Deployment Guide

Genesys Widgets Deployment Guide

5/6/2025

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.