



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

Widgets Developer Resources

ChannelSelector

Contents

- [1 Overview](#)
 - [1.1 Usage](#)
 - [1.2 Customization](#)
 - [1.3 Namespace](#)
 - [1.4 Mobile support](#)
 - [1.5 Screenshots](#)
- [2 Configuration](#)
 - [2.1 Example](#)
 - [2.2 Options](#)
- [3 Localization](#)
 - [3.1 Usage](#)
 - [3.2 Example i18n JSON](#)
- [4 API commands](#)
 - [4.1 close](#)
 - [4.2 open](#)
 - [4.3 configure](#)
- [5 API events](#)

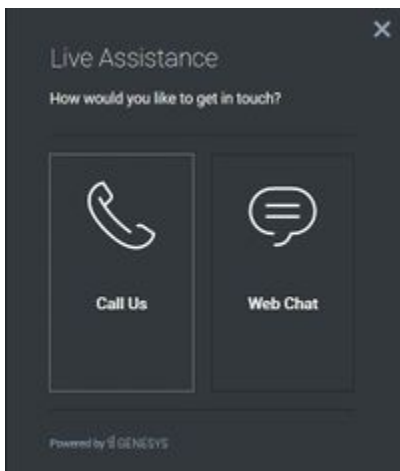
Learn how to provide your customers with a configurable list of channels as an entry point for contacting customer service in Genesys Cloud CX.

Related documentation:

-

Overview

The ChannelSelector Widget displays a configurable list of channels, as an entry point for customers to contact customer service. Channels are not limited to Genesys Widgets; you can add your own custom channels to open applications or open new windows as necessary.



Usage

Use the following methods to open ChannelSelector manually:

- Call the **ChannelSelector.open** command
- Create your own custom button, or link to open ChannelSelector (using the **ChannelSelector.open** command)

Important

By default, ChannelSelector has no channels configured. If not configured, the UI appears empty. See the configuration for examples and information on how to set up your own custom channels.

Customization

You can customize and localize the static text shown in the ChannelSelector Widget by adding entries into your configuration and localization options.

ChannelSelector supports themes. You can create and register your own themes for Genesys Widgets.

Namespace

The Channel Selector plugin has the following namespaces tied to each of the following types:

Type	Namespace
Configuration	channelselector
i18n—Localization	channelselector
CXBus—API commands & API events	ChannelSelector
CSS	.cx-channel-selector

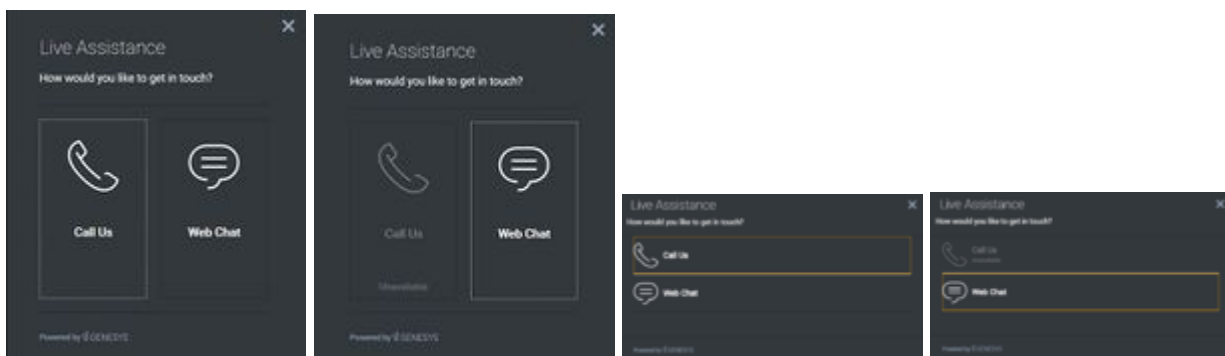
Mobile support

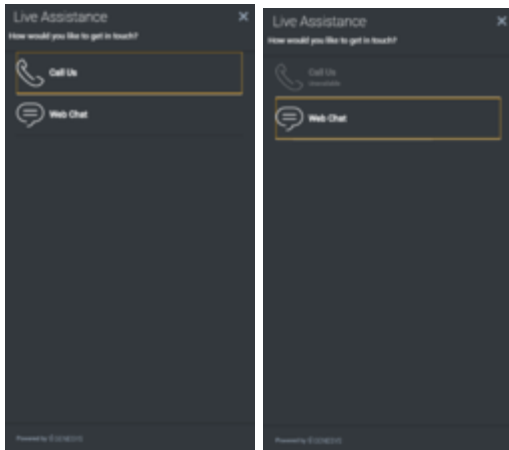
ChannelSelector supports both desktop and mobile devices. Like all Genesys Widgets, there are two main modes: Desktop and Mobile. Desktop is employed for monitors, laptops, and tablets. Mobile is employed for mobile devices. When ChannelSelector detects a mobile device, ChannelSelector switches to special full-screen templates, optimized for both portrait and landscape orientations.

