For example, you may want to suppress title bars and toolbars for videos playing in the background layer.) If so, you can always override the theme and apply custom attribute values to any widget in your app. To override the theme, you edit widgets directly in the app's *structure* (see section 2.3) or *content set* (see section 2.6).

Note: *If an app does not use a theme, or no custom attributes are defined in the app structure, widgets use hard-coded default values.*

2.5 Service sets

A *service set* defines a specific administrative setup for an MT Showcase app. Each MT Showcase installation can support multiple service sets. This allows MT Showcase designers to create multiple versions of an app, all with the same content and theme but each with a unique service set. For example, you may want to deploy the same app in your London and Paris offices but with a different service set in each location that specifies the local SMTP server.

In the current version of MT Showcase, service sets can include the following services. Later versions of MT Showcase may include additional services.

- The *Email Sending* service is used for sending screen content from MT Showcase to a specified email account. When you add the email sending service to a service set, you must define such attributes as the SMTP host, credentials for an SMTP user, the sender's email address and the email subject.

  For details, see section 12.1.

  Note: *An alternative method for saving screen content, and one that does not require email addresses, is provided by the Media Server service; see below.*

- The *Data Gathering* service collects content usage data that can be imported into third party data visualization tools such as Tableau Desktop. Usage data is stored in a PostgreSQL database as event records.

  Example events include hand and finger touches, opening or closing a widget, playing a video, viewing a PDF, browsing to a URL, adding items to a personal space, and emailing items from a personal space.

  For details, see section 12.2.

- The *Twitter Connection* service allows you to add Twitter feeds to your apps, with tweets displayed either in a cloud widget or finger menu. When you add a Twitter feed to a content set in your app, you can specify search terms to filter the tweets.

  For details, see section 12.3.

- The *Media Server* service is used for saving screen content from MT Showcase to a personal web page. This service is an alternative to the *Email Sending* service and has the advantage that it does not require a user's email address.

  From the end-user's viewpoint, they can drag screen content into their personal space in MT Showcase and then use a QR code reader on their mobile device to download this content.

  When you add the Media server service to a service set, you must specify which web server you want to use. You must also specify the *personal space collection mode*. For details, see section 12.4.

2.6 Content sets

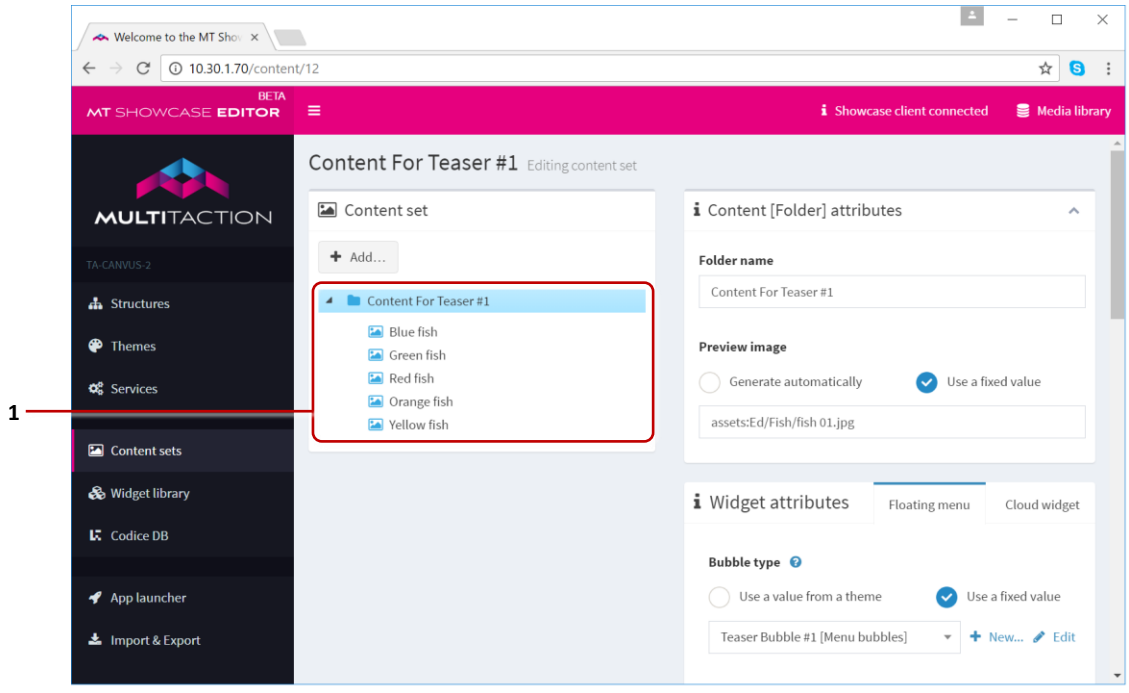
A *content set* is collection of items (images, videos and PDFs). Content sets can also include special items such as the Exit Showcase widget and the Twitter Feed widget. Content sets supply the items that are displayed by finger menus, teasers and clouds.

To create a content set, you simply drag items from the MT Showcase media library. Detailed instructions for setting up a content set are in section 5.1.

Note: *You cannot add items directly from, say, your laptop to a content set. You must first drag the items into the MT Showcase media library.*

You can define multiple content sets for use by different widgets. For example, your app may include multiple teasers (see page 13), each of which launches a unique collection of images. You can also use the same content set across multiple apps.

Items in a content set can be organized in a flat structure, or you can organize them hierarchically to reflect the menus and submenus that you want in your app. If your MT Showcase media library already contains items organized into the correct folders and subfolders, simply drag the folder directly into your content set to automatically create a corresponding menu structure.



MT Showcase Editor, Content Set screen. 1 *Items in content set. Also referred to as 'assets'.*

2.7 Media Library

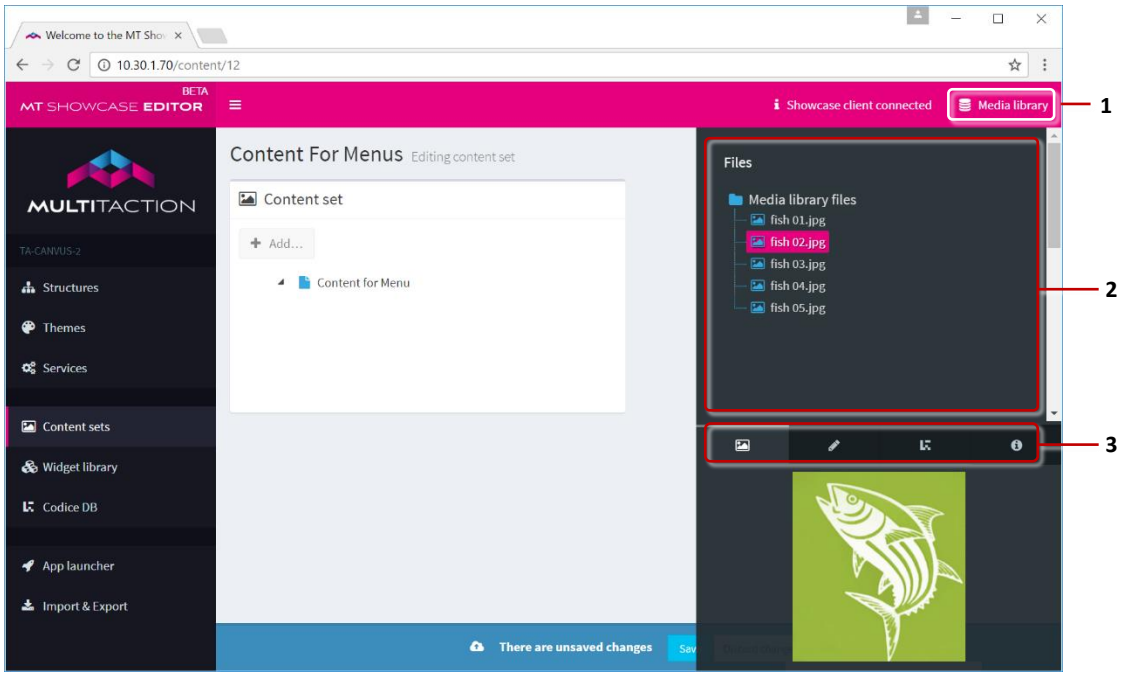
The media library is a collection of images, videos, and PDFs stored on your MT Showcase server. If you want to include these items in your MT Showcase app, you must first add them to your media library. Crucially, the media library is the source for items in content sets (see section 2.5).

To add items to the media library, you simply drag them from Windows Explorer (or from equivalent applications on Apple or Linux computers). You can then organize items in the media library into folders and subfolders as required.

When you click an item in the media library, tabs at the bottom of the screen provide information about the item:

- **Preview:** Shows a preview of the image, video or PDF
- **Description:** Provides an input box for you to add a description of the item and copyright details. *(This description is for your reference only.)*
- **Codice URL:** Provides an input box for you to add a URL for the item. For example, if a video in the media library is also available on your corporate website or YouTube, you can include its URL here.

  The purpose of this feature to avoid sending very large videos as email attachments. If a user sends this item from their personal space in MT Showcase, this URL is included in the email instead of the actual .mp4 file. See section 12.1.5.
- **Details:** Shows the file size, date last modified and MIME type.



MT Showcase Editor. 1 *Media library button. Click to show or hide the media library.*
2 *Media library.* 3 *Tabs showing information about currently selected item.*

2.8 Codice database

You register and configure Codices in the Codice DB screen.

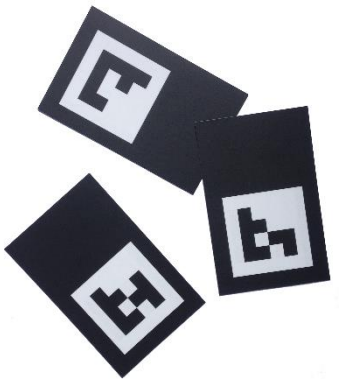
Codices are square 2D barcodes, or *markers*, each with a unique code. A Codice card is simply a Codice printed on paper or card. You can assign a Codice card to an individual user (a 'personal Codice'). You can also configure Codice cards to display specific content, ban proscribed tweets in a Twitter feed, or act as an eraser; you can make these cards available to all users (a 'utility Codice').

For example, a user can use their own Codice card to access their personal space. This Codice card is their *personal marker*. The user simply presents their card, ie holds it against the screen to open their personal space. They can then drag items into the folder from the MT Showcase app and send them to their registered email address, providing a simple method for exporting screen content out of MT Showcase.

Codice cards can also be used to open *Codice content.* This refers to the item that opens automatically when a user presents a specific Codice card. This item could be an image, video or even a browser. This content is available to anyone who presents the correct card (ie, any card with the correct Codice code).

You can also designate a Codice as an *eraser* or *tweet blocker*. Any user can use an eraser card to erase pen or finger annotations from an MT Showcase app. Likewise, any user can use a tweet blocker to permanently remove specific tweets from a Twitter cloud or menu.

For details about Codice detectors, personal markers, Codice content, and erasers, see section 13.



Example Codice cards

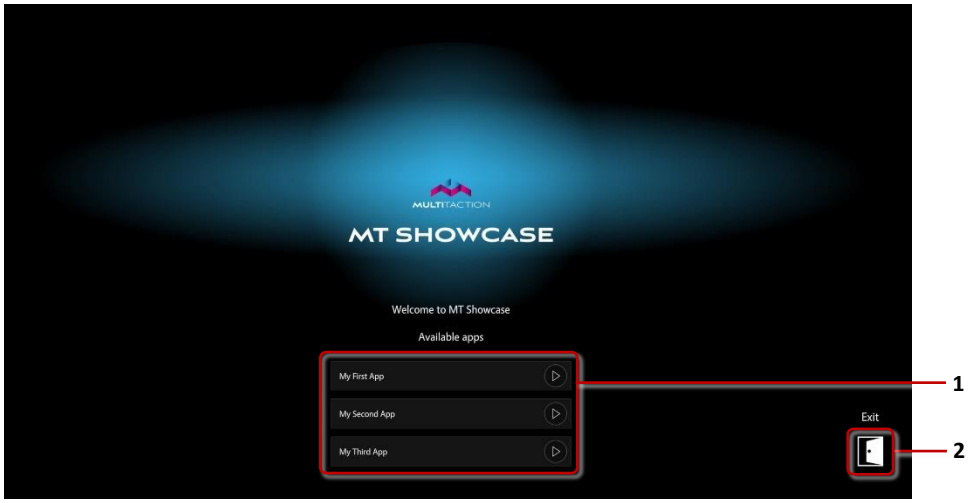
2.9 Welcome screen

Usually when MT Showcase starts, it automatically displays an *app*. For example, administrators can set up an MT Launcher tile that automatically launches a specific app.

However, there may be occasions when users are directed to the MT Showcase *welcome screen*. This may be intentional. For example, you can add app switcher widget (section 25.3.5) to return users to the welcome screen. Or it may be unintentional. For example, if a user taps an MT Launcher tile to open an MT Showcase app that no longer exists, the user is taken instead to the welcome screen.

In either case, you can customize the welcome screen to include

- A list of available apps. You also need to provide users with a method of getting to the welcome screen. See section 25.8.
- An Exit button, allowing users to quit back to MT Launcher. For instructions on how to display the Exit button, see the *MT Launcher Installation Manual*. Registered users can download this manual from https://cornerstone.multitouch.fi/mt-launcher.



Welcome screen. **1** *Available apps.* **2** *Exit button.*

3 Getting started

3.1 Using the touch screen

MT Showcase users can interact with the touch screen by using hand and finger gestures, an infrared pen, or Codice cards.

3.1.1 Hand and finger gestures

The hand and finger gestures recognized by MT Showcase are generally the same as the gestures for using smart phones or tablets. For example, use the familiar pinch and spread gestures to zoom in or out:

You can also use your finger to drag or swipe items on screen. Tap the screen with your finger to 'press' buttons and open or close menus.

To open a finger menu or rotating menu, tap and hold any empty area of the screen.

3.1.2 Infrared pens and erasers

Infrared pen

The pens supplied with MT Showcase have a touch-activated infrared LED in their tip. Use the pens to annotate screen items such as images, or to draw directly onto the screen.

To set the default color of the pen stroke for annotations:

- Drawn directly on the screen, edit the Annotation widget; see section 15.2.1.
- Drawn on a widget, use the widget's Color button; see section 15.2.2.

Use infrared pens to draw sketches and captions on a canvas

To delete annotations drawn with an infrared pen, use the eraser card supplied with MT Showcase. Place the card face down on the screen and rub out the annotation. (The eraser card is a special Codice card; see section 3.1.3.)

Eraser card

3.1.3 Codice cards

A Codice is a simply 2D barcode (or *marker*). A Codice card is simply a Codice printed on paper or card. You can issue users with a personal marker or Codice card, or you can make general Codice cards available to your users. You designate the actions associated with each Codice.

For example, users can present their personal Codice card (ie, hold it against the screen) to open a folder containing their personal items. Or they can present a general Codice card to launch specific content or to erase pen or finger annotations from an MT Showcase app.

For details about using Codice cards, see section 13.

3.2 Start the Editor

Terminology: *For simplicity, the instructions below refer to 'your laptop' when describing the device you use to run the Editor. Although this device is generally a laptop, you can run the Editor on any compatible device including desktop computers and tablets.*

The MT Showcase Editor is a web-based application, so you can browse to the Editor from your laptop.

Follow these steps:

1. *(Mandatory)* Verify that MT Showcase is installed on the application computer that drives your video wall. If in doubt, ask your MT Showcase administrator. (Alternatively, you can install MT Showcase on a laptop or desktop computer for app development and testing purposes.)

   Note: *Deployment instructions, plus recommended computer specifications, are in the MT Showcase Installation Manual. Registered users can download this manual from* https://cornerstone.multitouch.fi/showcase

2. *(Mandatory)* Verify that the MT Showcase server is running.

   If you need assistance, ask your MT Showcase administrator. Instructions for starting and stopping the server are in the *MT Showcase Installation Manual.*

   Note: *The MT Showcase server supplies the client with the data it needs to display MT Showcase apps. It also includes a web server that enables app designers to access the MT Showcase Editor.*

3. *(Recommended)* Verify that the MT Showcase client is running. The client displays MT Showcase on your video wall, allowing you to view changes to your app as you design it in the Editor.

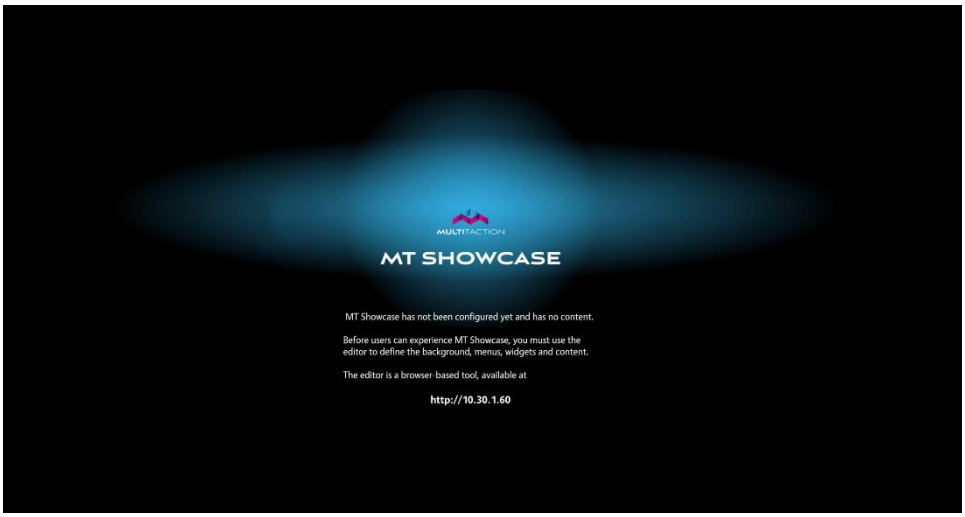
   Instructions for starting and stopping the client are in the *MT Showcase Installation Manual.* Briefly, on Ubuntu systems you launch the client from a desktop menu or by running a launch command in a terminal emulator. On Windows systems, you launch the client by double-clicking the MT Showcase desktop shortcut:

   Note: *The MT Showcase client displays content and handles touch events for MT Showcase apps, based on data received from the server.*

4. *(Applies only if the MT Showcase client is running; see step 3)*

   If no app has been set up on the MT Showcase server, MT Showcase displays the setup advisory screen and directs you to the URL for the MT Showcase Editor:

   MT Showcase setup advisory screen. This screen displays when no app is running.

5. *(Applies only if the MT Showcase client is running; see step 3)*

   We recommend that you position yourself with your laptop where you can see the video wall.

   Although you do not *need* to see the video wall while you are editing an MT Showcase app, for obvious reasons it is better if you can see it to assess how your edits are affecting the application's layout and content.

6. Start the MT Showcase Editor.

   To do this, browse to the IP address of the application computer.

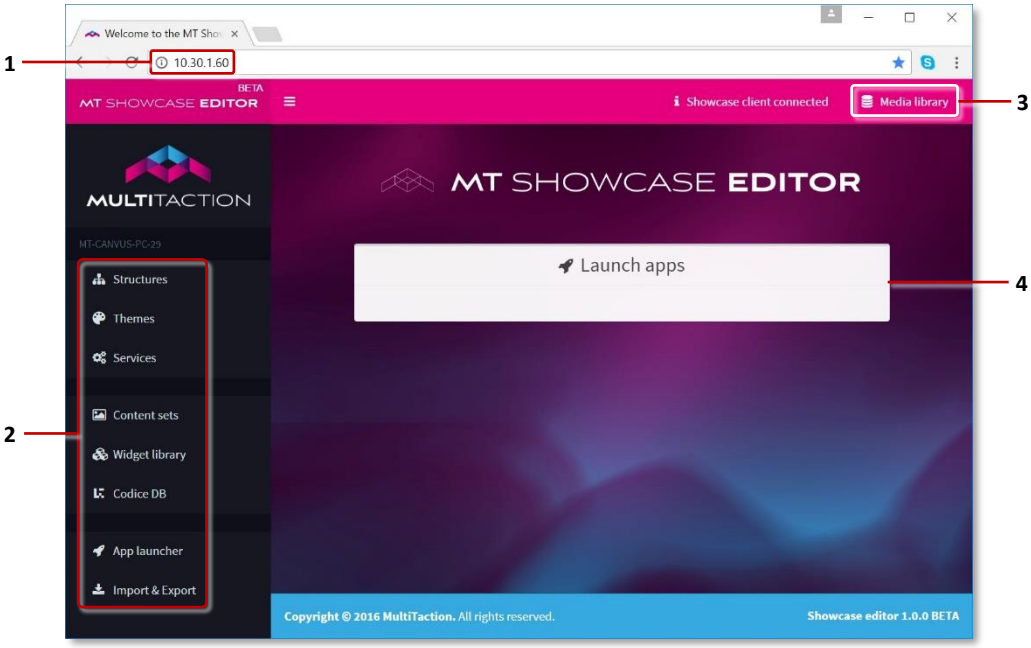
   By default, the Editor listens on port 80 so you do not need to specify a port number. However, if your network administrator specified a non-default port number when installing MT Showcase, you must add a port suffix to the IP address.

   For example, if the application computer IP address is 10.30.1.75 and the Editor is configured to listen on port 100, browse to:

   10.30.1.75:100

7. When the MT Showcase Editor starts up, it displays the start screen.

   You can now start creating your first app.



MT Showcase Editor start screen. **1** *Editor URL.* **2** *Editor menu.* **3** *Media library button.* **4** *App list. This list is empty until you create your first app.*

3.3 Create a simple app

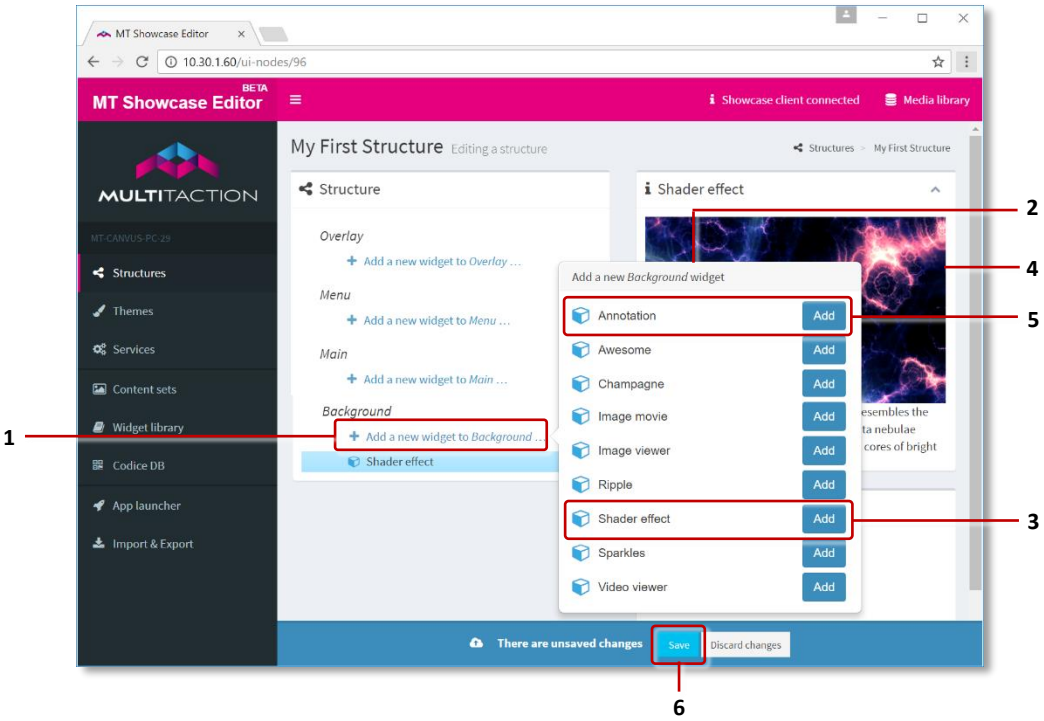
In this section, you will create a simple app that contains a browser and animated background.

Follow these steps:

1. Create a new structure
   a. Click Structures 

in the left-hand menu.
   b. Enter a name in the *New structure name* field. For example, My First Structure.
   c. Click the Create New button.
   The *Editing a Structure* screen appears.

2. Add an animated background to the structure
   a. Go to the Background section and click *Add a new widget to background*.
   b. From the pop-up Background widget menu, choose an *animated background*.
   This menu includes two animated backgrounds, **Awesome** and **Shader effect**. For this example, click the Add button for Shader effect.
   Note: *This menu includes animated backgrounds plus other items that can be added to an app's background layer (for example, image and video widgets, and special effects). For details about the background layer, see section 2.3.4.*
   c. Click the Save button to add the background to My First Structure.
   A preview image of your selected background now displays in the right-hand pane.



MT Showcase Editor, Editing a Structure screen. **1** *'Add a new widget to background' hyperlink.* **2** *Background widget menu.* **3** *Shader effect animated background widget and Add button.* **4** *Preview of shader effect background.* **5** *Annotation widget and Add button.* **6** *Save button.*

3. Add a web browser to the structure

   a. Still in the *Editing a structure* screen, go to the Main section and click *Add a new widget to main*.

      Note: *For an overview of widgets and attributes, see section 2.1.*

   b. From the pop-up menu, choose a web browser. (You need to click the corresponding Add button.)

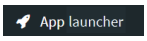
   c. Set the web browser attributes. Widget attributes control how the appearance and behavior of screen items. For this task, we only want to specify the URL for the web browser. We will experiment with other browser attributes such as location and size later.

      In the Web browser attributes pane, set the **URL** attribute to the web site you want. For example, www.multitaction.com.

      Tip: *Although you may be tempted to enable the* **Pin the widget when it opens?** *attribute, we recommend that you don't do this yet. Although this would allow you to browse web pages, it would also mean that you cannot move or resize the browser widget until you add a toolbar in section 3.5. For more information, see the note in step 5.*

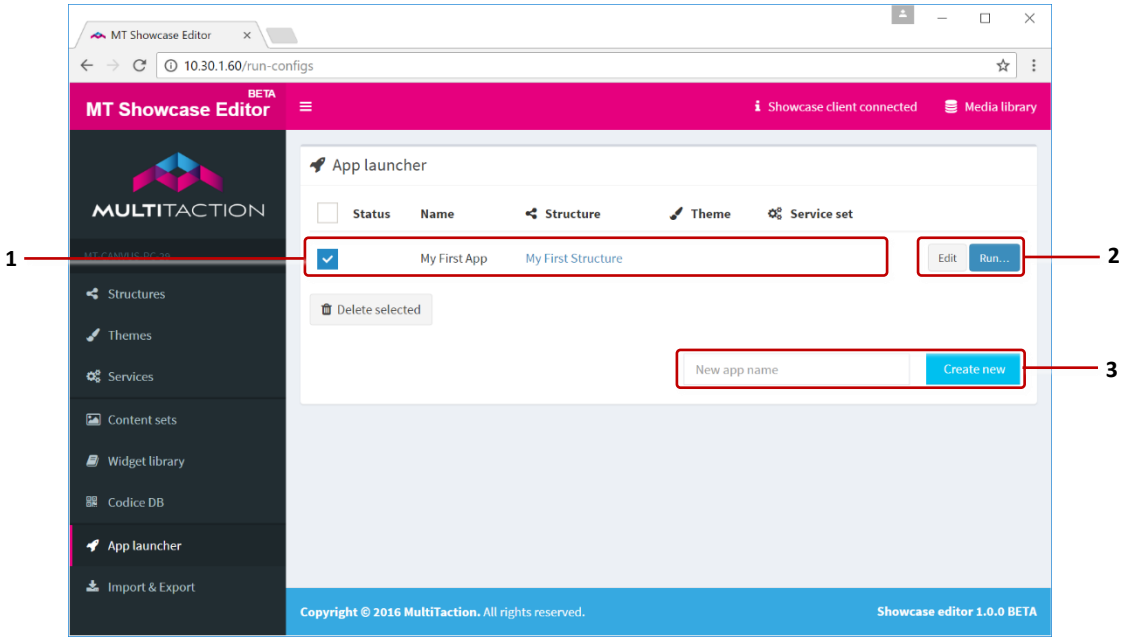
   d. Click the Save button to add the browser to My First Structure.

4. Create a new app

   a. Click App launcher  in the left-hand menu.

   b. Enter a name in the *New app name* field. For example, My First App.

   c. Click the Create New button.



MT Showcase Editor, App launcher screen. ***1*** *New app.* ***2*** *Edit, Run and Save buttons.* ***3*** *New app name field and Create New button.*

5. Add the structure to the new app

   a. Still in the *App launcher* screen, click the Edit button for My First App.

   b. In the Structure column, choose My First Structure from the drop-down list.

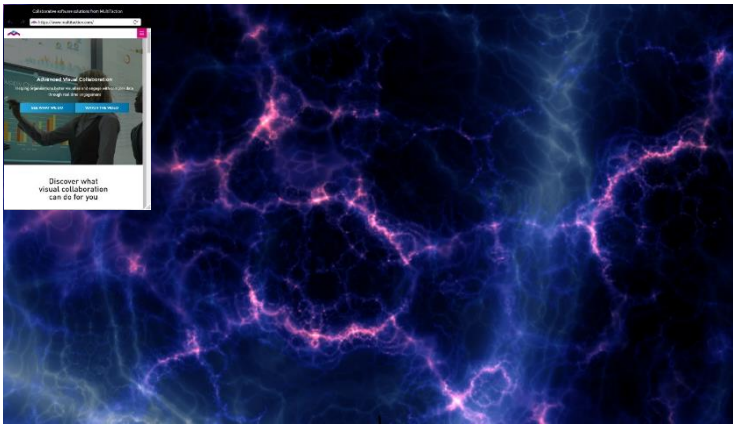
   c. Click the Save button for My First App.

6. Run the new app

   Still in the *App launcher* screen, click the Run button for My First App.

   The new app appears on your video wall. Use hand gestures (see section 3.1.1) to move, resize and rotate the browser. To remove the browser, drag it off the screen.

   Important: *If you try to browse pages on the web site, you will not be able to because the widget will move. This is because MT Showcase interprets your touches as move or resize gestures. To fix this, all browser widgets must be pinned before users can use them; this will be covered when you add a toolbar in section 3.5.1.*

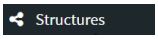


MT Showcase app, with web browser widget in default location

In the next section, you will learn how to edit your new app.

3.4 Edit your app

Like most design processes, designing an MT Showcase app requires trial and error adjustments until you get the results you want. In this section, you will experiment with adjusting the background and browser attributes.

1. Edit the background
   a. Click Structures  in the left-hand menu.
   b. Click the My First Structure hyperlink.
      The *Editing a structure* screen appears.
   c. In the Background section, click the Awesome widget.
   d. In the Awesome attributes pane, experiment with the animation speed.
      Try raising the value for the **Effect speed** attribute.
   e. Click the Save button.
2. Review the app changes on your video wall
   If My First App is still running, the background change takes effect immediately. Otherwise, follow the instructions in step 5 of section 3.3.
   If required, keep, adjust or undo the changes to the Awesome background.
3. Edit the browser
   a. Still in the *Editing a structure* screen, go to the Main section and click the Web browser widget.
   b. In the Web browser attributes pane, experiment with new values for the browser attributes. For example:
      - **Location:** You define a widget's location by specifying the horizontal and vertical coordinates of its top-left corner. By default, the coordinates for widgets in the main layer are (0,0), meaning they display in the top-left corner of the screen.
        Experiment with new locations. For example, enter co-ordinates of (200,0) to shift the browser 200 pixels to the right.
      - **Scale:** Experiment with different scaling factors for the widget. For example, enter a value of 1.5 to display the browser at 150% of its default size.
      - **Enable resize handle:** Set this attribute to Enabled to show a resize handle in bottom right corner of the browser.
      - **URL:** Experiment with a different web site URL.
   c. Click the Save button.
4. Review the app changes on your video wall
   If My First App is still running, the browser changes take effect immediately. Otherwise, follow the instructions in step 5 of section 3.3.
   If required, keep, adjust or undo the changes to the browser.

In the next section, you will learn how to extend your new app.

3.5 Add a toolbar

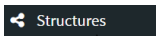
It's often useful, or even necessary, to add toolbars to widgets in your app. Toolbars can include Close, Pin and Maximize buttons, plus Color and Clear buttons for widget annotations (see section 14).

The following sections show you how to add a toolbar containing Close and Pin buttons to your web browser. (The Pin feature is explained in section 3.5.3.)

3.5.1 Create a new toolbar

Before you can assign a toolbar to the web browser, you must create and name a toolbar widget. Then you must edit the toolbar and select the toolbar buttons.

Follow these steps

1. Edit the Web browser attributes
   a. Click Structures 

in the left-hand menu.
   b. Click the My First Structure hyperlink.
   c. In the Main section, click the Web browser widget.
2. Add a new toolbar
   a. In the Web browser attributes pane, go to the **Toolbar** attribute.
   b. Click the New button.
   c. In the *Add a new Toolbar widget* pop-up, click the Add button.

Add a new Toolbar widget pop-up

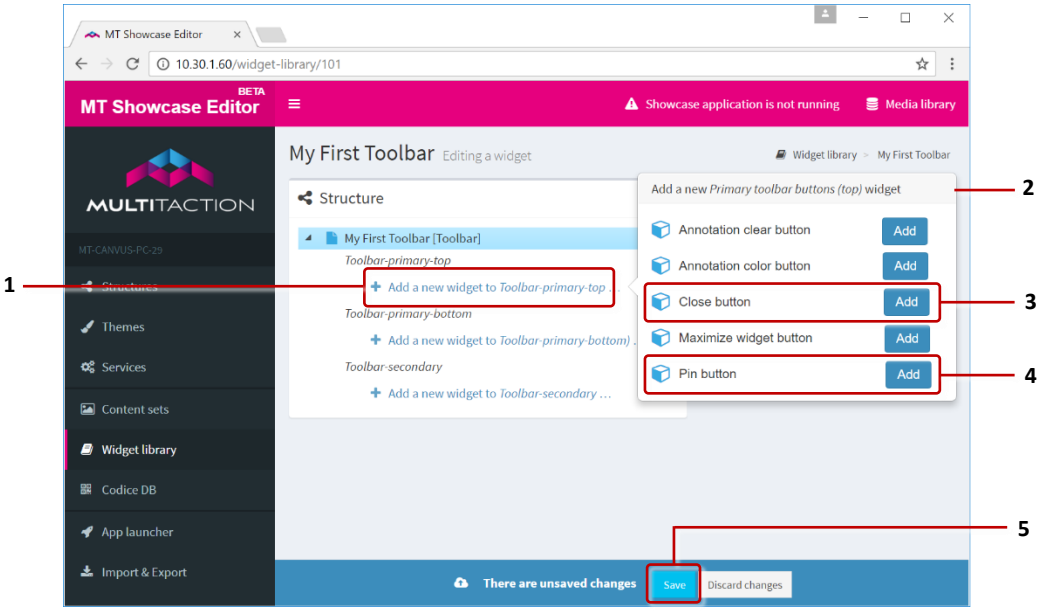
   d. Type a name for the new toolbar widget. For example, My First Toolbar.
   e. Click the Save button.
3. Now you can assign buttons to the new toolbar.
   a. Click the Edit button to display the *Editing a widget* screen.
   b. In the Toolbar structure pane, click the *Add a new widget to Toolbar primary-top* hyperlink.

   Note: *MT Showcase supports primary and secondary toolbars. The toolbar type determines the position of the buttons; see section 14.2.*

c. In the pop-up *Add a new Toolbar* menu, add the Close and Pin buttons. (Click the corresponding Add buttons.)

   Note: The Pin feature is explained in section 3.5.3.

d. Click the Save button.



Toolbar structure pane. 1 *'Add a new widget to Toolbar primary-top' hyperlink.*
2 *'Add a new toolbar' menu.* 3 *Close button.* 4 *Pin button.* 5 *Save button.*

3.5.2 Test the toolbar

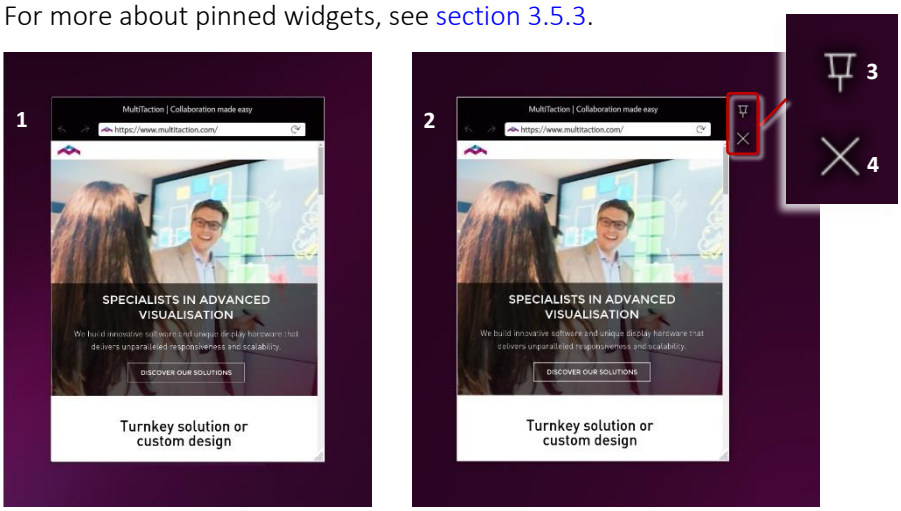
You have now added a new toolbar to your web browser. The and buttons in the toolbar allow users to pin or close the browser.

To test the toolbar:

1. Restart your app.
   a. Go to the *App launcher* screen.
   b. Click the Restart button for My First App.
2. The new toolbar is now attached to the web browser. Try using the new buttons.

   If you close the browser, you can repeat step 1 to re-start your app.

   For more about pinned widgets, see section 3.5.3.



*Example widget toolbar. **1** Browser widget with no toolbar. **2** Browser widget with toolbar. This toolbar is in the primary-top position. **3** Pin button. **4** Close button.*

3.5.3 About pin

Normally, you can freely move and resize individual screen items (widgets) in an app. But this is not always desired behavior. Sometimes you may want to lock the size and position of an item so you do not inadvertently move it off the screen or make it too small to be useful. The pin feature allows you to lock the size and position of screen items.

When designing your app, you can edit a widget's attributes to determine whether:

- The widget is pinned or not when it first opens.
- The widget has a toolbar with a Pin button that allows users to pin or unpin it.

Note that many widgets *must* be pinned before users can use them:

- *Cloud* widgets must be pinned before users can rotate the ball and select items for viewing.
- *Image movie* widgets must be pinned if you want to open the movie in interactive mode, allowing users to control the playback.

List continues on next page.

- *Input Visualizer* widgets (in the Main layer) must be pinned before users can test touch events.

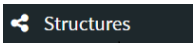
  Tip: *An Input Visualizer in the Overlay layer does not need pinning.*
- *Missile Command* and *Pong* widgets must be pinned before users can play these games.
- *PDF Book* and *PDF Flow* widgets must be pinned before users can page through the document.
- *Taction model* widgets must be pinned before users can interact with a 3D model of a MultiTaction Cell.
- *Web browser* widgets must be pinned before users can browse the web page.

4 Create a structure

If you skipped the instructions for creating a simple app in section 3.3, this section briefly describes how to create a new structure.

A *structure* defines combination of widgets in each layer of an MT Showcase app. There are four layers: Background; Main; Menu; and Overlay. For descriptions of these layers, see section 2.3.

To create a structure, follow these steps:

1. Click Structures 

in the left-hand pane.
2. Click My First Structure.
   The *Editing a structure* screen appears.
3. Now edit the Overlay, Menu, Main, and Background layers:
   - **Overlay layer:** This is the topmost layer in an MT Showcase app. Widgets in this layer operate above any screen content in the main layer.
     Overlay widgets include the Champagne and Ripple special effects widget and the Input Visualizer widget. For more information, see section 2.3.1.
     Instructions for adding the Input Visualizer are provided in section 13.1. To add items to the overlay layer, follow and adapt these instructions as required.
   - **Menu layer:** This layer is where you define a finger menu or rotating menu. To open a menu, users tap and hold any area of empty screen. Finger menus are described in section 5; rotating menus are in section 8.
   - **Main layer:** This layer includes the content widgets that users see when they use the app. It can include any combination or image, video, browser and PDF widgets. It can also include other widgets such as animated cloud widgets, content hotspots, teasers, and detectors for personal markers.
     - To add a teaser menu, see section 9.
     - To add a cloud, see section 10.
     - To add a content hotspot, see section 11.
     - To add a web browser to the main layer, see step 3 of section 3.3.

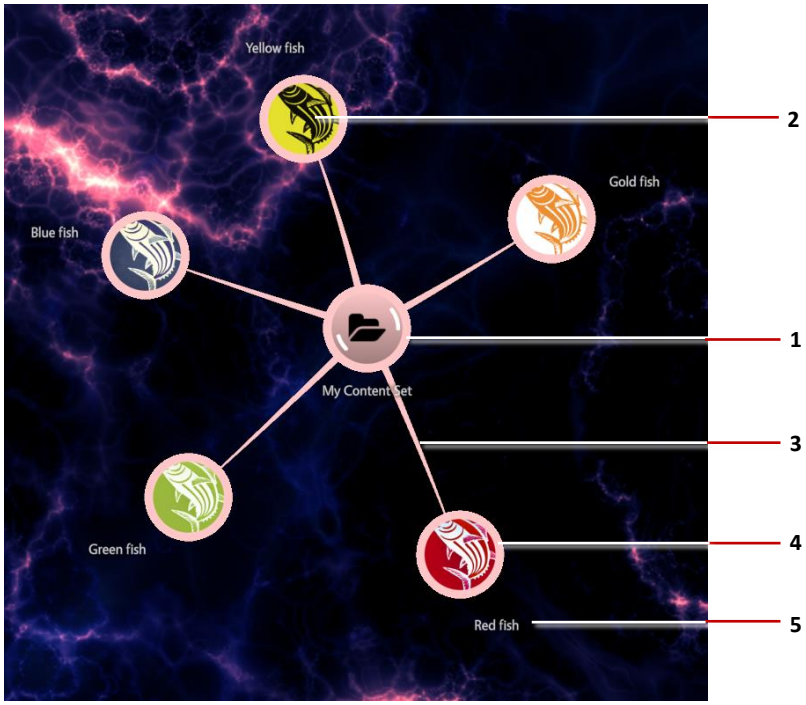
     To add other items to the main layer, follow and adapt the above instructions, as required.
   - **Background layer:** The background layer defines the screen background. It can include an animated background, interactive special effects, and images and movies in fixed positions on the screen. It can also include the annotation widget. For overviews of the available items, see section 2.3.4.
     Instructions for adding Awesome to the background layer are provided in step 2 of section 3.3. To add alternative items to the background layer, follow and adapt these instructions as required.

5 Create a finger menu

The following sections describe how to create a basic finger menu and the content set that supplies the menu.

A finger menu displays when a user taps and holds an empty area of the screen. They can also be launched from content hotspots or with a Codice card. A finger menu is a circular animated menu, with child bubbles radiating from a central bubble. Child bubbles represent individual menu items and can display thumbnail images. (Separate widgets control the appearance and behavior of the individual bubbles and the connectors that radiate out from the central bubble to the child bubbles.)

Note: *These sections do* not *describe how to leverage menu settings defined in a theme; that is covered later in section 6.*



Example finger menu.

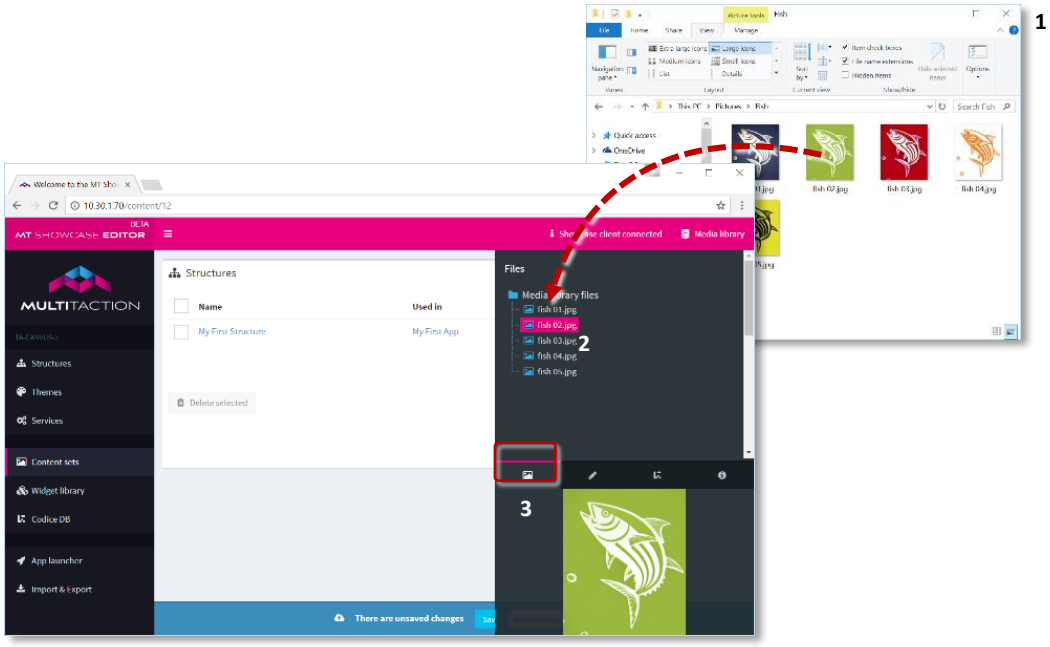
1 *Central bubble.* 2 *Child bubble. In this example, the bubbles display thumbnails (or 'preview images') of the menu items. Tap any child bubble to display the menu item.* 3 *Bubble connector.* 4 *Bubble background color.* 5 *Bubble visible name.*

5.1 Set up a content set

Before creating a menu, you must create the content set that supplies the menu with media such as images. To include images, videos or PDFs in your contents set, you must first add them to your media library. You can then populate the content set with items in your media library.

Follow these steps:

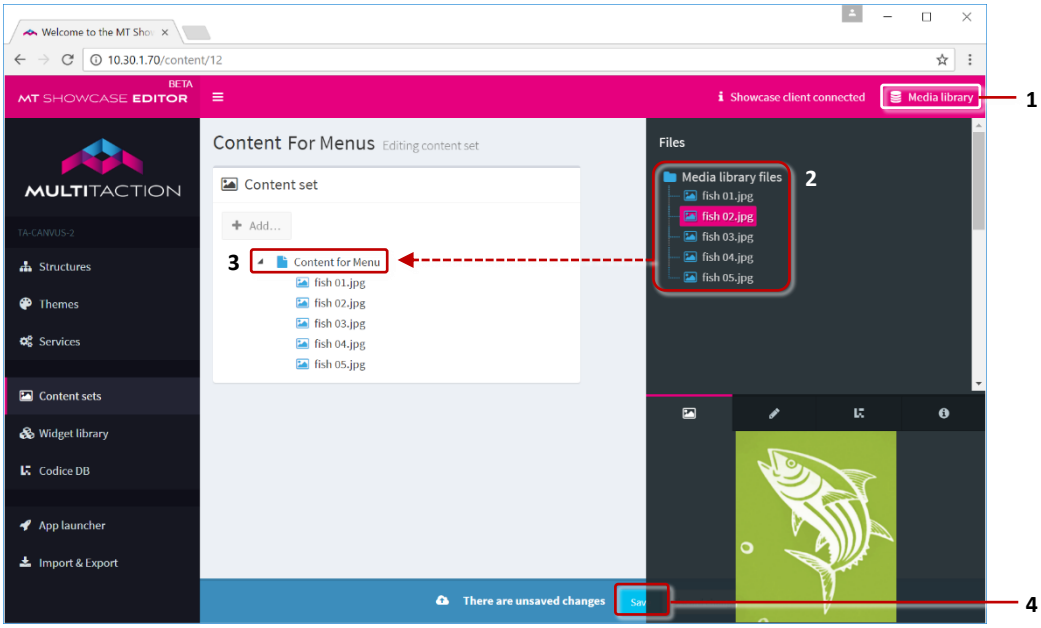
1. Drag content into the media library
   a. Click the Media library button to open the media library pane.
   b. Drag some content into the media library from Windows Explorer. For example, drag a folder that contains images.
   c. (Optional) You can rename items in the media library, organize items into subfolders, and preview images.
      Right-click items to rename them. Right-click the root folder to create a new subfolder. To preview an image, select it then click the Asset Preview tab.
   d. Click the Media library button again to close the media library pane.



MT Showcase Editor, Media library. Drag and drop images from Windows Explorer (**1**) *into the media library* (**2**)*. Preview items in the media library by clicking the Asset Preview tab* (**3**)*.*

2. Create a content set
   a. Click Content sets Content sets in the left-hand pane.
   b. Click the Create New button
   c. Enter the name of the new content set. For example, Content for Menus.
      The *Content Set* screen appears.

3. Drag items into the content set
   a. Click the Media library button to open the media library pane.
   b. Drag images from the media library to the root folder of the new content set. (The root folder shows the name of the content set, Content for Menus).
   c. Close the media library pane.
   d. Click the Save button to save the changes to the content set.



MT Showcase Editor, Editing a content set

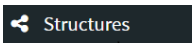
*Click the Media Library button (**1**) to show or hide the media library. Drag items from the media library (**2**) onto the content set root folder (**3**). Then hide the media library to show the Save button (**4**) and save the changes.*

Now you have created and populated your content set, you can add a finger menu to your app's structure. Go to section 5.2.

5.2 Add a finger menu to your app's structure

Now you need to add a finger menu to the structure you created earlier (see step 1 in section 3.3). Then you need to configure the menu to use the new content set, Content for Menus.

Follow these steps:

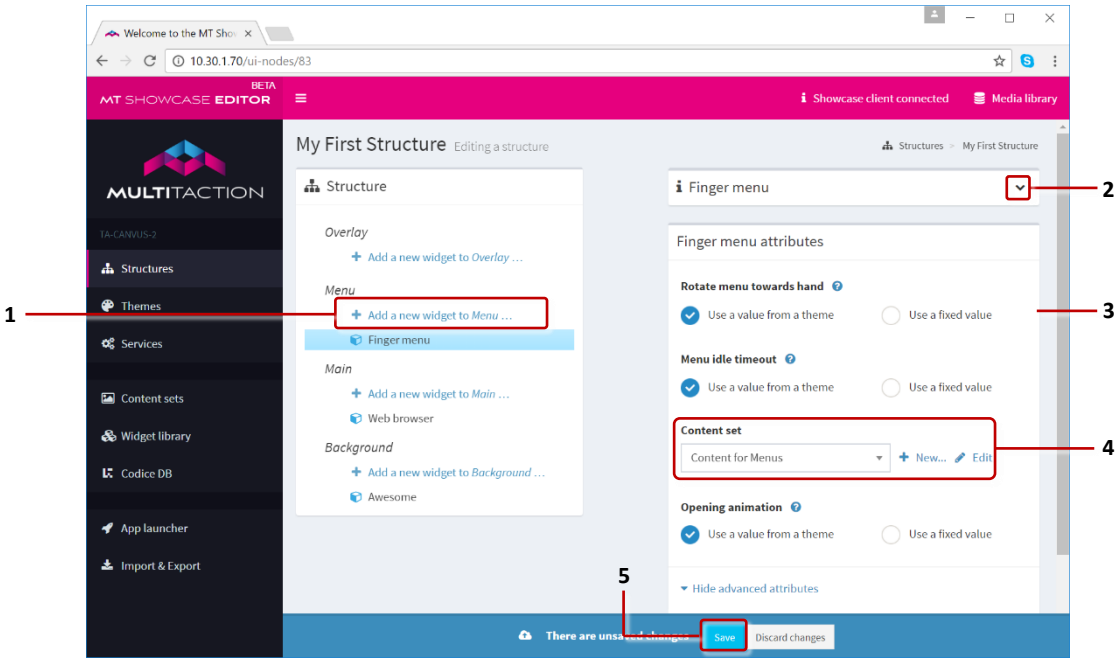
1. Click Structures  in the left-hand pane.
2. Click My First Structure.

   The *Editing a Structure* screen opens.
3. Go to the Menu section and click *Add a new widget to Menu*.
4. From the pop-up menu, add a **Finger menu**.

   Tip: *Click the Add button.*

   Tip: In the right-hand *Finger menu pane*, click the ⌄ drop-down arrow to show a description and preview image of a finger menu.
5. In the *Finger menu attributes* pane, select Content for Menus from the Content Set dropdown menu.
6. Click the Save button to add the menu to My First Structure.

Now you must edit the content set that supplies your finger menu. Go to section 5.3.



MT Showcase Editor, Editing a Structure screen. **1** *'Add a new widget to Menu' hyperlink.* **2** *Finger menu pane. Click the arrow to display a description and preview image.* **3** *Finger menu attributes pane.* **4** *Content Set dropdown menu with Add and Edit buttons.* **5** *Save button.*

5.3 Edit the content set

In this section, you are going to open the *Editing a content set* screen. From here, you will be able to configure the appearance and behavior of the new finger menu.

