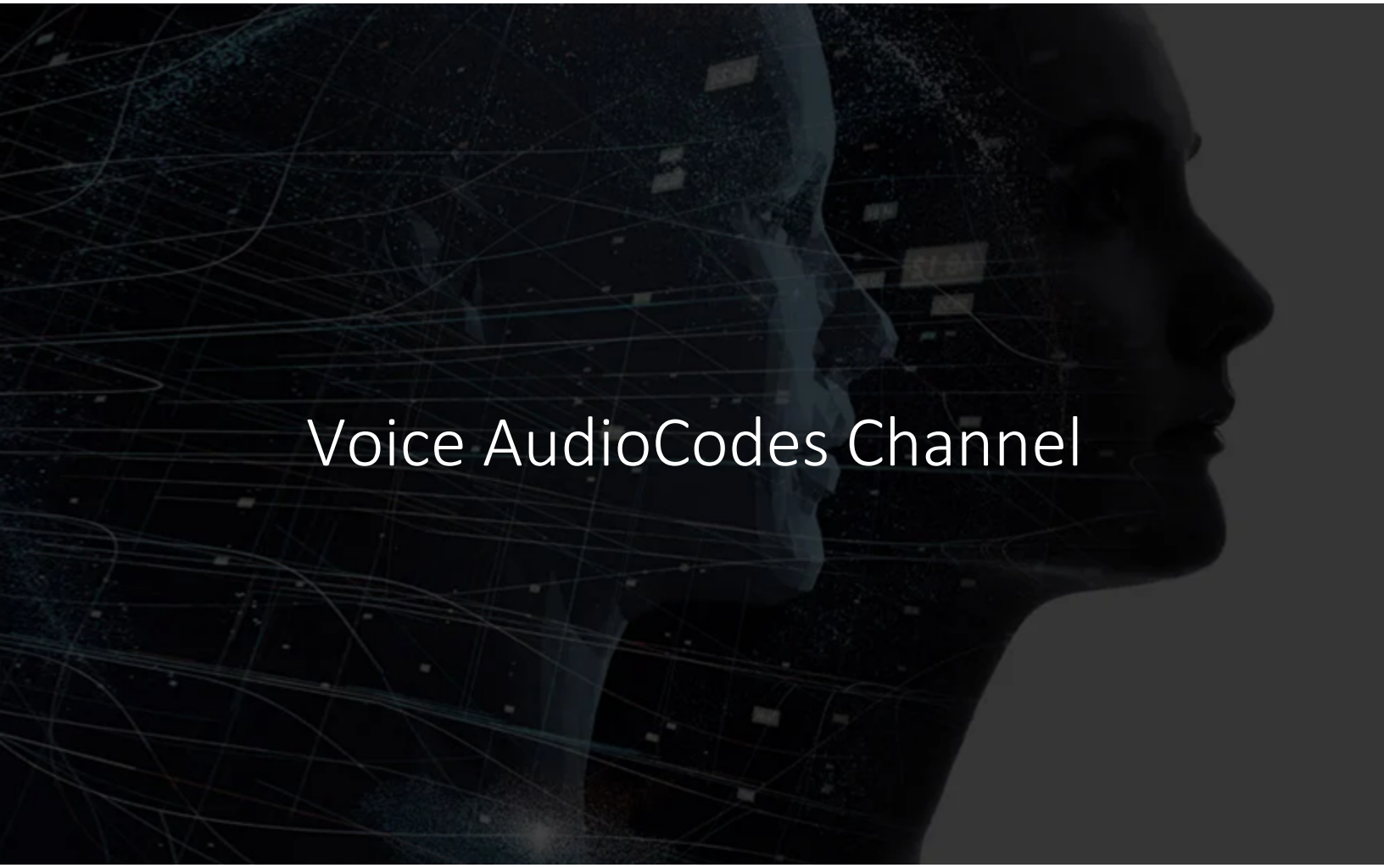


Voice AudioCodes Channel

Last update: February 02, 2023

CONTENTS

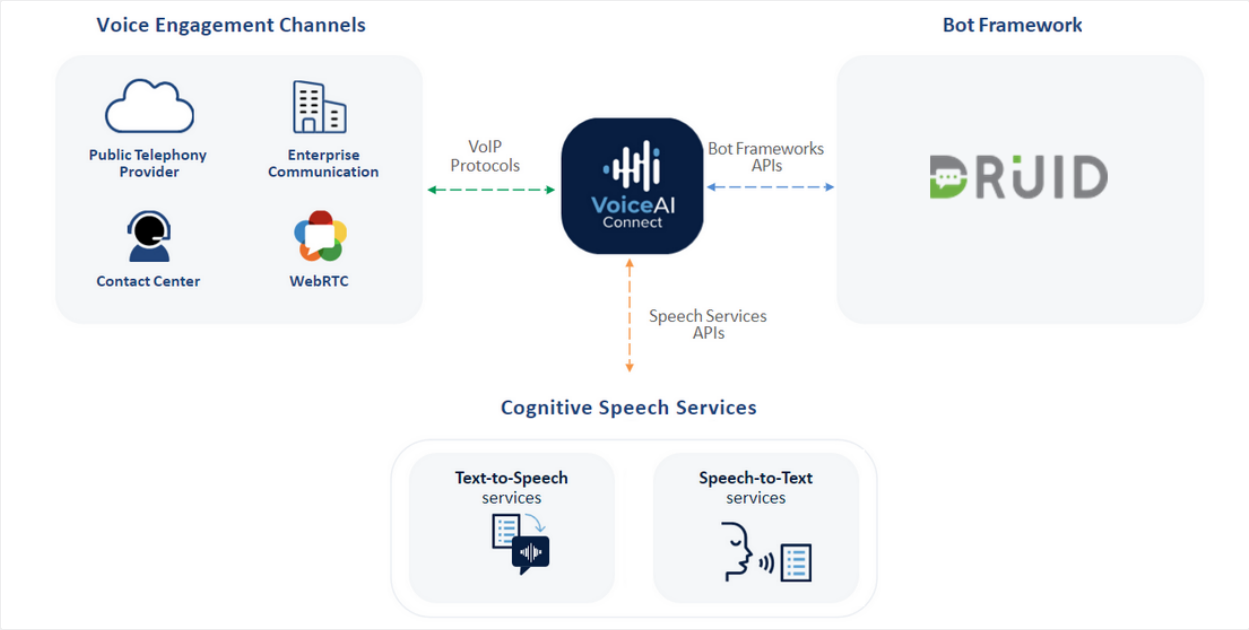
Voice AudioCodes Channel

Voice AudioCodes Channel

> **Note:** This channel is in technology preview as a tenant feature in DRUID version 5.7 and higher.

The Voice channel through VoiceAI Connect from AudioCodes enables you to deliver a seamless voice experience to the users talking to your DRUID virtual assistants.

