

# Desenvolvedores

## Como usar

---

### Pré-requisitos

- Crie uma Conta em **NotificaMe Hub**.
- **Node.js**.
- Gere um token de API no console da API NotificaMe Hub.
- Use o ID de usuário da sua conta como identificador do remetente ao enviar qualquer mensagem. Você pode encontrá-lo na plataforma **NotificaMe Hub**.

### SDK

Facilite o uso da nossa API incorporando nosso SDK diretamente em seu software.

Nossa biblioteca auxiliar está disponível na linguagem de programação **Node.js**.

### Instalação

```
npm install notificamehubsdk
```

terminal

### Importar biblioteca

js

```
// ES5
var notificamehub = require('notificamehubsdk');

// ES6 ou Typescript
import * as notificamehub from 'notificamehubsdk';
```

## Autenticação

O token de API é essencial para realizar as requisições na API.

Você pode encontrá-lo acessando sua conta **NotificaMe Hub** na página do Dashboard, ao lado da mensagem de **Bem-Vindo**.

js

```
// Inicialização com o seu token de API (x-api-token)
const client = new notificamehub.Client('SEU_TOKEN_DE_API');
```

## Exemplos

---

js

```
// Escolhendo o canal
const whatsapp = client.setChannel('whatsapp');

// Criando um conteúdo de texto
const content = new notificamehub.TextContent('mensagem de texto aqui');

// ES6
whatsapp.sendMessage('identificador-do-remetente', 'identificador-do-dest')
  .then(response => {
    // código...
  })
  .catch(error => {
    // manipule o erro aqui
  })
```

```
});

// ES8 ou Typescript. NodeJS 7.6.0 ou superior
try {
    const response = await whatsapp.sendMessage('identificador-do-remeter
    // código...
} catch (error) {
    // manipule o erro aqui
}
```

## Enviar mensagens

- Tipos de conteúdo que podem ser enviados:

Nome	Descrição
TextContent	Usado para enviar mensagens de texto.
FileContent	Usado para enviar mensagens de arquivo.
LocationContent	Usado para enviar mensagens de localização.
ContactsContent	Usado para enviar mensagens de contatos.
TemplateContent	Usado para enviar mensagens de modelo.
EmailContent	Usado para enviar mensagens de e-mail.
ReplyableTextContent	Usado para enviar mensagens de texto respondíveis.

- Tipos de conteúdos compatíveis por canal:

Tipos compatíveis	WhatsApp	Facebook	Instagram	Telegram	SMS	Email
TextContent	X	X	X	X	X	

FileContent	X	X	X	X
LocationContent	X			
ContactsContent	X			
TemplateContent	X			X
EmailContent				X
CardContent		X	X	X
CarouselContent		X	X	
ReplayableTextContent		X	X	X

- Para enviar uma mensagens, use o método `sendMessage` . Por exemplo:

js

```
// Enviando uma mensagem de texto através do canal SMS

// Importação
const { Client, TextContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Escolhendo o canal
const sms = client.setChannel('sms');

// Enviando a mensagem
const content = new TextContent('Sua mensagem de texto...');

// Coloque o token do seu canal da API, o destinatário e o conteúdo
sms.sendMessage('SEU_CANAL_DA_API', 'identificador-do-destinatário', content)
  .then(response => {
    console.log('Response:', response);
  })
```

```
.catch(error => {  
  console.log('Error:', error);  
});
```

## Receber mensagens

### Receber mensagens

- Para receber mensagens, use o método `createSubscription` para criar um objeto `MessageSubscription` e inscrever-se para receber mensagens. Por exemplo:

js

```
// Importação  
const { Client, MessageSubscription } = require('notificamehubsdk');  
  
// Inicialização com o seu token de API (x-api-token)  
const client = new Client('SEU_TOKEN_DE_API');  
  
// Cria uma nova assinatura de mensagem com a URL do webhook fornecida co  
const subscription = new MessageSubscription({  
  url: 'https://seu-webhook.company.com'  
},  
{  
  channel: 'SEU_CANAL_DA_API'  
});  
  
// Envia uma solicitação para criar uma assinatura usando o cliente confi  
client.createSubscription(subscription)  
  .then(response => {  
    console.log('Response:', response);  
  })  
  .catch(error => {  
    console.log('Error:', error);  
  });
```

A resposta será um objeto `ISubscription` em caso de sucesso ou um objeto `IError` em caso de erro.

Para descobrir como obter seu token de canal da API, acesse:

### Como obter o token do seu canal da API

#### Receber os estados das mensagens

- Para receber os estados das mensagens, use o método `createSubscription` para criar um objeto `MessageStatusSubscription` e inscrever-se para receber os estados das mensagens. Por exemplo:

```
js
// Importação
const { Client, MessageStatusSubscription } = require('notificamehubsdk')

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Cria uma nova assinatura de estados da mensagem com a URL do webhook
const subscription = new MessageStatusSubscription({
  url: 'https://seu-webhook.company.com'
},
{
  channel: 'SEU_CANAL_DA_API'
});

// Envia uma solicitação para criar uma assinatura usando o cliente confi
client.createSubscription(subscription)
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  });
```

Para descobrir como obter seu token de canal da API, acesse:

## Como obter o token do seu canal da API

### Receber eventos

#### Recebendo eventos de mensagem e eventos de estados de mensagens

Utilize uma classe WebhookController para criar o seu webhook e receber eventos de mensagens e eventos de estados de mensagens. A porta padrão é 3000 .

Caso informe os campos `client` , `url` e `channel` , será criada uma inscrição, a menos que já exista uma inscrição com estas configurações.

No campo `messageEventHandler` serão recebidos os eventos de mensagem.

No campo `messageStatusEventHandler` serão recebidos os eventos de estado da mensagem.

js

```
// Importação
const { Client, WebhookController } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

const webhook = new WebhookController({
  // Recebe os eventos de mensagem
  messageEventHandler: (messageEvent) => {
    console.log('Message event:', messageEvent);
  },
  // Recebe os eventos de estado da mensagem
  messageStatusEventHandler: (messageStatusEvent) => {
    console.log('Message status event:', messageStatusEvent);
  },
  client,
  url: 'https://seu-webhook.company.com',
```

```
channel: 'SEU_CANAL_DA_API',
});

// Inicializa o webhook
webhook.init();
```

Para receber os eventos dos exemplos executados na sua máquina, sugerimos o [ngrok](#).

Para descobrir como obter seu token de canal da API, acesse:

[Como obter o token do seu canal da API](#)

## Gerar relatórios

### Gerar relatórios de mensagens

Para obter informações sobre o número de mensagens enviadas ou recebidas durante um determinado período de tempo, utilize o método `getEntries` para listar objetos `IReportMessagesEntry`. Por exemplo:

```
// Importação
const { Client } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Obter relatório de mensagens
const reportClient = client.getMessageReportClient();
reportClient.getEntries({
  startDate: '2024-01-10',
  endDate: '2024-01-15',
})
.then(response => {
  console.log('Response:', response);
})
.catch(error => {
```

js



```
    console.log('Error:', error);  
  });
```

A resposta pode ser uma matriz de objetos `IReportMessagesEntry` em caso de sucesso ou um objeto `IError` em caso de erro.