Follow these steps:

1. In the *Editing a Structure* screen, click the new Finger menu widget.
2. In the **Finger menu attributes** pane, go to the **Content Set** attribute. (See the screenshot on page 41.)
3. Click the Edit button to display the *Editing a content set* screen.

Now you must configure the central bubble in your finger menu. Go to section 5.4.

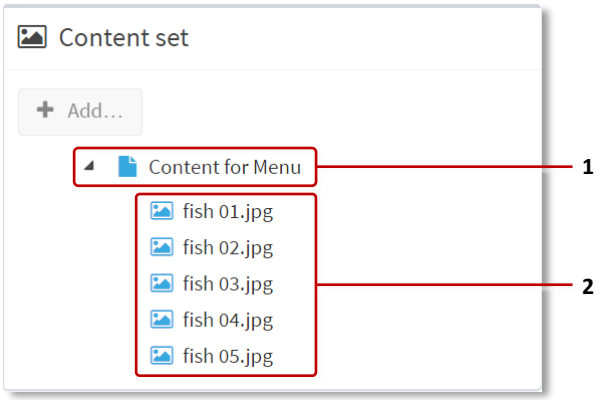
5.4 Configure the central bubble

In this section, you will edit the relevant attributes for the central bubble in your finger menu. As part of this procedure, you will create a shared widget (see section 2.2.4). This will be a *Menu bubbles* widget.

Follow these steps:

1. In the *Editing a content set* screen, click the root folder in your content set (labelled Content for Menus).

   The root folder represents the central bubble in your finger menu.



Example content set. **1** *Root folder.* **2** *Individual items in content set.*

2. In the right-hand attributes pane, go to the *Widget attributes* section and click the Finger menu tab.

   This tab lists the all the attributes, or configurable properties, of the central bubble. For each attribute, you have two options:

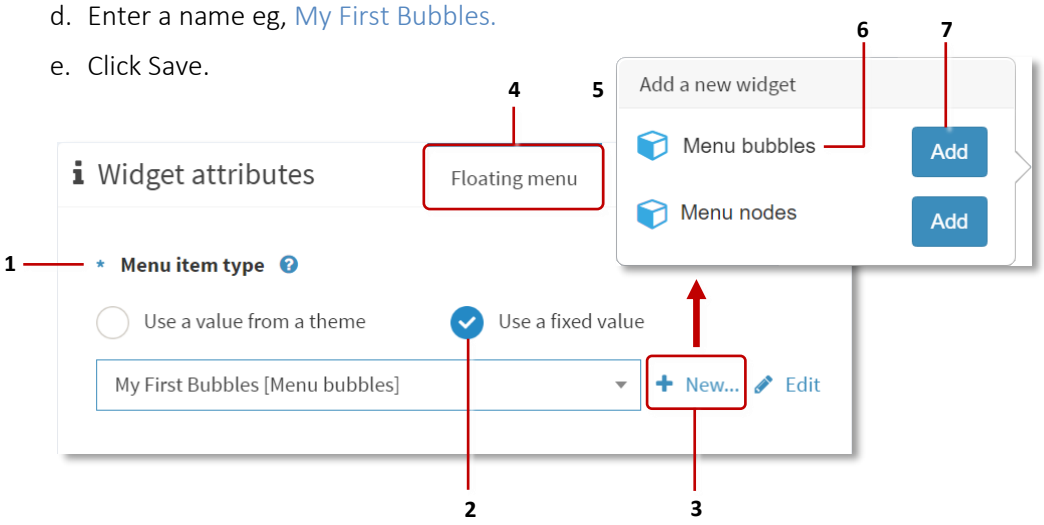
   - **Use a value from a theme:** When this option is selected, the attribute inherits its value from a *theme* (see section 2.4). If no theme has been created, the attribute inherits the default value.
   - **Use a fixed value:** When this option is selected, the attribute uses a custom value provided by you.

   For simple menus, you can leave most attributes to '*use a value from a theme*'.

3. (Optional) Go to the **Preview image** attribute and specify a thumbnail image for the teaser bubble.

   Click Media library to open the media library. Then drag the image you want into the Preview image field.

4. Go to the **Menu item type** attribute.
   a. Select 'Use a fixed value'
   b. Click **+ New...**
   c. Add the Menu Bubbles widget.
   d. Enter a name eg, My First Bubbles.
   e. Click Save.



'Editing a content set' screen, widget attributes pane.

1 *Menu Item Type attribute.* 2 *Use a fixed value check box.* 3 *New button.* 4 *Finger menu tab.*

5 *Add a new widget popup.* 6 *Menu bubbles widget.* 7 *Add button.*

Note: *You have now created a new* Menu bubbles *shared widget. This widget is now listed in the widget library; see section 2.2.4.*

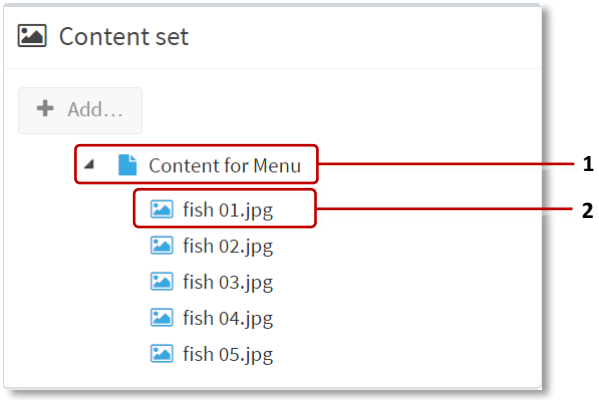
This completes the setup for the *central bubble*; you do not need to edit any of its other attributes to get the finger menu working. Now you need to edit the attributes of the *child bubbles*; go to section 5.5.

5.5 Configure the child bubbles

As part of this procedure, you will create another shared widget. This will be a *Bubble connector type* widget. You will also reuse the *Menu bubbles* widget that you created in the previous section.

1. In the *Editing a content set* screen, click the first item below the root folder in your content set.

   This item represents the first child bubble in your finger menu.



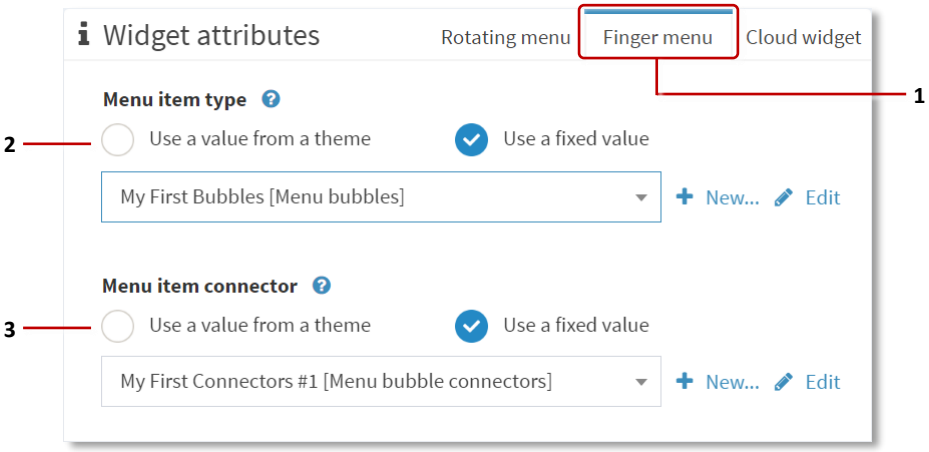
Example content set. ***1*** *Root folder.* ***2*** *First item in content set.*

2. In the right-hand attributes pane, go to the Widget attributes section and click the Finger menu tab.

   This tab lists the all the attributes, or configurable properties, of the child bubble. As with the central bubble, each attribute has two options:

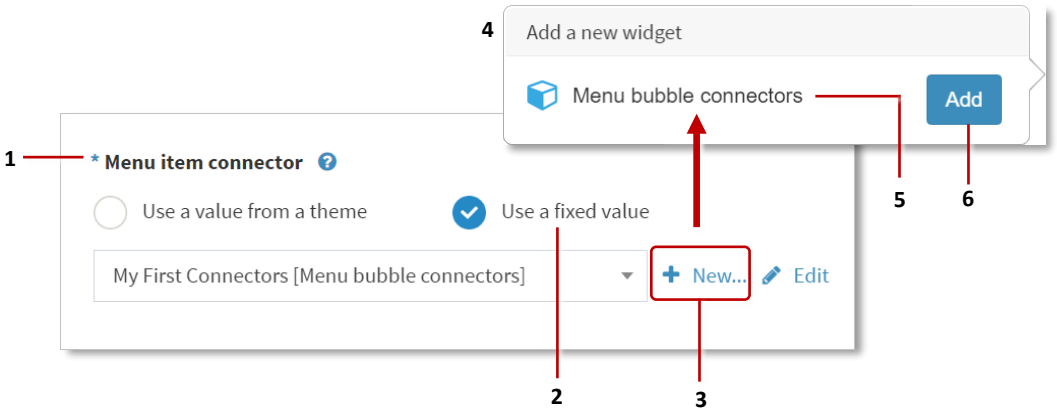
   - Use a value from a theme
   - Use a fixed value

   In general, you can leave most attributes to '*use a value from a theme*'. But in this section, you will configure one attribute to *'use a fixed value'*.



'Editing a content set' screen. Widget attributes pane for finger menu. **1** *Finger menu tab.* **2** *Menu item type attribute.* **3** *Menu item connector attribute.*

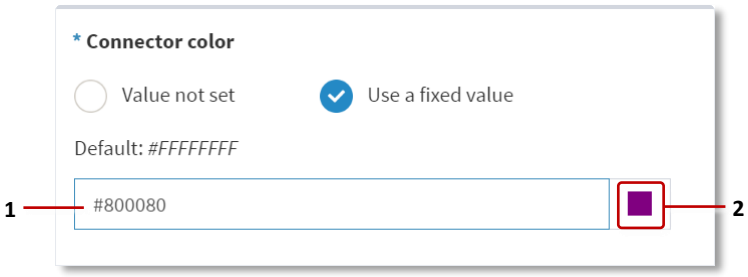
3. Go to the **Menu item type** attribute.
   a. Select 'Use a fixed value'.
   b. From the drop-down menu, choose the Menu Bubbles widget you created in the previous section ie, My First Bubbles.
   c. Click Save.
4. Go to the **Menu item connector** attribute.
   a. Click the New button.
   b. Add the Menu Bubble Connectors widget.
   c. Enter a name eg, My First Connectors.
   d. Click Save.



'Editing a content set' screen, widget attributes pane.
1 *Menu Item Connector attribute.* 2 *'Use a fixed value' check box.* 3 *New button.*
4 *Add a new widget popup.* 5 *Menu bubble connectors widget.* 6 *Add button.*

5. As before, to get a finger menu working, you do not need to edit any attributes of the new My First Connectors widget. But for illustration purposes, in this step you will edit the connector color.
   a. Go to the **Menu item connector** attribute and click **Edit**.
   b. In the Menu bubble connectors attributes pane, go to Connector color attribute.
   c. Select 'Use a fixed value'.
   d. By default, bubbles connectors are white. Specify a new connector color.
      For example, for a **purple** bubble connector, click the preview swatch to open a color picker or enter one of these in the input box:
      RGBA: `rgba(128,0,128,1)`
      Hex: `#800080`
      Name: `purple`
   e. Click Save.

Note: *You have now created a new* Menu Bubble Connectors *shared widget. This widget is now listed in the widget library; see section 2.2.4.*



*Connector Color attribute. **1** Input box. **2** Preview swatch. See section 25.3 for details of color input methods.*

6. Specify which Showcase widgets are used to display your menu items:
   a. Go to **Content view** attribute.
   b. Select 'Image Viewer' from the dropdown menu.
   c. Click Save.
7. Repeat steps 1 to 6 for each remaining item in your content set.

You have now finished configuring the finger menu. Next, you need to test the menu in your app. Go to section 5.6.

5.6 Test the finger menu

You have now updated the structure to use your customized finger menu.

To test the menu:

1. Restart your app.
   a. Go to the *App launcher* screen.
   b. Click the Restart button for My First App.
2. Tap and hold any empty area of the screen. After a brief delay, the menu opens:

Example finger menu with custom color bubble backgrounds and connectors

5.7 Next steps

The previous sections described the minimum steps necessary to create a finger menu in your app. But the MT Showcase Editor offers a rich selection of options to customize the appearance and behavior of these menus in your app.

5.7.1 Explore the available bubble and connector attributes

We recommend that you browse and experiment with the available attributes for your new My First Bubbles and My First Connectors widgets. (See section 5.4 and section 5.5 respectively.)

For example, you can configure the bubble label, size and color. You can also specify whether to display thumbnail images on the bubbles. For the bubble connectors, you can specify the color, thickness, how much the connector bends, and even a 'texture' image. In all cases, click the button to see the attribute help.

Tip: *You can browse the attributes for these shared widgets in the Widget Library; go to the* Menu bubbles *and* Bubble connector type *sections. See section 2.2.4 for details.*

5.7.2 Use a theme to streamline menu setup

The previous sections described the simplest way to set up a finger menu. But for the greatest flexibility, you can define menu, bubble and connector attributes in a *theme*, and then assign that theme to your app.

The key advantage to this approach is that menus are quicker to set up. If the menu, bubble and connector attributes are defined in a theme, all items in your *content set* will inherit these attributes automatically. You do not need to configure each item individually.

Likewise, if you have multiple Showcase apps, you can configure them to all use the same theme to ensure a consistent appearance and behavior across your apps

For example, you can set a global connector length and global 'physical' properties such as bubble mass and bubble friction (the virtual bubbles are designed to mimic the behavior of real objects).

To define menu attributes in a theme:

1. Click Themes in the left-hand pane.
2. In the Themes screen, click the theme you want.
3. In the *Editing a theme* screen, explore the available attributes for the following components:
   - Finger menu
   - Menu bubble connectors
   - Menu bubbles

For details about setting up a theme, see section 6.

5.7.3 Twitter menus

You can add a Twitter finger menu to your app. Menu items comprise individual tweets that match the specified search terms. To create a Twitter menu, you simply set up a content set that includes a Twitter Feed widget, and then assign this content set to a Finger menu. For details, see section 17.4.

Note: *Twitter menus require the Twitter Connection service to operate. For details about adding this service to your app, see section 12.3.*

6 Create a theme

A *theme* is a collection of attribute values for MT Showcase widgets. If your app uses a theme, all widgets in the app will inherit their attribute values from this theme by default.

For an overview of themes in MT Showcase, see section 2.4.

6.1 Why use a theme?

Themes are not essential, but we strongly recommend you use them. They offer the following benefits:

- **Faster setup:** Using a theme, it can be far quicker to set up an app. For example, instead of individually configuring each child bubble in a finger menu (see section 5.5), you can simply configure the *Menu bubbles* component in a theme. You only need to do this once. Subsequently, child bubbles in any menu will automatically inherit the attributes defined the theme's *Menu bubbles* component.
- **Consistency:** A theme provides consistent widget appearance and behavior across an app. In fact, a theme can provide consistency across multiple apps if they all use the same theme (see 'Re-use' below).
- **Re-use:** Themes simplify the process of creating apps, because the same theme can be used by one or more apps.

  For example, you may want the finger menus across all your MT Showcase apps to have a consistent look and feel. By using the same theme for all apps, you can ensure that all bubbles and bubble connectors use corporate colors and the central bubble is replaced by your corporate logo.

6.2 Theme FAQs

Can I override a theme?

Yes. When you add a widget to a structure, each widget attribute offers two options:

- **Value not set:** If you choose this option, the attribute will inherit the attribute value defined in the theme.

  If your app does not use a theme, or if this attribute is not configured in the theme, the attribute will use the out-of-the-box default value.
- **Use a fixed value:** If you choose this option, you can set a custom value for this attribute. This custom value overrides any value defined in the theme. It also overrides the out-of-the-box default value.

Can I re-use a theme?

Yes. In fact, this is the key advantage of using a theme. If you have multiple apps defined on your MT Showcase server, you can assign the same theme to each app.

Can I rename a theme?

Yes. In the MT Showcase Editor, go to the *Themes* screen and simply click the Rename button for the theme you want to rename.

Can I export and re-import a theme?

Yes. In the MT Showcase Editor, go to the *Import & Export* screen and select the check box for the theme you want to export. Then click the Export button.

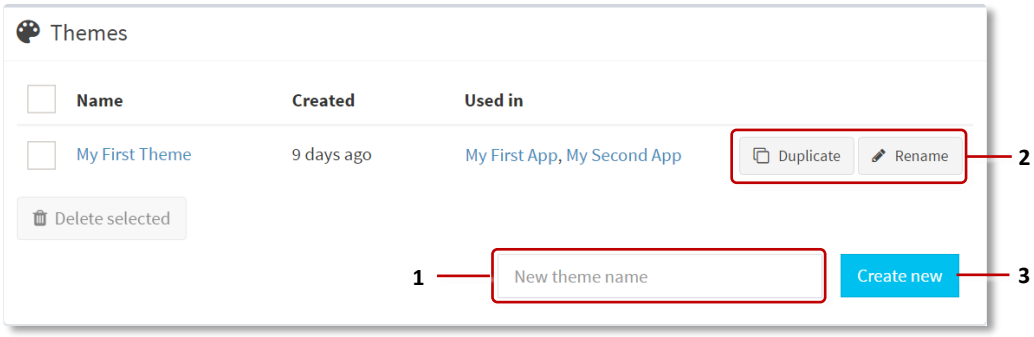
To re-import the theme, for example on a different MT Showcase server, go to the *Import & Export* screen. In the Import a File section, choose the file you previously exported. Then click the Upload button.

For full details, see section 23.

6.3 Create a new theme

Follow these steps:

1. Click Themes in the left-hand pane.
2. In the *Themes* screen, type the name of your new theme in the input box. For example, My First Theme.



Themes screen. **1** *Input box.* **2** *Duplicate and Rename buttons.* **3** *Create New button.*

3. Click the Create New button.

   The *Editing a theme* screen opens automatically.

4. Explore the *Editing a theme* screen.

   The main pane lists the available components. Expand any component to see its attributes. The right-hand navigation pane contains a hyperlinked list of the available components. Click any component to jump to that component in the main pane. See the screenshot on page 51.

5. Experiment with editing a component's attributes. For examples of how to use a theme, see the title bar example in section 6.3.1 and the finger menu example in section 6.3.2.

   Important! *Remember to click the Save button after editing the attributes for any of the components.*

6. Return to the *Themes* screen.

   My First Theme is now listed on this page. If required, you can rename the theme. You can also duplicate a theme; this allows you to use an existing theme as a template when setting up a new theme. Click the Duplicate and Rename buttons, as required.

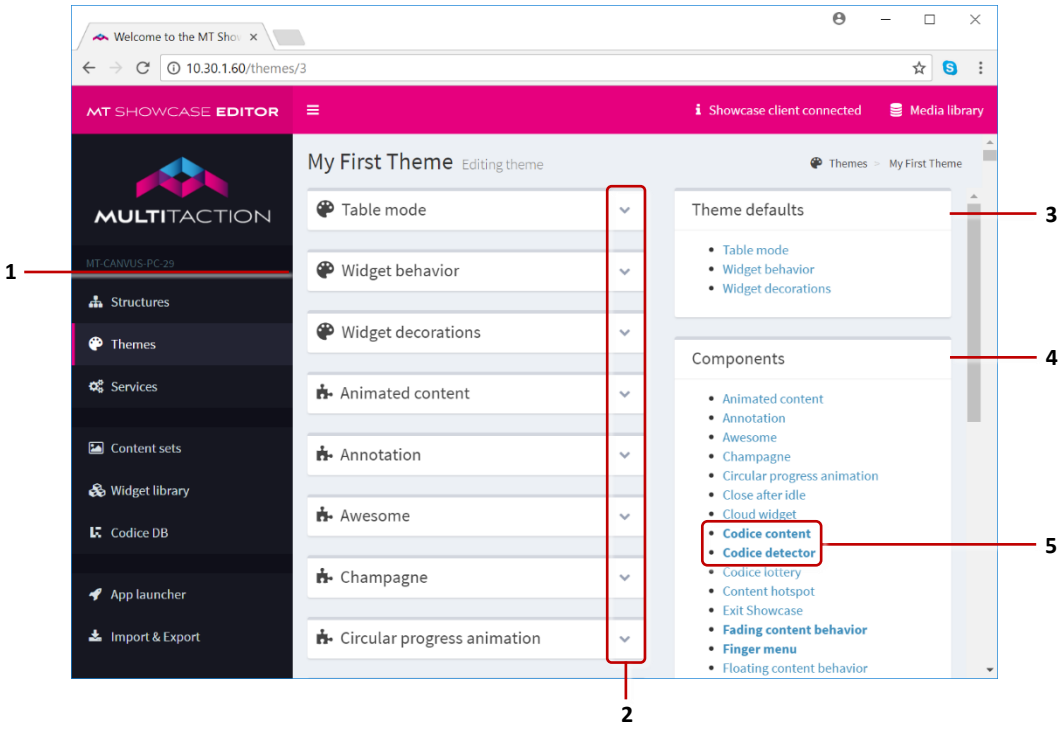
Now you need to add the new theme to your app. Go to section 6.4.

6.3.1 Title bar example

By default, image viewers have no title bar. But if you want the images in your app to have title bars, you can edit the theme so that this feature is enabled for all image viewer widgets. Follow these steps:

1. Click Themes in the left-hand pane.
2. In the *Themes* screen, click the theme you want to edit. For example, My First Theme.
3. In the *Editing a theme* screen, go to the **Image viewer** component.

   Tip: *Click the hyperlink in the Components pane.*



Editing a theme screen. **1** *Main pane, listing available components.* **2** *Drop-down arrows. Click to display the editable attributes.* **3** *Theme defaults pane.* **4** *Components pane. Click the hyperlinks to jump to the component attributes.* **5** *Edited components are shown in* ***bold****.*

4. Click the ⌄ drop-down arrow to display the Image viewer attributes.
5. Scroll down to the **Title** attribute.
6. Select the 'Used a fixed value' option.

7. Select the Enabled check box.
8. Click the Save button.

Now all image viewer widgets in your app will automatically inherit a title bar.

6.3.2 Finger menu example

If you add finger menu to your app (either added directly to the main layer of the structure, or launched from a teaser or content hotspot), you must define the bubbles and bubbles connectors. You can also edit the menu's default behavior.

This example shows how to add customized bubbles and connectors to a theme, and how to change the timeouts for finger menus and menu items.

Follow these steps:

1. Click Themes in the left-hand pane.
2. In the *Themes* screen, click the theme you want to edit. For example, My First Theme.
3. Edit the default menu timeouts.

   For this example, you will change the default timeout for idle (unused) menus and menu items. When the timeout expires, the menu or menu item closes automatically.

   a. In the *Editing a theme* screen, go to the **Finger menu** component.

      Tip: *Click the hyperlink in the Components pane.*

   b. Click the ⌄ drop-down arrow to display the finger menu attributes.
4. Go to the **Close after idle** attribute.

   This attribute sets the timeout for unused widgets, including unused menu items. (After a user opens a menu item, the item closes automatically when the user stops interacting with it.)

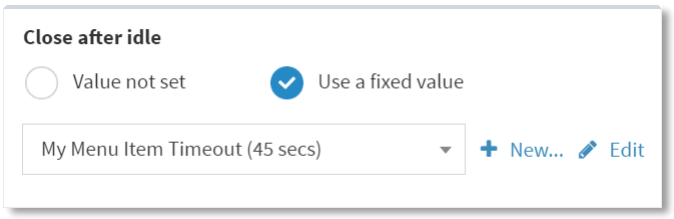
   a. Select the 'Used a fixed value' option.

   b. Click the New button to add a new 'Close after idle' shared widget. For example, My Menu Item Timeout (45 secs).

   c. Click the Edit button to display the Editing a widget screen.

   d. Set the timeout and fade duration (in seconds) for unused menu items.

   e. Click Save.

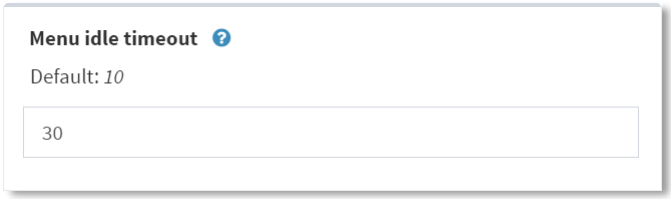


Editing a theme screen, Finger menu component: Close After Idle attribute

5. Go back to the **Finger menu** component in the *Editing a theme* screen.
   a. Click the Show advanced attributes hyperlink:

   b. Scroll down to the **Menu idle timeout** attribute.
   a. Select the 'Used a fixed value' option.
   b. By default, finger menus close after 10 seconds if unused. Enter a longer timeout, in seconds.
   c. Click Save.

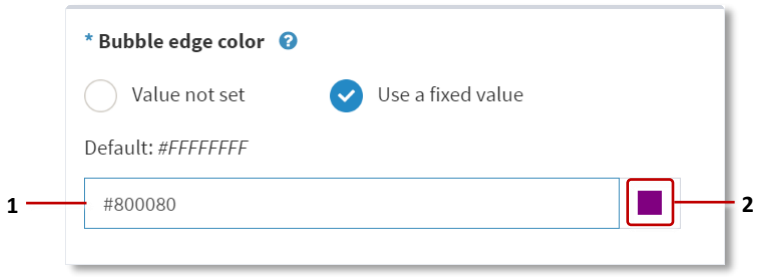


Editing a theme screen, Finger menu component: Menu Idle Timeout attribute

6. Edit the bubble attributes. For this example, you will set the color and size.
   a. In the *Editing a theme* screen, go to the **Menu bubbles** component.
   b. Click the ⌄ drop-down arrow to display the menu bubble attributes.
   c. Scroll to the **Bubble edge color** attribute.
   d. Select the 'Used a fixed value' option.
   e. Enter the color for the bubble edges (ie, the bubble border). For example, for a purple bubble edge, click the preview swatch to open a color picker or enter one of the following values:

      RGBA: `rgba(128,0,128,1)`
      Hex: `#800080`
      Name: `purple`

      Click the ? help button or see section 21.4 for details of color input methods.



Editing a theme screen, Menu Bubbles component: Bubble Edge Color attribute.
***1** Input box. **2** Preview swatch.*

   f. Scroll to the **Bubble size** attribute.
   g. Select the 'Used a fixed value' option.

h. Enter the bubble size in pixels. For example, to slightly increase the default size, enter 125.

i. Click the Save button.

7. Edit the bubble connector attributes. For this example, you will set the connector color to match the bubble color.

   a. In the *Editing a theme* screen, go to the **Menu bubble connectors** component.

   b. Click the ⌄ drop-down arrow to display the connector attributes.

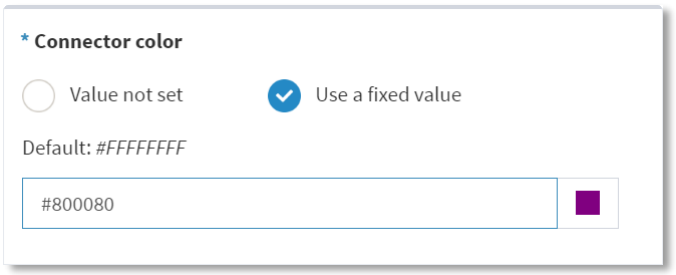
   c. Click the Show advanced attributes hyperlink.

   d. Scroll to the **Connector color** attribute.

   e. Select the 'Used a fixed value' option.

   f. Enter the same color that you used for the bubble edges in step 6c.

   g. Click the Save button.



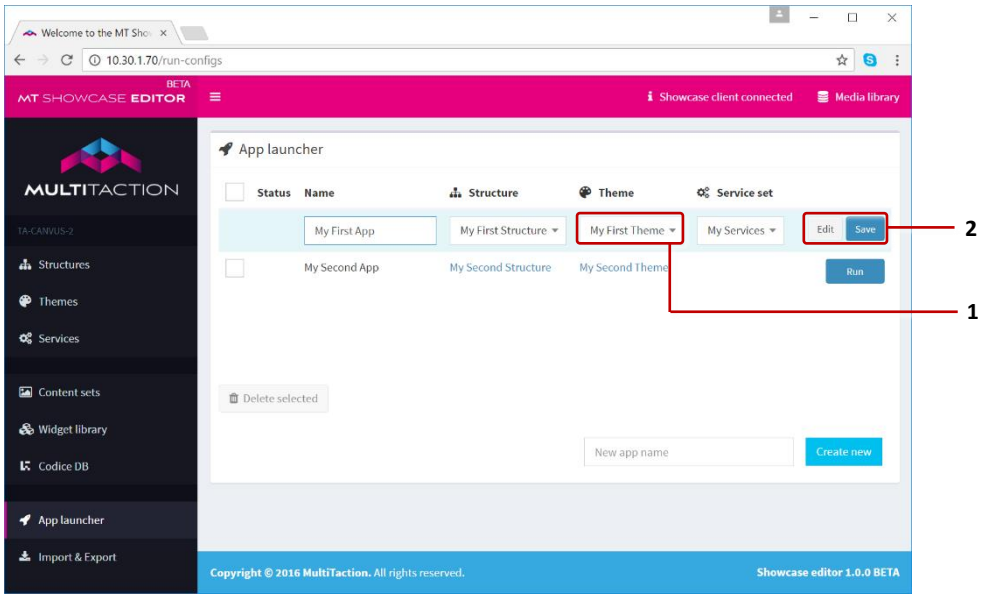
Editing a theme screen, Menu Bubble Connectors component: Connector Color attribute

Now all finger menus in your app will automatically inherit these menu, bubble and connector properties.

6.4 Add a theme to your app

Follow these steps:

1. Click App launcher in the left-hand pane.
2. Click the Edit button for your app. For example, My First App.
3. In the Theme column, choose My First Theme from the drop-down list.
4. Click the Save button for My First App.



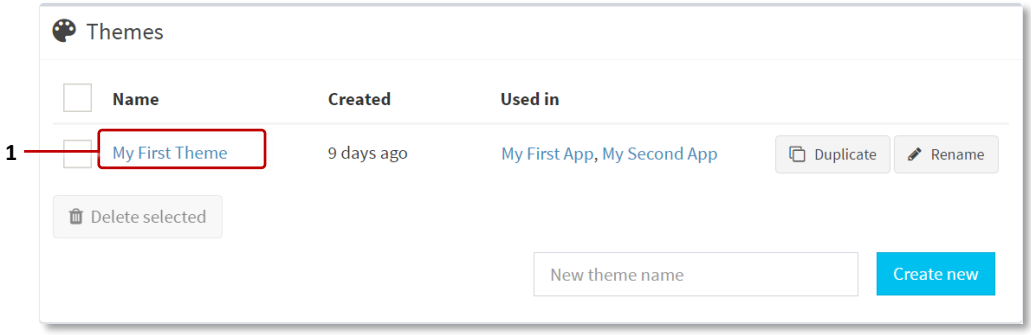
App launcher screen. **1** *Theme drop-down list.* **2** *Edit and Save buttons.*

6.5 Edit an existing theme

Follow these steps:

1. Click Themes in the left-hand pane.
2. In the *Themes* screen, click the theme you want,

   The *Editing a theme* screen opens automatically.



Themes screen. **1** *Theme name. Click to edit the theme.*

6.6 Set up theme defaults

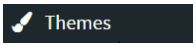
Each theme includes three sets of default attributes:

- **Table mode:** You can configure MT Showcase to run on touchscreen tables by enabling table mode. In this mode, MT Showcase automatically detects which side of the table the user is on when they tap a widget, open a menu, or place a Codice card on the screen. It then rotates screen content to face the user. See section 18.
- **Widget behavior:** You can configure default behavior for widgets in your app:
  - **Pin the widget when it opens:** Certain widgets, such the cloud and browsers, must be pinned before users can use them. This attribute ensures that all widgets are automatically pinned when they open.

    Note: *If you enable this theme default, users cannot move or resize widgets unless you also add a toolbar with a Pin button, enabling them to unpin the widget. See section 3.5.1*
  - **Maximize widget when it opens:** If enabled, all widgets, including menu items, expand to fill the entire screen when they open. (To reset a widget to its normal size, users tap the edge of the screen to display the  Unmaximize button.)
  - **Snap to angle:** You can constrain the incremental rotation of widgets so they automatically re-adjust themselves to a fixed angle of rotation. The effect is similar to a boat righting itself after keeling over. See section 21.3.
  - **Size limit:** This attribute sets a minimum and maximum size for widgets. If a user makes a widget too big or too small, it automatically resizes. See section 21.2.
  - **Close after idle:** This attribute sets the timeout for any unused widget, including unused menu items. If no user has interacted with a widget before the timeout expires, the widget closes automatically.
  - **Widget sounds** and **Interaction sounds:** These attributes play sounds (WAV files) when a widget opens and closes, or when a user touches the widget with their hand or infrared pen. They can also loop a sound continuously while a widget is open, or while an app is running. See section 16.
- **Widget decorations:** You can configure the default appearance of widgets in your app:
  - **Toolbar, Caption** and **Title:** These attributes are self-explanatory. If enabled, all widgets display these attributes by default.
  - **Apply an overlay:** This attribute displays a 'Pin to interact' overlay over each widget. (Widgets such as clouds, browsers and PDF documents must be pinned before users can use them.)
  - **Annotation:** This attribute applies an annotation overlay to each widget. This overlay allows users to annotate the widget using an infrared pin or their finger.

6.6.1 Edit the theme defaults

Follow these steps:

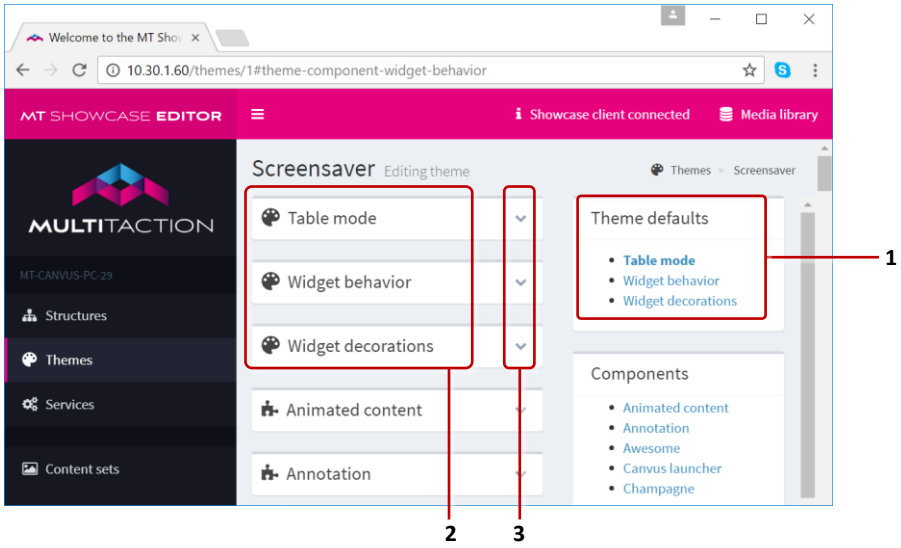
1. Click 

in the left-hand pane.
2. In the *Themes* screen, click the theme you want,

The *Editing a theme* screen opens automatically.

3. In the *Editing a theme* screen, go to **Theme defaults** pane and click the item you want. (In the main pane, the default set expands automatically to display the default attributes.)

4. Edit the default attribute and click the Save button.



Editing a theme screen, Theme defaults. 1 *Theme defaults pane.* 2 *Default sets.* 3 *Click here to expand the default set and display the default attributes.*

6.6.2 How to override a theme default

Theme defaults apply to all widgets. But you can override any theme default by assigning a different value to the attribute in any of the *theme components* or to a specific widget in your app's *structure*.

For example, you could configure **Close after idle** with a 60 second timeout as the general default, a 120 second timeout for Video Viewer widgets, and a 180 second timeout for a specific Video Viewer widget in your app. To do this, edit the following attributes in your app's theme and structure:

Type of default	Attribute location	Attribute value
General timeout for all widgets	Theme > Theme Defaults > Widget Behavior > Close After Idle	60 seconds
Default timeout for Video Viewers	Theme > Components > Video Viewer > Close After Idle	120 seconds
Timeout for specific Video Viewer	Structure > Main > Video Viewer > Close After Idle	180 seconds

7 Content widgets

This section provides an overview of widgets that display content such as images or videos.

7.1 Image Viewers

Image viewer widget displays image files, such as PNG, JPEG, BMP and TIF files.

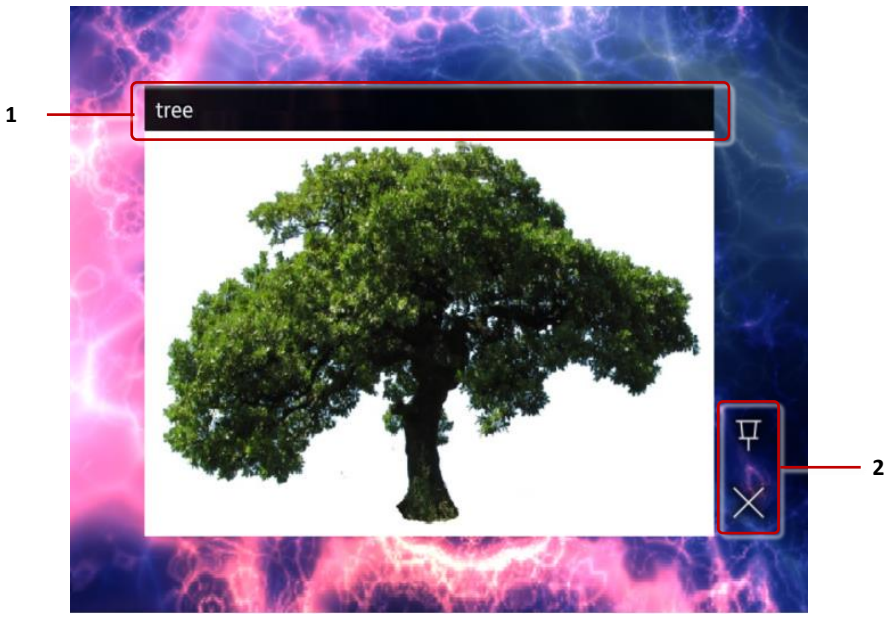


Image viewer widget. 1 *Image title.* 2 *Toolbar with pin and close buttons.*

You can add a new Image viewer widget to your app by dragging and dropping an image file from the media library to a structure or content set in the MT Showcase Editor. Alternatively, you can create a new empty Image viewer widget in a structure and drag and drop the image file into the *Image file* attribute, or create a new File asset node in a content set and drag and drop the image file into the *Asset* field. For more information about the media library, see section 2.7.

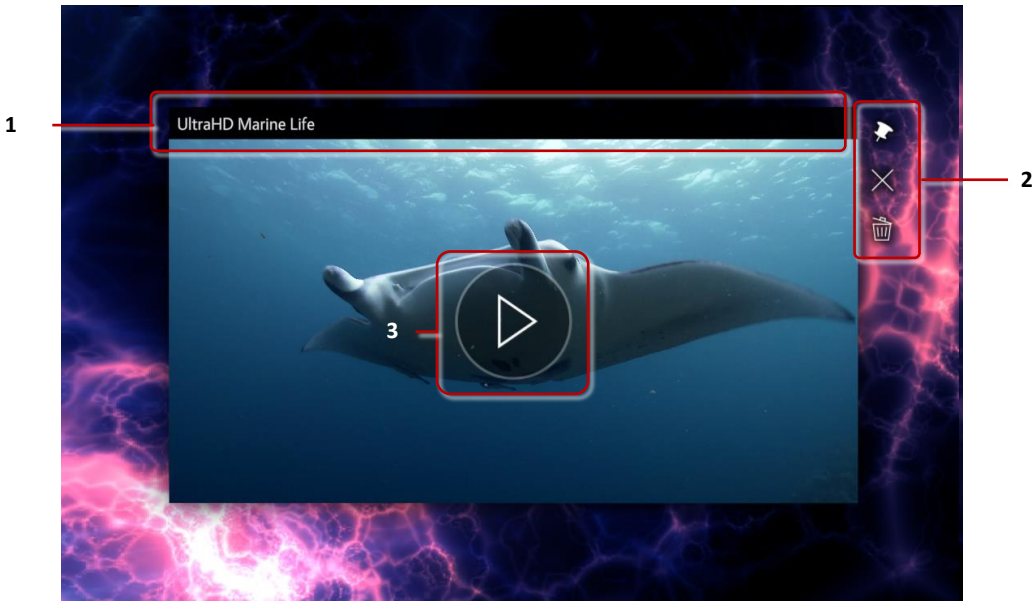
You can configure an initial location and size for the image viewer widget, and choose whether users can move it or not. You can add a toolbar to the widget, and select the available buttons, for example a pin and a close button. See section 14 for more information about toolbars. You can also add decorations to the image, such as a title bar or a caption. You can even allow users to annotate on top of the image with IR pens. See section 15 for more information about Annotations.

You can add images directly to the main or background layer in your app's structure, or allow user to launch them from a content hotspot, menu or a Codice card.

You can add hotspots to an image to open additional content when a user taps on a specific part of the image. See section 11 to learn more about content hotspots.

7.2 Video viewers

Video viewer widget displays video files such as MP4 files, as well as video streams and video capture devices such as web cameras.



Video viewer widget. 1 *Video title.* 2 *Toolbar with pin, close and clear annotations buttons.* 3 *Video play button.*

Like Image viewers, you can add Video viewers to your app by dragging and dropping a video file from the media library to your app's structure or a content set. Or you can add a new Video viewer widget to your app and fill in the *Video source* attribute. This is required for video streams and video capture devices. See section 7.2.1 for details about video streams.

Video viewers have many of the same configuration options as Image viewers. You can set the video initial location and size, add a toolbar and decorations to the video and allow users to annotate on it. In addition, you can modify attributes specific to videos. You can set the video to automatically start playing when opened and to automatically close after playing or to play in a loop. By default, user can play or pause the video by tapping anywhere on it. Video also displays buttons to play or pause and to mute or unmute it. If needed, you can disable tapping to play the video and hide the buttons.

Video displays a preview image while it is not playing. See section 7.2.2 for information about customizing the video preview.

You can add hotspots to a video to open additional content when a user taps on a specific area of the video. See section 11 to learn more about content hotspots.

7.2.1 Video streams

Aside from displaying video files, Video viewer can also display live video stream. The stream can be a video capture device connected to the PC, such as a web camera. Or it can be an online video stream coming from another machine on the network.

To add a video stream to your app, follow these steps:

1. Click Structures  in the left-hand pane.
2. Click My First Structure.

   The *Editing a Structure* screen opens.
3. Go to the Main section and click *Add a new widget to Main*.
4. From the pop-up menu, add a **Video viewer**.
5. In the Video viewer attributes pane, set the following attributes:
   - **Video source**: Input the path to the video stream. This can be an online video stream, for example rtsp://10.36.0.122:8554/presentations. Or it can be a capture device:

     Ubuntu examples: /dev/video0

     /dev/v4l/by-path/pci-0000:04:00.0-video-index0.

     Windows example: video=@device_sw_{860BB310-5D01-11D0-BD3B-00A0C911CE86}\{BBABFA61-0001-48CC-9CAB-324ECB2D4C6B}

     Tip: *On Windows, if you also have MT Canvus installed, you can use mt-video-check.exe included with Canvus to get the video input source. Note that the output of mt-video-check.exe is optimized for Canvus. You should replace all double backslashes (\\) in the path with single backslashes (\) to use the path in Showcase Editor.*
   - **Streaming mode:** Normally video viewers buffer the decoded video to ensure smooth playback. This adds latency when the video is a live stream. For optimal experience, enable Streaming mode to skip this buffering.
   - **Start video automatically**: Enable this attribute to automatically start playing the video instead of waiting for user to play the video.
   - *(Optional)* **Display video controls** and **Single tap toggles playback:** You can disable these attributes to hide the video control buttons and prevent user from pausing the video stream.
   - *(Optional)* **Video stream quality settings***:* You can use the advanced attributes Video stream minimum resolution, Video stream maximum resolution, Video stream minimum frames per second, Video stream maximum frames per second and Prefer uncompressed video stream to fine-tune video stream resolution and quality. Click the help button for details.
5. Click Save to save your changes.

7.2.2 Video preview

While a video is not playing, it displays a preview image. By default, this is an automatically chosen frame in the video: the current frame if the video is paused, or a frame 5% from the beginning of the video if the video has played to the end and stopped. Note that when the video is first opened, it is paused on the first frame.

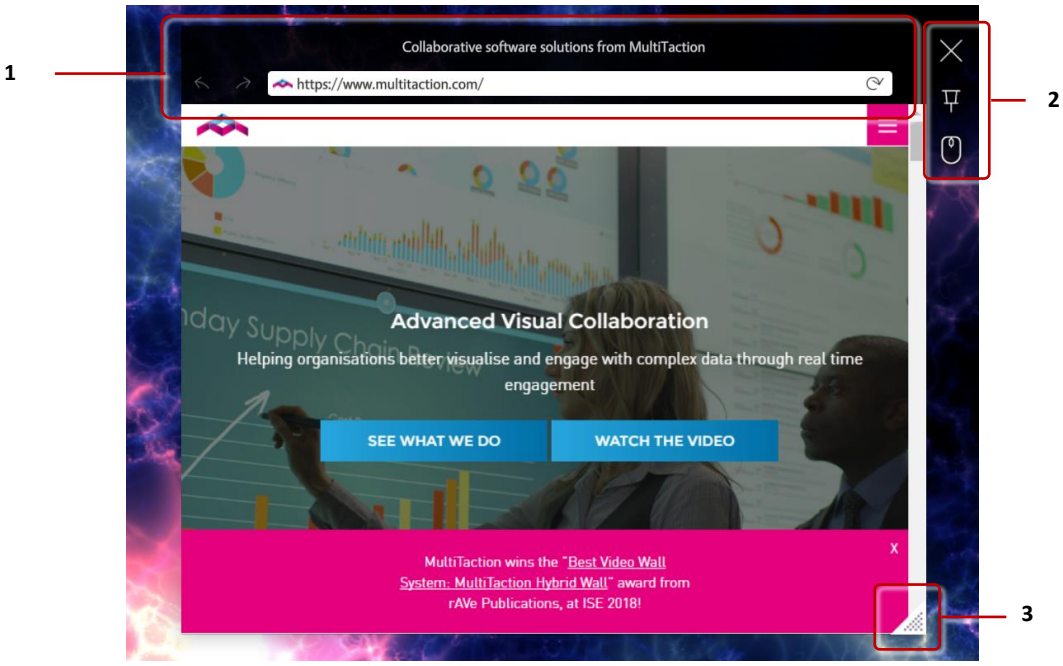
You can customize the preview image. You can set this to be a specific frame in the video, or a completely custom image. If a custom image is used, it is overlaid on top of the video, matching the video resolution.

To customize the preview image, follow these steps:

1. Select the Video viewer you want to edit.
2. In the Video viewer attributes pane, set the following attributes:
   - **Preview position:** Set the playback position in the video to use as a preview frame. Set the preview position as seconds. Or you can set a relative position as a percentage of the video duration by using calc, for example *calc(100%)* to use the last frame of the video as the preview image.
   - **Preview image**: You can use this attribute to set a custom image as the video preview. Drag and drop an image file from the media library into the input field.
   - *(Optional)* **Preview on pause:** Normally when video is paused, the current frame is used as the preview instead of the custom preview position or image. You can enable this attribute to display the custom preview also when the video is paused.
3. Click Save to save your changes.

7.3 Web browsers

Web browser widget displays web content, such as web pages and HTML files. It can also display images, videos, text files and PDFs.



Web browser widget. 1 *Address bar.* 2 *Toolbar with close, pin and toggle mouse emulation mode buttons.* 3 *Resize handle.*

You can set the initial URL for browser in the MT Showcase Editor. User can also write a new URL in the browser's address bar. You can hide the address bar by enabling *Hide address bar and navigation controls?*

Browser must be *pinned* to interact with the web page content. You can add a toolbar with pin button to the browser to allow user to pin the browser or enable the *Pin the widget when it opens?* attribute to automatically pin the browser.

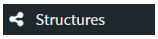
By default, browser operates in touch mode. But some web pages do not work correctly with multi touch gestures. You can enable *Emulate mouse* attribute to instead emulate mouse input. You can also add *Mouse emulation button* to the browser's toolbar to allow user to switch the mode as required.

7.3.1 Web browser cache

By default, all web browsers in Showcase use incognito mode. That is, each browser stores data such as sessions and cookies in a temporary cache, and when a user closes the browser that cache is deleted. But you can configure web browsers in Showcase to use a shared cache.

Why use a shared cache? You can use this feature to streamline the user experience. For example, you might have a company portal that requires a sign-in in your app. You can configure a shared cache for the browser to keep it signed in when the browser is closed and opened again.

For example, to add a browser cache to the web browser in My First Structure (see step 3 in section 3.3), follow these steps:

1. Edit the browser attributes:
   a. Click Structures 

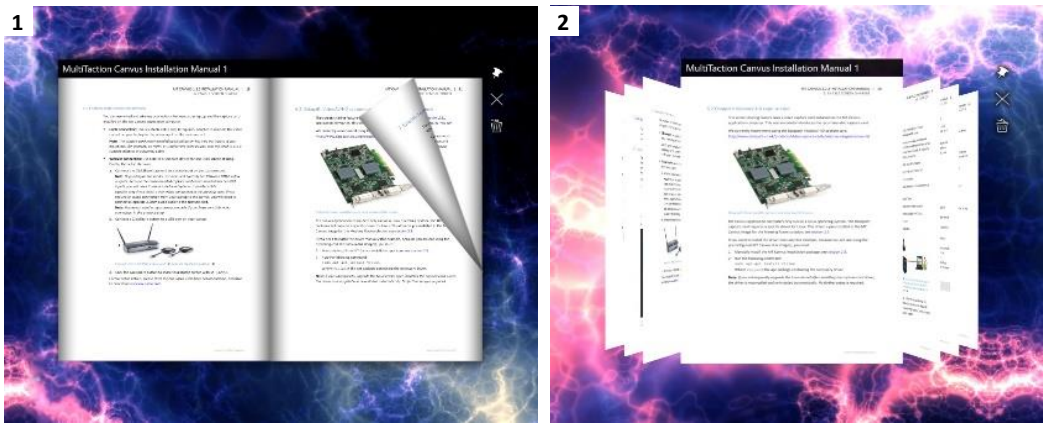
in the left-hand menu.
   b. Click the My First Structure hyperlink.
   c. In the Main section, click the Web browser widget.
2. Add a Browser cache.
   a. In the Web browser attributes pane, go to the **Browser cache** attribute.
   b. Click the New button.
   c. Type a name for the browser cache. For example, My Browser Cache.
   d. Click the Save button.
3. Edit the Browser cache.
   a. Click the Edit button to display the *Editing a widget* screen.
   b. In the *Editing a widget* screen, go to the right-hand attributes pane.
   c. Set the **Browser cache persistence**.

      This attribute determines how long the cache is persisted. If you choose **Session**, Showcase stores the cache in memory. When Showcase is shut down, the cache is cleared. If you choose **Persisted**, Showcase stores the cache on disk. The cache persists indefinitely and is loaded from disk whenever Showcase is launched.
   d. Click the Save button.

Note: *You have now created a new* Browser cache *shared widget. This widget is now listed in the widget library (see section 2.2.4) and can be assigned to any other browser in your app. All browsers using the same* Browser cache *share data. You can also set a default* Browser cache *in a theme.*

7.4 PDF books and flows

PDF book displays PDF documents as an open book. PDF flow displays PDF documents as a stack of pages.



PDF viewers. **1** *PDF book.* **2** *PDF flow.*

You can add PDF viewer widgets to your app by dragging and dropping a PDF file from the media library to your app's structure or a content set. Or you can add a new PDF book or PDF flow widget to your app and drag and drop a PDF file from the media library to *Book pdf* or *Source pdf* attribute.

PDF viewer widgets have many of the same configuration options as Image and Video viewers. You can set the widget initial location and size, add a toolbar and decorations to the widget and allow users to annotate on it.

Both book and flow must be *pinned* to turn the PDF document pages. You can add a toolbar with pin button to the widget to allow user to pin the widget or enable the *Pin the widget when it opens?* attribute to automatically pin the widget.

Book and flow also have buttons to turn to the previous or next page, and a scroll bar to quickly scroll through the PDF document. Tap anywhere on the widget to display the buttons, they are automatically hidden when the widget has no interaction. You do not need to pin the widget to use the buttons and scroll bar. You can disable displaying the buttons with the *Display controls* attribute.

7.5 Image movies

Image movie widget displays sequences of image files, such as PNG, JPEG, BMP and TIF files.

You can add a new Image movie widget to your app by dragging and dropping a folder from the media library to a structure or content set in the MT Showcase Editor. Alternatively, you can create a new empty Image movie widget in a structure and drag and drop the folder into the *Image sequence folder* attribute. For more information about the media library, see section 2.7.

