gh-cobra

A simple exercise using GitHub Cobra CLI to create an app that accesses the GitHub REST and GraphQl API's. It also gives help on basic Bash utilities.

You will need a Fine Grained Personal Access Token from Github for this to work. Login to Github, go to Developer Settings and request an access token. Copy it immediately and export it to an environment variable named GITHUB_TOKEN.

Note : I use GitHub Copilot in Visual Studio Code, which made it much easier and quicker to get working. Especially working with the GitHub GraphQL API

Commands

- $gh-cobra api [owner]
  - Print a list of public repositories on GitHub that belong to a specified owner.
  - Uses the GitHub REST API
  - see cmd/api.go
- $gh-cobra graphql [owner]
  - Print a list of public repositories on GitHub that belong to a specified owner.
  - Uses the GitHub Graphql API with a manual POST request
  - This command requires a GitHub bearer token in the environment variable GITHUB_TOKEN
  - see cmd/graphql.go
- $gh-cobra shurcool [owner]
  - Print a list of public repositories on GitHub that belong to a specified owner.
  - Uses the GitHub Graphql API using the shurcooL/graphql client package.
  - This command requires a GitHub bearer token in the environment variable GITHUB_TOKEN
  - see cmd/shurcool.go
- $gh-cobra gogithub [owner]
  - Print a list of public repositories on GitHub that belong to a specified owner.
  - Uses the GitHub Graphql API using the google/go-github client package.
  - This command requires a GitHub bearer token in the environment variable GITHUB_TOKEN
  - see cmd/shurcool.go
- $gh-cobra explain [ai prompt]
  - print help for basic Linux Bash utilties using Langchain and Chat-GPT
  - This command requires an OpenAI api key in the environment variable OPENAI_API_KEY
  - see cmd/explain.go and cmd/lc.go

Notes

- Built with Go version 1.8 or later
  - Run the app without building : 'go run . args..."
  - Build the app : 'go build ."
- The 'explain' command may give a warning like "Failed to calculate number of tokens...".
  - This warning is generated by the version of Langchain used in this app.
- You can add more to the list of supported names in the 'explain' command by updating the list in lc.go

Access Methods

REST API

From GitHub REST API Docs: "You can use GitHub's API to build scripts and applications that automate processes, integrate with GitHub, and extend GitHub. For example, you could use the API to triage issues, build an analytics dashboard, or manage releases."

GraphQL API

From GitHub GraphQL API Docs: "To create integrations, retrieve data, and automate your workflows, use the GitHub GraphQL API. The GitHub GraphQL API offers more precise and flexible queries than the GitHub REST API."

shurcooL vs gogithub

There is a tradeoff between using the go-github client vs the shurcooL client. Either will access the GraphQL API, but...

- The go-github client package is much much easier to use, as it provides direct functions to access the API and takes care of all the types for you.
- A drawback of the go-github package functions is that GitHub API types can contain a LOT of data elements. For example, when you use the client.Repositories.ListByUser, you get a ton of information and the associated overhead. This sort of defeats the purpose of GraphQL, which is designed to let you request only the data you want. You might as well use the REST API.
- the shurcooL client package is a bit more complicated to use.
- you have to figure out the types and formats of the queries and mutations. Which can take a bit of exploring using the GitHub schema and Explorer. In my case using GitHub Copilot made it 10 times easier to figue this stuff out.

Differences

REST vs GraphQL

"Under REST architecture, data is returned to the client from the server in the whole-of-resource structure specified by the server."

"A data format describes how you would like the server to return the data, including objects and fields that match the server-side schema"

On other words, GraphQL servers and client can mix data from multiple resources and specify just what is needed, where REST sends a single 'document'.

Which One to Use

Get the scoop from the source:

Comparing GitHub's REST API and GraphQL API

Langchain-OpenAI/Chat-gpt

While I was at it I did an example command that uses the Langchain module to get information about basic Unix/Linux command line tools. Langchain is a front end that facilitates programmable access to various large language models. Yes it doesn't really fit with the GitHub stuff but I wanted to try it out.