Deployment Guide

Genesys Widgets 8.5

12/29/2021

Table of Contents

Genesys Widgets Deployment Guide

This guide provides the steps required to instrument your website with Genesys Widgets.

Audience

This document is for website developers who are in charge of the website code. You should have knowledge of HTML, JavaScript, and CSS.

Cookies

Genesys Widgets uses cookies to store non-sensitive data in the browser. These cookies are used to restore a chat session, to track the state of the UI, to store a user's decision, and more. The end-user's browser must allow cookies for Genesys Widgets to operate properly.

> **Important**
> No personally identifiable information (PII) is ever stored in cookies, local storage, or session storage by Genesys Widgets

Sub-Domains

Normally, cookies can not be transferred between sub-domains of a website unless they are configured to do so. Genesys Widgets automatically detects the domain of the host site and configures all cookies to be transferable between sub-domains. For example, you could start a chat on **www.testsite.com** and restore that chat session on **store.testsite.com**, **support.testsite.com**, or **portal.testsite.com**.

Cookie Support in Test Environments

Genesys Widgets uses special cookies that persist across sub-domains. This is a critical feature for plugins like WebChat that need to restore an active chat session while navigating around a website. The side effect of using this type of cookie is they won't work when using test environment domain names such as "localhost" or an IP address. You must use a fully-qualified domain name (FQDN) such as "localhost.com" or any other variant that can be identified as a domain name. Cookies will also fail to work if you run the test site as an HTML file path directly in the browser.

One workaround is to update your system's **hosts** file to create an FQDN alias for "localhost", your test environment's name, or an IP address.

Example

```
127.0.0.1       localhost
127.0.0.1       localhost.com
```

How Do I Deploy Genesys Widgets?

1. Embed Genesys Widgets into your website
2. Configure your Genesys Widgets
3. Setup Localization Options and Languages
4. Styling Genesys Widgets

How to embed the Genesys Widgets into your website

Unzip the Genesys Widgets package to your web server, and then locate the path to your copy of the **widgets.min.js** file.

Note: Genesys Widgets require cookies to function. Cookies are utilized to save UI states and maintain active sessions with an agent while users navigate the website.

you can choose from the following methods to include the Genesys Widgets package on your webpage:

Inline JavaScript Include

Add a script tag to include the Genesys Widgets package file (widgets.min.js)

```
<script id="genesys-widgets-script" src="http://www.yourhost.com/path/to/
widgets.min.js"></script>
```

Important
Your configuration options must be defined on the page before widgets.min.js is loaded. Failing to do so may result in errors.

Important
Genesys Widgets is designed to avoid interoperability issues with your web page as much as possible. All dependencies with third-party libraries are loaded privately, and the public Widgets API is exposed within the _genesys namespace only. (In particular, usage of jQuery's noConflict() is not required on the parent website). If you find any unreasonable interoperability issues, please contact Genesys Customer Care.

Inline Stylesheet/CSS Include

Add a link tag to include the Genesys Widgets stylesheet package file (widgets.min.css)

```
<link id="genesys-widgets-styles" href="http://www.yourhost.com/path/to/widgets.min.css"
type="text/css" rel="stylesheet"/>
```

Dynamically Generate the JavaScript Include

The following code snippet will generate a script and link include tag automatically and provides you the option of loading the files immediately or delay loading the files until after the parent page has finished loading its content

```
// Widgets Instrumentation Script

<script>
    (function(o){var f = function(){var d = o.location;o.aTags = o.aTags || [];
    for(var i=0;i<o.aTags.length;i++){var oTag = o.aTags[i];var fs = d.getElementsByTagName(oTag.type)[0], e;
    if(d.getElementById(oTag.id)) return; e = d.createElement(oTag.type); e.id = oTag.id;if(oTag.type == "script"){e.src = oTag.path;}
    else{e.type = 'text/css';e.rel = 'stylesheet';e.href = oTag.path;}if(fs){fs.parentNode.insertBefore(e,
fs);}else{d.head.appendChild(e);}
    }},ol = window.onload;if(o.onload){typeof window.onload != "function"?window.onload=f:window.onload=function(){ol();f();}}else
f();})({

    location: document,
    onload: false,
    aTags: [

        {
            type:"script",
            id:"genesys-widgets-script",
            path:"http://www.host.com/path/to/widgets.min.js"
        },
        {
            type:"link",
            id:"genesys-widgets-styles",
            path:"http://www.host.com/path/to/widgets.min.css"
        }
    ]
});
</script>
```

This script will dynamically generate a JavaScript include <script> tag+, CSS include <link> tag+ and place it into the page. By default the ID for the generated script and link tags are "genesys-widgets-script" and "genesys-widgets-styles". You may modify these ID's in the above script.

The following object is passed in with additional options:

```
{
    location: document,
    onload: false,
    aTags: [
        {
            type:"script",
            id:"genesys-widgets-script",
            path:"http://www.yourhost.com/path/to/widgets.min.js"
        },
        {
            type:"link",
            id:"genesys-widgets-styles",
            path:"http://www.yourhost.com/path/to/widgets.min.css"
        }
    ]
}
```

location

location is your page document scope that can be used to inject widgets script and link tags.

onload

onload is an optional flag that, when set to true, will delay loading the Widgets files until after the webpage has fully loaded (window.onload). By default this value is false and the Widgets files are loaded immediately.

path

path is the URL or path to the location of the Genesys Widgets JavaScript and CSS files that Genesys provides. You can host these files on your own web servers or on a Content Delivery Network (CDN).

type

type is the html tag type that will be added in to the page i.e. 'script' tag for JavaScript file and 'link' tag for CSS file.

Now you can configure the Genesys Widgets and the products and services associated with it.

> **Important**
> Your configuration options must be defined on the page before widgets.min.js is loaded. Failing to do so may result in errors.

Tested Browsers

The following is a list of all Genesys-tested browsers for both web and mobile.

Important

If you do not see your device/OS/browser combination listed below, please contact Genesys support. Help will be decided on a per-case basis.

