Client Samples

Genesys Mobile Engagement 8.5.1

1/11/2022

Table of Contents

Genesys Mobile Services Client Samples

These pages provide a brief overview of the sample applications included with Genesys Mobile Services. As additional sample applications are made available, this guide will be updated to provide supporting information.

> **Important**
> The JavaScript Sample is now documented in the Service Management UI Help.

New in This Document

The following topics have been added or changed in the GMS 8.5.109 release.

- In page iOS Sample, the downloadable source file was updated.

The following topics have been added or changed in the GMS 8.5.108 release.

- In page Interaction Workspace Plugin Sample, the downloadable binary and sources were updated.

No topics have been added or changed in the GMS 8.5.107 release.

No topics have been added or changed in the GMS 8.5.106 release.

The following topics have been added or changed in the GMS 8.5.105 release.

- In page Android Sample, section Implemented Scenarios was updated.

The following topics have been added or changed in the GMS 8.5.104 release.

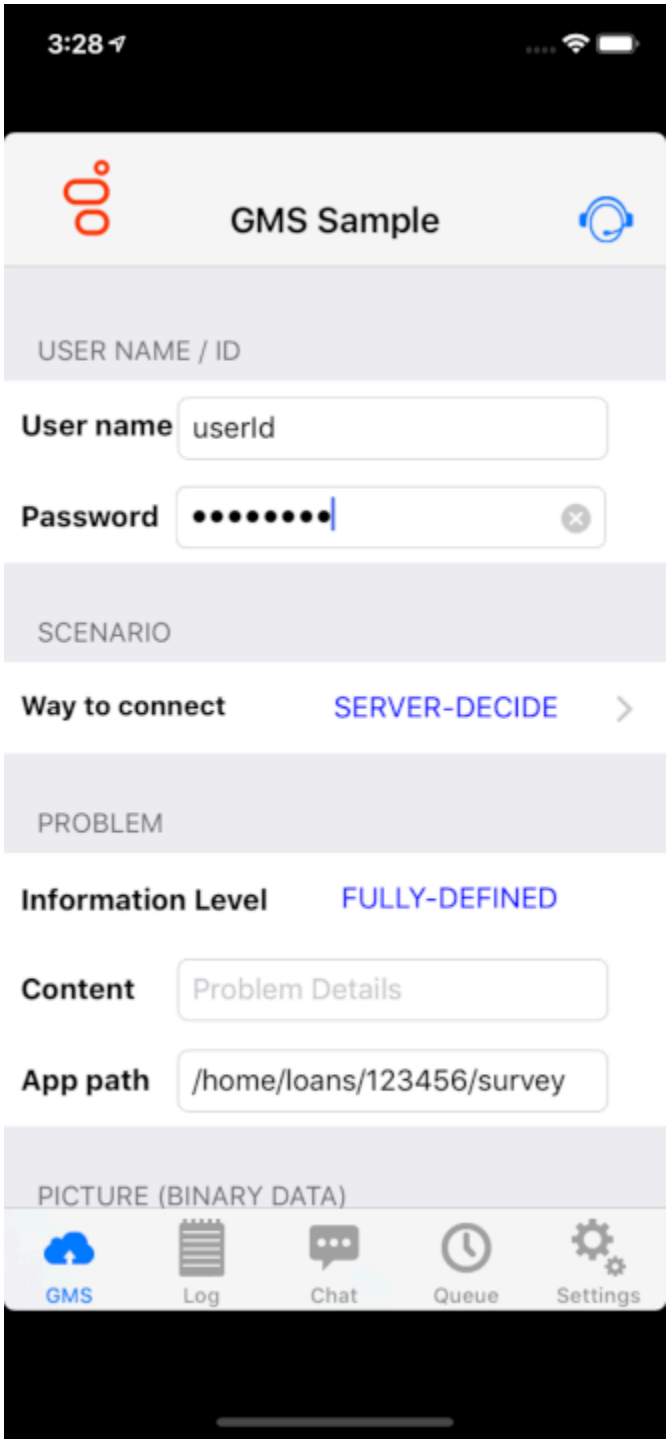
- In page Interaction Workspace Plugin Sample, section Customization (sources) was updated.

The following topics have been added or changed in the GME 8.5.102.05 release:

- The Java Script Sample has been moved to the Service Management Help.
- The Interaction Workspace Plugin Sample was enhanced to display a pop up for callback disposition.

No topics have been added or changed in the GMS 8.5.100.05 release.

iOS Sample


3:28
GMS Sample
USER NAME / ID
User name
userId
Password
SCENARIO
Way to connect
SERVER-DECIDE
PROBLEM
Information Level
FULLY-DEFINED
Content
Problem Details
App path
/home/loans/123456/survey
PICTURE (BINARY DATA)
GMS
Log
Chat
Queue
Settings

You can log into the UI hub at this URL:
`<GMS Local Host>:8080/genesys`

You can download the source code via the following zip file:
Genesys Mobile Services iOS Sample Source Code (1.4.14).

This sample application provides you with the following scenarios:

- Chat-Now
- Chat-Wait
- Voice-Now-UserOrig
- Voice-Now-UserTerm
- Voice-Wait-UserOrig
- Voice-Wait-UserTerm
- Voice-Scheduled-UserTerm

Notes:

- Go to **Settings** and change the GMS service URL to point to the Callback service that you configured.
- The application does not trim +1 from the phone number. Make sure the phone number in the **Settings** tab is compatible with your test environment if you are using either matching by DNIS+ANI or matching by ANI only. Depending on your setup, T-Server reports ANI with or without a leading country code.

Corrections

Version 1.4.14

- Built with updated distribution profile.

Version 1.4.13

- The iOS sample code now calls the iOS location manager API function to request access to location information and now properly sets `location_lat` and `location_long` parameters in callback requests. Previously, `location_lat` and `location_long` parameters were always set to zero in callback requests.
- The "_" character prefix has been removed from the following callback request parameters: `photo`, `app_path`, `problem`, `problem_info_level`, `date`, and `customer_segment`. These parameters are now added to the call's attached data.

Version 1.4.12

- This version is compatible with Xcode version 7.1.1 and iOS version 9.1. (GMS-2800)

- The chat transcript view is now correctly updated on 64 bit iOS devices.
- Selecting the **Queue** tab no longer forces the application to exit.

Version 1.4.11

- This version is compatible with Xcode version 7.0.
- To ensure compatibility with iOS 9, this version includes the application `plist` parameter: `"NSAppTransportSecurity / NSAllowsArbitraryLoads = true"`.

