

Como criar um ChatBot no Telegram com IBM Watson

- Conversation
- Text to Speech
- Speech to text





1 – Criando uma conta no Bluemix e Watson services


- Acesse: <https://console.ng.bluemix.net/registration/> e crie sua conta no Bluemix (gratuito por 30 dias).
- Crie os serviços que vamos usar neste tutorial. Acesse os links a seguir:
 - Conversation: <https://console.ng.bluemix.net/catalog/services/conversation/>
 - Text to Speech: <https://console.ng.bluemix.net/catalog/services/text-to-speech/>
 - Speech to Text: <https://console.ng.bluemix.net/catalog/services/speech-to-text/>


2 – Criando o Bot no Telegram


- Agora, faça login na sua conta do Telegram e inicie uma conversa com o @BotFather para criar seu Bot no Telegram (Salve seu API Token).


 **Leonardo**
/newbot


 **BotFather**
Alright, a new bot. How are we going to call it? Please choose a name for your bot.


 **Leonardo**
Watson tutorial bot


 **BotFather**
Good. Now let's choose a username for your bot. It must end in `bot`. Like this, for example: TetrisBot or tetris_bot.

 **Leonardo**
WatsonTutorialbrBot

 **BotFather**
Good. Now let's choose a username for your bot. It must end in `bot`. Like this, for example: TetrisBot or tetris_bot.

 **Leonardo**
WatsonTutorialbrBot

 **BotFather**
Done! Congratulations on your new bot. You will find it at t.me/WatsonTutorialbrBot. You can now add a description, about section and profile picture for your bot, see [/help](#) for a list of commands. By the way, when you've finished creating your cool bot, ping our Bot Support if you want a better username for it. Just make sure the bot is fully operational before you do this.

Use this token to access the HTTP API:


3 – Criando seu App com Node.js

- Crie um Aplicativo Node.js no Bluemix.
 - <https://console.ng.bluemix.net/catalog/starters/sdk-for-nodejs>
- Clique em “Download Starter Code”

4 – Entendendo Conversation

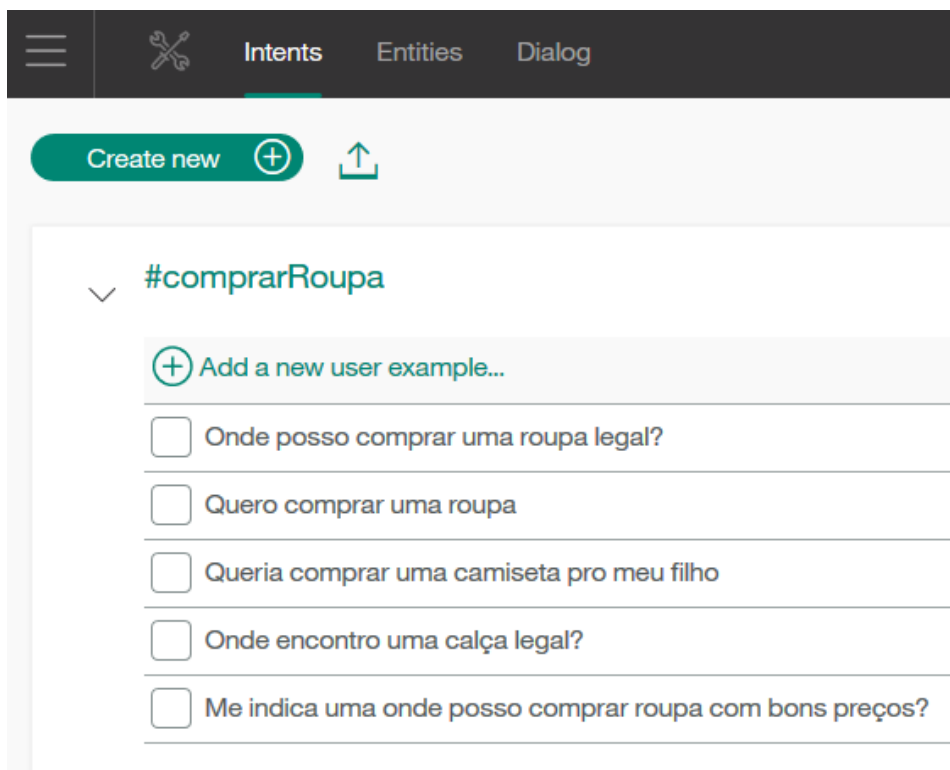
O Conversation é composto por três coisas, Intenções, Entidades e Dialogo, aqui vamos aprender a montar uma conversa.