Image movie widgets have many of the same general configuration options as Image viewers. You can set the widget initial location and size, add a toolbar and decorations to the widget and allow users to annotate on it.

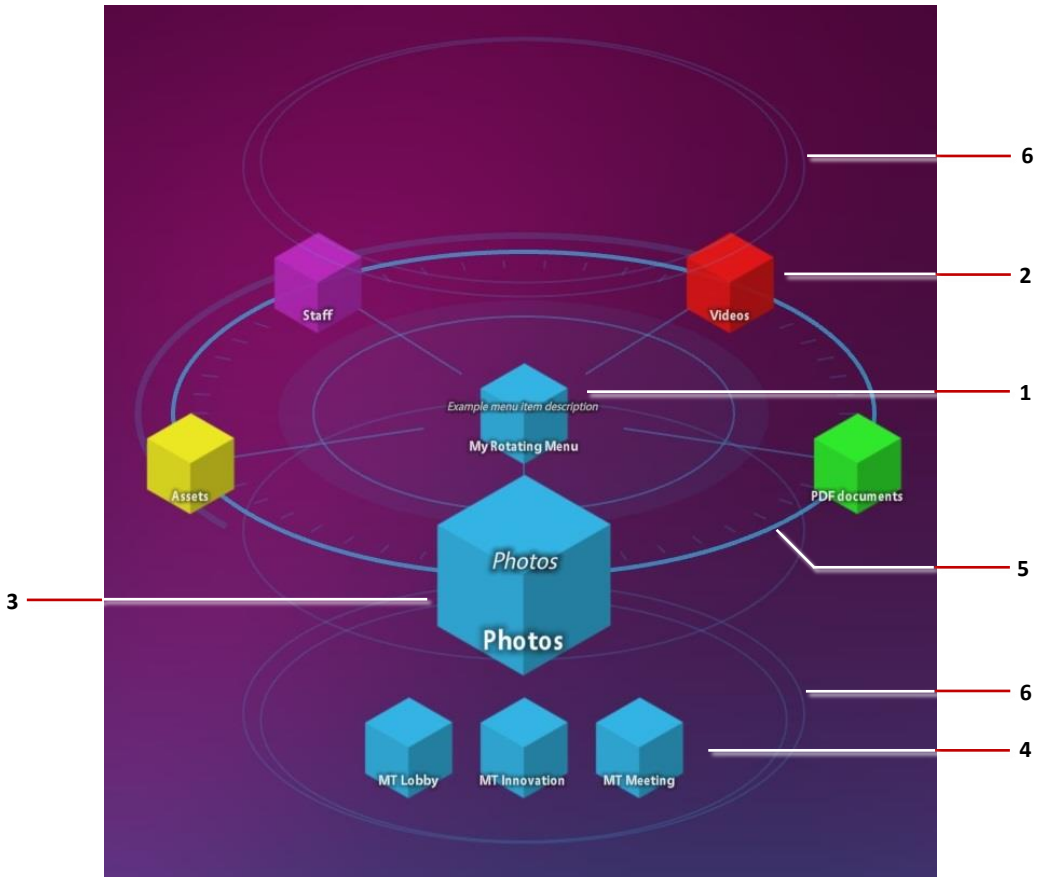
In addition, you can control how the movie is played. You can set the playback speed and choose a playback mode, such as play the movie in a constant loop, or play the movie once and then automatically close the widget. You can also set the movie in an interactive mode, where the movie doesn't play automatically but the user can scroll through the images. You can do this by setting *Playback speed* attribute to 0 and *Interactive mode* attribute to 1. The image movie must be *pinned* before user can scroll through the images.

8 Create a rotating menu

Rotating menus are oval animated menus, with items orbiting a central node. The effect is similar to satellites orbiting a planet. Users can tap an orbiting item to open a submenu of items such as images, videos and PDFs.

Rotating menus display when a user taps and holds an empty area of the screen. They can also be launched when a user taps a content hotspot or holds a Codice marker against the screen. Rotating menus are a direct alternative to finger menus.

By default, the central node, orbiting items and submenu items are all shown as bubbles, like the ones in your finger menu. If you set the *menu item type* to 'nodes', they display as blue cubes but you can specify custom cube colors or custom images.



Example rotating menu.

1 *Central node. Users can drag the central node to move the menu.* 2 *Orbiting items, or 'orbiters'. These orbit the central node. Tap any orbiting item to open a submenu.* 3 *As orbiting items approach the user, they get bigger for emphasis.*

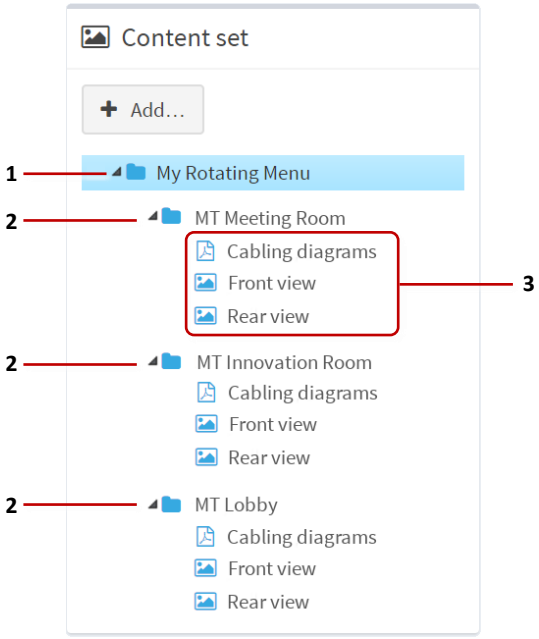
4 *Submenu items. These contain the actual menu content such as images, videos, and PDFs. By default, submenus display below the rotating menu (shown here), but you can display them on the left or right or above the menu.*

5 *Rings and spokes spin slowly around the central node.* 6 *Halo rings above and below the rotating menu gently rise and fall. You can customize the color and speed of these rings.*

8.1 Set up a content set

Before creating a rotating menu, you must create the content set that supplies the menu with content such as images, videos or PDFs.

Unlike finger menus and teaser menus, content sets for a rotating menu *must* adhere to a specific three-tier format. Specifically, the actual menu items—images, videos or PDFs—must be in subfolders immediately below the root-level folder. You cannot add content directly to the root-level folder.



Example content set for rotating menu.

1 *Root level folder. This folder is the central node in the rotating menu.*

2 *Subfolders. These subfolders correspond to items orbiting the central node.*

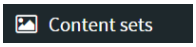
3 *Submenu items. These items are displayed in a submenu when a user taps an orbiting item.*

Follow these steps:

1. If you have not already done so, drag the content you want into the media library. You can then populate the content set with items from your media library. See step 1 in section 5.1.

2. Create a new content set.

   You can adapt an existing content set to meet the required format, but you may find it easier to simply create a new one specifically for rotating menus:

   Click Content sets 

in the left-hand pane.

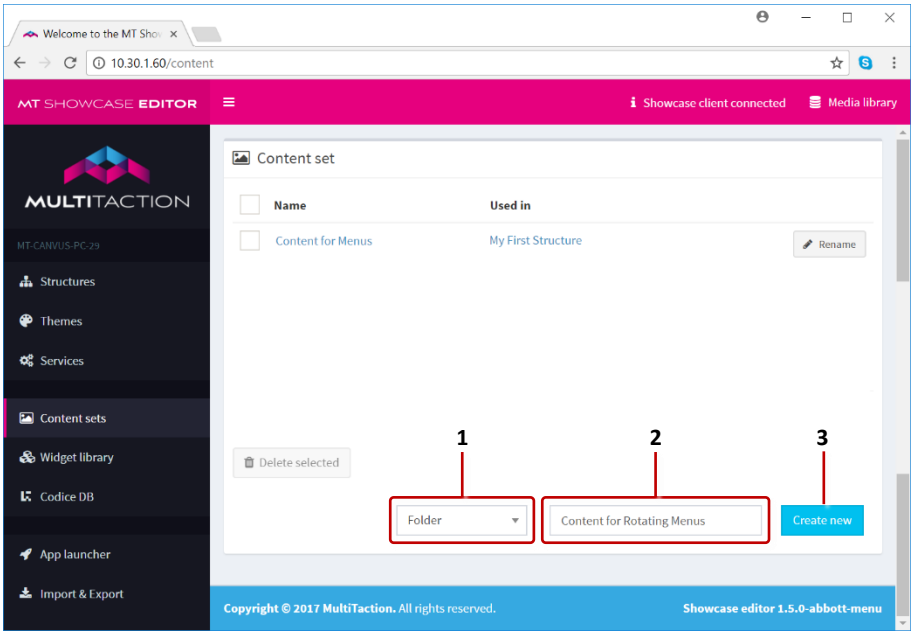
3. In the *Content Set* screen:

   a. Select 'Folder' from the drop-down menu at the bottom of the screen.

      This menu sets the node type (folder, image, widget, and so on) for the root node of the new content set.

b. In the name box, enter a name of the new content set. For example, Content for Rotating Menus.

c. Click the Create New button

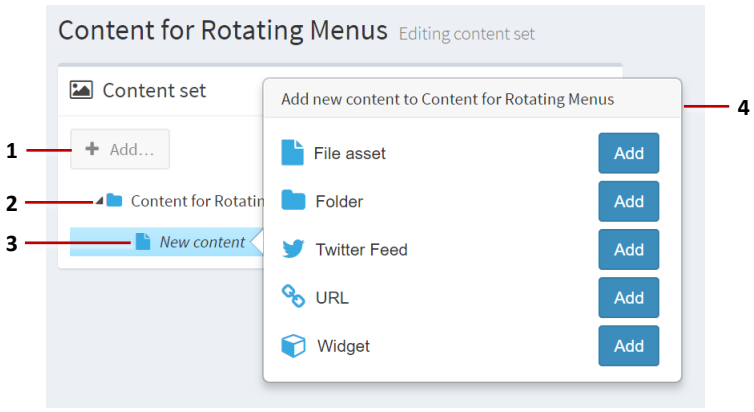


Content set screen. 1 *Drop-down menu for new content set's root node.* 2 *Name of new content set.* 3 *Create New button.*

4. Add the first orbiting item.

In the Editing a Content Set screen, select the root node. Then click the Add button.

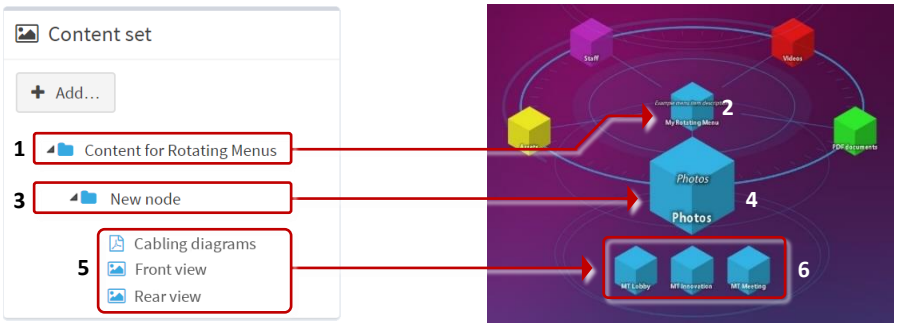
When the pop-up menu appears, add a folder. This folder is added as subfolder below the root node. In the rotating menu, this subfolder will be the first orbiter.



1 *Add button.* 2 *Content set root node.* 3 *New folder added as a subfolder below the root node.* 4 *Pop-up menu.*

5. Now add the submenu items for this orbiter.

   To do this, drag items from the media library into the subfolder you just created. (This action is also described in step 3, in section 5.1.) In the example below, the three items will form the submenu for the first orbiter:



Content set three-tier format:
The root folder of the content set (**1**) *maps to the menu's central node* (**2**).
The subfolder (**3**) *maps to an orbiter in the rotating menu* (**4**). *Note that the orbiter gets bigger as it approaches the user.*
Items in the subfolder (**5**) *appear as the items in the submenu* (**6**).

6. Repeat step 4 for each orbiter in your rotating menu. The example on page 67 shows subfolders, equating to three orbiters in the menu.

7. Add submenu items for each for each orbiter. That is, repeat step 5 for each subfolder you added to the content set.

8. *(Optional)* For greater visual impact, you can set matching colors for each orbiter and its submenu. In the screenshot above, the 'Photos' orbiter and its submenu are both blue. Follow these steps:

   a. First, set the orbiter color. Select the subfolder you want and edit the **Menu item color** attributes in the content set.

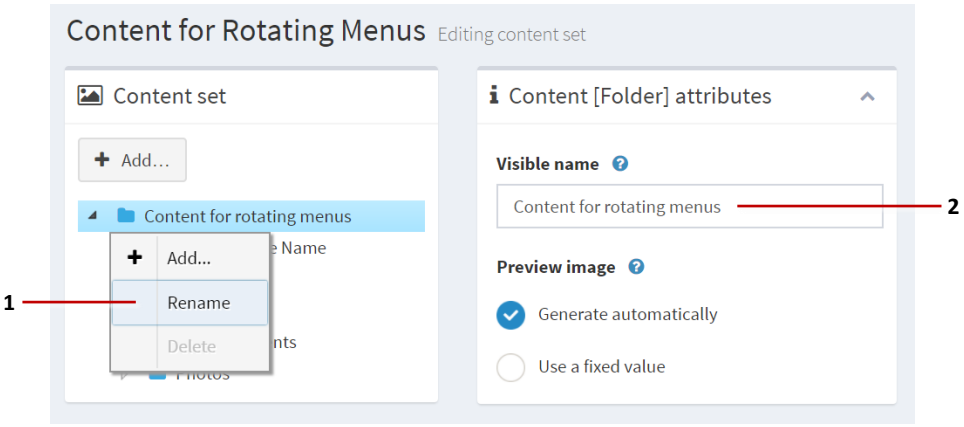
   b. Now set the color for the submenu items. Select each subfolder item and set the same **Menu item color** as for the subfolder. To do this quickly, you can copy and paste the rgba color value from the subfolder to the subfolder items.

   For guidance on specifying colors, see section 21.4.

9. Now edit the labels for the following menu items:

   - **Central node:** The root folder name in the content set appears as the central node name in the rotating menu. By default, the root folder name is the name of the content set. To edit this name, select right-click the root folder and choose Rename. Or edit the **Visible name** attribute in the right-hand *Content [Folder] attributes* pane.
   - **Orbiters:** Subfolder names in the content set appear as orbiter labels in the rotating menu. By default, each new subfolder is named New node. To edit these names, right-click the submenu and choose Rename. Or edit the **Visible name** attribute in the right-hand *Content [Folder] attributes* pane.
   - **Submenu items:** Item names in the content set appear as submenu labels in the rotating menu. By default, an item's name is simply its file name without the

filename extension. For example, Cabling diagram.pdf appears as Cabling diagram in both the content set and rotating menu. To change a submenu item's label, right-click the item in the content set and choose Rename. Or edit the **Visible name** attribute in the right-hand *Content [File asset] attributes* pane.



Renaming items in a content set. Right-click an item and choose Rename (**1**) *or edit the Visible name attribute* (**2**).

10. *(Optional)* Add brief descriptions for the orbiters and, if required, the central node. To add descriptions, edit the **Menu item description** attribute in the right-hand *Content [Folder] attributes* pane.

    Orbiter descriptions display when you open an orbiter's submenu. By default, these display below the rotating menu, but you can adjust this by editing the **Description Location** attribute. (see below).

    You can also add a description for the central node, but note this description is always visible. By default, it displays below the central node, potentially obscuring any orbiter descriptions. To remedy this, adjust the **Description Location** (see below).

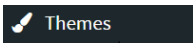
    Note: *The **Description Location** attribute adjusts where descriptions are displayed for the central node and orbiters. You can set a default Description Location in your theme (go to the **Rotating menu** component, then click the Show advanced attributes hyperlink). Or you can set a Description Location for individual folders and subfolders in the content set. Click the ? help button if you need details.*

11. Click the Save button to save the changes to the content set.

8.2 Set up rotating menus

In this section, you define the basic appearance and behavior of rotating menus. The easiest way to do this is to use a theme. In this section, you will edit the theme you created in section 6.

Follow these steps:

1. Click 

in the left-hand pane.
2. In the Themes screen, click the theme you created previously, My First Theme.
3. In the *Editing a theme* screen, go to the **Rotating menu** component.

   Tip: *Click the hyperlink in the Components pane.*
4. Set the **menu item type**. This attribute determines what the menu items in your rotating menu look like. By default, menu items display as *bubbles*, but you can alternatively configure them to display as *nodes* (cubes or custom images).

   This important step is described in detail in section 8.3.
5. If required, customize the rotating menu's behavior and appearance.

   For example, the **Menu color** attribute lets you specify the color of the menu rings. The **submenu location** attribute determines whether submenus open above or below, or to the left or right, of the rotating menu.

   You can also adjust the location of submenu items (for example, images or videos). By default, these items open directly over the submenu, obscuring other available items. To counter this, you can edit the **Content location** and **Use relative location** attributes.

   In the advanced attributes, the **Menu rotating animation duration** and **Menu floating animation duration** attributes let you adjust how fast the menu rings spin and how slowly the halo rings rise and fall. Finally, the **Allow moving**, **Allow rotating** and **Allow scaling** attributes control whether uses can move, change the angle, or resize rotating menus.

   In all cases, click the  button to see the attribute help.
6. Click Save to save the changes to your theme.

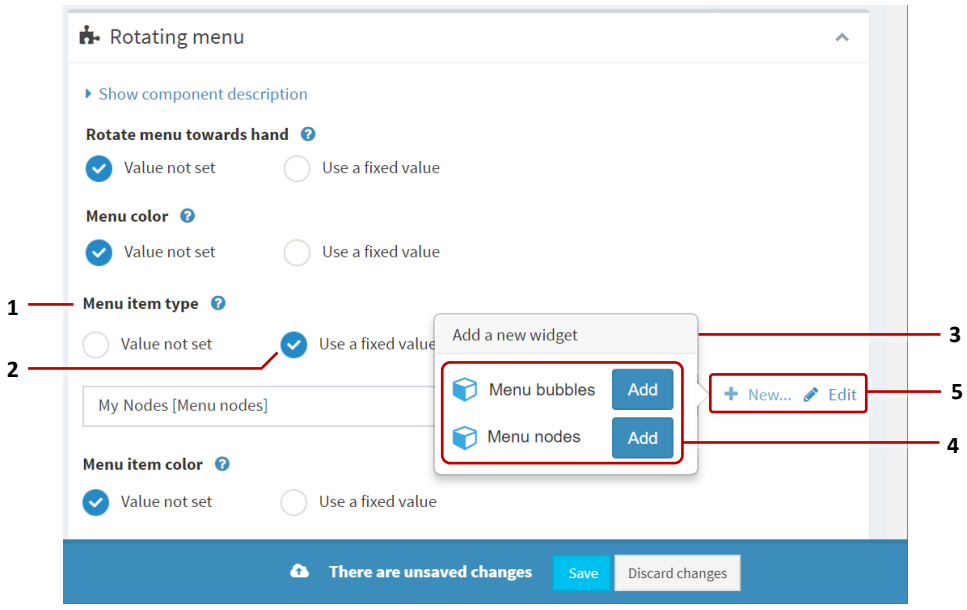
8.3 Configure the rotating menu to use nodes

This task is optional.

By default, items in a rotating menu display as *bubbles* (you practiced configuring bubbles in section 5.5). However, you can also configure menu items to display as *nodes* (cubes or custom images). First, you must configure your rotating menu to use nodes as the *menu type*. Then you will need to create a Menu Nodes widget. The easiest way to perform these tasks, is to use a theme. As in the previous section, you will edit the theme you created in section 6.

Follow these steps:

1. Click Themes in the left-hand pane.
2. In the Themes screen, click the theme you created previously, My First Theme.
3. In the *Editing a theme* screen, go to the **Rotating menu** component.
4. Now create a *Menu nodes* widget for rotating menus.
   a. Go to the **Menu item type** attribute.
   b. Select 'Use a fixed value' and then click the New button.
   c. In the *Add new widget pop-up*, add the *Menu nodes* widget.
   d. Enter a name eg, My Nodes.
   e. Click Save to add the new Menu Nodes widget to rotating menus in your theme.

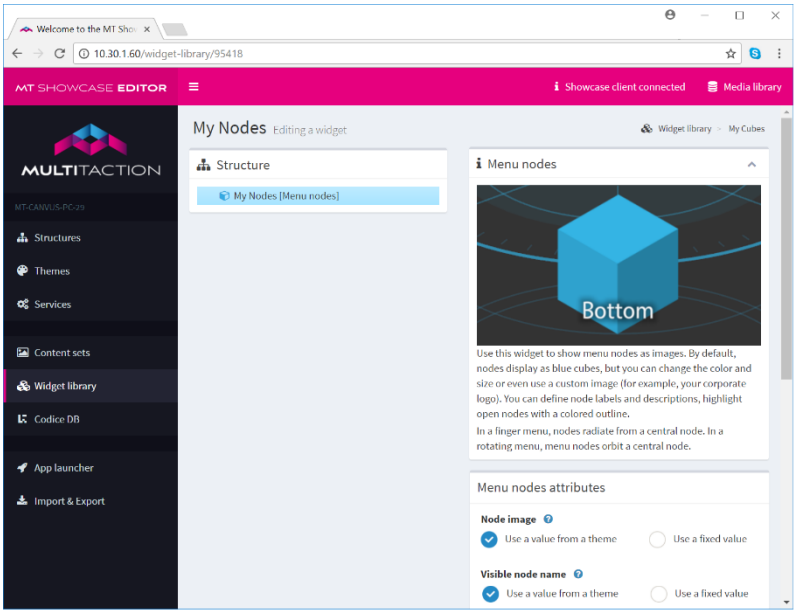


Editing a theme screen, Rotating menu component. **1** *'Menu item type' attribute.* **2** *'Use a fixed value'.* **3** *Add a new widget pop-up.* **4** *Available menu node types.* **5** *New and Edit buttons.*

5. Now configure the new Menu Nodes widget.
   a. Still in the *Editing a theme* screen, go to the **Menu item type** attribute
   b. Select My Nodes from the drop-down menu and click the Edit button.
   c. In the *Editing a widget* screen, edit the node attributes as required.

      By default, nodes display as blue cubes. You can change the default cube color or size. You can also define an overlay (used to highlight menu items with a colored outline). You can even specify a different image instead of the default cube image. In all cases, click the ? button to see the attribute help.

      For example, to specify gold cubes, set the **Menu item color** attribute to #FFD700 or rgba(255,215,0,1) or gold. See section 21.4 for details of color input methods.
   d. Click Save to save the changes to the Menu Nodes widget.
6. Click the Back button in your browser to return to the *Editing a theme* screen.
7. Click Save to save the changes to your theme.



Editing a widget screen, Menu Node widget

Tip: *This section showed how to set up a Menu Nodes widget with gold cubes. If you want different colored orbiters and submenus in your rotating menu, you can simply edit the* ***Menu item color*** *attribute for each subfolder and subfolder item in your content set; see step 8 in section 8.1.*

8.4 Add a rotating menu to your app's structure

Now you need to add a rotating menu to your app's structure. Then you need to configure the menu to use the new content set, Content for Rotating Menus.

Follow these steps:

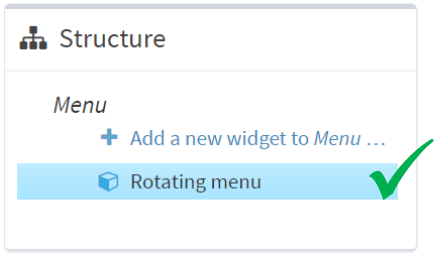
1. Click Structures 

in the left-hand menu.
2. Click the structure you want to use. Or create a new structure; see step 1 in section 3.3.

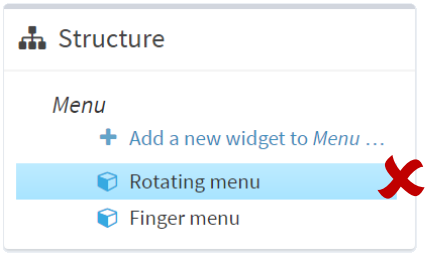
   Note: *You may prefer to create a new structure instead of using an existing one. This is because only one menu can be active in an app's Menu layer; see step 6.*
3. In the *Editing a Structure* screen, go to the Menu layer.
4. Click Add a new widget to Menu.
5. From the pop-up menu, add a **Rotating menu**.

   A preview image of a rotating menu now displays in the right-hand pane.
6. *(Applies only if your structure already has a finger menu defined)* You must delete the existing finger menu before adding a rotating menu.

   This is because the first-defined menu (chronologically) is always the active menu. It is not sufficient to simply re-order the menus so that the rotating menu is topmost.



Editing a structure screen, Menu layer. Correct setup. Only one menu is defined.



Editing a structure screen, Menu layer. Incorrect setup. The menu that was added first, regardless of display order, is always the active menu.

7. In the *Rotating menu attributes* pane, select Content for Rotating Menus from the **Content Set** dropdown menu.
8. Click the Save button to add the menu to your structure.

8.5 Add a theme to your app

Although themes are optional, we strongly recommend that your app uses a theme to define the basic appearance and behavior of your rotating menu. If you have not already added My First Theme to your app, instructions for doing so are in section 6.4.

8.6 Test the rotating menu

You have now updated the structure to use your new rotating menu. To test the menu:

1. Restart your app.
   a. Go to the *App launcher* screen.
   b. Click the Restart button for the app containing your rotating menu.
2. Tap and hold any empty area of the screen. After a brief delay, the menu opens.

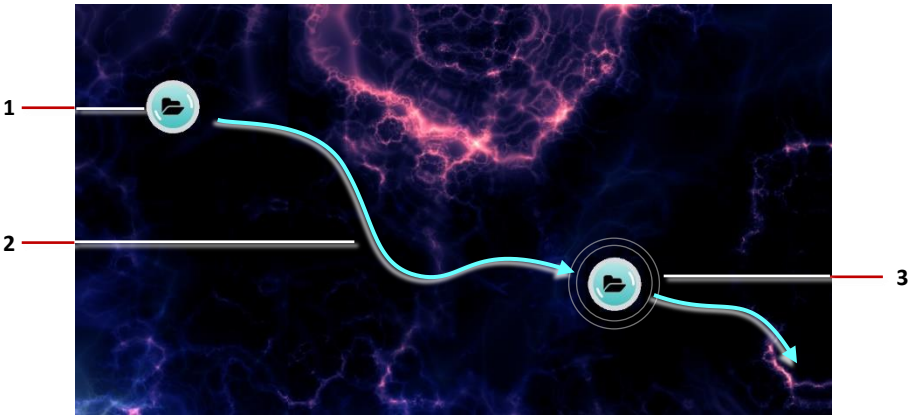
9 Create a teaser

This section describes how to set up a teaser.

A teaser is a menu launcher. It can be a single bubble drifting slowly across the screen, 'wobbling' briefly at regular intervals to attract users' attention, or a single bubble that fades in and out, reappearing at random screen locations. In both cases, users can tap the teaser to open a regular finger menu or teaser menu.

For greater impact, you can specify multiple instances of a teaser drifting across the screen at the same time, and you can use a custom image or video instead of a bubble.

For this example, the teaser menu will use some of the same components that you set up earlier ie, My First Structure and the My First Connectors shared widget. However, to give the teaser a unique appearance, you will need to create a new content set and two shared widgets, My Teaser Bubble and Teaser menu.



Example teaser floating across the screen.

1 *Teaser bubble. If a user taps this bubble, a regular finger menu opens.*

2 *Teaser drifts slowly across the screen. The general direction and degree of undulation are configurable.*

3 *At regular intervals, a teaser animation plays to get the users' attention. In effect, the teaser appears to briefly wobble or pulse.*

9.1 Set up a content set

As with finger menus, teasers require a content set to supply them with media such as images. For example, the teaser can re-use the content set (Content for Menus) that you set up in section 5.1. This will mean that the teaser inherits the same appearance as the central bubble in the finger menu you created previously. If this is OK, you can go directly to section 9.2.

But what if you wanted your teaser to have its own unique appearance? The solution is to create a unique content set for your teaser menu. Follow these steps:

1. Drag some content into the media library from Windows Explorer. For example, drag a folder that contains images.

   Tip: *Click the* Media library *button to open and close the media library pane.*

2. Click Content sets in the left-hand pane.

3. Create a new content set. For example, Content for Teaser.

   For more detailed instructions, see section 5.1.

4. Now add items to the new content set. In the *Editing a content set* screen:

   a. Drag items from the media library into the new content set.

   b. Close the media library and save the changes.

5. (Optional) Specify a thumbnail image for the teaser bubble.

   a. Still in the *Editing a content set* screen, click the root folder in your content set (labelled Content for Teaser).

   The root folder represents the teaser bubble.

   b. Go to the **Preview image** attribute and select 'Use a fixed value'.

   c. Click Media library then drag an image from the media library into Preview image field.

6. Now create a *Menu bubbles* widget for the new content set.

   a. In the *Widget attributes* pane, go to the Finger menu tab.

   b. Go to the **Bubble type** attribute.

   c. Select 'Use a fixed value' and then click the New button.

   d. Add the *Menu Bubbles* widget.

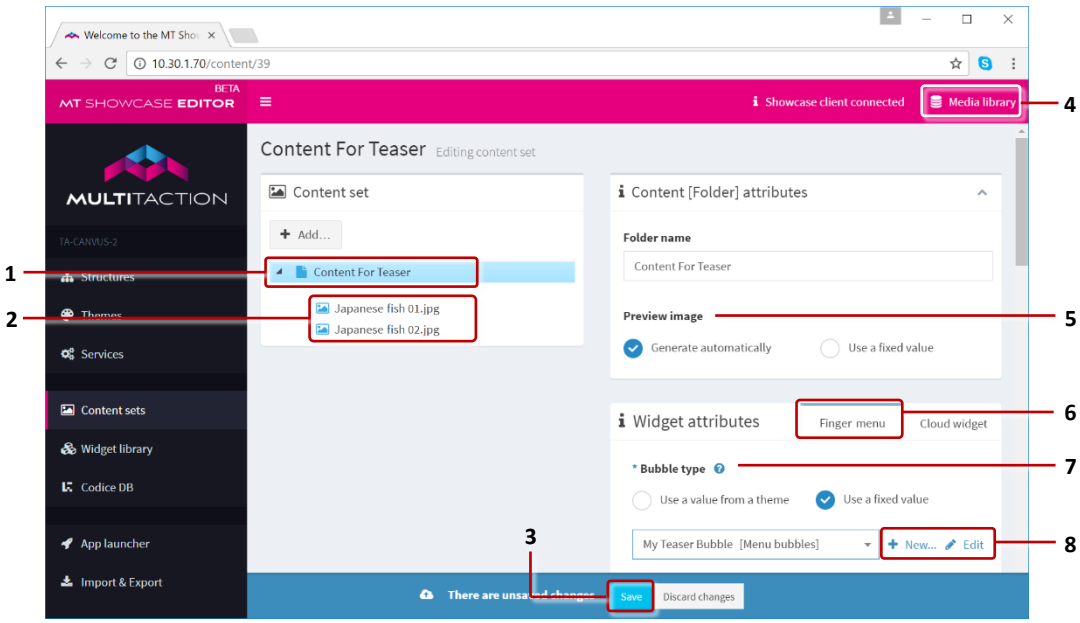
   e. Enter a name eg, My Teaser Bubble.

   f. Click Save.

   For more detailed instructions, see step 4 of section 5.4.

7. Now configure the teaser bubble.
   a. Still in the *Editing a content set* screen, go to the Bubble type attribute
   b. Select My Teaser Bubble from the drop-down menu and click the Edit button.
   c. Edit the bubble attributes as required.
      For example, edit the **Bubble size** and **Bubble edge color** attributes to change the teaser bubble size and color. (See section 21.4 for color input methods.)
   d. Click Save.

   For more detailed instructions, see section 5.5.



'Editing a content set' screen.
1 *Root folder of teaser content set.* **2** *Items in the teaser content set.* **3** *Save button.*
4 *Media library button.* **5** *Preview image attribute.* **6** *Finger menu tab.* **7** *Bubble type attribute.*
8 *New and Edit buttons.*

9.2 Add a teaser to your app's structure

Now you need to add a teaser to your app's structure. Then you need to configure the teaser to use a content set.

For this example, you will add a teaser to My First Structure; see section 3.3.

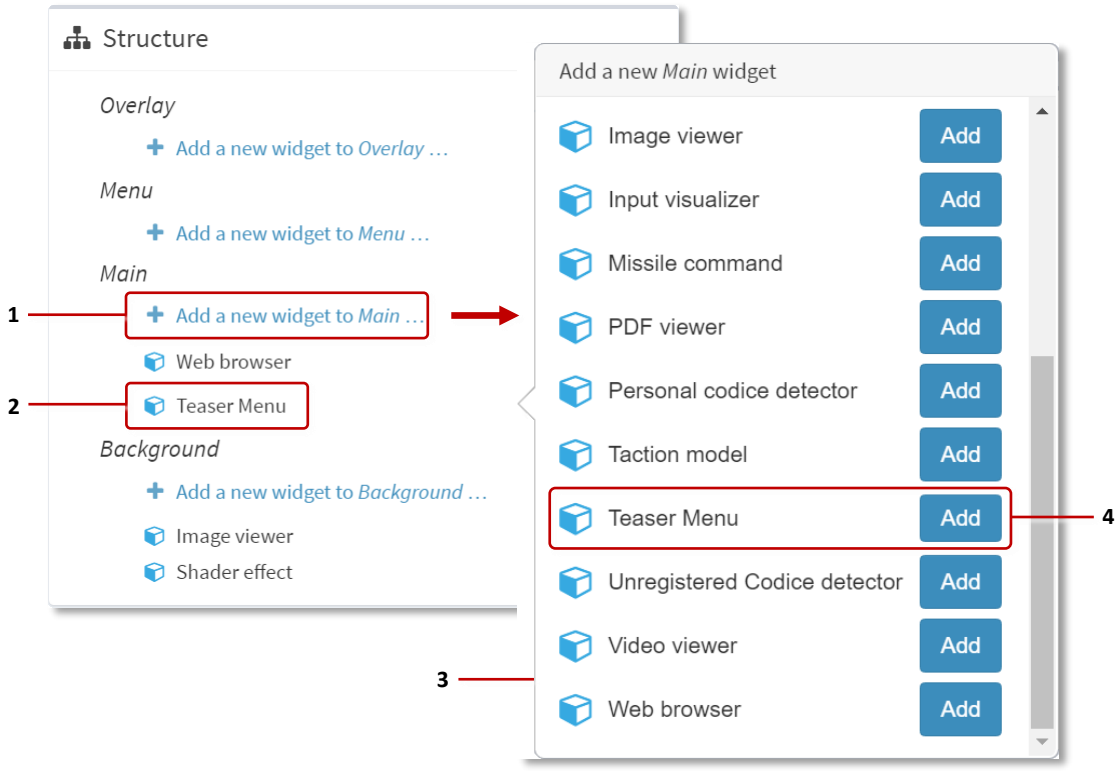
Follow these steps:

1. Click Structures Widget structures in the left-hand pane.
2. Click My First Structure.

   The *Editing a Structure* screen opens.
3. Go to the Main section and click *Add a new widget to Menu*.
4. From the pop-up menu, add a **Teaser menu**.

   See the screenshot below.
5. Click the Save button to add the teaser to My First Structure.

Now you need to set up the teaser. Go to section 9.3.



'Editing a content set' screen, widget attributes pane.
1 *'Add a new widget to Main' hyperlink.* **2** *Teaser menu widget added to Main layer.*
3 *Add a new Main widget popup.* **4** *Teaser menu widget and Add button.*

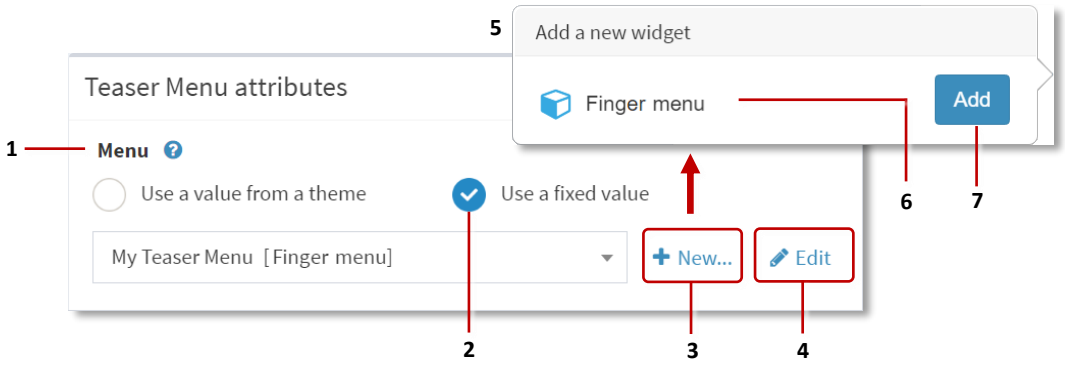
9.3 Set up the teaser

In this section, you will edit the essential attributes for the teaser. As part of this procedure, you will create two shared widgets, a *Finger menu* and a *Teaser operator*. You will also assign a content set to the teaser.

Note: *A teaser operator controls how a teaser floats across the screen, including its speed, direction, and animation properties. If you do not specify a teaser operator, the teaser will use default values (for example, it will slowly float diagonally downwards, left to right).*

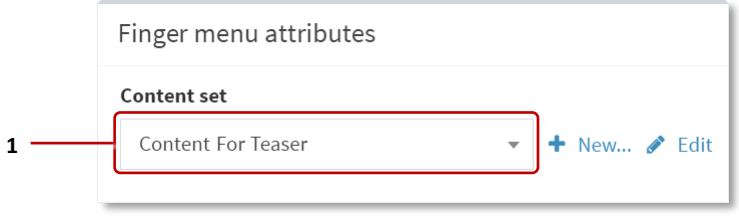
Follow these steps:

1. In the Editing a Structure screen, click the new Teaser menu widget.
2. In the *Teaser menu attributes* pane, go to the **Menu** attribute.
   a. Select 'Use a fixed value' and click the New button.
   b. Add the Finger menu widget.
   c. Enter a name eg, My Teaser menu.
   d. Click Save.



'Editing a structure set' screen, widget attributes pane
1 *Menu attribute.* 2 *Use a fixed value check box.* 3 *New button.* 4 *Edit button.*
5 *Add a new widget popup.* 6 *Finger menu widget.* 7 *Add button.*

3. Now assign a content set to My Teaser menu.
   a. Still in the *Teaser menu attributes* pane, stay with the **Menu** attribute but now click the Edit button (see screenshot above).
   b. In the *Editing a widget* screen, go to the Finger menu attributes pane.
   c. Go to the **Content Set** attribute and select the content set you created earlier (Content for Teaser) from the drop-down menu.
   d. Click Save and go back to the *Editing a Structure* screen.



'Editing a widget' screen, Finger menu attributes. 1 *Drop-down menu for Content Set attribute*

4. Now set the teaser behavior. Two behaviors are supported:
   - **Floating content behavior:** The teaser floats slowly across the screen, animating briefly at regular intervals.
   - **Fading content behavior:** The teaser fades in and out rhythmically, appearing at random screen locations.

   Follow these steps:

   a. Go back from the *Editing a widget* screen to the *Editing a Structure* screen.
   b. In the Teaser menu attributes pane, go to the **Teaser behavior** attribute.
   c. Select 'Use a fixed value' and click the New button.
   d. Add Floating content behavior or Fading content behavior.
   e. Enter a name. For example, My Teaser Behavior.
   f. Click Save.

5. (Optional) Finally, you can specify a *visualization widget*, replacing the traditional teaser bubble with a custom image or video.

   By default, a teaser is a single bubble, but you can instead specify an image, image movie or video. For example, you could use your logo as the teaser. When the user taps the image or video, the teaser menu launches as normal. Click New to add an image or video widget. Then save and click Edit to define the widget properties.

   Still in the Teaser menu attributes pane:

   a. Go to the **Teaser visualization widget** attribute.
   b. Select 'Use a fixed value'. Then click New.
   c. Enter a name eg, My Teaser Image. Then click Save.
   d. Still in the **Teaser visualization widget** attribute, click Edit.

      The *Editing a widget* screen appears.
   e. Now specify which image or video to use. Do this in the *Editing a widget* screen. If you added the:
      - **Image Viewer** widget, edit the **Image file** attribute.
      - **Image Movie** widget, edit the **Image sequence folder** attribute.
      - **Video Viewer** widget, edit the **Video source** attribute.

      For images and image movies, you do not generally need to change any other attributes (although you may want to change the size or playback speed).

      For videos, we strongly recommend you enable the **Start video automatically**, **Loop video** and **Mute video** attributes.

      Click the help button if you need details.
   f. Click Save to save changes to the Teaser visualization widget.

This completes the setup. You have now created the basic components of a teaser. Before you customize it further, you need to test it. Go to section 9.4.

Note: *In this section, you created a new Finger menu shared widget and either a Fading content behavior or Floating content behavior shared widget. These widgets are now listed in the widget library; see section 2.2.4.*

9.4 Test the teaser

You have now updated the structure to use your new teaser. To test the teaser:

1. Go to the *App launcher* screen.
2. Click the Restart button for My First App.
3. The teaser appears automatically and starts floating across the screen. When you tap the teaser, My Teaser menu opens (a regular finger menu).

Now you can customize the appearance and behavior of the teaser. Go to section 9.5.

9.5 Next steps

The previous sections described the minimum steps necessary to add a teaser to your app. But the MT Showcase Editor offers a rich selection of options to customize the teaser appearance and behavior.

9.5.1 Customize the general teaser attributes

You rarely need to change the general teaser attributes. However, there is one general attribute that you may want to edit. The **How many instances...** attribute lets you add multiple instances of the same teaser menu to your app for greater visual impact.

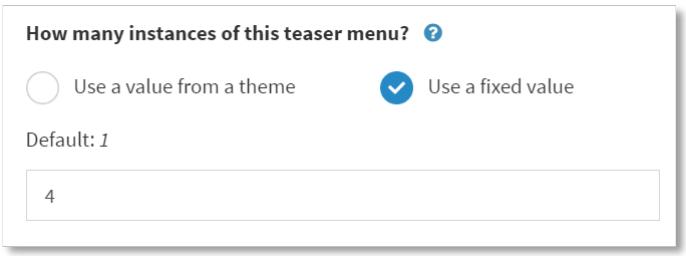
Other general teaser attributes let you customize the size and location of the 'teaser spawning area' and specify whether the teaser menu is open when it first appears on screen (by default, it is closed).

Follow these steps:

1. In the *Editing a Structure* screen, click the new **Teaser menu** widget.
2. In the *Teaser menu attributes* pane, you have already specified the **Menu** and **Teaser behavior** attributes. Now explore the other available attributes.

   For example, set the **How many instances...** attribute to 4 to create four instances of the same teaser menu to your app.

   Note: *You can also create multiple* different *teaser menus in your app ie, each teaser opens a different menu and, optionally, has a different appearance and behavior.*
3. Go back to the *Editing a Structure* screen.



Teaser menu attributes pane. 'How many instances of this teaser menu?' attribute

9.5.2 Customize the teaser menu behavior

Now explore the attributes available for the *Teaser behavior* shared widget.

For example, if you chose the **Floating content behavior** in step 4 of section 9.3, you can customize how the teaser floats across the screen. You can change its general direction, speed, and animation properties. You can also control how it undulates as it moves.

Alternatively, if you chose the **Fading content behavior**, you can customize how often the teaser appears and disappears, whether it fades in and out, and expands or shrinks, as it appears and disappears.

Follow these steps:

1. In the *Editing a Structure* screen, click the new **Teaser menu** widget.
2. Still in the *Teaser menu attributes* pane, go to the **Teaser behavior** attribute.
3. Click the Edit button for My Teaser Behavior.
4. In the *Editing a widget screen*, go to the right-hand attributes pane and explore the available attributes. If, in step 4 of section 9.3, you chose:
   - The **Floating content behavior**, edit the **Direction of content movement** attribute. This sets the ratio of horizontal to vertical movement of the teaser menu as it floats across the screen. For example, enter "4 1" to gently float right and downwards (four units to the right and one unit down). Or use negative numbers to float left and upwards. Similarly, edit the **Horizontal** and **Vertical distance of content waves** attributes to control the horizontal and vertical distance of the undulations (or waves) as the teaser drifts across the screen.
   - The **Fading content behavior**, edit the **Time between animations** attribute to set how often the teaser appears and disappears. If required, you can also edit **Enable fade animation** and **Enable scale animation** to prevent the teaser fading in and out or expanding and shrinking. (These attributes are enabled by default.)

Note: *You can apply the Fading content behavior* and *Floating content behavior shared widgets to* ***animated content****. That is, you can set up images or videos to drift across the screen, or fade in and out, in the same way as a teaser menu. For details about animated content, see section 25.2.*

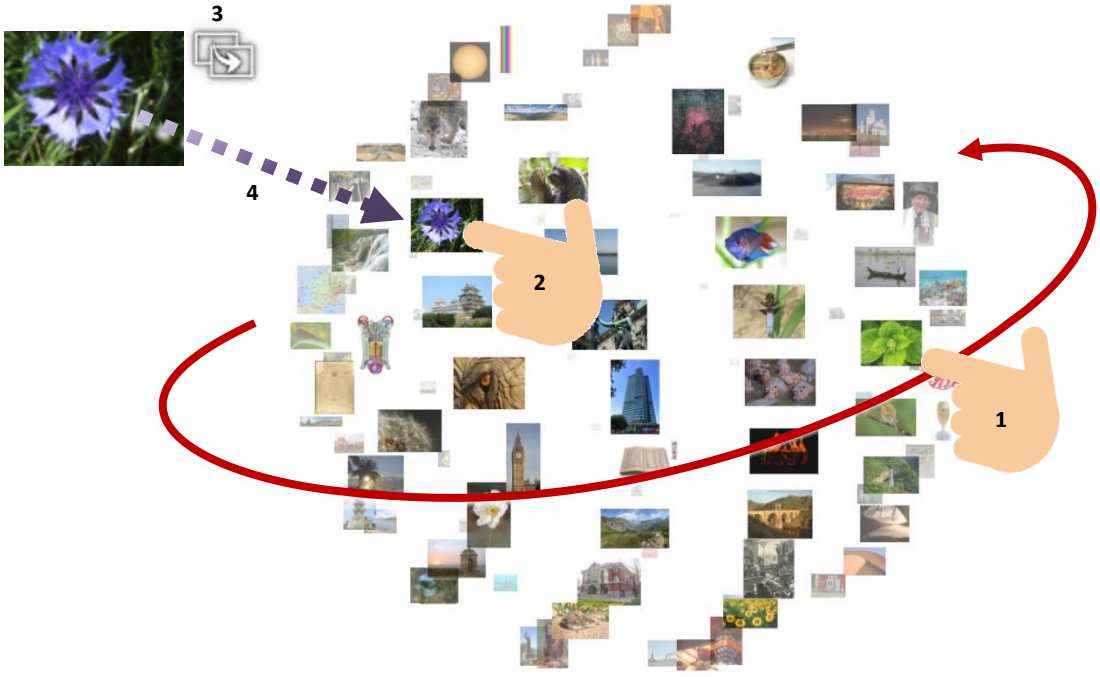
10 Create a cloud

The cloud is arguably the most striking widget available in MT Showcase. It displays a rotating collection of items (previews of images, movies, browsers, or tweets), clustered together in a ball. Users can rotate and resize the cloud to find the item they want.

When they see an item of interest, the user can tap it to detach it from the cloud. The user can then move and resize the item in the normal way. Note that the cloud is 'self-restoring'. After a period of inactivity, the detached item automatically drifts back to rejoin the cloud.

Note that items in the cloud are in fact *previews*. To see a full version of any item, you must tap the Open button (this button displays when you tap an item in the cloud). When the full version opens, you can interact with it in the normal way. For example, you can follow hyperlinks in a browser or tweet, or you can annotate an image or save it into your personal space.

Note: *You can configure the length of the inactivity period (also called the 'idle timeout').*



Cloud widget. Tap an item to detach it from the cloud.

1 *Drag with your finger to rotate and resize the ball.*

2 *Tap an item to detach it from the cloud.*

3 *When you tap an item, the Open button display. Tap this button to open a full version of the item.*

4 *Idle detached items automatically drift back to re-join the cloud.*

10.1 Cloud size

The cloud widget is particularly suited to large content sets (say, 20 or more images).

The cloud can even handle very large content sets (over 100 items) and at this level its visual impact is undeniably dramatic. However, you need to balance visual impact against ease of finding an item. If you simply want users to enjoy browsing an extensive collection of images, then a very large cloud is ideal. But if you want users to be able to quickly find specific items, then you may want to limit the size of the cloud.

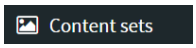
Conversely, the cloud's visual impact is somewhat reduced if the number of items falls below ten or twelve.

10.2 Set up a content set

Before creating a cloud, you must create the content set that supplies it with media such as images and videos, or with URLs or tweets. Follow these steps:

1. If you want to add images, videos or PDFs to the cloud's content set, you must first ensure that the items you want have been added to your media library.

   For instructions, refer back to step 1 in section 5.1.

2. Create a new content set.
   a. Click Content 

in the left-hand pane.
   b. Click the Create New button
   c. Enter the name of the new content set. For example, Content for Cloud.

3. In the *Content Set* screen, drag items into the content set:
   a. Click the 

button to open the media library pane.
   b. Drag images from the media library to the root folder of the new content set.

      Tip: *You can drag folders directly from the media library. You do not need to drag items individually.*
   c. Close the media library pane.

4. Click the Save button to save the content set.

10.3 Add a theme to your app

Although themes are optional, we strongly recommend that your app uses a theme if it also includes a cloud widget. This is because you must define the content viewers for all *opened* items in the cloud. For example, you need to ensure if a user opens an image from the cloud, it will display in an *image viewer* widget, and that an opened video will display in a *video viewer* widget, and so on.

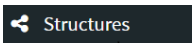
If your app uses a theme, these content viewers are specified by default in the theme. Conversely, if your app does not use a theme, you will need to manually specify the content viewer for each item in the cloud. *This could be a slow and laborious task!*

Instructions for adding a theme to your app are in section 6.4.

10.4 Add a cloud to your app's structure

Now you need to add a cloud to the structure you created earlier (see step 1 in section 3.3). Then you need to configure the menu to use a new content set.

Follow these steps:

3. Click Structures 

in the left-hand pane.
4. Click My First Structure.

   The *Editing a Structure* screen opens.
5. Go to the Main section and click *Add a new widget to Main*.
6. From the pop-up menu, add a **Cloud widget**.

   Tip: *Click the Add button.*
7. In the Cloud Widget Attributes pane, set the following attributes:
   - **Size** and **Cloud** location: These attributes are self-explanatory. You will probably want to change the default values. As a minimum, we recommend that your cloud is at least 1000 x 1000 pixels.
   - **Pin the widget when it opens:** A cloud must be pinned before users can spin the ball and select items. For this example, set this attribute to Enabled.

     Note: *If you pin the cloud when it opens, users will be unable to move or resize the cloud unless you also provide a toolbar with a Pin button; see section 3.5.1.*
   - **Timeout for detached items** and **Return velocity**: The timeout determines how long detached items wait when idle. The return velocity determines how fast they float back to rejoin the cloud when the timeout expires.
8. Now you must select the content set that supplies your cloud.
   a. Go to the **Content set** attribute.
   b. Select the content set you created earlier, Content for Cloud, from the drop-down menu.
9. Click the Save button to add the cloud to My First Structure.

10.5 Test the cloud

You have now updated the structure to use your new cloud widget. To test the cloud:

1. View your app on your video wall. The cloud is already displayed.

   (It was added to the app automatically when you clicked the Save button in the previous section. You do not need to restart the app to display the cloud.)

2. If you enabled the Pin the widget when it opens attribute, you can start browsing the cloud immediately; go to step 3.

   Otherwise, tap the Pin button in the cloud widget's toolbar.

3. Find an item in the cloud:

   a. Use your fingers to rotate the ball.

   b. Find an item you want. Then tap it once to activate the item and *briefly lift your finger from the screen.*

   c. Drag the item out of the cloud.

   d. When the **Timeout for detached items** expires, the item will automatically rejoin the cloud.

10.6 Twitter clouds

You can add a Twitter cloud to your app. The Twitter cloud comprises a ball of individual tweets that match the specified search terms. To create a Twitter cloud, you simply set up a content set that includes a Twitter Feed widget, and then assign this content set to a Cloud widget. For setup details, see section 17.4.

The tweets in a cloud are *previews*. If a tweet includes a hyperlink, you can tap the hyperlink directly (you may need to first rotate the cloud to bring the tweet to the front). To open the full version of a tweet—for example, so you can resize it to make it easier to read—tap the tweet preview and then tap the resulting Open button. To block inappropriate tweets, you can use a 'tweet blocker' Codice card; see section 13.5.