## Gerar relatórios de fluxo

Para obter informações sobre o estado atual das sessões (execuções) de fluxos em um determinado período de tempo, utilize o método `getEntries` para listar objetos `IReportFlowEntry`. Por exemplo:

```
// Importação  
const { Client } = require('notificamehubsdk');  
  
// Inicialização com o seu token de API (x-api-token)  
const client = new Client('SEU_TOKEN_DE_API');  
  
// Obter relatório de fluxo  
const reportClient = client.getFlowReportClient();  
reportClient.getEntries({  
  startDate: '2024-01-10',  
})  
  .then(response => {  
    console.log('Response:', response);  
  })  
  .catch(error => {  
    console.log('Error:', error);  
  });
```

js

A resposta pode ser uma matriz de objetos `IReportFlowEntry` em caso de sucesso ou um objeto `IError` em caso de erro.

## Operações em modelos

É possível executar operações CRUD em modelos. Por exemplo, utilizar o método `listTemplates` para listar um objeto `ITemplate`.

js

```
// Importação
const { Client } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

const whatsapp = client.setChannel('whatsapp');

// Lista de modelos
client.listTemplates('SEU_CANAL_DA_API')
  .then(response => {
    console.log('Response:', JSON.stringify(response));
  })
  .catch(error => {
    console.log('Error:', error);
  });
```

A resposta será uma matriz de objetos `ITemplate`.

## Listar canais

Para listar seus canais, utilize o método `listChannels` :

js

```
const { Client } = require('notificamehubsdk');

const client = new Client("SEU_TOKEN_DA_API");

client.listChannels()
  .then(response => {
    console.log('Response:', response);
  });
```

```
    })  
    .catch(error => {  
        console.log('Error:', error);  
    });
```

## WhatsApp

### Tópicos

1. Enviar mensagem de texto
2. Enviar mensagem de áudio
3. Enviar sticker
4. Enviar arquivo
5. Enviar localização
6. Enviar mensagem interativa - Botões
7. Enviar mensagem interativa - Lista
8. Enviar mensagem interativa - Link / CTA
9. Reagir a uma mensagem
10. Listar templates
11. Enviar template - Sem parâmetros
12. Enviar template - Com parâmetros (tipo número/positional)
13. Enviar template - Com parâmetros (tipo nome/named)
14. Fazendo o download de um arquivo criptografado

Enviando uma mensagem de texto:

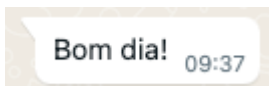
```
// Importação  
const { Client, TextContent } = require('notificamehubsdk');  
  
// Inicialização com o seu token de API (x-api-token)  
const client = new Client('SEU_TOKEN_DE_API');  
  
// Definir o canal  
const whatsapp = client.setChannel('whatsapp');
```

js

```
// Mensagem
const content = new TextContent('Bom dia!');

// O token do seu canal da API
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)
// O conteúdo
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', content
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  }));
```

### Resposta:



js

```
Response: {
  from: 'SEU_CANAL_DA_API',
  to: 'número-do-destinatário',
  contents: '{"type":"text","text":"Bom dia!"}',
  id: 'ID_DA_MENSAGEM',
  direction: 'OUT'
}
```

### Enviando uma mensagem de áudio:

js

```
// Importação
const { Client, FileContent } = require('notificamehubsdk');

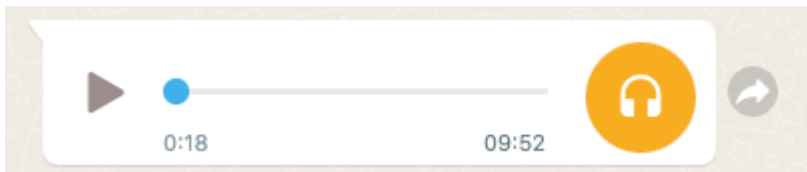
// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');
```

```
// Definir o canal
const whatsapp = client.setChannel('whatsapp');

// Áudio, tipo do arquivo, legenda, nome do arquivo
const content = new FileContent('LINK_DO_AUDIO.mp3', 'audio', 'File Capti

// O token do seu canal da API
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)
// O conteúdo
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', conter
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  }));
```

Resposta:



js

```
Response: {
  from: 'SEU_CANAL_DA_API',
  to: 'número-do-destinatário',
  contents: '{"type":"file","fileUrl":"LINK_DO_AUDIO.mp3","fileMimeType":
  id: 'ID_DA_MENSAGEM',
  direction: 'OUT'
}
```



Enviando um sticker:

js

```
// Importação
const { Client, FileContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const whatsapp = client.setChannel('whatsapp');

// Sticker, tipo do arquivo, legenda, nome do arquivo
const content = new FileContent('LINK_DO_STICKER.webp', 'sticker', 'File

// O token do seu canal da API
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)
// O conteúdo
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', conter
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  })
});
```

Resposta:



js

```
Response: {
  from: 'SEU_CANAL_DA_API',
```

```
to: 'número-do-destinatário',
contents: '{"type":"file","fileUrl":"LINK_DO_STICKER.webp","fileMimeType": "image/webp"}',
id: 'ID_DA_MENSAGEM',
direction: 'OUT'
}
```

## Enviando um arquivo:

js

```
// Importação
const { Client, FileContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const whatsapp = client.setChannel('whatsapp');

// Arquivo (exemplo: pdf, txt, docx, pptx), tipo do arquivo, legenda, nome do arquivo
const content = new FileContent('LINK_DO_ARQUIVO.pdf', 'document', 'FileContent');

// O token do seu canal da API
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)
// O conteúdo
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', content)
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  });
```

## Resposta:



js

```
Response: {
  from: 'SEU_CANAL_DA_API',
  to: 'número-do-destinatário',
  contents: '{"type":"file","fileUrl":"LINK_DO_ARQUIVO.pdf","fileMimeType": "application/pdf"}',
  id: 'ID_DA_MENSAGEM',
  direction: 'OUT'
}
```

### Enviando uma localização:

js

```
// Importação
const { Client, LocationContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const whatsapp = client.setChannel('whatsapp');

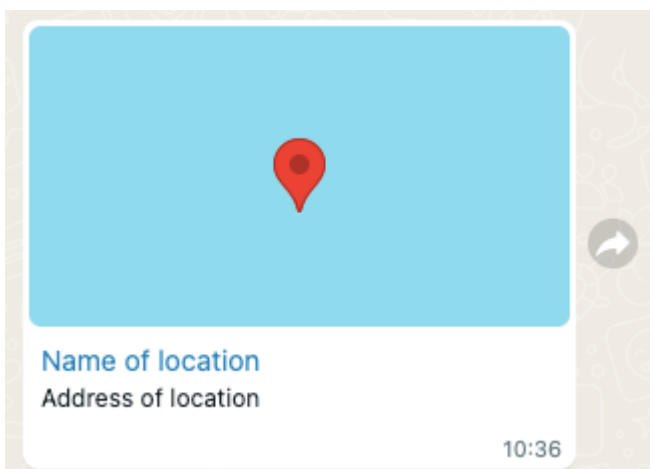
// Longitude, Latitude, Título, Descrição, URL
const content = new LocationContent(
  -48.310882,
  -25.510785,
  'Name of location',
  'Address of location',
  'URL'
);

// O token do seu canal da API
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)
```