Intenções: é a intenção do usuário, ou seja, o objetivo dele com aquela pergunta (texto inputado). Exemplo: 'Onde posso comprar uma roupa legal?' a intenção dele é: 'Comprar Roupa'.

Entidades: É como uma classe, composta com diversas palavras com o mesmo gênero. Exemplo: Entidade 'Produtos' composta por 'Celular, Computador, Televisão ...' todos eles são um produto.

Dialogo: No Conversation o dialogo é onde você configura o escopo da sua conversa e insere as respostas para cada intenção ou entidade.

4 – Criando Intenções



The screenshot shows the 'Intents' tab in the IBM Watson Assistant interface. At the top, there are tabs for 'Intents', 'Entities', and 'Dialog'. Below the tabs, there is a 'Create new' button with a plus icon and an upload icon. The main area displays the '#comprarRoupa' intent, which is expanded to show a list of user examples. Each example is preceded by a checkbox and a plus icon for adding new examples.

Intents

Create new

✓ #comprarRoupa

- + Add a new user example...
- ☐ Onde posso comprar uma roupa legal?
- ☐ Quero comprar uma roupa
- ☐ Queria comprar uma camiseta pro meu filho
- ☐ Onde encontro uma calça legal?
- ☐ Me indica uma onde posso comprar roupa com bons preços?

> **#comprarRoupa**
Onde posso comprar um roupa legal?

> **#fimConversa**
Até a proxima

> **#foraDeEscopo**
Como tá a temperatura?

> **#formaDePagamento**
Como posso pagar minhas contas?

> **#inicioConversa**
eae

> **#trocaDeMercadoria**
Como funciona a troca aqui?