Version 1.4.10

- Chat client now includes `transcriptPosition` in the Comet `/meta/connect` message.
- This beta release includes support for screen sharing.

Version 1.4.9

- This beta release includes support for screen sharing.

Version 1.4.8

- Text lines in the chat view are no longer overlapping on 64-bit iOS devices.

Version 1.4.7

- Max notification size is now set to 256 if iOS version is older than 8.0. Otherwise, the max size is set to 4096.

Version 1.4.6

- The app no longer switches from the **Chat** tab to the GMS tab when resumed from background.
- If you enable the "`Clear chat on new session`" option in the **Settings** tab, the **Chat** tab content of the previous chat session is cleared each time a new chat session is initiated.

Version 1.4.5

- If the app is in foreground state, the app no longer terminates when receiving a push notification.

Version 1.4.4

- The app can now work with chat and can handle background status introduced in 8.5.1.
- The User interface for setting the GMS base URL has changed. It is now set by three components:
  - Host–Set to the hostname or IP address.
  - Port–Set to the port number, for example `8080`.
  - Application–Set the the application name, by default `genesys`.

Version 1.4.3

- The app can now work with the new Callback Availability API.

Version 1.4.2

- The app no longer fails to install on iPhone 4S devices.

Version 1.4.1

- Editing the server URL now update the main settings field.
- Location latitude and longitude fields are now properly encoded in Callback requests.

Version 1.4

- Schedule Callback is now supported.
- In the GMS tab, you can now set `Way to Connect` to `VOICE-SCHEDULED-USERTERM` to enable the `Desired Time` and `Selected Time` fields.
  - Tap **Desired Time** and select a desired callback time.
  - If the contact center is opened at the given **Desired Time**, tap **Desired Time** to select callback time in the list of available slots.
  - If the contact center is closed at the **Desired Time**, **Selected Time** will display a message indicating that the contact center is closed.
  - Tap **Desired Time** to try another time.
- The GMS URL (in the **Settings** tab) is now configured in two parts.
  - Set URL to `http://host:port/genesys/1/service`
  - Set **Callback Service** to the execution name of the Callback service.

Disclaimer

Android Sample

Updated in 8.5.112, 8.5.114

You can download the source code via a zip file: Genesys Mobile Services Android Sample Source Code

Overview

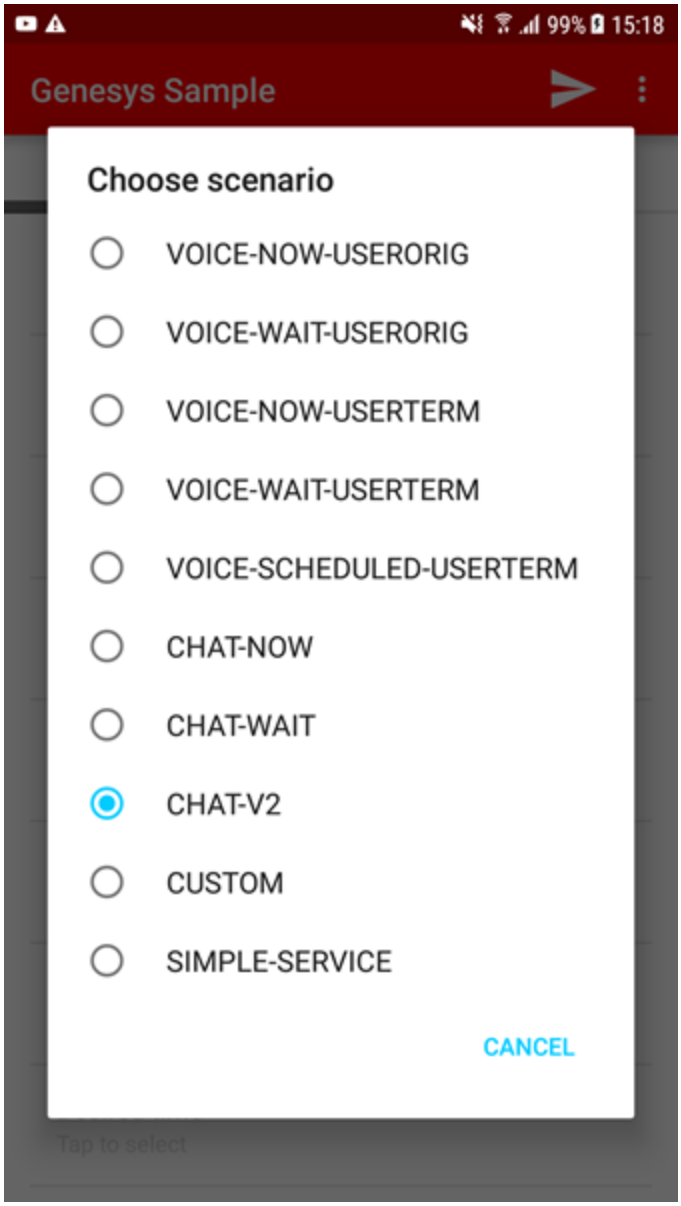
The Genesys Mobile Services (GMS) Android sample illustrates how to implement an Android app that communicates with GMS and performs supported contact scenarios. It is primarily meant to be used by developers as a reference to build an Android app with GMS. This sample app can also be used to test an existing GMS deployment because the scenario parameters are configurable on the GMS Service Management UI.

In 8.5.114, the app was extended to support:

- Chat V2 features, that are now available in the `CHAT-V2` scenario:
  - File Transfer
  - Typing notifications
  - Typing preview
- Mobile Push Notifications to Android devices using Firebase Cloud Messaging (FCM)
- ApiGee integration

Implemented Scenarios

The Android sample supports the **Click to Call** and **Click to Chat** scenarios, in addition to Callback Scenarios.



- Voice Callback scenarios:
  `VOICE-NOW-USERORIG`

  `VOICE-WAIT-USERORIG`

  `VOICE-NOW-USERTERM`

  `VOICE-WAIT-USERTERM`
- Chat V1 scenarios:
  `CHAT-NOW`

  `CHAT-WAIT`

- Chat V2 implementing CometD:
  `CHAT-V2`
- `SIMPLE-SERVICE`—Enables you to test any of your callback services.
- `CUSTOM`—Enables you to test a simple service (not only the Callback service). For instance, you can test a stat service. The sample displays the result in a dialog box.