You can also include a Twitter Feed in a conventional cloud, so that individual tweets appear alongside images, movies and browsers in the cloud.

Individual tweet. **1** *Hyperlinks. If the hyperlink opens a browser or PDF document, you will have to pin the resulting widget before you can interact with it.*

Note: *Twitter clouds require the Twitter Connection service to operate. For details about adding this service to your app, see section 12.3.*

11 Create a content hotspot

A content hotspot is a special area of the screen that supports custom user interactions. When users tap a content hotspot, items such as an image, video or PDF are displayed. You can even launch a finger menu or rotating menu from a hotspot.

You can add content hotspots directly to the Main layer in your app. You can also add content hotspots to images (*hotspot images*) and videos (*hotspot videos*). You can add hotspot images and videos directly to the Main layer in your app, to menus and to Codice content. Instructions below mention hotspot images, but they apply equally to hotspot videos.

- **Adding a content hotspot directly:** With this method, the hotspot has *a fixed size and location.* The hotspot is only visible to users if you use a visualization widget or a background color. This type of hotspot is suitable if you want hotspots in specific areas of the screen, for example, above areas of your app's background image. For details, see section 11.1.
- **Adding a hotspot image directly:** With this method, an Image Viewer widget in the Main layer of your app becomes a hotspot image. Users can move, rotate and resize the image as normal, but an item launches if they tap a hotspot on the image.

  A single hotspot image can include multiple hotspots, with each hotspot launching a separate item. For example, you can add multiple hotspots to key locations on a map or drawing. For details, see section 11.2.
- **Adding a hotspot image to a menu:** With this method, hotspot images (see above) are opened from a menu. That is, a menu item opens an Image Viewer widget. Users can then tap hotspots on the image to launch items. For details, see section 11.3.
- **Adding a hotspot image to Codice content:** With this method, hotspot images (see above) are opened with a Codice card. That is, a user places a Codice card on the screen to open an Image Viewer widget. The user can then tap hotspots on the image to launch items. For details, see section 11.4.

In all cases, you can set the hotspot size, location and (optionally) background color, plus the items that display when a user taps the hotspot.

For content hotspots and hotspot images, you can also assign a *visualization widget* (an image or video) to help users locate the hotspot and suggest the item that will be launched. The visualization widget displays above the hotspot but does not affect user interaction with the hotspot. For example:

- **Content hotspots:** An app includes a grid of hotspots to launch PDF product brochures for a range of graphics cards. Each hotspot uses a photo of the relevant graphics card as its visualization widget.
- **Hotspot images:** An app for a university includes a hotspot image of a campus map. A hotspot is defined for each faculty on the map, with  images or similar used as visualization widgets to alert users to the presence of hotspots. For an example, see section 11.7.

11.1 Add a content hotspot directly to the Main layer

This section applies to Content Hotspot widgets added directly to the Main layer of your app's structure.

11.1.1 Add a content hotspot to your app

Follow these steps:

1. Click Structures 

in the left-hand pane.
2. Click the structure you want to use. For example, My First Structure.
   The *Editing a Structure* screen opens.
3. Go to the Main section and click the *Add a new widget to Main* hyperlink.
4. From the pop-up menu, add a **Content Hotspot**.
5. In the *Content Hotspot attributes* pane, set the general hotspot attributes:
   - **Hotspot location** and **Hotspot size** define the hotspot area. However, these attributes are optional if you prefer to use *hotspot visualization widgets* (see section 11.1.2) to define the 'tappable' hotspot area.
   - **Background color** is designed to help app designers visualize the hotspot area while setting up the hotspot. Typically, you revert to the default (ie, transparent) background color when finalizing the app.
   - *(Optional)* If required, adjust the **Contents opening animation duration** attribute. This determines how quickly (or slowly) items take to fully open when a user taps the hotspot. You can set a duration of 0.1 to 2.0 seconds.

   Click the help button if you need details. See also section 25.6.2.
6. (Optional) If you intend to add multiple hotspots to your app (see section 11.8), you may want to give distinctive names to each hotspot in the Editor.
   To rename a hotspot, right-click the Content hotspot widget and click Rename.

Drop-down menu for content hotspot widget.

7. Click the Save button to add the content hotspot menu to My First Structure.

Now set up the content hotspot appearance and behavior. Go to section 11.1.2.

11.1.2 Set up the content hotspot

Now you need to specify which items display when a user taps the hotspot, and which images or movies (if any) are used as the hotspot background. Follow these steps:

1. Go to the Main section of the *Editing a Structure* screen.
2. Click the ▷ triangle next to the Content Hotspot widget.

   The widget expands to reveal three widget lists:
   - Hotspot visualization widgets
   - Widgets launched from hotspot around tap location
   - Widgets launched from hotspot at absolute screen locations
3. *(Optional)* Add a visualization widget.

   To help users locate the hotspot, you can add a visualization widget. This widget (an image or movie) serves as a background for the hotspot. In addition:
   - Visualization widgets can foreshadow the hotspot's content. For example, if your hotspot launches a finger menu of information about a new construction project, you can use an architect's drawing as your hotspot's background.
   - Visualization widgets are interactive. Users can tap a visualization widget to launch hotspot content in the same way as they can tap the hotspot area.
   - Visualization widgets can be used to extend the interactive hotspot area. You can position a visualization widget *anywhere* on the screen. Even if it is not within the hotspot area, and even if it doesn't have the same width or height ratio as the hotspot area, *the visualization widget is still interactive and part of the hotspot!*

   Note: *When you set the* Location *attribute of the visualization widget, its location co-ordinates are relative to the top-left corner of the hotspot area, not the top-left corner of the screen. (You defined the hotspot area in step 5 of the previous section.*)

   To add visualization widgets, follow these steps:

   a. Click the *Add new widget to Hotspot visualization widgets* hyperlink.

   b. Add the widget you want. Choose from Image Viewer, Image Movie or Video Viewer.

   c. In the right-hand pane, set attributes for the new widget. In particular, set the Location and Size attributes. If necessary, use section 2.9 for guidance on how to edit widget attributes.

   d. Click Save to save visualization widget for your hotspot.

4. Add the widget that opens when a user taps the hotspot. This can be any widget normally found in the Main or Menu layers of an app (such as a video, PDF, browser or finger menu). To specify this widget, you must add it to one of the *Widgets launched from* lists. There are two lists, each with its own interpretation of the widget's **Location** attribute:

   - **Widgets launched from hotspot around tap location:** Use this list if you want a widget to open close to where the user taps the screen. For this list, the widget's Location attribute is *relative to the touch event*. For example, if the location is set to 0,0, the widget will always open exactly where the user taps the screen.
   - **Widgets launched from hotspot at absolute screen locations:** Use this list if you want a widget to open at a specific screen location. For widgets in this list, the widget's Location attribute is *relative to the top left corner of the screen*. For example, if the location is set to 0,0, the widget will always open in the top left corner of the screen.

     Tip: *You can use this list to launch multiple widgets from a single hotspot, with each widget opening at a different screen location. For example, you can set up a hotspot to open a product movie and PDF brochure side-by-side on the screen.*

   To add the widget that opens from the hotspot:

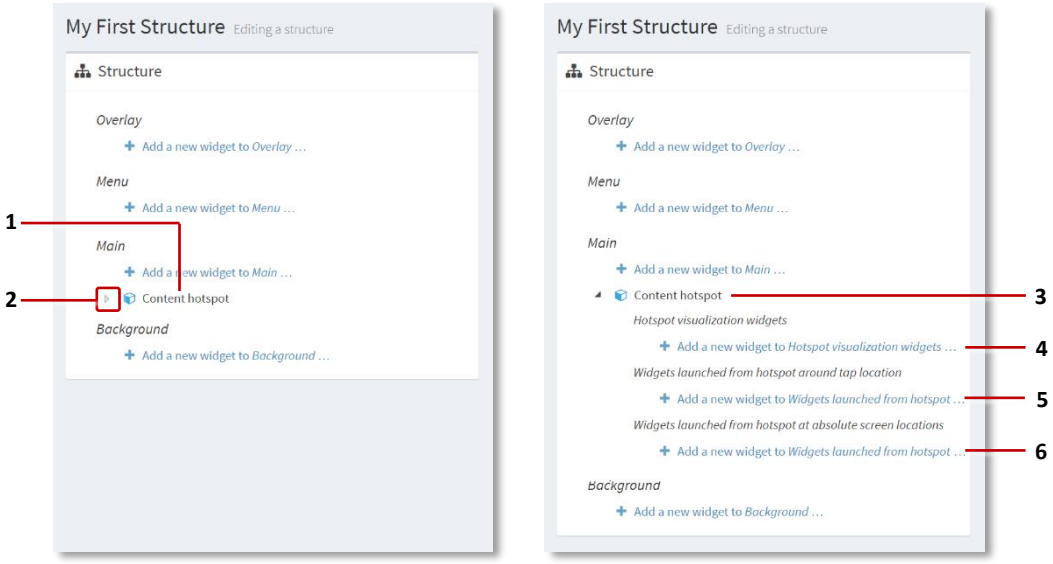
   a. Click the *Add* hyperlink for widgets from the hotspot.

   b. Add the widget you want.

   c. Set the **Location** attribute. See above for details of how this location is interpreted.

   d. Set the remaining attributes for the new widget. If necessary, use section 2.9 for guidance on how to edit widget attributes.

5. Click Save to save the hotspot setup.



Setting up a content hotspot. 1 *Content hotspot widget added to Main layer.* 2 *Triangle for content hotspot widget. Click to reveal widget lists.* 3 *Expanded content hotspot widget.* 4 *Hyperlink to add hotspot visualization widgets.* 5 *Hyperlink to add widgets that open around the tap location.* 6 *Hyperlink to add widgets that open at absolute screen locations.*

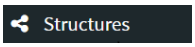
11.2 Add a hotspot image directly to the Main layer

This section applies to Image Viewer widgets in the Main layer of your app's structure.

If you want to add a hotspot to an image in your app, you must first add an image to your app. Then you add one or more content hotspots to the image. Finally, you must set up the content hotspots (that is, you need to specify which item displays when a user taps a hotspot).

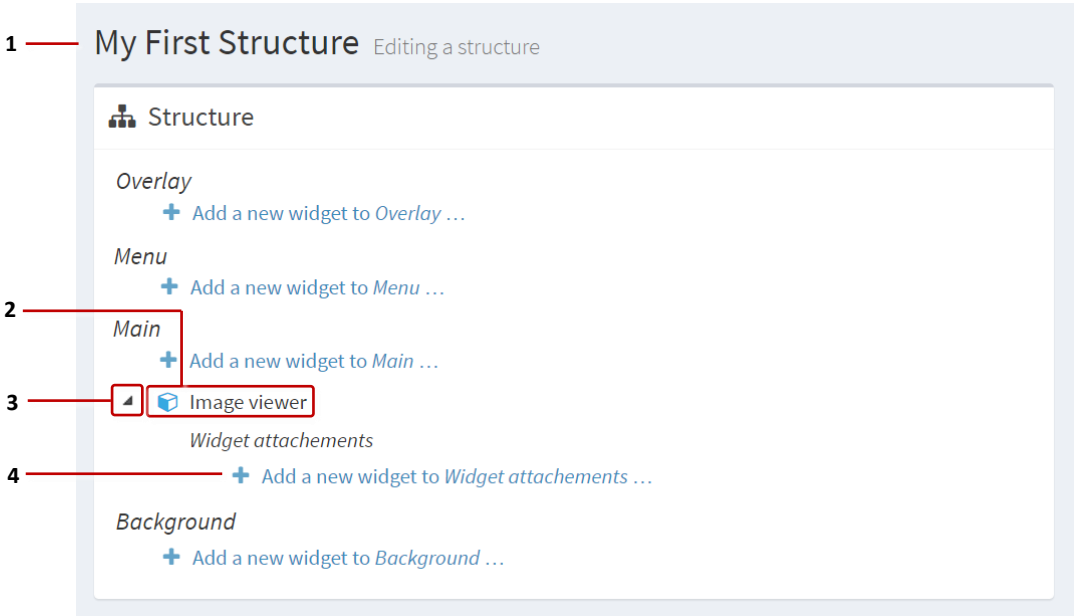
11.2.1 Add an image to your app

Follow these steps:

1. Click Structures 

in the left-hand pane.
2. Click the structure you want to use. For example, My First Structure.
   The *Editing a Structure* screen opens.
3. In the Main section, go to the Image Viewer widget you want to use. (Choose an existing Image view or add a new one.)
4. Click the ▷ triangle next to the Image Viewer widget.
   The widget expands to reveal a *widget attachments* list and the *Add a new widget to widget attachments* hyperlink.

Now continue to section 11.2.2.



Adding a content hotspot to an image. **1** *Editing a structre screen.* **2** *Image Viewer added to Main layer.* **3** *Image viewer triangle. Click to reveal widget attachments list.* **4** *Hyperlink to add 'new widget to widget attachments'.*

11.2.2 Add hotspots to the image

You now need to add content hotspots to the Image Viewer widget. This task is described in section 11.5.

11.2.3 Set up the hotspot image

Finally, you need to specify which items display when a user taps the hotspot image. This task is described in section 11.6.

11.3 Add hotspot images to a menu

This section applies to hotspots on images launched from a menu node. The menu node can be in a finger menu, rotating menu, or teaser menu.

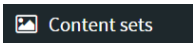
You can add 'hotspot images' to menu items. That is, when a user taps a menu item, an image opens. This image contains one or more content hotspots. If the user then taps the image, further items open, such as images, videos, PDFs or browsers or even another menu.

To add hotspot images to a menu, you must first add the hotspot images to a content set. Then you must add a menu to your app and configure the menu to use this content set.

11.3.1 Add images to your content set: Recommended method

This method adds a *widget* to menu node in the content set, and then designates the widget as an 'Image Viewer with a Content Hotspot attachment'. The advantage of this method is that your hotspot image is a *shared widget* that you can re-use in other content sets and menus. (Shared widgets are listed in the widget library; see section 2.2.4.)

Follow these steps:

1. If you have not already done so, drag the images you want into the media library. You can then populate the content set with these images. See step 1 in section 5.1.
2. Edit the content set you want to use.
   a. Click Content sets 

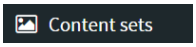
in the left-hand pane.
   b. Click the content set you want to use. For example, Content for Menus.
3. In the *Content Set* screen:
   a. Select the root node.
   b. Click the Add button and add a widget.
   c. Select the new widget.
4. Now set the widget type. This must be an Image Viewer widget. In the right-hand *Content [Widget] attributes* pane:
   a. Set a visible name. This name will display in your app as the label for the menu item.
   b. Go to the **Widget** attribute and click the New button.
   c. Add an Image Viewer and give it a name. For example, My Hotspot Image.
   d. Click the Save button to save the Image Viewer widget to your content set.
5. Still in the *Content [Widget] attributes* pane, return to the **Widget** attribute and this time click the Edit button
   The *Editing a widget* screen displays.

Now continue to section 11.3.3.

11.3.2 Add images to your content set: Alternative method

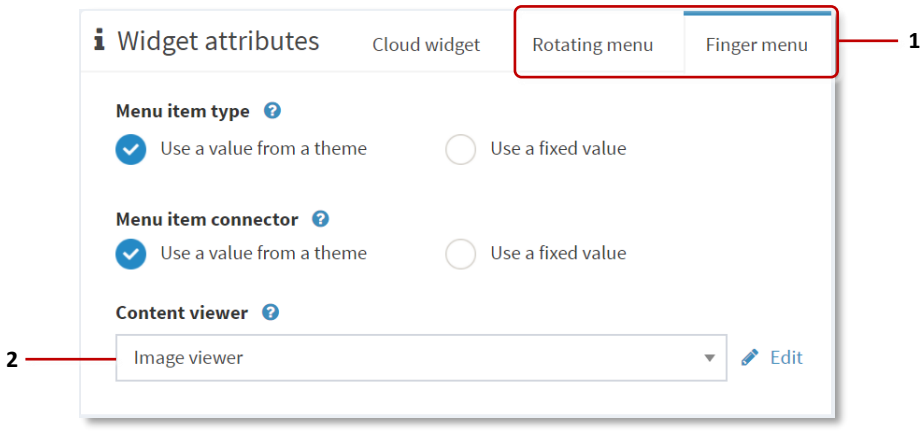
This method adds images from the media library directly to menu nodes in the content set. It then designates adds content hotspots to these images, The advantage of this method is that it is quick and simple. This disadvantage is that you cannot re-use the hotspot image in other menus or content sets.

Follow these steps:

1. If you have not already done so, drag the images you want into the media library. You can then populate the content set with these images. See step 1 in section 5.1.
2. Edit the content set you want to use.
   a. Click Content sets 

in the left-hand pane.
   b. Click the content set you want to use. For example, Content for Menus.
   Note: *The content set must have a 'Folder' root node.*
3. In the *Content Set* screen:
   a. Select the root node.
   b. Click the 

button to open the media library pane.
   c. Drag images from the media library into the new content set.
   Each image will be a *menu node* (that is, an item in the menu).
   d. Close the media library and save the changes.
4. Still in the *Content Set* screen, select the menu node that you want to convert to a hotspot image.
5. In the widget attributes pane:
   a. Click the Rotating menu tab or Finger menu tab, as required.
   b. Go to the **Content Viewer** attribute.
   c. Choose **Image viewer** from the drop-down list and click Save.
   d. Still in the **Content Viewer** attribute, click the Edit button.
   The *Editing a widget* screen displays.



Widget attributes pane. 1 *Menu tabs.* 2 *Content viewer attribute.*

Now continue to section 11.3.3.

11.3.3 Add hotspots to the image

Now add content hotspots to the Image Viewer widget. This task is described in section 11.5.

11.3.4 Set up the hotspot image

Now you need to specify which items display when a user opens a menu item and taps a hotspot image. This task is described in section 11.6.

Now add a menu to your app and then assign the 'hotspot content set' to that menu. Continue to section 11.3.5.

11.3.5 Add a hotspot menu to your app

You have now set up a 'hotspot content set' to supply a menu with hotspot images. You must now add a menu to your app and configure the menu to use the new hotspot content set.

Follow these steps:

1. Add a menu to your app's structure. For instructions, see:
   - Finger menu: section 5.2.
   - Rotating menu: section 8.4.
   - Teaser menu: see section 9.2.
2. In the *Editing a structure* screen, select the Finger menu or Rotating menu (in the Menu layer) or the Teaser menu (in the Main layer).
3. In the right-hand menu attributes pane:
   - Finger menu: Go to the **Content Set** attribute.
   - Rotating menu: Go to the **Content Set** attribute.
   - Teaser menu: Go to the **Menu** attribute.
4. *(Finger menus and Rotating menus only)* Select the content set with hotspot images from the dropdown list. This is the content set you edited in sections 11.3.1 or 11.3.2.
5. *(Teaser menus only)* Follow these steps:
   a. Click the Edit button for the **Menu** attribute.
   b. In the *Editing a widget* screen, go to the Content set attribute in the right-hand menu attributes pane.
   c. Select the content set with hotspot images from the dropdown list. This is the content set you edited in sections 11.3.1 or 11.3.2.
6. Click the Save button to add the content set to the teaser menu.

11.4 Add hotspot images to Codice content

This section applies to hotspots on images launched from a personal marker ie, a Codice card.

Codice content refers the item that opens automatically when a user presents a personal marker (ie, holds it against the screen). This item could be an image, video or even a web site. In fact, all the main 'content widgets' are supported, such as image viewers, web browsers, or cloud widgets. For details see section 13.4.

To set up a hotspot image as Codice content, you must first assign a normal image to a specific Codice code. Then you add one or more content hotspots to the image. Finally, you must set up the content hotspot (that is, you need to specify which items display when a user taps the hotspot).

11.4.1 Add an image to Codice content

You set up Codice content in the *Codice Database* screen. Follow these steps:

1. Open the *Codice Database* screen and select the marker that you want to launch a hotspot image. Then edit the marker's Codice content.
   For details, follow steps 1 through 4 in section 13.4.

2. In the *Editing a structure* screen, you need to add the hotspot image to a *Widgets launched* list. These widgets open when a user presents the relevant marker.
   There are two widget lists, each with its own interpretation of the widget **Location** attribute. You can add widgets to either list, or to both lists:
   - **Widgets launched around Codice marker:** For a widget in this list, the Location attribute is relative to *the position of the marker on the screen*. Use this list if you want the hotspot image to open close to the marker.
     For example, if the location of the hotspot image is 0,0, it will always open directly below the marker.
   - **Widgets launched at absolute screen locations:** For a widget in this list, the Location attribute is relative to *the top left corner of the screen*. Use this list if you want the hotspot image to always open at the same location on the screen.
     For example, if the location of the hotspot image is 0,0, then the image will always open in the top left corner of the screen.

   In the *Editing a structure* screen:
   a. Click either of the two *Add a new widget* hyperlinks.
   b. Add the image you want to use as a hotspot image.
   c. Set the **Location** attribute. See above for details of how this location is interpreted.
   d. Set the remaining attributes for the new hotspot image. If necessary, use section 2.9 for guidance on how to edit widget attributes.

3. Click the Save button.

Now continue to section 11.4.2.

11.4.2 Add hotspots to the image

Now add content hotspots to the Image Viewer widget. This task is described in section 11.5.

11.4.3 Set up the hotspot image

Now you need to specify which items display when a user presents a Codice card and taps the hotspot image. This task is described in section 11.6.

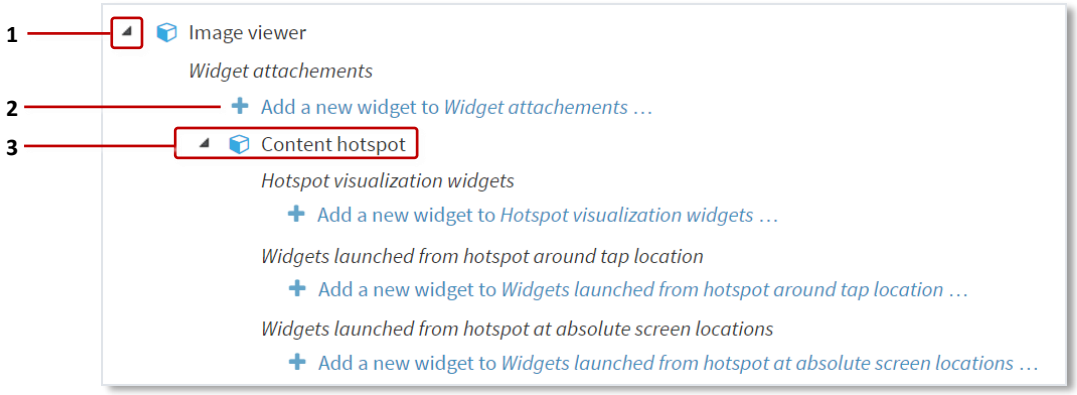
11.5 Add hotspots to the image

Terminology: *In this section, a hotspot image refers to an image with one or more hotspots defined.*

After adding an image to your app, content set or Codice content, you must add a Content Hotspot widget to image's list of attachments. Follow these steps:

1. Make sure you have completed the required preliminary steps. If adding a hotspot image:
   - Directly to the Main layer in your app, first complete the steps in section 11.2.1.
   - To a menu, first complete the steps in sections 11.3.1 or 11.3.2.
   - To Codice content, first complete the steps in section 11.4.1.

   After completing these sections, you will be in the *Editing a structure* screen or, if adding a hotspot image to a menu, in the *Editing a widget* screen.
2. If you have not already done so, assign an image file to the Image Viewer widget:
   a. Select the Image Viewer widget.
   b. In the right-hand *Image viewer attributes* pane, go to the **Image file** attribute.
   c. Click the Media library button to open the media library pane.
   d. Drag an image from the media library into the **Image file** attribute.
   e. Close the media library and save the changes.



Adding a content hotspot to an image. **1** *Image Viewer triangle. Click to reveal the widget attachments list.* **2** *Hyperlink to add 'new widget to widget attachments'.* **3** *Content hotspot added to widget attachments list.*

3. Click the ▷ triangle next to the Image Viewer widget.
   The widget expands to reveal a *widget attachments* list.
4. Click the *Add a new widget to widget attachments* hyperlink.
5. From the pop-up menu, add a **Content Hotspot**.

6. In the *Content Hotspot attributes* pane, set the hotspot location and size:
   - **Hotspot location:** This location is relative to the location of the image. Set the location '0,0' if you want to align the hotspot location exactly with the image's current location on the screen.
   - **Hotspot size**: Set the size to 'calc(100%),calc(100%)' if you want to make the entire image a single 'tappable' hotspot area.

   Click the help button for details about the other attributes.

   Note also that you can define multiple hotspots on a single image, each with its own location and size; see step 8.
7. Click the Save button to save the image as a hotspot image.
8. (Optional) If you want to add multiple hotspots to the image, repeat steps 4 to 7 for hotspot that you want to add. When defining the hotspot locations and sizes, make sure the hotspots do not overlap.

   For an example of an image with multiple hotspots m, see section 11.7.

Now you need to set up the hotspot image. That is, you must define the items that open when a user taps a hotspot on the image. Continue to section 11.6.

11.6 Set up the hotspot image

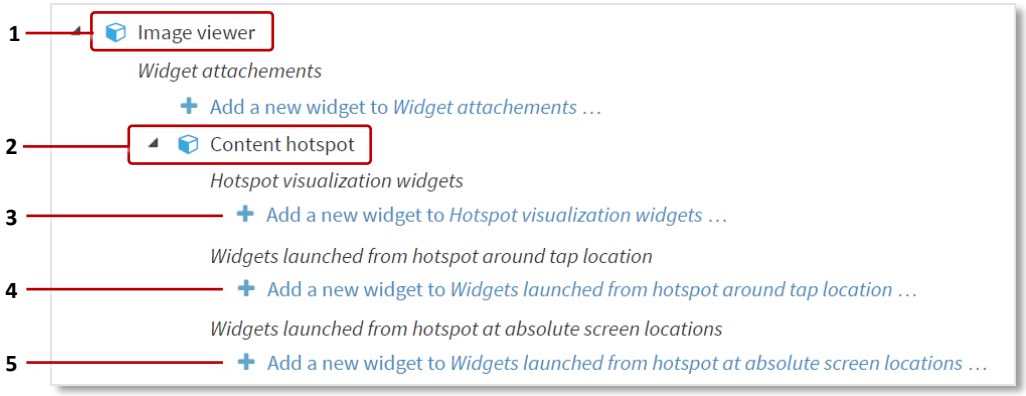
Now specify which items display when a user taps a hotspot on the image.
Follow these steps:

1. Make sure you have converted the image to a hotspot image.

   After completing the steps in section 11.5, you will be in the *Editing a structure* screen or, if adding a hotspot image to a menu, in the *Editing a widget* screen.

2. Click the Content hotspot that you added in step 5 of section 11.5.

   The widget expands to reveal three widget lists:
   - Hotspot visualization widgets.
   - Widgets launched from hotspot around tap location
   - Widgets launched from hotspot at absolute screen locations



Setting up a hotspot image. 1 *Image Viewer widget.* 2 *Content Hotspot widget.* 3 *Hyperlink to add hotspot visualization widget.* 4 *Hyperlink to add widgets that open around the tap location.* 5 *Hyperlink to add widgets that open at absolute screen locations.*

3. *(Optional)* Add a visualization widget.

   To help users locate the hotspots on your image, you can add a visualization widget; see the example in section 11.7. In addition:

   - Visualization widgets can foreshadow the hotspot's content. For example, if your hotspot launches a finger menu of information about a new construction project, you can use an architect's drawing as your hotspot's background.
   - Visualization widgets are interactive. Users can tap a visualization widget to launch hotspot content in the same way as they can tap the image hotspot.
   - Visualization widgets can be used to extend the interactive area of the hotspot. You can position a visualization widget *anywhere* on the screen. Even if it is not aligned with the hotspot image, and even if it doesn't have the same width or height ratio as the hotspot image, *the visualization widget is still interactive and part of the hotspot!*

   Note: *When you set the* Location *attribute of the visualization widget, its location co-ordinates are relative to the top-left corner of the hotspot area, not the top-left corner of the screen. (You defined the hotspot area in step 5 of the previous section.)*

To add a visualization widget, follow these steps

a. Click the *Add new widget to Hotspot visualization widgets* hyperlink (item 3 in the previous screenshot).

b. Add an Image Viewer widget.

c. In the right-hand *Image viewer attributes* pane, set attributes as required. In particular, set the **Location**, **Size** and **Image file** attributes.

If necessary, use section 2.9 for guidance on how to edit widget attributes.

d. Click Save to save visualization widget for your hotspot image.

4. Add the widget that opens when a user taps the hotspot on the image. This can be any widget normally found in the Main or Menu layers of an app (such as a video, PDF, browser or finger menu).

To specify this widget, you must add it to one of the *Widgets launched from* lists. There are two lists, each with its own interpretation of the widget **Location** attribute:

- **Widgets launched from hotspot around tap location:** Use this list if you want a widget close to where the user taps the screen. For widgets in this list, the widget's Location attribute is *relative to the touch event*. For example, if the location is set to 0,0, the widget will always open exactly where the user taps the screen
- **Widgets launched from hotspot at absolute screen locations:** Use this list if you want a widget to open at a specific screen location. For widgets in this list, the widget's Location attribute is *relative to the top left corner of the screen*. For example, if the location is set to 0,0, the widget will always open in the top left corner of the screen.

  Tip: *You can use this list to launch multiple widgets from a single hotspot, with each widget opening at a different screen location. For example, you can set up a hotspot to open a product movie and PDF brochure side-by-side on the screen.*

To add a widget to these lists, follow these steps:

a. Click the required *Add new widget to Widgets launched from hotspot* hyperlinks (ites 4 and 5 in the previous screenshot).

b. Add the widget you want.

c. Set the **Location** attribute. See above for details of how this location is interpreted.

d. Set the remaining attributes for the new widget. If necessary, use section 2.9 for guidance on how to edit widget attributes.

5. Click Save to save the new hotspot on the image.

6. *(Optional)* If you added multiple hotspots to the image (see step 8 in section 11.5), repeat steps 2 to 5 in this section for each hotspot in the image.

7. *(Applies only if adding a hotspot image to a menu)* These instructions complete the setup for a single menu node. (the node you selected in step 3.b of section 11.3.1 or step 4 of section 11.3.2).

   If you want to convert another menu node to a hotspot image, follow these steps for each node you want to convert:

   a. Do one of the following:
      - **Recommended method:** Repeat steps 4 and 5 in section 11.3.1.
      - **Alternative method:** Repeat step 5 in section 11.3.2.

   b. Repeat section 11.5.

   c. Repeat this section (section 10.6).

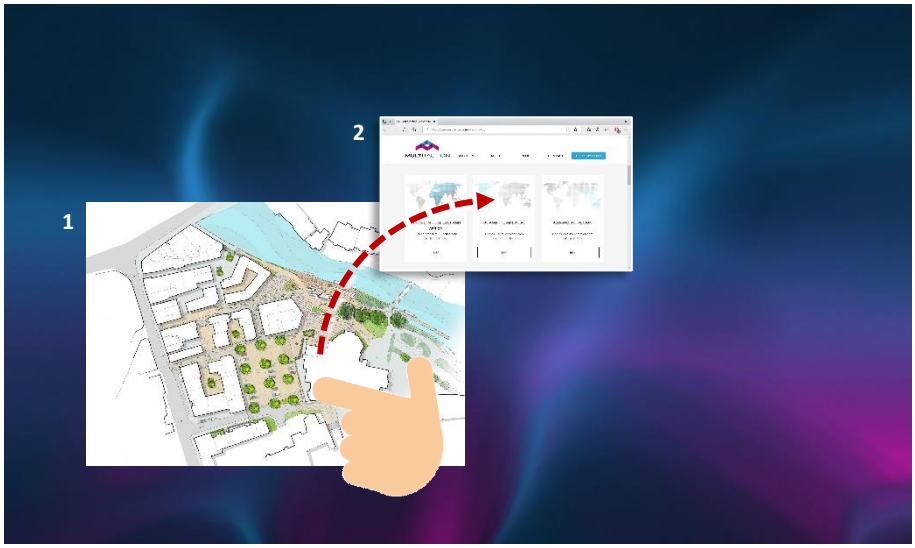
8. *(Applies only if adding a hotspot image to a menu)* Now add a menu to your app and then assign the 'hotspot content set' to that menu. Return to section 11.3.5.

This completes the hotspot image setup. When a user opens the image (either directly from your app, a menu, or by presenting a Codice card(), they can then tap a hotspot on the image to open further items.

11.7 Add multiple hotspots to an image

If required, you can define multiple hotspots on a single image. To help users locate the hotspots, you can also add a visualization widget to each hotspot.

In the first example below, a hotspot image has a single hotspot extending over the entire image. The user can tap the image anywhere to launch the hotspot content.



1 *Hotspot image with a single hotspot extending across the entire image.* **2** *Users can tap the image anywhere to launch the hotspot content.*

In the next example, the same image has multiple hotspots. Each hotspot has its own size and location (not visible to the user) and launches its own hotspot content. In this example, each item launched from a hotspot opens at an absolute screen location. To help users locate the hotspots, each hotspot uses a *visualization widget* (see step 3 in section 11.6).

1 *Hotspot image with three hotspots.* **2** *Visualization widgets identify the hotspots to users.* **3** *Each hotspot launches a different item. In this example, the items open at absolute screen locations. When open, the items are arranged in a column at the edge of the screen.*

11.8 Add multiple content hotspots

Applies to content hotspots and image hotspots added directly to your app.

Follow these steps:

1. Add the content hotspots or hotspot images to your app. Follow the instructions in section 11.1 or section 11.2.
2. Set the size and location of each content hotspot.

   In the *Content Hotspot attributes* pane:

   a. Go to the **Hotspot location** attribute and set the location coordinates of the hotspot top-left corner.

   b. Go to the **Hotspot size** attribute and set the hotspot width and height, in pixels.

   Important! *Make sure the hotspots do not overlap.*
3. Set up each content hotspot. Follow the instructions in section 11.1.2.

Example: To set up four hotspots in a grid with a 50-pixel border between them, use these attribute values:

	Hotspot location	Hotspot size	Preview
Hotspot #1	50, 50	450 x 450 px	
Hotspot #2	550, 50	450 x 450 px	
Hotspot #3	50, 550	450 x 450 px	
Hotspot #4	550, 550	450 x 450 px	

11.9 Test the content hotspot

You have now updated the structure to use your content hotspot. To test the hotspot:

1. Restart your app.

   a. Go to the *App launcher* screen.

   b. Click the Restart button for My First App.
2. Tap the hotspot screen region.

   When you tap the hotspot, the 'launched widget' that you added previously now opens.

   Tip: *If you used a visualization widget as described in step 2 of section 11.1.2, you can use the image or video to infer the hotspot boundaries.*

12 Create a service set

A *service set* defines a specific administrative setup for an MT Showcase app. Each MT Showcase installation can support multiple service sets. In the current version of MT Showcase, service sets can include for example the *Email Sending* service and *Data Gathering* service. For an overview of service sets, see section 2.5.

12.1 Email Sending service

The *Email Sending* service is used for sending screen content from a user's personal space (section 13.3) to a specified email account. When you add the Email Sending service to a service set, you must define the SMTP host, credentials for an SMTP user, the sender's email address, plus the subject and body text.

12.1.1 What do emails sent from MT Showcase contain?

Emails sent from MT Showcase can contain attachments of images, videos, and PDFs. Or they can include URLs to these items or a web site. The emails also include a configurable sender, subject and body text.

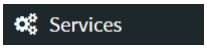
- **Sender, subject and body text**
  The Email Sending service has attributes for configuring:
  - **Sender:** Use the Email Sender Name and Email Sender Address attributes to identify the sender to the recipient. For example, Showcase and noreply@unipraxis.com.
    Note: *Some email services, including Gmail, will use the SMTP Username attribute for the sender's address instead of the supplied Email Sender Address.*
  - **Subject:** The Email Subject attribute is self-explanatory. For example, Content sent from Showcase.
  - **Body text:** The Email Template attribute defines the body text of any emails sent by MT Showcase. The template can include variables to customize the text content. For example, you can use a variable to add the recipient's name. For details about setting up an email template, see section 12.1.3.
- **Attachment and URL rules**
  To avoid sending very large videos or PDFs as email attachments, you can set up MT Showcase emails to include the URL to an item instead of sending the actual item as an attachment. For details about defining URLs in the Media library, see section 12.1.4.
  MT Showcase uses simple rules to determine what is included in the email:
  - **Images:** An image file is always sent as an attachment.
    If a URL to the image has been defined, the URL is always included in the body text. If a user has annotated (drawn on) the image using an infrared pen, the annotated image is always attached, *not the original image!*

- **PDFs:** *Applies to documents dragged from PDF Book and PDF Flow widgets.*
  If a URL to the PDF *has* been defined, only the URL is included; the PDF file is *not* sent. If a URL has *not* been defined, the PDF file is sent as an attachment
- **Videos:** The email rules depend on the video source. If the video source is a file such as an MP4 movie:
  - If a URL *has* been defined, only the URL is included; the video file is *not* sent. If a URL has *not* been defined, the video file is sent as an attachment.
  - If the video source is a video stream, the stream URL is included.
    For example, rtsp://10.36.0.122:8554/presentations.
  - If the video source is a video capture device such as a web cam, the URL for the device is included:
    Ubuntu example: /dev/video1
    Windows example: video=Datapath VisionSC-HD4+ Video 03:audio=Datapath VisionSC-HD4+ Audio Digital 03
- **Web page:** The web page URL is always included in the body text.

12.1.2 Set up the SMTP server

If you add the *Email Sending* service to your app's service set, you must set up the SMTP server that MT Showcase uses to send screen content.

Follow these steps:

1. Click 

in the left-hand pane.
2. Either click an existing service set.
   Or create a new service set:
   a. In the  Service Sets screen, click the Create New button
   b. Enter the name of the new services set. For example, My Service Set.
   c. Click the Create New button.
3. In the Editing service set screen, go to the *Add new service* section:
   a. Select **Email Sending** from the Select Service drop-down menu.
   b. Enter a name for the new email sending service.
   c. Click the Add button.
4. Still in the Editing service set screen, go to the *Email Sending* section.
5. Edit the email attributes, as required. You will need to supply details about your SMTP server, host, port and password. You will also need to specify the email accounts (sender and from) that this service will use.
   If required, you can also increase the timeouts for sending emails (for example, there is a 60 second timeout for attempts to connect to the SMTP server), although it is unlikely that you will need to change the default timeouts.

Finally, you can choose to ignore SSL errors. By default, MT Showcase does not send an email if it detects an SSL error (such as an invalid certificate) when connecting to the SMTP server. If you trust the connection, you can instruct MT Showcase to ignore SSL errors and send the email anyway.

Click the help button for details.

6. Click the Save button to add the Email Sending service to your service set.

Now set up the email template that this service will use. Go to section 12.1.3.

12.1.3 Set up an email template

An email template defines the body text of any emails sent by MT Showcase. The template can include variables to customize the email's text content. For example, you can use a variable to add the recipient's name.

Follow these steps:

1. Click Services in the left-hand pane.
2. Click the service set you want. For example, My Service Set.
3. In the *Editing service set* screen, go to the *Email Sending* section:
4. Go to the **Email subject** attribute and enter text for the email's subject field.
   For example, Content exported from MT Showcase.
5. Go to the **Email template** attribute and enter the text that appears in the body of the e-mail message.

   You can include these variables:

   <name> Adds the user name associated with the registered Codice card.

   <links> Adds URL hyperlinks to the body text. If you omit this variable the URL hyperlinks are added the end of the message; see section 12.1.4.

   For example:

   "Hi <name>

   Follow these links to find the MT Showcase items that interested you:
   <links>

   For information about other MultiTaction products, visit
   www.multitaction.com."

6. Click the Save button to update the service set with the changes to the Email Sending service.
7. If you have not already done so, remember to add the service set containing the Email Sending service to your app; see section 12.6.

Now confirm that the personal space collection mode is set to 'email'. Go to section 12.1.4.

12.1.4 Set the personal space collection mode

MT Showcase supports two methods for collecting screen items: content can be sent as an email (this is the default method) or it can be sent to media server (see section 12.4). To determine which method is used, you edit the *personal space collection mode*.

This mode is an attribute of the Personal Space widget. The Personal Space widget is only available in themes; you cannot add this widget directly to your app's structure.

To set the personal space collection mode, follow these steps:

1. Ensure your app is using a theme; see section 6.4.
2. Edit the theme; see section 6.5.
3. In the Editing a theme screen, go to the Personal Space component.
4. Confirm that the **Personal space collection mode** attribute is set to 'Email contents to user'.
5. *(Recommended)* Configure the remaining Personal space attributes. For example, you can set a title and caption to provide guidance on how to save screen content. Instructions for configuring the Personal space widget are in section 13.3.2.
6. Click the Save button to save any changes to the theme.

(Optional) You may now want to define URLs for certain media items to avoid sending them as very large email attachments. Go to section 12.1.5.

12.1.5 Define URLs for large items in media library

To avoid sending very large videos or PDFs as email attachments, you can set up MT Showcase emails to include the URL to an item instead of sending the actual item as an attachment.

For example, if your Showcase app includes an important video, you can include the URL for this video on your YouTube channel. Now, if a user adds this video to their personal space and sends it to their registered email account, this URL is included in the email instead of the actual .mp4 file.

The following rules apply:

- Images are always *attached* to an MT Showcase email. If a URL has been defined for the image in the media library, the URL is also included.

  Note: *If an image has been annotated with an infrared pen, the annotations are preserved on the image when it is sent as an email attachment.*
- Other items, especially videos and PDFs, are attached to the email *only if no URL has been defined for them*. If a URL has been defined, only the URL is included in the MT Showcase email ie, no file is attached.

To set up MT Showcase emails to include the URL to an item, you must define the URL in the media library. If a URL has been defined for an item, it is always included in the email.

Follow these steps:

1. Click the Media library button to open the media library pane.
2. In the media library, click the item you want.
3. Click the Codice URLs tab at the bottom of the media library pane.
4. Enter the URL for the item in the Codice URL input box.
5. Click the Save button.

12.2 Data Gathering service

The *Data Gathering* service collects content usage data that can be imported into third party data visualization tools such as Tableau Desktop. Usage data is stored in a PostgreSQL database as event records.

Example events include touch events (finger, hand, infrared pen, or Codice marker), opening or closing a widget, playing a video, viewing a PDF, browsing to a URL, adding items to a personal space, and emailing items from a personal space.

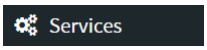
The Data Gathering service also records when an app starts and closes, when a media library asset (image, video or PDF) is opened in a widget, and when idle widgets are timed out and close automatically.

For details about the PostgreSQL database, event types, and the event record format, see the *MT Showcase Installation Manual*. Registered users can download this manual from https://cornerstone.multitouch.fi/showcase

12.2.1 Enable data gathering

To enable data gathering, you simply add the Data Gathering service to your MT Showcase app's service set. This causes MT Showcase to track content usage and generate event records whenever the app is running.

Follow these steps:

1. Click 

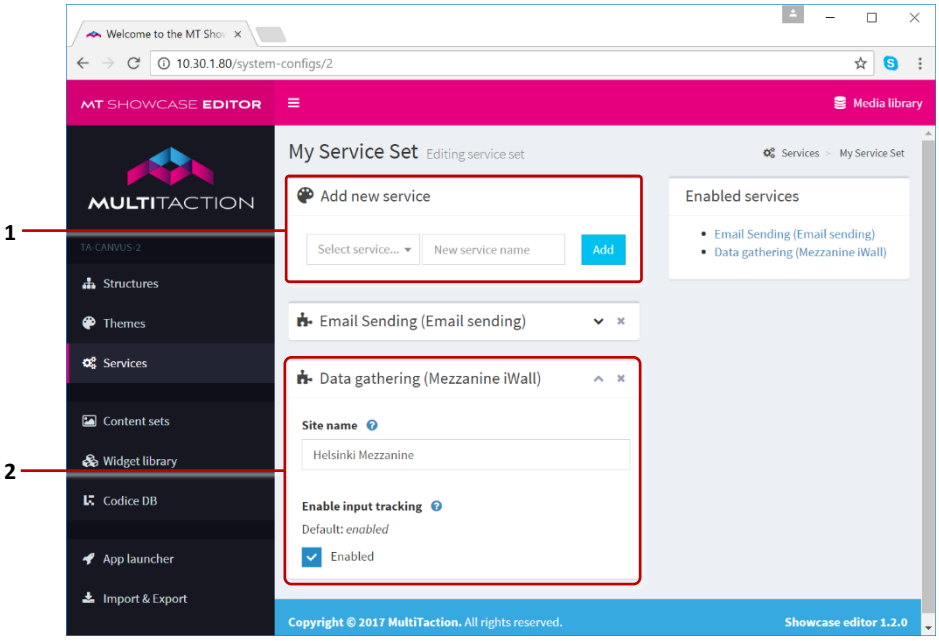
in the left-hand pane.
2. Either click an existing service set. For example, My Service Set.
3. Or create a new service set (see section 12.1.2).
4. In the *Editing service set* screen, go to the *Add new service* section:
   a. Select **Data Gathering** from the Select Service drop-down menu.
   b. Enter a name for the new data gathering service.
      We recommend you choose a name that identifies the video wall that MT Showcase runs on. For example, specify the room or office where the video wall is located.
   c. Click the Add button.
5. Still in the Editing service set screen, go to the Data Gathering section.
6. Edit the following attributes:
   - **Site name:** The site name is included in each event record and enables analysts to identify which video wall an event record is associated with.
     For example, if you run Showcase on video walls in multiple locations but store event records in a central database, you can create a Data Gathering service for each location. If you set the Site Name to be the name of the office, you can readily filter out event records for specific locations.

- **Enable input tracking:** By default, data gathering includes any user inputs detected on the video wall, including finger, pen or Codice touch events. Tracking these inputs can generate a lot of data and adversely affect app performance. You can disable this attribute to exclude input tracking from the data gathering service.

  Note: *When this attribute is disabled, the data gathering service only collects 'widget data' (for example, when widgets open or close).*

7. Click the Save button.

8. If you have not already done so, remember to add the service set containing the Data Gathering service to your app; see section 12.6.



Editing a service set screen, Data gathering service 1 *Add new service pane.* 2 *Data Gathering section. Contains the attributes for the data gathering service.*

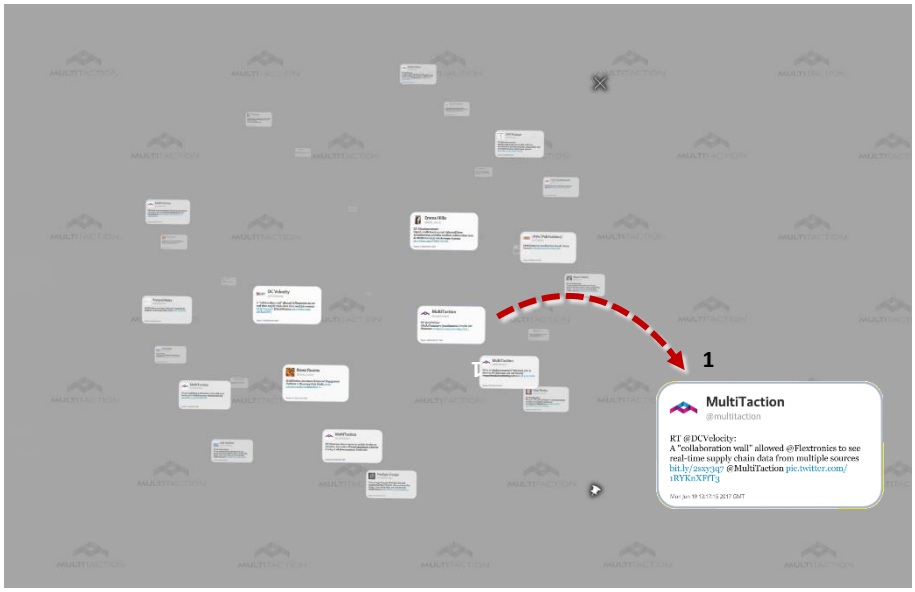
12.3 Twitter Connection service

The Twitter Connection service allows you to add Twitter feeds to your MT Showcase apps, with tweets displayed either in a cloud widget or finger menu. This service uses a Twitter app to retrieve the tweets from Twitter. You only need one Twitter Connection service to supply multiple Twitter feeds in your MT Showcase apps.

Setting up the Twitter Connection service is one step in the overall task of adding a Twitter feed to your MT Showcase app. Full instructions for adding a Twitter feed are in section 17.

After setting up the Twitter Connection service, remember to add the service set containing the Twitter Connection service to your app; see section 12.6.

Note: *You must already have created a Twitter app before you set up the Twitter Connection service. For details, see the MT Showcase Installation Manual. Registered users can download this manual from* https://cornerstone.multitouch.fi/showcase



Example Twitter cloud. **1** *Tap an individual tweet to detach and enlarge it.*

12.4 Media Server service

The *Media Server* service is used for saving screen content from a user's personal space (section 13.3) to a personal web page. This service is an alternative to the Email Sending service and has the advantage that it does not require a user's email address. Also, this web page can include screen items collected by the user from multiple video walls.

From the end-user's viewpoint, they can drag screen content into their personal space in MT Showcase and then use a QR code reader on their mobile device to download screen content they have collected on any video wall.

Note: *The URL for the user's personal web page is determined by the media server. The media server creates a unique web page for each Codice code detected by MT Showcase when users present their personal markers.*

12.4.1 Prepare the media server

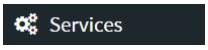
As its name suggests, this service uses a media server to host screen content collected by users. Before you add the Media server service to your app, you will need to prepare the media server. In addition, you must copy to the media server any images, videos or PDFs that are collectable in your MT Showcase app.

For full details, see the *MT Showcase Installation Manual*. Registered users can download this manual from https://cornerstone.multitouch.fi/showcase

12.4.2 Set up the Media Server service

After preparing the media server, you can set up the Media server service. You must specify which web server to use and the *personal space collection mode*.

Follow these steps:

1. Click 

in the left-hand pane.
2. Either click an existing service set. For example, My Service Set.

   Or create a new service set (see section 12.1.2).
3. In the Editing a service set screen, go to the Add new service section:
   a. Select **Media server** from the Select Service drop-down menu.
   b. Enter a name for the new Media server service. For example, you can choose a name to match the name of your web server.
   c. Click the Add button.
4. Still in the Editing service set screen, go to the new *Media Server* section.
5. You need to supply details about your media server. Specify the media **Server address** (or name) and **Server port** number.

   You also need to specify a **System name**. This name identifies the video wall that the MT Showcase is running on. For example, Helsinki Mezzanine. This name appears on the user's web page where they can download screen content. If a user collected

content from multiple video walls (for example, when visiting an exhibition), each video wall would have its own system name, allowing the user to identify where content items were collected from.

Click the help button for details.

6. Click the Save button to add the Media Server service to your service set.

7. Now specify the *personal space collection mode*.

   This mode is an attribute of the Personal Space widget. It determines how users collect the screen items in their personal space.

   a. Ensure your app is using a theme; see section 6.4.

      Note: *The Personal Space widget is only available in themes; you cannot add this widget directly to your app's structure.*

   b. Edit the theme; see section 6.5.

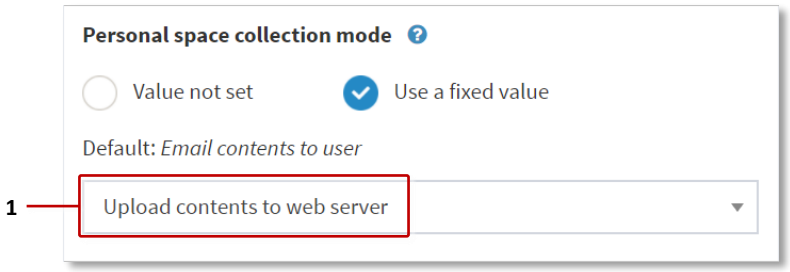
   c. In the *Editing a theme* screen, go to the **Personal Space** component.

   d. Now edit the **Personal space collection mode** attribute and select 'Upload contents to web server' from the drop-down list.

   e. *(Recommended)* Configure the remaining Personal space attributes. For example, you can set a title and caption to provide guidance on how to save screen content. Instructions for configuring the Personal space widget are in section 13.3.2.

   f. Click the Save button to save the change to the theme.

8. If you have not already done so, remember to add the service set containing the Media Server service to your app; see section 12.6.



