

# AI & CHATBOT

Aula 08 – Introdução à Síntese de Fala

Prof. Henrique Ferreira

Prof. Miguel Bozer

Prof. Guilherme Aldeia

Prof. Michel Fornaciali

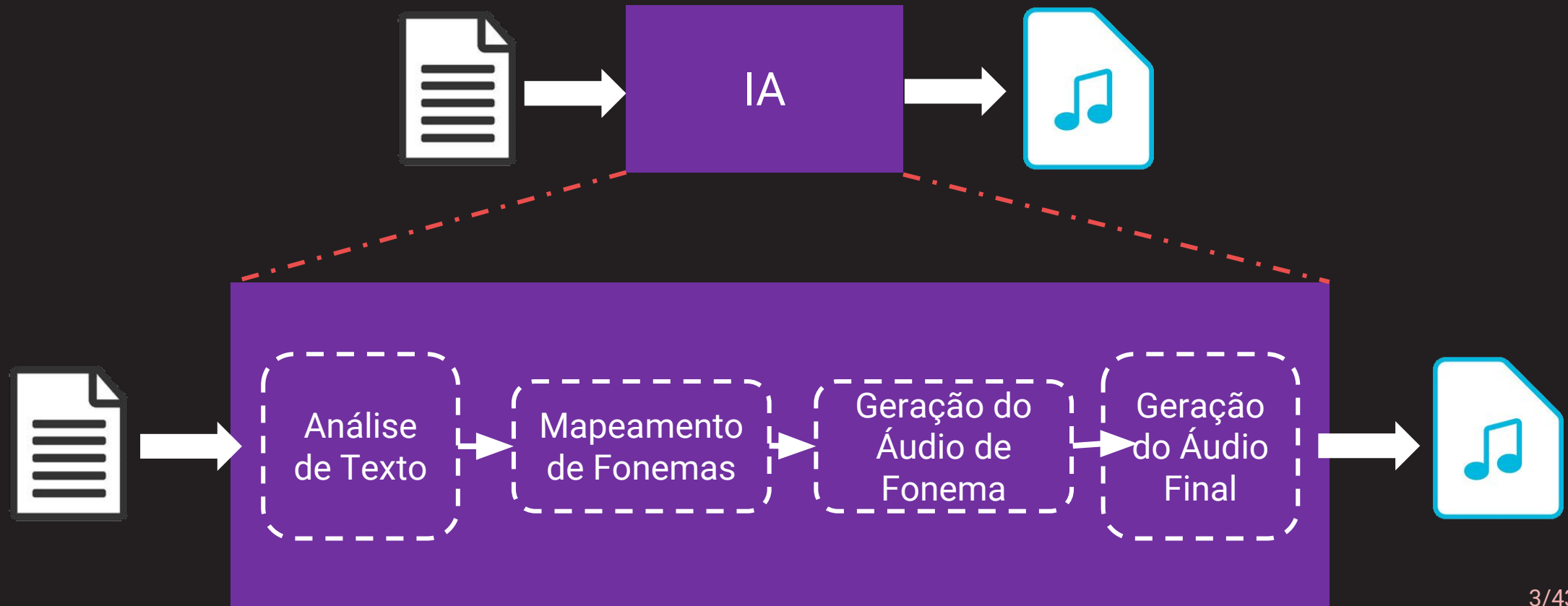
**FIAP**  
**GRADUAÇÃO**

# Síntese de Fala

Entendendo o que é a síntese de fala

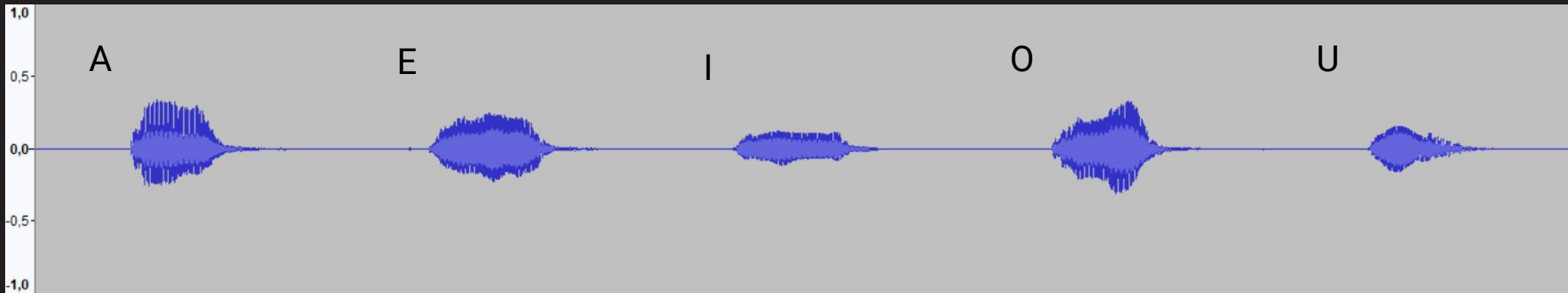
# Síntese de fala

- Na síntese de fala, o sistema recebe um texto e converte ele em um arquivo de áudio (Text-to-Speech - TTS):

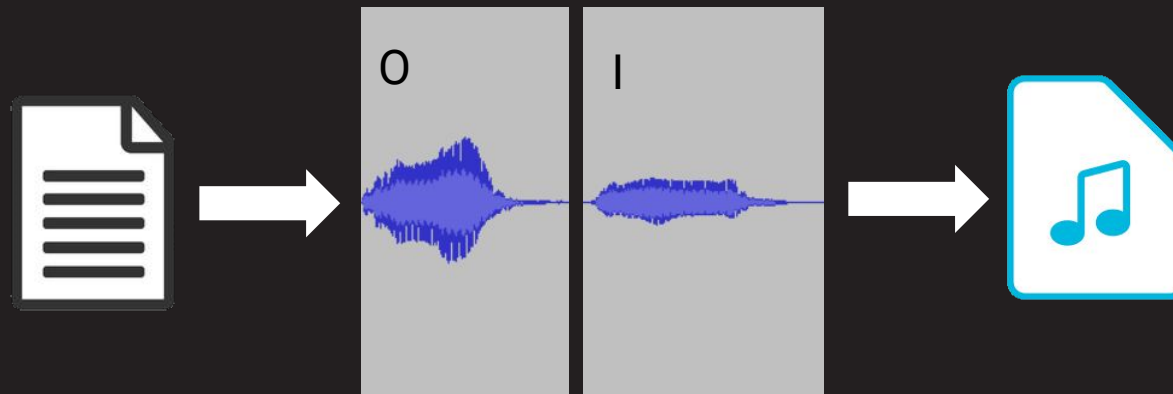


# Síntese de fala

- O efeito de voz robotizado acontece na hora de juntar o áudio de diferentes fonemas devido a falta de fluência do som;

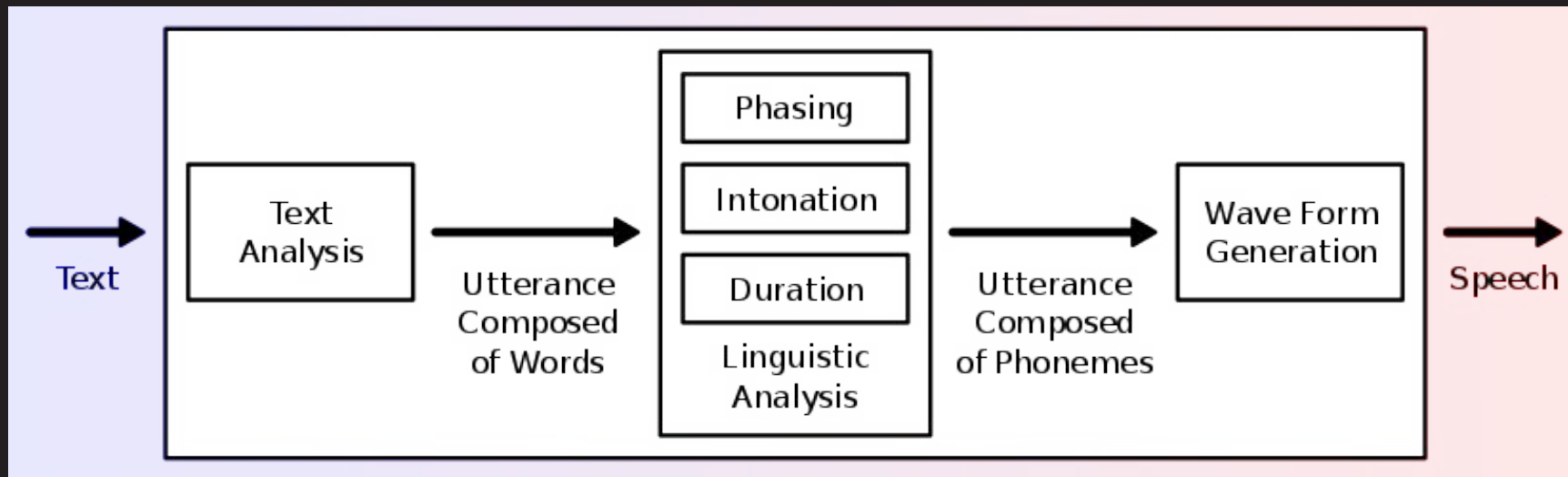


- Recebe o texto: “oi”



# Síntese de fala

- Assim como Reconhecer fala, sintetizar fala não é uma tarefa fácil e exige muito conhecimento em processamento de sinais e linguagem natural:



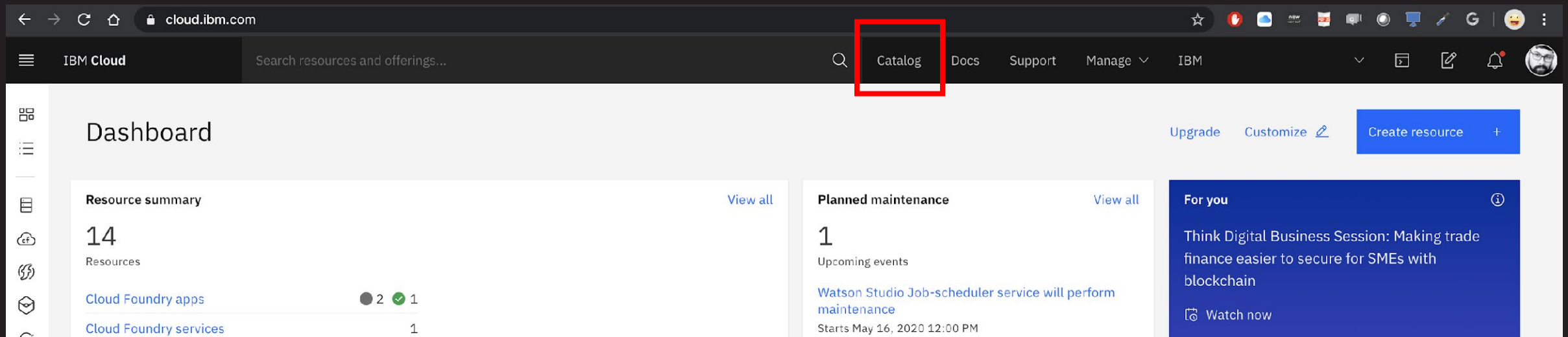
- Acesse os áudios dos primeiros sintetizados para ouvir o efeito robótico da concatenação dos áudios  
[https://en.wikipedia.org/wiki/Speech\\_synthesis](https://en.wikipedia.org/wiki/Speech_synthesis)