These scenarios are server-driven, which means that the server instructs the client with the actions needed to carry out the scenario. The client just needs to perform those actions and follow up the dialog with the server. Therefore, the client is flexible enough to support any scenario that is built using the same kind of actions. The following actions are supported:

- `DialNumber` - The app makes a phone call.
- `ConfirmationDialog` - The app displays a message.
- `DisplayMenu` - The app displays a menu for the user to select an item that may affect how the scenario proceeds.
- `StartChat` - The app starts a chat conversation. Asynchronous HTTP notifications (comet messages) are used for receiving chat server events.

Push notifications through Firebase Cloud Messaging are supported. Delayed scenarios are supported by using push notifications only; the app will not poll the server to be notified about agent availability.

Prerequisites

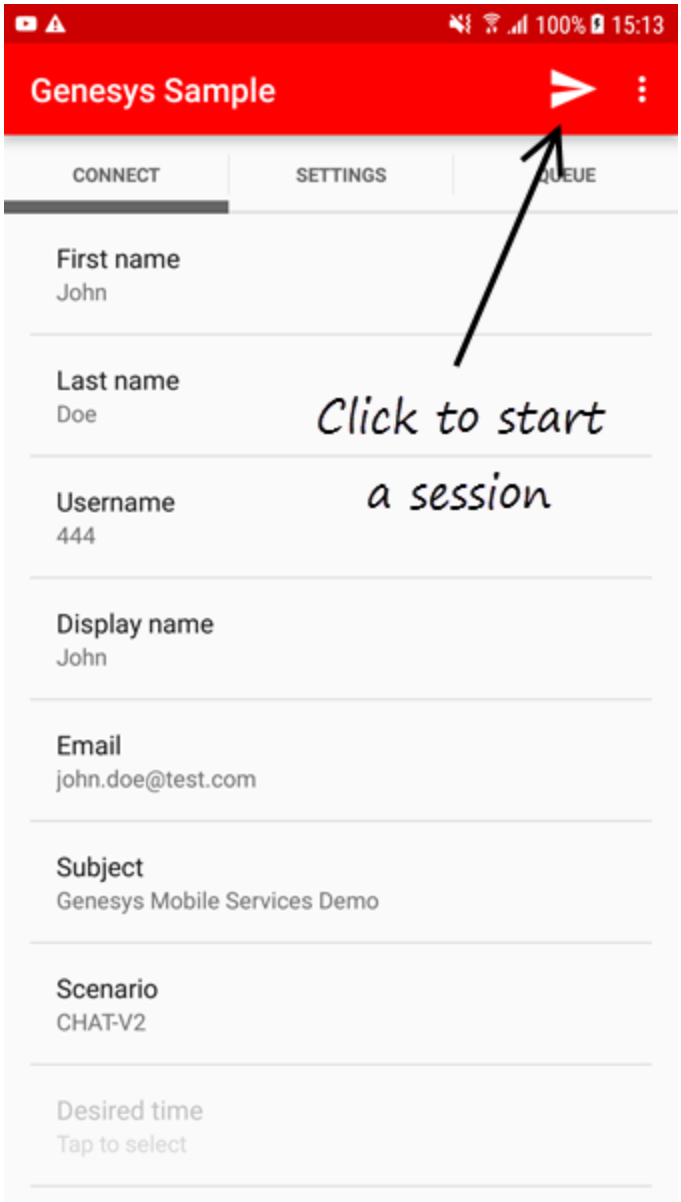
In order to use the Android sample app, you need to have GMS version 8.5.114 and higher installed and running. Additionally:

- Create your **gmstester-callback** service in the Configured Services panel of the Service Management UI.
  - Click create and select callback in the **Service Template** list.
  - Enter `gmstester-callback` for the **Service Name** and select `sample` in **Common Default Configuration**.
- Deploy and configure the services that you want to use in the Configured Services panel of the Service Management UI or in the GMS Configuration for the Digital Channel APIs. For example, create the `chat.customer-support` service for **CHAT-V2** in your GMS configuration.
  - See Adding a Callback Service for further details about callback services.
  - See ORS-Chat Services for Chat V1 services.
  - See Configuring the Digital Channels API for details about Chat V2 configuration.
- Configure Firebase Cloud Messaging. Edit the `fcm` section of the GMS configuration to provide your FCM API key.
- Enable push notifications for Firebase Cloud Messaging in your GMS configuration:
  `pushEnabled=fcm`

> **Important**
> Restart GMS after configuring Firebase Cloud Messaging in the `fcm` and `push` sections.