Support for the device/OS/browser combinations listed below will only be available for as long as Genesys labs can properly reproduce the issue.

Please let Genesys know of any issues you encounter with any of our tested browsers.

Desktop Browsers

Browser	Release Version
Microsoft Edge 14+	8.5.008.08
Microsoft Internet Explorer 11	8.5.000.11
Google Chrome 47+	8.5.000.11
Firefox 43+	8.5.000.11
Safari 8+	8.5.000.11

Mobile Browsers

OS Family	Device	Operating System	Browser	Release Version	Known Limitations
Android	Galaxy S5 Mini	Android 4.4	Chrome 30	8.5.004.15	There is currently limited Unicode emoji support. At a minimum, the following emoji are supported on the Android Galaxy 5S Mini: • ☺ U+263A

OS Family	Device	Operating System	Browser	Release Version	Known Limitations
					• 👍 U+1F44D • ☹ U+2639
	Galaxy S5	Android 4.4	Chrome 49.0	8.5.004.15	n/a
	Galaxy Note 3	Android 4.3	Chrome 49.0	8.5.004.15	n/a
	Galaxy Tab 4	Android 4.4	Chrome 49.0	8.5.004.15	n/a
	Google Nexus 9	Android 5.1	Chrome 54.0	8.5.004.15	n/a
	Google Nexus 7	Android 6	Chrome 49.0	8.5.004.15	n/a
	Kindlle Fire HDX 7	Android 4.3	Android Browser 4	8.5.004.15	n/a
iOS	iPhone 5S	iOS 7.1	Safari 7	8.5.004.15	n/a
	iPhone 6	iOS 8.1	Safari 8	8.5.004.15	n/a
	iPhone 6 Plus	iOS 8.1	Safari 8	8.5.004.15	n/a
	iPad Mini 2	iOS 8.1	Safari 8	8.5.004.15	n/a
	iPad Air	iOS 8.3	Safari 8	8.5.004.15	n/a
	iPad Pro	iOS 9.3	Safari 9	8.5.004.15	n/a
	iPad Pro	iOS 10	Safari 10	8.5.004.15	n/a
Windows Phone	Lumia 630	Windows Phone 8.1	IE11	8.5.004.15	n/a
	Lumia 930	Windows Phone 8.1	IE11	8.5.004.15	n/a

Required Versions of Genesys Components

The following is a list of minimum versions required to achieve end-to-end functionality.

Component	Version
Chat Server	8.5.105.05
Genesys Co-browse Server	8.5.002.00
Genesys Mobile Services	8.5.106.14
Genesys Web Engagement Server	8.5.000.15
Genesys Knowledge Center Server	8.5.200.11

Additionally, for supporting Typing Preview:

Component	Version
Workspace Desktop Edition	8.5.108.11

Additionally, for supporting CometD:

Component	Version
Chat Server	8.5.108.+
Genesys Mobile Services	8.5.109.+

Configuring Genesys Widgets

Genesys Widgets supports multiple widgets including Webchat, Web Engagement, and Co-browse. You can configure all widgets and services in the same configuration object. When you add new Genesys products and services, you must update your Genesys Widgets configuration to enable those widgets.

After you deploy the Genesys Widgets on your website, configure the CX Widget by defining the **global window._genesys** JavaScript object.

To include the JavaScript script, you can choose one of the following options:

- Place the script inline on your website; or
- Place it in a separate JavaScript file, and then include the file on your page.

The following example is a basic view of the global Genesys Widgets configuration object:

```
<script>

        if(!window._genesys)window._genesys = {};
        if(!window._gt)window._gt = [];

        window._genesys.widgets = {

                main: {},
                webchat: {},
                cobrowse: {},
                gwe:{}
        };

</script>

// include widgets.min.js after defining your configuration options
```

The following example is a populated Widget configuration that includes configuration options for Webchat, SendMessage, Web Engagement (GWE), and Co-browse (GCB):

Important

Your configuration options must be defined on the page **before** widgets.min.js is loaded. Failing to do so may result in errors.

Tip

Note that Web Engagement and Co-browse is optional.

```
<script>

        if(!window._genesys)window._genesys = {};
        if(!window._gt)window._gt = [];

        window._genesys.widgets = {
                main: {

                        theme: "dark",
                        lang: "en",
                        i18n: "http://HOST:PORT/path/to/languages/file.json",
                        customStylesheetID: "<ANY_ID_NAME>",
                        plugins: [
                                "cx-webchat",
                                "cx-webchat-service",
                                "cx-cobrowse",
                                "cx-gwe",
                                "cx-send-message",
                                "cx-send-message-service",
                                 ],
                        mobileMode: 'auto',
                        mobileModeBreakpoint: 600
                        },
                webchat: {

                        apikey: "0123456789", // Used for Apigee service only
                        dataURL: "http://HOST:PORT/path/to/chat/service",
                        userData: {},
                        proactive: {

                                enabled: true,
                                idleTimer: 5,
                                cancelTimer: 30
                        },
                        chatButton: {

                                enabled: true,
                                template: false,
                                openDelay: 1000,
                                effectDuration: 300,
                                hideDuringInvite: true
                        }
                },
                cobrowse: {

                        src: "<COBROWSE_SERVER_URL>/cobrowse/js/gcb.min.js",
                        url: "<COBROWSE_SERVER_URL>/cobrowse"
                },
               gwe: {
                       httpEndpoint: "http://<GWE_SERVER_URL>:<PORT>",
                       httpsEndpoint: "https://<GWE_SERVER_URL>:<PORT>",
                       dslResource: "http://HOST:PORT/path/to/domain-model.xml"
               },
              sendmessage: {

                        SendMessageButton : {

                                enabled: true,
                                template: false,
                                openDelay: 1000,
                                effectDuration: 300
                        },
                        apikey: "0123456789", // Used for Apigee service only
                        dataURL: "http://HOST:PORT/path/to/sendmessage/service"
                },
        };

        (function(o){
        var f = function(){
                var d = o.location;
                o.aTags = o.aTags || [];
                for(var i=0;i<o.aTags.length;i++){
                        var oTag = o.aTags[i];
                        var fs = d.getElementsByTagName(oTag.type)[0], e;
                        if(d.getElementById(oTag.id)) return; e = d.createElement(oTag.type); e.id = oTag.id;
                        if(oTag.type == "script"){e.src = oTag.path;}
                        else{e.type = 'text/css';e.rel = 'stylesheet';e.href = oTag.path;}
                        if(fs){fs.parentNode.insertBefore(e, fs);}else{d.head.appendChild(e);}
                }},ol = window.onload;
                if(o.onload){typeof window.onload != "function"?window.onload=f:window.onload=function(){ol();f();}}else f();
})({location: document,
     onload: false,
     aTags: [{type:"script", id:"genesys-cx-widget-script", path:"http://www.yourhost.com/path/to/widgets.min.js"},
                 {type:"link", id:"genesys-cx-widget-styles", path:"http://www.yourhost.com/path/to/widgets.min.css"}]});
```

```
</script>
```