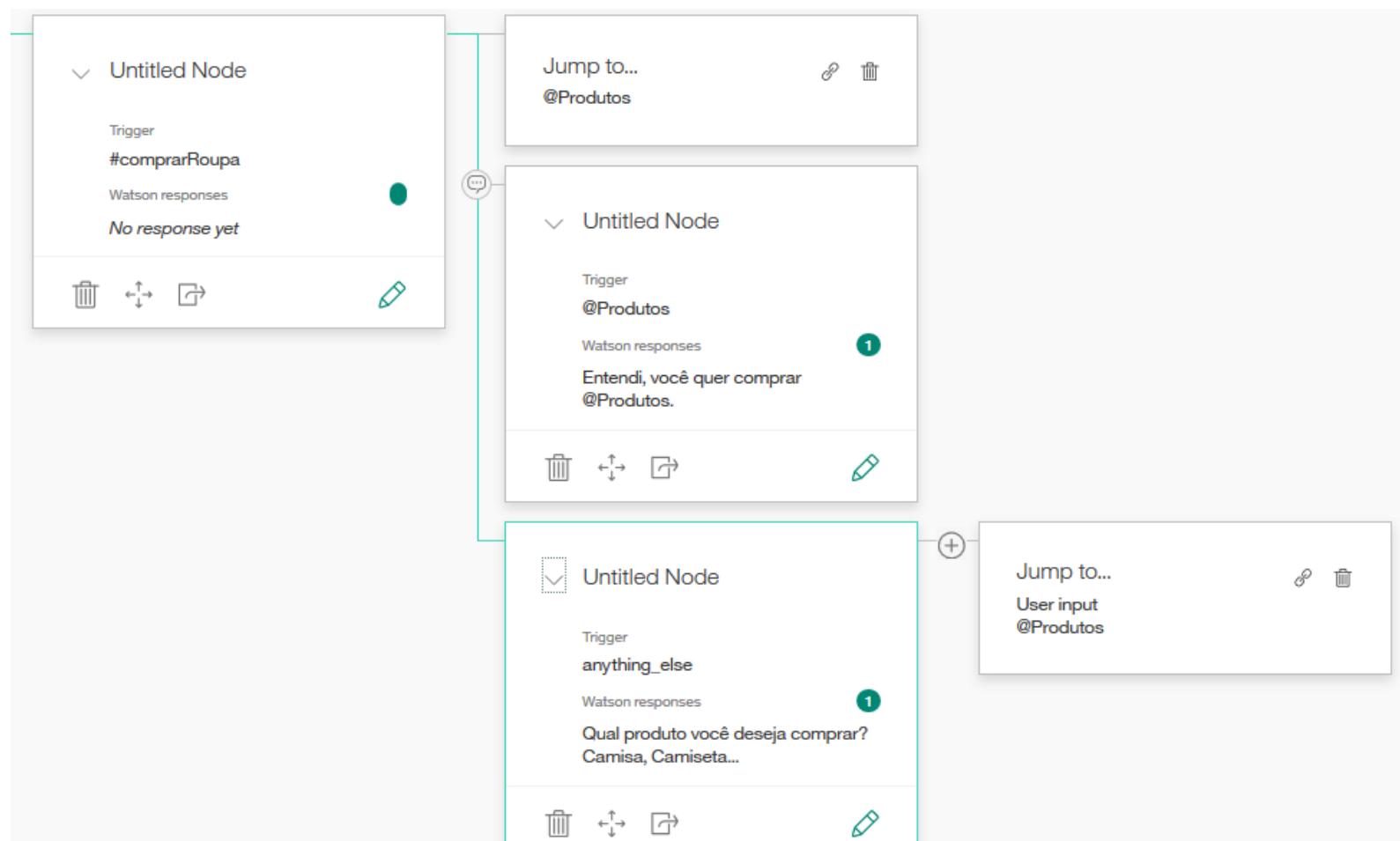
4 – Criando Entidades

The screenshot shows the 'Entities' tab in the IBM Watson Assistant interface. At the top, there's a navigation bar with 'Intents', 'Entities' (selected), and 'Dialog'. Below this, there are tabs for 'My entities' and 'System entities'. A 'Create new' button with a plus icon and an upload icon is visible. The main area shows a list of entities under the '@Produtos' category. Each entity has a checkbox, a name, and a link to 'Add synonyms...'. The entities listed are Blusa, Boné, Calca, Camisa, Camiseta, Casaco, and Bolsa. The 'Bolsa' entity has a synonym 'Mala' listed below it.

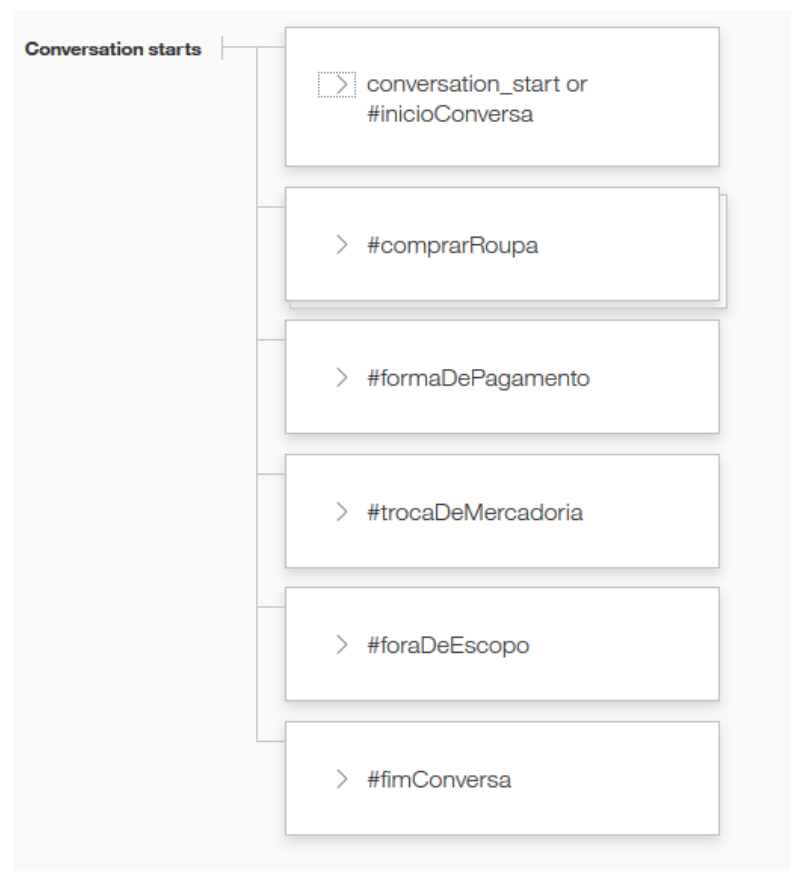
Entity	Synonyms
Blusa	
Boné	
Calca	
Camisa	
Camiseta	
Casaco	
Bolsa	Mala

