# Sestek

Text To Speech
( TTS )
Service

REST API
Version 11.2

Developer Documentation

# Contents

# Introduction

Welcome to the Sestek Text To Speech (TTS) Service REST API!

You can use TTS Service REST API to  make  Speech Synthesis (SR).

Simply Speech Synthesis is  the artificial production of human speech. A text-to-speech (TTS) system converts normal language text into speech.

# Text To Speech( TTS )Service

## Service Description

At this service, you can make TTS.

In order to make TTS you chose Voice and Audio Format you want to get. Then send those information to service with the text you want to make TTS.The service return audio file as raw bytes in response body at the format you specift at your request.

Supported audio file formats : wav,opus,mp3,flv.

For wav files 3 encoding formats are avaliable : pcm(linear pcm) , u-law, a-law

# Service Paths

## GET Voices

**Url:**

v1/speech/synthesis/voices

**Method:**

GET

**HTTP Request:**

GET v1/speech/synthesis/voices

**Summary:**

Return avaliable voices info.

**Description:**

The Voices endpoint provide information about the voices available for synthesizing speech.

The response includes total number of avaliable voices and list of avaliable voices info.

**Request:**

No arguments

**Query Parameters For Request:**

You can query on avaliable voices by culture and gender

For example :

```
voices/?language=tr-TR&gender=female
```

**Response:**

Avaliable Voices info in Json format

**Request Example:**

*GET http://acme-pc/v1/speech/synthesis/voices HTTP/1.1*

*Accept: application/json, application/xml, text/json, text/x-json, text/javascript, text/xml*

*Accept-charset: utf-8*

*User-Agent: RestSharp/105.2.3.0*

*Host: acme-pc:20000*

*Accept-Encoding: gzip, deflate*

*Connection: Keep-Alive*

**Success Response Example:**

```
{
   "voices":[
      {
         "gender":"Male",
         "name":"GVZ Craig 8k",
         "language":"en-US"

      },
      {
         "gender":"Female",
         "name":"GVZ Gul 8k_HV_Premium",
         "language":"tr-TR"
      }
   ],
   "count":2,
   "success":true,
   "errorMessage":null,
   "errorCode":null,
   "moreInfo":null
}
```

**Error Response Example:**

```
{
   "voices":[

   ],
   "count":0,
   "success":false,
   "errorMessage":"Server Can Not Do Its Job",
   "errorCode":"Internal-Server-Error",
   "moreInfo":null
}
```

**Response Fields Descriptions:**

| **Response** | | |
|---|---|---|
| | **Name** | **Description** |
| | Voices | An array of Voice objects that provides information about all available voices. |
| | Count | Number Of Avaliable Voices |
| | Success | True  => The request succeeded.<br>False => The request failed |
| | ErrorMessage | When the request failed, failure message |
| | ErrorCode | When the request failed, failure error code. [e.g Internal Server Error] |
| | MoreInfo | Any extra info about response |
| | | |

| **Voices** | | |
|---|---|---|
| | **Name** | **Description** |
| | Gender | The gender of the voice: Male , Female, Neutral |
| | Name | The name of the voice. Use this value as the voice identifier in all requests that accept a voice such as tts. |
| | Language | The language and region of the voice; for example en-US for US English. |

# POST TTS

**Url:**

v1/speech/synthesis/tts

**Method:**

POST

**HTTP Request:**

POST v1/speech/synthesis/tts

**Summary:**

Synthesizes text to spoken audio.

**Description:**

Synthesizes text to spoken audio, returning the synthesized audio  stream as an array of bytes.

**Request:** Json Format

**Request Feilds Descriptions:**

| **Request** | | |
|---|---|---|
| | **Name** | **Description** |
| | **Text** | The text to be synthesized.Plain text. |
| | **Voice** | The voice information to be used for synthesis |
| | **Audio** | Audio format information for returning audio stream. |
| | **License** | If required ,license info as dictionary. Otherwise "License":null" |
| | | |

| **Voice** | | |
|---|---|---|
| | **Name** | **Description** |
| | **Name** | The voice to be used for synthesis. |
| | **Rate** | The speaking rate of the voice |
| | **Volume** | The base volume (loudness) level of the voice. |
| | | |

<table>
<tr><th colspan="6">Audio</th></tr>
<tr><td></td><td><b>Name</b></td><td colspan="4"><b>Description</b></td></tr>
<tr><td></td><td><b>Format</b></td><td colspan="4">Audio output[The requested type of the audio] format : wav,opus,mp3,flv</td></tr>
<tr><td rowspan="6"></td><td rowspan="6"><b>FormatDetails</b></td><td colspan="4"><b>Details about audio format</b></td></tr>
<tr><td rowspan="2"><b>For<br>Wav output</b></td><td><b>Encoding</b></td><td colspan="2">Companding (occasionally called compansion) algorithm used for generating wave format " pcm", "u-law"  ve "a-law"</td></tr>
<tr><td><b>SampleRate</b></td><td colspan="2">Audio output sample rate</td></tr>
<tr><td></td><td></td><td colspan="2"></td></tr>
<tr><td rowspan="2"><b>For opus,mp3,flv Output</b></td><td><b>SampleRate</b></td><td colspan="2">Audio output sample rate</td></tr>
<tr><td><b>BitRateKbps</b></td><td colspan="2">Audio output bitrate in kbps</td></tr>
</table>