Main Configuration

Genesys Widgets is a hub for multiple Genesys products and services. Some configuration options are set globally and therefore apply to all products and services running on the CX Widget platform. In the main application configuration you can configure options such as visual theme, language, and mobile support.

For detailed information on configuration options, see App Configuration Options.

Widget Configuration Options

- WebChat Configuration
- WebChatService Configuration
- SendMessage Configuration
- SendMessageService Configuration
- CoBrowse Configuration
- GWE Configuration
- Channel Selector Configuration

For a complete list of Widget configuration options, please see Genesys Widgets Reference.

Launcher

Launcher is a sample page which shows how widgets display on any host website.

Using a sample page has the following advantages:

- Viewing Genesys Widgets with your own configuration.
- Copying Configuration Script. Using the details you entered on the form, the configuration script is generated in the **Need Configuration Script** box. You can copy this script and use it on your website to launch widgets.

How to use Launcher

Webchat
Click the checkbox next to **Webchat** to enable it. Enter the mandatory field URL (marked with an asterisk) and click the **Launch** button. Other values are optional and self explanatory.

Co-browse
Click the checkbox next to the **Co-browse** heading to enable it. Enter required fields (marked with an

asterisk) and click the **Launch** button.

GWE
Click the checkbox next to the **GWE** heading to enable it. Enter required fields (marked with an asterisk) and click the **Launch** button.

Send Message
Click the checkbox next to the **Send Message** heading to enable it. Enter required fields (marked with an asterisk) and click the **Launch** button.

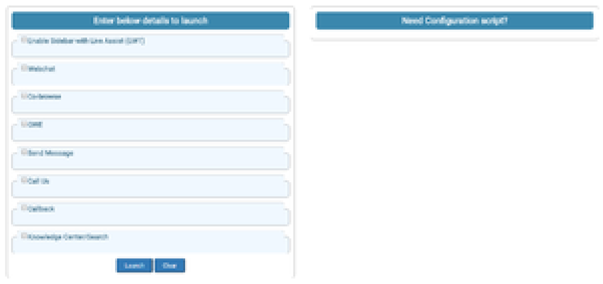
CallUs
Select the check box next to “Call Us” and provide the configuration data. Here, the **Edit/Use Sample Config** option is also provided and you can use this to prefill with the sample configuration and edit it for your own details. Ideally, call us is shown in the Live Assist widget or it can also be launched with bus command "CallUs.open".

ChannelSelector
Select the checkbox next to “Enable Sidebar with Live Assist (EWT)” to include ChannelSelector plugin. Enter the Stats URL field followed by Chat and CallUs Vitual Queue names to fetching Stats service Estimated Wait Time details. For the Live Assist widget to show Chat, SendMessage and CallUs channels, please make sure you select those plugins as well.

Callback
Select the checkbox next to "Callback" to include Callback and Calendar plugins. Enter the callback service provider URL field and other details as required, ensure "Enable Sidebar with Live Assist" is selected and click the "Launch" button.

Knowledge Center/Search
Select this checkbox to include Knowledge Center plugin. Ensure all the details are entered along with sidebar plugin enabled and click "Launch" button.

Important
Sidebar is not an officially released plugin; it is being used here here for the purpose of the launcher tool. Ideally, it should be launched with bus command "ChannelSelector.open".


Enter below details to launch
Enable Sidebar with Live Assist (EWT)
Webchat
Co-browse
GWE
Send Message
Call Us
Callback
Knowledge Center/Search
Launch
Clear
Need Configuration script?

Launcher tool (click to enlarge)

Localization

Genesys Widgets allow for localization of user messages and prompts. First, you must create and host a Language Pack that Genesys Widgets can access and use. The Language Pack is a file written in JSON format. Specify your Language Pack file by using Genesys Widgets configuration options, which you can configure in the **window_genesys.widgets.main** section.

Example:

```
<script>

    if(!window._genesys)window._genesys = {};
    if(!window._gt)window._gt = [];

    window._genesys.widgets = {

        main: {

            theme: "dark",
            lang: "en",

            // Enter a URL that points to
            i18n: "http://HOST:PORT/path/to/lanaguages/file.json"

            // OR define the JSON object inline
            i18n: {

                "en": {

                    "webchat": {

                        "ChatStarted": "Chat Started",
                        "ChatEnded": "Chat Ended",
                        ...
                    },

                    "sendmessage": {

                        "EmailFormFirstname": "First Name",
                        "EmailFormLastname": "Last Name",
                         ...
                     }
                }
            }
        }
    };
</script>
```

Configuration Options

main.lang
Type: string
Default: "en"
Requirement: Optional

Description: A language code to specify which language to display in the Widgets. Language codes are set by the customer.

main.i18n (external file)
Type: string
Default: built-in English words and phrases
Requirement: Optional
Description: A URL that the Widgets use to fetch the Language Pack file upon startup.Can be partial or complete. Unspecified strings will use default values.

main.i18n (inline object)
Type: object
Default: built-in English words and phrases
Requirement: Optional
Description: An inline JSON object. Can be partial or complete. Unspecified strings will use default values.

