August 17, 2020

Chat Events API

User Manual

Table of contents

Overview

The Chat Events API is a JavaScript-based API that allows customers to listen for events fired by Nuance Digital Engagement Platform (NDEP) and track that information within third-party web analytic solutions to supplement any existing web reporting with information generated by NDEP.

This document provides background and script on how to use APIs for capturing and linking actions to specific chat events.

Chat Event Notification

Nuance provides the ability for our clients to write custom event handlers via APIs so that actions can be defined and linked to certain chat events. Event handlers can be designed to run tasks such as capturing event information to backend systems, reporting event information to analytics engines, or handling other types of notification or custom changes that should be linked to a chat event.
Supported events include the following:

1. Initiation of a proactive or reactive chat
2. C2C events as the chat button is displayed or clicked
3. Sale conversions

Adding Handlers to Chat Events

To add custom handlers to chat events, the client must first declare a function to receive a chat event. Example

Note. "evt" is an information object that Nuance passes to our clients. Inside the “evt” object is data applicable to the type of event the listener is targeting. There are several variables contained within that will be passed, which are contained in the examples.

Chat Listeners

```
<script>
       /* Chat Launched Listener Example */
       var chatLaunchedListener = {
                onChatLaunched: function(evt){ alert("Chat Launched: chatID=" +evt.chatID+
                 ",customerID=" +evt.customerID + " Chat Type: " + evt.chatType + " Biz Rule
              Name: " + evt.bizRuleName + " Event Type: " + evt.evtType);}
       };

       /* Chat Closed Event Listener Example */
       var chatClosedListener = {
              onChatClosed: function(evt){
                  alert("Chat Engaged: chatID=" +evt.chatID+ ", chatType="+evt.chatType+ ",
               evtType=" +evt.evtType + " Biz Rule Name: " + evt.bizRuleName +
               ",customerID=" +evt.customerID);}
       };

       /* Click to Chat State Changed Example */
           var c2cStateChanged = {
               onC2CStateChanged: function(evt) {
               alert("C2C State Changed - rule= "+evt.bizRuleName+", oldstate: " +
               evt.oldState + ", newstate: "+evt.newState + ",customerID="
               +evt.customerID);}
            };
                 //evt.newState values: ready, busy, offline, and progress
</script>
```

C2V Listeners

```
<script>
     /* C2C Displayed Listener Example */
     var c2cDisplayed = {
         onC2CDisplayed: function(evt) {
            alert("C2C Displayed: Business Rule Name: " + evt.bizRuleName + " Customer ID: "
         + evt.customerID ); }
     };
     /* C2C Clicked Listener Example */
     var c2cClickedListener = {
         onC2CClicked: function(evt) {
            alert("C2C Clicked: Business Rule Name: " + evt.bizRuleName + " Customer ID: " +
         evt.customerID );
         }
     };
</script>
```

Sales Listeners

```
<script>
     /* Initialize the Sale Landing Listener */
     //product: "<string value>, quantity: "<string value>
     var saleLandingListener = {
         onSaleEvent: function(evt) {
             alert("Sale Landing: products=" + evt.products + ", quantities = " +
evt.quantities);
         }



     };


     /* Sale Qualified Event Listener */
     var saleQualifiedListener = {
         onSaleQualifiedEvent: function(evt) {
             alert("Sale Qualified: chatID=" + evt.chatID + ", customerID = " +
evt.customerID + ", agentID = " + evt.agentID + ", bizRuleName = " + evt.bizRuleName);
         }
     };


     /* Sold listener example */
     var soldListener = {
         onSoldEvent: function(evt) {
                alert("Sale Completed: saleID=" + evt.saleID + ", agentID = " + evt.agentID +
" , products = " + evt.products + ", quantities = " + evt.quantities);
         }
     };

     /* Chat Engaged Listener Example */
     var chatEngagedListener = {
         onChatEngagedEvent: function(evt) {
         alert("Chat Engaged: chatID=" +evt.chatID+ ", chatType="+evt.chatType+ ",
         evtType=" +evt.evtType + " Biz Rule Name: " + evt.bizRuleName + ",customerID="
         +evt.customerID);}
     };
     //evt.chatType values: C2C, POPUP, or PERSISTENT
</script>
```

Registering the Listeners

```
<script>
     /* Last, register all listeners (that are being used) */
     var InqRegistry = {
         chatListeners: [chatLaunchedListener, chatClosedListener, c2cStateChanged],
         listeners: [c2cDisplayed, c2cClickedListener],
         saleListeners: [saleLandingListener, saleQualifiedListener, soldListener,
         chatEngagedListener]
     };
</script>
```

Errors

If the customer makes a mistake in the event handler, the specific event function will not execute and will skip to another event. An error message will be sent to the client log as a warning.