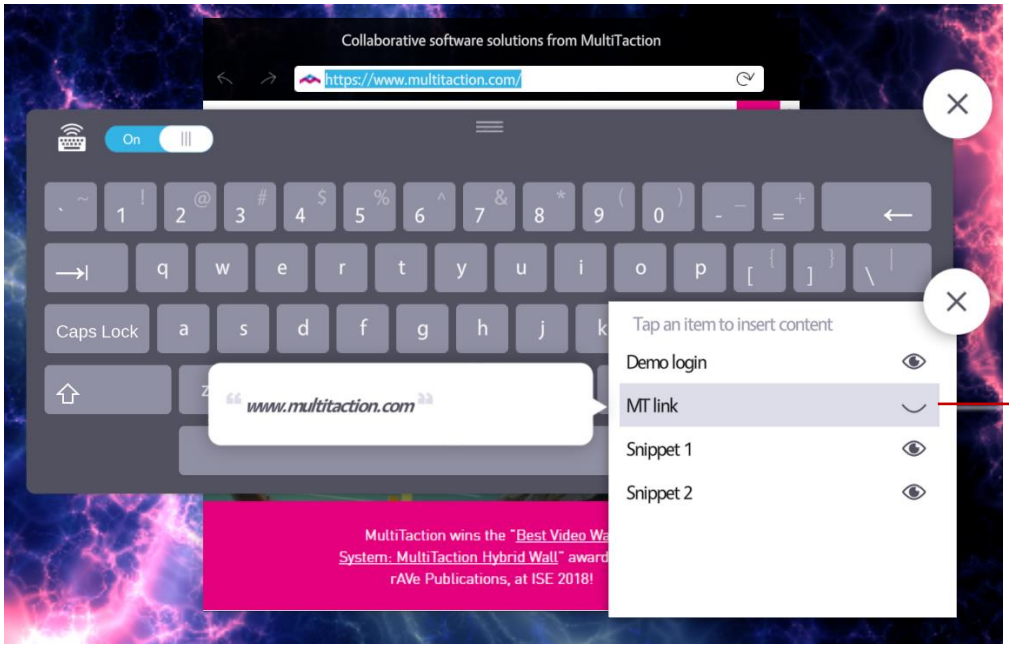
Editing a theme screen. **1** *Personal space collection mode set to 'Upload contents to web server'.*

12.5 Snippets service

The Snippets service allows you to add predefined text snippets to on-screen keyboards in your MT Showcase apps. User can tap a text snippet item to easily enter the text into a text field without manually typing it in. This is useful when you need to repeatedly enter the same text string such as a web site URL or email address.

Full instructions for adding a Snippets service are in section 24.2.

After setting up the Snippets service, remember to add the service set containing the Snippets service to your app; see section 12.6.



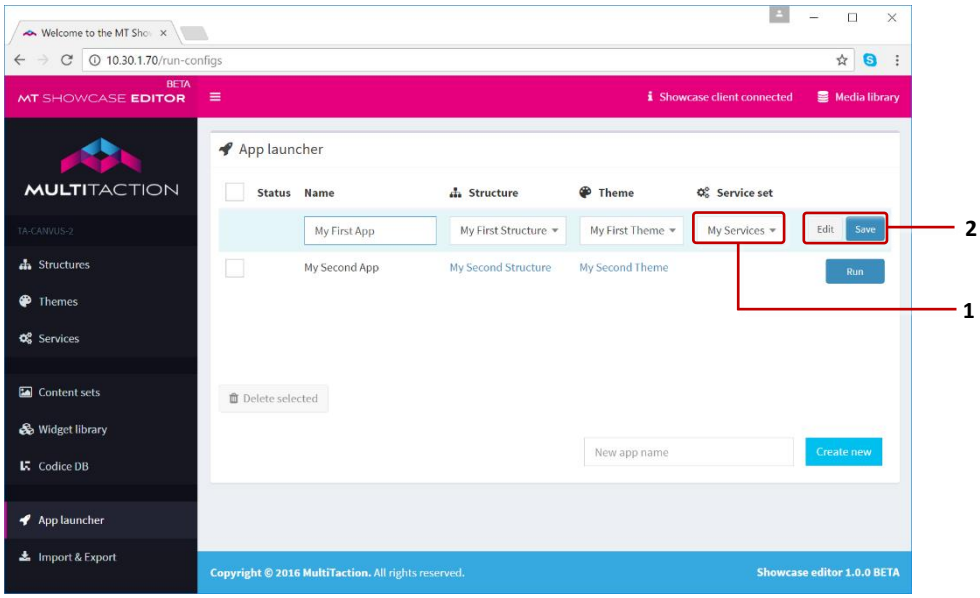
Showcase on-screen keyboard. **1** *Tap an item to enter text in the text field.*

12.6 Add a service set to your app

After setting up any services that you want to use, you must add the service set that contains these services to your app.

Follow these steps:

1. Click App launcher in the left-hand pane.
2. Click the Edit button for your app.
3. In the Service set column, choose the service set you want from the drop-down list.
4. Click the Save button for your app.



App launcher screen. 1 *Service set drop-down list.* 2 *Edit and Save buttons.*

13 Codice cards

A Codice is a simply 2D barcode (or *marker*). A Codice card is simply a Codice printed on paper or card. You designate the actions associated with each Codice.

When MT Showcase runs in table mode (section 18), Codice cards are often attached to the underside of a *tangible user interface* ie, a physical object such as a handheld plastic block or even a whiskey glass!

There are four types of Codice card:

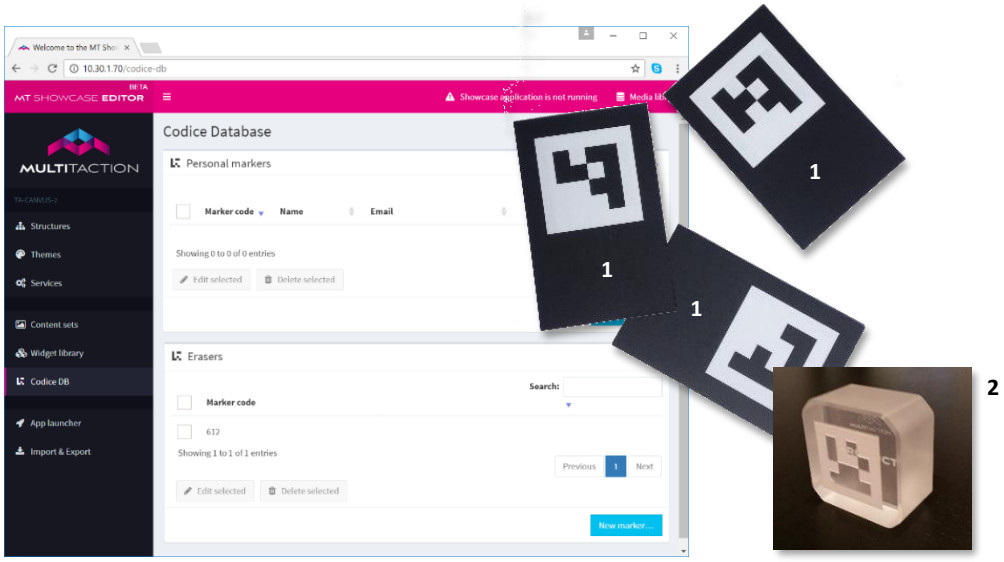
- **Personal markers:** A user has their own Codice card to access their personal space. A user's personal Codice card is more commonly referred to as a *personal marker*. See section 13.3.

  Note: *With a personal marker, a user can access their personal space from any app running on your MT Showcase server.*
- **Codice content markers:** A Codice card can be used to open *Codice content*. Any user can use a designated to launch Codice content; see section 13.4.
- **Markers that ban tweets:** A Codice card can be used to permanently remove specific tweets from a Twitter cloud or menu; see section 13.5.
- **Erasers:** You can also designate a Codice as an *eraser*. Any user can use an eraser card to erase pen or finger annotations from an MT Showcase app. See section 13.6.

Before your users can start using Codice cards, you must:

1. Add a Codice detector to your app. This widget allows the app to detect when Codice cards are held against the screen. See section 13.2.
2. Register each Codice before it can be used in an MT Showcase app. After registering the Codices. See section 13.2.

You register Codices for individual users and designate erasers in the *Codice DB* screen.



Codice Database screen. **1** *Example Codice cards.* **2** *Codice as a tangible user interface.*

13.1 Identify Codice codes with the Input Visualizer

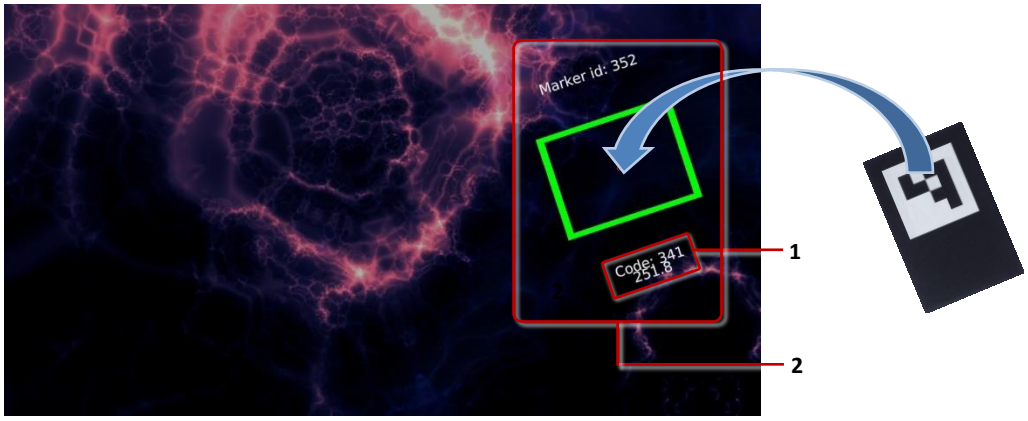
Before you can register individual Codices, whether as personal markers or erasers, you need to know the Codice's code number. This section describes how to identify these codes if you do not know them already.

It is possible to manually calculate the code from the binary value of the 2D barcode. However, it is far easier to use the Input Visualizer widget. Follow these steps:

1. Add the Input Visualizer to the **Overlay** layer of your app's structure.
   a. Click Structures 

in the left-hand menu.
   b. Click the structure you want to use. For example, My First Structure.
      The *Editing a Structure* screen appears.
   c. Go to the **Overlay** section and click the *Add a new widget to Overlay* hyperlink.
   d. Add the Input Visualizer widget.
   e. Save the changes to the structure.

   Tip: *The advantage of adding the Input Visualizer to the Overlay layer is that you do need to edit any widget attributes; all screen touches are visualized, anywhere on the screen. Conversely, if you added the Input Visualizer to the Main layer, you would then need to edit its attributes to make the Input Visualizer easier to locate.*
2. In the app, hold a Codice card against the screen over the Input Visualizer widget.
   The display now reveals the Codice code. Make a note of the code.
3. When you have finished identifying Codice codes, *delete the Input Visualizer from the structure of your app!*
   In the *Editing a Structure* screen, right-click the Input Visualizer and click Delete.



Input Visualizer example. With the Input Visualizer in the overlay layer, you can hold the Codice card anywhere on the screen to see the Codice code. **1** *Codice code number.* **2** *Marker visualization.*

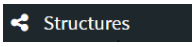
13.2 Add a Codice detector

The Codice detector is a special widget that enables your app to detect when a user holds a personal marker or an unregistered Codice card against the screen:

- **Personal markers:** If a user presents a registered personal marker, the app opens a personal space or launches Codice content.
- **Unregistered Codice cards:** If a user presents an unregistered card, the app opens a temporary personal space that allows the user to self-register ie, the user can gather items, enter an impromptu email address, and send the items to this address. The user also has the option to permanently register their Codice card and email address.

If your app does not include a Codice detector, it will only recognize erasers. It will ignore all other Codice cards. (Erasers are always recognized and do not need the Codice detector.)

Follow these steps:

1. Click Structures 

in the left-hand pane.
2. Click My First Structure to open the *Editing a Structure* screen opens.
3. Go to the Main section and click *Add a new widget to Main*.
4. Add the Codice detector widget.
5. In the *Codice detector* attributes pane, edit these attributes:
   - **Rotate personal space towards Codice** and **Codice rotation offset:** These attributes are used when MT Showcase runs in table mode (see section 18.2).
   - **Allow unregistered Codice markers?** This attribute determines how your app handles unregistered Codice cards. If this attribute is enabled, the app opens a temporary personal space that allows the user to self-register (see above). If it is disabled, the app ignores the Codice card.

     To enable this attribute, select 'Use a fixed value' and then select the Enabled check box.
   - **Collect widgets with Codice:** Users can place a Codice marker on a widget to add it to their personal space. This attribute determines what happens to the widget. Choose the option you want from the drop-down list. Click the help button for details.

   Note: If you want to add a toolbar to a personal space, you must edit the Personal Space widget; see section 13.3.2.
6. Click Save to save the changes to the structure.

13.3 Personal space

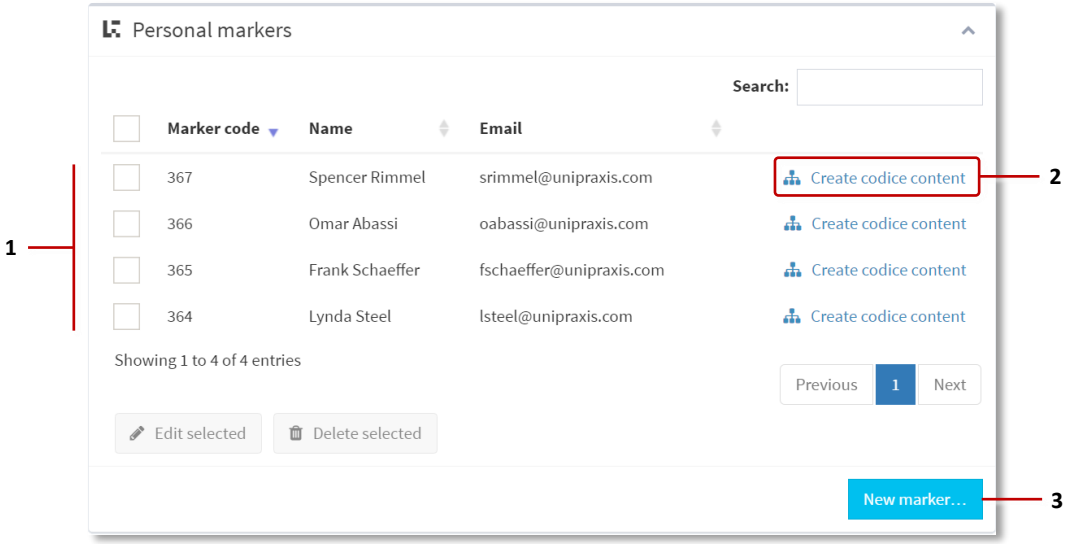
A *personal space* (also called a *personal folder*) provides a simple method for exporting screen content out of MT Showcase. It opens automatically when a user presents their personal marker against the screen.

13.3.1 Register personal markers

To enable an individual user to access their personal space in an MT Showcase app, you must assign a *personal marker* to that user. A personal marker is a Codice card that has been registered in the Codice database.

To register a personal marker, follow these steps:

1. Click Codice DB in the left-hand pane.
2. In the *Codice Database* screen, go to the Personal Markers pane.
3. Click the New Marker button.
4. Enter the following details:
   - **Marker Code:** Enter the code for Codice assigned to the user.
   - **Name:** Enter the user's name. This name can be formal or informal. It is used as a greeting when the user opens their personal space in an MT Showcase app.
   - **Email:** Enter the user's email address. If the user sends items to themself from their personal space, the items will be sent to this address.
5. Click the Save button.
6. Print and distribute the new personal marker to the user.
7. The user can now access their personal space as described in section 13.3.4.



Codice Database screen, Personal markers pane.
1 *Registered Codices.* **2** *Create codice content hyperlink.* **3** *New Marker button.*

13.3.2 Configure the Personal Space widget

The Personal Space widget opens when a user holds their personal marker against the screen. You configure the widget by editing its attributes. For example, you can set the *personal space collection mode*. You can also specify a title and caption to provide the user with guidance on how to export screen content from the MT Showcase app.

The Personal Space widget is only available in themes; you cannot add this widget directly to your app's structure. To configure the widget, follow these steps:

1. Ensure your app is using a theme; see section 6.4.
2. Edit the theme; see section 6.5.
3. In the Editing a theme screen, go to the Personal Space component.
4. Edit the attributes, as required.
   - **Personal space collection mode:** This attribute determines which method is used for collecting screen content. Users can send content to their email address (this is the default method) or they can send content to a personal web page on a media server. If you choose:
     - 'Email contents to user', the personal space includes an envelope button that users tap to send items to their email address.
     - 'Upload contents to web server', the personal space includes a QR code that links to the user's personal web page on the media server. Users can use a QR code reader on their mobile device to open this web page and download any items in their personal space. (The URL for this web page is also shown.)
   - **Personal space title:** Specify a title for the widget. For example, Welcome to your personal space.
   - **Personal space info text:** The info text is a caption that displays in the personal space. For example, you can add text that provides guidance on how to email or upload screen content.

     The info text can include a <link> variable. If the collection mode is set to:
     - 'Email contents to user', <link> adds the user's registered email address.
     - 'Upload contents to web server', <link> adds the URL for the user's personal web page where they can download screen items.

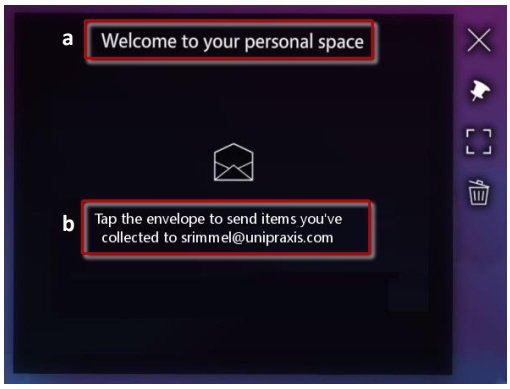
     See the examples in section 13.3.3.
   - **Rotate widget towards hand:** This attribute is used when MT Showcase runs in table mode; see section 18.2.
   - **Toolbar**: You do not need to add a toolbar to the Personal Space widget, but users may find it convenient if it has a toolbar with Pin and Close buttons. Select 'Use a fixed value' and then select an existing toolbar or create a new one (see section 3.5.1).
5. Click the Save button to save any changes.

13.3.3 Title and info text examples

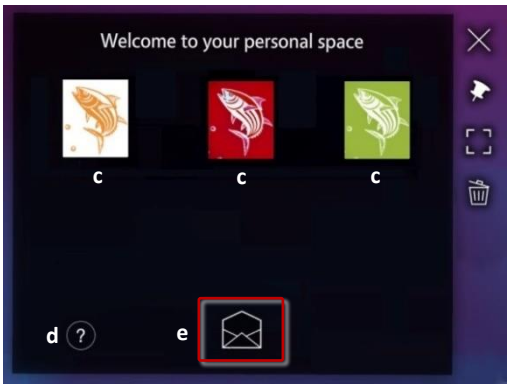
In examples **1** and **2** below, the **personal space collection mode** is set to 'Email contents to user'. You can customize the title and info text accordingly. In the examples below, the info text is "*Tap the envelope to send items you've collected to <link>*" where *<link>* displays the email address registered to the user's personal marker.

If the personal space is empty, the info text displays automatically. If the personal space contains items collected by the user, the user can tap the ? info button to display the info text in a popup.

1



2

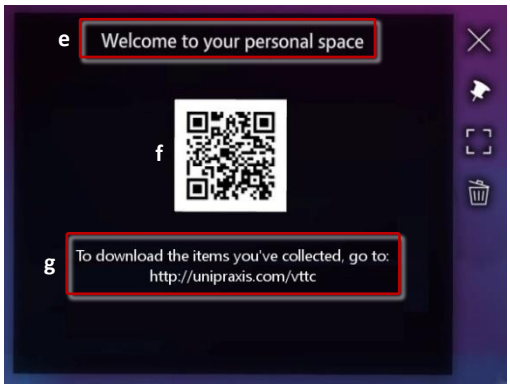


1 *Empty personal space.* **2** *Personal space with screen content.* **a.** *Title.* **b.** *Info text customized for emails.* **c.** *Screen item collected by user.* **d.** *Info button.* **e.** *Envelope button*

In example **3** and **4** below, the **personal space collection mode** is set to 'Upload contents to web server'. The personal space includes a QR code (and URL) that links to the user's personal web page.

In the examples below, the info text is "*To download the items you've collected, go to <link>*", where *<link>* displays the actual URL for the user's personal web page.

3



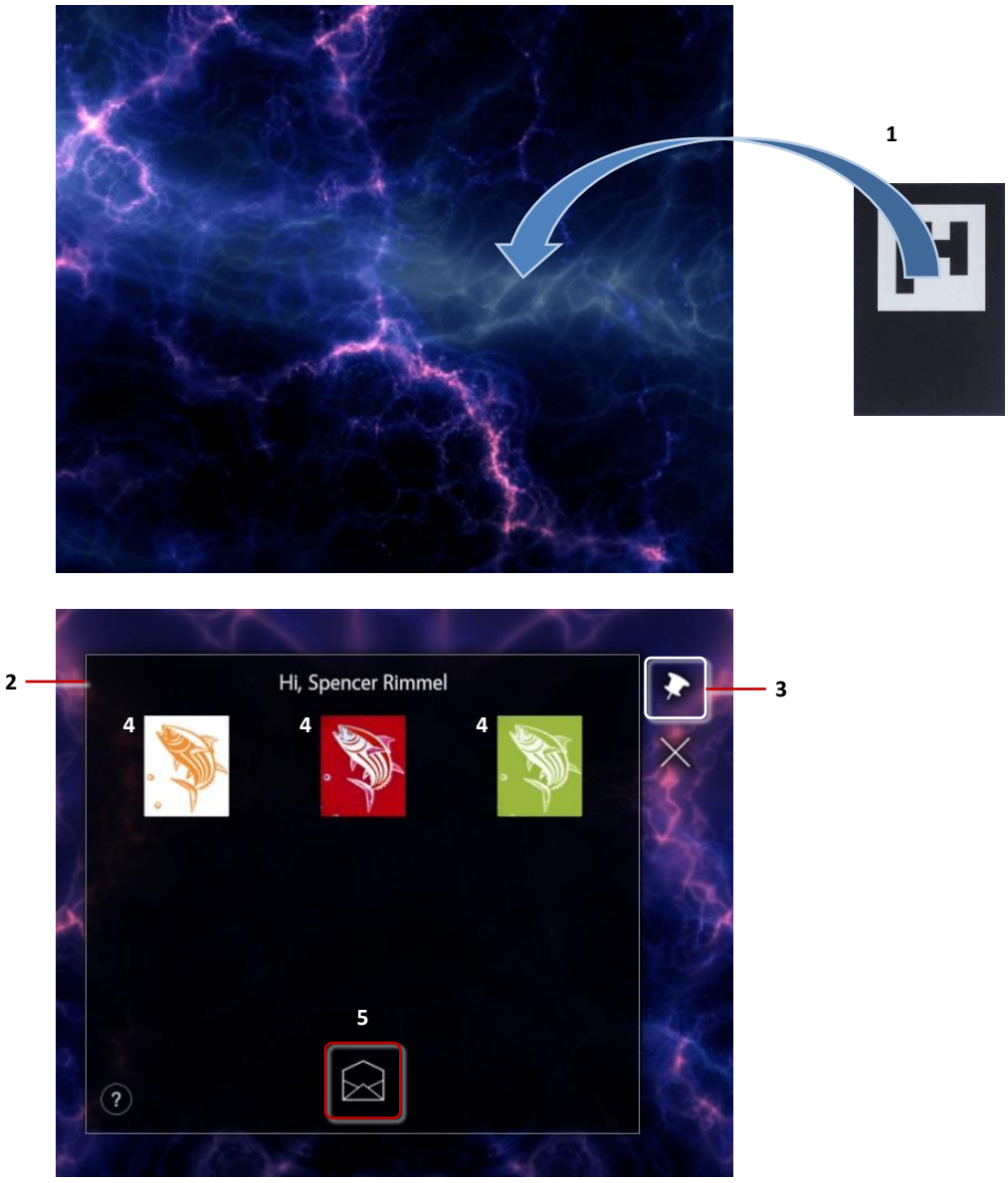
4

3 *Empty personal space.* **4** *Personal space with screen content.* **e.** *Title.* **f.** *QR code for the user's personal web page.* **g.** *Info text customized for uploads.* **h.** *Upload status indicator. This shows whether is the item is available to be downloaded from the web page.* **i.** *QR code and URL for the user's personal web page.* **k.** *Info button.*

13.3.4 Drag items into a personal space

The user simply holds their personal marker against the screen to open their personal space. They can then drag any screen items into this folder from the MT Showcase app. To remove an item, the user simply drags it out of their personal space.

Note: *With their personal marker, a user can access their personal space from any app running on your MT Showcase server.*



Using a personal space. 1 *The user holds their personal marker face down as shown.*
2 *Personal space.* 3 *Pin button in personal space toolbar.* 4 *Screen items collected by the user.*
5 *Email button (only appears if 'Personal space collection mode' is set to Email).*

13.3.5 Send email attachments (or URLs)

Users can send items in a personal space to a registered email address.

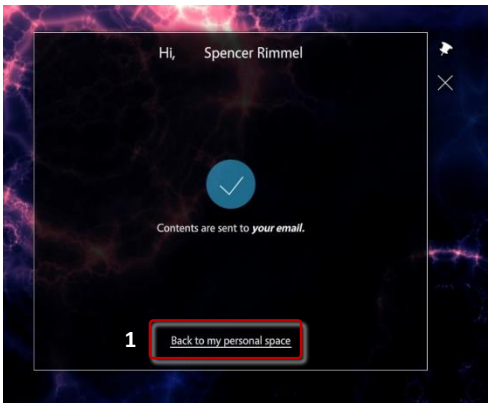
Note: *For very large videos or PDFs, the email can include a URL instead of an attachment. For details about what an email contains, see section 12.1.1.*

The user must follow these steps:

1. The user holds their personal marker against the screen to open their personal space.
2. In the personal space, the user taps the email button and then waits while all items are sent as email attachments (or URLs).

Sending items as email attachments. **1** *Items in personal space.* **2** *Email button.* **3** *Wait while items are sent.*

3. The user taps the *Back to my personal space* hyperlink to return to their personal space.



Sending items as email attachments. **1** *'Back to my personal space' hyperlink.*

13.3.6 Save items to a personal web page

Users can save items to a personal web page. When a user drags a screen item into their personal space, the item is automatically added to the user's personal web page on your media server. The user can use a QR code reader to download the item to their mobile device or personal computer.

This method is an alternative to sending screen items as attachments and has the advantage that it does not require a user's email address. Also, the user's personal web page can include screen items collected from multiple video walls.

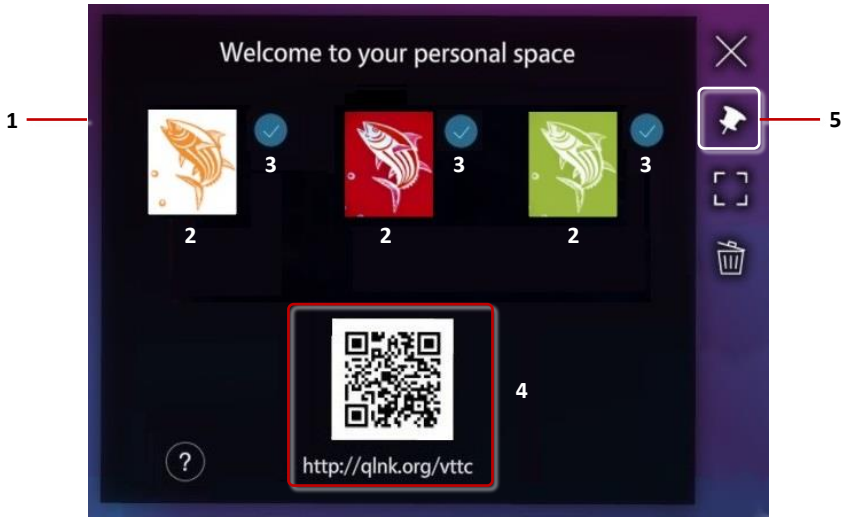
Note: *The URL for the user's personal web page is determined by the media server; see section 12.4.*

The user must follow these steps:

1. The user holds their personal marker against the screen to open their personal space.
2. The personal space includes a QR code that links to the user's personal web page on the media server. Using a QR code reader on their mobile device, the user can open this web page and download any items in their personal space.

   (The personal space also displays the URL for this web page, allowing the user to browse to the web page directly.)

Information about setting up a media server to support personal web pages is available in the *MT Showcase Installation Manual*. Registered users can download this manual from https://cornerstone.multitouch.fi/showcase



Saving screen items to a personal web page. 1 *Personal space.* 2 *Screen items collected by the user.* 3 *Upload status indicators. These shows whether the item is available to be downloaded from the web page.* 4 *QR code and URL for the user's personal web page.* 5 *Pin button in personal space toolbar.*

13.4 Codice content

Codice content refers the item that opens automatically when a user presents a personal marker (ie, holds it against the screen). This item could be an image, video or even a web site. In fact, all the main 'content widgets' are supported, such as image viewers, web browsers, or cloud widgets.

Codice content is assigned to a specific Codice code. The content is available to anyone who presents a Codice card with the correct code.

For example, you may decide to associate Codice 99 with a video. After setting up the Codice content, you simply print a set of markers (ie, Codice cards) bearing Codice 99 and distribute them to your users. When any user presents a 'Codice 99' marker, the video launches automatically.

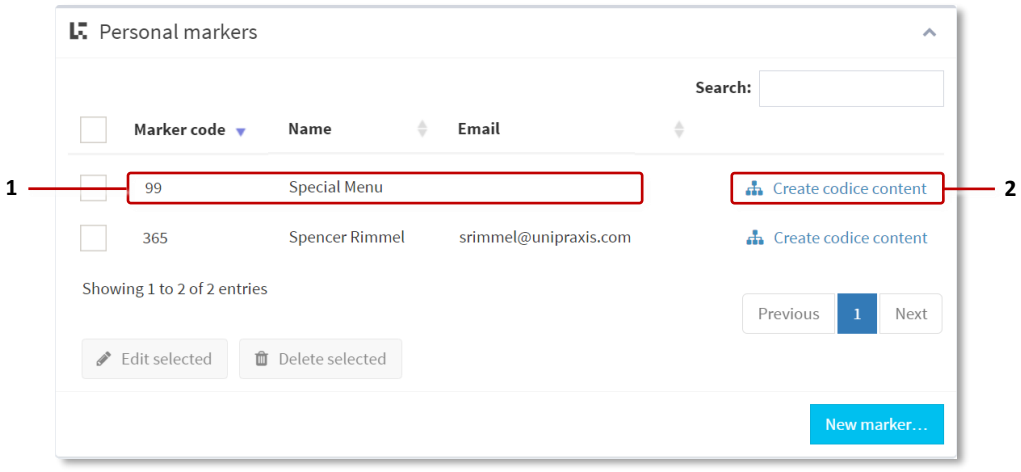
You set up Codice content in the *Codice Database* screen. Follow these steps:

1. Click Codice DB in the left-hand pane.
2. In the *Codice Database* screen, go to the **Personal Markers** pane.
3. Specify the marker that you want to associate with Codice content.

   For example, if you want to use Codice 99 to open a finger menu, add a new marker with these details:

   Marker code: 99
   Name: Special menu
   Email: *You do not need to provide an email address.*
4. Click the *Create codice content* hyperlink.

   This opens the *Editing a structure* screen.



Codice Database screen, Personal markers. **1** *New 'Codice content' marker. There is no need to assign an email address to this marker.* **2** *Create codice content hyperlink.*

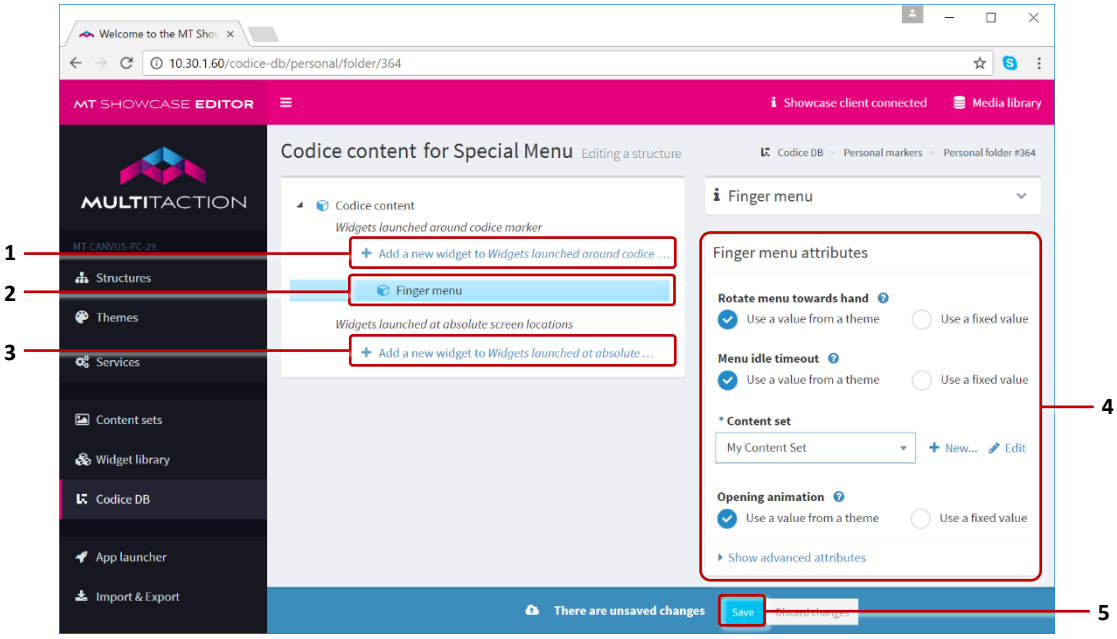
5. Edit the Codice content attributes as required.

   - **Move contents with the Codice**: Enable this attribute to control the launched content widget by moving and rotating the Codice. For most widgets this means that when a user moves a Codice, the widget follows it. Some widgets have special interaction with Codices. If you launch a menu with a Codice, you can rotate the Codice to highlight and select menu nodes. Leave the Codice still for a moment to launch the highlighted node's content. You can fine-tune or disable this behavior in the menu's attributes.

     Note: Codice control only applies to widgets launched around the Codice marker. It does not affect widgets launched at absolute screen locations.

   - **Dismiss content when Codice is lifted:** Enable this attribute to automatically close the content when the Codice is removed from the screen.

   - **Reverse mode:** Normally content is launched when a Codice is placed on the screen, and optionally closed when the Codice is lifted. Enable this attribute to reverse the behavior, so content is launched when a Codice is lifted from the screen, and optionally closed when the Codice is replaced.



Example Codice content in the Editing a structure screen. 1 *Hyperlink to add 'widgets launched around Codice marker'.* 2 *Launched widget. This example launches a finger menu when a user presents the relevant Codice marker.* 3 *Hyperlink to add 'widgets launched at absolute screen locations'.* 4 *Attributes for the launched widget.* 5 *Save button.*

6. Now define the *Widgets launched* lists. These widgets open when a user presents the relevant marker. They can be any item normally found in the Main or Menu layers of an app's structure (for example, an image, video, PDF or menu). You typically only launch a single widget but, if required, you can launch multiple widgets.

There are two widget lists, each with its own interpretation of the widget **Location** attribute. You can add widgets to either list, or to both lists:

- **Widgets launched around Codice marker:** For a widget in this list, the Location attribute is relative to *the position of the marker on the screen*. Use this list if you want a widget to open below or next to the marker.

  For example, if the location of an image widget is 0,0, then the image will always open directly below the marker.

- **Widgets launched at absolute screen locations:** For a widget in this list, the Location attribute is relative to *the top left corner of the screen*. Use this list if you want a widget to always open at the same location on the screen.

  For example, if the location of an image widget is 0,0, then the image will always open in the top left corner of the screen.

In the *Editing a structure* screen:

a. Click either of the two *Add a new widget* hyperlinks.

b. Add the widget you want.

c. Set the **Location** attribute. See above for details of how this location is interpreted.

d. Set the remaining attributes for the new widget. If necessary, use section 2.9 for guidance on how to edit widget attributes.

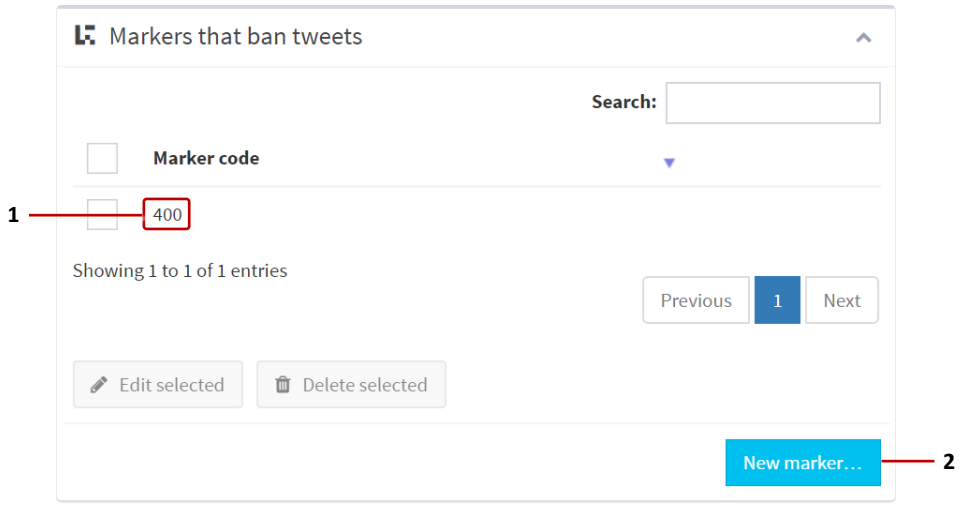
7. Click the Save button.

13.5 Markers that ban tweets

If your app includes a Twitter feed, either in a Twitter cloud or finger menu, there may be occasions when the feed retrieves and displays tweets that are inappropriate. To remedy this, you can designate a Codice as a 'tweet blocker'. This enables users to remove such tweets from the app by simply tapping them with the tweet blocker card.

You can designate any Codice as a tweet' blocker. Follow these steps:

1. Click Codice DB in the left-hand pane.
2. In the *Codice Database* screen, scroll to the **Markers that ban tweets** pane.
3. Click the New Marker button.
4. In the Marker Code column, enter the code for the Codice you want to designate as a tweet blocker.
5. Click the Save button.
6. Print and distribute the new tweet blocker card.



*Codice Database screen, Markers that ban tweets. **1** New 'tweet blocker' marker code. In this example, Codice 400 is designated as the tweet blocker. **2** New marker button.*

13.5.1 Are tweets permanently removed?

No. If you restart the MT Showcase client, a banned tweet can potentially reappear in the Twitter feed if it has not been supplanted by more recent feeds.

Note: *The client displays MT Showcase on your video wall. Instructions for restarting the client are in the MT Showcase Installation Manual. See also step 3 in section 3.2.*

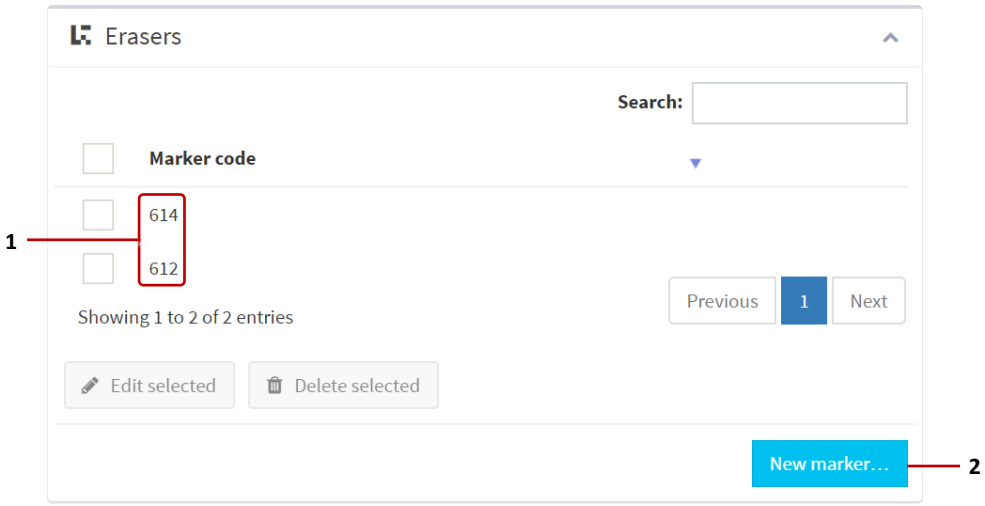
13.6 Erasers

You use *eraser cards* to erase pen or finger annotations from an MT Showcase app.

By default, Codices 612 and 614 are designated as *erasers*. You simply print either of these Codices onto paper or card to create *eraser cards*. You then distribute these cards to your users. Any user can use an eraser card to erase on-screen annotations.

You do not normally need to change the default eraser numbers but, if required, you can designate any Codice as an eraser. Follow these steps:

1. Click Codice DB in the left-hand pane.
2. In the *Codice Database* screen, scroll down to the **Erasers** pane.
3. Click the New Marker button.
4. In the Marker Code column, enter the code for the Codice you want to designate as an eraser.
5. Click the Save button.
6. Print and distribute the new eraser card.



*Codice Database screen, Erasers. **1** New eraser marker codes. In this example, Codice 612 and 614 are designated as erasers. **2** New marker button.*

13.7 Create your own Codice cards

If required, you can download ready-made sets of Codice markers in PDF format. Alternatively, you can generate your own Codice markers in PNG format using the MultiTaction MarkerFactory application. In either case, you can then print the markers onto paper or card and distribute to your users. For full details about creating your own Codice markers, see the *MultiTaction Cell User Manual*. This manual is available on the MultiTaction web site:
https://cornerstone.multitouch.fi/

14 Toolbars

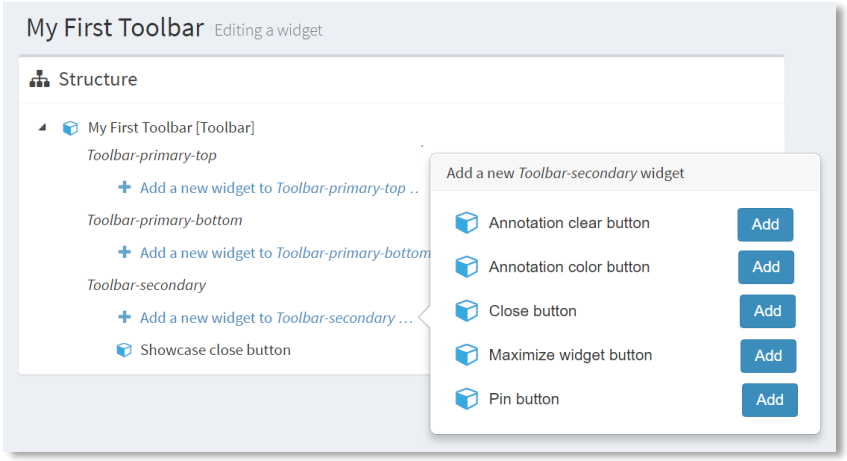
It's often useful, or even necessary, to add toolbars to widgets in your app. Toolbars can include Close, Pin and Maximize buttons, plus Color and Clear buttons for widget annotations.

14.1 Adding a toolbar

This section briefly summarizes how to add a new or existing toolbar to a widget. For a detailed example on how to add a new toolbar to a web browser, see section 3.5.

To add a toolbar:

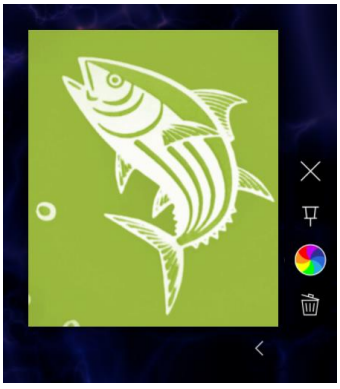
1. Go to the *Editing a structure screen* and select the widget.
2. In the widget attributes pane, go to the **Toolbar** attribute.
3. Add a new or existing toolbar:
   - **Existing toolbar:** Select an existing toolbar from the drop-down-menu.
   - **New toolbar:** Click the Add button to create a new toolbar. For example, My First Toolbar. Then click the Edit button to add buttons to the toolbar.



Editing a Toolbar Widget screen and pop-up of available buttons

4. Save the changes to the widget. The widget now displays a toolbar in your app.

Image viewer without toolbar

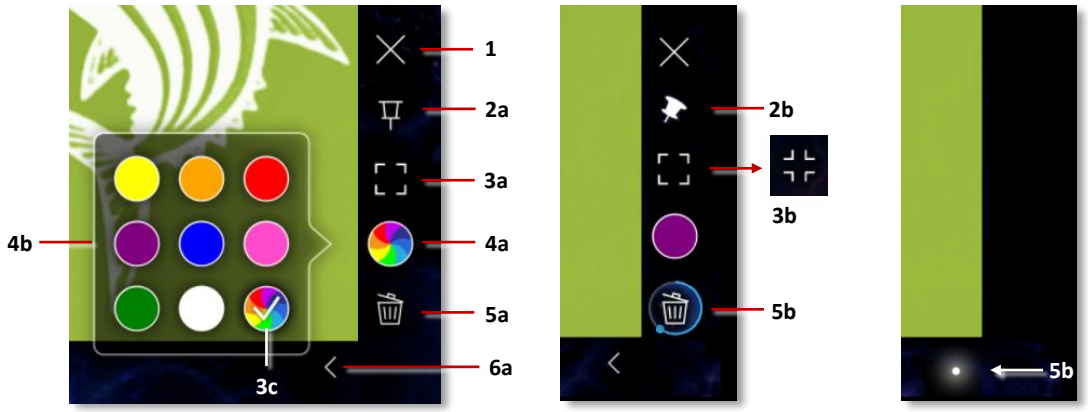


Same widget with toolbar

14.2 Toolbar buttons

Widget toolbars can include the following buttons:

- **Close:** Closes the widget.
  Tip: *You can also remove widgets by dragging them off the screen.*
- **Pin:** Pins or unpins the widget; see section 3.5.3.
- **Maximize:** Expands the widget to fill the entire screen or, if configured, the closest maximization area; see section 25.5.
- **Color:** Opens a color palette for annotations drawn on the widget. The palette includes an option for random colors.
- **Clear:** Clears any annotations drawn on the widget. Tap and keep pressing this button to clear all annotations on the widget.
  Tip: *Keep pressing the Clear button until the circular progress bar completes.*
- **Hide:** *Only available on secondary toolbars.* Hides the toolbar.
  When a toolbar is hidden, this button is replaced by the unhide button (a subtle white dot). Tap the dot to unhide the toolbar.



Toolbar buttons and tools
1 *Close button.*
2a *Pin button.* 2b *Unpin button.*
3a *Maximize button.* 3b *Unmaximize button. Available only when widget is maximized.*
4a *Annotation color button.* 4b *Annotation color palette.* 4c *Random color option.*
5a *Clear button.* 5b *Circular progress bar; keep pressing this button until the progress bar completes.*
6a *Hide toolbar button.* 6b *Unhide toolbar button.*

14.3 Primary and secondary toolbars

When you create a new toolbar, you can create a primary or secondary toolbar. Primary toolbars are permanently visible. Secondary toolbars include buttons to hide and unhide the toolbar. Primary and secondary toolbars are both optional.

Toolbar widgets can display buttons in three positions:

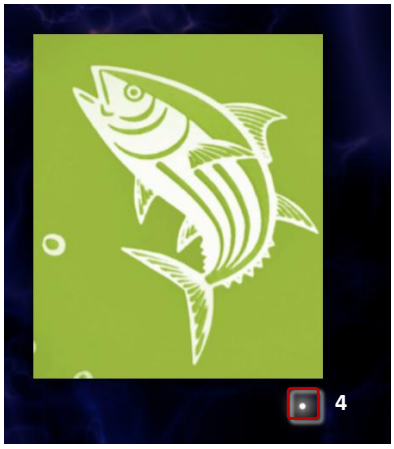
- **Toolbar-primary top:** Buttons are on the top-right of a widget. These toolbars are permanently visible.
- **Toolbar-primary-bottom:** Buttons are on the bottom -right of a widget. These toolbars are permanently visible.

1 *Primary-toolbar-top*

2 *Primary-toolbar-bottom*

- **Toolbar-secondary:** Buttons are on the bottom -right of a widget. This toolbar can be optionally hidden by the user.

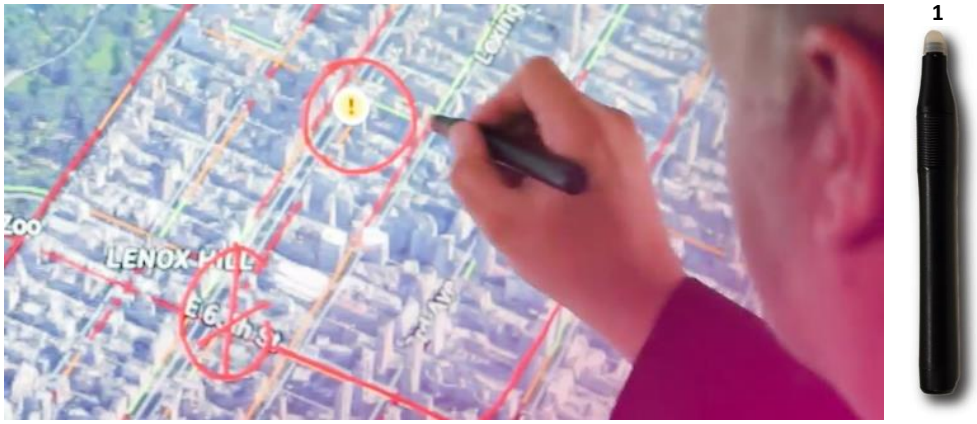
3 *Secondary-toolbar, buttons visible*



4 *Secondary toolbar, buttons hidden*

15 Annotations

The Annotation feature allows users to draw directly on the app background or on individual widgets with their finger or an infrared pen.



Use infrared pens to draw sketches and captions. 1 *The pens supplied with MT Showcase have a touch-activated infrared LED in their tip.*

15.1 Background annotation or widget annotation?

In an MT Showcase app, you can enable annotations on the app's background and on individual widgets. Background annotations do not overlap screen items in the app's main layer and users cannot change the annotation color.

Conversely, widget annotations *can* extend over the background (if the stroke starts over the widget). Also, users can set the annotation color and delete annotations with the Color and Clear toolbar buttons.

1 *Background annotations do not overlap widgets.* 2 *Widget annotations can extend over the app background.* 3 *Color button* 4 *Clear button. These buttons apply to widget annotations only.*

15.2 Enable annotations

In an MT Showcase app, you can enable annotations on the app's background and on individual widgets.

15.2.1 Enable background annotations

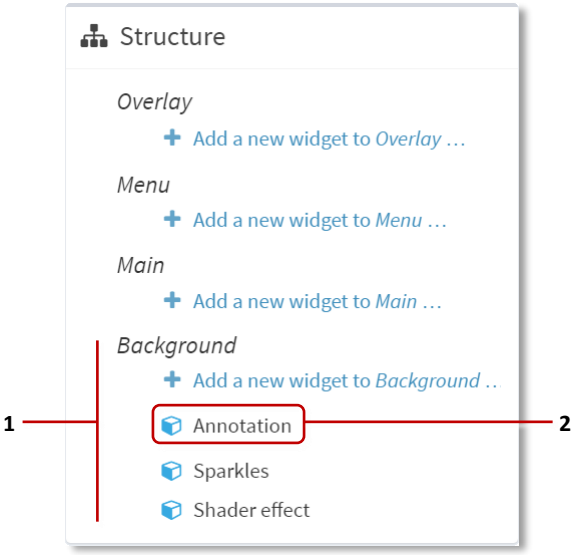
These annotations display on the app background only.

Note: *You cannot draw background annotations on top of items in the app's main layer (ie, on top of screen items such as an image or web browser). If any background annotations overlap these screen items, the annotations display behind these items.*

Follow these steps

1. Edit a structure.
   a. Click Structures 

in the left-hand menu.
   b. Click the structure you want to edit. For example, My First Structure.
   The *Editing a Structure* screen appears.
2. Add the annotation widget to the background level.
   a. Go to the Background section and click *Add a new widget to background*.
   b. From the pop-up Background widget menu, add the Annotation widget.
   c. Make sure the Annotation widget is at the top of the list of background widgets.
   Important! *Annotations will only work in the background layer if the Annotation widget is topmost!*
   d. Click the Save button.



Editing a Structure screen. 1 *Widgets in background layer.*
2 *Annotation widget is top of the list.*

3. Edit the Annotation widget attributes:
   - **Default color** and **Use random colors:** These attributes set the color of the pen or finger stroke. *Users cannot override these colors.*

     For guidance on specifying colors, see section 21.4.
   - **Enable drawing with pens** and **Enable drawing with fingers:** When enabled, these attributes allow users to draw on the app background.
   - **Annotation lifetime:** This attribute defaults to 2 minutes. When the lifetime expires, any annotations on the app background automatically fade and disappear. If you set the lifetime to zero seconds, annotations never fade and can only be removed with an eraser card (see section 15.3).
4. Click the Save button.

15.2.2 Enable annotations on a widget

These annotations are associated with a specific widget only.