# Watson Text-to-Speech

Criando o serviço de TTS da IBM

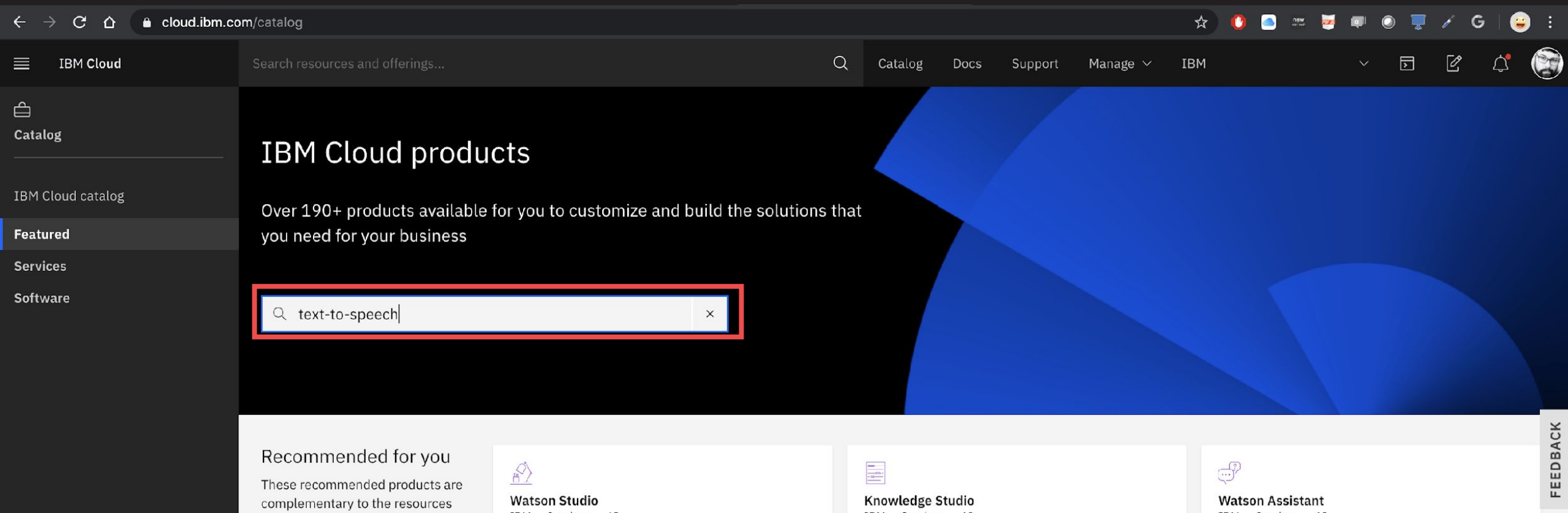
# Adicionando e criando o serviço Watson TTS

- Para pegar a chave de autorização (apikey) o primeiro passo é fazer o login em <https://cloud.ibm.com> e clicar em Catálogo (“Catalog”):



# Adicionando e criando o serviço Watson TTS

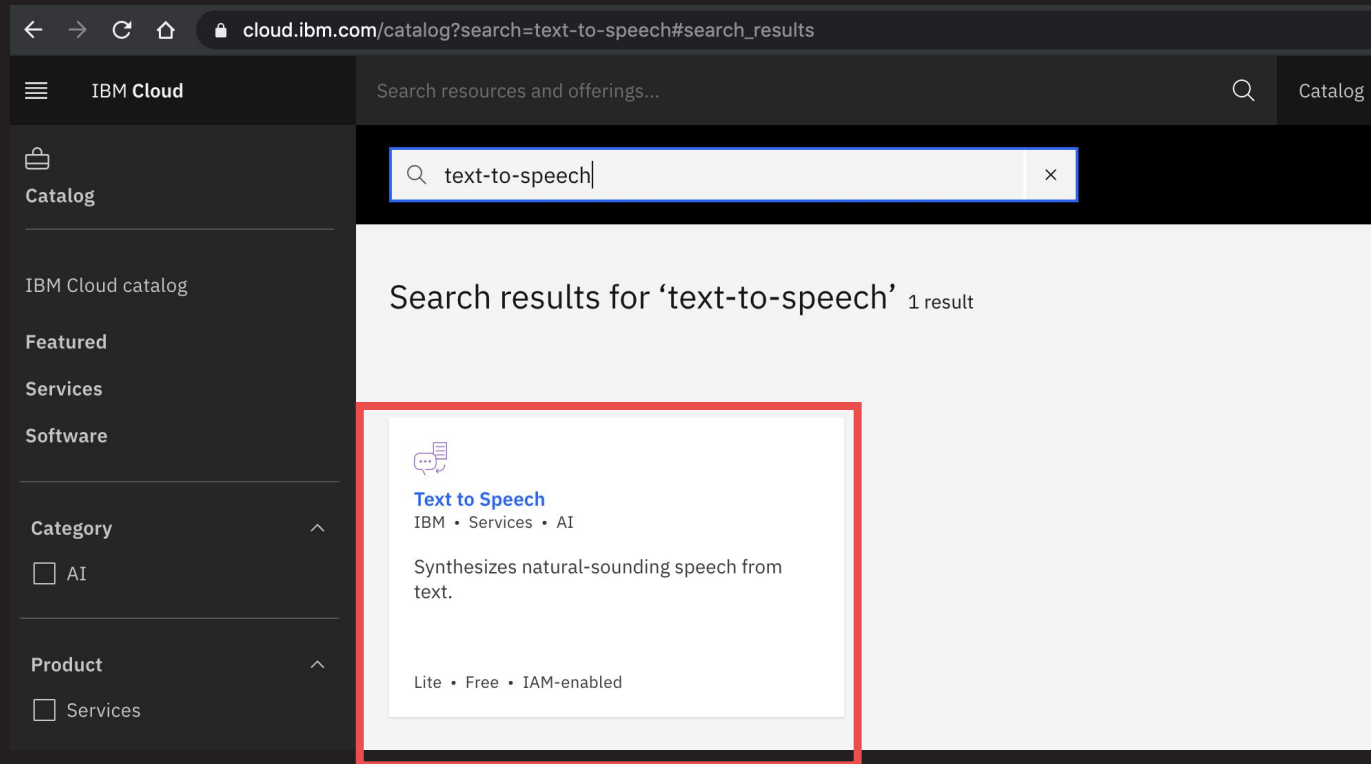
- No catalogo digite Text-to-speech:





# Adicionando e criando o serviço Watson TTS

- Seleccione o serviço do Watson Text-to-speech:



# Adicionando e criando o serviço Watson TTS

- Clique em “Create”. Não é necessária nenhuma configuração nesse momento.

IBM Cloud

Search resources and offerings...

Catalog / Services /

## Text to Speech

Author: IBM • Date of last update: 05/14/2020 • Docs • API docs

Create About

Select a region

Select a region

Dallas

Select a pricing plan

Displayed prices do not include tax. Monthly prices shown are for country or region: [United States](#)

Plan	Features	Pricing
Lite	10,000 Characters per Month	Free
The Lite plan gets you started with 10,000 characters per month at no cost. When you upgrade to a paid plan, you will get access to Customization capabilities. Lite plan services are deleted after 30 days of inactivity.		
Standard	Standard Characters	\$0.02 USD/THOUSAND CHAR
Premium	Everything in Standard plus... Usage and Training Data is Private – Stored in an Isolated Single Tenant Environment High Availability and Service Level Uptime Guarantee IBM Cloud Service Endpoints HIPAA - Washington DC Only Custom Voice (Beta)	

Create

Add to estimate

View terms

Summary

Text to Speech Free

Region: Dallas

Plan: Lite

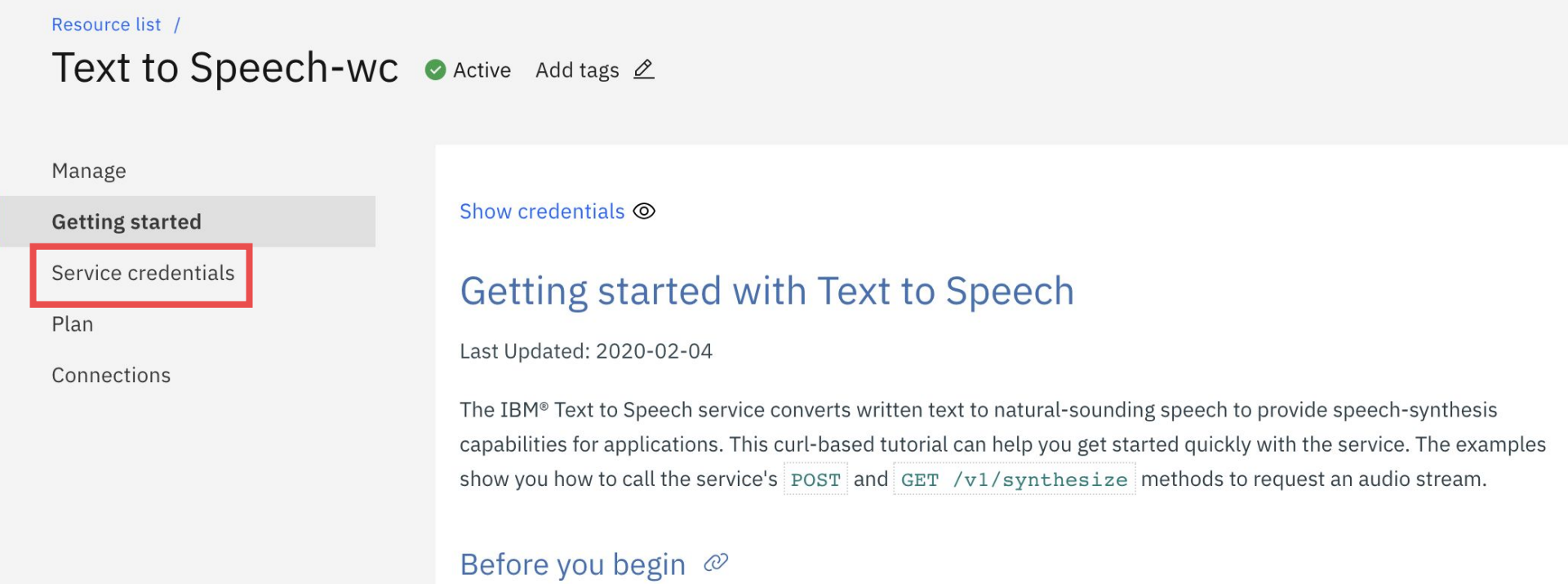
Service name: Text to Speech-wc

Resource group: Default

FEEDBACK

# Adicionando e criando o serviço Watson TTS

- Pronto, você criou o seu TTS. Agora basta clicar em “Service Credentials” para começarmos a configurar o serviço do Node-red.



