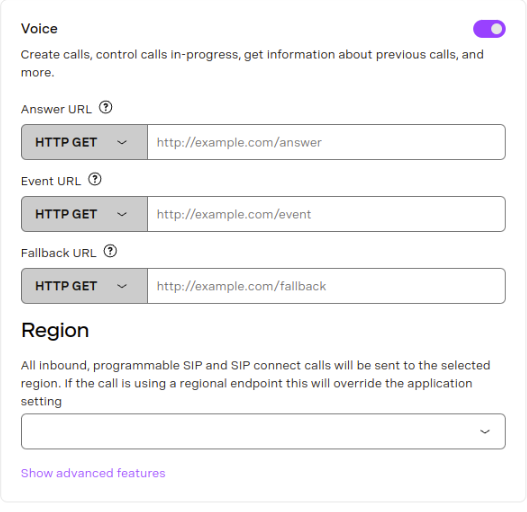
**Vonage Voice API**

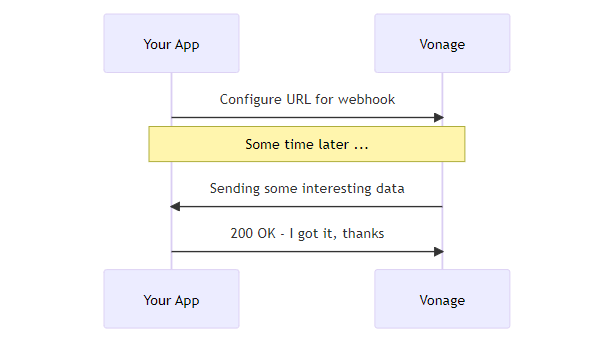
The Vonage Voice API allows you to connect people around the world and automate voice interactions that deliver a frictionless extension of your brand experience using AI technologies.

1. Text to Speech with over 50+ languages
2. Create IVRs and Voice Bots
3. Speech to text and WebSockets
4. Embed calls in web and mobile apps
5. Web & Mobile (iOS, Android) SDKs
6. Record and store inbound or outbound calls

**Enable the Voice capability**

****

**Voice API Webhooks**

****

Webhooks provide a convenient mechanism for Vonage to send information to your application for events such as an incoming call or message, or a change in call status.

**Webhooks**

* Answer Webhook
* Event Webhook
* Fallback URL

**Answer Webhook**

* Answer webhook is sent when a call is answered. This is for both incoming and outgoing calls.
* When an incoming call is answered, an HTTP request is sent to the answer\_url.

**Answer webhook data fields**

|  |  |
| --- | --- |
| **Fields** | **Description** |
| to | The number that answered the call. |
| from | The number that called **to.** This could be a landline or mobile number |
| From\_user | The username that called **to** only if the call was made using the client SDK. |
| uuid | Unique identifier for this call. |
| Conversation\_uuid | Unique identifier for this conversation |
| Region\_url | Regional API endpoint which should be used to control the call with **REST API**. |
| Custom\_data | Custom data object, optionally passed as parameter. |

**Event webhook**

* Event webhook is sent for all the events that occur during a call. Your application can log, react to or ignore each event type.
* HTTP requests will arrive at the event webhook endpoint when there is any status change for a call.
* By default the incoming requests are POST requests with a JSON body.

**Event webhook data fields**

|  |  |
| --- | --- |
| **Fields** | **Descriptions** |
| from | The number the call came from |
| to | The number the call was made to |
| uuid | Unique identifier this call |
| conversation\_uuid | Unique identifier this conversation |
| status | Call status |
| direction | Call direction |
| timestamp | Timestamp (ISO format) |

**Status Types**

Started

Ringing

Answered

Busy

Cancelled

Unanswered

Disconnect

Rejection

Failed

Timeout

Human / machine

Completed

Record

Input

Transfer

**Direction Type**

Inbound

Outbound

**Fallback URL**

Fallback URL is used when either the Answer or Event webhook fails or returns an HTTP error status.

**example**

{

  "reason": "Connection closed.",

  "original\_request": {

    "url": "https://api.example.com/webhooks/event",

    "type": "event"

  }

}

**NCCO - Nexmo Call Control Objects**

A Call Control Object (NCCO) is represented by a JSON array. You can use it to control the flow of a Voice API call.

The Call event model is asynchronous.  When a Call is placed to your number, Vonage makes a synchronous request to the webhook endpoint you set as the answer\_url for your number and retrieves the NCCO object that controls the Call.

**NCCO instruction are:**

* Action - something to be done in the Call.
* Option - how to customize an *action*.
* Type - describes an *option*. For example, type=phone for an endpoint option.