Language Pack JSON Format

The Language Pack is written in JSON format.

```
// Root
{
    // Language Code
    "en": {

        // Widget name
        "webchat": {

            // Localized strings
            "ChatStarted": "Chat Started",
            "ChatEnded": "Chat Ended",
            "ChatFailed": "There was a problem starting the chat session. Please Retry.",

            // Customer Defined Strings - Match & Replace messages received from chat server
            "SYS0001": "An Agent will be with you shortly"
        },

        "sendmessage": {

              // Localized strings
             "SendMessageButton": "Send Message",
             "EmailFormFirstname": "First Name",
             "EmailFormLastname": "Last Name",

             //Errors
             "ErrorServerNotAvailable": "Unable to reach server. Please try again.",
             "ErrorAttachfileSizeMax": "Total size of attachments exceeds limit: "
    }
}
```

Localization Namespaces

Plugin	Namespace
WebChat	webchat
SendMessage	sendmessage
CallUs	callus
ChannelSelector	channelselector
CallBack	callback
KnowledgeCenter	knowledgecenter
Offers	offers
Calendar	calendar

Language Codes

To allow flexibility in the way that your website currently handles multiple languages and language codes, there are no rules for language codes other than that they must be strings. You can use any language code system. The language code that you set in **window._genesys.widgets.main.lang** must correlate to a language code in the Language Pack File.

Plugin Localization Options

- WebChat Localization
- WebChatService Localization
- SendMessage Localization
- SendMessageService Localization
- CoBrowse Localization
- GWE Localization
- Callback Localization
- CallUS Localization
- ChatDeflection Localization
- Search Localization

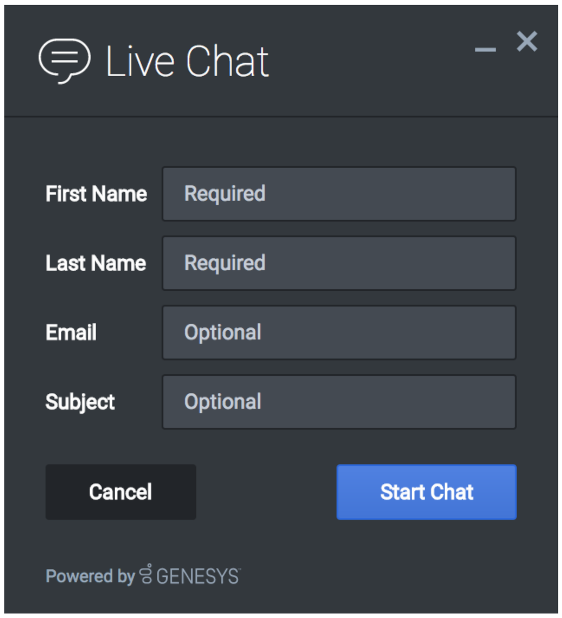
Styling the Widgets

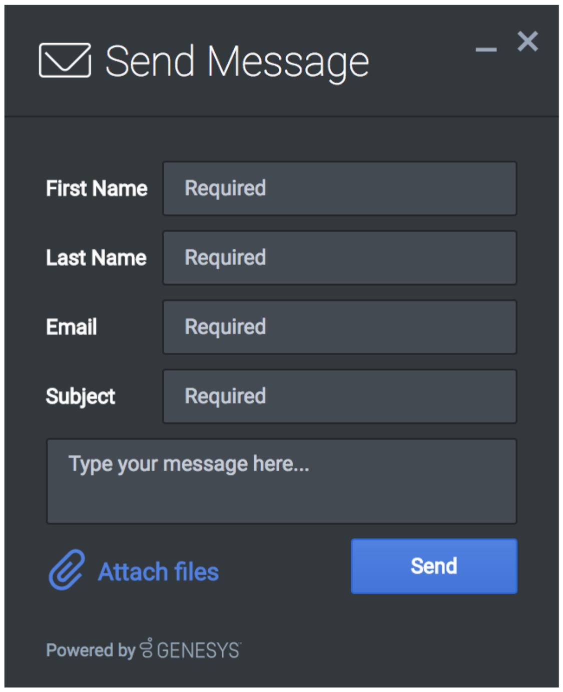
Themes

You can change the appearance of Genesys Widgets using *themes*. Themes allow you to change colors and fonts for all widgets.

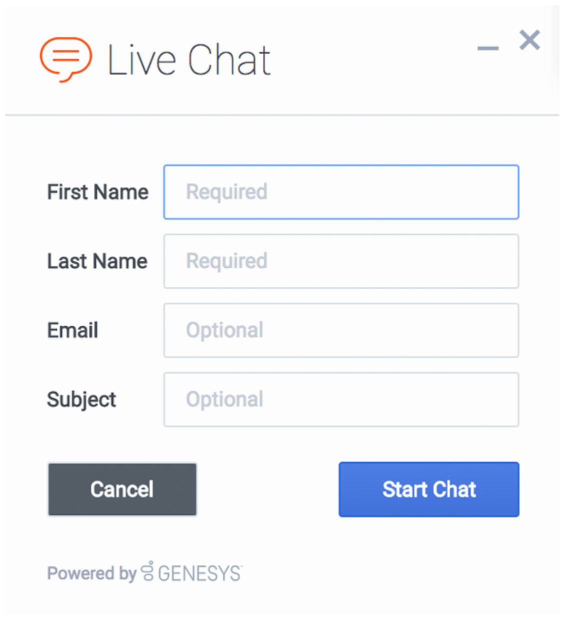
Genesys Widgets includes two built-in themes, "dark" and "light". The "dark" theme is active by default.