VoiceAI Connect acts like a hub connecting different telephony systems (telephony channel, public telephony provider, contact center, enterprise communication platform, or any platform communicating via WebRTC) to the DRUID bot framework and voice AI cognitive services.



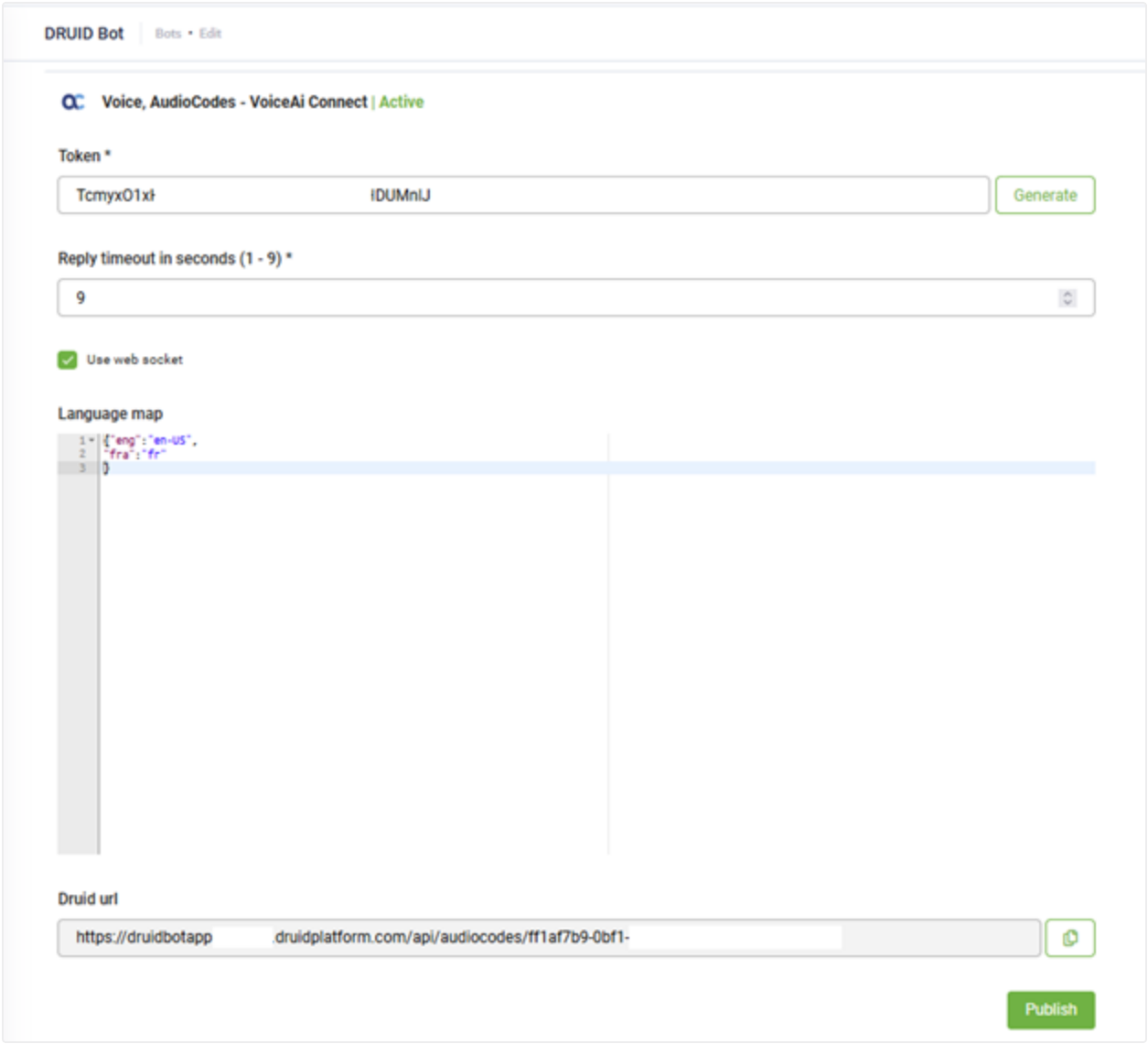
In a typical bot deployment, VoiceAI Connect receives a phone call and connects it to your bot.