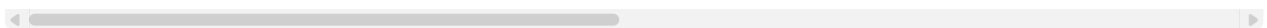
```
// 0 conteúdo
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', conter
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  }));
```

### Resposta:



js

```
Response: {
  from: 'SEU_CANAL_DA_API',
  to: 'número-do-destinatário',
  contents: '{"type":"location","longitude":-48.341675,"latitude":-25.516',
  id: 'ID_DA_MENSAGEM',
  direction: 'OUT'
}
```



### Enviando uma mensagem interativa com botões:

js

```
// Importação
const { Client, InteractiveContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
```

```
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const whatsapp = client.setChannel('whatsapp');

// Tipo, body, botões (até 3)
const content = new InteractiveContent({
  type: "button",
  body: {
    text: "Texto do Botão"
  },
  action: {
    buttons: [
      {
        type: "reply",
        reply: {
          id: "1",
          title: "Botão 1"
        }
      },
      {
        type: "reply",
        reply: {
          id: "2",
          title: "Botão 2"
        }
      },
      {
        type: "reply",
        reply: {
          id: "3",
          title: "Botão 3"
        }
      }
    ]
  }
});

// O token do seu canal da API
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)
```

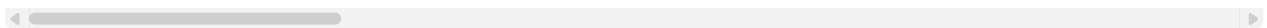
```
// O conteúdo
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', conter
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  }));
```

### Resposta:



js

```
Response: {
  from: 'SEU_CANAL_DA_API',
  to: 'número-do-destinatário',
  contents: '{"type":"interactive","interactive":{"type":"button","body":',
  id: 'ID_DA_MENSAGEM',
  direction: 'OUT'
}
```



### Enviando uma mensagem interativa com listas:

js

```
// Importação
const { Client, InteractiveContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
```

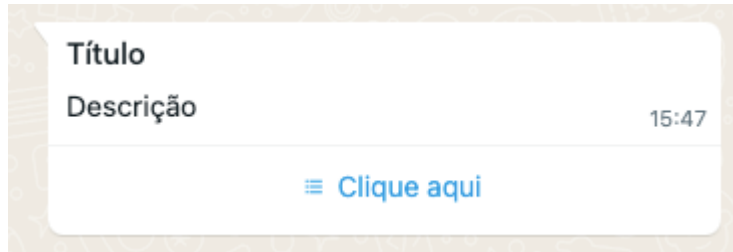
```
const whatsapp = client.setChannel('whatsapp');

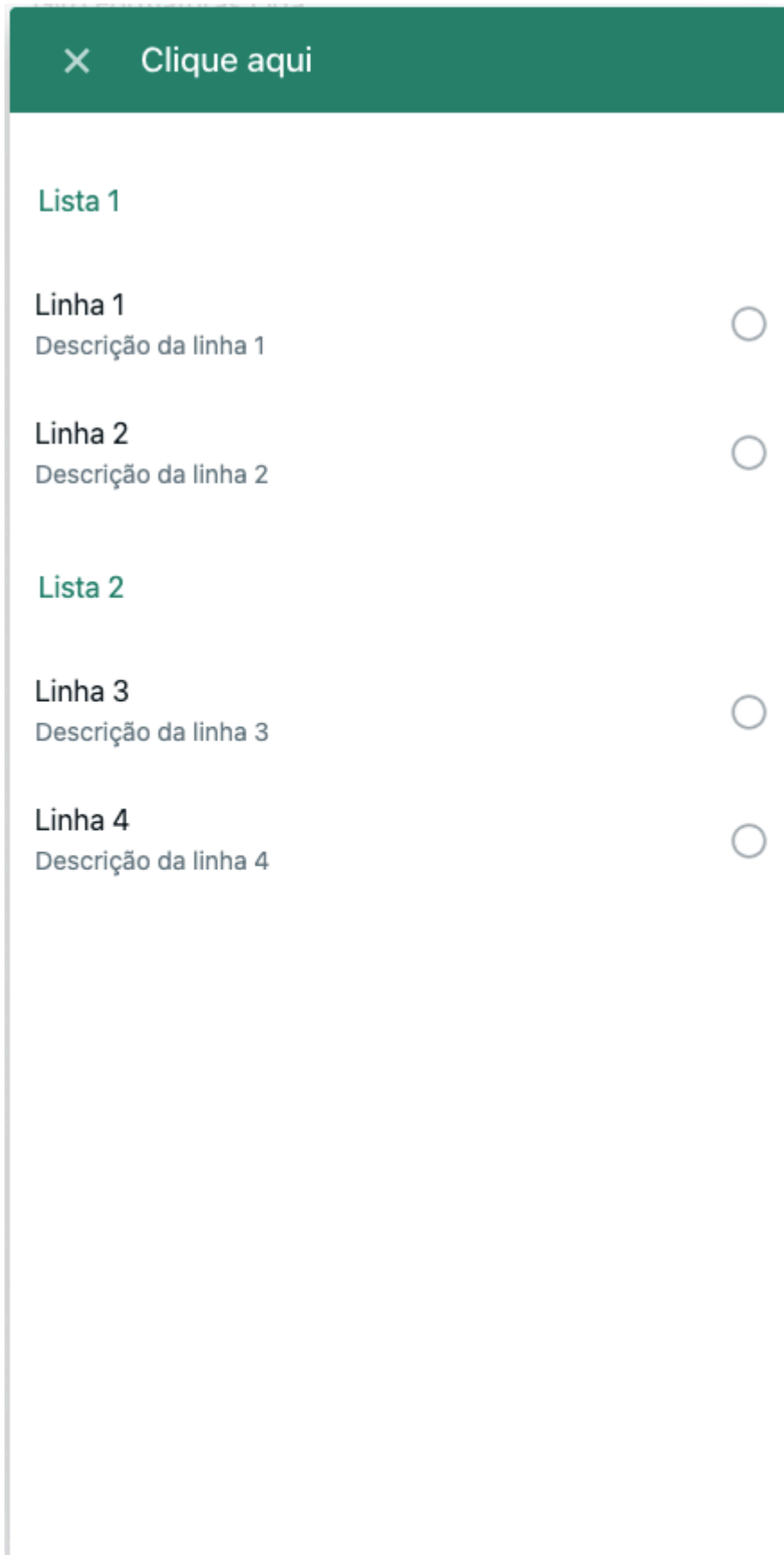
// Tipo, header, body, listas (linhas com id, título e descrição), botão
const content = new InteractiveContent({
  type: "list",
  header: {
    type: 'text',
    text: 'Título',
  },
  body: {
    text: "Descrição"
  },
  action: {
    sections: [
      {
        title: "Lista 1",
        rows: [
          {
            id: 1,
            title: "Linha 1",
            description: "Descrição da linha 1"
          },
          {
            id: 2,
            title: "Linha 2",
            description: "Descrição da linha 2"
          }
        ]
      },
      {
        title: "Lista 2",
        rows: [
          {
            id: 3,
            title: "Linha 3",
            description: "Descrição da linha 3"
          },
          {
            id: 4,
            title: "Linha 4",
```

```
        description: "Descrição da linha 4"
      }
    ]
  }
],
  button: "Clique aqui"
}
});

// O token do seu canal da API
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)
// O conteúdo
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', conter
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  })
});
```