Note: *When drawing annotations with a pen or finger, the stroke must start over the widget. You can draw a stroke that extends beyond the widget onto the app background. However, you cannot start a stroke on the background and extend it over the widget.*

Follow these steps

1. Edit a structure.
   a. Click Structures 

in the left-hand menu.
   b. Click the structure you want to edit. For example, My First Structure.

      The *Editing a Structure* screen appears.
2. Add an Annotation widget.
   a. Go to the Main section and click the widget you want, such as an Image Viewer widget.
   b. In the widget attributes pane, go to the **Annotation** widget.
   c. Click the New button.
   d. In the *Add a new Annotation widget* pop-up, add the Annotation widget.

      Add a new Annotation widget pop-up
   e. Type a name for the new Annotation widget. For example, Annotations for Widgets.
   f. Click the Save button.

3. Edit the new Annotation widget
   a. In the widget attributes pane, go to the **Annotation** widget.
   b. Click the Edit button to display the *Editing a widget* screen.
   c. Edit the following attributes:
      - **Default color** and **Use random colors:** These attributes set the color of the pen or finger stroke. Users can override these colors if you add a toolbar with a Color button. For guidance on specifying colors, see section 21.4.
      - **Enable drawing with pens** and **Enable drawing with fingers:** When enabled, these attributes allow users to draw on the widget.
      - **Annotation lifetime:** This attribute defaults to 2 minutes. When the lifetime expires, any annotations on the widget automatically fade and disappear. If you set the lifetime to zero seconds, annotations never fade and can only be removed with an eraser card or a toolbar Clear button (see section 15.3).
   d. Click the Save button.

15.3 Remove annotations

There are three ways to remove annotations:

- User can erase annotations with an eraser card; see section 13.5.
- You can configure the **Annotation lifetime** attribute to automatically erase annotations after a specified period.
- (*For widget annotations only*) If a widget has a toolbar, you can include the Clear button to allow users to manually erase the widget's annotations; see section 14.2.

16 Sounds

Sounds (or *sound effects*) can be an important part of your users' MT Showcase experience. For example, playing a sound when a widget closes may provide feedback for users, or you can use sounds to reinforce your branding or for entertainment purposes.

MT Showcase apps can play a sound (a WAV file) when a widget opens and closes, and when a user touches the widget with their hand or infrared pen. Apps can also loop a sound continuously while a widget is open, or while the app is running.

MT Showcase supports *interaction sounds* and *widget sounds*:

- **Interaction sounds:** You can play a WAV file when a user touches a widget with their hand or infrared pen. If you add multiple sounds, one sound is played at random each time. You can assign interaction sounds to individual widgets, or you can set global interaction sounds for all widgets.
- **Widget sounds:** You can add a WAV file that plays when a user opens or closes a widget, or that loops while a widget is open. You can also loop a background sound continuously while the app is running.

  If you add multiple Open and Close sounds, one sound is played at random each time. If you add multiple Background sounds, they play in a random sequence by default, but you can choose to play them in a fixed sequence.

In both cases, you can set the volume for these sounds and, if set up for your video wall, enable *positional audio* (sounds are directed to the nearest speaker).

16.1 Guidelines

- For optimum impact, sound effects should be used sparingly and appropriately. If you add too many sounds to your app, especially sounds that loop while a widget is open or while the app is running, your users may find the audio output excessive or distracting.
- By default, when you add a widget to your app's structure, its sounds are set to 'None'. However, if you define default sounds in a theme (by editing the *Widget Behavior* theme default; see section 6.6), then sounds are effectively 'switched on' for all widgets in your app.
- Keep all touch sounds and open/close sounds very short (less than one second). We also recommend that you make them non-verbal, moderate volume, and pleasant, not jarring. For example, use subtle beeps or clicks to provide users with auditory feedback.

Useful sound guidelines are also available on the *Windows Dev Center* web site: msdn.microsoft.com/en-us/library/windows/desktop/dn742487(v=vs.85).aspx

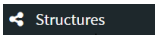
16.2 Add sounds to your media library

Before you can add sounds to your app, you need to drag the WAV files into your media library. For instructions on adding items to your media library, see step 1 in section 5.1.

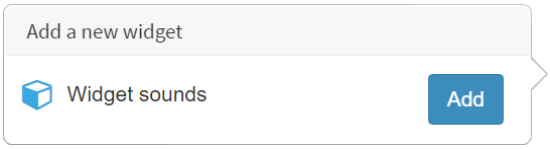
Note: *MT Showcase does not currently support MP3 files. If you want to use MP3 files, you will need to use a media converter app or website to convert them to WAV files.*

16.3 Add sounds to an individual widget

This section describes how to create two shared widgets (*interaction sounds* and *widget sounds*) and assign them to an individual widget. For example, to assign sounds to the web browser in My First Structure (see step 3 in section 3.3), follow these steps:

1. Edit the widget attributes:
   a. Click Structures 

in the left-hand menu.
   b. Click the My First Structure hyperlink.
   c. In the Main section, click the Web browser widget.
2. Add a new *Widget sounds* widget.
   a. In the Web browser attributes pane, go to the **Widget sounds** attribute.
   b. Click the New button.
   c. In the *Add a new widget* pop-up, click the Add button.



Add a new widget pop-up for 'Widget sounds'

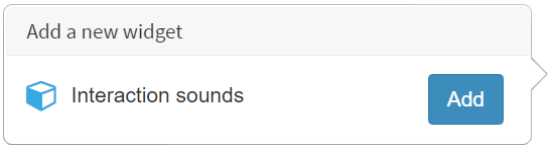
   d. Type a name for the *Widget sounds* widget. For example, My widget sounds.
   e. Click the Save button.
3. Edit the widget sounds.
   a. Click the Edit button to display the *Editing a widget* screen.
   b. In the right-hand attributes pane, select 'Use a fixed value' for each attribute you want to edit.
   c. Drag sound files from the media library into the **Open sound** and **Close sound**. These sounds play when a widget opens and closes (for example, when it is included in a finger menu). If you add multiple sounds, one sound is played at random each time.
   d. Edit the **Background sound** and **Shuffle background** sounds attributes as required. These attributes loop sounds and are described in section 16.4.

e. Enable the **Positional audio** attribute if your video wall has been set up for positional audio ie, sounds are output to the closest physical speaker(s).
This feature is described in the *MT Showcase Installation Manual*; registered users can download this manual from:
https://cornerstone.multitouch.fi/mt-showcase-manuals

f. Set the **Sound volume**. You typically enter a value from 0 to 1. This value is a multiplier for the native volume (or loudness) of the audio files, where 0 is silence and 1 is normal volume for the file. Or you can enter a value higher than 1 to play sounds louder than their native volume. For example, 1.5 means sounds are 50% louder than normal.

Note: *The actual volume heard by users also depends on volume settings for your application computer and speakers.*

4. Add a new *Interaction sounds* widget.
   a. In the Web browser attributes pane, go to the **Interaction sounds** attribute.
   b. Click the New button.
   c. In the *Add a new widget* pop-up, click the Add button.



Add a new widget pop-up for 'Interaction sounds'

   d. Type a name for the *Interaction sounds* widget. For example, My interaction sounds.
   e. Click the Save button.

5. Edit the interaction sounds.
   a. Click the Edit button to display the *Editing a widget* screen.
   b. In *Editing a widget* screen, go to the right-hand attributes pane.
   c. For each attribute you want to edit, select 'Use a fixed value'.
   d. Drag sound files from the media library into the **Touch sound** and **Pen sound** attributes. These sounds play when a user touches a widget with their hand or infrared pen. If you add multiple sounds, one sound is played at random each time.
   e. Edit the **Positional audio** and **Sound volume** attributes as required; see step 3 for details.
   f. Click Save and go back to the *Editing a Structure* screen.

6. Click the Save button to save the changes to your app's structure.

16.4 Loop sounds continuously

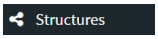
The previous sections described how to add sounds that play once (for example, when a widget opens or when a user touches a widget). But you can also add *background sounds* that loop continuously.

- **Continuous sound while a widget is open:** You can add a background sound to an individual widget. This sound loops while the widget is open.
- **Continuous sound while an app is running:** You can add a background sound to the Background layer of your app. This sound loops while the app is running.

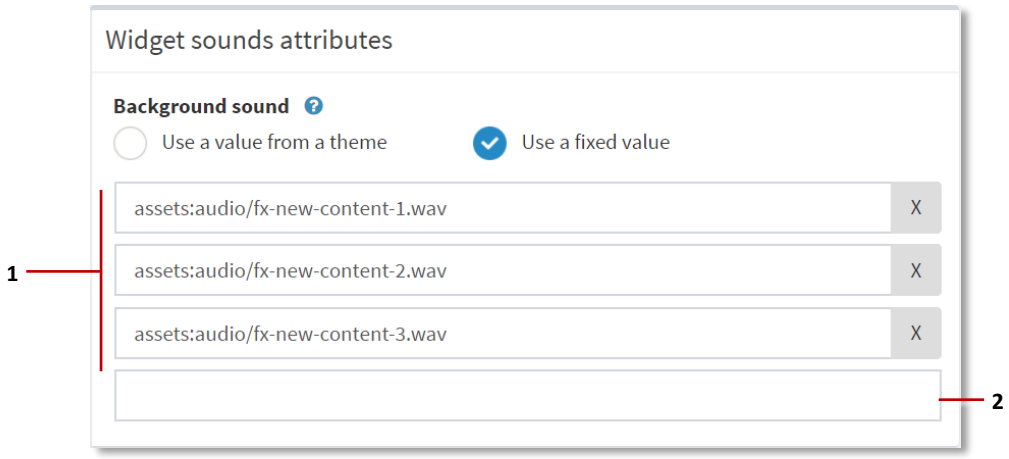
In both cases, you can add multiple background sounds. By default, they all get played continuously in a *random sequence*, but if required you can disable the shuffle so they play continuously in a *fixed sequence* (ie, in the order they are listed).

16.4.1 Loop sounds while a widget is open

This section describes how to loop sounds while a widget is open. For example, to loop sounds while the web browser in My First Structure is open, follow these steps:

1. Edit the widget attributes:
   a. Click Structures 

in the left-hand menu.
   b. Click the My First Structure hyperlink.
   c. In the Main section, click the Web browser widget.
2. In the Web browser attributes pane, go to the **Widget sounds** attribute and select My widget sounds from the drop-down list. (You created this widget in section 16.3.)
3. Click the Edit button to display the *Editing a widget* screen.
4. In the right-hand pane, edit the **Background sound** attribute. These sounds play while the widget (in this example, a web browser) is open. If you add multiple sounds, by default they all get played in a random sequence.



*Widget Sounds, Background sound attribute with multiple sounds (**1**). Each time you add an extra sound, the Editor automatically creates a new empty input field (**2**).*

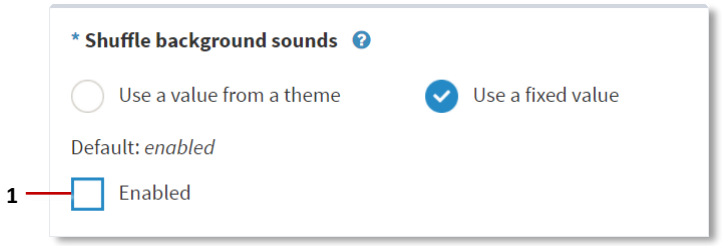
5. Click the *Show advanced attributes* hyperlink:

6. *(Optional)* Disable the **Shuffle background sounds** attribute.

   If you added multiple background sounds in step 4, the shuffle is enable by default and background sounds play in a random order. But you can disable the shuffle to play background sounds in a fixed sequence ie, the order in which they are listed.

   a. Select 'Use a fixed value'

   b. Select the Enabled check box.



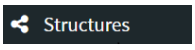
*Widget Sounds, Shuffle background sounds attribute. Clear the Enabled check box (**1**) to turn off the shuffle feature and play background sounds in a fixed sequence.*

7. Click Save and go back to the *Editing a Structure* screen.

8. Click the Save button to save the changes to your app's structure.

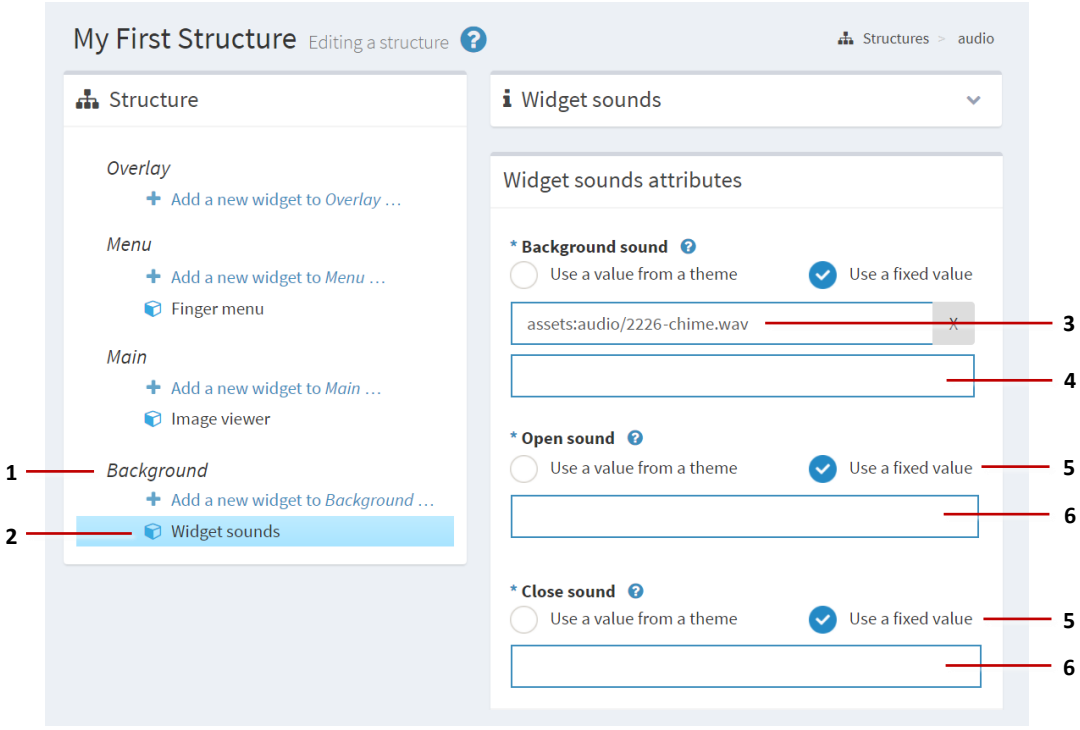
16.4.2 Loop sounds while an app is running

This section describes how to loop sounds continuously while an app is running. Follow these steps:

1. Click Structures 

in the left-hand pane.
2. Click the structure you want to edit. For example, My First Structure.

   The *Editing a Structure* screen opens.
3. Go to the Background section and click *Add a new widget to Background*.
4. From the pop-up widget menu, add *Widget sounds*.
5. In the right-hand *Widget sounds attributes* pane:
   - Edit the **Background sound** and **Shuffle background sounds** attributes. These attributes were described in section 16.4.1.
   - Set the **Open sound** and **Close sound** attributes to 'Use a fixed value' but leave the input fields empty. This effectively sets these attributes to *null* and ensures they cannot inherit any sounds defined in your theme.
6. Click the Save button to save the changes to your app's structure.



Editing a structure screen. **1** *Background layer.* **2** *Widget Sounds widget. Background sound, with audio file added* (**3**)*. A new, empty input field is added automatically* (**4**)*. Open sound and Close sound attributes are null ie, set to 'Use a fixed value'* (**5**) *but no audio files added* (**6**)*.*

16.5 Set default sounds

There are two main ways to define default sounds in your app. You can set custom sounds for different *types of widget* (for example, menu bubbles or video viewers). And you can set *global* sounds, applicable to any widget that has no custom sounds defined. Both approaches require you to edit a theme. See the following sections for details.

Note: *In all cases, you can override any default sounds by directly editing a widget's* ***Interaction Sounds*** *or* ***Widget Sounds*** *in your app's structure.*

16.5.1 Set custom sounds for different types of widget

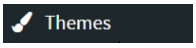
You can assign different sounds to each type of widget. For example, you can set a custom Touch sound for all menu bubbles by editing the **Interaction sounds** attribute for the *Menu bubbles* component in your theme. Likewise, you can set a custom Open sound for all videos by editing the **Widget sounds** attribute for the *Video Viewer* component.

In each case, when you edit the Widget sounds or Interaction sounds *attribute*, you must choose a *shared widget* from your widget library (or create a new one).

Note: *Custom sounds for widget types only get played if the following condition is true:*

The sound attributes for a widget in the app's structure are set to 'Use a value from a theme'. In the case of menu bubbles, the menu in the app's structure must use a menu item type *whose sound attributes are set to 'Use a value from a theme'.*

The following instructions show how to add custom sounds to menu bubbles.
Follow these steps:

1. Click 

in the left-hand pane.
2. In the Themes screen, click the theme you want.
3. In the *Editing a theme* screen, go to the **Menu bubbles** component.

   Tip: *Click the hyperlink in the Components pane.*
4. Set the following attributes:
   - **Widget sounds:** Select an existing shared widget from the drop-down menu or create a new one. These shared widgets play sounds when a widget opens or closes. They can also loop sounds while a widget is open.
   - **Interaction sounds:** Select an existing shared widget from the drop-down menu or create a new one. These shared widgets play sounds when a user touches a widget with their hand or infrared pen.

   Tip: *You created the My widget sounds and My interaction sounds shared widgets in section 16.3.*
5. Click Save to save the changes to your theme.

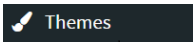
16.5.2 Set global sounds for all widgets

You can set *global* sounds, applicable to any widget that has no custom sounds defined. To do this, you must edit the **Widget sounds** and **Interaction sounds** attributes for the *Widget Behavior* theme default. In both cases, when you edit the *attribute*, you must choose a *shared widget* from your widget library (or create a new one).

Note: *Global sounds only get played if the all following conditions are true:*

- *The sound attribute for a widget (such as a Video Viewer) in the app's structure is set to 'Use a value from a theme'* and
- *The sound attribute for the corresponding* component *in your app's theme (such as the Video Viewer component) is set to 'Value not set'.*

Follow these steps:

1. Click 

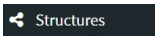
in the left-hand pane.
2. In the Themes screen, click the theme you want.
3. In the *Editing a theme* screen, go to the **Widget behavior** theme default.
4. Set the following attributes:
   - **Widget sounds:** Select an existing shared widget from the drop-down menu or create a new one. These shared widgets play sounds when a widget opens or closes. They can also loop sounds while a widget is open.
   - **Interaction sounds:** Select an existing shared widget from the drop-down menu or create a new one. These shared widgets play sounds when a user touches a widget with their hand or infrared pen.

   Tip: *You created the My widget sounds and My interaction sounds shared widgets in section 16.3.*
5. Click Save to save the changes to your theme.

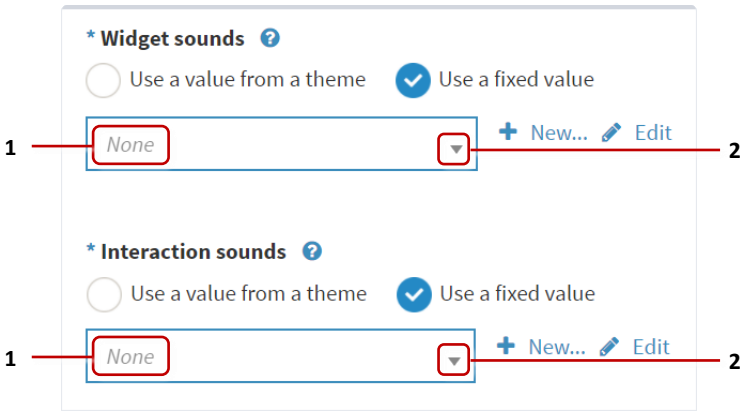
16.6 Turn off sounds for an individual widget

You can turn off sounds for an individual widget by editing its sound attributes in your app's structure.

Follow these steps

1. Click Structures 

in the left-hand menu.
2. Click the structure you want to open the *Editing a structure* screen.
3. In the left-hand pane, click the widget that you want to silence.
4. In the right-hand attributes pane:
   a. Go to the **Interaction sounds** attribute and select 'None' from the drop-down list.
   b. Go to the **Widget sounds** attribute and select 'None' from the drop-down list.
5. Click the Save button.

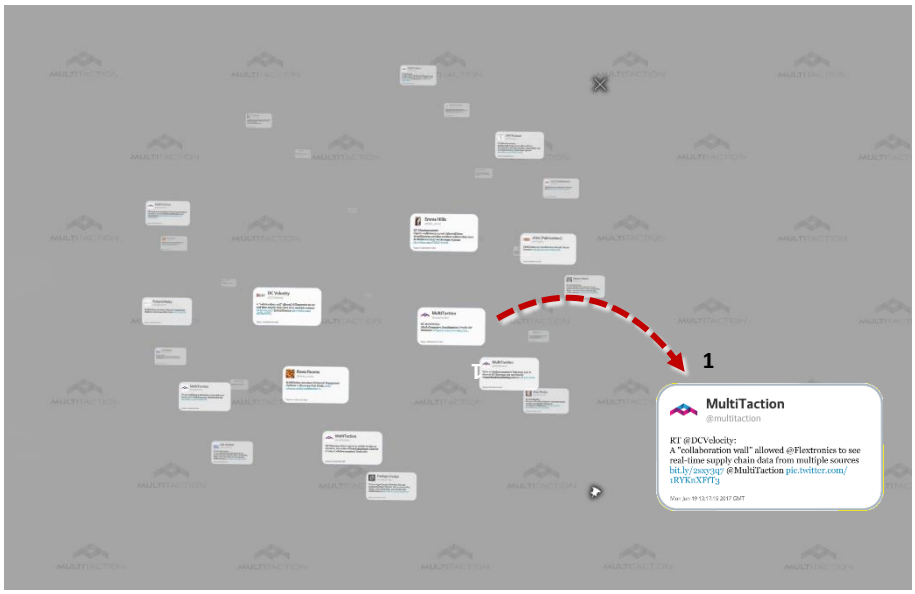


1 *Widget sounds and Interaction sounds attributes. set to None.* **2** *Drop-down menu selector.*

17 Add a Twitter feed

You can add a Twitter feed to your app based on search terms defined by you. For example, you can add a Twitter feed that displays tweets relating to MultiTaction iWall or @multitaction. To enable Twitter feeds, you must also add the *Twitter Connection service* to your MT Showcase app.

Note: *MT Showcase can only retrieve tweets; it cannot post tweets on behalf of the Twitter account being used and it cannot access the Twitter account's personal data.*



Example Twitter cloud. **1** *Tap an individual tweet to detach and enlarge it.*

17.1 Setup overview

The full setup involves these steps:

1. **Create a Twitter app:** The Twitter app enables the Twitter Connection service to access Twitter and generate Twitter feeds for your MT Showcase apps. See section 17.2
2. **Set up the Twitter Connection service:** You need this service to enable Twitter feeds in your MT Showcase apps.

   When setting up the service, you will need to add the consumer key and consumer secret from your Twitter app. These are needed to authenticate communication between the Twitter Connection service and Twitter itself when searching for tweets. You will also need to authorize your Twitter app to use a designated 'runtime' Twitter account when searching for relevant tweets. These tasks are in section 17.3.
3. **Add a Twitter feed to your MT Showcase app.** When adding a Twitter feed, you will need to add a Twitter Feed widget to a content set. At this stage, you can specify search terms to filter the tweets. You must then assign the content set to a finger menu or cloud widget. See section 17.4.

17.2 Create a Twitter app

First, you must create a *Twitter app* on Twitter's Application Management site. The Twitter Connection service will use this app to access Twitter and generate the Twitter feed displayed in your *MT Showcase app*

Full instructions are in the *MT Showcase Installation Manual*. Registered user can download this manual from https://cornerstone.multitouch.fi/showcase.

17.3 Set up the Twitter connection service

The Twitter Connection service allows you to add Twitter feeds to your MT Showcase apps, with tweets displayed either in a cloud widget or finger menu. The screenshot in section 17.4 shows a Twitter feed displayed in a cloud widget.

You set up the Twitter Connection service in the Editor's *Editing a service set* screen. First you add the service to a service set. Then you supply the consumer key and consumer secret for your Twitter app. You must also authorize the Twitter app to use a designated runtime Twitter account.

Finally, remember to add the *service set* that contains the *Twitter Connection service* to your MT Showcase app.

Note: *You only need one Twitter Connection service to supply multiple Twitter feeds in your MT Showcase apps.*

17.3.1 Retrieve the consumer key and consumer secret

The consumer key and consumer secret were generated automatically when your Twitter app was created. The Twitter Connection service needs the consumer key and consumer secret to authenticate communication with Twitter when searching for tweets. To retrieve these details from your Twitter app, follow these steps:

1. Go to Twitter's Application Management site:
   https://apps.twitter.com
2. Log in to Twitter with the Twitter account you used to create the Twitter app.
   If you are already logged in to Twitter, the *Twitter Apps* page displays immediately.
3. In the *Twitter Apps* page, click the name of your Twitter app.
4. When the Details page for your Twitter app opens, Click the tab for the Keys and Access Tokens page.
5. When this page opens, make a note of the **Consumer Key** and the **Consumer Secret**.
   Tip: Copy and paste the *key and secret to a temporary location*

17.3.2 Add the Twitter Connection service to a service set

Follow these steps:

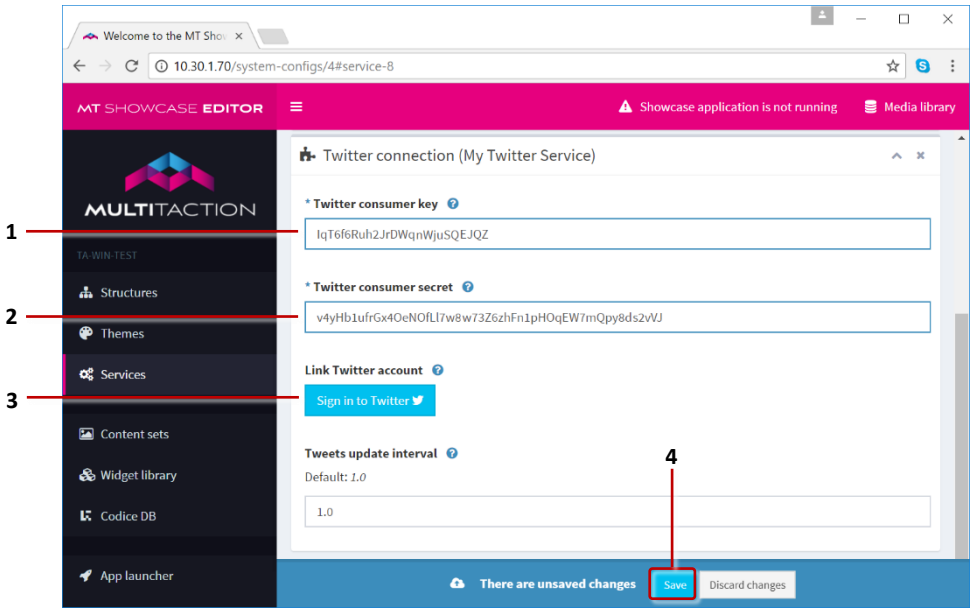
1. In the MT Showcase Editor, click Services in the left-hand pane.
2. Either click an existing service set. For example, My Service Set.
   Or create a new service set (see section 12.1.2).
3. In the *Editing a service set* screen, go to the Add new service section:
   a. Select **Twitter connection** from the Select Service drop-down menu.
   b. Enter a name for the new Twitter connection service. We recommend you choose a service name to match the name of your Twitter app.
   c. Click the Add button.
4. Click the Save button to save the Twitter connection service in your service set.

Now add the consumer key and consumer secret to the service; continue to section 17.3.3.

17.3.3 Add the consumer key and consumer secret

The Twitter Connection service needs the Twitter app's consumer key and consumer secret to authenticate communication with Twitter when searching for tweets. Follow these steps:

1. In the MT Showcase Editor, click Services in the left-hand pane.
2. Select the service set that contains the Twitter Connection service. For example, My Service Set.
3. In the *Editing service set* screen, go to the *Twitter Connection* section.



Editing a service set screen, Twitter Connection section. 1 *Twitter consumer key.* 2 *Twitter consumer secret.* 3 *Sign in to Twitter button.* 4 *Save button.*

4. In the Twitter connection section:
   a. Enter the **Twitter consumer key** and **Twitter consumer secret** for the Twitter app that you will use. Enter these values *exactly* as they appeared in the Twitter Application Management website; see section 17.3.1.

      Important! *We recommend that you copy and paste the key and secret directly from the Twitter Application Management website to avoid typing errors. Be careful not to copy and paste any leading or trailing white spaces.*

   b. *(Optional)* Edit the **Interval between new tweets** setting.

      The *Twitter connection* service regularly searches for new tweets and, if it finds any, loads them into the Twitter feed one at a time. By default, it adds one new tweet (and removes an old tweet) every second, but you can optionally set a longer interval.

5. Click the Save button.

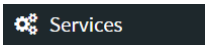
   Important! *Do not skip this save! You must save the consumer key and consumer secret before you can authorize the Twitter connection service to log into Twitter in section 17.3.4.*

Now authorize the app to use a Twitter account; continue to section 17.3.4.

17.3.4 Authorize the app to use a Twitter account

The Twitter Connection service uses your Twitter app (see section 17.2) to access Twitter and retrieve relevant tweets for a Twitter feed. However, this Twitter app will need a valid Twitter account to connect to Twitter. You must now authorize the Twitter app to use a designated runtime Twitter account.

Follow these steps:

1. In the MT Showcase Editor, click 

in the left-hand pane.
2. Select the service set that contains the Twitter Connection service. For example, My Service Set.
3. In the Editing service set screen, go to the Twitter Connection section.
4. Go to the Authorize access to Twitter setting and click the Sign in to Twitter button.
5. The next step depends on whether i) you are already logged into Twitter and ii) the Twitter app is already authorized to use a Twitter account.
   - If you already logged into Twitter *and* your Twitter app is already authorized to use a Twitter account, no further input is required. When you click the Sign in to Twitter button, Twitter automatically redirects you to the MT Showcase Editor. Go directly to step 8.
   - If you are already logged into Twitter but your Twitter app has not been authorized to use a Twitter account, go to step 6.
   - If you are not currently logged into Twitter, go to step 7.

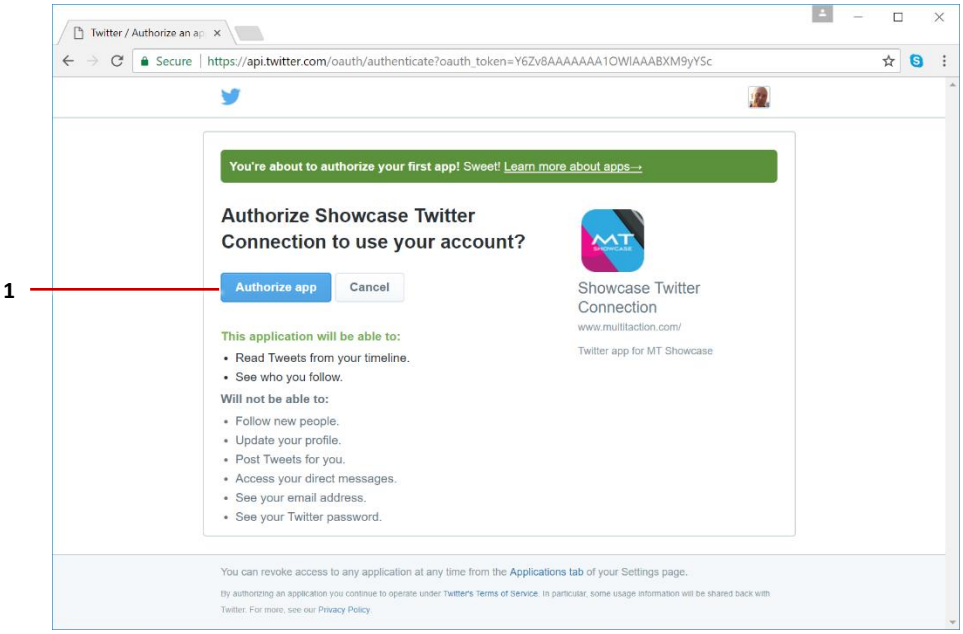
6. *(Applies only if you are already logged into Twitter but your Twitter app has not been authorized to use a Twitter account.)*

   The **Authorize [App] to use your account?** screen now displays. This screen prompts you to authorize the app to use your current Twitter account.

   For example, if you are currently logged into Twitter as @MultiTactionTweetFinder, your Twitter app will use this account when searching for tweets.

   Click the Authorize app button. You are then redirected back to the MT Showcase Editor.

   Now go to step 8.



Example 'Authorize [App] to use your account?' Screen. 1 *Authorize app button.*

7. *(Applies only if you are not already logged into Twitter)*

   The **Authorize [App] to use your account?** screen now displays. This screen prompts you to sign in to Twitter.

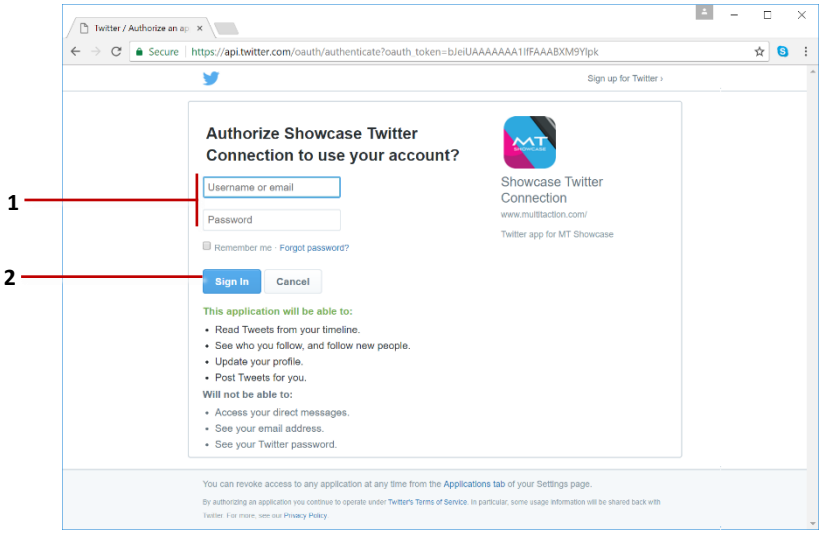
   The Twitter app will use the account you enter here. For example, if you sign into Twitter as @MultiTactionTweetFinder, the Twitter app will be automatically authorized to use this account when searching for tweets.

   Follow these steps:

   c. Enter the Twitter account name and password.

   d. Click the Sign In button.

   You are now redirected back to the MT Showcase Editor.



Example 'Authorize [App] to use your account?' Screen. **1** *Twitter account name and password.* **2** *Sign In button.*

8. Save the service set with your new Twitter Connection service.

17.3.5 Add the Twitter Connection service to your app

After setting up the *Twitter Connection service,* you must add the *service set* that contains this service to your app. This task is described in section 12.6.

17.4 Add a Twitter Feed to your app

Adding a Twitter feed to your app is a three-step process:

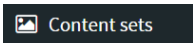
1. Add the Twitter Connection service to your app; see section 17.3.5.
2. Add the Twitter Feed widget to a content set; see section 17.4.1.
   At this stage, you can set a visible name for the Twitter feed and specify search terms to filter the tweets.
3. Assign the content set to a finger menu or cloud widget in your app's structure. See section 17.4.2 or section 17.4.3.

Notes

- *You can filter the Twitter to remove unwanted or inappropriate tweets; see section 17.5.*
- *MT Showcase can only retrieve tweets; it cannot post tweets on behalf of the Twitter account being used and it cannot access the Twitter account's personal data.*

17.4.1 Create a Twitter feed content set

Follow these steps:

1. Create a content set
   a. Click Content sets 

   in the left-hand pane.
   b. Enter the name of the new content set. For example, Twitter Feed Content Set.
   c. Click the Create New button
2. In the *Editing a content set* screen:
   a. Click the root folder.
   b. Click the Add button.
   c. From the pop-up *Add new content* menu, add a Twitter Feed to the content set.
3. In the right-hand *Content Attributes* pane:
   a. Set the **Maximum tweet count**. Specify the maximum number of tweets that can be shown in a Twitter cloud or in a finger menu. Note that tweets are refreshed continuously. (The Twitter feed refresh frequency is defined in the Twitter Connection service.)
   b. Set the **Visible name** for your Twitter feed. This name is used as the label for the central bubble if you launch the Twitter feed from a finger menu.
   Choose a name that reflects the type of tweets you want to display. For example, MultiTaction iWall.

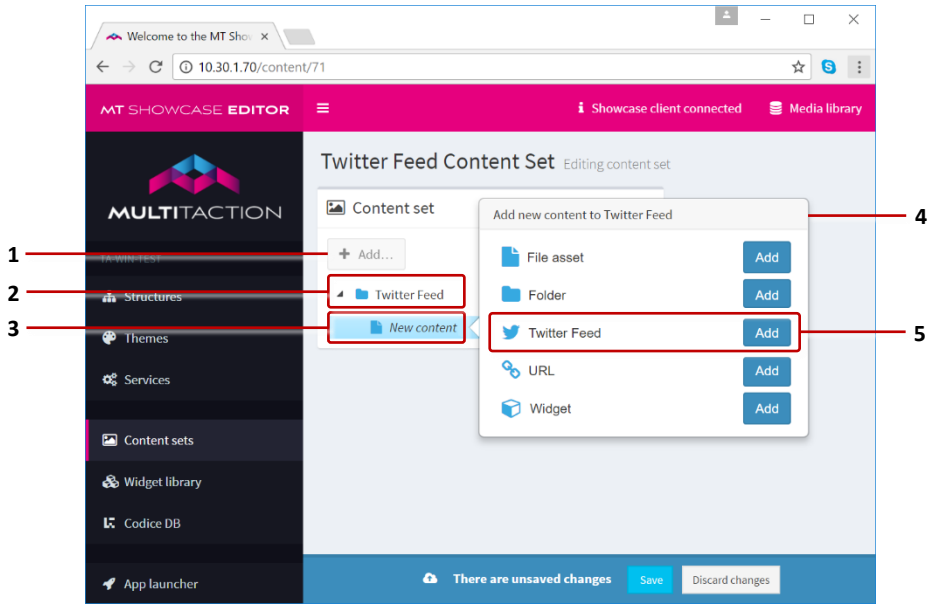
c. Define the **Search terms** for your Twitter feed. For example:

- multitation finds tweets with MultiTaction, multitaction or MULTITACTION.
- multitaction iWall finds tweets with MultiTaction AND iWall.
- multitaction OR iWall finds tweets with MultiTaction OR iWall (or both).
- multitaction -iWall finds tweets with MultiTaction but not iWall.
- from:MultiTaction finds tweets sent by the MultiTaction account.
- "MultiTaction Cells" finds tweets with the exact phrase 'MultiTaction Cells'.
- #multitaction finds tweets containing the hashtag 'multitaction'.

For more search filters, see section 17.5.2.

4. Click the Save button to save your new Twitter Feed content set.

You can now assign this content set to a cloud widget or finger menu. Go to section 17.4.2 or section 17.4.3.



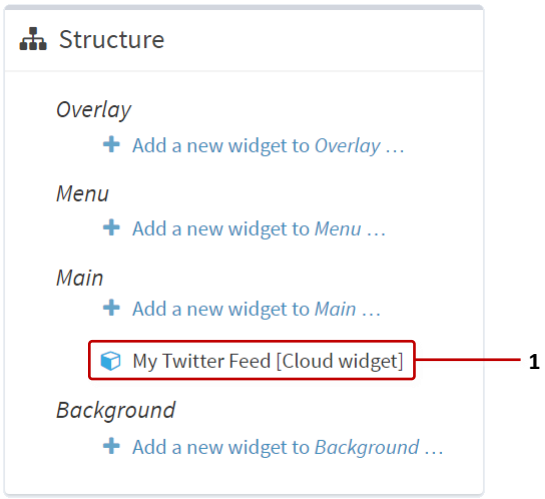
Editing a content set screen. Adding a Twitter Feed to a content set.
1 *Add button.* **2** *Root folder.* **3** *New content set item. The Twitter feed is added here.*
4 *Pop-up 'Add new content' menu.* **5** *Add a Twitter Feed to the content set.*

17.4.2 Set up a Twitter cloud

You can add a Twitter cloud to your app. The Twitter cloud comprises a ball of individual tweets that match the specified search terms. To create a Twitter cloud, you simply add a Cloud widget and assign the Twitter content set you created previously.

Follow these steps:

1. Add a cloud widget to your app's structure as described in section 10.4.
2. Go to the Main section of the *Editing a structure* screen and click the Cloud widget.
3. Rename the Cloud widget. For example, My Twitter Feed.



Edit a structure screen. **1** *Twitter cloud. (Cloud widget using a Twitter Feed content set.)*

4. In the *Cloud widget attributes* pane, go to the **Content set** attribute.
5. Select the content set you created previously, Twitter Feed Content Set, from the drop-down menu.

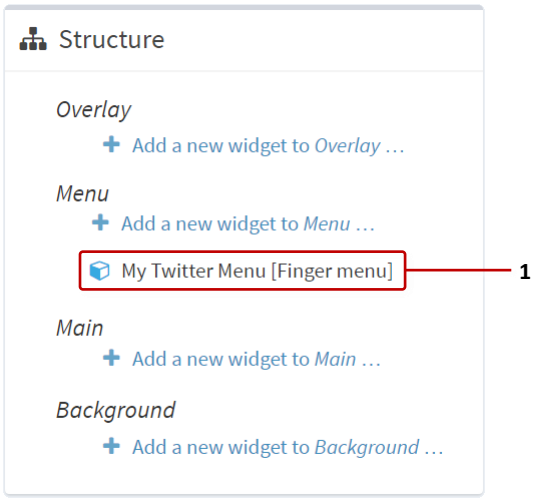
Cloud widget attributes: Content set attribute.

6. Click the Save button to add the Twitter cloud to your app.

17.4.3 Set up a Twitter finger menu

You can add a Twitter finger menu to your app. Menu items comprise individual tweets that match the specified search terms. To create a Twitter finger menu, you simply add a finger menu and assign the Twitter content set you created previously.

1. Add a finger menu to your app's structure as described in section 5.2.
2. Go to the Menu section of the *Editing a structure* screen and click the Finger menu.
3. Rename the Finger menu. For example, My Twitter Menu.



Edit a structure screen. **1** *Twitter menu. (Finger menu using a Twitter Feed content set.)*

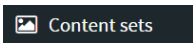
4. In the Finger menu attributes pane, go to the Content set attribute.
5. Select the content set you created earlier, Twitter Feed Content Set, from the drop-down menu.
6. Click the Save button to add the Twitter menu to your app.

17.5 Filter the Twitter feed

There may be occasions when your Twitter feed retrieves tweets that are inappropriate. To remedy this, you can filter the tweet search and or remove unwanted tweets by tapping them with a tweet blocker Codice card.

17.5.1 Add a search filter to the Twitter feed content set

You can filter the tweet search by editing the search terms in the Twitter feed content set. Follow these steps:

1. Click Content sets 

in the left-hand pane.
2. Select your Twitter feed content set. For example, Twitter Feed Content Set.
3. In the *Editing a content set* screen, edit the **Search Terms** attribute in the right-hand pane.
4. Edit the search terms for your Twitter feed using any of the supported search filters; see section 17.5.2.

17.5.2 Search filters

The MT Showcase Editor supports the standard Twitter search filters. For example, use the following filters when editing the **Search Terms** attribute to filter the tweets retrieved in the Twitter feed.

Tweets from a specific account

Use from:<account> to find tweets sent by a specific Twitter account. For example:
from:MultiTaction finds tweets sent by the MultiTaction account.

Tweets replying to a specific account

Use to:<account> to find replies to tweets from a specific Twitter account. For example:
to:MultiTaction finds tweets replying to tweets from MultiTaction.

AND

A space between search terms is an AND operator. For example:
multitaction iWall finds tweets with MultiTaction AND iWall.

OR

Type 'OR' between search terms for an OR operator. For example:
multitaction OR iWall finds tweets with MultiTaction OR iWall (or both).

NOT

Prefix a search term with a minus sign for a NOT operator. For example:

multitaction -iWall finds tweets with MultiTaction but not iWall.

iWall –from:MultiTaction finds tweets with iWall but excludes 'iWall' tweets sent by the MultiTaction account.

List continues on next page.

Retweets

Use -filter:retweets to exclude retweets.

Safe searches

Add filter:safe to only retrieve safe tweets.

Hashtags

To retrieve tweets with a specific hashtag, simply add the hashtag as a search term: #multitaction finds tweets containing the hashtag “multitaction”.

Upper case or lower case?

Search terms are not case-sensitive and don't need quotes. For example: multitaction finds tweets with MultiTaction, multitaction or MULTITACTION. MULTITACTION finds the same tweets as multitaction.

Exact phrases

Use double quotes to find tweets containing phrases. For example: "MultiTaction Cells" finds tweets containing the exact phrase ‘MultiTaction Cells’.

Can I combine filters?

Yes. You can combine multiple filters to create a complex search term. For example: "touch displays" OR iWall –from:MultiTaction -filter:retweets finds tweets with ‘touch displays’ or ‘iWall’, but excludes such tweets sent by the MultiTaction account and also excludes retweets.

For more about tweet filtering, see the Twitter Developer web site: https://developer.twitter.com/en/docs/tweets/rules-and-filtering

17.5.3 Remove tweets with a Codice card

You can designate any Codice as a ‘tweet blocker’. This enables users to remove such tweets from the app by simply tapping them with a tweet blocker Codice card.

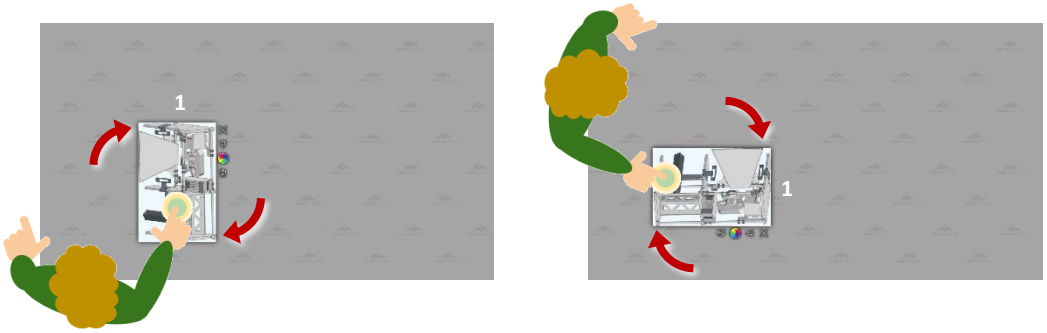
Note that if you restart MT Showcase, a banned tweet can potentially reappear in the Twitter feed if it has not been supplanted by more recent feeds.

For instructions on designating a Codice as a tweet’ blocker, see section 13.5.

18 Table mode

You can configure MT Showcase to run on touchscreen tables by enabling *table mode* in your app. In this mode, MT Showcase automatically detects which side of the table the user is on when they tap a widget, open a menu, or place a Codice card on the screen. It then rotates screen content to face the user.

All the content widgets, including the various image, PDF and video widgets, plus the finger menu and teaser menu support table mode.



MT Showcase in table mode. For example, when a user taps an Image Viewer widget (**1**)*, the widget rotates to face the user.*

18.1 Enable table mode

Follow these steps

1. Edit a structure.
   a. Click Structures  in the left-hand menu.
   b. Click the structure you want to edit. For example, My First Structure.
      The *Editing a Structure* screen appears.
2. In the Editing a Structure screen, click the widget you want to edit.
3. In the right-hand *attributes* pane, go to the **Rotate widget towards hand** attribute.
   Or, for finger menus and teaser menus, go to the **Rotate menu towards hand** attribute.
   c. Select 'Use a fixed value'
   d. Select the Enabled check box.
   e. Click Save.

   Note: *Alternatively, edit the* **Rotate widget towards hand** *attribute in the Table Mode theme defaults; see section 6.6.*

18.2 Enable Codice support for table mode

(Optional) You can also configure Codice handling to support table mode. For example, you can rotate Codice content to face the user. You can also close Codice content when the user lifts their Codice marker from the screen.

Note: *Codice content refers to widgets launched when a user presents a Codice marker on the screen. See section 13.4.*

1. Click Themes in the left-hand pane.

1. In the *Themes* screen, click the theme you want,

   The *Editing a theme* screen opens automatically.

2. In the *Editing a theme* screen, go to Theme defaults pane (see section 6.6.1) and click the **Table mode** defaults set.

   In the main pane, the Table mode defaults set expands automatically to display the default attributes.

3. Edit the following attributes:

   - **Move contents with the Codice:** When this attribute is enabled, widgets launched by a Codice card automatically rotate to face the user.

     Note: *In technical terms, the Codice content rotates to match the orientation of the Codice marker.*

   - **Dismiss contents when Codice is lifted:** When this attribute is enabled, widgets launched by a Codice card are closed when the user lifts the Codice from the screen.

   - **Codice rotation offset:** This attribute is provided to correct the orientation of Codice markers printed upside down on business cards or attached the wrong way around to tangible user interfaces (section 12). *Generally, you only edit this attribute if instructed to do so by MultiTaction support staff.*

   For each attribute, click the help button for more details.

4. In the *Editing a theme* screen, go to the **Codice detector** component and edit the following attributes:

   - **Rotate personal space towards Codice:** When this attribute is enabled, the user's personal space automatically rotates to face the user.

   - **Codice rotation offset:** See the description in step 3.

5. Click Save.

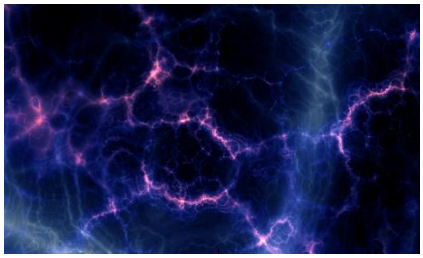
19 Backgrounds and animation effects

You can enhance your MT Showcase apps by including animated backgrounds and animation effects.

The following animated backgrounds are available:

- **Awesome:** Displays a background of drifting clouds, similar in effect to the Aurora Borealis. By default, the clouds are purple and cyan but you can fully customize the colors and animation speed.
- **Shader Effect:** The default shader resembles the night sky, with dramatic purple and magenta nebulae drifting across the screen, punctuated with cores of bright white light. However, you can use a custom shader file if required.

Awesome background (default colors)

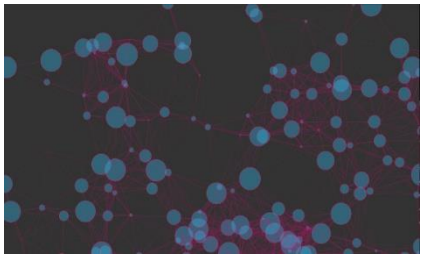


Shader Effect background

For extra visual impact, you can also add animation effects. These effects are interactive and respond to touches, emanating outwards from the user's fingers:

- **Champagne:** Displays streams of rising bubbles, like in a glass of sparkling wine.
- **Particles:** Displays small floating circles, randomly connecting to nearby circles.
- **Ripple:** Displays wavelets, like ripples on a pond.
- **Sparkles:** Displays swirls of particles. Its effect is like dragging your hand through a sparkling fluid.

Champagne animation effect



Particles animation effect

Ripple animation effect

Sparkles animation effect

19.1 Add an animated background

Follow these steps:

To add a special effect:

1. Click Structures 

in the left-hand pane.
2. Click the structure you want to edit. For example, My First Structure.

   The *Editing a Structure* screen opens.
3. Go to the Background section and click *Add a new widget to Background*.
4. From the pop-up widget menu, choose a background.
5. In the right-hand attributes pane, edit the background attributes.

   For example, you can edit the animation speed. For the awesome background, you can also customize the colors; see section 19.3.
6. Click the Save button to add the background to your app's structure.

19.2 Add animation effects

These effects are purely cosmetic and display above or below items in the app's main layer. You can add them to overlay layer or background layer. When added to the overlay layer, the effects are superimposed above the main layer. When are added to the background layer, they appear to be underneath the main layer.

Note: *The Sparkles effect is only available in the background layer.*

Follow these steps:

1. Click Structures 

in the left-hand pane.
2. Click the structure you want to edit. For example, My First Structure.

   The *Editing a Structure* screen opens.
3. Go to the Background section and click *Add a new widget to Background*.

   Or go to the Overlay section and click *Add a new widget to Overlay*.
4. From the pop-up widget menu, choose an animation effect: Champagne, Particles, ripple, or Sparkles.
5. In the right-hand attributes pane, edit the animation effect attributes.

   For example, you can edit the colors and size of the Particles animation and the level of bubbles generated in the Champagne effect.
6. Click the Save button to add the effect to the app's structure.

19.3 Customize colors for Awesome background

By default, the Awesome background of drifting clouds uses a purple and cyan color scheme but you can fully customize the colors. For example, you can specify a background that complements your corporate branding.

