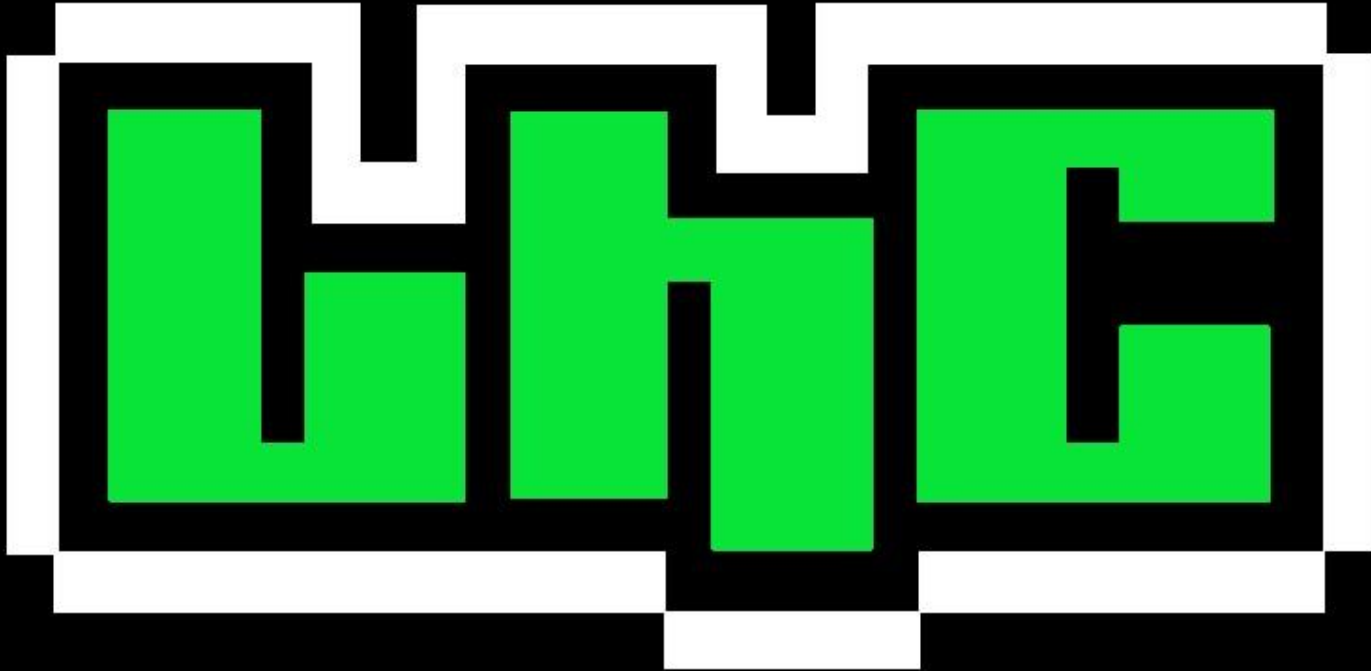


Laboratório Hacker de Campinas



Desenvolvendo BOTS de Telegram com Python
por Rafael Estevam

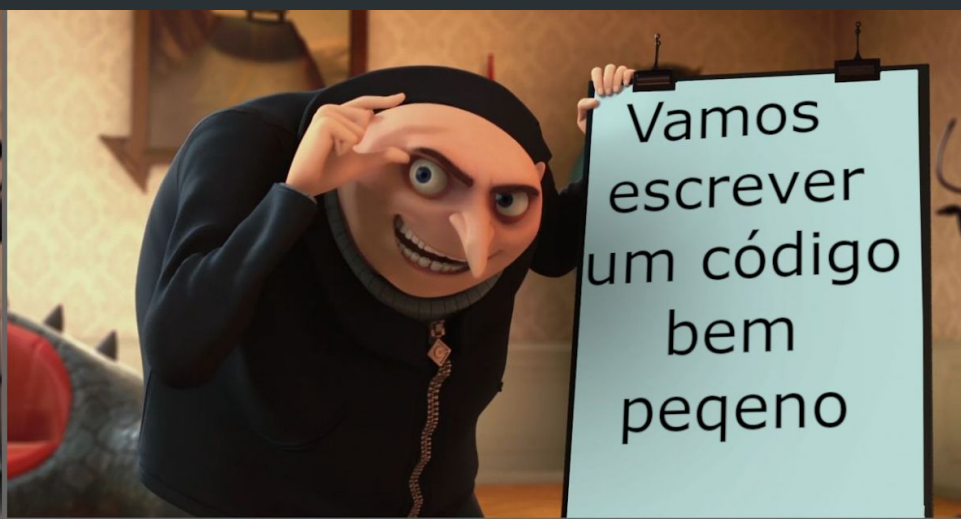
Requisitos

- Python instalado
 - Eu estou usando Python 3
- PIP Instalado
 - Usaremos o pacote python-telegram-bot
<https://python-telegram-bot.org/>
- Conta ativa no App Telegram
 - Só é necessário estar em 1 dispositivo (geralmente o celular), porém será muito prático estar com ele no PC para copiar/colar o Token

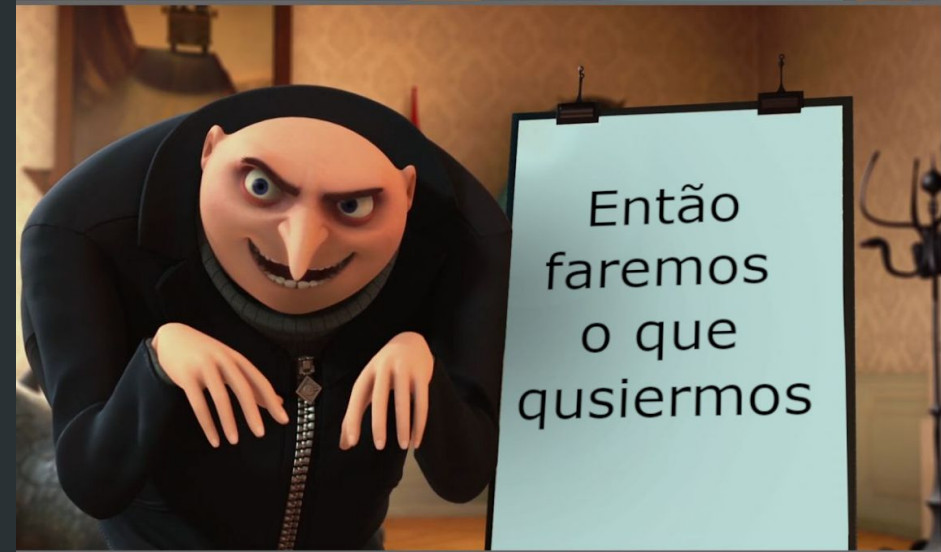
O plano é simples....

A Minion character from the Despicable Me franchise is shown from the chest up. He is wearing his signature black jumpsuit with a silver zipper and a grey turtleneck. He has a large, bulbous nose and a small, balding head. He is holding a white sign with black text. His right hand is raised in a fist, and he has a determined, slightly angry expression.


Falar
com o
BotFather

The Minion character is shown in a similar pose to the first panel, but now he is leaning forward with his right hand on his forehead, looking at the sign with a more intense, almost desperate expression.

Vamos
escrever
um código
bem
pequeno

The Minion character is shown in a similar pose to the first panel, but now he is leaning forward with his hands on the sign, looking at it with a more intense, almost desperate expression.

Então
faremos
o que
quisermos

The Minion character is shown in a similar pose to the first panel, but now he is leaning forward with his hands on the sign, looking at it with a more intense, almost desperate expression.

```
File "bot.py",  
    line 42  
print("User");  
    ^  
SyntaxError:  
PythonDon'tHavePontoVirgula
```

BotFather ?