Resposta:





js

```
Response: {  
  from: 'SEU_CANAL_DA_API',  
  to: 'número-do-destinatário',
```

```
    contents: '{"type":"interactive","interactive":{"type":"list","header":  
    id: 'ID_DA_MENSAGEM',  
    direction: 'OUT'  
  }  
}
```

## Enviando uma mensagem interativa com link (Call To Action):

js

```
// Importação  
const { Client, InteractiveContent } = require('notificamehubsdk');  
  
// Inicialização com o seu token de API (x-api-token)  
const client = new Client('SEU_TOKEN_DE_API');  
  
// Definir o canal  
const whatsapp = client.setChannel('whatsapp');  
  
// Tipo, body, cta (call to action)  
const content = new InteractiveContent({  
  type: "cta_url",  
  header: {  
    type: "text",  
    text: "Datas Disponíveis"  
  },  
  body: {  
    text: "Toque no botão abaixo para ver as datas disponíveis."  
  },  
  footer: {  
    text: "Datas sujeitas a alterações."  
  },  
  action: {  
    name: "cta_url",  
    parameters: {  
      display_text: "Ver Datas",  
      url: "SEU_LINK"  
    }  
  }  
});
```



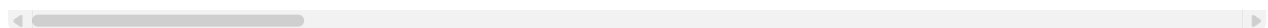
```
// O token do seu canal da API
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)
// O conteúdo
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', conter
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  }));
```

Resposta:



js

```
Response: {
  from: 'SEU_CANAL_DA_API',
  to: 'número-do-destinatário',
  contents: '{"type":"interactive","interactive":{"type":"cta_url","head
  id: 'ID_DA_MENSAGEM',
  direction: 'OUT'
}
```



Reagir a uma mensagem:

js

```
// Importação
const { Client, ReactionContent } = require('notificamehubsdk');

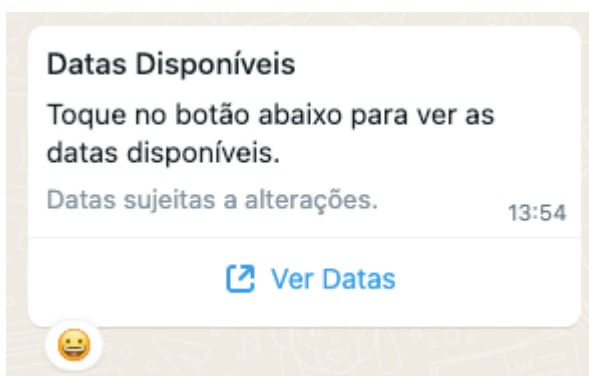
// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');
```

```
// Definir o canal
const whatsapp = client.setChannel('whatsapp');

// MESSAGE_ID e emoji
const content = new ReactionContent({
  message_id: "wamid.MESSAGE_ID",
  emoji: "\uD83D\uDE00"
});

// O token do seu canal da API
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)
// O conteúdo
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', content)
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  });
```

Resposta:



js

```
Response: {
  "messaging_product": "whatsapp",
  "contacts": [{
    "input": "número-do-destinatário",
    "wa_id": "wa-id-do-destinatário"
  }],
}
```

```
"messages": [{  
  "id": "wamid.MESSAGE_ID"  
}]  
}
```

## Listar templates:

js

```
// Importação  
const { Client } = require('notificamehubsdk');  
  
// Inicialização com o seu token de API (x-api-token)  
const client = new Client('SEU_TOKEN_DE_API');  
  
// Coloque o token do seu canal da API  
client.getTemplate('SEU_CANAL_DA_API')  
  .then(response => {  
    console.log('Response:', response);  
  })  
  .catch(error => {  
    console.log('Error:', error);  
  });
```

js

```
Response: {  
  data: [  
    {  
      name: 'NOME_DO_TEMPLATE',  
      components: [Array],  
      language: 'pt_BR',  
      status: 'STATUS',  
      category: 'CATEGORIA',  
      id: 'ID_DO_TEMPLATE'  
    }  
  ]  
}
```

## Enviando um template sem parâmetros:

js

```
// Importação
const { Client, TemplateContent } = require('notificamehubsdk');

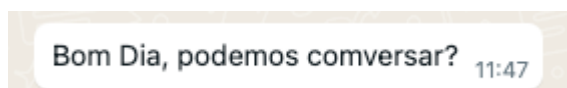
// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const whatsapp = client.setChannel('whatsapp');

// Nome do template, componentes, parâmetros (se possuir), idioma
const content = new TemplateContent({
  name: "nome_do_template",
  components: [{
    type: "body",
    parameters: []
  }],
  language: { "code": "pt_BR" }
});

// O token do seu canal da API
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)
// O conteúdo
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', content)
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  });
```

## Resposta:



js

```
Response: {
  from: 'SEU_CANAL_DA_API',
  to: 'número-do-destinatário',
  contents: '{"type":"template","template":{"name":"nome_do_template"},"cc',
  id: 'ID_DA_MENSAGEM',
  direction: 'OUT'
}
```

### Enviando um template com parâmetros (número):

js

```
// Importação
const { Client, TemplateContent } = require('notificamehubsdk');

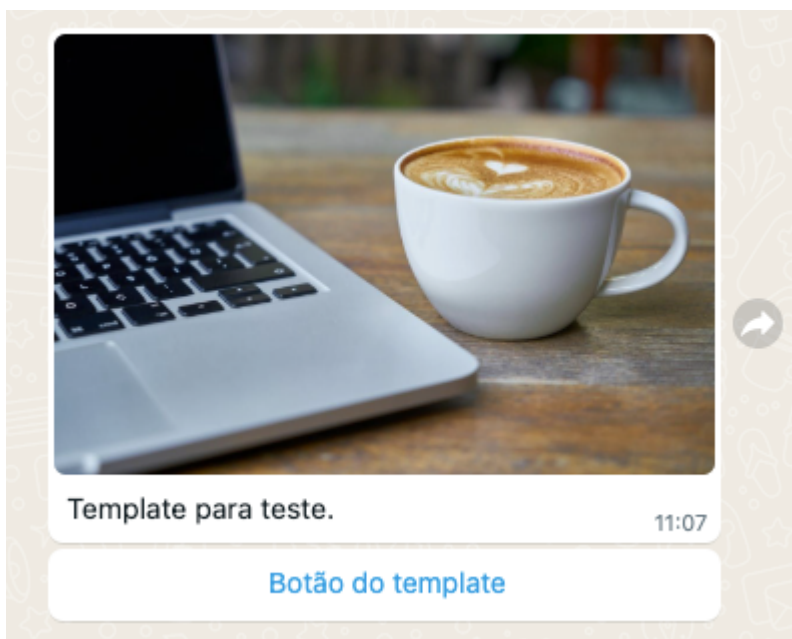
// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const whatsapp = client.setChannel('whatsapp');

// Nome do template, componentes (header, body, button...), parâmetros (s
const content = new TemplateContent({
  name: "nome_do_template",
  components: [
    {
      type: "header",
      parameters: [{
        type: "image",
        image: {
          link: "link_da_imagem"
        }
      }]
    },
    {
      type: 'button',
      sub_type: 'tipo_do_botão',
      index: "index_do_botão (0, 1, 2...)",
      parameters: []
    }
  ]
});
```

```
    },  
  ],  
  language: { "code": "pt_BR" }  
});  
  
// O token do seu canal da API  
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)  
// O conteúdo  
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', {  
  .then(response => {  
    console.log('Response:', response);  
  })  
  .catch(error => {  
    console.log('Error:', error);  
  })  
});
```