To activate the channel, follow these steps:

1. In the DRUID Portal go to your bot settings. Click the **Channels** tab, then click **Voice, AudioCodes – VoiceAi Connect**. The channel info section expands.
2. Generate a token by clicking the **Generate** button.

3. By default, the communication between DRUID and VoiceAi Connect is done via the WebSocket protocol. For special deployments where network special restrictions may deny communication via WebSocket, do not use Web Socket, disable the checkbox.
4. In the **Reply timeout in seconds** field, enter the threshold the bot can respond before the call is automatically disconnected. For more information, see Handle Conversation Disconnect.
5. If the Speech-To-Text service provider uses a language code other than DRUID language code (that is, ISO 639-1), in the **Language map** JSON field, provide a one-to-one mapping between the language codes used by the Speech-To-Text (STT) service provider (key on the left) and DRUID-specific language codes (key on the right). In addition, if the STT service provider has voice only en-UK and not for en-US, you can map “en-UK” to “en-US”.

> **Note:** If the **Language map** field is empty or you provide invalid language codes, the bot default language will be used.



6. Send the **token** and **DRUID URL** to your DRUID representative and DRUID Team in partnership with AudioCodes Professional Services will set up the connection between your virtual assistant and VoiceAI Connect.

After channel's activation, the following fields are available in DRUID:

- **[[ChatUser]].ChannelId** = "audiocodes". It identifies the channel.
- **[[ChatUser]].Phone** – Stores the user's phone number.
- **[[ChatUser]].CalleePhoneNumber** – Stores the bot's phone number.

Author Flows for Voice

DRUID provides authors with a simple way to configure flows for Voice channel by providing the SpeakMessage in the Voice section on flow steps.

> **Note:** In the Voice channel, the **SpeakMessage** has priority over the step message.

Best Practices

To make the bot's voice sound more natural, follow these best practices:

- For multi-channel bots do not customize voice messages per channel, instead check the steps message spelling and punctuation marks and correct the mistakes if any. Also properly use diacritics and accented characters.
- Provide short messages in the SpeakMessage field on steps.
- Use properly the punctuation marks. This way you make it easier for the bot to read using pauses and the pitch of the voice to make the message clear.
- For hero, thumbnail and choice steps, if you want the bot to speak what's in the cards, buttons, etc., provide the desired voice message in the **SpeakMessage** field on these steps.

Capture a Collection of Dual Tone Multi Frequency (DTMF) Digits

You can configure flow steps to capture a collection of digits the user presses at the phone's keypad, either within a specific time frame between digits, or a maximum number of digits or the digits pressed prior a specific digit set on the flow step.

To capture digits, on the flow step, click the **Metadata** section, tap on **Advanced Editing** and in the JSON field add the "audioCodesDTMF" object and set the parameters described in the table below.

Parameter	Type	Description	Mandatory
sendDTMF	Boolean	To capture a collection of digits, set the parameter **sendDTMF** to **false**; otherwise, the bot will capture only the first digit pressed by the user on the phone's keypad.	Yes
bargeInOnDTMF	Boolean	When set to true, allows the user to interrupt the bot by pressing a DTMF digit which terminates the bot response. **Note:** To prevent users from interrupting the bot, we strongly recommend you to set the parameter **bargeInOnDTMF** to false.	No
dtmfCollect	Boolean	Set this parameter to true to capture all the DTMF digits entered by the user. Default value is false; that is, the bot captures only the first digit pressed by the user.	Yes

Parameter	Type	Description	Mandatory
dtmfCollectInterDigitTimeoutMS	Number	The timeout in milliseconds the bot waits for the user to press another digit before it captures the digits. The timeout is triggered after the user enters the first DTMF digit and is reset after each digit. The default value is **2000ms**. **Note:** » The parameter is applicable only when the **dtmfCollect** parameter is set to **true**. » If the timeout is reached, the parameters **dtmfCollectMaxDigits** and **dtmfCollectSubmitDigit** are disregarded.	Yes*

Parameter	Type	Description	Mandatory
dtmfCollectMaxDigits	Number	The maximum number of DTMF digits the user is expected to press on the phone's keypad. The default is **5**. **Note:** » The parameter is applicable only when the **dtmfCollect** parameter is configured to true. » If the timeout is reached, the parameters **dtmfCollectInterDigitTimeoutMS** and **dtmfCollectSubmitDigit** are disregarded. » If **dtmfCollectMaxDigits** is set to **0**, digits are captured based on the parameters **dtmfCollectInterDigitTimeoutMS** and **dtmfCollectSubmitDigit**.	Yes*

Parameter	Type	Description	Mandatory
dtmfCollectSubmitDigit	String	Defines a special DTMF "submit" digit that when received from the user, the bot captures the digits pressed before it without waiting for the timeout to expire or for the maximum number of expected digits. The valid value is any symbol on a phone keypad. The default is **#** (pound key). **Note:** The parameter is applicable only when the dtmfCollect parameter is configured to **true**.	Yes*

*The parameter controls how the bot captures the digits. You can use these parameters in any combination, but at least one is mandatory.