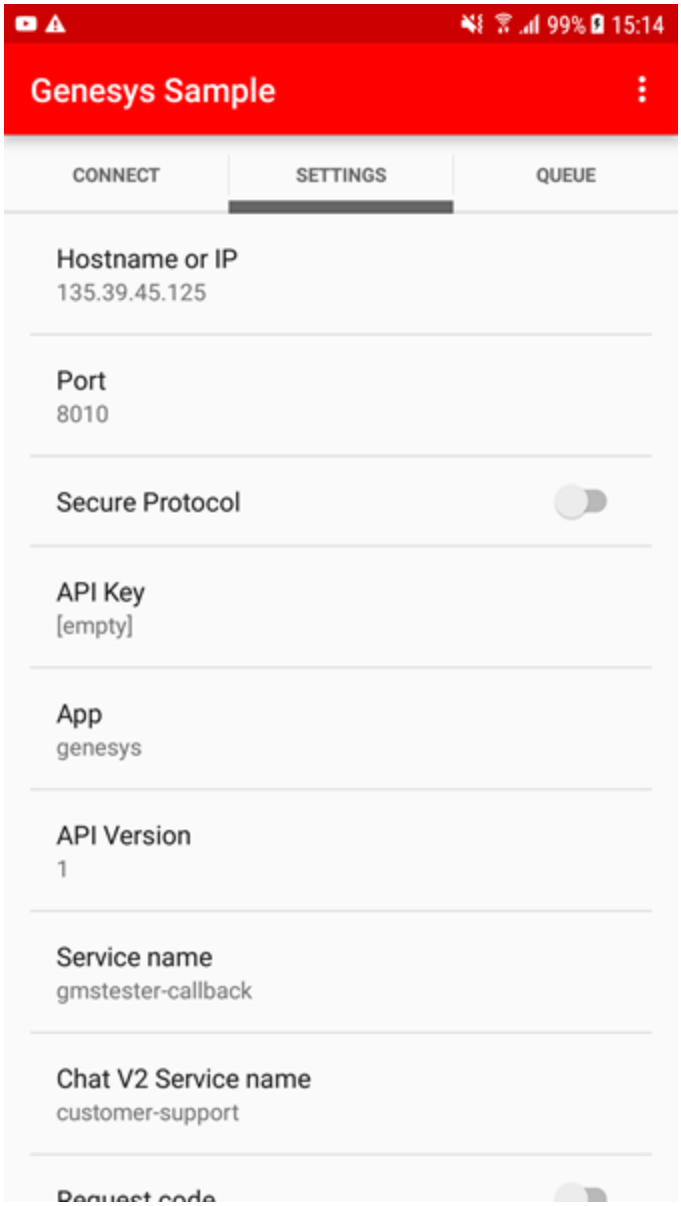
Running the Sample

In order to run the Android sample, you can just download the APK file and deploy it on your Android device. Alternatively, you can also compile the source code to make your own APK file by using the Android SDK. Using the sample is easy, as long as GMS is already deployed.

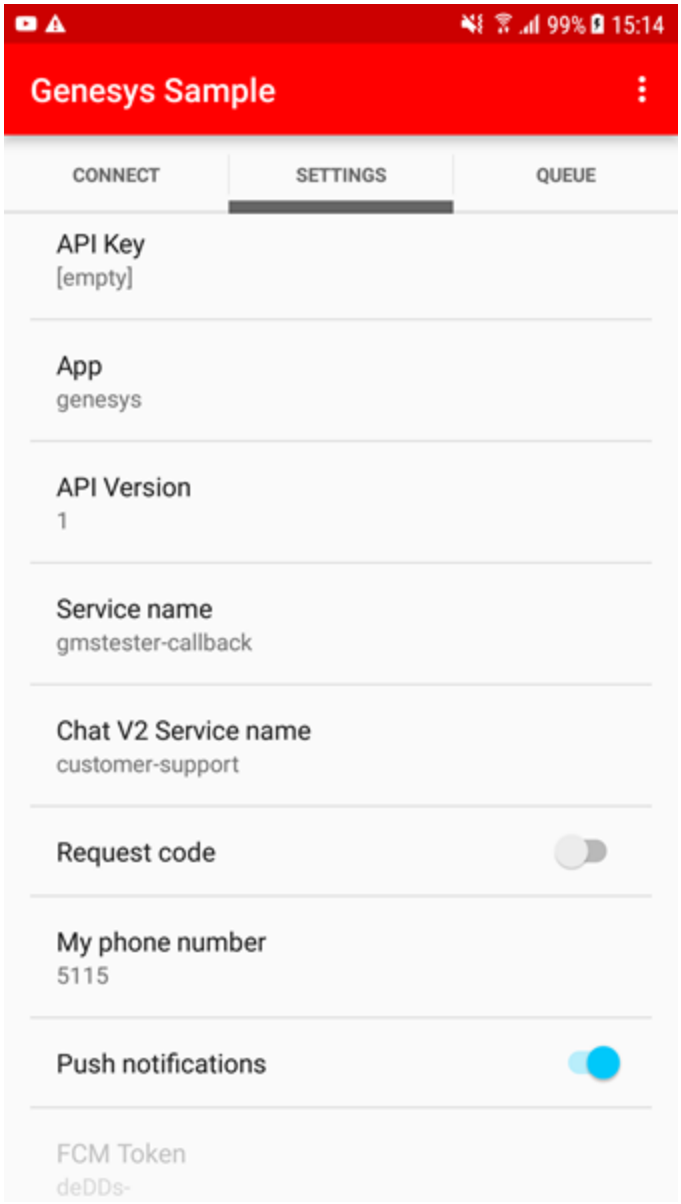


The startup screen is a configuration screen. Remember that in order for push notifications to work, you need to have registered a project in Google Cloud, and have enabled the *Firebase Cloud Messaging (FCM) for Android* API. In the app, you have to fill the FCM sender id with your Google Cloud project number. For more details, please refer to Firebase Cloud Messaging - Getting Started.

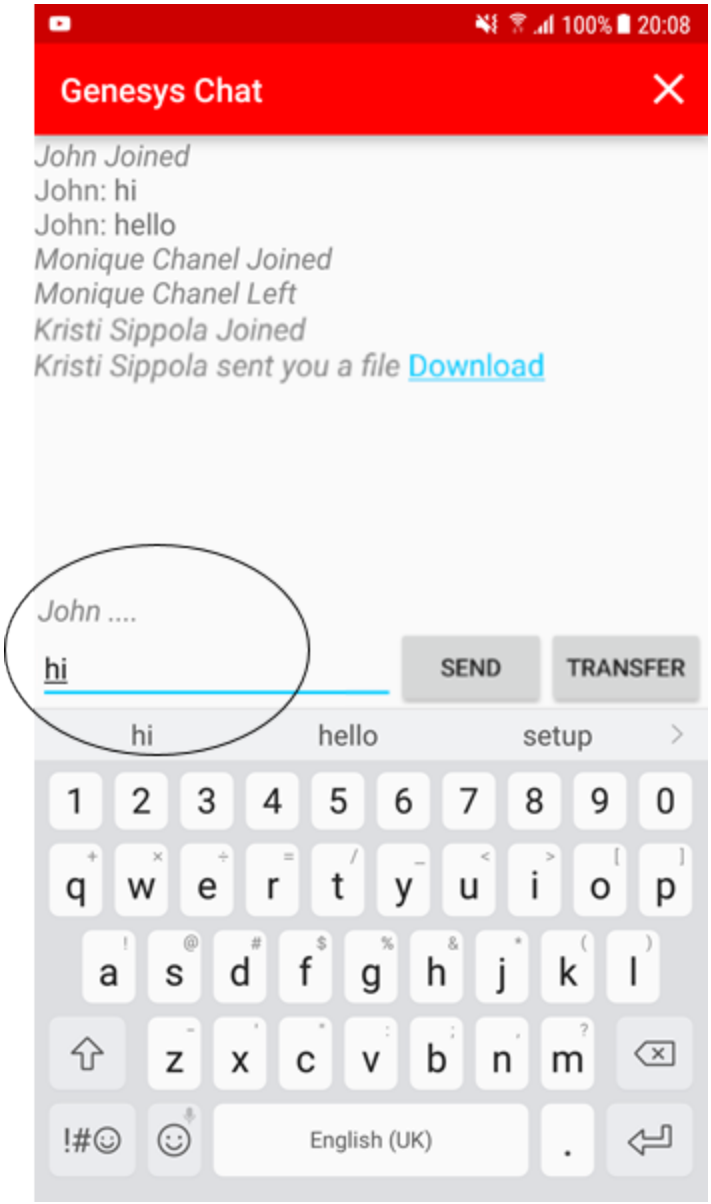
Click or tap **Scenario** to display the list of available scenarios, then select one of them, for example, CHAT-V2.