Resposta:



js

```
Response: {  
  from: 'SEU_CANAL_DA_API',  
  to: 'número-do-destinatário',  
  contents: '{"type":"template","template":{"name":"nome_do_template"},"cc',  
  id: 'ID_DA_MENSAGEM',
```

```
    direction: 'OUT'  
  }  
}
```

## Enviando um template com parâmetros (nome):

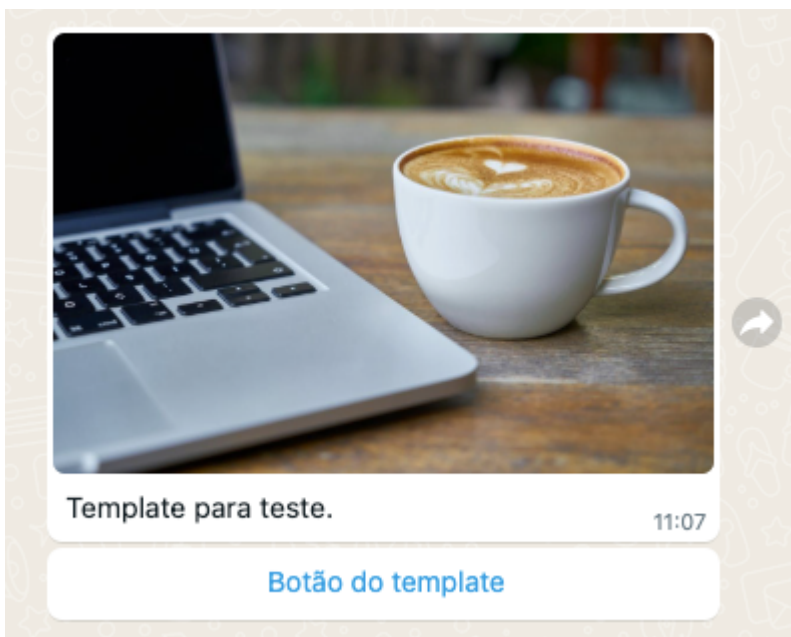
js

```
// Importação  
const { Client, TemplateContent } = require('notificamehubsdk');  
  
// Inicialização com o seu token de API (x-api-token)  
const client = new Client('SEU_TOKEN_DE_API');  
  
// Definir o canal  
const whatsapp = client.setChannel('whatsapp');  
  
// Nome do template, componentes (header, body, button...), parâmetros (s  
const content = new TemplateContent({  
  name: "nome_do_template",  
  components: [  
    {  
      type: "header",  
      parameters: [{  
        type: "image",  
        image: {  
          link: "link_da_imagem"  
        }  
      }]  
    },  
    {  
      type: "body",  
      parameters: [  
        {  
          type: "text",  
          parameter_name: "nome_do_parametro",  
          text: "texto_do_parametro"  
        }  
      ]  
    }  
  ],  
  {  
    type: "button",  
    parameters: [  
      {  
        type: "text",  
        parameter_name: "nome_do_parametro",  
        text: "texto_do_parametro"  
      }  
    ]  
  }  
}]
```

```
    type: 'button',
    sub_type: 'tipo_do_botão',
    index: "index_do_botão (0, 1, 2...)",
    parameters: []
  },
],
language: { "code": "pt_BR" }
});

// O token do seu canal da API
// O usuário de destino (formato de exemplo: +XXYYZZZZZZZZZZ)
// O conteúdo
whatsapp.sendMessage('SEU_CANAL_DA_API', 'número-do-destinatário', conter
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  })
});
```

Resposta:



js

```
Response: {
  from: 'SEU_CANAL_DA_API',
```



```
    to: 'número-do-destinatário',
    contents: '{"type":"template","template":{"name":"nome_do_template"},"cc
    id: 'ID_DA_MENSAGEM',
    direction: 'OUT'
  }
```

### Fazendo o download de um arquivo criptografado:

js

```
// Importação
const { Client, FileContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const whatsapp = client.setChannel('whatsapp');

// Link do arquivo, tipo do arquivo (png, jpeg, pdf, mp4...)
const content = new FileContent('https://lookaside.fbsbx.com/whatsapp_bus

// O token do seu canal da API
// O conteúdo
whatsapp.downloadMedia('SEU_CANAL_DA_API', 'whatsapp', content)
  .then(async response => {
    const fs = require("fs");

    // Nome do arquivo . extensão do arquivo
    let writeStream = fs.createWriteStream('arquivo.jpeg');
    writeStream.write(response, 'binary');
    writeStream.on('finish', () => {
      console.log('Download concluído');
    });
    writeStream.end();
  })
  .catch(error => {
```

```
    console.log('Error:', error);  
  });
```

### Resposta:

js

```
{  
  from: 'SEU_CANAL_DA_API',  
  to: 'whatsapp',  
  contents: [  
    FileContent {  
      type: 'file',  
      fileUrl: 'https://lookaside.fbsbx.com/whatsapp_business/attachments',  
      fileMimeType: 'jpeg',  
      fileCaption: undefined,  
      fileName: undefined  
    }  
  ]  
}
```

## Facebook

### Enviando uma mensagem de texto:

js

```
// Importação  
const { Client, TextContent } = require('notificamehubsdk');  
  
// Inicialização com o seu token de API (x-api-token)  
const client = new Client('SEU_TOKEN_DE_API');  
  
// Definir o canal  
const facebook = client.setChannel('facebook');  
  
// Sua mensagem  
const content = new TextContent('Mensagem teste');
```

```
// Coloque o token do seu canal da API, o usuário de destino e o conteúdo
facebook.sendMessage('SEU_CANAL_DA_API', 'conta-do-destinatário', content
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  }));
```