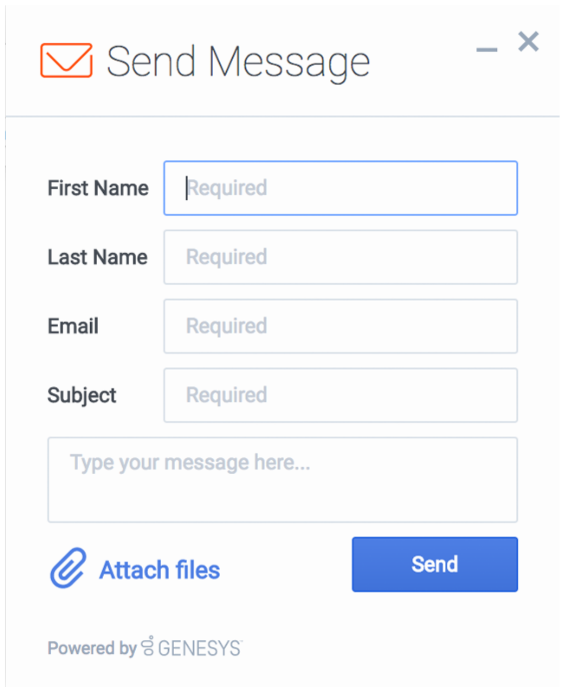
Dark Theme





Light Theme





How do I set the active theme?

There are two methods for setting the active theme:

Configuration

```
window._genesys.widgets.main.theme = "light"; // or "dark"
```

Widget Bus Command

```
window._genesys.widgets.bus.command("App.setTheme", {theme: "light"}); // or "dark"
```

How do I create my own themes?

Theme Templates

Genesys Widgets uses special LESS files called "Theme Templates" to define themes. Using this Theme Template, you can modify the color palette and add new styles. Everything is laid out clearly in the template file.

LESS syntax is used because we can define local variables that allow us to create a clear color palette. The LESS file color palette consists of no less than 28 separate color variables. These are grouped by their usage:

- Background Colors
- Text Colors
- Icon Colors
- Border Colors
- Outline Colors

At a bare minimum, you can create a new style by simply changing the color values in the color palette. You may add or remove colors from this palette as you see fit.

Color Palette Example

```
/* Color Palette */

@bg_color_1:                 #33383D; // Main Background Color
@bg_color_2:                 #444A52; // Form Inputs
@bg_color_3:                 #222529; // Button default
@bg_color_4:                #5081E1; // Button primary gradient 1
@bg_color_5:                #4375D6; // Button primary gradient 2
@bg_color_6:                #CCCCCC; // Button disabled / scrollbar color

@txt_color_1:                 #FDFDFD; // Main text color
@txt_color_2:                 #98A7B8; // footer text
@txt_color_3:                 #FDFDFD; // Button default & primary / autocomplete text hover
color
@txt_color_4:                 #5081E1; // Hyperlink color
@txt_color_5:                #C5CCD6; // Placeholder color
@txt_color_6:                #EA4F6B; // Alert/error color

@icon_color_1:                #FDFDFD; // Base icon color
@icon_color_2:                #8C8C8C; // Secondary icon color (multitone only)
@icon_color_3:                #000000; // Icon shadow color (multitone only)
@icon_color_4:                #000000; // Icon secondary shadow color (multitone only)
@icon_color_5:                #98A7B8; // Window control icon color
@icon_color_6:                #98A7B8; // Form input icon overlay color (e.g. "clear" icon)
@icon_color_7:                 #5081E1; // Interactive icon color 1 (attach files, delete
file, etc)
@icon_color_8:                 #4AC764; // Positive Color (confirmation, availability,
usually green)
@icon_color_9:                 #EA4F6B; // Negative Color (error, exception, usually red)
@icon_color_10:         #F8A740; // Warning Color (warning, pending, offline, usually yellow
or orange)

@border_color_1:         #222529; // Main border color
@border_color_2:        #2E69DB; // Button primary
@border_color_3:        transparent; // Button default
@border_color_4:        transparent; // Button disabled
@border_color_5:        #EA4F6B; // Alert/error color

@outline_color_1:         #75A8FF; // Form input focus outline / autocomplete hover
background color
```

Example Theme Template Files

Note: Clicking the example template files automatically downloads them to your computer.

theme-template-dark.less.rar
theme-template-light.less.rar

Theme Templates are LESS files, which must be converted to CSS before being used on a website. Use a website or tool to convert them when you're ready to test and implement them on your site.

By default, the Theme Template will override styles for all Genesys Widgets generally, but you may add more specific changes that affect only a specific widget. More information on this is provided later on this page.

Naming Your Theme

In the "dark" theme template file, the first class selector is defined as:

.cx-widget.cx-theme-dark

.cx-widget is the base class for all Genesys Widget UI. The outermost container of every widget or standalone UI element has this class and is used to identify UI elements that belong to Genesys Widgets.

.cx-theme-dark is the class name created for the "dark" theme. Themes are applied by searching for all elements with the **.cx-widget** class and appending the theme's classname to it. Thus, the combined class selector indicates styles that will be applied only when your custom theme is active in the configuration object.

You may name your theme classname anything you wish. There are no restrictions or limitations.

In a later step, you will register this theme classname in your configuration.

Customization Guidelines

When creating your own themes, they are restricted to the following CSS properties:

- color
- background
- font-family
- font-style
- border-color
- border-style
- and other non-structural properties

Warning
Exercise caution when you make changes to the CSS to make sure structure and functionality are retained.

Warning

Avoid setting CSS properties that change height, width, thickness, size, visibility, or other properties that change the structure of widgets. These properties are not supported and changing them could break widget stability and usability.

Important

By default, the Widgets CSS refers to the Roboto font, available at https://fonts.google.com/

How do I register my themes with Genesys Widgets?

You can register themes in the Genesys Widgets configuration.

```
window._genesys.widgets.main.themes = {

    "blue": "cx-theme-blue"
};
```

The name:value pair used here consists of a key ("blue") and the theme's CSS classname ("cx-theme-blue").

You can add as many themes to this list as you need and select the active theme using one of the key values.