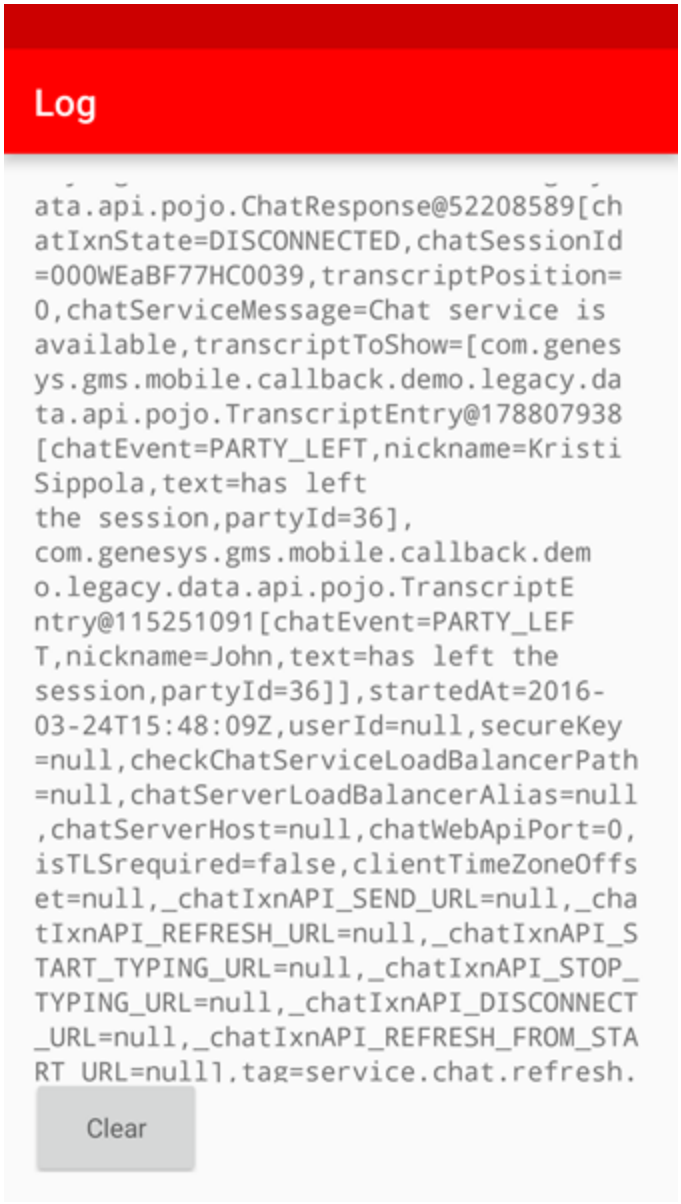
Fill the configuration options with the appropriate values for your GMS environment. You will need your sample service name, for instance, `gmstester-callback`, and the service name used to implement your CHAT-V2 scenario, for example, `customer-service`.



Scroll down and make sure that push notifications are enabled. Then, connect from the first screen.



The Chat session will start, enabling you to send and receive messages and files. You will be able to see when the other participant is typing and to leave the chat session by clicking the '**X**' icon.



In order to review in detail the past communication between the client and the server, you can use the *Log* screen, which you can open from the *show log* option in the startup screen.

You can also inspect the standard Android logging, by using the Android logcat tool, for obtaining more details or displaying logs more comfortably on your computer screen. Filter by the *GenesysService* tag in order to see the client-server communication.

About the Code

The app is designed to be minimal so that developers can focus on the implementation of Mobile Services scenarios.

The app consists of three screens (Android activities):

- `GenesysSampleActivity` - Displays the app configuration options, allows launching the Mobile Services scenarios, and interprets the server-driven actions.
- `GenesysChatActivity` - Chat screen.
- `LogActivity` - Displays logs.

The `GenesysSession` class is in charge of the client-server communication via HTTP for a Mobile Services user session, including comet notifications. It relies on the Jetty HTTP Client, and on the CometD Java Client. `GenesysSession` can be reused "as is" in your code. (Note: this is provided as sample code, so always test your app thoroughly. See the disclaimer below.)

The `GenesysService` class implements an Android service responsible for creating `GenesysSession` objects and keeping them running in the background, even when the user switches to other apps. `GenesysService` distributes server responses and comet messages to the preceding activities.

The `FcmMessagingServiceclass` is in charge of receiving push notifications. It is implemented following Android recommendations, using the Google Play Services library. See Firebase Cloud Messaging - Implementing FCM Client.

The `ChatSession` class and some companion classes implement chat operations. These classes are meant to be reusable in your code. (See the disclaimer below).

Disclaimer

Classic Callback Sample

You can download the Classic Callback (VoiceXML) sample application via a zip file. The zip file also includes Composer callflow diagrams.

Genesys Mobile Services Classic Callback Sample

> **Important**
> This example's strategy was created with Composer 8.1.300. If you are using Composer 8.1.400 or newer, you must upgrade the Composer project.

Overview

The Genesys Mobile Services (GMS) Classic Callback sample illustrates how to implement an IVR (Genesys Voice Platform VoiceXML) application that communicates with GMS and performs classic Callback scenarios. It is primarily meant to be used by developers as a reference to build a Composer callflow application that interacts with GMS.

Scenarios

The following scenarios are implemented as one Composer project, which consists of one driver callflow and a corresponding subcallflow for each scenario.

IVR Controlled Callback Using GMS Stat Server API

1. Caller is in Genesys Voice Platform (GVP) and requests an Agent.
2. GVP gets the estimated wait time (EWT) using GMS Stat Server API.
3. GVP logic determines Callback should be offered (EWT threshold).
4. GVP offers Caller the choice of a Callback ASAP or to stay on the line and wait for an Agent.
    - Caller chooses Callback ASAP and hangs up. GVP creates a new service request to GMS using Callback API. Caller is called back when an Agent about to become available.
    - Caller chooses to stay on the line and wait for an Agent, and the call is connected when an Agent is available.