## Enviando uma mensagem com arquivo:

js

```
// Importação
const { Client, FileContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const facebook = client.setChannel('facebook');

// Link do arquivo, tipo do arquivo, descrição, nome do arquivo
const content = new FileContent('https://seu-audio.com/audio.mp3', 'audio')

// const content = new FileContent('https://sua-imagem.com/imagem.jpg', 'image')
// const content = new FileContent('https://seu-video.com/video.mp4', 'video')

// Coloque o token do seu canal da API, o usuário de destino e o conteúdo
facebook.sendMessage('SEU_CANAL_DA_API', 'conta-do-destinatário', content
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  }));
```

## Enviando uma mensagem de texto:

js

```
// Importação
const { Client, TextContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const instagram = client.setChannel('instagram');

// Link do arquivo, tipo do arquivo, descrição, nome do arquivo
const content = new TextContent('Mensagem teste');

// Coloque o token do seu canal da API, o usuário de destino e o conteúdo
instagram.sendMessage('SEU_CANAL_DA_API', 'conta-do-destinatário', content)
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  });
```

## Enviando uma mensagem com arquivo:

js

```
// Importação
const { Client, FileContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const instagram = client.setChannel('instagram');

// Link do arquivo, tipo do arquivo, descrição, nome do arquivo
const content = new FileContent('https://sua-imagem.com/imagem.jpg', 'ima
```

```
// const content = new FileContent('https://seu-video.com/video.mp4', 'vi

// Coloque o token do seu canal da API, o usuário de destino e o conteúdo
instagram.sendMessage('SEU_CANAL_DA_API', 'conta-do-destinatário', content
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  }));
```

### Enviando uma mensagem com botões interativos:

js

```
// Importação
const { Client, TemplateContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const instagram = client.setChannel('instagram');

// Conteúdo do template
const content = new TemplateContent({
  template_type: "button",
  text: "Seu texto...",
  buttons: [
    {
      type: "web_url",
      url: "https://notificame.com.br",
      title: "Botão 1"
    },
    {
      type: "web_url",
      url: "https://notificame.com.br",
      title: "Botão 2"
    },
    {

```

```

        type: "postback",
        title: "Começar chat",
        payload: "DEVELOPER_DEFINED_PAYLOAD"
      }
    ]
  })

```

```

// Coloque o token do seu canal da API, o usuário de destino e o conteúdo
instagram.sendMessage('SEU_CANAL_DA_API', 'conta-do-destinatário', content)
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  });

```

### Fazendo uma postagem no feed:

js

```

// Importação
const { Client, FeedContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const instagram = client.setChannel('instagram');

// Link do arquivo, legenda
const content = new FeedContent('https://sua-imagem.com/imagem.jpg', 'Legenda');

// Coloque o token do seu canal da API, coloque o token do canal novamente
instagram.sendMessage('SEU_CANAL_DA_API', 'SEU_CANAL_DA_API', content)
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {

```

```
    console.log('Error:', error);  
  });
```

## Telegram

Enviando uma mensagem de texto:

js

```
// Importação  
const { Client, TextContent } = require('notificamehubsdk');  
  
// Inicialização com o seu token de API (x-api-token)  
const client = new Client('SEU_TOKEN_DE_API');  
  
// Definir o canal  
const telegram = client.setChannel('telegram');  
  
// Sua mensagem  
const content = new TextContent('Teste NotificaMe');  
  
// Coloque o token do seu canal da API, o usuário de destino e o conteúdo  
telegram.sendMessage('SEU_CANAL_DA_API', 'conta-do-destinatário', content  
  .then(response => {  
    console.log('Response:', response);  
  })  
  .catch(error => {  
    console.log('Error:', error);  
  }));
```

Enviando uma mensagem com arquivo:

js

```
// Importação  
const { Client, FileContent } = require('notificamehubsdk');  
  
// Inicialização com o seu token de API (x-api-token)  
const client = new Client('SEU_TOKEN_DE_API');
```

```
// Definir o canal
const telegram = client.setChannel('telegram');

// Insira seu arquivo, o tipo do arquivo e uma descrição
const content = new FileContent('https://hub.notificame.com.br/assets/img

// Coloque o token do seu canal da API, o usuário de destino e o conteúdo
telegram.sendMessage('SEU_CANAL_DA_API', 'conta-do-destinatário', content
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  }));
```

## E-mail

Para enviar um e-mail basta seguir o passo a passo do código:

js

```
// Importação
const { Client, EmailContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const email = client.setChannel('email');

// Para enviar anexos, basta criar uma lista de objetos com os arquivos c
const Anexos = [
  {
    fileUrl: "https://filesamples.com/samples/image/jpg/s
    fileName: "sample_640x426.jpg"
  }, {
    fileUrl: "https://freetestdata.com/wp-content/uploads
    fileName: "Free_Test_Data_117KB_JPG.jpg"
```



```

    }
  ]

// Monte a estrutura do E-mail
const content = new EmailContent(
    subject = "Assunto",
    attachments = Anexos, // Caso não envie
    html = "<!DOCTYPE html> <html> <body> <
);

// Coloque o token do seu canal da API, o email de destino e o conteúdo
email.sendMessage('SEU_CANAL_DA_API', 'email@email.com', content)
.then(response => {
    console.log('Response:', response);
})
.catch(error => {
    console.log('Error:', error);
});

```

## WebChat

Enviando uma mensagem de texto:

```

// Importação
const { Client, TextContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const webchat = client.setChannel('webchat');

// Mensagem
const content = new TextContent('Mensagem teste');

// Coloque o token do seu canal da API, o usuário de destino e o conteúdo
webchat.sendMessage('SEU_CANAL_DA_API', 'id-do-destinatário', content)
    .then(response => {

```

```
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  });
```

### Enviando uma mensagem com arquivo:

js

```
// Importação
const { Client, FileContent } = require('notificamehubsdk');

// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const webchat = client.setChannel('webchat');

// Link do arquivo, tipo do arquivo, descrição, nome do arquivo
const content = new FileContent('https://hub.notificame.com.br/assets/img

// Coloque o token do seu canal da API, o usuário de destino e o conteúdo
webchat.sendMessage('SEU_CANAL_DA_API', 'id-do-destinatário', content)
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  });
```



## Mercado Livre

### Responder comentário:

js

```
// Importação
const { Client, TextContent } = require('notificamehubsdk');
```

```
// Inicialização com o seu token de API (x-api-token)
const client = new Client('SEU_TOKEN_DE_API');

// Definir o canal
const mercadolive = client.setChannel('mercadolive');

// Mensagem da resposta
const content = new TextContent('Resposta');


// Coloque o token do seu canal da API, o comentário de destino e o conteúdo
mercadolive.sendMessage('SEU_CANAL_DA_API', 'id-do-comentário', content)
  .then(response => {
    console.log('Response:', response);
  })
  .catch(error => {
    console.log('Error:', error);
  });
```

## Canais

---

### Como obter o token do seu canal da API

1. Acesse sua conta do **NotificaMe Hub**.
2. Conecte-se a um canal seguindo os passos de um dos tutoriais abaixo na área de Usuários.
  - [WhatsApp](#)
  - [Facebook](#)
  - [Instagram](#)
  - [Telegram](#)
  - [SMS](#)
  - [E-mail](#)
3. Depois de conectar um canal, na página do Dashboard, procure pela seção **Canais Conectados**.