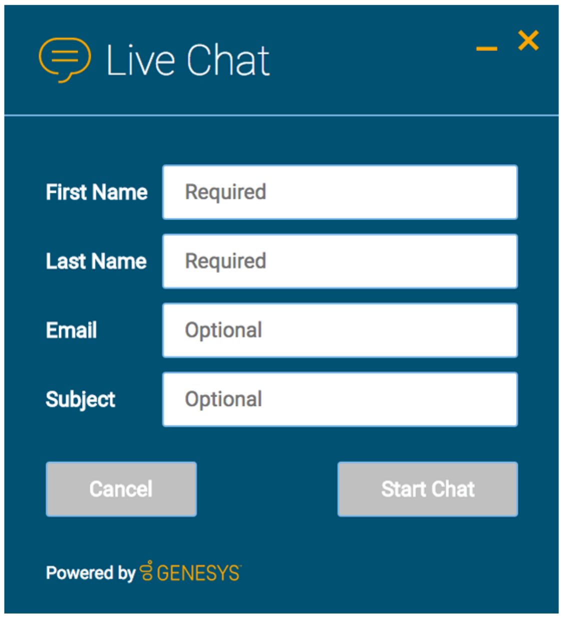
```
window._genesys.widgets.main.theme = "blue";

// OR

window._genesys.widgets.bus.command("App.setTheme", {theme: "blue"});
```

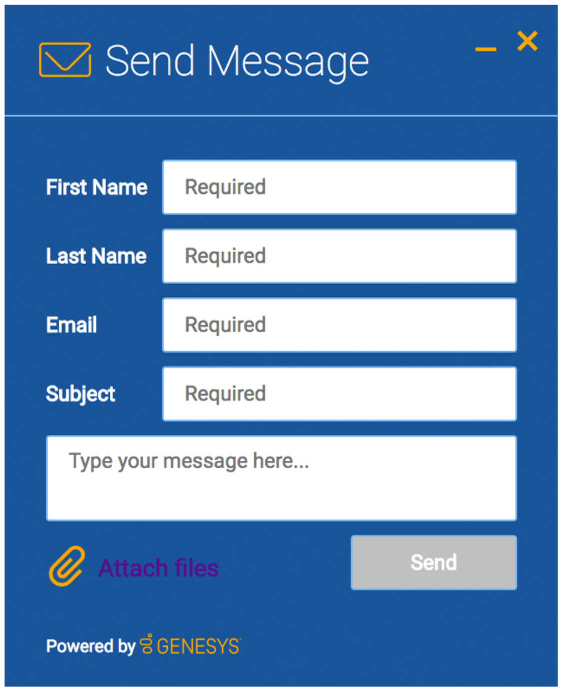
How do I change styles for a specific widget?

It is possible to specify specific widgets and even specific elements within a widget by appending the widget's CSS classname to the theme classname.

In the following example, we extend the "cx-theme-blue" class with a widget-specific entry that makes the WebChat widget's background color a darker shade.

```
.cx-widget.cx-theme-blue, .cx-widget .cx-container{

    color: #FDFDFD;
    background: #1e5799;
}
.cx-widget.cx-theme-blue *{

    border-color: #7DB9E8;
}
.cx-widget.cx-theme-blue.cx-webchat, .cx-widget.cx-theme-blue .cx-webchat{

    background: #225897;
}
```





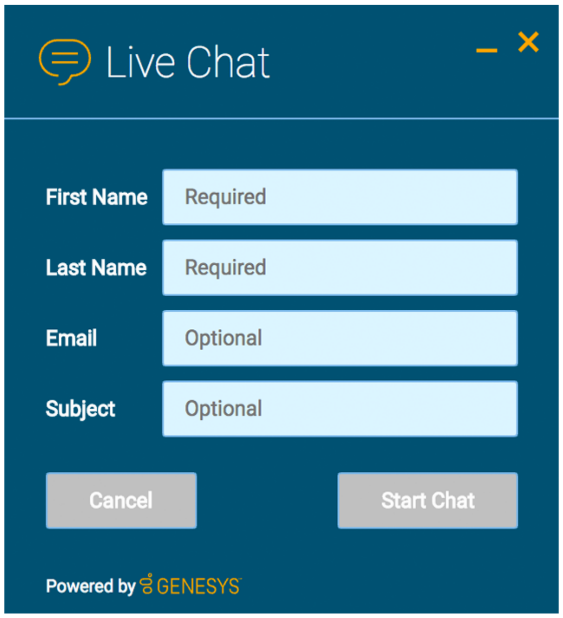
Important

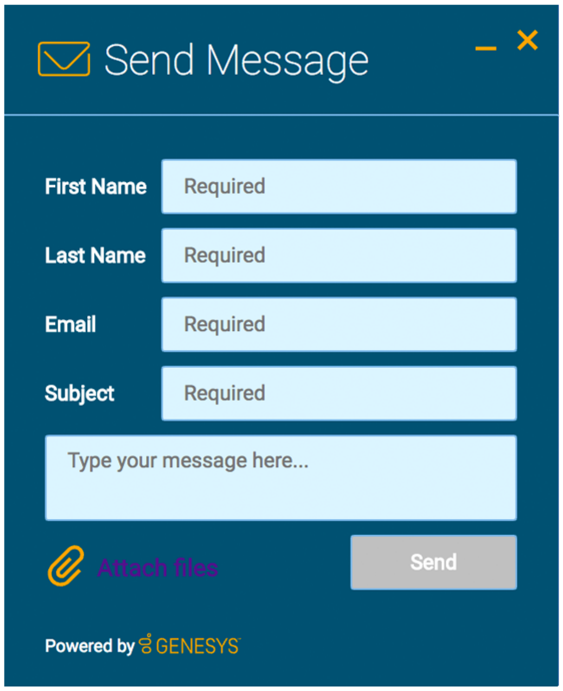
Notice the dual CSS selector used when specifying a widget. This is required to make sure your styles always apply properly.