IVR Controlled Callback Using GMS Callback API

1. Caller is in GVP and requests an Agent.

2. GVP start a new GMS service (HTTP API) and a virtual interaction is placed in the queue to hold the position of the caller.
3. GVP gets the estimated wait time (EWT) and Position for the virtual interaction using the GMS Callback API.
4. If the EWT-Position Threshold is not met, implying that the wait time is not long.
    1. Wait for an Agent by continuing polling GMS Callback Service API until Agent becomes available or the EWT threshold is reached.
5. If the EWT-Position Threshold is met, implying that the wait time is long.
    1. GVP offers Caller the choice of a Callback as soon as an Agent becomes available and without losing position in the queue, or to stay on the line and wait for an Agent. Optionally informs the caller about the EWT and Position in the queue.
    2. Caller chooses Callback and hangs up. The caller is called back when an Agent is about to become available.
    3. Caller chooses to stay on the line and wait for an Agent, and the call is connected when an Agent is available.

Prerequisites

In order to use this sample application, the following prerequisites are required:

- GMS installed and running, with JDK 1.7.
- The services that you want to use must be deployed.
- The sample can be deployed on all web servers supported by Composer 8.5.
- (Optional to play VoiceXML) MRCP enabled for GVP using the tts option. GVP uses MRCP speech synthesis technology to incorporate text-to-speech for use in voice applications.

Running the Sample

1. Download the zip file.
2. Extract the zip into the following directory: `<gmsinstalldir>/webapps`.
3. Modify the `<gmsinstalldir>/webapps/ClassicCallbackSample/src/AppRoot.vxml` file to set values for the following variables, which are dependent on the environment:
    - `gms_uri`
    - `gms_service_name`
    - `stat_api_params` – this value should be same as the Virtual Queue specified in the GMS service configuration for the preceding service (gms_service_name)
    - `gms_inbound_service_rp` – any routepoint where a strategy/workflow can route the call to the agent.
    - `gms_transfer_routepoint` - routepoint where the GMS classic call inbound service is configured.

4. Edit the start.ini file to make sure that it contains:

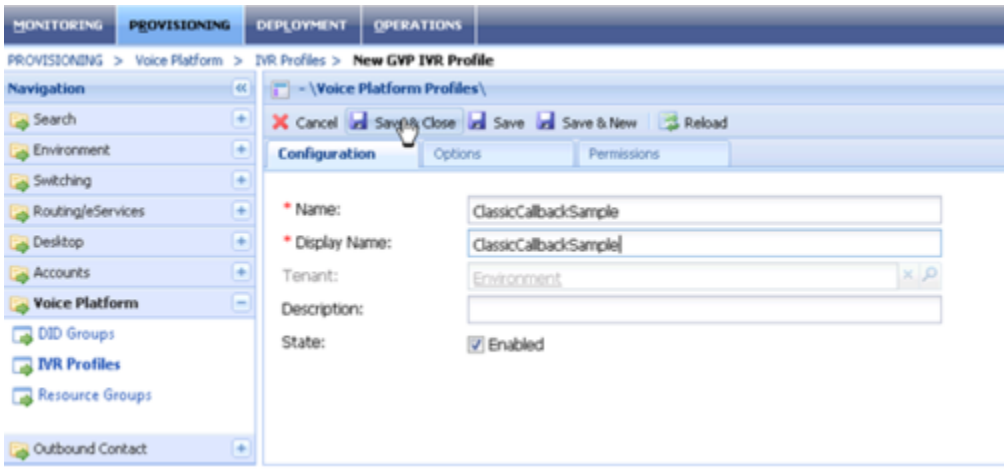
```
--
module=server,jsp,jmx,resources,websocket,ext,plus,annotations,deploy,security,servlets,continuation
etc/jetty.xml
etc/jetty-ssl.xml
etc/jetty-deploy.xml
etc/jetty-http.xml
etc/jetty-https.xml
jetty.send.server.version=false
```

> **Important**
>
> Comment any **rewrite** line. You should not run the sample in a Production server.

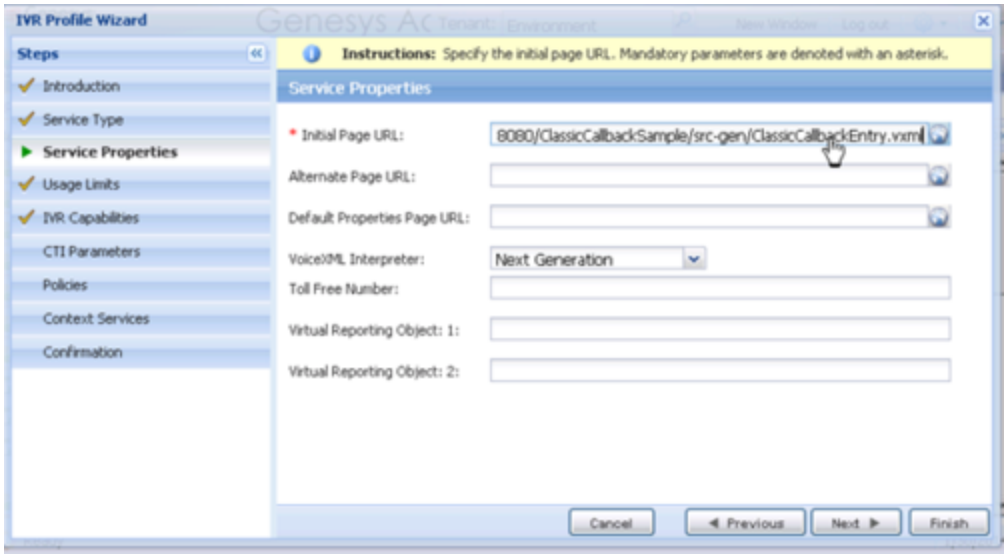
Verify that the following URL is accessible: http://<gmshost>:<gmsport>/ClassicCallbackSample/src-gen/ClassicCallbackEntry.vxml

Configure IVR Objects

1. In Genesys Administrator > Provisioning, create a new IVR Profile named `ClassicCallbackSample` with a Service Type of `VoiceXML`.