Resource list /

## Text to Speech-wc ✓ Active [Add tags](#) [✎](#)

Manage

**Getting started**

**Service credentials**

Plan

Connections

[Show credentials](#) [👁](#)

### Getting started with Text to Speech

Last Updated: 2020-02-04

The IBM® Text to Speech service converts written text to natural-sounding speech to provide speech-synthesis capabilities for applications. This curl-based tutorial can help you get started quickly with the service. The examples show you how to call the service's `POST` and `GET /v1/synthesize` methods to request an audio stream.

[Before you begin](#) [🔗](#)

# Adicionando e criando o serviço Watson TTS

- Agora basta copiar e colar o valor “apikey” sem as aspas no atributo do nó do Watson TTS do Node-RED.

Resource list /

Text to Speech-wc Active [Add tags](#) [Details](#) [Actions...](#)

Manage

Getting started

**Service credentials**

Plan

Connections

Service credentials

You can generate a new set of credentials for cases where you want to manually connect an app or external consumer to an IBM Cloud™ service. [Learn more](#)

Search credentials...

New credential +

Key name	Date created
Auto-generated service credentials	MAY 16, 2020 - 08:29:02 PM

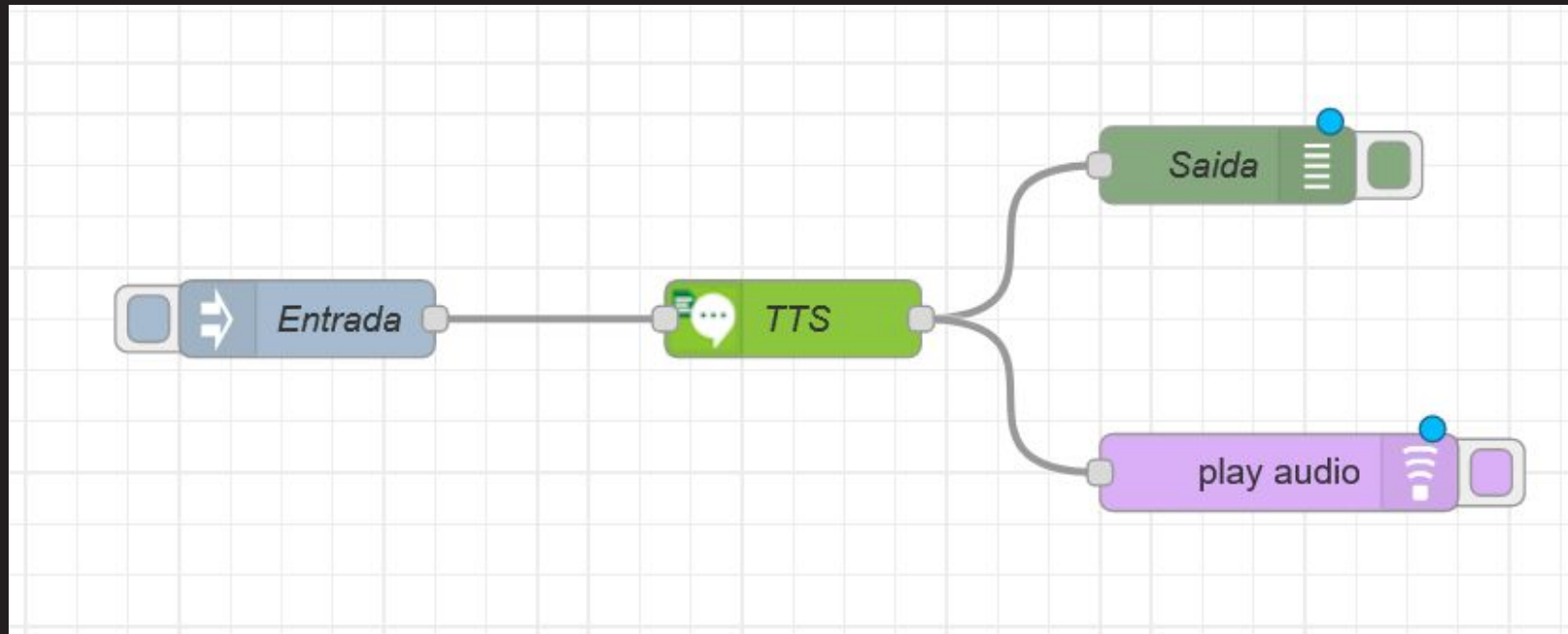
```
{
  "apikey": "imPqZcB4dpa9puDAcCR_FRVUndEJJbt2NE_weAp-U1L2",
  "iam_apikey_description": "Auto-generated service credentials",
  "iam_apikey_name": "Auto-generated service credentials",
  "iam_role_crn": "crn:v1:bluemix:public:iam::::serviceRole:Manager",
  "iam_serviceid_crn": "crn:v1:bluemix:public:iam-identity::a/6759b38be0ae9252454476395cc787e4::serviceid:ServiceId-f42918b8-060b-432e-9642-030242b23054",
  "url": "https://api.us-south.text-to-speech.watson.cloud.ibm.com/instances/35f365a3-e87f-40f2-b3c9-7eff5c5aae51"
}
```

# Hello World com TTS

Realizando os primeiros testes com Watson Text-to-Speech

# Testando o TTS

- Vamos adicionar 4 nós: inject, Watson Text-to-Speech, debug e output play áudio. O resultado deve ser:



# Testando o TTS

- Dentro de cada nó temos:

## Watson Text to Speech

**Properties**

Name: TTS

Username: Username

Password: Password

API Key: [Redacted]

Service Endpoint: https://stream.watsonplatform.net/text-to-speech/

Language: Portuguese Brazilian

Voice: IsabelaV3

Format: WAV

☒ Place output on msg.payload

## Inject

**Properties**

Name: Entrada

msg.payload = Oi, eu sou a Isabela

## Debug

**Properties**

Output: msg.payload

To: ☒ debug window  
☐ system console  
☐ node status (32 characters)

Name: Saida

## Play audio

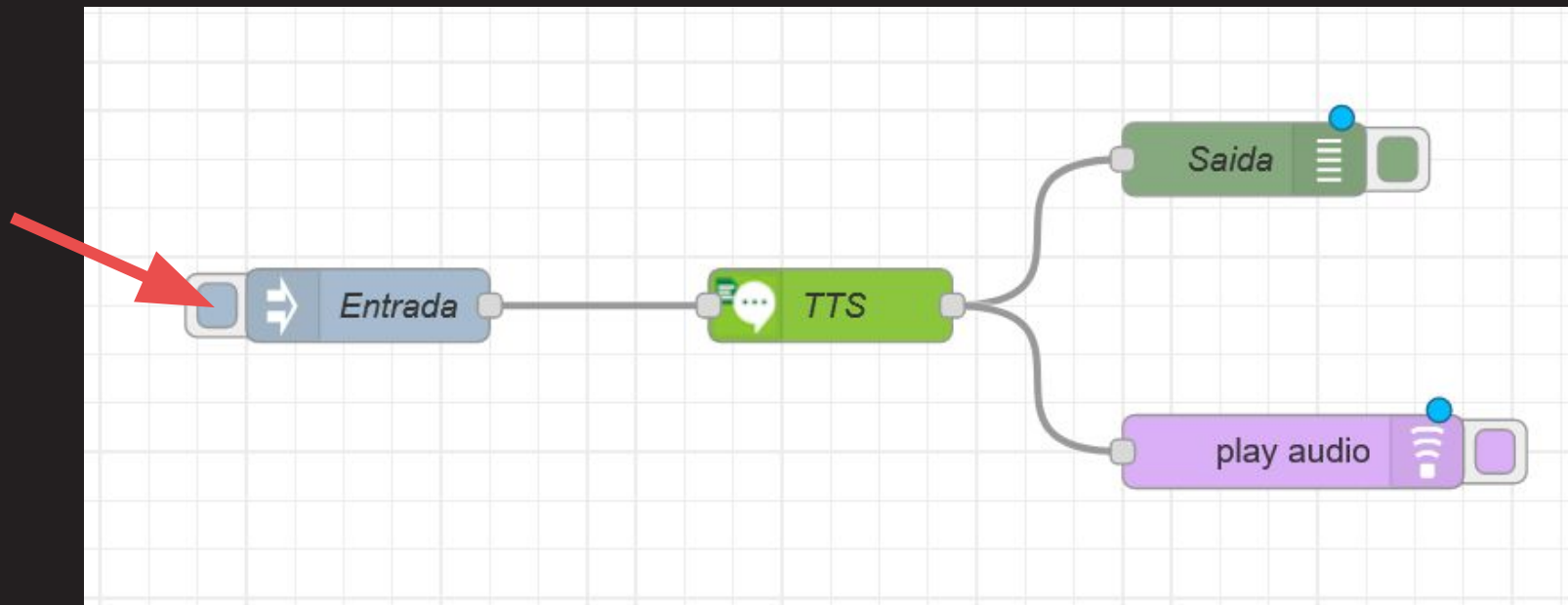
**Properties**

TTS Voice: Microsoft Maria Desktop - Portu

Name: Name

# Testando o TTS

- Aperte Deploy e teste (não esqueça de apertar o botão de inject).

