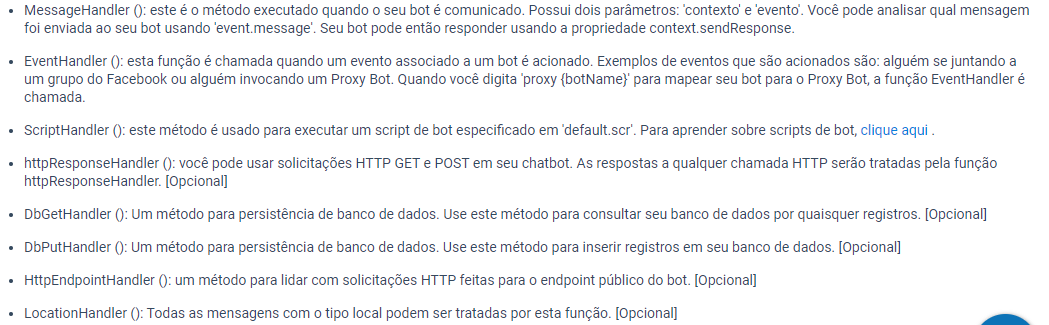
**GUPSHUP:**

A plataforma GupShup é uma plataforma para envio de mensagens automáticas que suporta vários tipos de plataformas de mensagens, entre elas as principais WhatsApp e Facebook.

O projeto consiste em uma API que será desenvolvida por mim e irá comunicar com a plataforma GUPSHUP e está enviará as mensagens via WhatsApp aos usuários, sendo assim temos o seguinte formato de comunicação: API > GUPSHUP > Cliente via WhatsApp.

**Métodos fornecidos pelo BOT Builder:**



**Consultando informações do usuário que se comunica com o chatBot:**

Essas informações podem ser acessadas pelo link: <https://www.gupshup.io/developer/docs/bot-platform/guide/different-objects-sent-to-bot>

Mas entre elas temos algumas principais:

**event.message:**

Retorna a mensagem do usuário para o bot. Pode ser um URL de texto ou imagem ou URL de localização, etc.

**event.sender:**

Retorna o ID do usuário no canal de mensagens, no nosso caso do WhatsApp retorna o numero do celular.

**event.senderobj.display:**

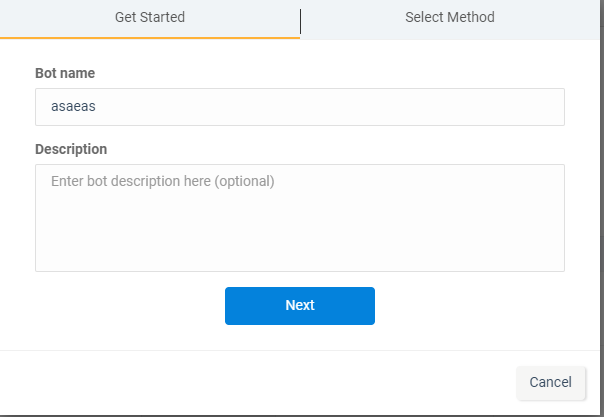
Retorna o nome do usuário no canal de mensagens.