**actions you can use in an NCCO are:**

* record - all or part of a call
* conversation - create a standard or hosted conversation
* connect - connect to a connectable endpoint such as a phone number or Vonage Business Cloud extension
* talk - send synthesized speech to a conversation
* stream - send audio files to a conversation
* input - collect digits from the person you are calling, then process them

**Creating a custom call or conversation for each user**

When you make an outbound call or accept an inbound call, Vonage makes a request to your webhook endpoint at answer\_url and retrieves your NCCO. This request contains the following parameters:

|  |  |
| --- | --- |
| **Name** | **Description** |
| to | The endpoint being called. |
| from | The endpoint you are calling from. |
| conversation\_uuid | Unique ID for this conversation |
| uuid | Unique ID for this call |

**Code examples show how to provide the NCCO that controls your call or conversation**

from flask import Flask, request, jsonify

app = Flask(\_\_name\_\_)

HOST = "localhost"

PORT = 3000

@app.route("/webhooks/answer")

def answer\_call():

    call\_from = request.args['from']

*# Dynamically build NCCO based on source phone*

    if call\_from == "447700900000":

        ncco = [{

                "action": "talk",

                "text": "Hi John, we will be with you shortly."

            }]

    elif call\_from == "447700900001":

        ncco = [{

                "action": "talk",

                "text": "Hi Jane, we will be with you shortly."

            }]

    else:

            ncco = [{

                "action": "talk",

                "text": "Hello, sorry, we do not recognize your number."

            }]

    return jsonify(ncco)

if \_\_name\_\_ == 'main':

    app.run(*host*=HOST, *port*=PORT

**Web Sockets**

WebSockets is a computer communications protocol that enables two-way communication over a single, persistent TCP connection without the overhead of the HTTP request/response model.

Using Vonage’s Voice API, you can connect phone calls to WebSocket endpoints. This means that any application that hosts a WebSocket server can be a participant in a Vonage voice conversation. It can receive raw audio from and play audio into the call in real time.

**Automating calls with bots to perform tasks such as food ordering or requesting information from field experts.**

**Working with WebSockets**

when establishing the WebSocket connection.

* Return an NCCO instructing Vonage to connect to your WebSocket endpoint
* Accept this WebSocket connection
* Handle JSON text-based protocol messages
* Handle mixed call audio binary messages

**Connecting to a WebSocket**

* Vonage to connect to a WebSocket your application server must return an NCCO when requested from your Vonage Application's answer\_url
* NCCO must contain a connect action with an endpoint.type of websocket

Example

[

    {

       "action": "connect",

       "endpoint": [

           {

                "type": "websocket",

                "uri": "wss://example.com/socket",

                "content-type": "audio/l16;rate=16000",

                "headers": {

                    "language": "en-GB",

                    "caller-id": "447700900123"

                }

           }

       ]

     }

]

The specific data fields for webhooks

|  |  |
| --- | --- |
| **Fields** | **Descriptions** |
| uri | Endpoints of your websocket server that vonage will connect to |
| Content-type | String representing the audio sampling rate, **audio/l16;rate=16000** or **audio/l16;rate=8000** and **audio at 8kHz**. |
| headers | additional optional properties to send to your Websocket server |

**Handling incoming WebSocket messages**

The initial message sent on an established WebSocket connection will be text-based and contain a JSON payload, it will have the event field set to websocket:connected and detail the audio format in content-type, along with any other metadata that you have put in the headers property of the WebSocket endpoint in your NCCO connect action.

**example**

{

    "event":"websocket:connected",

    "content-type":"audio/l16;rate=16000",

    "prop1": "value1",

    "prop2": "value2"

}

**Binary audio messages**

Messages that are binary represent the audio of the call. The audio codec presently supported on the WebSocket interface is Linear PCM 16-bit, with either a 8kHz or a 16kHz sample rate, and a 20ms frame size.

To choose the sampling rate set the Content-Type property to audio/l16;rate=16000 or audio/l16;rate=8000 depending on if you need the data at 16kHz or 8kHz.

|  |  |  |
| --- | --- | --- |
| **Sampling rate** | **Number of samples in 20ms** | **Bytes per message** |
| 8000 | 160 | 160 \* 2 = 320 |
| 16000 | 320 | 320 \* 2 = 640 |

**Call Flow**

**Try the API**

The primary way that you'll interact with the Vonage API voice platform is via the [public API](https://developer.vonage.com/en/api/voice). To place an outbound call, you make a POST request to <https://api.nexmo.com/v1/calls>

|  |  |
| --- | --- |
| **Key** | **Description** |
| **VONAGE\_NUMBER** | Your Vonage number that the call will be made from. |
| **TO\_NUMBER** | The number you would like to call to in E.164 format. |

**Python:**

client = vonage.Client(

*application\_id*=VONAGE\_APPLICATION\_ID,

*private\_key*=VONAGE\_APPLICATION\_PRIVATE\_KEY\_PATH,

)

response = client.voice.create\_call({

  'to': [{'type': 'phone', 'number': TO\_NUMBER}],

  'from': {'type': 'phone', 'number': FROM\_NUMBER},

  'answer\_url': ['https://raw.githubusercontent.com/nexmo-community/ncco-examples/gh-pages/text-to-speech.json']

})

pprint(response)