Widget-Specific and Element-Specific

In the next example, we further extend the "cx-theme-blue" class with a widget and element-specific entry that changes the background color of the input fields within the WebChat widget to a light shade of blue.

```
.cx-widget.cx-theme-blue, .cx-widget .cx-container{

    color: #FDFDFD;
    background: #1e5799;
}

.cx-widget.cx-theme-blue *{

    border-color: #7DB9E8;
}

.cx-widget.cx-theme-blue.cx-webchat, .cx-widget.cx-theme-blue .cx-webchat{

    background: #225897; // Darker Shade
}

.cx-widget.cx-theme-blue.cx-webchat .form input, .cx-widget.cx-theme-blue .cx-webchat .form
input{

    background: #DCF5FF; // Lighter Shade
}
```





How do I change the layout and structure of widgets?

Genesys Widgets only support customizing a limited set of styles through themes. If you wish to use an alternate layout of your own design, you can disable the widget you want to replace and utilize the provided service plugins to build your own.

Choosing Which Plugins to Load

Refer to the 'plugins' configuration option here: App Configuration

Service Plugins

Service plugins provide a high-level API for quickly integrating a UI with back-end services. Each widget is matched with a corresponding service plugin. This separation allows for advanced integrations.

- WebChatService
- SendMessageService
- CallbackService
- KnowledgeCenterService

Warning

Changing the layout of official Genesys Widgets is not supported. We cannot guarantee that your changes will remain when upgrading to a newer version of Genesys Widgets.

How do I change fonts?

By default, Genesys Widgets downloads and uses the Google font 'Roboto'. You can change the font used in Genesys Widgets by using the following CSS:

```
.cx-widget{ font-family: name-of-font-here; }
```

Choose whichever font you wish to use and it will then apply throughout Genesys Widgets.

Disabling Google Font Download

If you would like to prevent the Roboto font file from being downloaded at startup, you can disable the download by changing the configuration option `main.downloadGoogleFont` to `false`:

```
_genesys.widgets.main.downloadGoogleFont = false;
```

If set to "**false**", the Google font 'Roboto' will not be downloaded. If set to "**true**", the Google font 'Roboto' will be downloaded. The default value is "**true**".

Important

Use this configuration option if you have security concerns regarding including fonts from 3rd party sources, to optimize your page load time, or if you already include Roboto on your website.

Icons

Genesys Widgets utilize icons in SVG format. Using SVG allows icons to be themed with different color fills and other SVG CSS properties. Using SVG icons also provides the highest rendering quality on all devices, no matter the zoom level, resolution, or DPI of the screen. Our icons scale to fit in any container, can be used inline or as blocks, and can be animated in orientation, colors, and other styles.

Our icons are available to our customers to use in their own custom extensions. This allows custom extensions to retain the look and feel of our prebuilt widgets. Some examples might be:

- Creating a custom launcher button for chat, using the chat icon
- Creating a custom widget with your choice of icon in the title bar
- Using our icons inline, inside text copy to refer to widgets by icon

Widgets includes two different sets of icons. The Multi-tone icon set utilizes multiple layers and colors per icon. The Outline icon set offers a minimalist approach to both design and color. You may use any of the icons for your own purpose.

Currently, icons cannot be customized or replaced.

How to use icons

Automatic HTML Injection

By applying the CSS class "cx-icon" and the attribute "data-icon" to an element, you can specify which icons you want and where.

Example:

<div id="your-element" class="cx-icon" data-icon="chat"> ...SVG icon will be inserted here </div>

You then pass your element as a jquery wrapped set or HTML string into the `CXCommon.populateAllPlaceholders( $("#your-element") )` function, Widgets will insert the appropriate SVG icons into each specified element and return the HTML to you.

