3.11 Desktop JavaScript API Specification

Bright Pattern Documentation

Generated: 11/01/2024 12:35 am

Table of Contents

Purpose

ServicePattern Desktop JavaScript API Specification describes the Desktop JavaScript API that provides access to a number of functions of the ServicePattern Agent Desktop application from the web pages or IFrames loaded into this application from other domains.

For general information about the ServicePattern Agent Desktop application, see *ServicePattern Agent Guide*. For information about loading web pages into the Agent Desktop from scenarios, see description of the Web Screen Pop block in the *ServicePattern Scenario Builder Reference Guide*.

Audience

This guide is intended for the IT personnel responsible for the data infrastructure of the ServicePattern-based contact centers. Readers of this guide are expected to have expertise in web application development as well as solid understanding of contact center operations.

General Information

The API supports a number of desktop telephony functions including call initiation, voice and screen recording, interaction completion, and setting of interaction dispositions and notes.

These functions are available via a global object called *window.bpspat.api*, for example:

window.bpspat.api.dialNumber(“1234567”);

The object is created when the script file is included in the web page loaded into the Agent Desktop application as follows:

<script type="text/javascript" src="[agent-desktop-web-server]/app/libs/servicepatternapi-dev.js"></script>

For example:

<script type="text/javascript" src="https://barco.brightpattern.com/app/libs/servicepatternapi-dev.js"></script>

Dial Number

Syntax	*dialNumber(number);*
Description	Makes a call to the specified number. For more information, see sections How to Make an Internal Call and How to Make an Outbound Call of the *ServicePattern Agent Guide*.
Parameters	*number* – string containing the number to be dialed; required

Select Service

Syntax	*selectService(name);*
Description	Selects the service that will be associated with subsequent call attempts (until another service is selected using this method or via Agent Desktop). For more information, see sections How to Make an Internal Call and How to Make an Outbound Call of the *ServicePattern Agent Guide*.
Parameters	*name* – string containing the name of the selected service; optional; if omitted or empty, subsequent call attempts will not be associated with any service

Single-Step Transfer

Syntax	*singleStepTransfer(number);*
Description	Initiates a single-step (blind) transfer of the current call to the specified number.
Parameters	*number* – string containing the number that the call is to be transferred to; required

Single-Step Conference

Syntax	*singleStepConference(number);*
Description	Initiates a single-step conference with the current party on the call and the party at the specified number.
Parameters	*number* – string containing the number that the call is to be conferenced with; required

Terminate Interaction

Syntax	*terminateInteraction();*
Description	Unlike Complete Interaction, this method only releases the communication channel of the current interaction. I.e., if after-call work (ACW) is configured for the corresponding service, interaction processing will continue until ACW is completed.

Set Notes

Syntax	*setNotes(notes);*
Description	Sets the free-form notes for the current interaction to the specified string.
Parameters	*notes* – string containing the interaction notes

Set Disposition

Syntax	*setDisposition(dispositionCode);*
Description	Sets the disposition for the current interaction to the value corresponding to the alphanumeric code specified for this disposition.
Parameters	*dispositionCode* – string containing the numeric code of the desired disposition

Complete Interaction

Syntax	*completeInteraction();*
Description	Completes the current interaction. If the interaction requires a disposition in order to be completed, the Agent Desktop will show a warning message.

Complete Interaction with Disposition and Notes

Syntax	*completeInteractionWithDisp(dispositionCode, notes);*
Description	Completes the current interaction and sets its disposition and notes to the specified values (i.e., combines the actions of the Set Notes, Set Disposition, and Complete Interaction methods).
Parameters	*dispositionCode* – string containing the alphanumeric code of the desired disposition *notes* – string containing the interaction notes

Get Call Recording Status

Syntax	*getCallRecordingStatus(callback);*
Description	Requests the current status of voice recording of the current call.
Parameters	*callback* – function that will be called upon completion of the method

Start Call Recording

Syntax	*startCallRecording(callback);*
Description	Starts voice recording for the current call. For more information, see section How to Record a Call of the *ServicePattern Agent Guide.*

Parameters	*callback* – function that will be called upon completion of the method; see *Callback*
Note	Prior to version 3.8, this method was called *startCurrentCallRecording();* backward compatibility is preserved

Stop Call Recording

Syntax	*stopCallRecording(callback);*
Description	Stops voice recording for the current call.
Parameters	*callback* – function that will be called upon completion of the method; see *Callback*
Note	Prior to version 3.8, this method was called *stopCurrentCallRecording();* backward compatibility is preserved

Mute Call Recording

Syntax	*muteCallRecording(callback);*
Description	Mutes voice recording for the current call. Unlike Stop Call Recording, this method will continue voice recording for the current call, but any voice signal will be replaced with silence.
Parameters	*callback* – function that will be called upon completion of the method

Unmute Call Recording

Syntax	*unmuteCallRecording(callback);*
Description	Unmutes the previously muted voice recording for the current call.
Parameters	*callback* – function that will be called upon completion of the method

Get Screen Recording Status

Syntax	*getScreenRecordingStatus(callback);*
Description	Requests the current status of screen recording of the user session.
Parameters	*callback* – function that will be called upon completion of the method

Stop Screen Recording

Syntax	*stopScreenRecording(callback);*
Description	Stops screen recording of the user session.
Parameters	*callback* – function that will be called upon completion of the method

Pause Screen Recording

Syntax	*pauseScreenRecording(callback);*
Description	Pauses screen recording of the user session. For the period when screen recording is paused, the recording will contain a static snapshot of the desktop at the moment when pause was applied.
Parameters	*callback* – function that will be called upon completion of the method

Resume Screen Recording

Syntax	*resumeScreenRecording(callback);*
Description	Resumes the previously paused screen recording of the user session.
Parameters	*callback* – function that will be called upon completion of the method

Callback

Syntax	*callback = function(data) {*
	data.recording = 1/0;
	data.muted = 1/0;
	data.paused = 1/0;
	}

Description	Function that is called upon completion of the call/screen recording methods.

Parameters	*data.recording* – boolean; indicates if voice/screen recording is currently in progress; note that voice recording is applied to interactions (calls), while screen recording is applied to user sessions *data.muted* – boolean; indicates if voice recording is currently muted; returned only for the voice recording methods *data.paused* – boolean; indicates if screen recording is currently paused; returned only for the screen recording methods