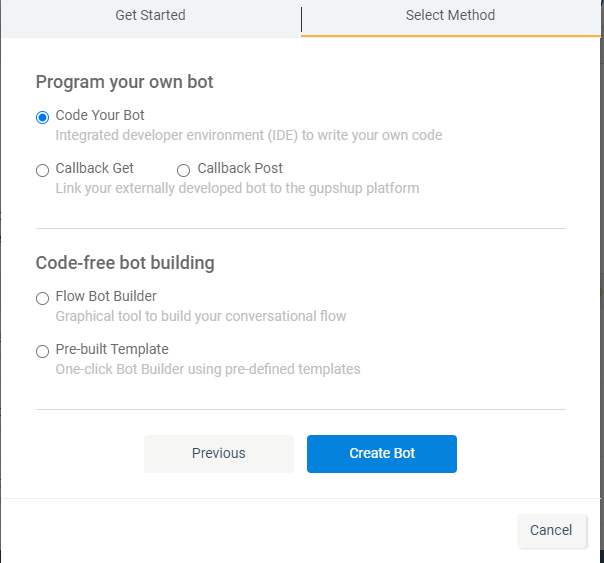
**event.messageobj.address:**

Retorna o endereço do usuário

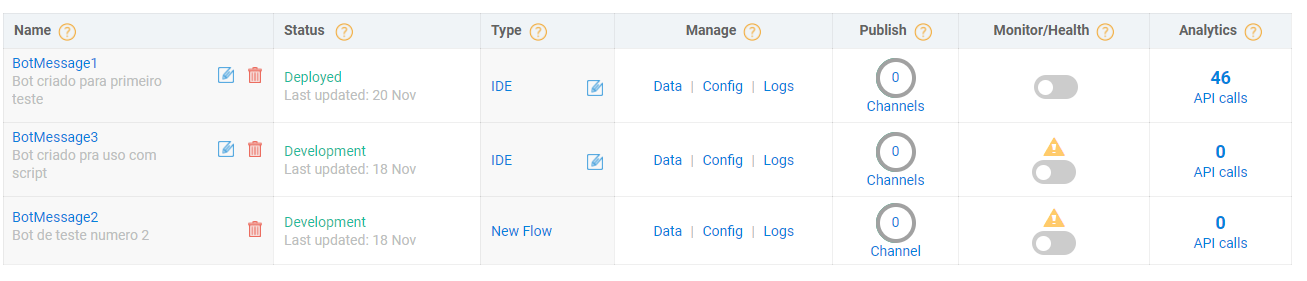
**Adicionando um novo BOT:**

Para isso basta acessar a guia DatshBoard > MyBots > Sinal de mais

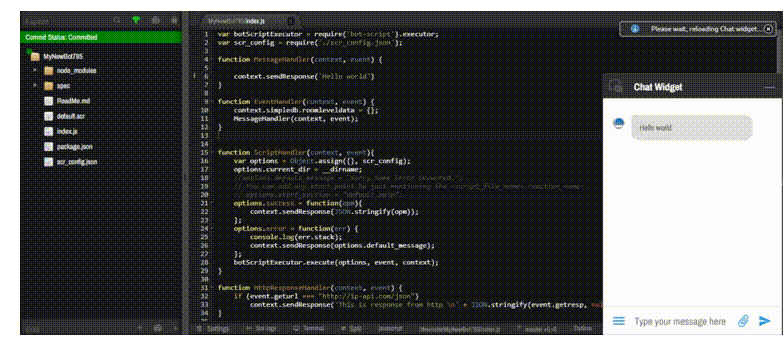




Após isso basta acessar a IDE do seu novo bot:



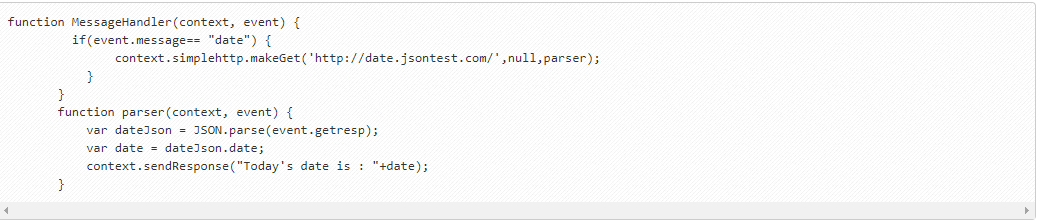
**Criando um Hello Word com nosso BOT:**



Após isso, basta clicar com o botão direito na pasta com o nome do BOT, e em seguida em Start Server.

**Requisições HTTP:**

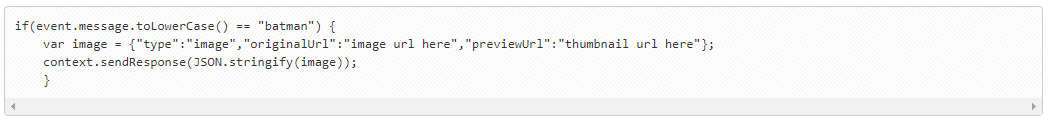
Para fazermos uma requisição HTTP, devemos seguir a seguinte sintaxe:



Nos parâmetros passados a requisição temos o endereço da quisição, em seguida a chave de segurança da API(se necessária) e em seguida uma função de call-back no nosso caso essa função se chama “parser”.

**Envio de mídias via BOT:**

Para enviar um arquivo seja ele de qualquer formato, devemos criar um objeto para enviarmos como parâmetro na chamada da função *context.sendResponse(),* confira o exemplo:



Este objeto deve conter obrigatoriamente os campos “type” e “url”.

**Envio de várias mensagens juntas:**

Para enviar varias mensagens de uma vez basta criar um Array com todas as mensagens que desejamos enviar, e passar esse Array como parâmetro de chamada da função contexto.sendResponse().



**Detalhes da construção da API de respostas a ser requisitada pelo GUPSHUP:**

A aplicação é composta por um arquivo de dados onde se encontram todas as respostas a serem enviadas pelo BOT, esse arquivo é organizado da seguinte forma:

Um Array contendo todos os fluxos, em cada posição de fluxo temos um novo Array contendo três posições, na primeira todas as possíveis mensagens que o usuário enviará, na segunda posição a resposta e na terceira o redirecionamento para o próximo fluxo ( que é a posição do array anterior referente ao BD – Que será usado pra um PUT no banco de dados, para indicar o fluxo que o BOT se encontra, ou seja, a posição do array no arquivo de dados).

**Banco de dados:**

O banco de dados da aplicação é composto por 3 principais tabelas:

* **Usuário:**

Contendo o nome do usuário, seu ID na plataforma de conversa, endereço, CEP, bairro, fluxo no qual usuário se encontra, fluxo anterior, e usaremos a data do último update para verificar atendimentos encerrados e em andamento.

* **Pedidos (Opcional):**

Contendo o ID do usuário na plataforma de conversa, pedido, usaremos essa tabela para o usuário consultar seu último pedido.

**Rotinas de funcionamento:**

* **ChatBOT comercial da cardaps:**

- Apresentação (Usar variável nome, fornecida pela GUPSHUP)

-Enviar menu contendo:

-Informações sobre contratação do serviço

-Simulação de atendimento

-Opção de falar com um vendedor