HSBCNet VA Panel Integration - HSBCNet Version

This will document the process and changes specifically for HSBCNet

Major Changes for v2.x

The code itself has now been refactored into a self-contained module called LivePersonVirtualAssistantModule. All code relating to manipulation of styling / showing and hiding of the panel has been **REMOVED** - this should now be handled directly by the VA Panel code within CV that listens to specific events raised by this module to indicate when to show/hide the panel based on the interaction with the chat window. Events are now triggered by the module when actions should be taken or the button status has changed.

High Level Process Summary

- Website page opens - already tagged with LivePerson code for monitoring etc (via Tealium)
- VA Panel is loaded onto page
    - at this call the LivePersonVirtualAssistantModule.init() function to start the process of event bindings and listeners behind the scenese
- Visitor Opens VA Panel
    - Registering the Chat Button Div Container to load the hidden button
        - call LivePersonVirtualAssistantModule.injectButtonContainer()

        - **Only do this once you know you have shown the panel to the customer and it has been expanded** -- otherwise a button could be loaded off screen which will impact reporting and availability in a negative way
    - Embedded Chat Button Events fire and return a status of either ONLINE/BUSY/OFFLINE which is stored by the LivePersonVirtualAssistantModule
    - Depending on the button state, CV VA Panel will show/hide custom HTML content which is responsive to the display of the device (desktop/mobile)
        - The ONLINE version should include a call to the LivePersonVirtualAssistantModule.startChat(faqHistorySoFar) function which will begin the chat process if agents are ONLINE
        - e.g.

```
  <p id="button-container">
    <a id="lp-va-panel-button-online"
href="#" class="btn btn-success lp-va-panel-
button hide-lp-button"
onclick="LivePersonVirtualAssistantModule.st
artChat(faqHistorySoFar);">Click to chat</a>
    <a id="lp-va-panel-button-offline"
href="#" class="btn btn-warning lp-va-panel-
button hide-lp-button">All Agents are
OFFLINE/BUSY</a>
  </p>
```

- Visitor has conversation with CV A.I. FAQ engine and can see the chat button at the bottom of panel
- If Visit Clicks Chat Button

    - call custom function to "fake" click the hidden button to start the chat -- see LivePersonVirtualAssistantModule.escalateToChat(faqHistorySoFar)
    - (optioanl) feed in the faqHistorySoFar by passing in an array of strings representing the current conversation history you wish to recap to the agent and consumer inside the LE2 chat window

- CV VA Panel should Listen for Chat Session Started Event to hide VA Panel
    - see lpTag.events.bind("lpUnifiedWindow", "state"...
    - see internal method hideVaPanel() for how this is done in the POC
- CV VA Panel should Listen for Chat Session Ended Event/Exit Survey Submitted Event to re-show VA Panel
    - see lpTag.events.bind("lpUnifiedWindow","conversationInfo"...
    - see internal method showVaPanel() for how this is done in the POC

"*How do I know when to show/hide the Panel?*"

- Bind to the events fired by the LivePersonVirtualAssistantModule as follows:

```
lpTag.events.bind('LP_VA_PANEL_MODULE', '*', function
(eventData, info){
  console.log('! LP_VA_PANEL_MODULE //
',info.eventName);
  if(info.eventName == 'SHOULD_SHOW_BUTTON_CONTENT')
{
    // put code here to show/hide the relevant live
chat button HTML within the VA panel as required
based on the button status online/offline/busy etc...
  }
```

```
    if (info.eventName == 'SHOULD_SHOW_VA_PANEL') {
      // code goes here to show the panel again
    }
    if (info.eventName == 'SHOULD_HIDE_VA_PANEL') {
      // code goes here to HIDE the panel
    }
  });
```

- info.eventName object will tell you which event has fired and what you should do next.

Events List

- SHOULD_SHOW_VA_PANEL => recommended to show the panel again
- SHOULD_HIDE_VA_PANEL => recommended to hide the panel again
- SHOULD_SHOW_BUTTON_CONTENT => a button has been loaded on the page which is related to the VA Panel. Inspect the eventData object to check the button state (ONLINE/BUSY/OFFLINE) and then show the relevant HTML content within the panel accordingly.
    - eventData.status = 'ONLINE' / 'OFFLINE' / 'BUSY'
    - eventData.state_enums
        - BUSY: 4
        - OFFLINE: 2
        - ONLINE: 1
        - UNKNOWN: 0
    - eventData.state_descriptions =
        - 0: "UNKNOWN"
        - 1: "ONLINE"
        - 2: "OFFLINE"
        - 4: "BUSY"
    - eventData.state = 0 | 1 | 2 | 4
    - eventData.reason = description why the event was shown

Configuration through _config

Which options can I change?

- _config.USING_PROXY_BUTTON
    - true ==> tells the code you will be displaying your own HTML buttons for the online/offline state of the LivePerson Embedded button which will be loaded.
        - **PLEASE NOTE** This option is recommended for responsive requirements as any images deployed by the LivePerson system will not responsive. We can deploy HTML which you style via CSS if needed.
        - If using this option ensure that whatever HTML elements of your **ONLINE** button state have an onclick function call to triggerChatButtonClick (or equivalent code) which will start the chat window by calling our API and fake clicking the actual button loaded on the page - which will probably be hidden/have no actual viewable HTML content.
        - If you do not wish to handle the custom elements then suggest setting this to false
    - false ==> will exclude the following functions from execution ... hideLivePersonButtonContainers AND refreshProxyButtonStatus ... this is because the LivePerson button shown will contain the HTML required for the online/offline states.

- _config.SEND_FAQ_CONVERSATION_AS_PRECHAT_LINES
    - true ==> **ONLY HAS AN IMPACT IF _config.USING_PROXY_BUTTON == true** -- because default chat button clicks cannot be intercepted. Only using a proxy button allows this feature to be in scope.
        - enables the addPreChatLinesToChat example function

to return some preChatLines to the chat window as part of the triggerChatButtonClick function call.
        - Follow this function's approach if you want to pass something like the last question asked to AskAndrew into the chat window for the agent and consumer to see in the header.
        - **PLEASE NOTE** this feature shows the key API calls you need to make but how you get the information in question from AskAndrew is down to you.
        - The only requirement is you pass it into the click call with the named parameter as an array [] of Strings.
    - false (default) ==> disables the behaviour