Discord Bot

Construindo um bot para o Discord em Python.

Vantagens de se usar Python (nesse contexto)

- Ecosistema de bibliotecas robusto
- Conseguimo integrar modelos já existentes sem necessidade de portar
- Construção simples
- Impacto de performance negligenciável para essa aplicação

Ferramentas

- Python (🤡)
- Gerenciador de pacotes
  - Para essa demo, usaremos o micromamba
  - Quase qualquer outro funciona, mas vantagens e desvantagens se aplicam
- Bibliotecas
  - Para essa demo, usaremos o discord.py
- Editor de texto
- Conta de desenvolvedor no Discord
  - https://discord.com/developers/applications
- Opcionalmente, Docker

Modelo de comunicação

- Não é necessário expor portas
- A maior parte da comunicação parte do servidor (bot)
- Combinação de WebSockets e REST
- Discord API

Setup do projeto

Criação do ambiente inicial

```
# Cria um novo ambiente virtual
micromamba create -n discord-bot-lesson -c conda-forge python=3.12
# Ativa o ambiente na sessão atual
micromamba activate discord-bot-lesson
# Instalação das dependências iniciais
micromamba install -c conda-forge python-dotenv discord.py
```

Criação do projeto

Estrutura base:

```
.
└── demo-bot/
    ├── .env                    # Arquivo de configuração do ambiente (chave de API)
    ├── Dockerfile              # Para executar o bot em um container Docker
    ├── docker-compose.yml      # Para executar o bot em um container Docker
    ├── env.yml                 # Arquivo de configuração do ambiente (Conda/Micromamba)
    └── demo-bot/
        ├── __init__.py         # Delcara o diretório como um pacote
        ├── __main__.py         # Executa o bot
        └── main.py             # Entrypoint
```

O discord.py

- Biblioteca de Python para o Discord que abstrai a API do Discord simplificando o desenvolvimento de bots e outras aplicações.
- Assíncrona por meio de `asyncio`
- É possível estruturar o bot via classes ou anotações.

Chave de API

1. Tendo uma conta no Discord, basta acessar o portal de desenvolvedor

- https://discord.com/developers/applications

2. Criar um novo aplicativo
3. Acessar a aplicação criada e ir na aba **"Bot"**
4. Ir na seção "Token" e resetar o token (chave de API).

- Salvar o token no `.env`, nunca compartilhar essa chave, já que ela controla o bot.
- Usaremos a variável `DISCORD_TOKEN` para o token.

5. Modificar demais configurações de acordo com sua vontade e necessidade do bot.

Adicionando o bot ao servidor: dashboard/manual

1. Na mesma tela da aplicação, ir na aba "OAuth2" e clicar em "URL de autorização".
2. Selecione as permissões necessárias para o bot.
3. Copie o link e abra o navegador.

Adicionando o bot ao servidor: gerando a URL

É possível gerar o link programaticamente, tendo o seguinte formato:

```
`https://discord.com/api/oauth2/authorize?client_id=${client_id}&permissions=${permissions}&scope=${scope}`
```

Onde o `client_id` é o ID do cliente (não o token :p), o `permissions` é a lista de permissões (bit flags, possível montar na aba "Bot") e o `scope` é o escopo de acesso do bot
(montável na aba "OAuth2", geralmente `bot applications.commands`, que fica `bot%20applications.commands` na URL).

Configurando o bot

Por conta de alguns exemplos, recomenda-se habilitar o `Intents.message_content` e `Intents.members`
no dashboard do Discord.

Iniciando o desenvolivmento

Obrigada!

Qualquer dúvida, podem me chamar no Discord ou Telegram: `@JustLelis`