Example: Capture the user's CNP on a prompt step

This section describes how to configure a prompt step to capture the CNP provided by the users prior to pressing **#** (pound key) on their phone keypad.

Enter [[Account]].ClientCNP in Input mapping on the step.

On the prompt step **Metadata** section, click **Advanced editing** and in the JSON editor add the following code:

```
"audioCodesDTMF":{

"sendDTMF":false,

"bargeInOnDTMF":false,

"dtmfCollect":true,
```

```
"dtmfCollectSubmitDigit":"#"

}
```

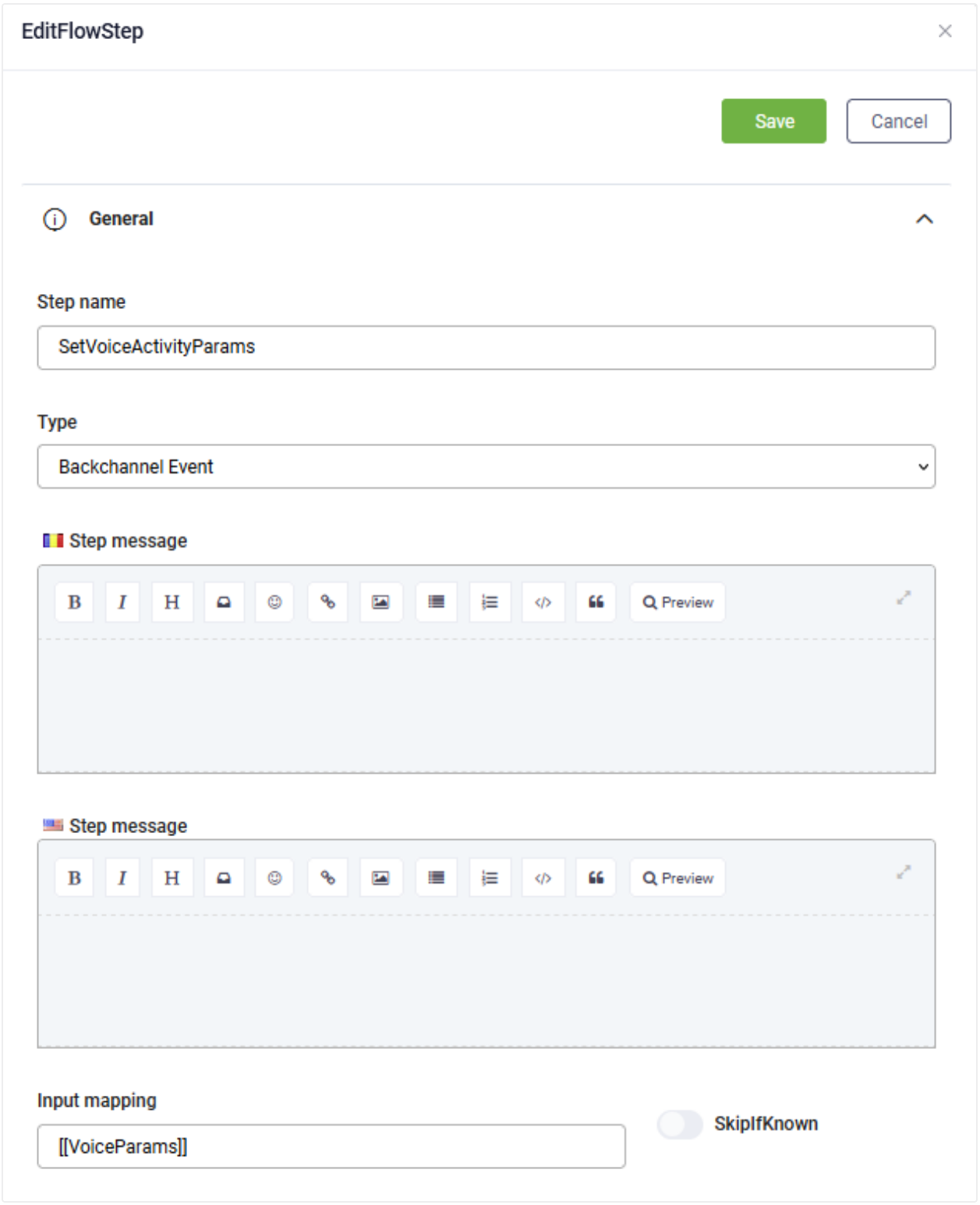
Language Change Detection

To provide users with a flawless conversation and avoid interrupting the conversation due to language change, DRUID takes for granted the language identified by the STT language detection. For situations when you want to trigger the Language changed flow), you can activate DRUID language detection on the bot.