Para seu BOT se comunicar com o Telegram ele precisa de uma autorização
Esta autorização se chama **TOKEN**

Para conseguir um Token você precisa pedir ele a um
BOT especial, o @BotFather

Para isso vá no seu Telegram e procure pelo @BotFather

E numa conversa com ele mande o comando /start

Ele vai se apresentar e então você pode pedir à ele
Para que adicione um novo BOT



Como pedir um BOT ao BotFather

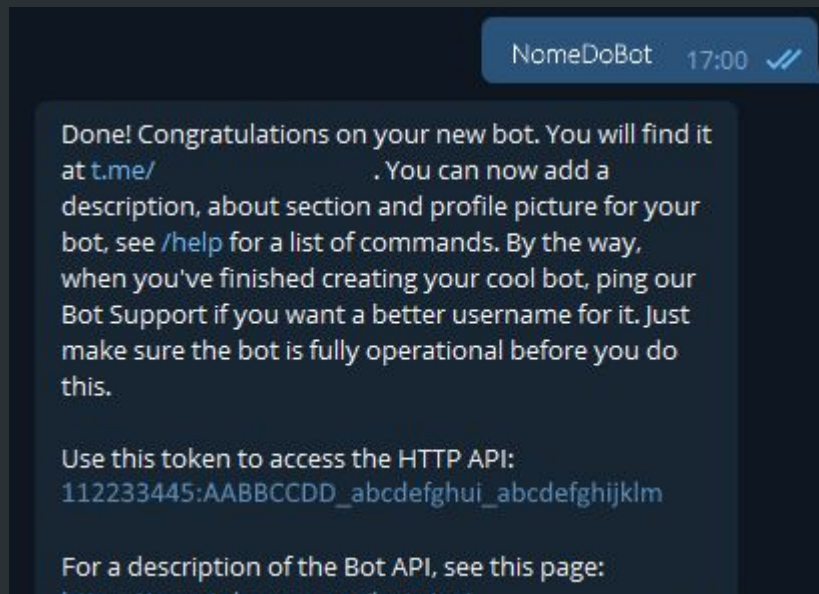
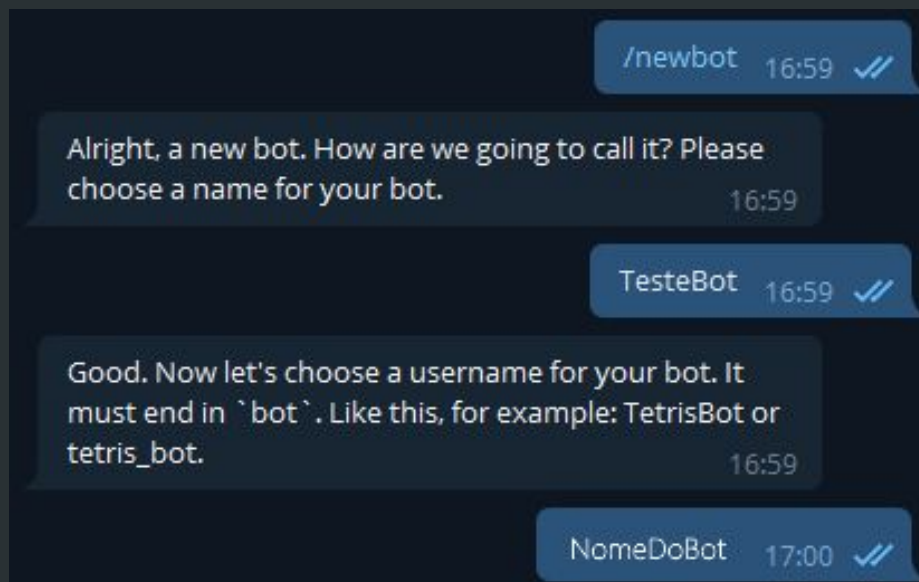
Ao dar /start no BotFather ele te dará várias opções para criar e controlar seu BOT. A opção que usaremos é /newbot

Ao utilizar o comando /newbot será solicitado o nome do BOT. Escolha um nome que achar bacana

Depois será solicitado um UserName para o BOT, este UserName deve ser único e deve terminar com as letras “bot”

Se tudo der certo ele o BotFather vai ativar seu BOT e te dar um **TOKEN**

A conversa com o BotFather vai ficar assim:



DISCLAIMER

Para fins didáticos e principalmente **restrições de tempo** vou abrir mão de alguns detalhes:

- Não farei o código Orientado à Objetos
- Vou abrir mão de algumas boas práticas por causa da diversidade de computadores, sistemas operacionais e níveis de conhecimento
- Na primeira versão deixarei informação sensível Hardcoded
 - respira, é temporário, eu prometo
- Vou abrir mão de outra centena de boas práticas (ops)

Estrutura do BOT (do arquivo fonte)

1. Adicionamos recursos externos (imports)
2. Dizemos o que o BOT deve fazer
 - a. Criamos uma função que será chamada a cada comando enviado ao BOT
 - b. Em termos técnicos estas funções são chamadas de Callbacks
3. Criamos uma instância do BOT passando para ele qual é o Token que o BotFather nos deu
4. Explicamos à instância do BOT qual comando chamará cada função
 - a. Explicaremos: Quando você receber o comando “abc” chame a função ExecutaAbc
5. Diremos ao BOT para começar a trabalhar
6. Pediremos que ele continue trabalhando até eu fechar ele

Vamos tentar !