- > **@Produtos**
Blusa, Boné, Calca, Camisa, Camiseta, Casaco, Bolsa
- > **@Tamanho**
Grande, Médio, Pequeno

4 – Criando Dialogo



4 – Criando Dialogo



Entendendo o código

- Faça download deste repositório no GitHub: <https://github.com/lfurnielis/telegram-bot>

5 – Entendendo o código (payload)

A variável *payload* é usada para passar os parâmetros necessários para a API Conversation, a variável é carregada com *workspace_id*, *context* e *input.text* (texto do usuário).

```
var payload = {  
  workspace_id: {},  
  context: {},  
  input: {  
    text: resposta.toString()  
  }  
};
```

5.1 – Entendendo o código (Conversation)

Neste trecho chamamos o método message passando como parâmetros o payload, o callback retorna um erro e a resposta. Se não houver erro, salvamos o context e retornamos a resposta.

```
conversation.call = function(payload, callback) {  
  payload.workspace_id = workspace;  
  payload.context = context;  
  
  wdc_conversation.message(payload, function(err, res) {  
    if (err) {  
      console.log('error:' + err);  
      return;  
    }  
    context = res.context;  
    callback(res);  
  });  
}
```

5.2 – Entendendo o código (Text to Speech)

Aqui temos o código com o método *synthesize* enviamos para ele a variável *params*(json) informamos o *text* (texto que será transformado em áudio), *voice* (idioma) e *accept* (formato do arquivo de áudio que será retornado). Criamos o arquivo de áudio com o código `fs.createWriteStream('voice-audio.ogg')` quando o arquivo estiver completo retornamos com o callback.

```
TTS.sintetizar = function(message, callback) {  
  
    var params = {  
        text: message,  
        voice: config.texttospeech.voice,  
        accept: 'audio/ogg'  
    };  
  
    text_to_speech.synthesize(params).on('error', function(error) {  
        console.log('Error:', error);  
    }).pipe(fs.createWriteStream('voice-audio.ogg'))  
        .on('finish', function(){  
            callback();  
        });  
}
```

5.3 – Entendendo o código (Speech to Text)

Agora, vamos entender como funciona o Speech to Text, ao chamarmos a função *createRecognizeStream*, passamos os seguintes parâmetros, *model*(idioma) e *content_type*(formato do arquivo de áudio), feito isso você terá a resposta como retorno em texto.

```
STT.telegramVoice = function(link, callback) {  
  paramsTelegram = {  
    model: 'pt-BR_BroadbandModel',  
    content_type: 'audio/ogg;codecs=opus',  
    continuous: true,  
    interim_results: false  
  };  
  
  var recognizeStream = speech_to_text.createRecognizeStream(paramsTelegram);  
  recognizeStream.setEncoding('utf8');  
  recognizeStream.on('results', function(data) {  
    if (data && data.results && data.results.length > 0 && data.results[0].alternatives[0].transcript) {  
      var result = data.results[0].alternatives[0].transcript;  
      callback(null, result);  
    }  
  });  
};
```

6 – Hands On

- Copie os seguintes arquivos para o seu projeto (**app.js**, **package.json**), e a pasta “**app**”.
- Em **/app/config.js** vamos configurar suas Credenciais e o Conversation Workspace.
 - Para conseguir suas credencias, abra o seu serviço, na aba Service Credentials:

6 – Hands On

[←](#) Watson

Conversation-fm

Manage

Service credentials

Connections

Service credentials

Credentials are provided in JSON format. The JSON snippet lists credentials, such as the API key and secret, as well as connection information for the service.