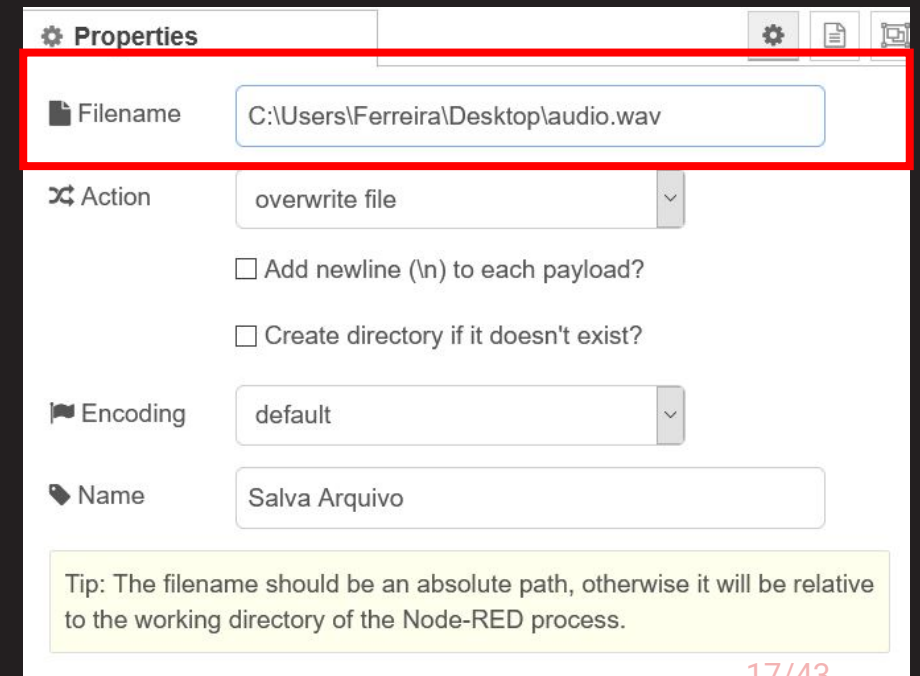
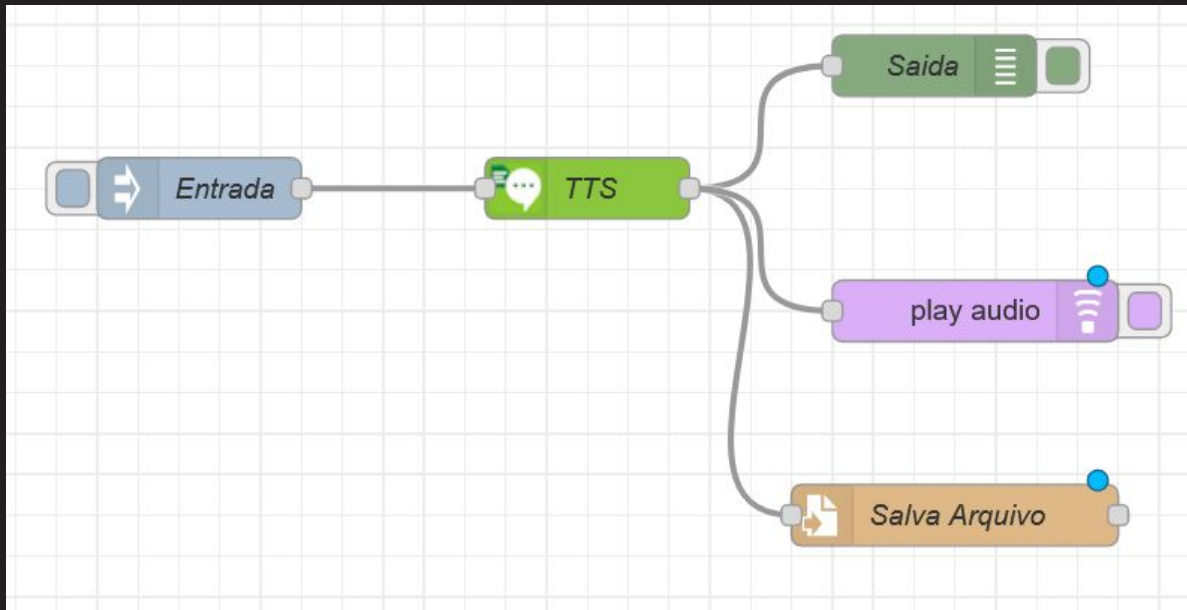


- Remova a virgula da frase do nó inject. Ocorre alguma diferença?



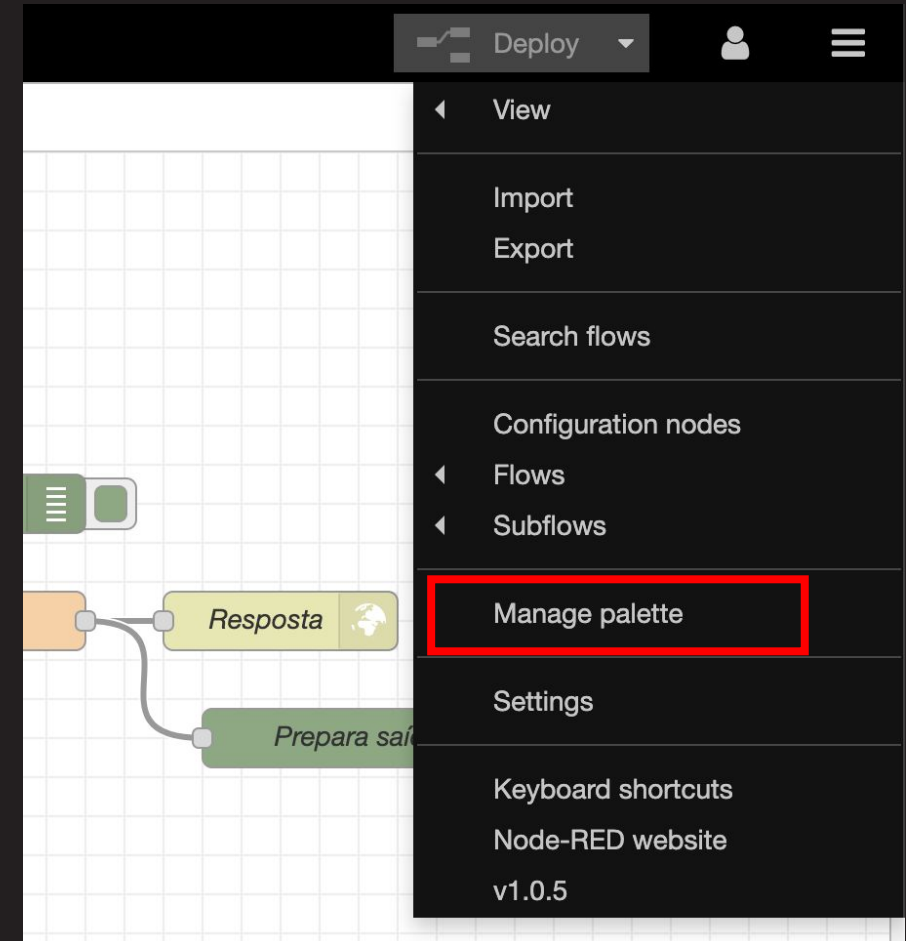
# Salvando arquivo de áudio do TTS

- Caso você queira salvar o arquivo de áudio do seu TTS, basta adicionar o nó **Storage file**. Perceba que o formato do arquivo deve ser o mesmo de retorno do TTS. Além disso o nome do arquivo (filename) deve conter o caminho absoluto e a extensão do arquivo.



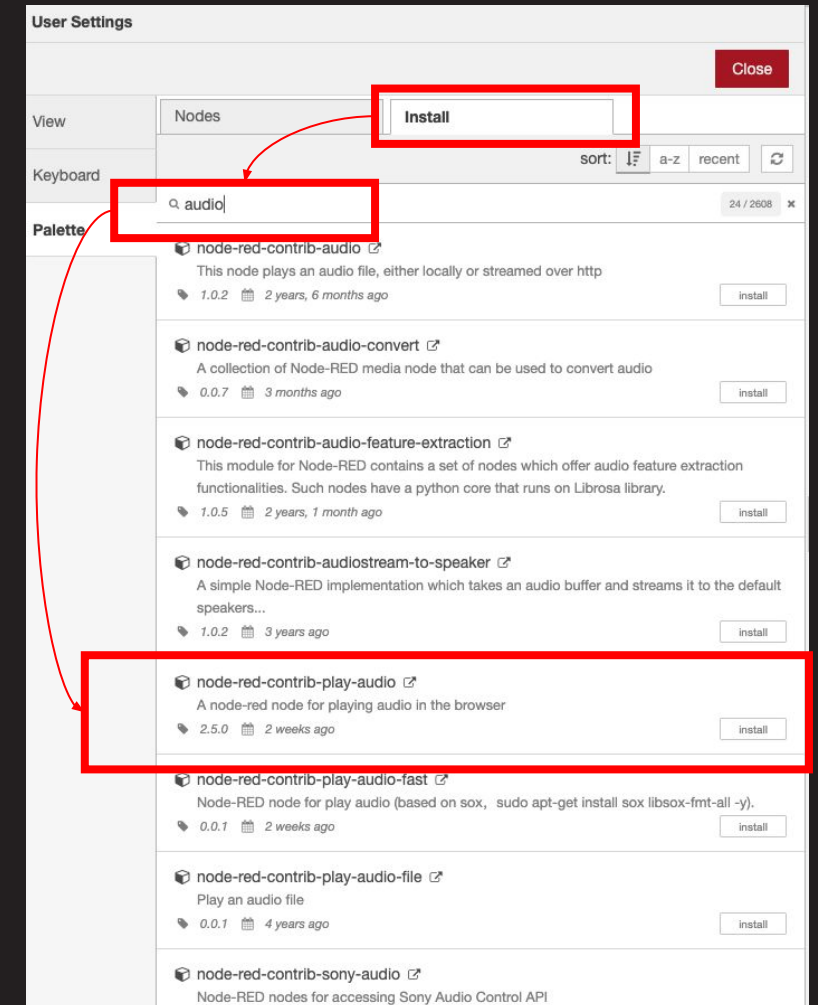
# Caso não aparece o nó de Play Audio no menu de Output

- Para conseguir ouvir o que o Assistente está dizendo temos que incluir o nó correspondente ao alto-falante.
- Como esse nó não existe no Node-RED padrão então temos que instalá-lo manualmente clicando em “Manage palette”