Awesome background, color variations

Note: *This section describes how to customize the Awesome colors by editing an app directly. But you can also edit the Awesome colors in a theme. Open the theme you want, go to the* Awesome *component, and repeat the instructions below.*

To edit the Awesome colors directly, follow these steps:

1. Click Structures 

in the left-hand pane.
2. Click the structure that contains the Awesome background you want to edit. For example, My First Structure.

   The *Editing a Structure* screen opens.
3. Go to the Background section and click the Awesome widget
4. In the right-hand attributes pane, go to the *Awesome attributes* section.
5. Edit the following attributes:

   Background color
   Background highlights color
   Foreground color
   Foreground highlights color

   You can quickly experiment with different background and foreground color combinations. You typically choose a lighter color for the highlights. For guidance on specifying colors, see section 21.4.

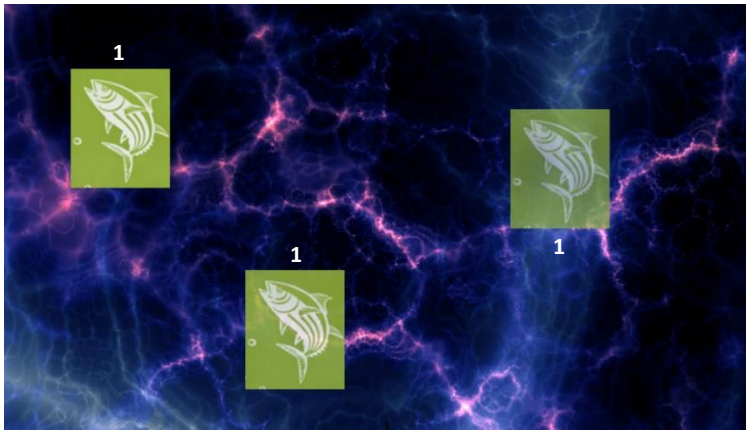
   If required, other attributes enable you to adjust the background 'lightness' and the amount of foreground color in the animation.
6. Click the Save button.

20 Screensaver

You can define one or more screensavers in your MT Showcase app.

A single screensaver can comprise a background such as Awesome or Shader effect, an image or movie, or animated content. The screensaver is displayed when the app has been idle for a specified time. To clear the screensaver and return to the app, the user taps the screen.

- **Screensaver content:** The simplest screensaver shows an animated background such as Awesome or Shader effect (see section 2.3.4). Alternatively, a screensaver can display an Image viewer, Image movie, or Video viewer widget; you specify the image or video that is displayed. The final screensaver option is Animated content (see section 25.2).
- **Multiple screensavers:** By default, a single screensaver fills the entire screen, but you can specify a custom size and screen area. For example, you may choose to define several screensavers, each showing separate content, covering different areas of the screen.
- **Lock screen:** Optionally, you can add locking to a screensaver. User can manually activate the screensaver and lock the screen. While locked, the screensaver cannot be dismissed and prevents all input on the underlying content. For more details see section 20.2.



*Example screensaver: animated content on top of Shader effect background. Here, the animated content comprises an Image Viewer widget (**1**) that fades in and out at random screen locations. Users tap anywhere on the screen to hide the screensaver and return to the Showcase app.*

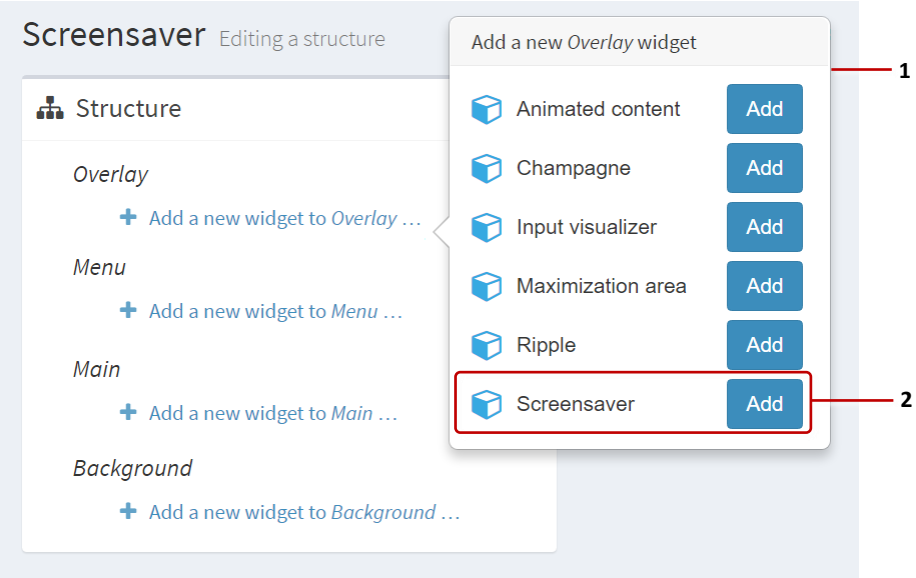
20.1.1 Add a screensaver to your app

Follow these steps:

1. Click Structures  in the left-hand pane.
2. Click the structure you want to edit. For example, My First Structure.

   The *Editing a Structure* screen opens.

3. Go to the Overlay section and click *Add a new widget to Overlay*.
4. From the pop-up widget menu, choose **Screensaver**.



Editing a structure screen. 1 *Pop-up menu of Overlay widgets.* 2 *Screensaver widget and Add button.*

5. In the *Screensaver attributes* pane, set the general screensaver attributes:
   - **Size** and **Location:** By default, a single screensaver fills the entire screen. (To manually configure a screen saver to fill the screen, set both attributes to *0 0*.) Alternatively, you can specify a custom size and location for the screensaver.
   - **Idle time:** Specify how much idle time (in seconds) must elapse before the screensaver launches. 'Idle time' means a period in which no users touch the screen or interact with the app.
   - **Clip contents:** You do not normally need to change this attribute. Click the help button for details.
6. (*Optional*) If you intend to add multiple screensavers to your app, you may want to give distinctive names to each screensaver in the Editor.

   To rename a screensaver, right-click the Screensaver widget and click Rename.
7. Click the Save button.

Now set up the content for your screensaver. Go to section 20.1.2.

20.1.2 Set up content for your screensaver

Follow these steps:

1. In the *Editing a Structure* screen, click the ▷ triangle next to the Screensaver widget that you added in section 20.1.1.

   The widget expands to reveal the *Add a new widget to Screensaver* hyperlink.

2. Click the *Add a new widget* hyperlink.

   a. Add the widget(s) you want. Choose from:

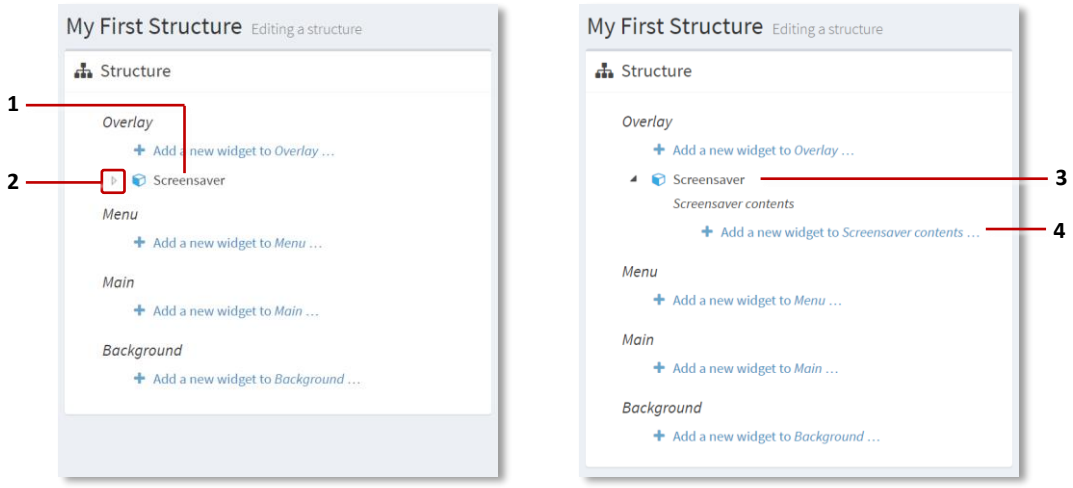
   - Image Viewer, Image Movie or Video Viewer.
   - Awesome or Shader effect animated backgrounds.
   - Animated content; see section 25.2 for setup details.

   Tip: *You can add multiple images, videos or animated content to a single screensaver.*

   b. Set the attributes for the new widget(s).

   Note: *If you want an image or video screensaver to fill the entire screen, you must manually set the Size and Location attributes to fill your video wall.*

3. Click the Save button.



Setting up a screensaver. **1** *Screensaver added to Overlay layer.* **2** *Triangle. Click to reveal 'Add a new widget' hyperlink.* **3** *Expanded Screensaver content widget.* **4** *'Add a new widget to Screensaver contents' hyperlink.*

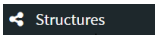
20.2 Lock screen

You can use a screensaver to lock the screen. While screen is locked, the screensaver cannot be dismissed, and it prevents all input on the underlying content.

User can manually activate the screensaver and set either a PIN or Codice card to lock the screen. The screen can be unlocked later by presenting the same PIN or Codice card.

20.2.1 Add locking behaviour to a screensaver

You can add locking to an existing screensaver or create a new one. For example, to add locking to the screensaver in My First Structure (see section 20.1.1), follow these steps:

1. Edit the screensaver attributes:
   a. Click Structures  in the left-hand menu.
   b. Click the My First Structure hyperlink.
   c. In the Main section, click the Screensaver widget.
2. Add a new Lock behaviour.
   a. In the screensaver attributes pane, got to the **Lock behaviour** attribute.
   b. Click the New button.
   c. Type a name for the new Lock behaviour. For example, My screen lock.
   d. Click the Save button.
3. Edit the Lock behaviour.
   a. Click the Edit button to display the *Editing a widget* screen.
   b. In *Editing a widget* screen, go to the right-hand attributes panes.
   c. Set the **Unlocking method**.

      This attribute determines the method for locking and unlocking the screen. You can set this to PIN or Codice card or allow user to choose the method when locking the screen. The actual PIN or Codice is set by the user when locking the screen.
   d. Click the Save button.
4. *(Optional)* Disable automatic screensaver activation. Normally screensaver activates after the app has been idle for a while. But you can set the **App idle timeout** attribute for the screensaver to 0 to prevent the screensaver from activating automatically.

Now set up a method for the user to activate the screensaver. Go to section 20.2.2.

20.2.2 Add a Screensaver trigger widget to your app

Screensaver must be activated manually to lock the screen. If the screensaver activates after an idle timeout, the screen is not locked even if the screensaver has a locking behaviour configured.

You can add a *Screensaver trigger widget* to your app to allow user to activate the screensaver. You can add this widget to a menu, hotspot or Codice content.

Note that if your app has multiple screensavers, the screensaver trigger activates *all* of them.

For example, to add a Screensaver trigger widget to a content hotspot, follow these steps.

1. Add a Content hotspot to your app's structure. Follow the instructions in section 11.1.
2. *(Optional)* Configure the hotspot location and appearance.
   a. Configure the **Location** and **Size** attributes for the hotspot.
   b. Add a visualization widget to help user locate the hotspot. Drag an image from the media library to the **Hotspot visualization widgets** list.
   c. Save the changes.
3. Add a Screensaver trigger widget to the hotspot.
   a. Click the *Add* hyperlink for the **Widgets launched from hotspot around tap location** list.
   b. Select the Screensaver trigger widget.
   c. Save the changes.

This completes the lock screen configuration. Now test locking and unlocking the screen, continue to section 20.2.3.

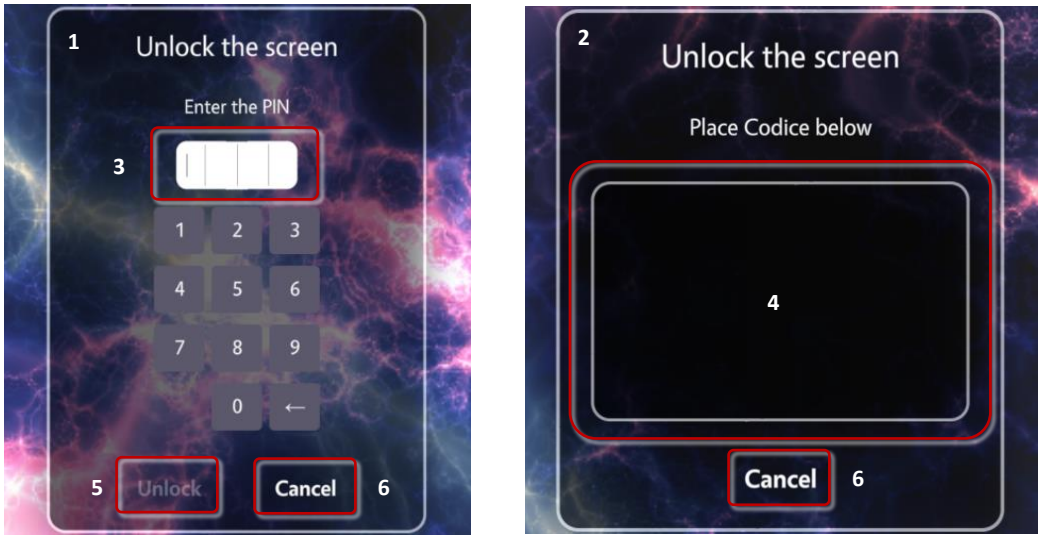
20.2.3 Test lock screen

To lock the screen, tap the hotspot with the screensaver trigger widget. The screensaver is activated, and a dialog is displayed to input a PIN or Codice to lock the screen.

- **PIN**: Use the virtual numpad to type in a 4-digit PIN. Tap Lock. Then type in the same PIN again to confirm it. Finally tap Lock again to lock the screen.
- **Codice**: Place your Codice card on the dialog. Review the displayed code to make sure it was read correctly. Then tap Lock to lock the screen.

Screen lock dialogs. 1 *Lock dialog for PIN.* 2 *Lock dialog for Codice.*
3 *Input PIN.* 4 *Place Codice.* 5 *Lock button.* 6 *Cancel button.*

To unlock the screen, tap anywhere on the screensaver area to bring up an unlocking dialog. Type in the PIN or place a Codice card on the dialog. Then tap Unlock. You can also simply place the Codice on the screen without first tapping the screen to unlock the screen.



Screen unlock dialogs. 1 *Unlock dialog for PIN.* 2 *Unlock dialog for Codice.*
3 *Input PIN.* 4 *Place Codice.* 5 *Unlock button.* 6 *Cancel button.*

21 Widget customization

This section introduces common customization you can add to most widgets in MT Showcase.

21.1 Drop shadows

You can add a drop shadow to widgets such as image viewers and web browsers to simulate 3D depth, giving the impression that the widget is raised above the background.

You can customize the size, position and color of the drop shadow. The default color is black, but white or pale gray may be more effective on darker backgrounds.

Image viewer widget with a white drop-down shadow. In this example, the shadow is offset by 20 pixels below and to the left of the widget

To add a drop shadow:

1. Click Structures 

in the left-hand pane.
2. Click the structure you want to edit. For example, My First Structure.

   The *Editing a Structure* screen opens.
3. Go to the Main section and click the widget you want to edit. The following widget types support drop shadows:
   - Book
   - Image viewer
   - Video viewer
   - Web browser
4. In the widget attributes pane, click the *Show advanced attributes* hyperlink:

5. Edit the drop shadow attributes as required.

For example, to add a white drop shadow, set the **Drop shadow color** attribute to rgba(255,255,255,1) or #FFFFFF or white. Click the help button or see section 21.4 for details of color input methods.

6. Click the Save button.

21.2 Size limits

You can set a maximum and minimum size for widgets. If a user makes a widget too big or too small, the widget automatically resizes. This is useful to prevent users inadvertently super-sizing a widget so it covers the whole screen or making a widget too small to be usable. For example, to prevent widgets being bigger than a single screen, you can set the maximum size to 1920 x 1080 pixels.

21.2.1 Maximum widget size

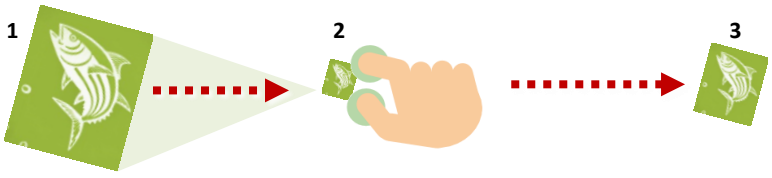
If a user tries to enlarge a widget, they can only permanently enlarge it up to its maximum size. While the user is interacting with the widget, you can allow it—within limits—to temporarily exceed its permitted maximum size. When the user stops interacting with the widget (moving or resizing it), it automatically shrinks back to its maximum size.

Note: *This effect of temporarily overriding the maximum size gives widget interactions a more natural feel. If you were to set a 'hard' maximum size, users might wrongly infer that the resizing feature was faulty.*

1 *Widget starting size.* 2 *Widget temporarily enlarged beyond its maximum size.*
3 *Widget maximum size. When user interaction stops, the widget shrinks back to this size.*

21.2.2 Minimum widget size

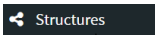
If a user tries to shrink a widget, they can only permanently shrink it down to its minimum size. While the user is interacting with the widget, you can allow it—within limits—to temporarily be smaller than its permitted minimum size. When the user stops interacting with the widget (moving or resizing it), it automatically expands back to its minimum size



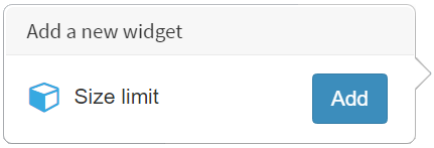
1 *Widget starting size.* 2 *Widget temporarily shrunk below its minimum size.* 3 *Widget minimum size. When user interaction stops, the widget expands back to this size.*

21.2.3 Set the maximum and minimum sizes

For example, to assign a size limit widget to the web browser in My First Structure (see step 3 in section 3.3), follow these steps:

1. Edit the widget attributes:
   a. Click Structures 

in the left-hand menu.
   b. Click the My First Structure hyperlink.
   c. In the Main section, click the Web browser widget.
2. Add a new size limit widget.
   a. In the Web browser attributes pane, go to the **Size limit** attribute.
   b. Click the New button.
   c. In the *Add a new widget* pop-up, click the Add button.



Add a new widget pop-up for 'Size limit'

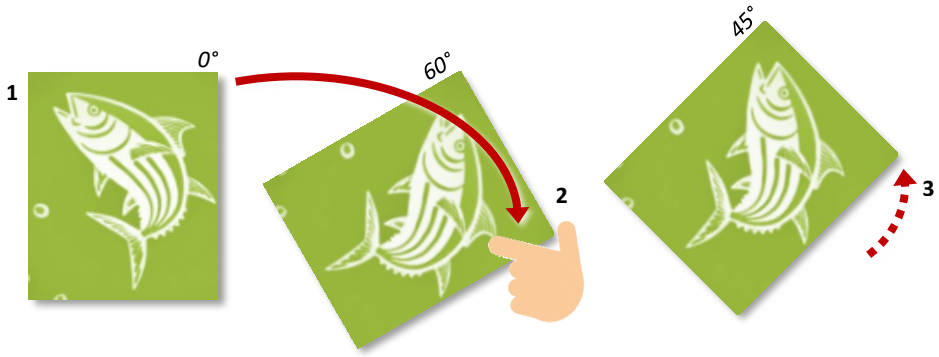
   d. Type a name for the new size limit widget. For example, My Size Limit.
   e. Click the Save button.
3. Edit the new size limit widget.
   a. Click the Edit button to display the *Editing a widget* screen.
   b. In *Editing a widget* screen, go to the right-hand attributes pane.
   c. Set the **Minimum size**. Enter the minimum width and height. For example, enter 10cm 10cm to prevent widgets having a width or height of less than 10 centimeters.
   d. Set the **Maximum size**. Enter the maximum width and height. For example, enter 1920px 1080px to set a maximum widget size that matches the size of a single MultiTaction Cell in landscape mode. If a user tries to make a widget bigger than this, it will shrink back to its permitted maximum.
   e. Set the **Maximum size multiplier**. You can allow users to make widgets bigger than their permitted maximum size while they interact with the widget. This multiplier defines how much a widget can exceed its maximum size. For example, a multiplier of 1.6 means a widget can expand 60% beyond its normal maximum size. When the user stops interacting with the widget, it shrinks back.
4. Click the Save button.

Note: *You have now created a new* Size limit *shared widget. This widget is now listed in the widget library (see section 2.2.4) and can be assigned to any other widget in your app. You can also set default* Size limit *values in a theme.*

21.3 Snap to angle

By default, you can freely rotate any screen item. But in some cases, you may want to constrain the incremental rotation so that a rotated widget automatically re-adjusts itself to a fixed angle of rotation. For example, you may want to ensure that a Video Viewer widget or a Web Browser widget always returns to zero degrees (ie, its correct orientation) if it is rotated by a user.

The **snap to angle** shared widget meets this requirement by causing a rotated widget to automatically right itself to a preferred angle of rotation. (The effect is similar to a boat righting itself after keeling over.)



Snap to angle example. Here, the number of permitted rotation divisions is 8, equating to stopping points at 45° intervals.
1 *An image starts at 0°.* **2** *A user rotates the image to about 60°.* **3** *When the user releases the image, it automatically rights itself back to the closest point ie, 45°.*

For example, to assign a snap to angle widget to the web browser in My First Structure (see step 3 in section 3.3), follow these steps:

2. Edit the widget attributes:
   f. Click Structures  in the left-hand menu.
   g. Click the My First Structure hyperlink.
   h. In the Main section, click the Web browser widget.
3. Add a new snap to angle widget.
   a. In the Web browser attributes pane, go to the **Snap to angle** attribute.
   b. Click the New button.
   c. In the *Add a new widget* pop-up, click the Add button.

Add a new widget pop-up for 'snap to angle'

   d. Type a name for the new snap to angle widget. For example, My 45° Angle Snap.
   e. Click the Save button.

4. Edit the snap to angle widget.
   a. Click the Edit button to display the *Editing a widget* screen.
   b. In *Editing a widget* screen, go to the right-hand attributes pane.
   c. Set the **Rotation divisions**.

      This attribute determines the number of possible 'permitted angles' if you rotate a widget through a full circle (ie, 360°). To calculate the interval between permitted angles, divide 360 by the number of rotation divisions.

      For example, if you specify 4 rotation divisions, the permitted angles are at 90° intervals (ie, 0°, 90°, 180°, 270°). Likewise, if you specify 2 rotation divisions, the permitted angles are 0° and 180°. And if you specify 1 rotation division, the widget will always correct itself to 0°.

      Click the help button for details.
   d. Set the **Duration of adjustment to snap angle**.

      When the user releases a rotated widget, it automatically adjusts its rotation to the nearest permitted angle This attribute determines how quickly (or slowly) that adjustment takes.
   e. Click the Save button.

Note: *You have now created a new* Snap to angle *shared widget. This widget is now listed in the widget library (see section 2.2.4) and can be assigned to any other widget in your app. You can also set default* Snap to angle *values in a theme.*

21.4 Colors

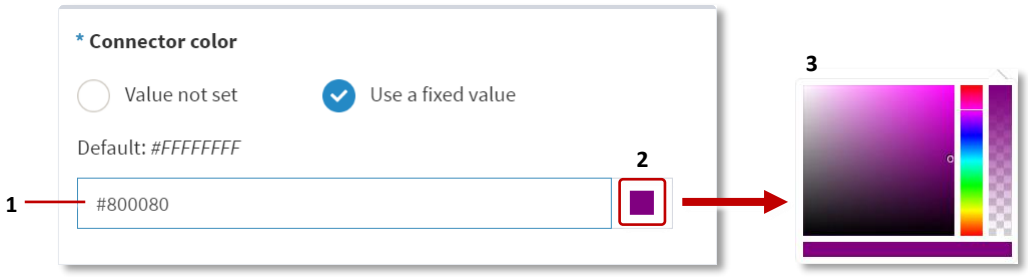
Many MT Showcase widgets have *color* attributes. That is, you can define the attribute's color. For example, you can define the color of bubbles and connectors in a finger menu, on-screen annotations, drop shadows, particles in the Sparkles animation, and items in the Input visualizer.

In all cases, you have four methods for specifying color. You can specify:

- **RGBA values:** Enter an RGBA color in the input box. For example, you can specify solid purple as rgba(128,0,128,1).
  - **Native format:** Whichever color method you use, MT Showcase always translates the specified color to its RGBA equivalent value and displays this RGBA value in the MT Showcase Editor.
  - **Transparency:** The 'a' element in the RGBA value represents the alpha channel (also called the opacity channel). Its possible values range from 0 to 1. When its value is 0, the color is fully transparent ie, invisible; when its value is 1, the color is fully opaque ie, solid.
- **Hex codes:** Enter a hex code, or hex triplet, in the input box. Remember to add a leading '#' character. For example, you can specify purple as #800080.

  You can only specify solid colors using hex codes; you cannot specify transparent or semi-transparent colors. You can, however, edit the alpha channel after MT Showcase translates the hex code to an RGBA value.
- **CSS color names:** Enter the color name eg, purple or darkmagenta. Color names are *not* case-sensitive. For a list of color names supported by modern browsers, see www.w3schools.com/colors/colors_names.asp.

  Named colors are always solid; you cannot specify transparent or semi-transparent colors. You can, however, edit the alpha channel after MT Showcase translates the color name to an RGBA value.
- **Color picker:** Click the preview swatch to open a color picker. Click the color you want, then click outside the picker to close it.



Menu bubble widget, connector color attribute. 1 *Input box. This accepts RGBA values, hex codes and color names.* 2 *Preview swatch.* 3 *Color picker.*

22 Schedules for apps and widgets

You can control when apps and widgets run by creating a schedule of events.

- The *app schedule* is a collection of *app events*. Each app event defines when a specific app runs (the start and stop times). For example, you can schedule a specific app to run every Monday morning.

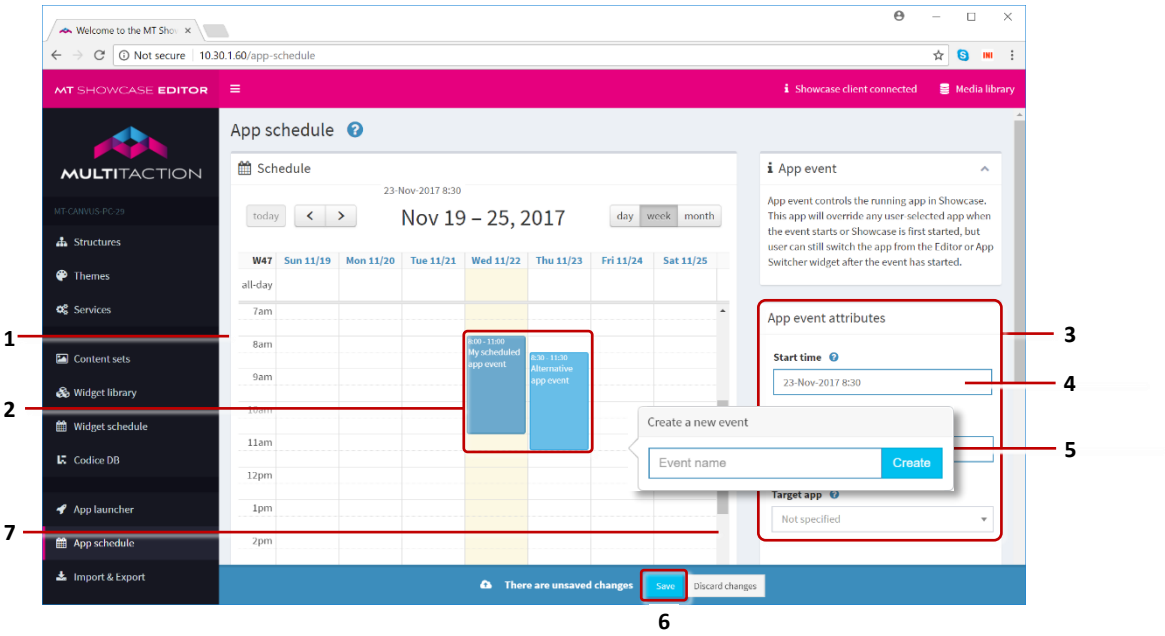
  When a scheduled app starts, it will replace any app currently on the screen. You can display a countdown to alert users when a scheduled app is due to start. You can also allow users to cancel the scheduled app.

  Note that while a scheduled app is running, users can still use App Switcher widgets (section 25.3) or the Editor's App launcher (section 2.1) to switch to a different app.

- The *widget schedule* is a collection of *widget events*. Each event defines a period, or a recurring period, when a widget is active. You can assign events to a widget when you edit your app's structure. For example, you can assign an event to a video viewer widget to play a movie at a specific time each day.

22.1 Create an app event

An app *event* defines when a specific app runs. An event can only be assigned to one app; you cannot assign the same event to multiple apps. The event defines the app run period (eg, 2-3pm) and repeat interval (eg, daily or weekly). You create app events in the *App schedule screen*.

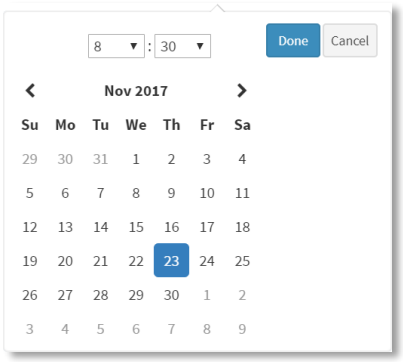


App schedule screen. 1 *Calendar.* 2 *Example app events.* 3 *App event attributes pane.* 4 *Start Time attribute.* 5 *Create new event pop-up.* 6 *Save button.* 7 *Scroll down the screen to see the event list.*

Follow these steps:

1. Click App Schedule App schedule in the left-hand pane.
2. In the *App schedule* screen, click a timeslot in the calendar to create a new event. (You will able to edit the times and dates in step 4.)
3. In the pop-up dialog, enter a name for the new event and click the Create button.
4. In the right-hand *App event attributes* pane, edit the following attributes:
   - **Start Time** and **End Time:** Specify when the app starts running and the end of its scheduled runtime. Click the input box to display a pop-up date & time picker.

     Notes
     - *A scheduled app will continue running after the **End Time** elapses unless a user switches to a different app or it is replaced by another scheduled app.*
     - *The **End Time** only takes effect if MT Showcase starts up midway through a scheduled app event. If so, the Editor uses the **Start Time** and **End Time** to calculate each event's runtime and so launch an event if it is supposed to be running 'now'.*



Date & time picker

   - **Event recurrence:** Specify how often the app runs.

     Tip: *If you choose a weekly repeat, you can specify which days of the week the app runs on.*
   - **Target app:** Choose which app runs when the current schedule starts.
   - **Warning duration:** By default, a warning countdown appears on-screen before a scheduled app starts. You can set how long the countdown is, or you can set the countdown to zero so that the scheduled app starts with no warning
   - **Allow cancel:** By default, the warning countdown includes a Cancel button, allowing users to cancel the scheduled app. But if required, you can disable this attribute to hide the Cancel button.
5. Click the Save button to save the app event.

   The target app will run automatically at the scheduled time(s).

22.2 Create a widget event

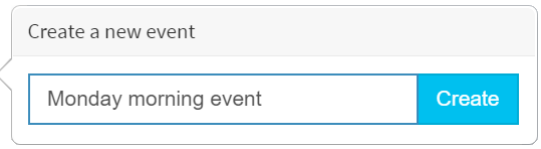
A widget *event* defines when a widget is visible (or active). The event defines the active period (eg, 2-3pm) and recurrence (eg, daily).

An individual widget can only have one event assigned to it, but you can assign the same event to multiple widgets. For example, you can assign the same event to a video viewer widget and background widget to play a specific movie, and display a specific background, at fixed times each day.

Follow these steps:

1. Click Widget Schedule Widget schedule in the left-hand pane.
2. In the *Widget schedule* screen, click a timeslot in the calendar to create a new event. (You will able to edit the times and dates in step 4.)

   Tip: *The Widget schedule screen has the same layout as the App schedule screen; see the screenshot on page 178.*
3. In the pop-up dialog, enter a name for the new event and click the Create button. For example, Monday morning event.



Create new event pop-up

4. In the right-hand *Widget event attributes* pane, edit the following attributes:
   - **Start time** and **End time:** Specify when the widget becomes active or visible on screen, and when it stops or closes. Click the input boxes to display a pop-up date & time picker; see the screenshot on page 179.
   - **Event recurrence:** Specify how often the widget is active.

     Tip: *If you choose a weekly repeat, you can specify which days of the week the event occurs on.*
5. Click the Save button to save the widget event.

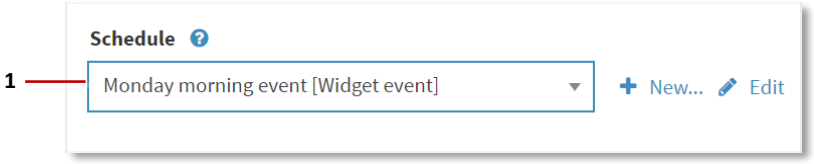
Now assign the new event to a widget in your app; continue to section 22.3.

22.3 Assign a widget event

You can now assign your new widget event to a widget. For example, to assign the event to the web browser in My First Structure (see step 3 in section 3.3), follow these steps:

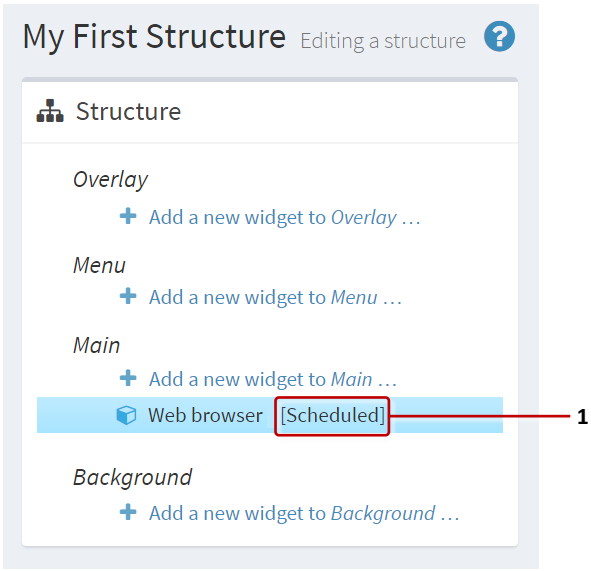
1. Edit the widget attributes:
   a. Click Structures Structures in the left-hand menu.
   b. Click the My First Structure hyperlink.
   c. In the Main section, click the Web browser widget.

2. Assign the widget event to the web browser widget.
   a. In the *Web browser attributes pane*, go to the **Schedule** attribute.
   b. Choose an event from the drop-down list. For example, Monday morning event.



Schedule attribute. **1** *Drop-down list of widget events.*

3. Click the Save button to assign the event.
4. Scheduled widgets are identified in the structure



Editing a structure screen. **1** *Scheduled widget indicator.*

22.4 Manage app and widget events

You can easily edit app events and widget events. For example, you can edit the start and stop times, or the recurrence. You can also rename events. Follow these steps:

1. Click App Schedule App schedule or Widget Schedule Widget schedule in the left pane.
2. In the Schedule screen, scroll down to the event list. (The event list is shown below the calendar.)
3. Click the event name to edit the event attributes or click the rename button.

   To delete an event, select its checkbox and click the Delete Selected button.

23 Import & export

You can export or import MT Showcase apps. You may want to do this for backup purposes or to import apps onto a different video wall.

You can export apps in their entirety, or you can selectively export structures, themes, service sets, Codice data, and media library assets. You can subsequently re-import data that you exported previously.

23.1 Exported files

Exported data is downloaded as a showcase file to your local computer. On Windows computers, the file is saved to the \Downloads folder in your user profile.

Exported files use this name format: <name>.showcase

Where <name> is the optional name or description you provide when you run an export operation. For example, MyExport.showcase.

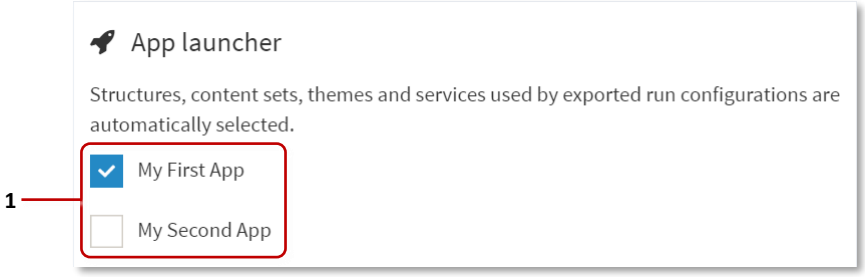
23.2 Export data from MT Showcase

You can export any combination of data on your MT Showcase server. Follow these steps:

1. Click Import & Export Import & Export in the left-hand menu.
2. Go to the *Export a File* section.
3. (*Optional*) Add a short name or description for your file. This name or description is incorporated into the exported filename.
4. You can export entire apps or individual structures, themes and service sets.

   Apps are always exported in their entirety with their associated structure, theme, and service set. If you do not want to export an entire app, you can selectively export its structure, theme or service set. Do one of the following

   - Select which apps you want to export, or
   - Select which structure, theme or service set you want to export.



Import & Export screen, Export a File section. **1** *Check boxes for exporting apps*

5. (*Optional*) Export the Codice database.

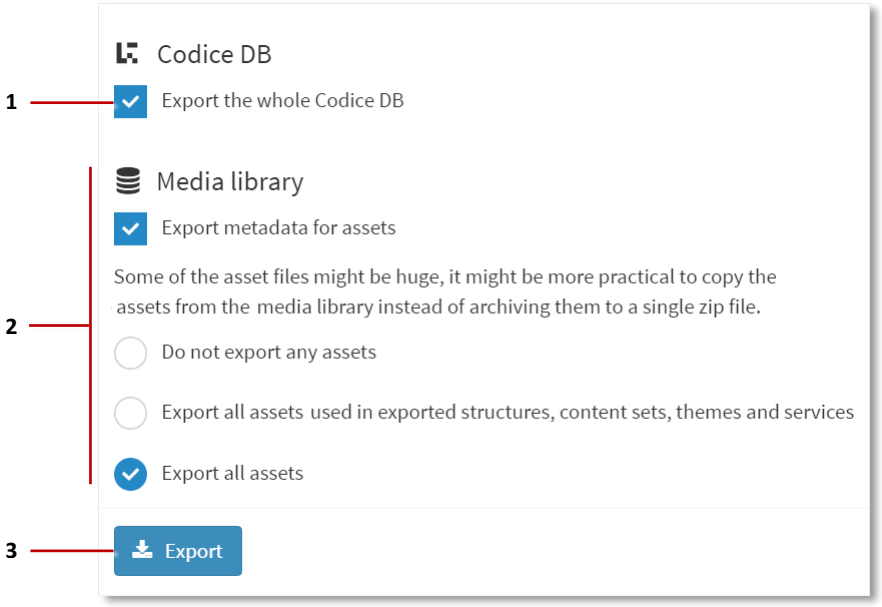
   Select the Codice DB checkbox to export details about all registered Codices (personal markers and erasers; see section 13).

6. (*Optional*) Export the media library.

   You can choose to export all items ('assets') in the media library or only items that are currently used by a structure, theme or content set.

   Tip: *If your media library is very large, it may be more practical to maintain separate copies of library items (for example, in a backup folder on your network), instead of exporting all items to a showcase file.*

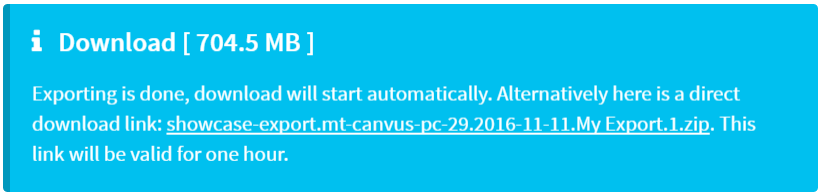
7. Click the Export button.



Import & Export screen. 1 *Export check box for Codice database.* 2 *Export options for media library.* 3 *Export button.*

8. Exported data is saved to a showcase file, as described in section 23.1.

   An advisory at the bottom of the *Import & Export screen* provides a time-limited hyperlink to the downloaded file.



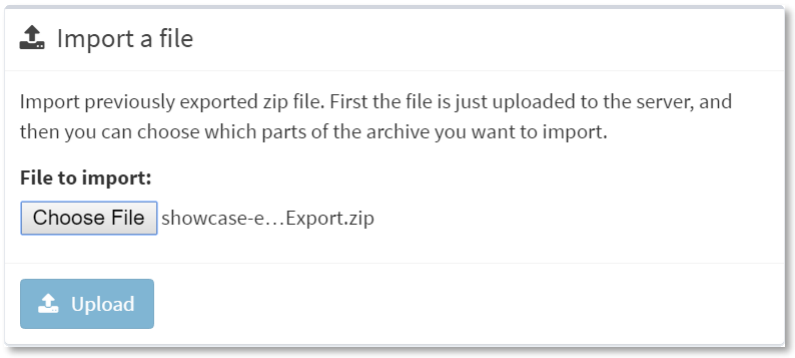
Download advisory for showcase file containing exported data

23.3 Import data into MT Showcase

You can re-import files of MT Showcase data that you exported previously.

Follow these steps:

1. Click Import & Export Import & Export in the left-hand menu.
2. Go to the Import a File section.
3. Click the Choose File button and browse to the showcase file you want.
4. Click the Upload button.



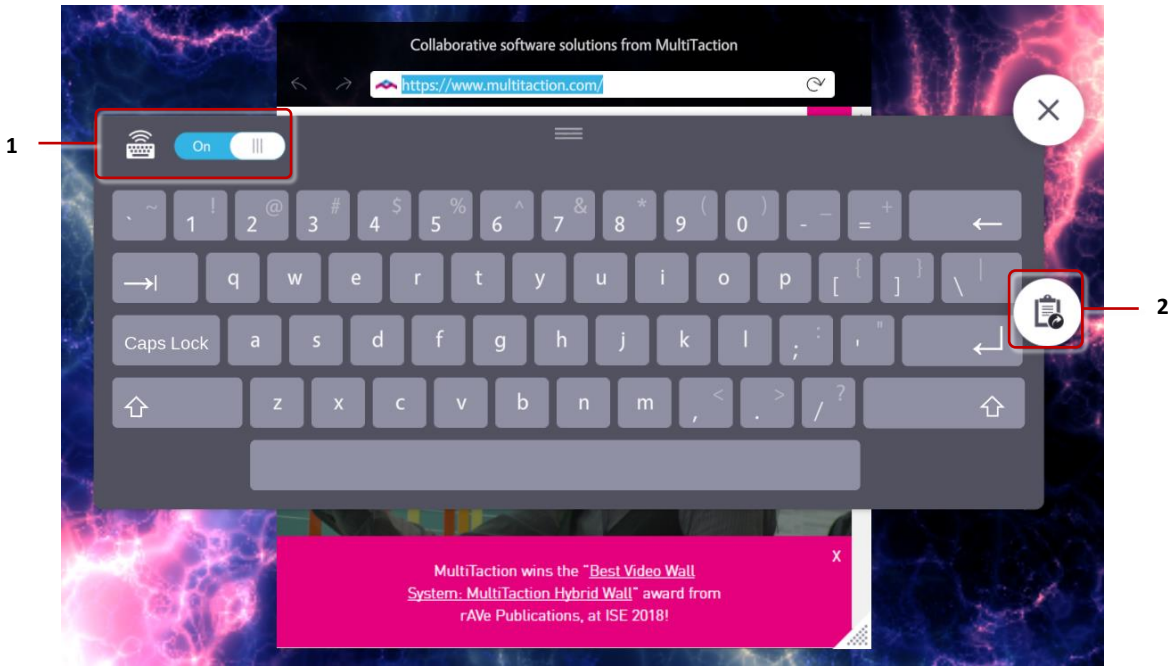
Import & Export screen, Import a File section

24 Keyboard

When a user taps an editable text field in Showcase, for example a web browser address field, an on-screen keyboard is displayed. User can use this on-screen keyboard to write text in the field. There can be multiple text fields active at the same time, each with their own on-screen keyboard.

You can also connect a physical keyboard to the machine wirelessly or with a cable. See section 24.1 for details about controlling which text field gets the physical keyboard input.

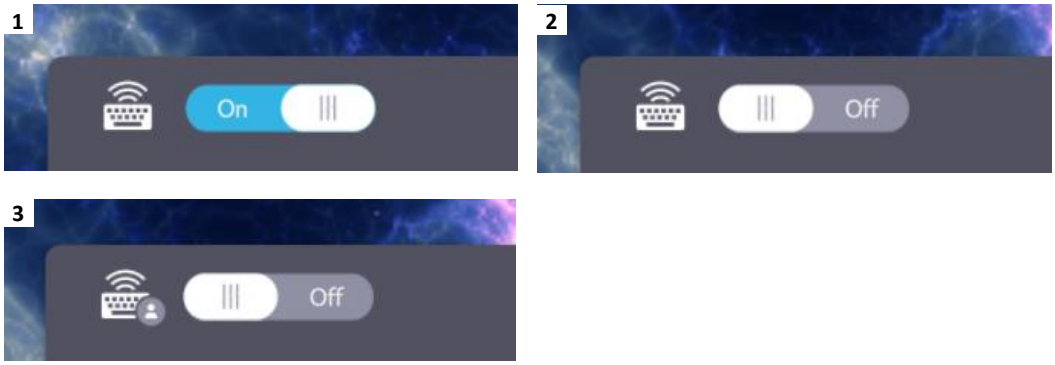
To speed up text entry and avoid typing errors, MT Showcase supports predefined text, also known as 'Snippets'. User can simply tap an item in the snippets list, attached to the on-screen keyboard, to input the text to the text field. See section 24.2 for more information on configuring text snippets for Showcase.



Showcase on-screen keyboard. 1 *Physical keyboard focus indicator.* 2 *Snippets list button.*

24.1 Keyboard focus

There can be multiple text fields active in Showcase at the same time, each with their own on-screen keyboard. In addition, a single text field can have the physical keyboard focus, that is, when you type with the physical keyboard the text goes to that field. When you tap an editable text field in Showcase, it automatically gets the physical keyboard focus if nothing else currently has focus. Otherwise you can use the focus slider on the on-screen keyboard to move the physical keyboard focus to the current text field.



Keyboard focus indicator. 1 *Current field has focus.* 2 *Nothing has focus.* 3 *Another field has focus.*

24.2 Snippets

To speed up text entry and avoid typing errors, MT Showcase supports predefined text, also known as 'Snippets'. When you use an on-screen keyboard to type, you can insert predefined snippets. This is useful when you need to repeatedly enter the same text string such as a web site URL or email address.

24.2.1 Add snippets to your app

Snippets are defined in a *Snippets* service. Follow these steps to add a Snippets service to your app:

1. Click Services in the left-hand pane.
2. Either click an existing service set. For example, My Service Set.

   Or create a new service set (see section 12.1.2).
3. In the Editing a service set screen, go to the Add new service section:
   a. Select **Snippets** from the Select Service drop-down menu.
   b. Enter a name for the new Snippets service. For example, My Snippets.
   c. Click the Add button.
4. Still in the Editing service set screen, go to the new *Snippets* section.
5. Add your predefined text snippets to the Snippets service.
   a. Click Create new button.

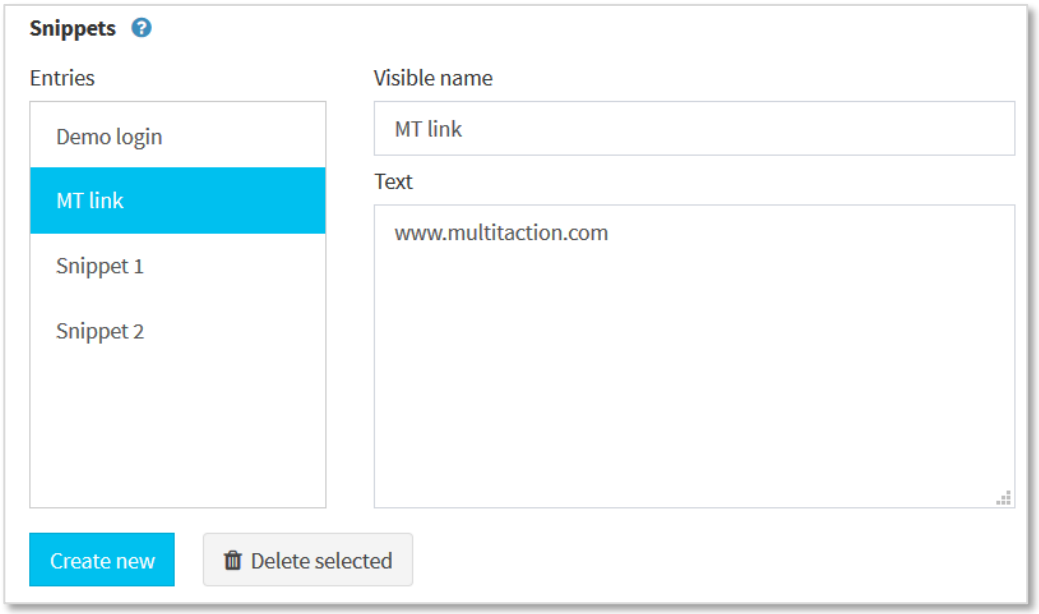
b. *(Optional)* Enter a **Visible name** for the snippet. This name will be displayed in the snippets list. If you do not define a Visible name, it is generated automatically from the text content.

c. Enter the **Text** for the snippet. This text is entered in the text field when user taps this snippet.

d. Repeat for any other snippets you wish to add.

6. Click the Save button to add the Snippets to your service set.

7. If you have not already done so, remember to add the service set containing the Snippets service to your app; see section 12.6.

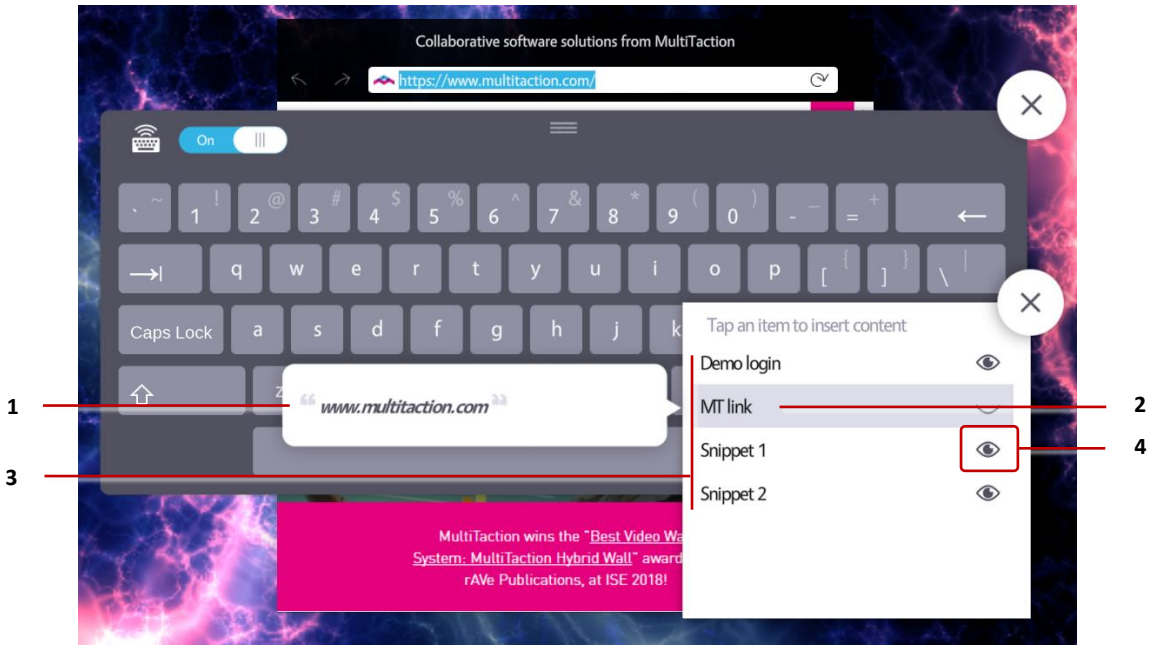


Editing a service set screen, Snippets service. Text snippets editor.

24.2.2 Insert snippets to text fields

Once you have added a *Snippets* service with predefined text snippets to your app, the on-screen keyboard in Showcase displays a new Snippets list button. Tap the button to display a list of configured snippets.

Tap a snippet item in the list to enter the associated text to the text field. You can tap the 👁 button to see the actual text that will be inserted.



On-screen keyboard with configured text snippets. 1 *Text content of a snippet.*
2 *Visible name of a snippet.* 3 *Tap an item to enter the text into the text field.*
4. *Tap the eye button to see the actual text that will be inserted.*

25 Other features

This section briefly introduces some other useful features in the MT Showcase Editor.

