



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 Reference

API Commands

Contents

- 1 API Commands
 - 1.1 setTheme
 - 1.2 getTheme
 - 1.3 reTheme
 - 1.4 themeDemo
 - 1.5 setLanguage
 - 1.6 closeAll
 - 1.7 updateAJAXHeader
 - 1.8 removeAJAXHeader
 - 1.9 registerExtension
 - 1.10 registerAutoLoad
 - 1.11 deregisterAutoLoad

API Commands

Once you've registered your own plugin on the bus, you can call commands on other registered plugins. Below we'll quickly register a new plugin on the bus using the global bus object.

Important

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

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

setTheme

Sets the theme for Genesys Widgets from the list of registered themes. Default themes are 'light' and 'dark'. You can register as many new themes as you need.

Example

```
oMyPlugin.command('App.setTheme', {theme: 'light'}).done(function(e){  
    // App set theme successfully  
}).fail(function(e){  
    // App failed to set theme  
});
```

Options

Option	Type	Description
theme	string	Name of the theme you want to use. This name is specified in <code>window._genesys.main.themes</code> . Default themes are 'light' and 'dark'.

Resolutions

Status	When	Returns
resolved	Theme exists and is successfully changed	The name of the theme that was chosen. For example 'light'.
rejected	Theme does not exist	'Invalid theme specified'

getTheme

Get the CSS classname for the currently selected theme.

Example

```
oMyPlugin.command('App.getTheme').done(function(e){  
    // App got theme successfully  
    // e == CSS classname for current theme  
}).fail(function(e){  
    // App failed to get theme  
});
```

Resolutions

Status	When	Returns
resolved	Always	CSS classname for the currently selected theme. For example 'cx-theme-light'.
rejected	Never	n/a

reTheme

Accepts an HTML reference (either string or jQuery wrapped set) and applies the proper CSS Theme Classname to that HTML and returns it back. When widgets receive the 'theme' event from App, they pass-in their UI containers into App.reTheme to have the old theme classname stripped and new classname applied.

Example

```
oMyPlugin.command('App.reTheme', {html: '<div>Test Theme</div>'}).done(function(e){  
    // App set theme successfully  
}).fail(function(e){  
    // App failed to set theme  
});
```

Options

Option	Type	Description
html	string or jQuery Wrapped Set	HTML string or jQuery Wrapped Set you want to have modified

Resolutions

Status	When	Returns
resolved	When HTML is provided and theme is updated	HTML that was passed-in and modified
rejected	When no HTML is provided	'No HTML provided by [plugin name]'

themeDemo

Start an automated demo of each theme. All registered themes will be applied with a default delay between themes of 2 seconds. You can override this delay. This command is useful for comparing themes or testing themes with official or custom widgets.

Example

```
oMyPlugin.command('App.themeDemo', {delay: 1000}).done(function(e){  
    // App demo successfully started  
}).fail(function(e){  
    // App failed to start demo  
});
```

Options

Option	Type	Description
delay	number	Number of milliseconds between theme changes. Default value is 2000 milliseconds.

Resolutions

Status	When	Returns
resolved	Always	n/a
rejected	Never	n/a

setLanguage

Changes the language

Important

Internal use only.

Example

```
oMyPlugin.command('App.setLanguage', {lang: 'eng'}).done(function(e){
    // App set language successfully started
}).fail(function(e){
    // App failed to set language
});
```

Options

Option	Type	Description
lang	string	Change the language of Genesys Widgets. Switches all strings in Widgets to selected language.

Resolutions

Status	When	Returns
resolved	When language successfully changed	n/a
rejected	When no language code is provided	No language code provided
rejected	When no matching language code is specified in your language pack	No matching language code found in language pack

closeAll

Publishes the 'App.closeAll' event that requests all widgets to close.

Example

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

Resolutions

Status	When	Returns
resolved	Always	n/a
rejected	Never	n/a

updateAJAXHeader

Introduced: 9.0.002.06

Updates the Authorization header.

Example

```
_genesys.widgets.bus.command('App.updateAJAXHeader', {header:
    {'Authorization': 'value'}}
});
```

Resolutions

Status	When	Returns
resolved	When header is updated	n/a
rejected	Never	No request header found

removeAJAXHeader

Introduced: 9.0.002.06

Removes the set Authorization header.

Example

```
_genesys.widgets.bus.command('App.removeAJAXHeader');
```

Resolutions

Status	When	Returns
resolved	Always	n/a

registerExtension

Introduced: 9.0.002.06

Allows you to register and initialize new extensions at runtime instead of predefining extensions before Genesys Widgets starts up.

Options

Option	Type	Description
undefined	function	Your extension function. Receives the following arguments: \$ (jQuery), CXBus, Common

Resolutions

Status	When	Returns
resolved	Valid 'extension' object provided	n/a
rejected	Invalid 'extension' option provided	n/a

registerAutoLoad

(For use with lazy-loading only) Allows you to register a plugin into the **preload plugins array** so that it can be pre-loaded at the startup rather than lazy loading later. This can be useful when there is an active session maintained by your Widget and you would like to show it immediately at startup during page refresh or navigating across pages.

Note: This command is intended for use when running widgets in lazy-loading mode. You may also use this to register and pre-load your own custom-made plugins.

Options

Option	Type	Description
name	string	The name of the plugin that needs to be registered for auto loading.

Resolutions

Status	When	Returns
resolved	When a plugin is added into the preload list	n/a

Status	When	Returns
rejected	Never	n/a

deregisterAutoLoad

(For use with lazy-loading only) Allows you to de-register a plugin from the **preload plugins array** so that it will not be pre-loaded at startup. This can be useful when there is no more active session maintained by your Widget and you don't want to show it on the screen immediately at startup.

Note: This command is intended for use when running widgets in lazy-loading mode. You may also use this to de-register your own custom-made plugins.

Options

Option	Type	Description
name	string	The name of the plugin that needs to be de-registered from auto loading.

Resolutions

Status	When	Returns
resolved	When a plugin is removed from the preload list	n/a
rejected	Never	n/a