Switching between Desktop and Mobile modes is automatic, by default. If necessary, configure Genesys Widgets to switch between Desktop and Mobile modes manually.

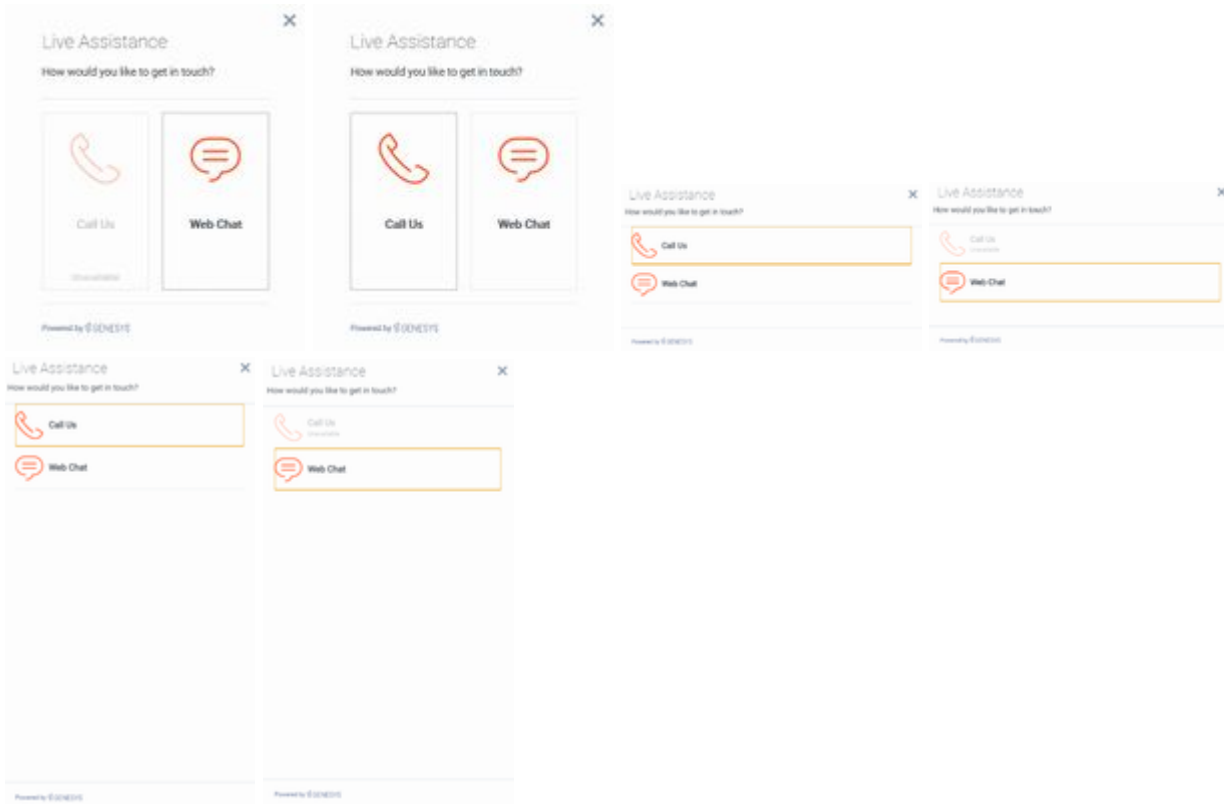
Screenshots

Dark theme





Light theme



Configuration

ChannelSelector shares the **_genesys.widgets.channelselector** configuration namespace. ChannelSelector has UI options to enable and disable channels, hide channels, add new channels. All

the channels are displayed based on the array of objects order defined in the channel's configuration. To hide a particular channel, simply remove the corresponding array object.

Example

```
window._genesys.widgets.channelselector = {
  channels: [{
    enable: true,
    clickCommand: 'CallUs.open',
    displayName: 'Call Us',
    i18n: 'CallusTitle',
    icon: 'call-outgoing',
    html: '',
  },
  {
    enable: true,
    clickCommand: 'WebChat.open',
    displayName: 'Web Chat',
    i18n: 'ChatTitle',
    icon: 'chat',
    html: '',
  }]
};
```

Options

Name	Type	Description	Default	Required
channels[].enable	boolean	Enable/disable a channel.	true	N/A
channels[].clickCommand	string	The CXBus command name for opening a particular widget when this channel is clicked.	none	Always
channels[].displayName	string	A channel name to display on the ChannelSelector Widget.	none	Always
channels[].i18n	string	To support localization of the channel display name, this takes a key parameter of the channelselector section in the language pack file. Overrides above displayName.	none	N/A
channels[].icon	string	Select from one of the Genesys Widgets icons by specifying icon css class name.	none	Always

Name	Type	Description	Default	Required
channels[].html	string	Overrides and replaces the icon section of a channel with the html (image tag) defined here.	none	N/A

Localization

Important

For information on how to set up localization, refer to the Localize widgets and services guide.