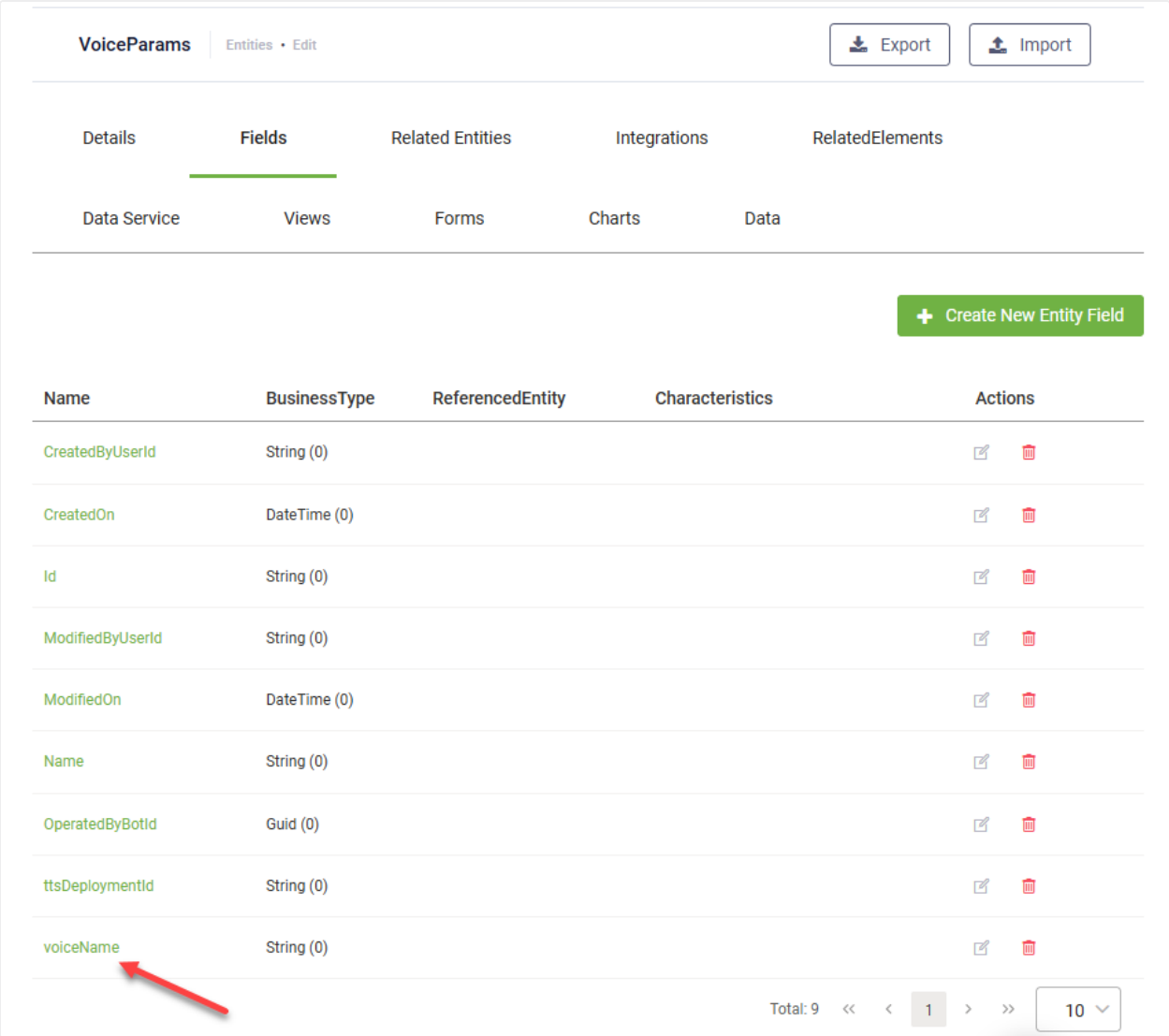
On language change on the bot, you can do extensive speech customizations on two levels, as follows:

- On conversation activity (for each conversation). Add a backchannel step called **SetVoiceActivityParams**.
- On session level (for the entire conversation). In this regards, add a backchannel step called **SetVoiceSessionParams**. You can also use this backchannel step to handle bot delays. For more information, see Handling Bot Delay.