4. Clique nos  no canto superior direito do painel do canal desejado.
5. Com o menu aberto, clique em **Editar**.
6. Ao lado de "Editar URL de retorno", está o seu token do canal. Copie-o e utilize para definir o canal em sua API.

## Whatsapp

O canal do WhatsApp pode ser utilizado após a sua ativação na plataforma **NotificaMe Hub**.

Para ativar o WhatsApp, você precisa de uma conta na plataforma do NotificaMe Hub e de um número registrado no WhatsApp Business API.

Entre em contato com os consultores do NotificaMe Hub para criar a sua conta.

### Limitações

A API do WhatsApp possui algumas limitações:

1. Para iniciar uma conversa com alguém, você precisa de um tipo específico de mensagem chamado mensagem de modelo, que requer aprovação antes de ser usada.
2. Ao enviar imagens PNG com fundo transparente, você pode obter um resultado final inesperado devido ao processamento de imagem realizado pelo WhatsApp para converter a imagem em JPEG.
3. Mensagens que não são do tipo conteúdo de modelo só podem ser entregues dentro de uma janela de 24 horas desde a última mensagem enviada pelo cliente para a empresa.

Mídia	Tipo de Conteúdo	MIME Type	Tamanho da Mídia Após Processamento
-------	------------------	-----------	-------------------------------------

Documento	.txt	text/plain	25 MB
	.xls	application/vnd.ms-excel	
	.xlsx	application/vnd.openxmlformats-officedocument.spreadsheetml.sheet	
	.doc	application/msword	
	.docx	application/vnd.openxmlformats-officedocument.wordprocessingml.document	
	.ppt	application/vnd.ms-powerpoint	
	.pptx	application/vnd.openxmlformats-officedocument.presentationml.presentation	
	.pdf	application/pdf	
Imagem	.jpeg	image/jpeg	5 MB
	.png	image/png	
Adesivos	.webp	image/webp	500 KB
Áudio	.acc	audio/acc audio/mp4 audio/amr audio/mpeg audio/ogg	16 MB
	.m4a		
	.amr		
	.mp3		
	.ogg		
	Obs: O tipo base de áudio/ogg não é suportado.		
	Vídeo	.mp4 .3gp	
Obs: - Apenas o codec de vídeo H.264 e o codec de			

áudio AAC  
são  
suportados.  
- Apenas  
vídeos com  
uma única  
faixa de  
áudio são  
suportados.

O tamanho do arquivo de mídia após ser criptografado. O limite máximo para o tamanho de arquivos de mídia que podem ser carregados é de 64MB.

### Remetente e destinatário do WhatsApp

- Quando você envia uma mensagem para um contato usando o canal do WhatsApp:

**Destinatário:** o número de telefone do contato.

**Remetente:** o ID do remetente do WhatsApp registrado na plataforma NotificaMe Hub.

- Quando você recebe uma mensagem de um contato, o remetente e o destinatário são invertidos:

**Destinatário:** ID do remetente do WhatsApp registrado na plataforma NotificaMe Hub.

**Remetente:** o número de telefone do contato.

O remetente está no valor da chave `from` e o destinatário está no valor da chave `to` do objeto.

## Facebook

Para usar o canal do Facebook, é necessário conectar uma página na plataforma **NotificaMe Hub**.

### Limitações do Facebook

Antes de enviar mensagens para um contato, é preciso configurar um webhook para receber eventos em uma URL configurada.

As limitações de tipo de conteúdo e tamanho de mídia da API do Facebook para o envio:

Mídia	Tipo do Conteúdo	MIME Type	Tamanho da Mídia
Vídeo (reel/story)	.mp4 <b>Obs:</b> 9:16, 24 a 60 FPS, duração 3 a 9 segundos, um reel publicado como um story em uma Página do Facebook pode ter no máximo 60 segundos, recomendado 1080 x 1920 pixels, codec de vídeo: H.264, H.265, áudio AAC, taxa de 48kHz	video/mp4	25 MB
Imagem	.jpeg .png .jpg	image/jpeg image/png image/jpg	10 MB
Vídeo	.mp4	video/mp4	16 MB
Documento	.pdf	application/pdf	25 MB

### Remetente e destinatário no Facebook

- Quando você recebe uma mensagem de um contato pelo canal do Facebook:

**Destinatário:** ID da sua página do Facebook.

**Remetente:** ID do contato na sua página do Facebook (PSID - ID de escopo de página).

- Quando você envia uma mensagem para um contato, o remetente e o destinatário são invertidos:

**Destinatário:** ID do contato na sua página do Facebook (PSID - ID de escopo de página).

**Remetente:** ID da sua página do Facebook.

O remetente está no valor da chave `from` e o destinatário está no valor da chave `to` do objeto.

## Instagram

O canal do Instagram pode ser usado após a sua ativação na plataforma **NotificaMe Hub**.

Entre em contato com os consultores da NotificaMe Hub para conectar a sua conta.

### Limitações do Instagram

1. Antes de enviar mensagens para um contato, é preciso configurar um webhook para receber eventos em uma URL configurada.
2. A janela de resposta é de 24 horas, com exceção quando um agente humano está respondendo ao contato, então a janela de resposta é aumentada para 7 dias.

As limitações de tipo de conteúdo e tamanho de mídia da API do Instagram para o envio:

Mídia	Tipo do Conteúdo	MIME Type	Tamanho da Mídia
Vídeo (reel)	.mp4	video/mp4	25 MB
	.mov	video/mov	
	<b>Obs:</b> áudio AAC, máxima de 48kHz, 1 ou 2 canais (mono ou estéreo), de 23 a 60 FPS, codec de vídeo: HEVC ou H.264, duração de 3 segundos até 15 minutos		



Imagem	.jpeg <b>Obs:</b> largura mínima: 320, largura máxima: 1440, sRGB	image/jpeg	8 MB
Vídeo (story)	.mp4 .mov <b>Obs:</b> áudio AAC, máxima de 48kHz, 1 ou 2 canais (mono ou estéreo), de 23 a 60 FPS, codec de vídeo: HEVC ou H.264, duração de 3 segundos até 60 segundos	video/mp4 video/mov	25 MB
Imagem (story)	.jpeg <b>Obs:</b> recomendado 9:16, sRGB	image/jpeg	8 MB

### Remetente e destinatário do Instagram

- Quando você recebe uma mensagem de um contato do canal do Instagram:

**Destinatário:** seu ID de conta do Instagram (não é o seu @ de conta).

**Remetente:** o ID do contato em sua conta (não o @ do contato e ele será diferente entre contas).

- Quando você envia uma mensagem para um contato, o remetente e o destinatário são invertidos:

**Destinatário:** o ID do contato em sua conta (não o @ do contato e ele será diferente entre contas).

**Remetente:** seu ID de conta do Instagram (não é o seu @ de conta).

O remetente está no valor da chave `from` e o destinatário está no valor da chave `to` do objeto.