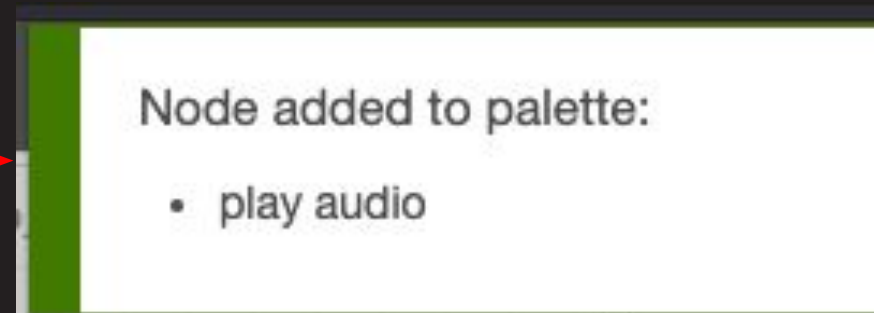
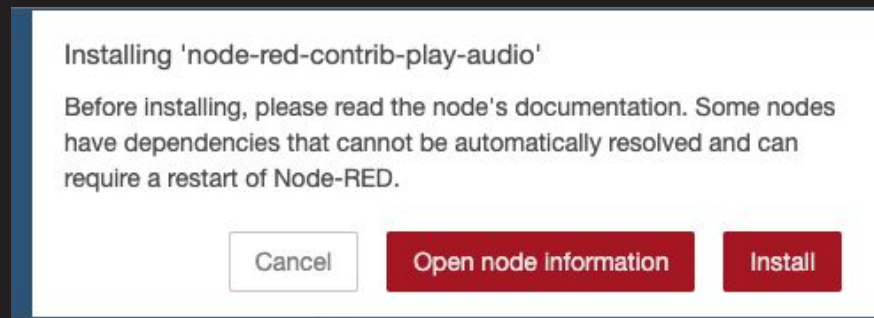
# Caso não aparece o nó de Play Audio no menu de Output

- Agora clique na aba “Install”, depois digite audio e vamos instalar o pacote “node-red-contrib-play-audio” clicando em “install”
- Uma caixa de confirmação aparecerá no topo da tela.



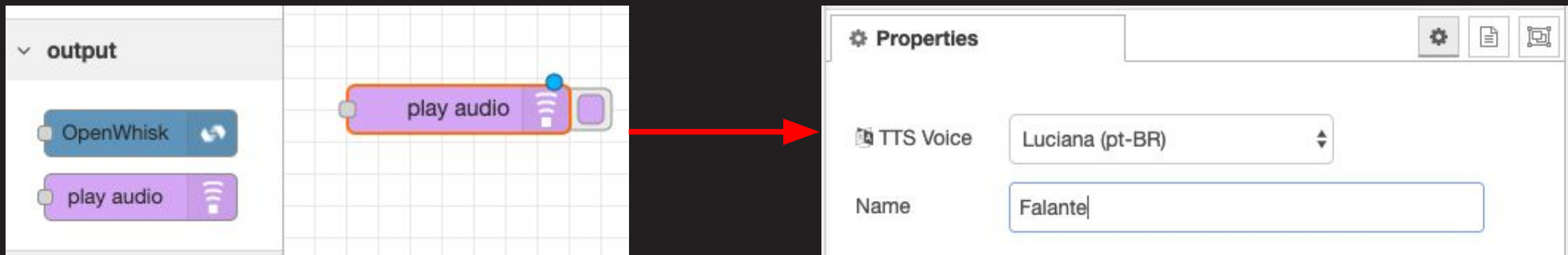
# Caso não aparece o nó de Play Audio no menu de Output

- Confirme a instalação clicando em “Install” na janela que se abrirá no topo da tela e logo o pacote estará instalado e uma mensagem em verde, como a mostrada abaixo, aparecerá.



# Caso não aparece o nó de Play Audio no menu de Output

- O alto-falante está instalado e agora você pode incluí-lo no fluxo, configurá-lo com a voz “Luciana (pt-BR)” e renomeá-lo, como demonstrado abaixo:

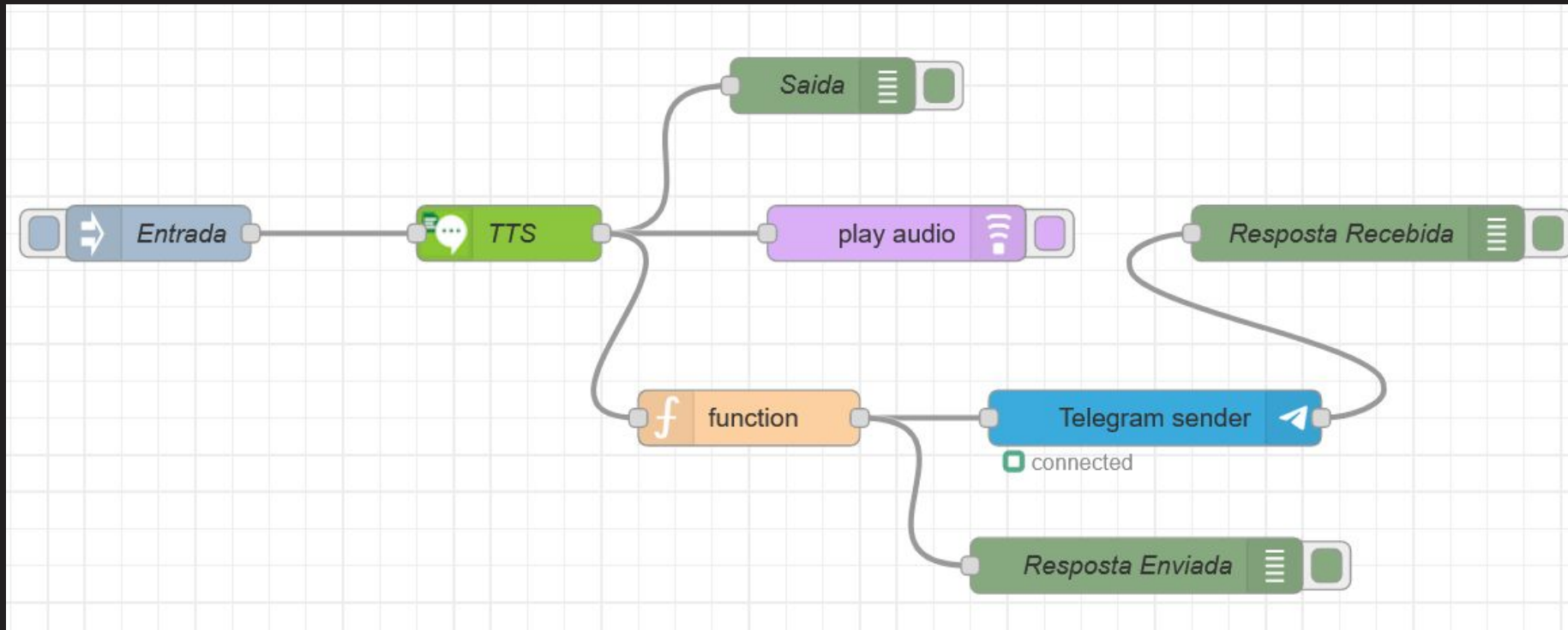


# Ensinando o bot a falar I

Enviando mensagens de voz para o Telegram

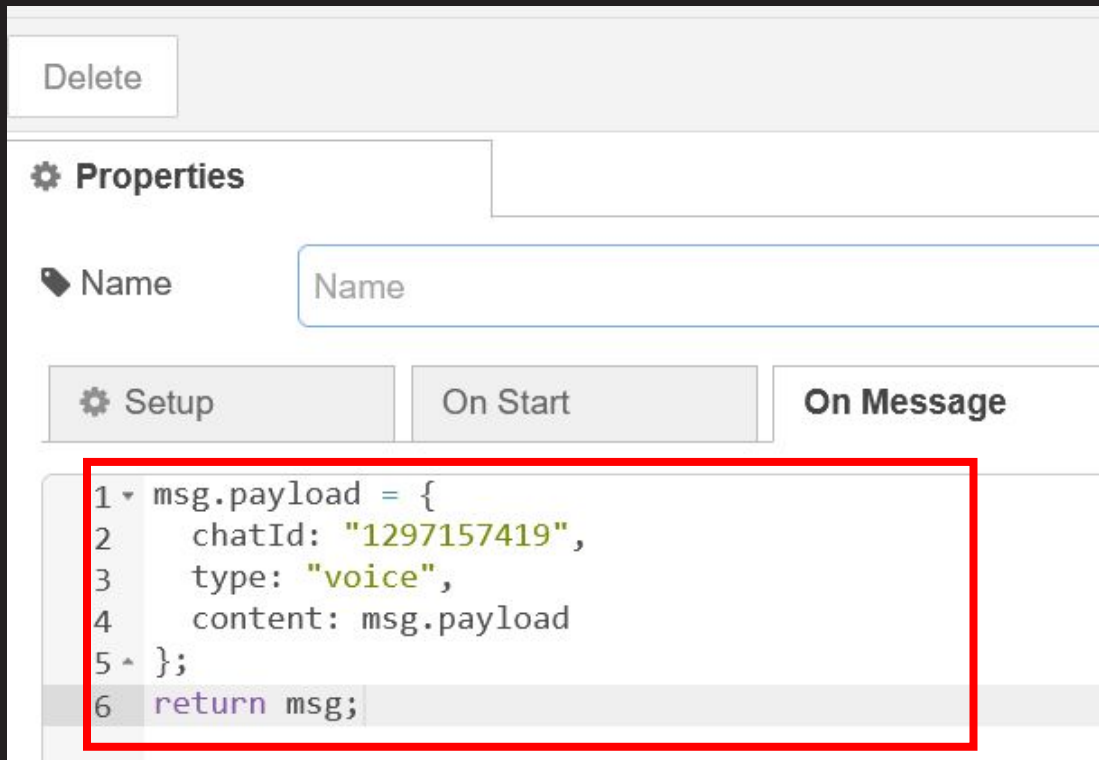
# Enviando o arquivo de TTS para o Telegram

- Vamos testar o envio de mensagem de voz para o telegram. Para realizar isso vamos precisar mais dois nós: **function** e **telegram sender**. Também vamos adicionar 2 debugs para observar os padrões de mensagens trocados:



# Enviando o arquivo de TTS para o Telegram

**Function:** atenção ao chatId que deve ser o número do seu telegram-bot



Delete

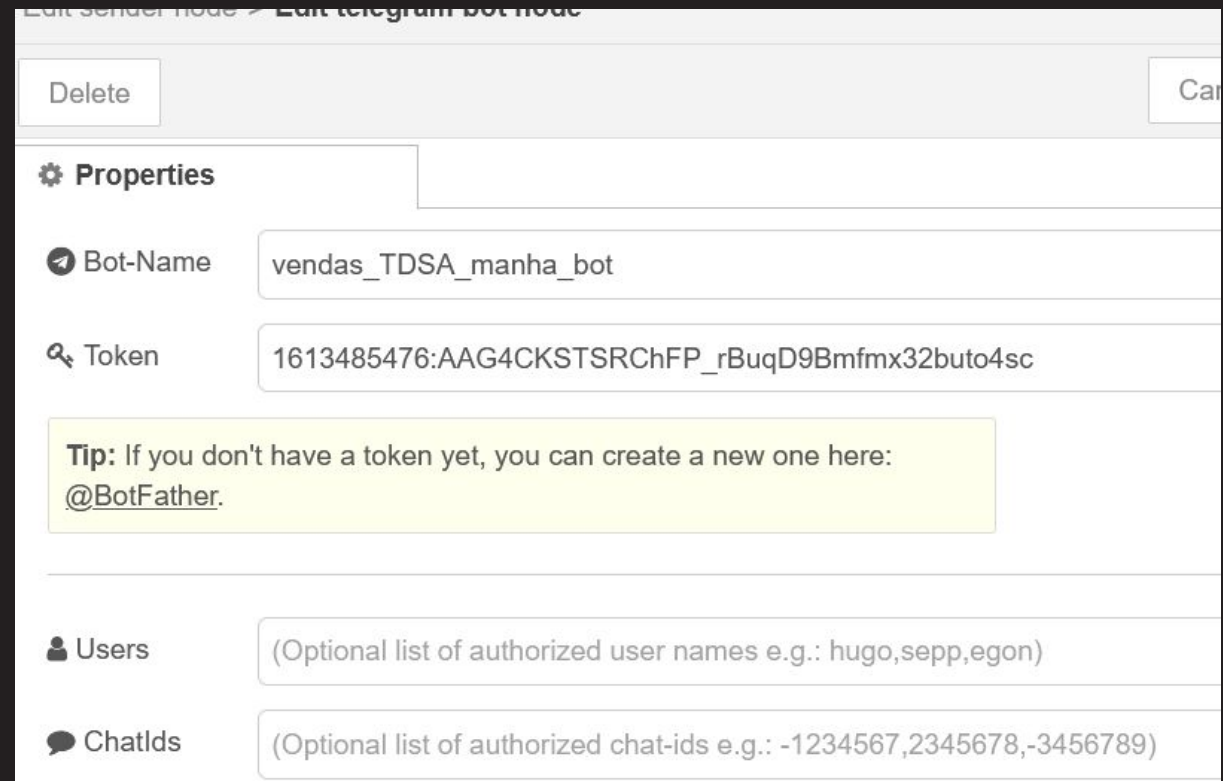
⚙️ Properties

🏷️ Name

⚙️ Setup On Start On Message

```
1 msg.payload = {  
2   chatId: "1297157419",  
3   type: "voice",  
4   content: msg.payload  
5 };  
6 return msg;
```

## Telegram Sender



Edit sender node > Edit telegram bot node

Delete Cancel

⚙️ Properties

🔑 Bot-Name

🔑 Token

Tip: If you don't have a token yet, you can create a new one here: [@BotFather](#).

👤 Users

💬 ChatIds



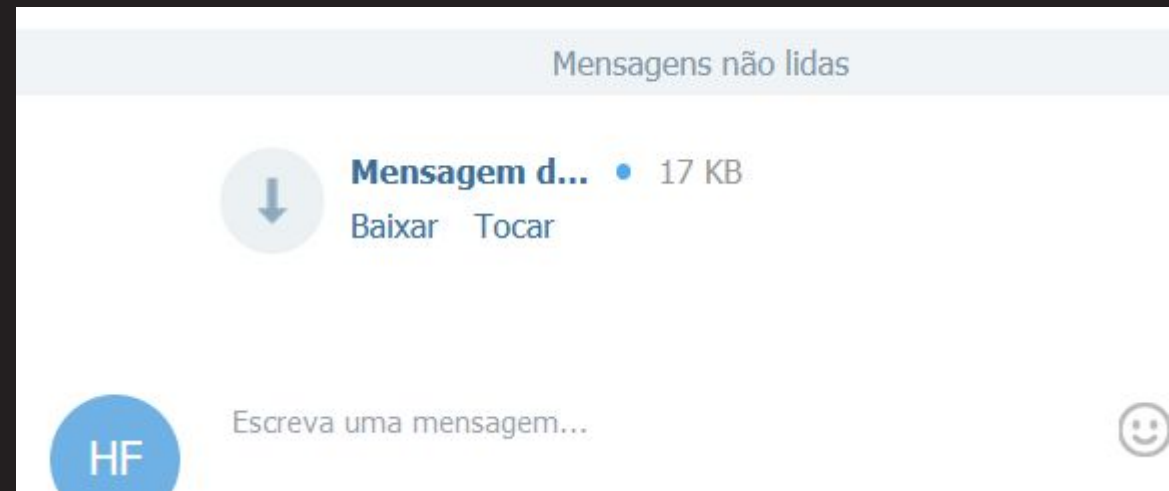
# Enviando o arquivo de TTS para o Telegram

- Resultado deve ser:

## Debug

```
msg.payload : Object
  ▼ object
    chatId: "1297157419"
    type: "voice"
    ▶ content: buffer[17538]
```

## Telegram

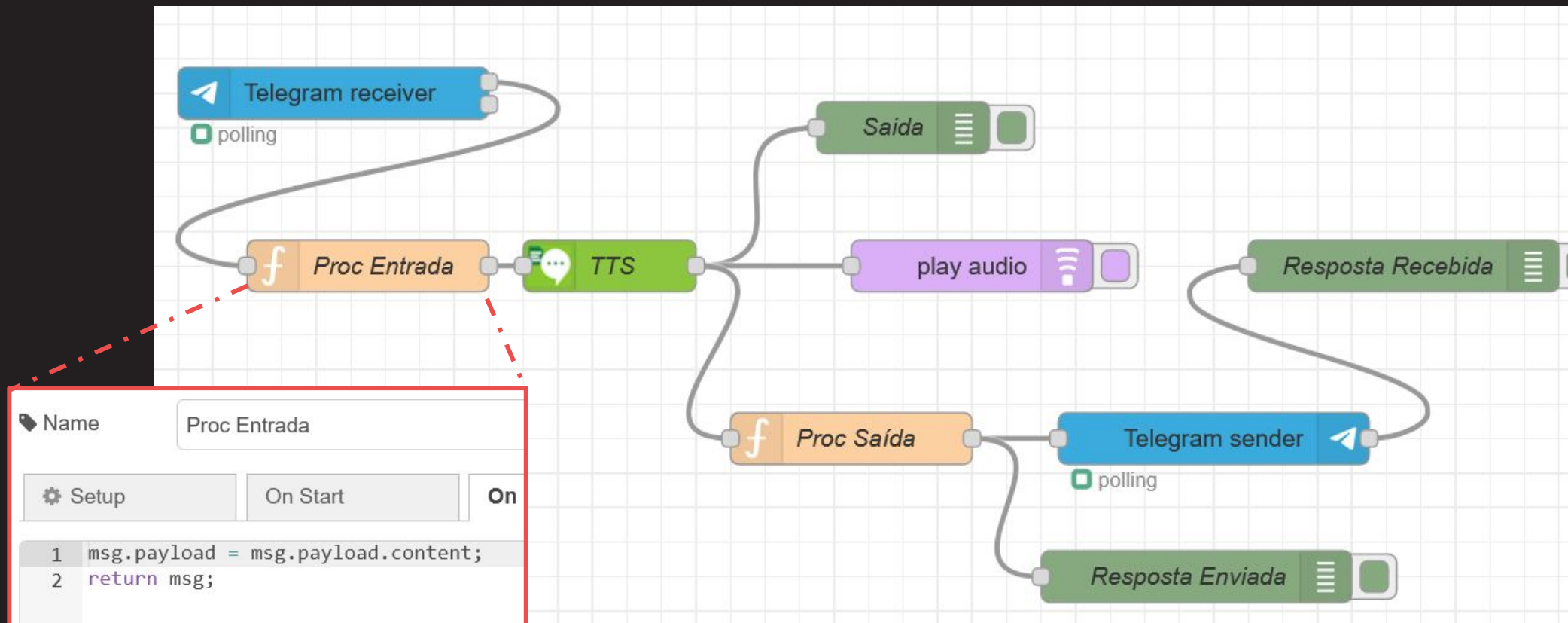


# Ensinando o bot a falar II

Transformando mensagens de texto em áudio

# Transformando texto enviado em áudio

- Vamos modificar o fluxo do exemplo anterior, trocando o nó de Inject de Entrada pelo Telegram Receiver. Vamos precisar adicionar um nó de processamento (function) entre a entrada e o TTS.