Usage

Use the **channelselector** namespace when you define localization strings for the ChannelSelector plugin in your i18n JSON file.

The following example shows how to define new strings for the **en** (English) language. You can use any language codes you wish; there is no standard format. When selecting the active language in your configuration, you must match one of the language codes defined in your i18n JSON file. You must only define a language code once in your i18n JSON file. Inside each language object, you must define new strings for each widget.

Example i18n JSON

```
{
  "en": {
    "channelselector": {
      "Title": "Live Assistance",
      "SubTitle": "How would you like to get in touch?",
      "UnavailableTitle": "Unavailable",
      "AriaClose": "Live Assistance Close",
      "AriaWindowLabel": "Live Assistance Window"
    }
  }
}
```

API commands

Once you've registered your plugin on the bus, you can call commands on other registered plugins. Here's how to use the global bus object to register a new

plugin on the bus.

Important

The global bus object is a debugging tool. When implementing Widgets on your own site, do not use the global bus object to register your custom plugins. Instead, see Genesys Widgets Extensions for more information about extending Genesys Widgets.

```
var oMyPlugin = window._genesys.widgets.bus.registerPlugin('MyPlugin');
oMyPlugin.command('ChannelSelector.open');
```

close

Closes the ChannelSelector UI.

Example

```
oMyPlugin.command('ChannelSelector.close').done(function(e){
    // ChannelSelector closed successfully
}).fail(function(e){
    // ChannelSelector failed to close
});
```

Resolutions

Status	When	Returns
resolved	ChannelSelector is successfully closed	N/A
rejected	ChannelSelector is already closed	Already closed

open

Opens the ChannelSelector UI.

Example

```
oMyPlugin.command('ChannelSelector.open').done(function(e){
    // ChannelSelector opened successfully
}).fail(function(e){
    // ChannelSelector failed to open
});
```

Resolutions

Status	When	Returns
resolved	ChannelSelector Widget is successfully opened	N/A
rejected	ChannelSelector Widget is already open	'Already open'

configure

Modifies the ChannelSelector configuration.

Example

```
oMyPlugin.command('ChannelSelector.configure', {
    channels: [
        {
            enable: true,
            clickCommand: 'CallUs.open',
            readyEvent: 'CallUs.ready',
            displayName: 'Call Us',
            i18n: 'CallusTitle',
            icon: 'call-outgoing',
            html: ''
        }
    ]
}).done(function(e){
    // ChannelSelector configured successfully
}).fail(function(e){
    // ChannelSelector failed to configure
});
```

Options

Option	Type	Description
channels	array	Array containing each channel configuration object. The order of channels is displayed based on the order defined here.
channels[].enable	boolean	Enable/disable chat channel.
channels[].clickCommand	string	The CXBus command name for opening a particular widget when this channel is clicked.
channels[].readyEvent	string	Subscribes to this ready event published by a plugin and enables the channel when that plugin is ready.
channels[].displayName	string	A channel name to display in the

Option	Type	Description
		ChannelSelector Widget.
channels[].i18n	string	To support localization of channel display name, this takes a key parameter of the channelselector section in the language pack file. Overrides above displayName.
channels[].icon	string	Select from one of the Genesys Widgets icons by specifying icon css class name.
channels[].html	string	Overrides and replaces the icon section of a channel with the html (image tag) defined here.

Resolutions

Status	When	Returns
resolved	Configuration options are provided and set	N/A
rejected	No configuration options are provided	'Invalid configuration'

API events

Once you've registered your plugin on the bus, you can subscribe to and listen for published events. Here's how to use the global bus object to register a new plugin on the bus.

Important

The global bus object is a debugging tool. When implementing Widgets on your own site, do not use the global bus object to register your custom plugins. Instead, see Genesys Widgets Extensions for more information about extending Genesys Widgets.

```
var oMyPlugin = window._genesys.widgets.bus.registerPlugin('MyPlugin');
oMyPlugin.subscribe('ChannelSelector.ready', function(e){});
```

Name	Description	Data
ready	ChannelSelector plugin is initialized and ready to accept commands	N/A
opened	ChannelSelector Widget has appeared on screen	N/A

Name	Description	Data
closed	ChannelSelector Widget has been removed from the screen	N/A