## Telegram

O Telegram pode ser usado após sua ativação na plataforma **NotificaMe Hub**.

Para ativar o Telegram, você precisa de uma conta habilitada na plataforma NotificaMe Hub e um Bot registrado na API do Telegram Bot.

## Limitações do Telegram

Antes de enviar mensagens para um contato, é necessário configurar um webhook, que permite receber eventos na URL configurada.

### Tipos de conteúdo suportados e os tamanhos:

Mídia	Tipo do Conteúdo	MIME Type	Tamanho da Mídia
Imagem	.jpeg	image/jpeg	5 MB
	.gif	image/gif	
	.png	image/png	
	.webp	image/webp	
Documento	.pdg	application/pdf	20 MB
Vídeo	.mov	video/quicktime	20 MB
	.mp4	video/mp4	
Áudio	.mp3	audio/mpeg	20 MB

### Remetente e destinatário do Telegram

- Quando você recebe uma mensagem de um contato do canal do Telegram:

**Destinatário:** o nome de usuário do bot do Telegram habilitado na plataforma NotificaMe Hub.

**Remetente:** o ID da conversa (não é o número de telefone).

- Quando você envia uma mensagem para um contato, o remetente e o destinatário são invertidos:

**Destinatário:** o ID da conversa (não é o número de telefone).

**Remetente:** o nome de usuário do bot do Telegram habilitado na plataforma NotificaMe Hub.

O remetente está no valor da chave `from` e o destinatário está no valor da chave `to` do objeto.

## SMS

O canal de SMS pode ser usado após a sua ativação na plataforma **NotificaMe Hub**.

Entre em contato com os consultores da NotificaMe Hub para conectar a sua conta.

### Limitações do SMS

1. Um SMS pode ter até 140 bytes (equivalente a 1120 bits). Ele pode usar a codificação padrão, que suporta 160 caracteres (7 bits cada), ou a codificação Unicode, que suporta 70 caracteres (16 bits cada).
2. A codificação padrão, representa uma quantidade limitada de caracteres, enquanto o Unicode oferece uma gama mais ampla. Por exemplo, o Unicode permite a representação de caracteres de diferentes idiomas, símbolos matemáticos e científicos, caracteres especiais e até mesmo emojis:

Ç, ç, Á, É, Í, Ó, Ú, À, È, Ì, Ò, Ù, Â, Ê, Î, Ô, Û, Ä, Ë, Ì, Ö, Ü, Ñ, Ñ, Æ, æ, Š, š, Ž, ž

### Mensagens longas

Para possibilitar o envio de mensagens longas, os fabricantes de celulares criaram uma função chamada `SMS concatenadas`, que divide a mensagem em partes menores e as combina novamente ao serem entregues.

- Esta função não é suportada por todas as operadoras.
  - Grandes operadoras como TIM, Vivo, Claro e Oi possuem suporte para essa função.
  - Algumas operadoras regionais também podem oferecer suporte.
- O tamanho máximo para mensagens longas atualmente é de 1520 caracteres.

- O custo é mais elevado para este tipo de mensagem, uma vez que envolve o envio de várias mensagens.
- A função é automaticamente aplicada a mensagens codificadas em Unicode com mais de 70 caracteres, e a mensagens com codificação padrão com mais de 160 caracteres.
- Para identificar cada parte da mensagem, usamos alguns bytes como um tipo de "cabeçalho". Isso significa que cada parte da mensagem é mais curta do que uma mensagem normal: até 152 caracteres no modo padrão e até 66 caracteres no modo Unicode. Sempre cortamos a mensagem nos "espaços", então algumas partes podem ser ainda menores.
  - Por causa disso, palavras com mais de 152 caracteres na codificação padrão ou mais de 66 caracteres na codificação Unicode não são aceitas.

## Remetente e destinatário do SMS

- Quando você envia uma mensagem para um contato por SMS:

**Destinatário:** código do país + telefone completo.

**Remetente:** a chave da conta de SMS conectada no NotificaMe Hub.

- Quando você recebe uma mensagem de um contato, o remetente e o destinatário são invertidos:

**Destinatário:** a chave da conta de SMS conectada no NotificaMe Hub.

**Remetente:** código do país + telefone completo.

O remetente está no valor da chave `from` e o destinatário está no valor da chave `to` do objeto.



## E-mail

O canal de E-mail pode ser usado após a sua ativação na plataforma **NotificaMe Hub**.

**Tipos de conteúdo suportados e os tamanhos:**

Mídia	Tipo do Conteúdo	Tamanho
Document	Any valid MIME type.	25 MB

O tamanho máximo é igual ao tamanho total do e-mail.

### Remetente e destinatário do E-mail

- Quando você envia uma mensagem para um contato por E-mail:

**Destinatário:** endereço de e-mail do contato.

**Remetente:** endereço de e-mail conectado ao NotificaMe Hub.

- Quando você recebe uma mensagem de um contato, o remetente e o destinatário são invertidos:

**Destinatário:** endereço de e-mail conectado ao NotificaMe Hub.

**Remetente:** endereço de e-mail do contato.

O remetente está no valor da chave `from` e o destinatário está no valor da chave `to` do objeto.

## Códigos de status HTTP

Os métodos HTTP são utilizados para manipular recursos. No entanto, dado que nem todos os recursos permitem todas as operações HTTP, é importante consultar a referência de cada recurso abaixo.

### Códigos HTTP

- Operações realizadas com endpoints de coleção:

Tipo da Requisição	Ação	Status
--------------------	------	--------

GET	Lista os itens da coleção.	Retorna o status HTTP 200 - OK.
POST	Cria um novo item.	Retorna o status HTTP 200 - OK.

- Operações realizadas com endpoints de item:

Tipo da Requisição	Ação	Status
GET	Recupera um item.	Retorna o status HTTP 200 - OK.
DELETE	Exclui um item.	Retorna o status HTTP 204 - Sem conteúdo.
PATCH	Atualiza um item.	Retorna o status HTTP 200 - OK.

Quando uma requisição é executada com sucesso, a API responderá com um código de status HTTP 2xx.

## Códigos de Erro

Código	Mensagem	Status
VALIDATION_ERROR	Validation error	Retorna o status HTTP 400
AUTHENTICATION_ERROR	No authorization token was found	Retorna o status HTTP 401
NOT_FOUND	Not found	Retorna o status HTTP 404

DUPLICATED	Entity already exists	Retorna o status HTTP 409
INTERNAL_ERROR	Internal error	Retorna o status HTTP 500

Quando a resposta da requisição possuir status HTTP 500 e código `INTERNAL_ERROR` a API tentará realizar a requisição novamente.

Quando ocorre um erro em uma requisição, a API responderá com um código de status HTTP 4xx ou 5xx junto com os detalhes do erro em um Objeto.

[< PREVIOUS](#)[Usuários](#)[NEXT >](#)[Desenvolvedores API](#)