2. Provision the IVR Profile to point to the URL shown in Step 4.



3. Setup a DID group (a trunk group to the IVR Profile mapping).

4. Configure the DID as a `TrunkGroup` that points to Resource Manager.

5. Make a call to the DID and follow the voice prompts.

Brief Introduction to the Code (Composer Callflow Diagrams)

ClassicCallbackEntry.callflow

1. Common entry into the IVR samples for all of the supported scenarios.
2. Acts as a driver application for the sample.
3. Plays welcome message.
4. Collects user data (account number).
   - 1111 - for Scenario 1.
   - 2222 – for Scenario 2.
   - Other – play error prompt and repeat Step 4.
5. Based on the selected scenario, invokes the following subroutines:
   - SubGMSStatsCallback.callflow for Scenario 1.
   - SubIVRConrolledGMSCallback.callflow for Scenario 2

SubGMSStatsCallback.callflow

1. Get EWT using GMS Statistic API .../genesys/1/statistics (EstimatedWaitTime).
2. Get current interactions waiting in queue .../genesys/1/statistics (current_In_Queue).
3. Check Callback offer threshold.

- Offer Callback.
  - Invoke SubCallbackOfferDialog.
  - Callback declined.
    - Yes, default route the call (continue with no Callback).
    - No, exit.
- Do not offer Callback.
  - Default route the call (continue with no Callback).

SubIVRControlledGMSCallback.callflow

1. Start Callback service (USERORIGINATED).
2. Get EWT and Position using GMS Callback API .../genesys/1/callback (check-queue-position).
3. Check agent available.
   - Agent available.
     - Get access number.
     - Blind transfer to access number.
     - Exit.
   - Agent not available, so check Callback offer threshold.
     - Offer Callback.
       - Invoke SubCallbackOfferDialog.
       - Callback declined.
         - Yes, default route the call (continue with no Callback).
         - No, exit.
     - Do not offer Callback.
       - Go to Step 2.

Disclaimer

Custom Callback Plugin Sample

You can download the GMS Custom Callback Plugin sample application via the following zip file, which includes Composer call flow and workflow diagrams.

- Genesys Mobile Services Custom Callback Plugin Sample, version 8.5.112

For a description of the On-Dial plugin interface and configuration, see the Playing Treatments page in the Callback Solution Guide.

Plugin Description

The Genesys Mobile Services (GMS) Custom Callback Plugin sample implements an On-Dial plugin to interface with the GMS Callback service. Developers should use this sample as a reference to build a Composer application that is invoked as a plugin from GMS Callback. Though the actual functionality of the plugin can be extended to execute custom logic like sending push notifications, SMS, or emails, it is beyond the scope of this sample to demonstrate these. For implementing custom logic, refer to GMS, ORS, and Composer documentation.

This example covers:

- Parsing the JSON content of the `_user_data` parameter into an object.
- Checking the `_interaction_id` parameter to determine if the call was answered.
- Checking the `_call_state` parameter for the human answer and machine answer use cases.
- Playing a music file message for scenarios involving a machine answer.
- Invoking a VXML application for scenarios involving a human answer.
- Setting user data in the interaction.
- Implementing a reusable sub-workflow that detaches the interaction and returns the plugin reply.

Note that the Callback Service template provides default treatments when an outbound call reaches a human or not. Customization of the template is required to provide custom treatment. In this release, the custom treatment is supported through SCXML or VoiceXML plugins as configurable options for the Callback Service template. This reduces the customization efforts and gives you more control over how the outbound call is treated.

Prerequisites

In order to use this sample application, you need the following:

- GMS installed and running, with JDK 1.7 or JDK 1.8.

- The services that you want to use must be deployed.
- The sample can be deployed on all web servers supported by Composer 8.5.

Running the Sample

1. Download the zip file here
2. Unzip to C:\temp to create a `GMSExamplePlugin` folder.
3. Open Composer.
4. Browse `File > Import > General` and select the `GMSExamplePlugin` folder. Click **OK** to import.
5. Generate the code and deploy the project.
6. Make sure that the scxml application is accessible via `http`.
7. Configure _plugin_on_dial_url = <scxml http path>
8. Execute the `VOICE-USERTERM-NOW` scenario.

Disclaimer

Interaction Workspace Plugin Sample

You can download the Interaction Workspace Plugin sample application binary and sources using the following links. The `Sources` zip file contains a Visual Studio project for customization.

> **Important**
> Interaction Workspace (IW) is also known as Workspace Desktop Edition (WDE).

Modified in 8.5.108.02, 8.5.109.05, 8.5.111.04

Sources

- Genesys Mobile Services IWS Plugin Sample – Sources version 2.1

Overview

The Interaction Workspace Plugin (IWS) sample illustrates how to implement an IWS plugin application that communicates with GMS and allows an agent to do the following:

- Preview a Callback, and then decide to accept or reject the request.
- Set disposition on completed/cancelled Callbacks, and reschedule or retry a Callback when the outbound call is unsuccessful.

The sample is primarily meant to be used by developers as a reference to build an IWS plugin application that integrates with Interaction Workspace and interacts with GMS to provide Preview/ Retry Callback.

Scenarios

- For the Preview feature, see the User Terminated Preview scenario.
- For the Retry feature, all User Terminated scenarios can be used with:
  - _enable_disposition_dialog = true
  - _business_hours_service set with the name of your Office Hours service.

See Preview and Disposition Scenarios page for further details.

Prerequisites

In order to use this sample application, the following prerequisites are required:

- GMS 8.5.005.xx and higher for Preview Callback
- GMS 8.5.006.xx and higher for Preview and Retry Callback
- Interaction Workspace 8.1.401.20 or Workspace Desktop Edition 8.5.106.21 or higher
- (Optional for customization) Visual Studio Development Environment (use Visual Studio Professional 2013) for Interaction Workspace 8.1.401.xx version or for Workspace Desktop Edition 8.5.106.21+ versions
  - Install .NET framework 3.5 (for IWS 8.1.401.xx) or .NET framework 4.5 (for IWD 8.5.106.21+); then, select the correct .NET framework to build the project.

Compiling the Sample

Download the Visual Studio project and extract the files to create a folder `GMS_IWSPlugin`. Open the `GMS_IWSPlugin.sln` file in Visual Studio.