**Request Example For Wav File:**

```
{
   "Text":"Merhaba",
   "Voice":{
      "Name":"GVZ Gul 16k_HV_Premium",
      "Volume":1.0,
      "Rate":1.0
   },
   "Audio":{
      "Format":"wav",
      "FormatDetails":{
         "Encoding":"pcm",
         "SampleRate":"8000"
      }
   },
   "License":{
      "acme":"acme",
      "acme2":"acme2"
   }
}
```

**Request Example For Opus File:**

```
{
   "Text":"Merhaba",
   "Voice":{
      "Name":"GVZ Gul 16k_HV_Premium",
      "Volume":1.0,
      "Rate":1.0
   },
   "Audio":{
      "Format":"opus",
      "FormatDetails":{
         "BitRateKbps":"8",
         "SampleRate":"8000"
      }
   },
   "License":{
      "acme":"acme",
      "acme2":"acme2"
   }
}
```

**Response:**

Returns the audio stream for the specified text as an array of bytes in the specified MIME type : wav,opus,mp3,flv.

**Response Codes :**

| Response Codes | | |
|---|---|---|
| | Status | Description |
| | 200 OK | The request succeeded |
| | 500 Internal Server Error | The service experienced an internal error |
| | 400 Bad Request | Required request parameter value  is not valid or not supported. |
| | 404 Not Found | Requested voice is not found |
| | 412 Precondition Failed | Cloud License parameters are invalid or requested voice is not licensed or license credits are consumed completely |
| | 422 Unprocessable Entitiy | Bit rate or sample rate is invalid, or a 8k voice is requested instead of 16k voice |

**Success Response Example:**

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
Content-Type: audio/wav
Server: Microsoft-HTTPAPI/2.0
Date: Wed, 27 Apr 2016 08:54:13 GMT
RIFF  WAVEfmt ....
..........
```

Content type for

Synthesized audio

Synthesized audio stream as an array of bytes in

response body

**Failure Response Example:**

```
{
    "errorCode": "http-404",
    "errorMessage": "unknown voice name",
    "moreInfo": "",
    "success": false
}
```

# What is New with 11.2 ?

- Added new English female voice Emily
- Neural voices base volume, rate and pitch levels are made configurable from info.json

# What is New with 11.1 ?

- Fix: set default voice when only single voice is loaded
- Fix: in case of  no license, https Voice is not licensed error corrected to no license
- Decreased license refund duration limit to 500ms from 1000ms

## What is New with 11.0 ?

- Neural network (Sakura) voices are supported for REST and MRCP.
- Naural voices have a naming format Sestek [Name] [SampleRate] and developed in 24k sample rate.

  Sestek Gul 24k_HV_Premium
  Sestek Sinan 24k
  Sestek Oliver 24k
  Sestek Melissa 24
  Sestek Yasmin 24k
  Sestek Melissa 24k

## What is New with 10.9 ?

- HTTP Error responses are converted into json and error codes are diversified

## What is New with 10.7 ?

- Cloud Licensing Support is re-added. This feature was available in 10.4 but not available in 10.6.
- When a 8k voice is specified in REST request (such as GVZ Delal 8k), an error response is returned, the clients are expected to send requests only for 16k voices. However, they can specify and sampling rate in the request with “SampleRate” attribute.
- For ssml format <audio> tags, linear 8bit offset format is supported. The prompt files encoded with this format can be used in such requests.

## What is New with 10.6 ?

- Ukrainian (Darya) and English (Oliver) voices are added
- For ssml format, prosody tag support is added:
  <prosody rate=”fast” volume=”high”> hello world </prosody>

## What is New with 10.4 ?

- Russian (Kristina) and Hindi (Deepti) voices are added
- While using **SsmlTextContent** header field, some control character artifacts  the plain text output is removed
- Ffmpeg is used for all encoding/compression types
- Starting with this version, flv format valid sample rates to be sent in the requests are: 44100, 22050, and 11025
- For ssml requests, unescaped & character is correctly handled

## What is New with 10.1 ?

- Ssml texts can also be sent in addition to plain texts. SSML tag's are supported as in SESTEK MRPC TTS Service.

- With **SsmlTextContent** header field, it is made possible to get the plain text sent in the ssml requests. This field is in ***Base64*** format and after converting it to UTF8 string, it can be observed to have a plain text such as:

  "<speak>hello world!</speak>" to "hello world!"

## iOS Integration Details

- It is important to set "acceptableContentTypes" property of the json serializer as:
  @"audio/opus"  if opus format is used
  @"audio/wav" if wav format is used.

- Because the return type is not a json object but a byte array, ResponseSerializer should be appropriately specified and "AFHTTPResponseSerializer" should be used instead of "AFJSONResponseSerializer"
- For every request, the following fields should be added to the header:
  setValue:@"application/json; charset=UTF-8"
  forHTTPHeaderField:@"Content-Type"  and
  setValue:@"application/json" forHTTPHeaderField:@"Accept"

- To the body of TTS request, you should not add some line ending characters like "\n", You can use [strJSON dataUsingEncoding:NSUTF8StringEncoding] to add Json string to the request body.