1. Adicionamos recursos externos (imports)
 - `from telegram.ext import Updater, CommandHandler`
2. Dizemos o que o BOT deve fazer
 - `def start(bot, update):`
 `OQueDeveFazerNoComandoStart()`
3. Criamos uma instância do BOT passando para ele o Token
 - `bot = Updater('TOKEN QUE NÃO DEVE FICAR NO CÓDIGO')`
4. Explicamos ao BOT que o comando 'start' fica no `start`
 - `bot.dispatcher.add_handler(CommandHandler('start', start))`
5. Inicia o BOT e manda ele esperar
 - `bot.start_polling()`
 - `bot.idle()`

Exemplo de código completo

```
# Adicionamos recursos externos (imports)
from telegram.ext import Updater, CommandHandler
# Dizemos o que o BOT deve fazer
def start(bot, update):
    update.message.reply_text("Hello, World!")

# Criamos uma instância do BOT passando para ele o Token
bot = Updater('TOKEN QUE NÃO DEVE FICAR NO CÓDIGO')

# Explicamos ao BOT que o comando 'start' fica no start
bot.dispatcher.add_handler(CommandHandler('start', start))

# Inicia o BOT e manda ele esperar
bot.start_polling()
bot.idle()
```

Bora dar uma melhoria ?

1. Adicionar Log de atividades

- `import logging`
 `logging.basicConfig(format='%(asctime)s - %(message)s',`
 `level=logging.INFO)`
 `logger = logging.getLogger(__name__)`

2. Responder o Hello, World com o nome de quem chamou o BOT

- `update.message.from_user.first_name`

3. Responder com mensagem formatada em Markdown

- `from telegram import ParseMode, ChatAction`
 `bot.send_message(chat_id=update.message.chat.id,`
 `text='Olá, *negrito*',`
 `parse_mode=ParseMode.MARKDOWN)`

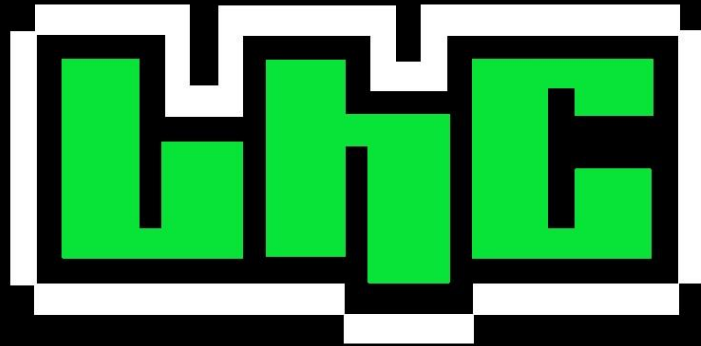
E o MAIS importante...

- PRECISAMOS tirar o Token do código ...
1. Passar o Token como parâmetro do BOT
 - a. `import sys`
....
`token = sys.argv[1]`
 - b. `py ./bot.py "abcdef:qwedfsdfasdsa_1234556789"`
 2. Pedir via Input ao usuário no início do BOT
 - a. `Token = input("Digite o Token")`
 3. Usar uma biblioteca de Configuração como o `pickle` ou o `configparser`

BOTs de Hackerspaces

- LHC
 - O botelho foi escrito na linguagem GO e está disponível em:
<https://github.com/lhc/telegram-bot>
- Garoa - São Paulo
 - O Chuvisco foi escrito em python e está disponível em:
<https://github.com/garoa/ChuviscoBot>
 - O chuvisco utiliza a mesma biblioteca que utilizamos !

Obrigado



Telegram: @RafaelEstevam
GitHub: RafaelEstevamReis/LHC_PyBOT