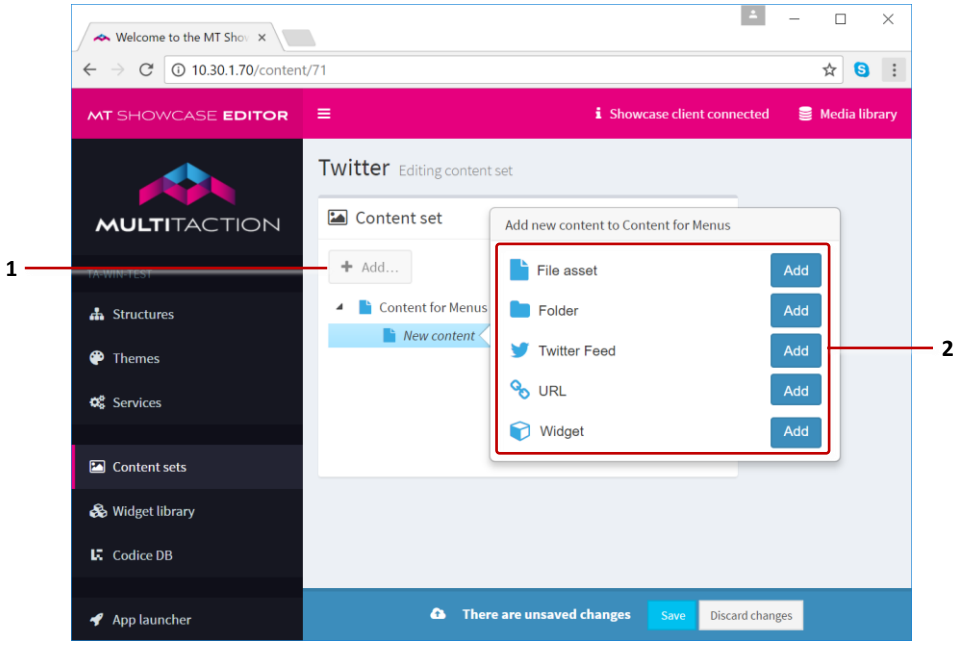
25.1 Advanced content sets

Note: *For an overview of content sets, see section 2.6.*

A *content set* supplies the items that are displayed by a finger menu, teasers or clouds. They typically include images, videos and PDFs from the media library, arranged in a flat structure. To add these items to a content set, simply drag items from the MT Showcase media library. Detailed instructions for setting up a content set are in section 5.1.

However, content sets can also include URLs, Twitter feeds, and widgets such as clouds and the Exit Showcase widget. Items in a content set can also be organized hierarchically to reflect the menus and submenus that you want in your app.

To add advanced features, you click the Add button in the *Editing a content set* screen.



Editing a content set screen. Adding advanced items. ***1*** *Add button.*
2 *Add new content pop-up menu.*

25.1.1 Folders and subfolders

If your MT Showcase media library already contains items organized into the correct folders and subfolders, simply drag the folder directly into the root folder of your content set to automatically create a corresponding menu structure.

Alternatively, follow these steps to create a new folder structure:

1. In the *Editing a content set* screen, click the root folder.
2. Click the Add button.

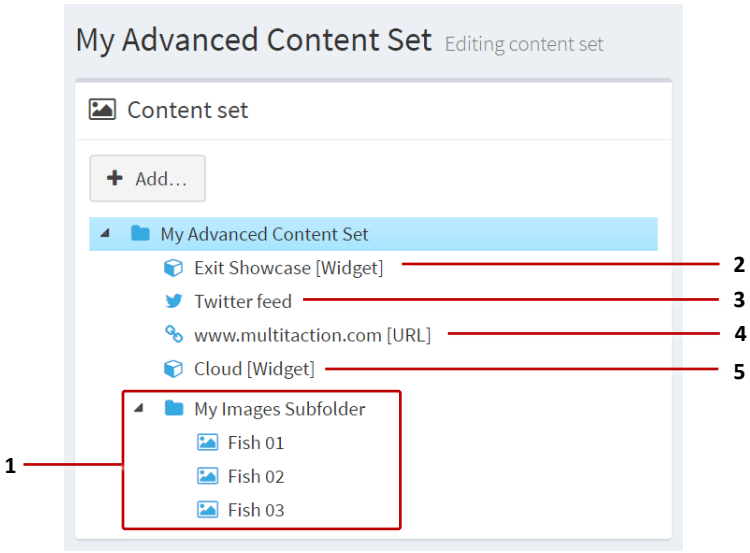
3. From the pop-up Add new content menu, add a folder.

   The folder is added to the content set as a subfolder.

4. *(Optional)* By default, the subfolder is named *New node*. But you can rename the subfolder to reflect its contents. For example, My Images Subfolder.

5. Add or drag items into the subfolder.

6. Click Save to save the content set.



Example advanced content set with subfolder. **1** *Subfolder and items for submenu.*
2 *Exit Showcase widget.* **3** *Twitter feed.* **4** *URL.* **5** *Cloud widget.*

25.1.2 Add a URL

You can add a specific URL to your content set. The URL automatically opens in a web browser. Follow these steps:

1. In the *Editing a content set* screen, click the root folder.

2. Click the Add button.

3. From the pop-up *Add new content* menu, add a URL.

   The URL is added as a new item in the content set (see the screenshot above).

4. Rename the new widget. For example, MyURL.com.

5. Configure the URL. In the right-hand *Content Attributes* pane:

   a. Set the **Visible name.** For example, MyURL.com.

   b. Set the **URL** field. For example, www.multitaction.com.

6. Click Save to save the content set.

25.1.3 Add a Twitter feed

You can add a Twitter feed to your app based on search terms defined by you. For example, you can add a Twitter feed that displays tweets relating to MultiTaction iWall or @multitaction.

Instructions for adding a Twitter feed to a content set are in section 17.4.1.

25.1.4 Add a widget

You can add a wide range of widgets to a content set. You typically use this feature to add items such as a cloud, an Exit Showcase widget or an Input Visualizer.

- **Cloud:** Instructions for adding this widget to a content set are in section 10.2.
- **Exit Showcase:** Instructions for adding this widget to a content set are in section 25.4.2.
- **Input Visualizer:** You may want to add an Input Visualizer to a content set for administrative purposes. For example, see section 11.1.

  Instructions for adding this widget to a content set are in section 25.1.5.

Note: *You can use this feature to add almost any widget to your content set, such as an Image Viewer or Video Viewer. But if you want to add items such images, videos or PDFs to your content set, it is easier to drag them directly from the media library as described in section 5.1.*

25.1.5 Widget example: Add an Input Visualizer to a content set

Follow these steps:

1. In the *Editing a content set* screen, click the root folder.
2. Click the Add button.
3. From the pop-up *Add new content* menu, add the Widget.
   The widget is added as a new item in the content set.
4. Rename the new widget. For example, Input Visualizer.
5. Configure the new widget to be an Input Visualizer. In the right-hand *Content Attributes* pane:
   a. Set the **Visible name** to Input Visualizer.
   b. Go to the **Widget** field and click the New button.
   c. In the *Add a new widget* pop-up, add the Input Visualizer widget.
   d. Click Save to save the content set.

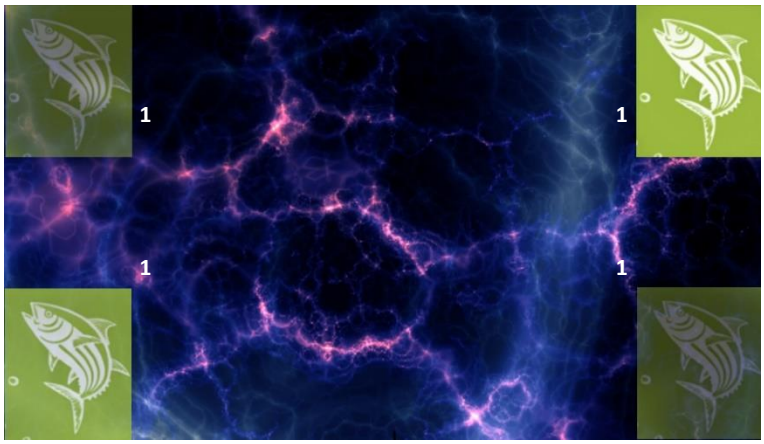
25.2 Animated content

The 'Animated content' feature allows you to apply animation to images or videos in the Overlay or Background layers of your app. Specifically, you can apply animation to Image viewer, Image movie, and Video viewer widgets.

This feature supports two types of animation:

- **Floating images or videos:** The image widget or video widget floats slowly across the screen. (Teaser menus also use this animation style; see section 6.)
- **Fading images or videos:** The image widget or video widget pulses rhythmically, fading in and out, at random screen locations, or random locations inside the screensaver area; or in the screen corners.

In both cases, the images or videos are non-interactive. For example, users cannot resize animated content images or pause animated content videos.



Animated content example. Here, the animated content comprises an Image Viewer widget (**1**) *that fades in and out at the screen corners.*

25.2.1 Add animated content to your app

Follow these steps:

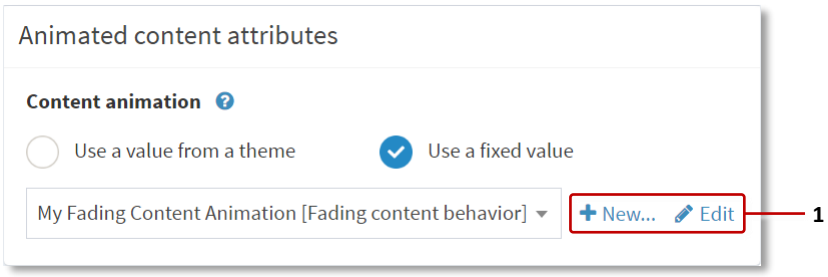
1. Click Structures  in the left-hand pane.
2. Click the structure you want to edit. For example, My First Structure.

   The *Editing a Structure* screen opens.
3. Go to the Background section and click *Add a new widget to Background*.

   Or go to the Overlay section and click *Add a new widget to Overlay*.

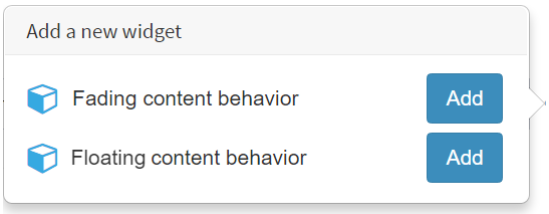
   Note: *You can add animated content to the Overlay or Background layers. Overlay animated content displays on top of content in the Main and Background layers. Background animated content displays behind content in the Main and Overlay layers.*
4. From the pop-up widget menu, choose Animated content.

5. Add a content animation widget to your structure:
   a. In the *Animated content attributes* pane, select a **Content animation** widget.
   b. Click the New button.



Animated content attributes pane, Content animation attribute. **1** *New and Edit buttons.*

   c. In the *Add a new widget* pop-up, choose the type of animation you want: **Fading content behavior** or **Floating content behavior**.



Add a new widget pop-up: available behaviors for content animation

   d. Type a name for the new Content animation widget. For example, Fading Content Animation.
   e. Click the Save button.

6. Edit the new content animation widget
   a. Still in the *Animated content attributes* pane, click the Edit button to display the *Editing a widget* screen.
   b. Edit the attributes for your content animation widget. If you added:
      - **Fading content behavior:** The available attributes determine how often images or videos appear and disappear and whether they fade in and out, and expand or shrink, as they appear and disappear. Note also the **Only use corners** attribute; if this attribute is enabled, items only appear in the screen corners.
      - **Floating content behavior:** The available attributes cover the speed and direction of widget movement, the 'shape' of the widget undulations across the screen, and so on. (See also section 9.5.2.)

7. Click the Save button.

Now set up the content that you want to animate. Go to section 25.2.2.

25.2.2 Set up the animated content

Follow these steps:

1. In the *Editing a Structure* screen, click the ▷ triangle next to the Animated content widget that you added in section 25.2.1.

   The widget expands to reveal the *Add a new widget to Animated content* hyperlink. (See the screenshot below.)

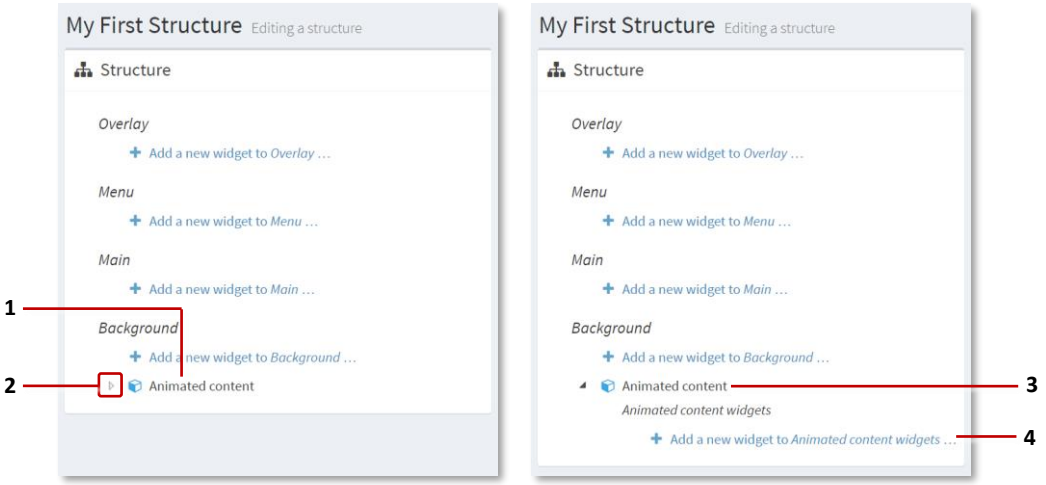
2. Click the *Add a new widget* hyperlink.

   a. Add the widget(s) you want. Choose from Image Viewer, Image Movie or Video Viewer.

   b. Set the attributes for the new widget(s). If necessary, use section 2.9 for guidance on how to edit widget attributes.

   Tip: *You can add multiple images or videos to a single animated content widget.*

3. Click Save to save the animated content.



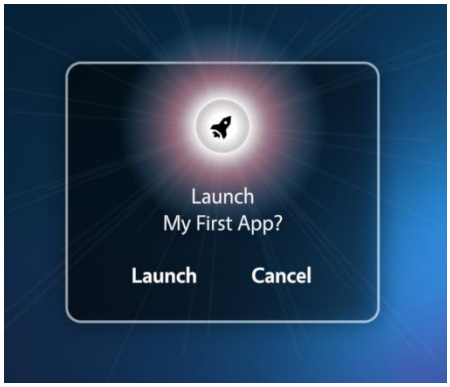
*Setting up animated content. **1** Animated content widget added to Background layer. **2** Triangle. Click to reveal widget hyperlink. **3** Expanded animated content widget. **4** 'Add a new widget to Animated content' hyperlink.*

25.3 App switcher widget

To allow users to easily switch to a different app, you can include an App switcher widget in your app. You can include this widget in, for example, a finger menu, a content hotspot, or as Codice content. For example, you could set up a 'switch' hotspot in the corner of your video wall screen.

By default, MT Showcase displays a confirmation dialog when it switches apps. But you can optionally configure the widget so that MT Showcase switches immediately, without prompting users to confirm or cancel the app switch.

Note: *Typically, you set up the app switcher to switch to a specific app. But you can instead set up the app switcher to display the welcome screen, allowing users to choose which app they switch to.*



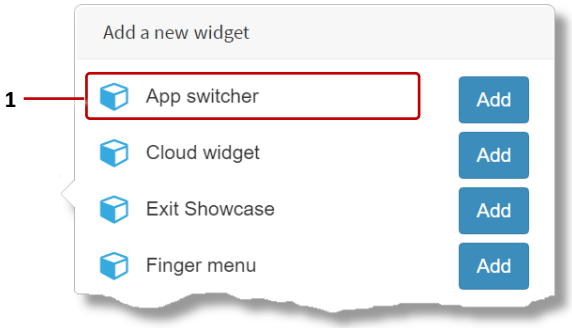
Switch app confirmation dialog

25.3.1 Create an App switcher content hotspot

Follow these steps:

1. Add a hotspot to your app's structure
   a. Follow the instructions in section 11.1. But when you set the general hotspot attributes in the *Content Hotspot attributes* pane:
      - Set the **Hotspot location** and **Hotspot size** to define a 100 x 100 pixel hotspot in the lower right screen corner. For example, on a MultiTaction Meeting Room video wall (with three Cells in portrait mode), set the Location to '**3160 1820**' and the Size to '**100 100**'.
      - Set the **Background color** for the Exit Showcase hotspot to white ie, rgba(255,255,255,1).
   b. (*Optional*) Right-click the content hotspot widget and click Rename. Enter a new name for the hotspot, such as 'Switch apps'.
   c. Click the Save button to add the content hotspot menu to your app's structure.

2. Add the App switcher widget to the hotspot.
   a. Follow the steps in section 11.1.2. For this example, we will not specify a hotspot visualization widget. But if required, you could specify a suitable image (such as the app launcher icon) to foreshadow the hotspot's content.
   b. Click the *Add* hyperlink for the **Widgets launched from hotspot around tap location** list.
   c. Add the App switcher widget.



Add a new widget pop-up menu. **1** *App switcher widget*

3. Set up the app switcher.

   You must now specify which app to switch to. You can also play sounds to accompany the app switch and, if required, suppress the confirmation dialog.

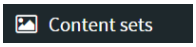
   Go to the right-hand *App switcher attributes* pane and follow the instructions in section 25.3.5.

This completes the setup for adding the App switcher widget to a content hotspot. Now you can test the hotspot; see section 11.4.

25.3.2 Add the App switcher widget to a finger menu

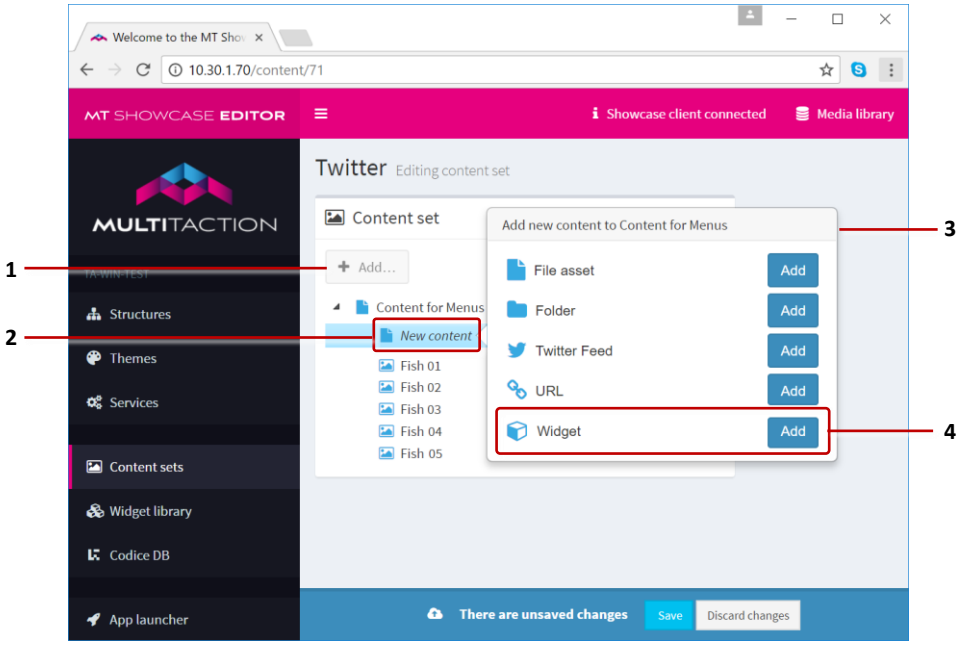
To add the App switcher widget to a finger menu, you simply edit the content set that 'supplies' the menu. For this example, you will edit the Content for Menus content set that you created in section 5.1. The finger menu that you set up in section 5.2 will then automatically include an App switcher menu item.

Follow these steps

1. Click Content sets 

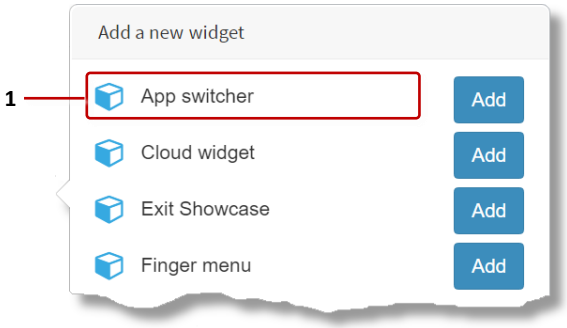
in the left-hand pane.
2. Click the Content for Menus content set.
3. Add a new widget to the content set.
   a. In the *Editing a content set* screen, click the root folder.
   b. Click the Add button.
   c. From the pop-up add new content menu, add the Widget.

   The widget is added as a new item in the content set.



Editing a content set screen. Adding a widget to a content set.
1 *Add button.* **2** *Rename the new content set item.* **3** *Pop-up 'Add new content' menu.*
4 *Add a Widget to the content set.*

4. Configure the new widget to be an App switcher widget. In the right-hand *Content Attributes* pane:
   a. Set the **Visible name**. This is the name for the new menu item that users will see. Enter a short descriptive name such as Switch apps or Show App #2.
   b. Go to the **Widget** field and click the New button.
   c. In the *Add a new widget* pop-up, add the App switcher widget.



Add a new widget pop-up menu. ***1*** *Exit Showcase widget*

   d. Enter a new name for the app switcher widget. For example, My app switcher.
   e. Click Save to save the content set.

4. You must now specify which app to switch to. You can also play sounds to accompany the app switch and, if required, suppress the confirmation dialog.

   Still in the right-hand *Content Attributes* pane:

   a. Go to the **Widget** field and click the Edit button.

   b. The *Editing a widget* screen now displays. Follow the instructions in section 25.3.5.

This completes the setup for adding the App switcher widget to a finger menu. Now you can test the menu; see section 5.6.

25.3.3 Add the App switcher widget to Codice content

You can add the app switcher to Codice content so that MT Showcase automatically switches apps when a user presents a Codice card (ie, holds it against the screen). For full setup instructions, see section 13.4.

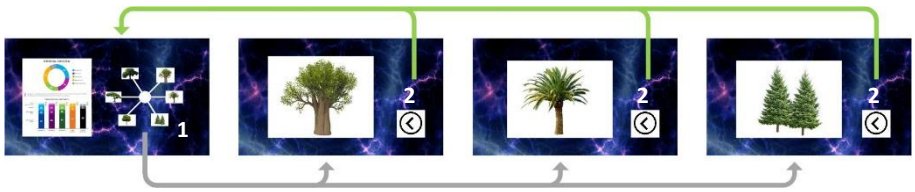
Briefly, you need to:

1. Click Codice DB in the left-hand pane.
2. In the *Codice Database* screen, go to the **Personal Markers** pane.
3. Specify the marker that you want to associate with the app switcher.
4. Click the *Create codice content* hyperlink.
5. Add the App switcher to the *Widgets launched* list.
6. Set up the app switcher; see the instructions in section 25.3.5
7. Click the Save button.

25.3.4 Create a collection of linked apps

Using app switcher widgets, you can create a collection of linked apps. Users can tap strategically organized app switchers to navigate around the collection.

For example, you can create a master app that features app switcher widgets, allowing users to navigate to the app they want. In the example below, the master app has a finger menu of app switchers linked to child apps. Each child app has an app switcher content hotspot that links back to the master app.



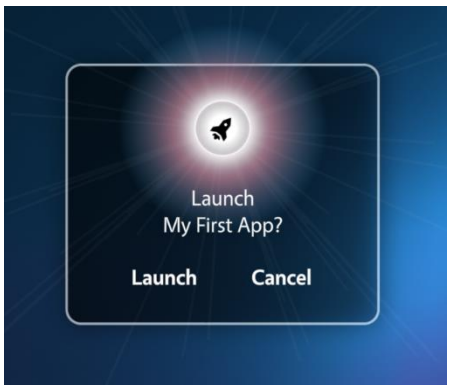
Example collection of linked apps. **1** *App switchers as finger menu items in master app.* **2** *App switchers as content hotspots in child apps.*

Note: *An alternative, simpler approach is to simply list the available apps on the welcome screen; see section 25.8.*

25.3.5 Set up the app switcher

By default, the app switcher widget causes MT Showcase to display a confirmation dialog when it switches apps. You can configure the size and location of the dialog, and you can play sounds to accompany the app switch. Alternatively, you can configure the app switcher to switch apps immediately, without displaying the confirmation dialog.

You can also specify a 'none' app switcher that simply switches back to the welcome screen.



Switch app confirmation dialog

After adding an app switcher to a content hotspot (section 25.3.1) or a menu (section 25.3.2), you can set up the app switcher by editing its attributes in the *Editing a widget* screen. Follow these steps:

1. Go to the right-hand *App switcher attributes* pane.
2. Go to the **Target app** attribute:
   a. Select 'Use a fixed value'.
   b. From the drop-down menu, choose the app you want to switch to.
      Or select 'None' to switch back to the MT Showcase welcome screen, allowing users to choose which app to launch.
3. *(Optional)* Disable the **Show confirmation before switch** attribute. This suppresses confirmation dialog, causing MT Showcase to switch apps immediately.
4. Edit the other attributes as required. For example, you can set **Interaction sounds** that play when the app switcher activates (see section 16).
5. Save the changes to the app switcher app.

25.4 Exit Showcase widget

To allow users to easily quit from MT Showcase, you can include the Exit Showcase widget in your app. You can include this widget in, for example, a finger menu, a content hotspot, or as Codice content. For example, you could set up a 'quit' hotspot in the corner of your video wall screen.

By default, MT Showcase displays a confirmation dialog when it exits. But you can optionally configure the widget so that MT Showcase exits immediately, without prompting users to confirm or cancel the exit.

Exit Showcase confirmation dialog

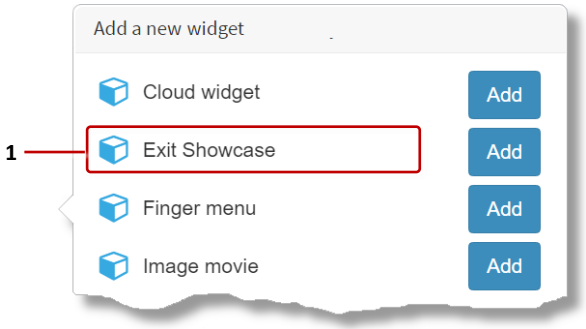
25.4.1 Create an Exit Showcase content hotspot

Follow these steps:

1. Add a hotspot to your app's structure
   a. Follow the instructions in section 11.1. But when you set the Add a content hotspot directly general hotspot attributes in the *Content Hotspot attributes* pane:
      - Set the **Hotspot location** and **Hotspot size** to define a 100 x 100 pixel hotspot in the lower right screen corner. For example, on a MultiTaction Meeting Room video wall (with three Cells in portrait mode), set the Location to '**3160 1820**' and the Size to '**100 100**'.
      - Set the **Background color** for the Exit Showcase hotspot to white ie, rgba(255,255,255,1).
   b. (*Optional*) Right-click the content hotspot widget and click Rename. Enter a new name for the hotspot, such as 'Exit Showcase'.
   c. Click the Save button to add the content hotspot menu to your app's structure.
2. Add the Exit Showcase widget to the hotspot.
   a. Follow the steps in section 11.1.2. For this example, we will not specify a hotspot visualization widget although you could, if required, specify an image of a standard Exit icon to foreshadow the hotspot's content.

b. Click the *Add* hyperlink for the **Widgets launched from hotspot around tap location** list.

c. Add the Exit Showcase widget.



Add a new widget pop-up menu. ***1*** *Exit Showcase widget*

3. *(Optional)* Disable the exit confirmation dialog. By default, MT Showcase displays a confirmation dialog when it exits. But you can optionally configure MT Showcase to exit immediately, without prompting users to confirm or cancel the exit.

   a. In the *Editing a structure* screen, click the Exit Showcase widget you added in step 2.

   a. Go to the right-hand *Exit Showcase attributes* pane.

   b. Disable the Show confirmation before exit attribute.

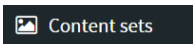
   c. Save the changes.

This completes the setup for adding the Exit Showcase widget to a content hotspot. Now you can test the hotspot; see section 11.4.

25.4.2 Add the Exit Showcase widget to a finger menu

To add the Exit Showcase widget to a finger menu, you simply edit the content set that 'supplies' the menu. For this example, you will edit the Content for Menus content set that you created in section 5.1. The finger menu that you set up in section 5.2 will then automatically include the Exit Showcase menu item.

Follow these steps

1. Click Content sets 

in the left-hand pane.

2. Click the Content for Menus content set.

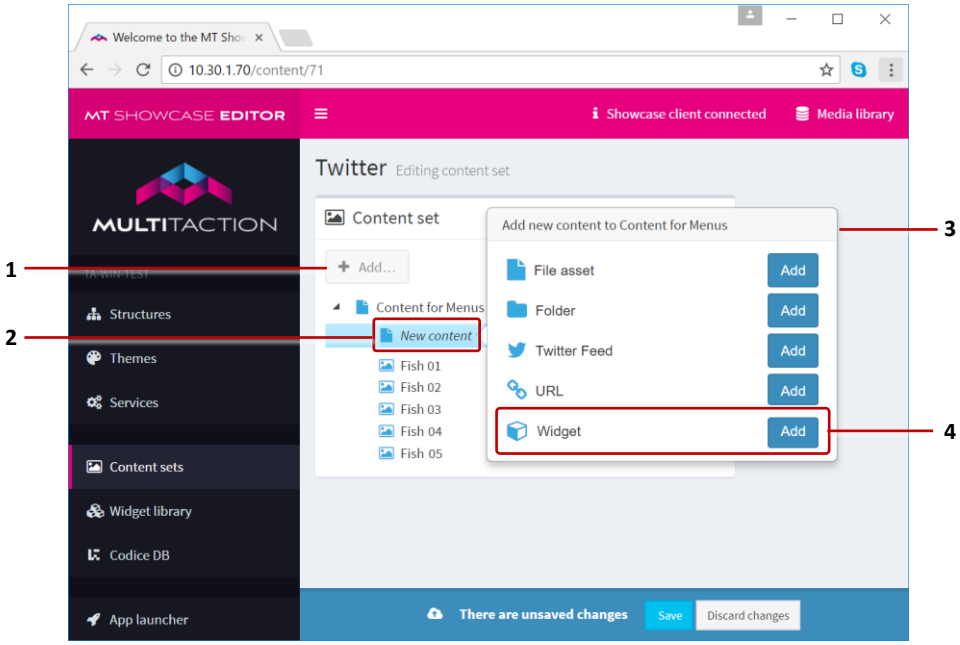
3. Add a widget to the content set.

   a. In the *Editing a content set* screen, click the root folder.

   b. Click the Add button.

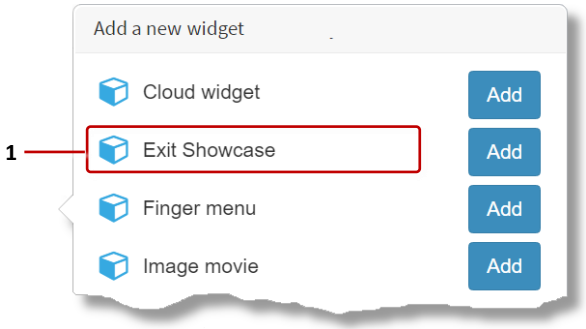
   c. From the pop-up add new content menu, add the Widget.
   The widget is added as a new item in the content set.

   d. Rename the new widget to Exit.



Editing a content set screen. Adding a widget to a content set.
1 *Add button.* **2** *Rename the new content set item.* **3** *Pop-up 'Add new content' menu.*
4 *Add a Widget to the content set.*

4. Configure the new widget to be an Exit Showcase widget. In the right-hand *Content Attributes* pane:
   a. Set the **Visible name** to Exit.
   b. Go to the **Widget** field and click the New button.
   c. In the *Add a new widget* pop-up, add the Exit Showcase widget.



Add a new widget pop-up menu. ***1*** *Exit Showcase widget*

   d. Enter a new name for the Exit Showcase widget. For example, My Exit Showcase.
   e. Click Save to save the content set.

5. *(Optional)* Disable the exit confirmation dialog. By default, MT Showcase displays a confirmation dialog when it exits. But you can optionally configure MT Showcase to exit immediately, without prompting users to confirm or cancel the exit.

   Still in the right-hand *Content Attributes* pane:

   a. Go to the **Widget** field and click the Edit button.

   b. In the *Editing a widget* screen, go to the right-hand Exit Showcase attributes pane.

   c. Disable the Show confirmation before exit attribute.

   d. Save the changes.

This completes the setup for adding the Exit Showcase widget to a finger menu. Now you can test the menu; see section 5.6.

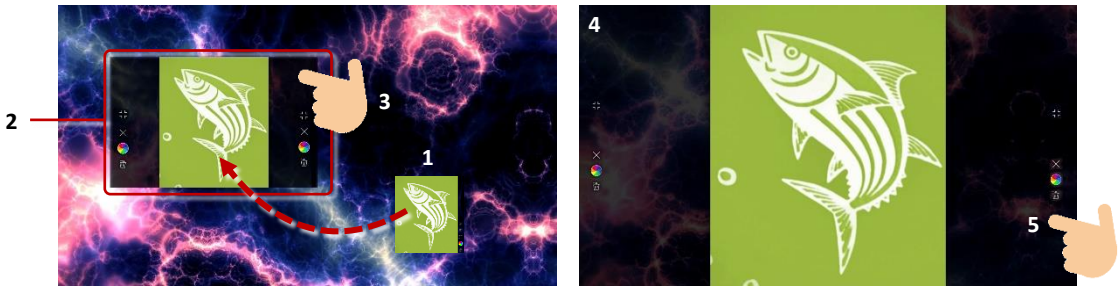
25.5 Maximization areas

A maximization area is an area on the screen that displays maximized widgets. An app can include multiple maximization areas.

When a widget is maximized (for example, when a user taps the Maximize toolbar button), it floats to the nearest maximization area and scales up to fill the area. If no maximization area is defined, the widget scales up to fills the entire screen. Note that when widgets are scaled, they retain their width-height ratio.

Why define a maximization area? They are mainly used on video walls that include a breakout display (a screen that is physically separate from the main video wall). In this situation, they allow widgets to be maximized on the main video wall only, without extending onto the breakout display. They are also useful if you want a non-default background color or timeout for maximized widgets.

Note: *While maximized, a widget is pinned and cannot be unpinned. Consequently, users cannot move a maximized widget, for example, to interact with other items hidden below the maximized widget.*



Example maximization area

1 *Original Image Viewer widget. If a user maximizes this image, it floats to the nearest maximization area and scales to fill the area.*

2 *Maximization area with default background color.*

3 *To unmaximize a widget, users tap either edge of the maximization area to display the Unmaximize button and (if configured) the widget toolbar.*

4 *If no maximization area is defined in the app, maximized images or videos expand to fill the entire screen area, including any breakout displays (not shown here).*

3 *To unmaximize a widget, users tap the screen edge to display the Unmaximize button.*

25.5.1 How to maximize and unmaximize a widget

You can configure an app to maximize automatically by enabling the **Maximize widget when it opens** attribute. All content widgets (for example, Image viewers, Video viewers and Web browsers) have this attribute. This attribute is also available as a *theme default*; see section 6.6.

Alternatively, users can manually maximize a widget if you can include a Maximize button in the widget's toolbar; see section 14.1.

To reset a maximized widget to its normal size, users tap the edge of the maximization area to display the Unmaximize button. This button will always display, even the maximized widget has no toolbar.

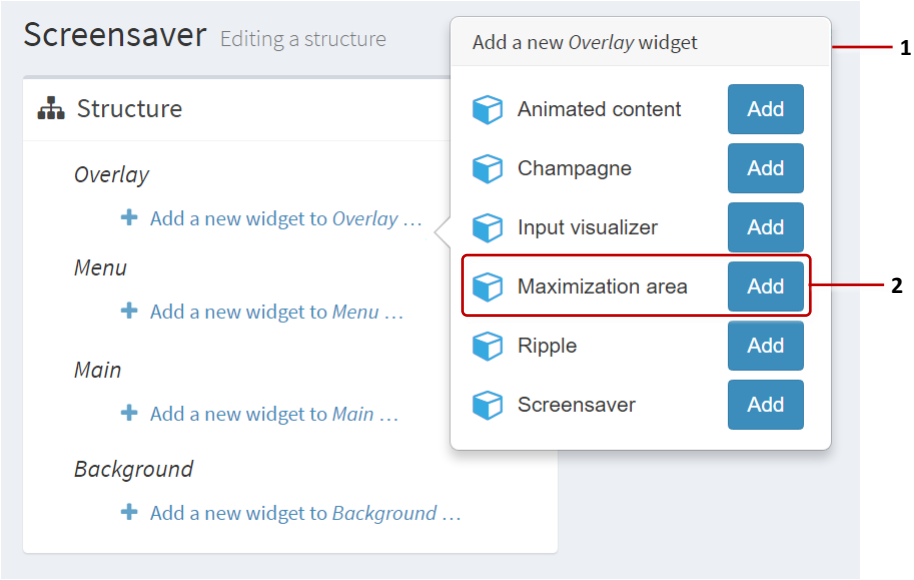
25.5.2 Add a maximization area

Follow these steps:

1. Click Structures 

in the left-hand pane.
2. Click the structure you want to edit. For example, My First Structure.

   The *Editing a Structure* screen opens.
3. Go to the Overlay section and click *Add a new widget to Overlay*.
4. From the pop-up widget menu, choose **Maximization area**.



Editing a structure screen. **1** *Pop-up menu of Overlay widgets.* **2** *Maximization area widget and Add button.*

5. In the *Maximization area attributes* pane, set the size and location attributes:
   - **Size** and **Location:** These attributes are self-explanatory.
   - **Background color:** (*Optional*) While a widget is maximized, you can configure the maximization area to display with a colored background. The default color is semi-transparent black ie, rgba(0,0,0,0.8).

     See section 21.4 for color input methods.
   - **Toolbars timeout:** Set the timeout for idle toolbars in the maximization area. If a toolbar is not used before the timeout expires, it closes. For widgets that have no toolbar, this timeout also applies to the Unmaximize button.
6. (*Optional*) If you intend to add multiple maximization areas to your app, you may want to give distinctive names to each area in the Editor.

   To rename an area, right-click the Maximization area widget and click Rename.
7. Click the Save button.

25.6 Opening animations

There are two types of opening animation in MT Showcase:

- **Opening animation:** This refers solely to an animation that plays when a user opens a finger menu. Two opening animations are supported:
  - **Circular:** This animation shows a circular progress bar when a user taps the screen to open a finger menu. You can specify the color of the progress bars.
  - **Stars:** This animation shows a spinning spiral of small stars when a user taps the screen to open a finger menu. You can specify the color of the stars and spiral rotation speed.
- **Content opening animation:** This refers to the visual effect of a widget fading in and expanding when it opens from a finger menu or content hotspot, or when a user opens some Codice content. You can control the speed of this animation to give your apps a slick, professional feel or a smooth, graceful feel.

  To control the animation speed, you simply set the duration of the animation. If you want items to open quickly, set a short duration (eg, 0.3 seconds). If you want smoother, more graceful behavior, set a longer duration (eg, 2 seconds).

The following sections describe how to set animations for finger menus, content hotspots, and Codice content.

25.6.1 Finger menu animations

You can choose which opening animation plays (Circular or Stars) when the finger menu opens. You can also specify how long menu items take to open.

Notes

- *Finger menus are described in section 4.*
- *This section describes how to set up opening animations by editing an app directly. But you can also set up opening animations in a theme. Open the theme you want, go to the* **finger menu** *component, and repeat the instructions below.*

To edit a finger menu directly, follow these steps:

1. Click Structures 

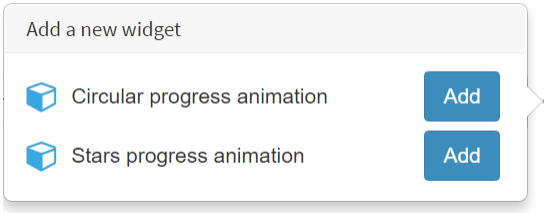
in the left-hand pane.

5. Click the structure that contains the finger menu you want to edit. For example, My First Structure.

   The *Editing a Structure* screen opens.

6. Go to the Menu section and click the Finger menu widget

7. In the right-hand attributes pane, go to the *Finger menu attributes* section.
   a. Go to the **Opening animation** attribute and click the New button.
   b. In the *Add a new widget* pop-up, click an Add button to select the animation you want, Circular or Stars.



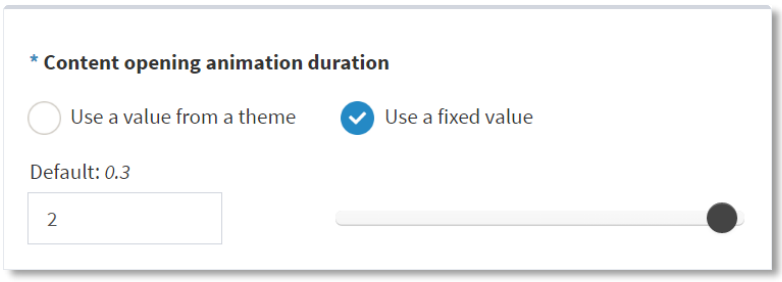
Add a new widget pop-up for opening animations.
Note: *In this pop-up, the opening animations are called 'progress animations'.*

   c. Enter a name eg, My Circular Menu Opening Animation.
   d. Click the Save button.

Note: *You have now created a new 'opening animation' shared widget. This widget is now listed in the widget library (see section 2.2.4) and can be assigned to finger menus in other apps. It is listed under the 'Circular progress animation' or 'Stars progress animation' widget types.*

8. (*Optional*) Customize the opening animation. For example, you can specify the color of the Circular animation.
   a. Click the Edit button.
   b. In *Editing a widget* screen, go to the right-hand attributes pane.
   c. Edit the attributes, as required. For example, edit the **Color** attribute.
   d. Click the Save button.

9. Now specify how long menu items take to open.
   a. Still in the *Finger menu attributes* section, go to the **Content opening animation duration** attribute.
   b. Specify how long menu items take to open (in seconds).
   c. Click the Save button.



Editing a structure screen: Content opening animation duration attribute.

25.6.2 Content hotspots

You can choose how long widgets take to open when launched from a content hotspot.

Notes

- *Content hotspots are described in section 11.*
- *This section describes how to set the duration of content opening animations by editing an app directly. But you can also set this duration in a theme. Open the theme you want, go to the* content hotspot *component, and repeat the instructions below.*

To edit a content hotspot directly, follow these steps:

1. Click Structures  in the left-hand pane.
2. Click the structure that contains the content hotspot you want to edit. For example, My First Structure.
   The *Editing a Structure* screen opens.
3. Go to the Main section and click the Content Hotspot widget.
4. In the right-hand attributes pane, go to the *Content hotspot attributes* section.
   a. Click the Show advanced attributes hyperlink:
   
   b. Go to the Content opening animation duration attribute.
   c. Specify how long menu items take to open (in seconds. See the screenshot in the previous section.
   d. Click the Save button

25.6.3 Codice content

You can choose how long widgets take to open when a user launches Codice content by presenting a Codice card.

Notes

- *Codice content is described in section 13.4.*
- *This section describes how to set the duration of content opening animations by editing an app directly. But you can also set this duration in a theme. Open the theme you want, go to the* codice content *component, and repeat the instructions below.*

To edit a Codice content directly, follow these steps:

1. Click  in the left-hand pane.
2. In the Codice Database screen, go to the Personal Markers pane.
3. Go to a personal marker that has Codice content assigned to it.

4. Click the Edit codice content hyperlink.
   The *Editing a Structure* screen opens.
5. In the left-hand *Structure* pane, click the Codice Content widget.
6. In the right-hand *Codice content attributes* pane:
   a. Click the Show advanced attributes hyperlink:

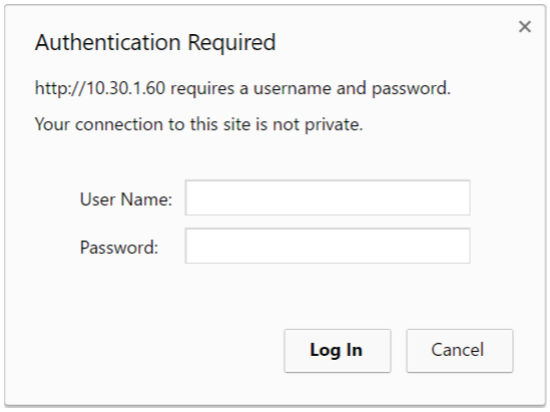
      

   b. Go to the Content opening animation duration attribute.
   c. Specify how long menu items take to open (in seconds). See the screenshot on page 207.
   d. Click the Save button.

25.7 Password protection

The MT Showcase Editor can be password-protected to prevent unauthorized changes to apps, structures, themes and so on. When the Editor is protected, any app designer who wants to open the Editor must enter the correct user name and password.

Password protection for the Editor can be enabled (or disabled) by an MT Showcase administrator. It is the administrator's responsibility to inform app designers of the user name and password needed to access the Editor. For setup details, see the *MT Showcase Editor Installation Manual*. Registered users can download this manual from https://cornerstone.multitouch.fi/showcase



MT Showcase Editor authentication dialog

25.8 Welcome screen with app list

By default, available apps are listed on the MT Showcase welcome screen. This is generally useful, though you can remove apps if required:

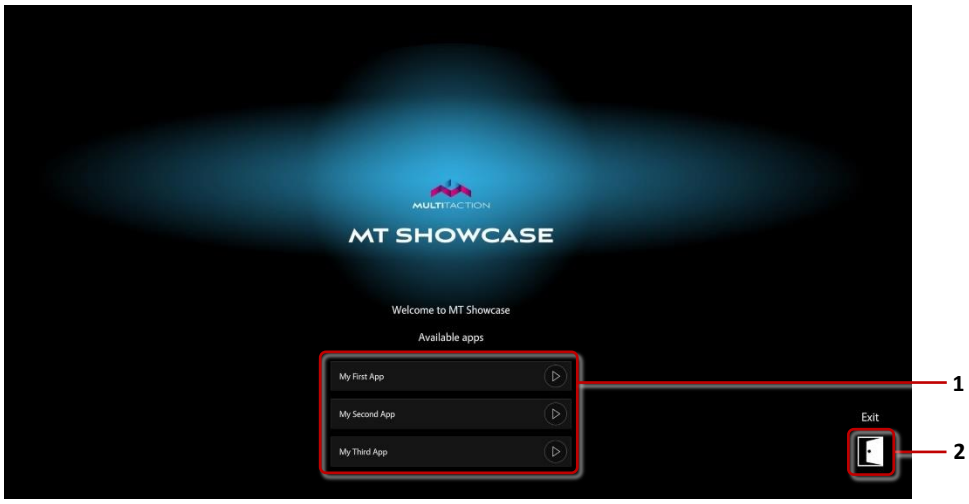
- **Safeguard against deleted apps:** Listing the available apps is a useful safeguard if users are inadvertently directed to the welcome screen. For example, if a user taps an MT Launcher tile to open an MT Showcase app that no longer exists, the user is taken instead to the MT Showcase welcome screen.

  Note: *MT Launcher is designed to run on video walls and provide end-users with a simple method for launching applications such as MT Showcase or MT Canvus. For setup details, see the MT Launcher Installation Manual. Registered users can download this manual from* https://cornerstone.multitouch.fi/mt-launcher.

- **Allow users to choose an app:** Listing the available apps can also be a useful way of allowing users to choose the app they want. If so, you will need to provide users with a method of getting to the welcome screen; see section 25.8.1.

  Note: *An alternative, more sophisticated approach is to create a collection of linked apps, based on app switcher widgets. Users can tap strategically organized app switchers to navigate around the collection; see section 25.3.4.*

- **Remove apps from the welcome screen:** Alternatively, you may want to keep some apps private. That is, you do *not* want them listed on the welcome screen; see section 25.8.2.



Welcome screen. 1 *Available apps.* 2 *Exit button.*

25.8.1 Provide a method to access the welcome screen

You can allow users to access the welcome screen from an MT Showcase app or from a tile in MT Launcher:

- You can add a 'none' app switcher widget to your app that switches back to the welcome screen. The app switcher can be a finger menu item, a content hotspot, or Codice content. For details, see section 25.3.5.
- You can set up an MT launcher tile that, instead of launching an app, opens the welcome screen. To do this, you must append `--app,none` to the `arguments` setting in MT Showcase pipeline configuration screen.

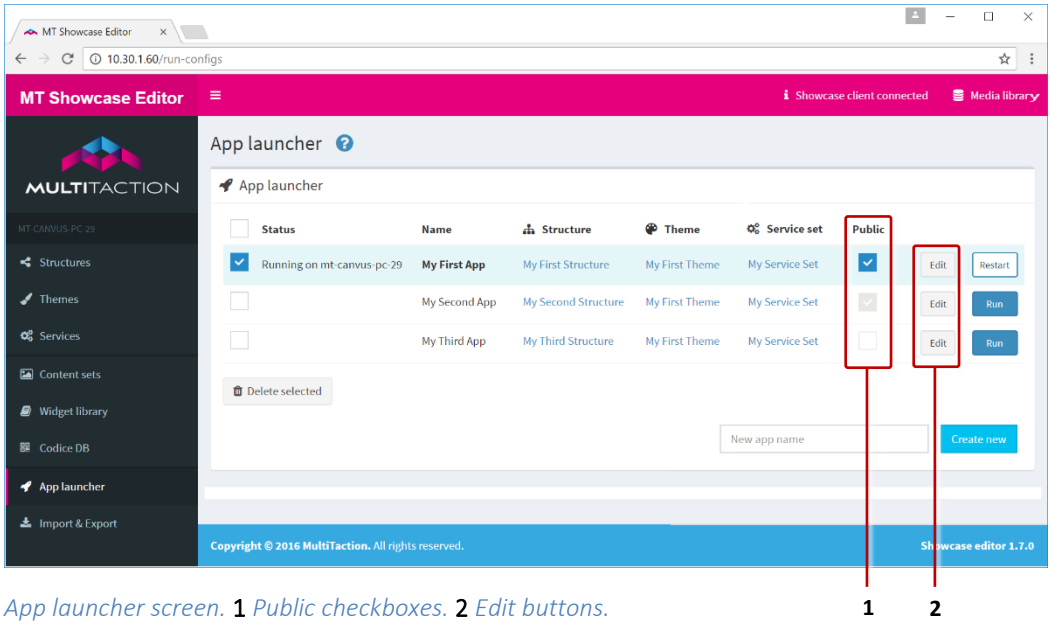
  For full details, see the *MT Launcher Installation Manual*. Registered users can download this manual from https://cornerstone.multitouch.fi/mt-launcher

25.8.2 Remove an app from the welcome screen

Follow these steps:

8. Go to the *App launcher* screen.
9. Click the Edit button for the app you want.
10. Clear the Public checkbox.
11. Click the Save button.

The app is no longer listed on the welcome screen.



App launcher screen. 1 *Public checkboxes.* 2 *Edit buttons.*

25.9 Closing content

There are several ways to close content in Showcase:

- Widget interaction: You can simply throw a content widget such as an Image viewer or Video viewer out of the screen. It is automatically closed when it is outside the visible area. You can tap the root node of a menu to close the menu.
- Close button: You can add a close button to the widget's toolbar. User can tap the button to close the widget. See section 14 for more information on widget toolbars.
- Close after idle: You can make content close automatically after it has been idle for a while. Most content widgets have a *Close after idle* attribute. You can set this attribute to a shared widget that automatically closes the content when it is idle, and choose the timeout value. Menus also have a *Close after idle* attribute that affects all the widgets launched from the menu. You can make the menu itself close by editing the *Menu idle timeout* attribute.
  See section 6.3.2 for an example of configuring *Close after idle* attribute for a Finger menu.
- Close content on next tap: Menus and content hotspots have a *Close content on next tap?* attribute. When enabled for a hotspot, it will close its content when tapped instead of launching new content if it already had open content. When enabled for a menu, tapping a menu node will close any previous content launched from the same menu and then open that node's content if it was not previously open.
- Close content when launcher is closed: Menus, content hotspots and Codice content each have an attribute to close content when the menu, hotspot or Codice is removed. Normally this only affects content launched directly by that launcher. For example, if you enable this for a menu, when you launch a hotspot image from that menu only the image is closed when the menu is closed. Anything launched from the hotspot image is left open. If you also enable this attribute for the hotspot, then closing the menu closes the hotspot, which in turn closes any additional content launched from that hotspot.
- Screen wipe: You can add a *Screen wipe* widget to your app. You can add this widget to a menu, hotspot or Codice content. When this widget is launched, it immediately closes all user-launched content on screen. This includes content such as open menus and widgets launched from menus, hotspots and Codices.
- App switch: When you switch apps or restart an app from the Editor, the current app is completely reset. All content is closed, videos are reset to start, annotations are cleared etc. Next time you load the app, it is back in its initial state.