Introduction to Nexmo Voice API

Introduction to Nexmo Voice API

Use the Voice API to make and receive calls, play audio, send and receive DTMF tones, and to record calls.

Workshop plan:

- Introduce concepts and vocabulary (mostly talking)
- Make and receive calls (hands-on)
- Interact with user input (hands-on)

NCCO: Nexmo Call Control Object

NCCOs describe the flow of the call. They are a series of stepsi described in JSON, such as this example showing text-to-speech:

```
[
  {
    "action": "talk",
    "text": "You are listening to a Call made with Voice API"
  }
]
```

You can find a full reference here: https://developer.nexmo.com/voice/voice-api/ncco-reference

NCCO: Nexmo Call Control Object

Elements in an NCCO may include:

- text-to-speech
- playing audio (optionally looping)
- recording a call
- accepting DTMF input
- transferring a call (to a conversation, or a new NCCO)
- ... and much more

Calls vs Conferences

There are two types of conversation that you might use:

- A "call" is a temporary conversation that only exists for as long as the call is taking place
- A "conference" is a conversation with a name, that additional callers can be added to. This type of conversation persists and can be reused.

```
{
  "action": "conversation",
  "name": "nexmo-conference-standard",
  "record": "true"
}
```

The Voice API

Make an API call to:

- make an outgoing call (we'll do this in a bit)
- hang up a call
- transfer a call
- interact with an in-progress call
- get information about current and past calls

The Voice API

The Voice API is an HTTP API so you can access it in many different ways:

- Explore the API with Postman or your favorite HTTP client
- Use `request(s)` or whichever library you prefer in your application
- Try one of our Server SDKS: https://developer.nexmo.com/tools (recommended)

You will find lots of code examples and the API reference on https://developer.nexmo.com

NCCO + API = Many Good Things

Combining the NCCOs to control program flow and the API calls to react to events allows us to create interesting and fully-featured applications.

Voice API Examples

- IVR
  - Incoming call, serve NCCO to answer it
  - Prompt user for DTMF input
  - DTMF input arrives as a webhook, return a new NCCO
- Proxy
  - Incoming call, serve NCCO to answer it
  - Put user into conference
  - API call to place outgoing call to other user, with NCCO to join same conference

Voice Webhooks

- Webhooks are events over HTTP
- Nexmo sends information about events and changes in call state as they happen
- These events are webhooks: incoming HTTP requests
- Your application needs to be able to receive requests and respond

The URL is set up in advance, as part of the application configuration

Voice Webhooks

Webhooks can be expected:

- When the call is answered, an HTTP request to your application's `answer_url`
- When events such as "ringing", "answered", "completed" occur, HTTP requests to your application's `event_url`
- When a user enters digits during an `input` action, an HTTP request to the URL specified in the NCCO
- When a recording is completed and available, an HTTP request to the specified `recording_url`
- When a `notify` action occurs

Webhooks on Dev Platforms

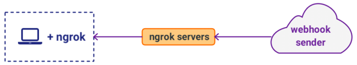
https://ngrok.com/ - secure tunnel to your dev platform

Use this tool to:

- webhook into code running locally
- inspect the request and response of the webhook
- replay requests and see the responses

Ngrok for Testing Webhooks

Start the tunnel on your laptop: receive a public URL



We have a blog post about this: https://www.nexmo.com/blog/2017/07/04/local-development-nexmo-ngrok-tunnel-dr

The Answer Webhook

When someone calls your Nexmo number, you get a webhook like this:

```
{
  "from": "442079460000",
  "to": "447700900000",
  "uuid": "aaaaaaaa-bbbb-cccc-dddd-0123456789ab",
  "conversation_uuid": "CON-aaaaaaaa-bbbb-cccc-dddd-0123456789ab"
}
```

Your code must return a valid NCCO

The Event Webhook

Many different events can produce webhooks to the `event_url`:

- Changes in call state e.g. "ringing"/"answered"
- `record` and `input` actions can specify a URL, which may be the same as the event URL
- Errors will also be sent to the `event_url`

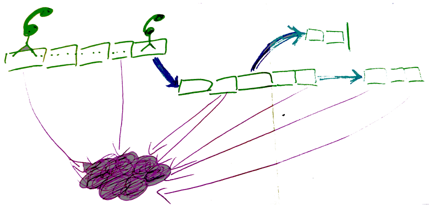
Detailed reference: https://developer.nexmo.com/voice/voice-api/webhook-reference#event-webhook

Voice Events Logger

A tool you can use to direct your `event_url` to, it just acknowledges the webhook and displays what arrived.

https://github.com/Nexmo/voice-event-logger - it can be run locally or deployed to Heroku

Pieces of the Voice API



Further Reading

- Use Cases for Voice API: https://developer.nexmo.com/voice/voice-api/use-cases/