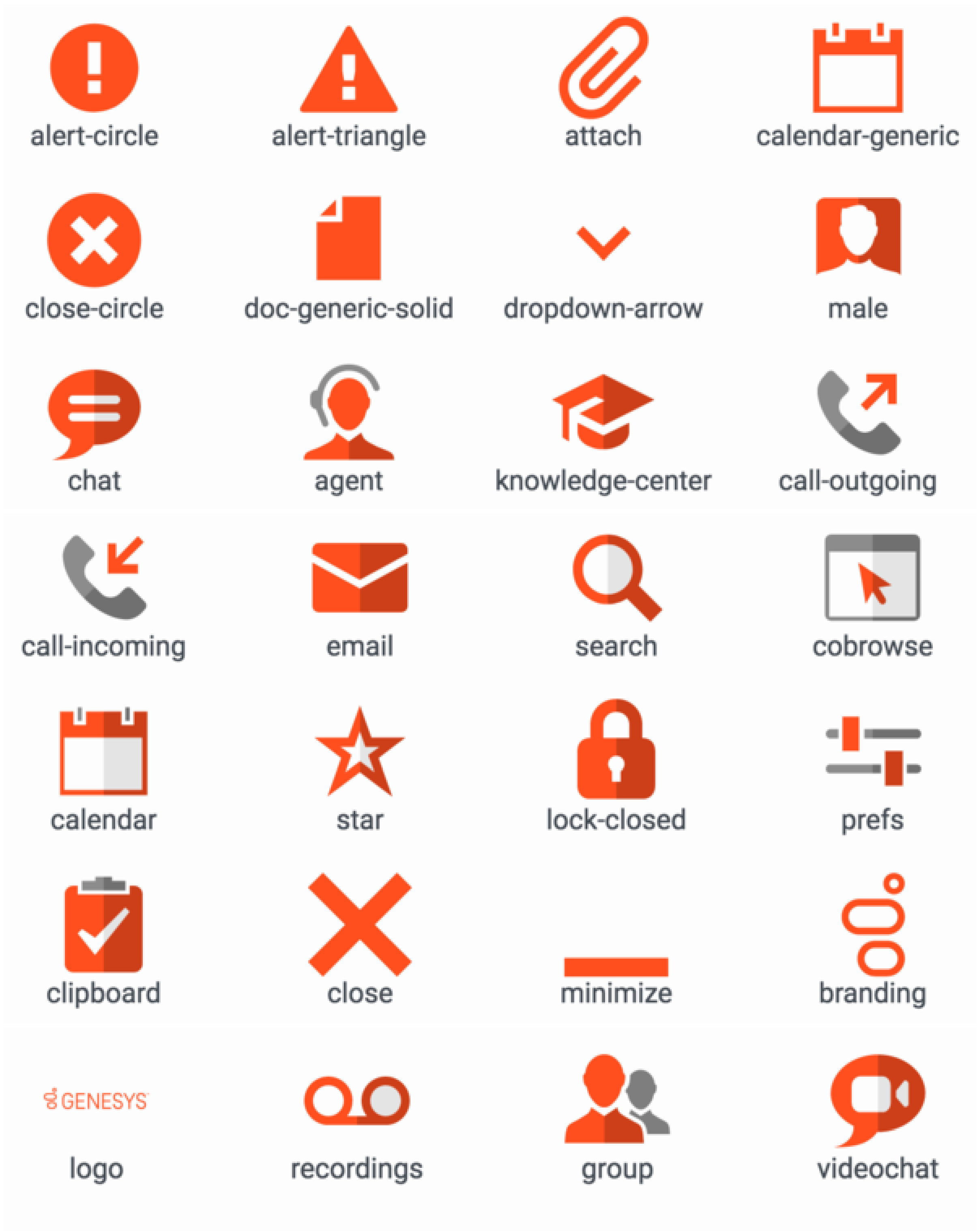
Fetching SVG Icon Markup

You can also fetch the markup for each SVG icon manually.

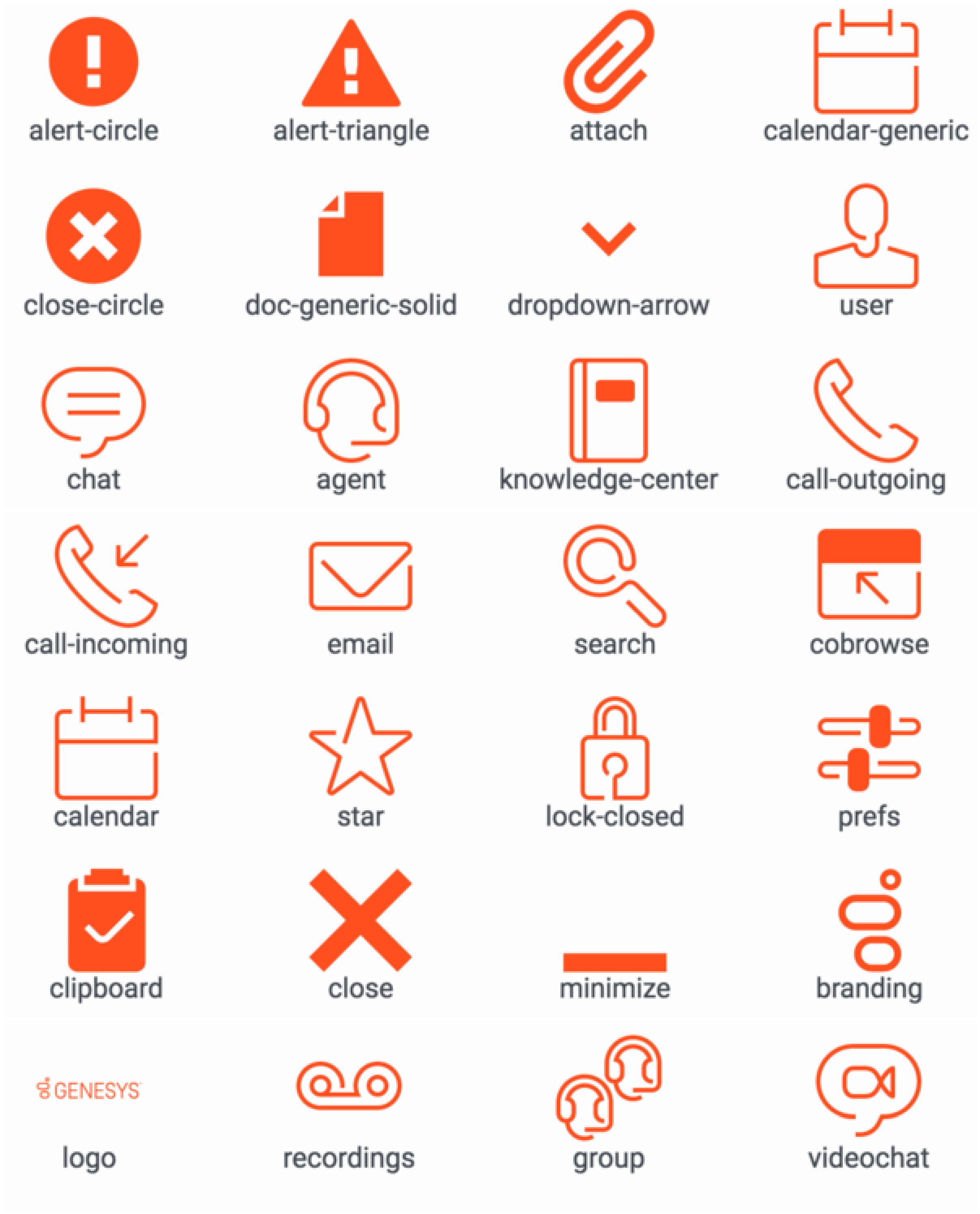
Example:

$("#your-element").append(CXCommon.Generate.Icon("chat"));

Multi-tone


alert-circle
alert-triangle
attach
calendar-generic
close-circle
doc-generic-solid
dropdown-arrow
male
chat
agent
knowledge-center
call-outgoing
call-incoming
email
search
cobrowse
calendar
star
lock-closed
prefs
clipboard
close
minimize
branding
GENESYS
logo
recordings
group
videochat

Outline


alert-circle
alert-triangle
attach
calendar-generic
close-circle
doc-generic-solid
dropdown-arrow
user
chat
agent
knowledge-center
call-outgoing
call-incoming
email
search
cobrowse
calendar
star
lock-closed
prefs
clipboard
close
minimize
branding
GENESYS
logo
recordings
group
videochat