# Transformando texto enviado em áudio

- Teste!
- O resultado esperado no Telegram será algo como:

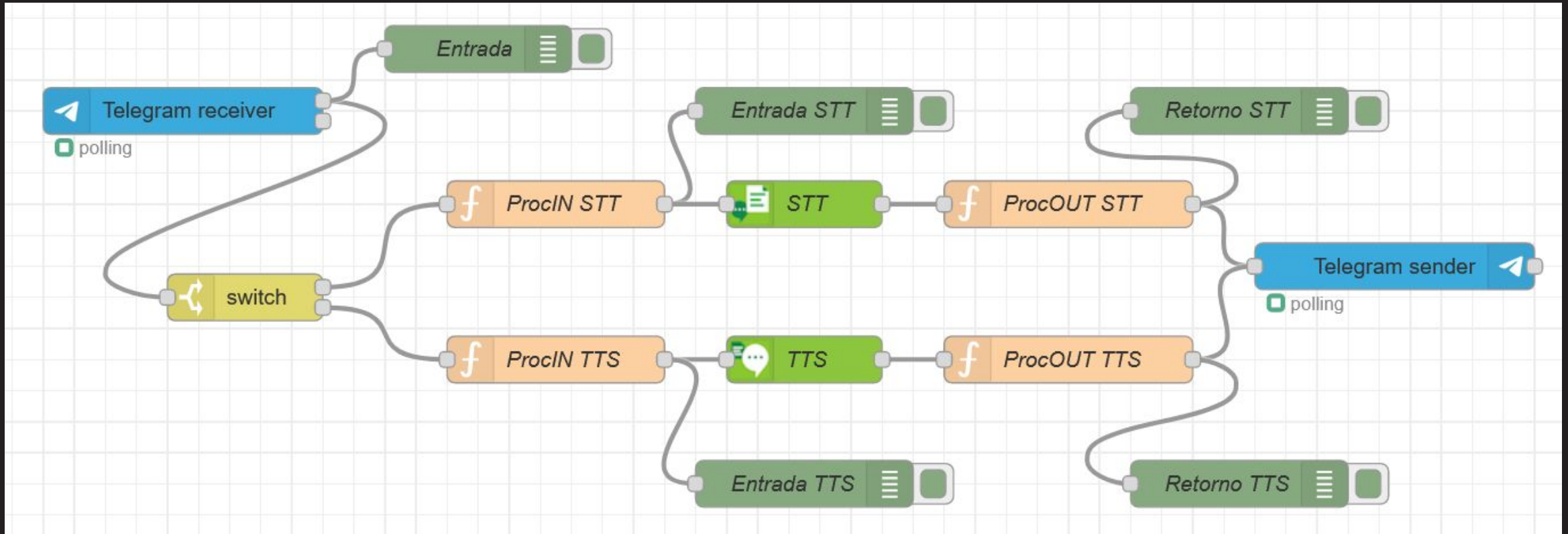


# Ensinando o bot a falar III

Bot conversor de texto e áudio

# Criando um bot conversor Áudio-Texto

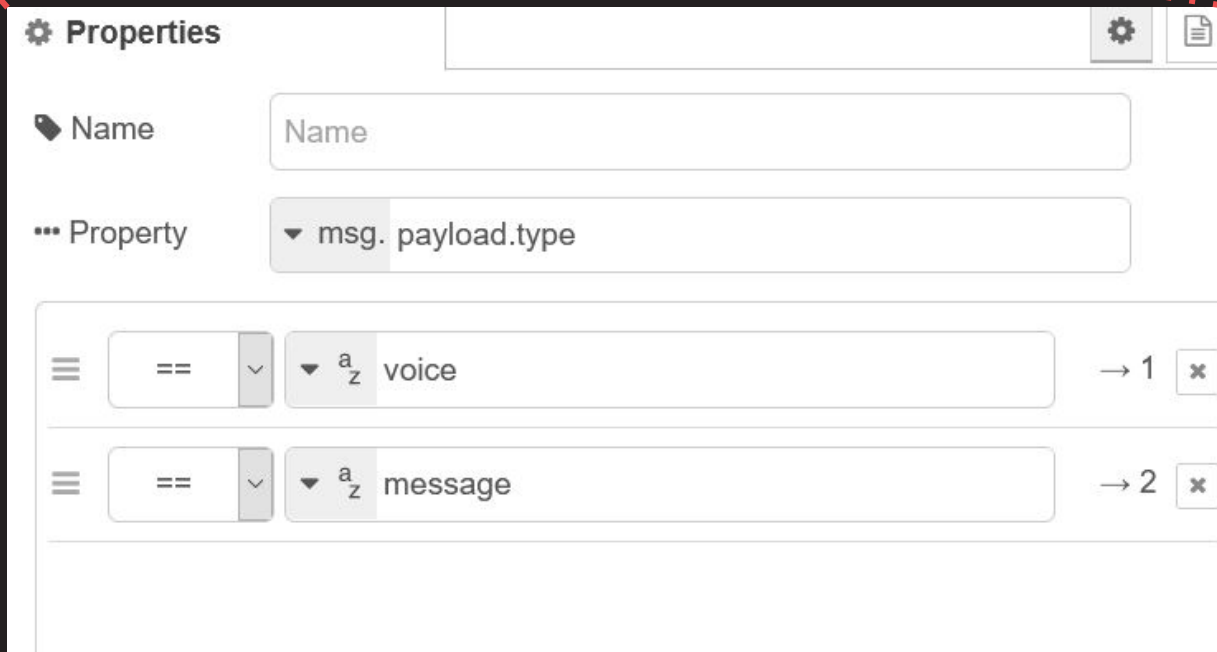
- Agora vamos criar um bot que saiba **transcrever áudio** e **falar texto**:



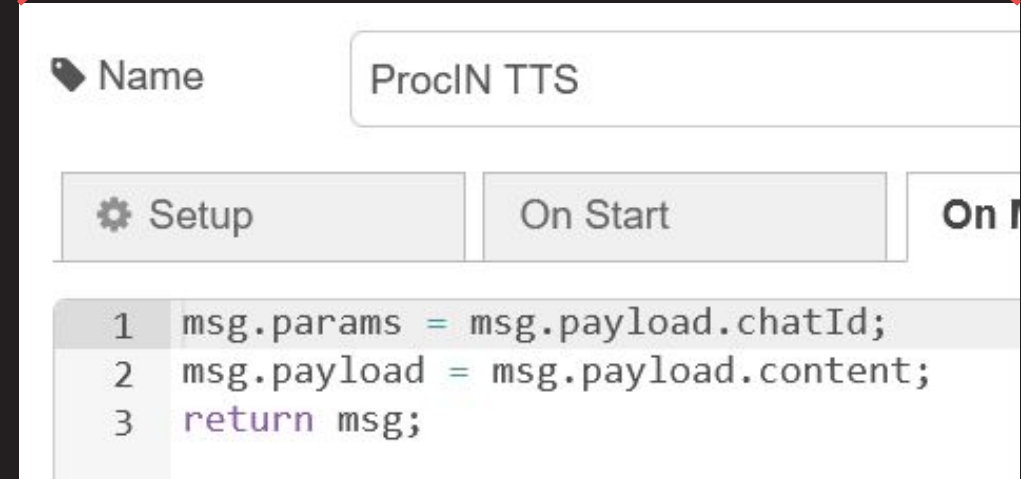
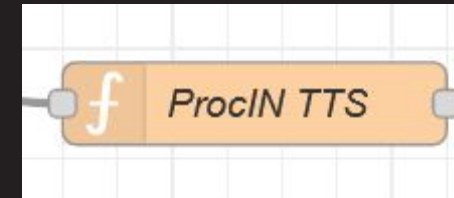
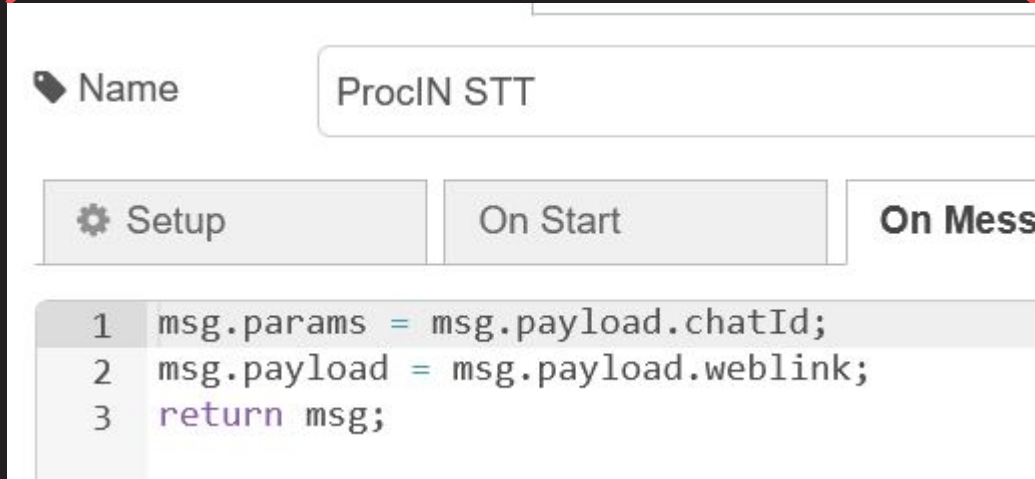
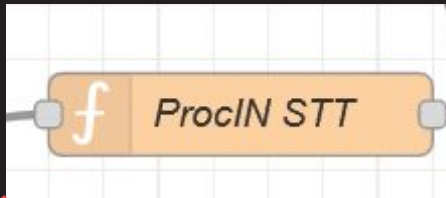
# Criando um bot conversor Áudio-Texto



- Preenchendo o nó de Switch;
- Esse nó é responsável por separar mensagens de texto e de voz;



# Criando um bot conversor Áudio-Texto





# Criando um bot conversor Áudio-Texto

STT

Username

Password

API Key

Service Endpoint  
<https://stream.watsonplatform.net/speech-to-text>

Language  
Portuguese Brazilian

Quality  
BroadbandModel

Max Alternative Transcripts  
1

Keywords

Keywords Threshold  
0,5

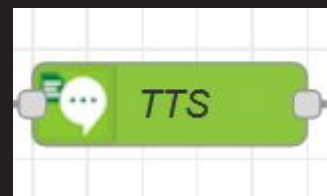
☐ Word Confidence

☐ Speaker Labels

☐ Smart Formatting

☒ Place output on msg.payload

☐ Disable Audio format pre-check



TTS

Username

Password

API Key

Service Endpoint  
<https://stream.watsonplatform.net/text-to-speech/>

Language  
Portuguese Brazilian

Voice  
IsabelaV3

Format  
WAV

☒ Place output on msg.payload

# Criando um bot conversor Áudio-Texto



**Name** ProcOUT STT

**Setup** **On Start** **On Message**

```
1 msg.payload = {
2   chatId : msg.params,
3   content : msg.payload,
4   type : 'message'
5 }
6 return msg;
```

**Name** ProcOUT TTS

**Setup** **On Start** **On Message**

```
1 msg.payload = {
2   chatId : msg.params,
3   content : msg.payload,
4   type : 'voice'
5 }
6 return msg;
```