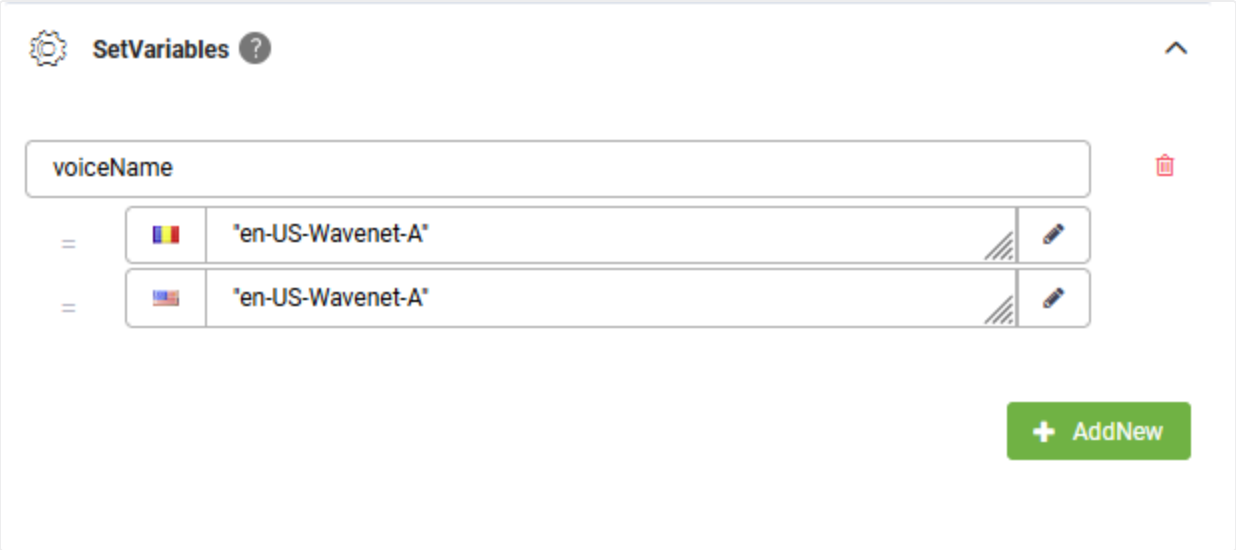
Based on what you want to achieve, configure the specific backchannel step (**SetVoiceActivityParams** or **SetVoiceSessionParams**) as follows: in **Input mapping** provide the entity which stores the speech related data.


EditFlowStep
Save
Cancel
General
Step name
SetVoiceActivityParams
Type
Backchannel Event
Step message
Preview
Step message
Preview
Input mapping
[[VoiceParams]]
SkipIfKnown

Make sure that the entity you provide in Input mapping contains fields named exactly as the parameters expected by AudioCodes' VoiceAI Connect. For the complete list of parameters, see VoiceAI Connect documentation.



In the **SetVariables** section of the backchannel step, set value(s) for the desired speech fields the bot will send to VoiceAI Connect.



Handling Bot Delay

Handling bot delays is particularly useful when the bot executes an integration which might take longer to complete.

By setting timeouts, you can configure the following actions to address situations when the bot takes time to respond to a message sent to it:

- Play a textual prompt to the user
- Play an audio file to the user
- Disconnect the call
- Resume speech recognition (so the call will not remain hanging).

To handle not delay, add a backchannel step called **SetVoiceSessionParam**s. In **Input mapping** provide the entity which stores the timeouts and actions you want to address.

Make sure that the entity you provide in Input mapping contains fields named exactly as the parameters expected by VoiceAI Connect. For the complete list of parameters, see VoiceAI Connect documentation.

In the **SetVariables** section of the backchannel step, configure the timeouts and define the actions based on your needs.

Handle Conversation Disconnect

A call disconnects if the bot does not respond within the **Reply timeout in seconds** threshold set on the channel or if the user says nothing for 120 seconds.

You can configure what happens on conversation disconnect. Go to the bot details, click the **Dialogue management** section header and from the **Voice call terminate flow** field, select the flow to be triggered on disconnect. If no such flow is set, the call disconnects.

When the call disconnects, the following data is logged in the conversation context:

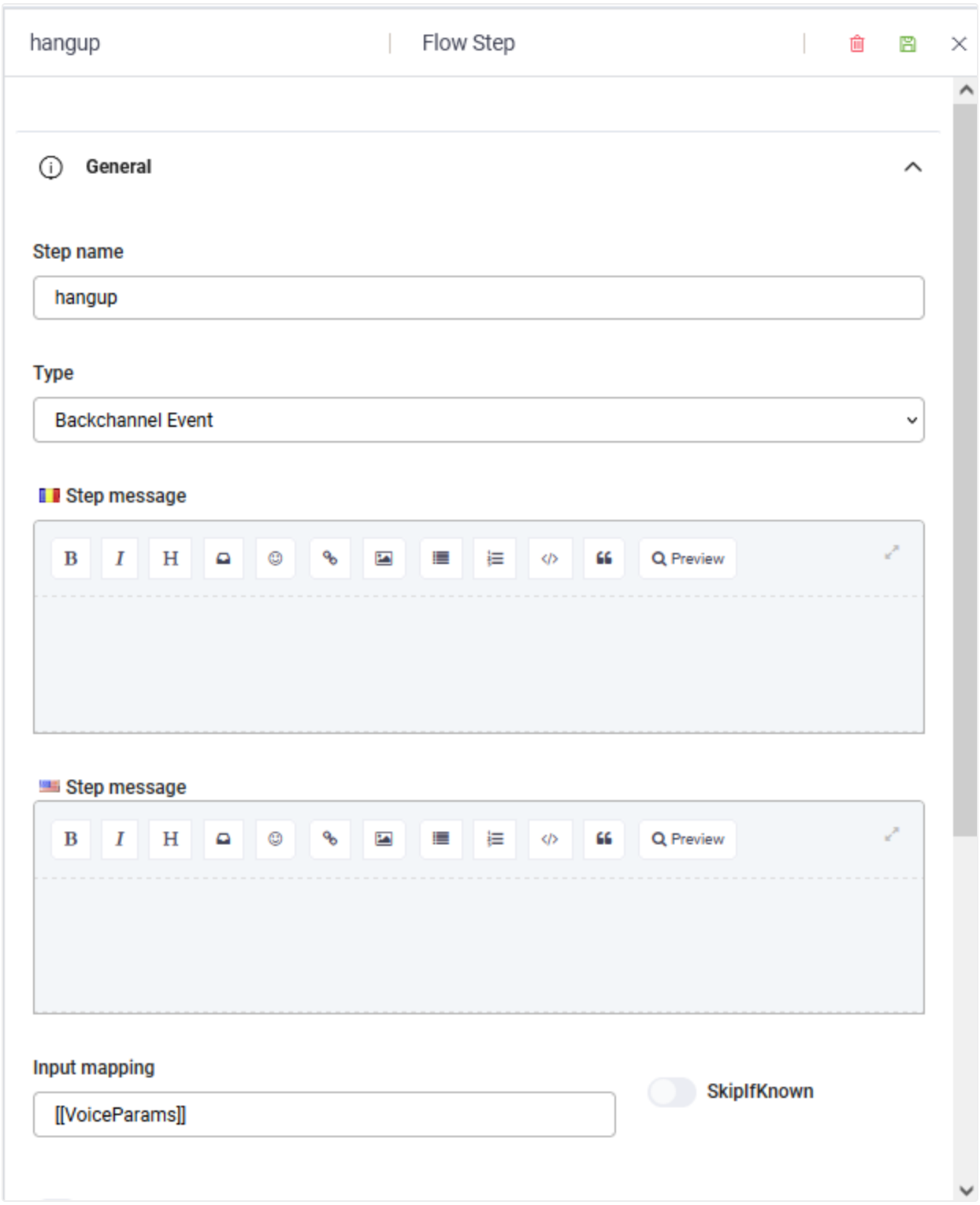
- **[[ChatUser]].VoiceConversationTerminatedReason** - The reason for which the call is disconnected. E.g., “Client Side”. The reason of the disconnect can be one of the following:
  - **SocketInterrupted** – The connection was interrupted.
  - **UserBecameSilent** – The user said nothing for 120 seconds.
  - **CallTerminatedByCaller** – The user terminated the call.
- **[[ChatUser]].VoiceConversationTerminatedReasonCode** – The code (text) associated to the disconnect reason. E.g., “client-disconnected”.

Disconnecting the call

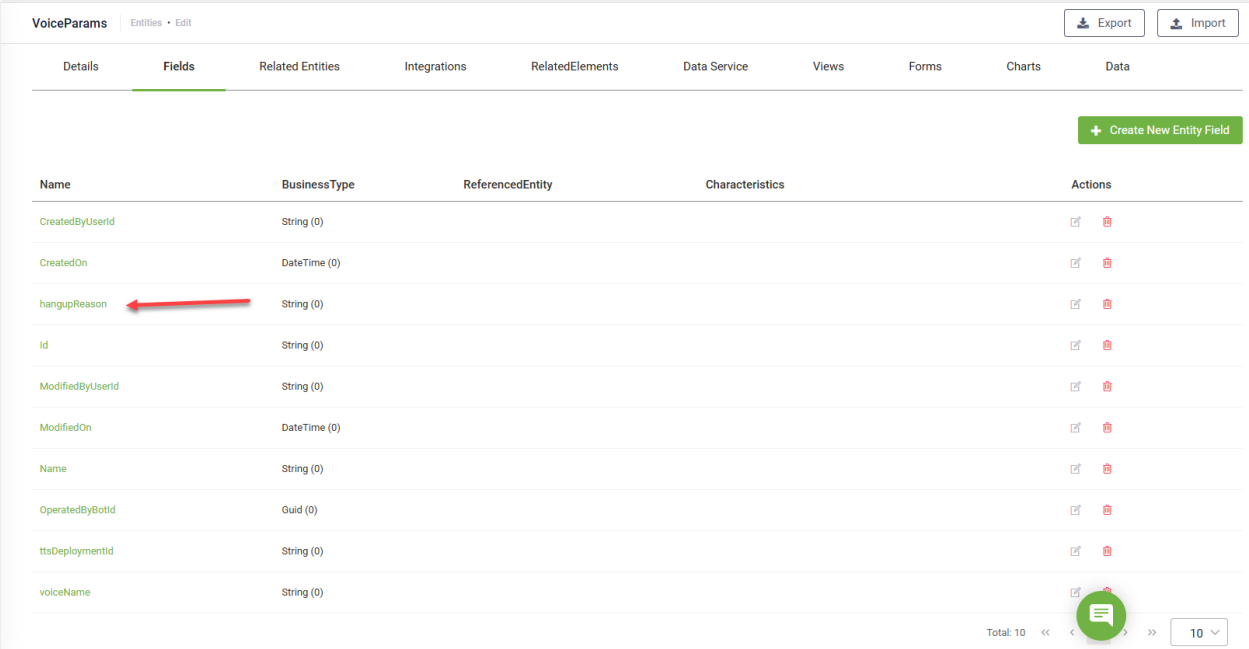
At any stage of the conversation, the bot can disconnect the conversation. For the bot to disconnect the call, add a backchannel step named **hangup**.