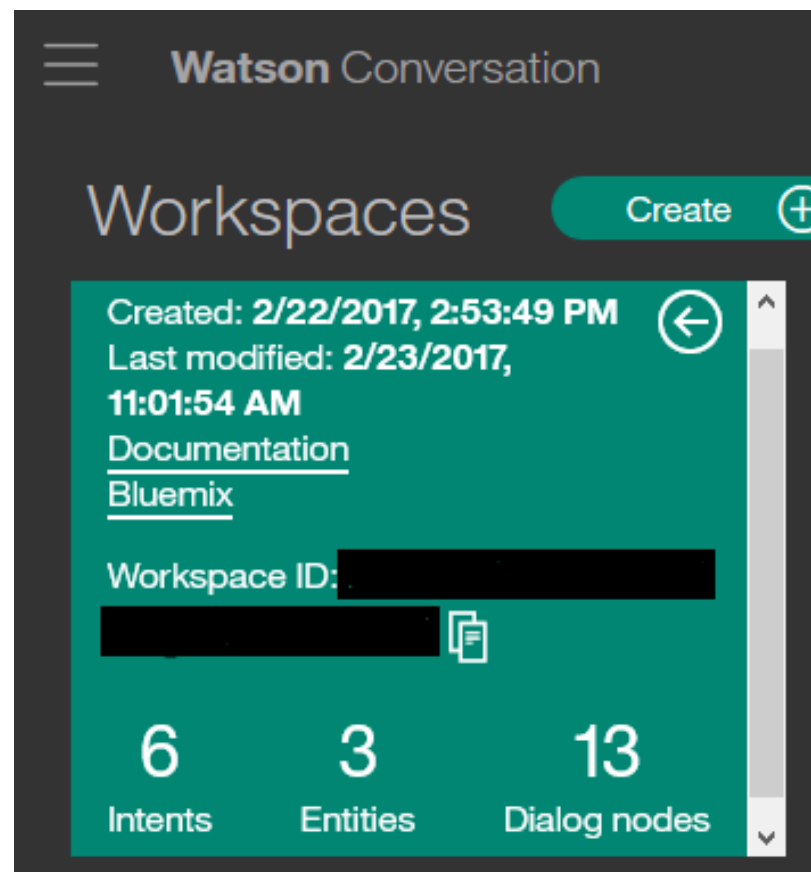
Service credentials

New credential +

<input type="checkbox"/> KEY NAME	DATE CREATED	ACTIONS
<input type="checkbox"/> Credentials-1	Mar 30, 2017 - 02:01:34	View credentials ▲

```
{
  "url": "https://gateway.watsonplatform.net/conversation/api",
  "username": "XXXXXXXXXXXXXXXXXXXX",
  "password": "XXXXXXXXXX"
}
```


6 – Hands On



6 – Hands On

```
1  var config = {};  
2  
3  config.conversation = {};  
4  config.texttospeech = {};  
5  config.speechtotext = {};  
6  config.telegram = {};  
7  
8  // Config Conversation  
9  config.conversation.workspace = 'workspace';  
10 config.conversation.username = 'username';  
11 config.conversation.password = 'password';  
12 config.conversation.versionid = '2017-04-04';  
13  
14 // Config Text to Speech  
15 config.texttospeech.username = 'username';  
16 config.texttospeech.password = 'password';  
17 config.texttospeech.voice = "pt-BR_IsabelaVoice";  
18  
19 // Config Speech to Text  
20 config.speechtotext.username = 'username';  
21 config.speechtotext.password = 'password';  
22  
23 // Telegram apikey  
24 config.telegram.apikey = "apikey";  
25  
26 module.exports = config;
```

6 – Hands On

- Agora você pode fazer o push do seu app para o Bluemix.
- Abra o cmd, vá até o diretório do seu projeto, e digite '**cf push**' se estiver tudo certo, seu app estará rodando no, agora basta testar no seu aplicativo do Telegram.