# Testando

## Debug Node-RED

```
18/04/2021 11:57:55 node: Entrada
msg.payload : Object
▶ { chatId: 1297157419, messageId: 630, type: "message", content: "Teste texto para áudio", date: 1618757875 }

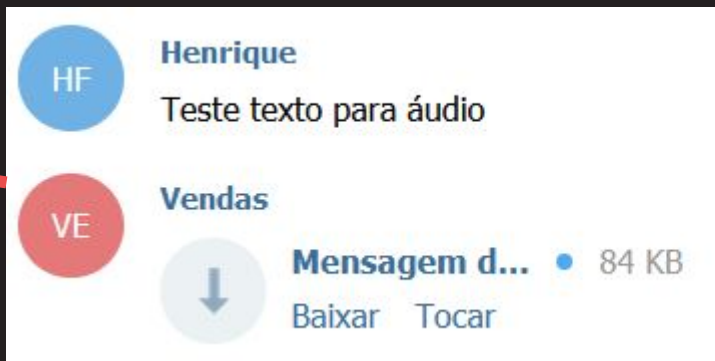
18/04/2021 11:57:55 node: Entrada TTS
msg.payload : string[22]

"Teste texto para áudio"

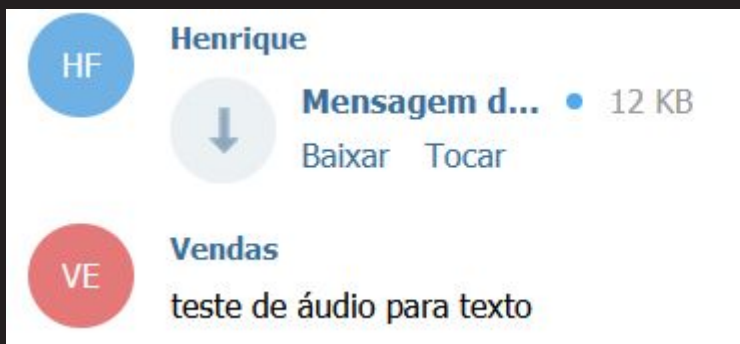
18/04/2021 11:57:57 node: Retorno TTS
msg.payload : Object
▶ { chatId: 1297157419, content: buffer[86318], type: "voice" }
```

## Telegram

### Text-to-Speech



### Speech-to-Text



## Debug Node-RED

```
18/04/2021 11:57:57 node: Retorno TTS
msg.payload : Object
▶ { chatId: 1297157419, content: buffer[86318], type: "voice" }

18/04/2021 11:58:52 node: Entrada
msg.payload : Object
▶ { chatId: 1297157419, messageId: 632, type: "voice", content: "AwACAgEAAxkBAAICeGB8SSy81S1Lzg...", caption: undefined ... }

18/04/2021 11:58:53 node: Entrada STT
msg.payload : string[97]

"https://api.telegram.org/file/bot1613485476:AAG4CKSTSRChFP_rBuqD9Bmfmx32buto4sc/voice/file_22.oga"

18/04/2021 11:58:58 node: Retorno STT
msg.payload : Object
▶ { chatId: 1297157419, content: "teste de áudio para texto ", type: "message" }
```

# Ensinando o bot a falar IV

Integração do TSS com o Watson Asssintat



# Criando um bot com saída de voz



Name: ProcIN WA

Setup On Start On Message

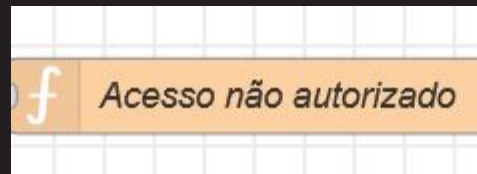
```
1 msg.params = {};  
2 msg.params.session_id = msg.payload.chatId;  
3 msg.payload = msg.payload.content;  
4 return msg;
```



Name: ProcOUT WA

Setup On Start On Message

```
1 msg.payload = msg.payload.output.generic[0].text;  
2 return msg;
```



Name: Acesso não autorizado

Setup On Start On Message On Stop

```
1 msg.payload.content = "Usuário sem permissão de acesso."  
2 return msg;
```

Name: ProcOUT TTS

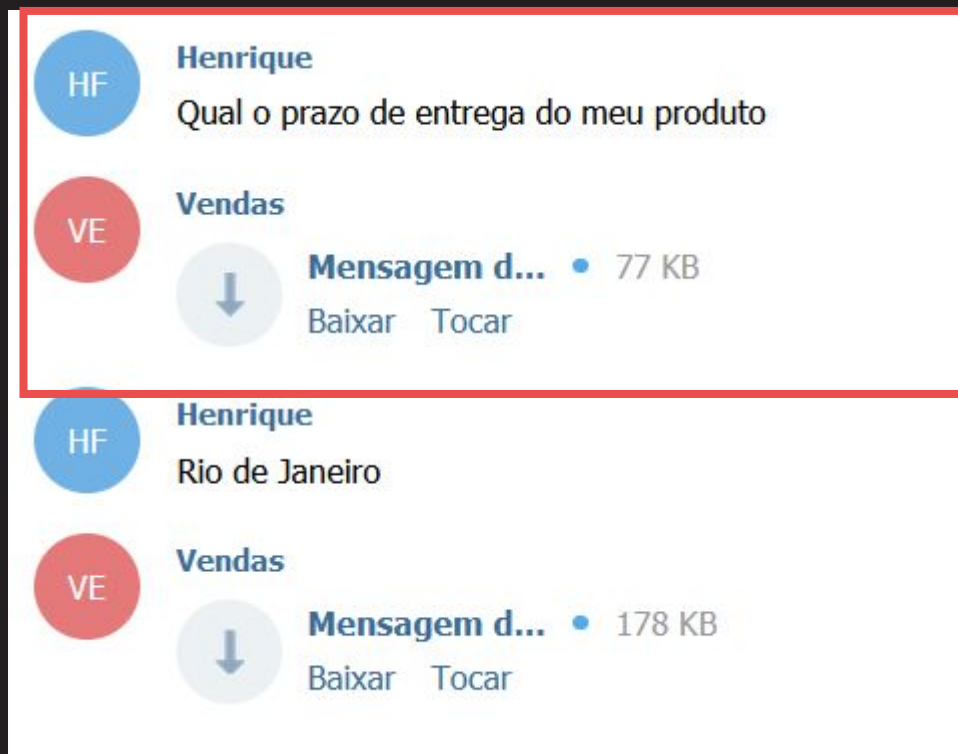
Setup On Start On Message

```
1 msg.payload = {  
2   chatId : msg.params.session_id,  
3   content : msg.payload,  
4   type : 'voice'  
5 }  
6 return msg;
```



# Testando

## Telegram



## Debug Node-RED

```
18/04/2021 12:46:21 node: Entrada
msg.payload : Object
  ▶ { chatId: 1297157419, messageId: 653, type: "message", content: "Qual o prazo de entrega do meu...", date: 1618760781 }

18/04/2021 12:46:23 node: Saída WA
msg.payload : Object
  ▶ { output: object, user_id: "cd88b991-3af7-46f2-8c84-97a893...", context: object, session_id: "cd88b991-3af7-46f2-8c84-97a893..." }

18/04/2021 12:46:23 node: Entrada TTS
msg.payload : string[22]
  "Qual a sua localidade?"

18/04/2021 12:46:25 node: Retorno TTS
msg.payload : Object
  ▶ { chatId: 1297157419, content: buffer[78398], type: "voice" }
```

Enviado para o Telegram

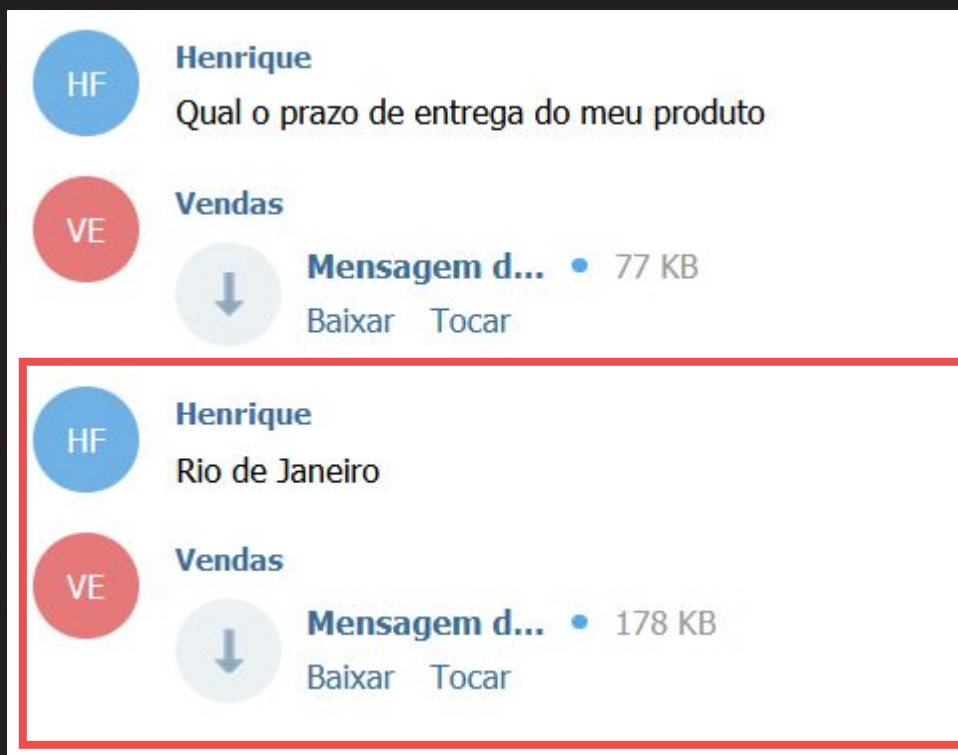
Recebido do WA

Enviado para o TTS

Recebido do TTS e enviado para o Telegram

# Testando

## Telegram



## Debug Node-RED

```
18/04/2021 12:46:34 node: Entrada
msg.payload : Object
  { chatId: 1297157419, messageId: 655, type: "message", content: "Rio de Janeiro", date: 1618760794 }

18/04/2021 12:46:35 node: Saída WA
msg.payload : Object
  { output: object, user_id: "cd88b991-3af7-46f2-8c84-97a893...", context: object, session_id: "cd88b991-3af7-46f2-8c84-97a893..." }

18/04/2021 12:46:35 node: Entrada TTS
msg.payload : string[54]
  "Em até 20 dias úteis você terá o produto na sua casa. "

18/04/2021 12:46:37 node: Retorno TTS
msg.payload : Object
  { chatId: 1297157419, content: buffer[181798], type: "voice" }
```

Enviado para o Telegram

Recebido do WA

Enviado para o TTS

Recebido do TTS e enviado para o Telegram



**Agora é com você!**

# Exercícios

1. Junte tudo que vimos nas duas últimas aulas fazendo o nosso bot Vendas **falar e ouvir**, assim como **escrever e ler** no **Telegram**. Dicas: use o nó de **switch** para escolher fluxos para cada tipo de mensagem recebida; não esqueça de manter o contexto para que o Watson Assistant saiba estar falando com a mesma pessoa.

**Copyright © 2022**

**Slides do Prof. Henrique Ferreira, com adaptações dos  
slides dos Prof. Marcelo Grave - FIAP**

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).