Procedure

1. This project is primarily dependent on the IWS libraries. To ensure that the dependency libraries are available to the project build process, copy the contents of `<Interaction Workspace Install Folder>/InteractionWorkspace` (folder containing `InteractionWorkspace.exe`) in the build target folder. You can do this by setting `Project Properties > Build Events > Open file > Pre-build` event command line. For example: `xcopy "C:\Program Files (x86)\GCTI\Interaction Workspace\InteractionWorkspace" "$(TargetDir)" /y /i /s`
2. For debugging, make sure that the `Project Properties > Debug` settings are correct.
3. After customization is complete and a successful build is done, copy the contents of `$TargetDir` back to the folder containing `InteractionWorkspace.exe` in the IWS installation.

```
<Dictionary EnglishName="English" CultureName="English" Culture="en-US">
<Value Id="InvitationDialog.FormCaption" Title="Callback Invitation" />
<Value Id="InvitationDialog.RemainingTime" Title="seconds remain" />
<Value Id="InvitationDialog.Message" Text="A personal callback is offered to you. Please
refer to the information below and use one of the response buttons." />
<Value Id="DisplayData_1" Translation="Field 1" />
<Value Id="DisplayData_2" Translation="Field 2" />
<Value Id="DisplayData_3" Translation="Field 3" />
<Value Id="DisplayData_4" Translation="Field 4" />
<Value Id="DisplayData_5" Translation="Field 5" />
<Value Id="DisplayData_6" Translation="My New Preview Field" />
<Value Id="ResponseButton_accept" Translation="Accept" />
<Value Id="ResponseButton_reject" Translation="Reject" />
<Value Id="ResponseButton_reschedule" Translation="Reschedule" />
<Value Id="ResponseButton_xcancel" Translation="Cancel" />
</Dictionary>
```

Running the Sample

Download the binary files and extract them to the Interaction Workspace or Workspace Desktop installation directory. Make sure that the binary files are extracted into the same folder as the `InteractionWorkspace.exe`.

Setting Preview Field Labels and Adding/Removing Fields

1. Open the file `<plugin dir>/Languages/Genesyslab.Desktop.Modules.GMS.CallbackInvitation.en-US.xml`.
2. Update the **bolded** text to intended values, and then restart Interaction Workspace. These new values will be displayed in the IWS plugin dialog on GMS preview.

```
<Dictionary EnglishName="English" CultureName="English" Culture="en-US">
  <Value Id="InvitationDialog.FormCaption" Title="Callback Invitation" />
  <Value Id="InvitationDialog.RemainingTime" Title="seconds remain" />
  <Value Id="InvitationDialog.Message" Text="A personal callback is offered to you. Please
refer to the information below and use one of the response buttons." />
  <Value Id="DisplayData_1" Translation="'''Field 1'''" />
  <Value Id="DisplayData_2" Translation="'''Field 2'''" />
  <Value Id="DisplayData_3" Translation="'''Field 3'''" />
  <Value Id="DisplayData_4" Translation="'''Field 4'''" />
  <Value Id="DisplayData_5" Translation="'''Field 5'''" />
  <Value Id="DisplayData_6" Translation="'''Field 6'''" />
  <Value Id="ResponseButton_accept" Translation="Accept" />
  <Value Id="ResponseButton_reject" Translation="Reject" />
  <Value Id="ResponseButton_reschedule" Translation="Reschedule" />
  <Value Id="ResponseButton_xcancel" Translation="Cancel" />
  <Value Id="ResponseButton_done" Translation="Done" />
  <Value Id="DispositionDialog.FormCaption" Title="Callback Disposition" />
  <Value Id="DispositionDialog.RemainingTime" Title="seconds remain" />
  <Value Id="DispositionDialog.Message" Text="A disposition is offered to you. Please refer
to the information below and use one of the response buttons." />
</Dictionary>
```

3. Add or remove the `<Value Id=”DisplayData_...` items as necessary while ensuring that the `DisplayData_1, DisplayData_2...` order is maintained.

About the Code

UserEventListener.cs

- `EventHandlerTServerReceived` – Listens to all user events and processes only the Callback preview user events from GMS Callback.

InvitationDialog.xaml.cs

- `PlaceDisplayData` – Loads the preview data from `userevent` into the preview dialog.
- `PlaceResponseButtons` – Creates clickable buttons to allow agents to take action after previewing the displayed Callback information.
- `ResponseButtonClick` – Returns an agent response to GMS based on the button clicked.

DispositionDialog.xaml.cs

- `PlaceDisplayData` – Loads the disposition data from `userevent` into the disposition dialog.
- `PlaceResponseButtons` – Creates clickable buttons to allow agents to send disposition after taking actions according to the displayed Callback disposition information.
- `ResponseButtonClick` – Returns an agent response to GMS based on the button clicked.

Disclaimer

Statistic Custom Template Sample

You can download the sample custom template for statistics using the following link:

Statistic Custom Template Sample

Overview

GMS enables you to easily develop your own service by using services composition and custom business logic. The sample template provided here addresses a statistic use case on how to get information about `maxInteractions`, `currentMargin`, and `currentInteraction` from a group of approximately 500 agents. The goal is to use this statistic on the customer-side (browser) to show how quickly a request has been answered.

The response from the GMS statistic will be similar to the Pulse statistic. For example:

```
{
    "maxInteractions": 2,
    "currentInteractions": 0,
    "currentMargin": 2
}
```

While the GMS Stat Service APIs allow you to query statistics against Stat Server, the APIs return the statistic response as a JSON object. For the use case described here, the response from GMS for this type of statistic may not meet business needs with the response size, CPU consumption, and security.

However, the benefits of using the Statistic custom template are:

- Network Load - Size of transmitted information is significantly reduced.
- CPU Consumption - Reduced on client-side; all parsing/formatting process is completed on the GMS/ORS-side.
- Security - Sensitive information is not exposed.

Prerequisite

GMS 8.5.005.xx and higher.

Running the Sample

1. Download the zip file. You can save the zip file to the same directory as the templates that are included with the GMS installation: `<GMS installation directory>/service_templates`, or you can create a

new directory to store your custom templates.

2. Go to the GMS Service Management UI > Tools > Service Templates.
3. Load the template:
   1. Click `Add Service Template`.
   2. Select the `get-gastat.zip` file.
4. Create the service:
   1. Go to the GMS Service Management UI > Services > Configured Services.
   2. Click `Add Service`.
   3. Select the `get-gastat` service, and enter a name for your service.
5. You can now configure the custom statistic service.
6. When you have finished with the configuration, you can call your new service using a REST client.

About the Code

The template is a zip file that contains two files:

- `get-gastat.json` - Contains the configuration options associated with the service (service type, statistic details, and so on).
- `get-gastat.scxml` - Contains the logic of the service. In this sample, it contains a call to the GMS Stat Service API as well as the parsing and formatting logic.

Disclaimer