You can configure the backchannel step so that upon disconnect the bot provides a textual reason that will be passed to the peer on the SIP Reason header and will appear in the CDR of the call. In addition, you can also configure the backchannel step to add SIP headers and their values, which will be included in the SIP BYE message.

To add the disconnect reason and additional SIP headers, on the **hangup** backchannel step, in Input mapping provide the entity which stores the desired values.


hangup
Flow Step
General
Step name
hangup
Type
Backchannel Event
Step message
Preview
Step message
Preview
Input mapping
[[VoiceParams]]
SkipIfKnown

Make sure that the entity you provide in **Input mapping** contains fields named exactly as the parameters expected by VoiceAI Connect. For the complete list of parameters, see VoiceAI Connect documentation.

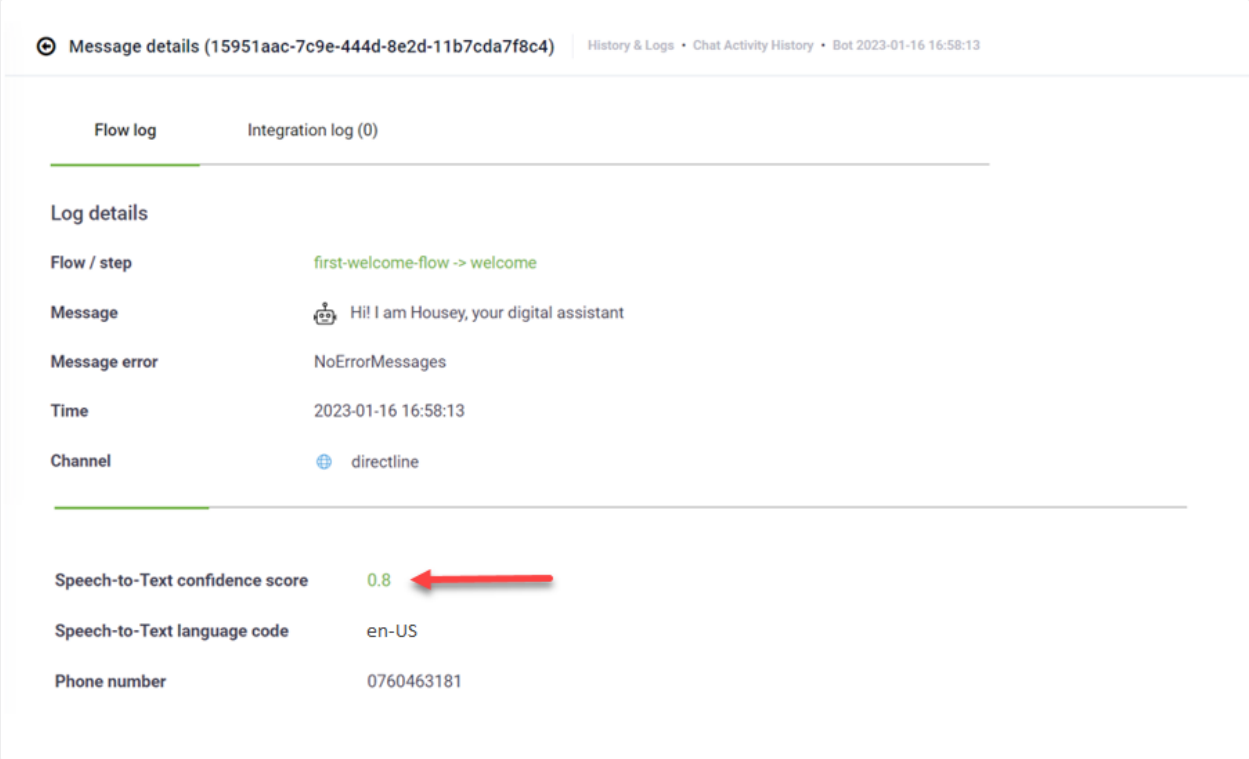


In the **SetVariables** section of the **hangup** backchannel step, set the disconnect reason and/or additional SIP headers.

Conversation History

All voice conversations begin with “[Voice start event]”. This is particularly useful for debugging purposes to measure the time from the moment when the call was initiated (bot picks up the call) until the bot says the first message.

For this channel, DRUID also logs in the Conversation History the **Speech-to-Text Confidence Score**, that is, the value representing the confidence level of the recognition received from the speech-to-text provider.



When the call disconnects due to an error, the disconnect reason logged in the Conversation History is “PlatformIntegrationError” (the message status is Platform Integration error).

Store call initiation metadata in [[QueryParams]] for future usage

By default, VoiceAI Connect sends an initial event to the bot when the call is initiated together with specific SIP headers. By default, **[[ChatUser]].Phone** stores the user’s phone number and **[[ChatUser]].CalleePhoneNumber** stores the bot’s phone number.

Storing incoming custom SIP headers (metadata sent by the contact center solution together with initiation call event) can be particularly useful for by companies when running outbound dialing campaigns, whereby the dialer automatically initiates calls with potential customers.

You can store additional incoming custom SIP headers in dedicated fields in the **[[QueryParams]]** system entity. For that, you need to create dedicated field that have the same name as the incoming SIP header key.

For more information about sensing SIP headers to the bot, see VoiceAI Connect documentation.