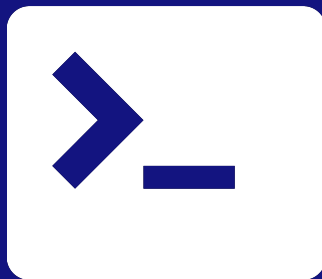
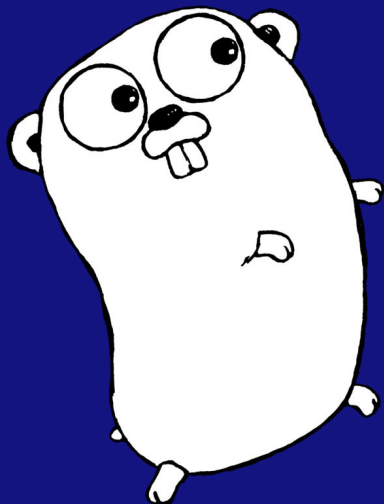


Vlastní CLI tooly v Go



Ondřej Šika

ondrej@sika.io

@ondrejsika

jOpenSpace, 8. 10. 2022

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika

Ondřej Šika

Jsem DevOps lektor, architekt a konzultant z Prahy.

Navrhnou a implementuji Vám na míru DevOps architekturu od verzování v Gitu po provoz v Cloudu.

Dělám populární školení, kde své znalosti předávám tak, abyste si mohli vše udělat sami a bez zbytečných přešlapů a slepých cest.

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Píšu knihu o DevOps

- **Kultura DevOps**
- **Moderní DevOps Stack**
- **Jak navrhnout moderní DevOps architekturu**
- **Kontejnery, Kubernetes, Terraform, ... deep dive**
- **Automatizace a GitOps**
- **Mnoho příkladů a ukázek**

kniha.sika.io

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Vlastní CLI Tool. Proč?



Zjednodušení opakujících se tasků

Člověk je líný a chce si ulehčit práci



Co to znamená v praxi

- Generování kódu při vývoji
- Práce s Kubernetes
- Generování dat pro testování
- Samotné testování
- Automatické commit messages
- Utility pro shell

...



Řešení?

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Shell Scripty 🎉

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Shell Scripty 🙄



Shell Scripty

- Jednou je napíšeme a příště jen použítme
- Shell funguje všude
- Jednoduše se píše
- Jeden Git repozitář, kde jsou všechny skripty uloženy



Ale ...

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Ale ...

- Je všude opravdu Bash, ZSH nebo Shell?
- Píší se opravdu tak jednoduše?
- Distribuce Git repozitáře do kontejneru?
- Distribuce bez internetu?



Shell funguje všude

Ne všude je shell. Není například na Windows. Pokud chceme, aby naše utility fungovaly i tam, musíme zvolit něco jiného.



Shell se jednoduše píše

Pokud píšete jednoduchý skript na pár řádků, pak ano.

Ale pokud chcete parametrizovat, složitější funkcionalitu, používat api a knihovny, autocomplete, ... tak pak už zas tak dobře ne.



Distribuce shell scriptů

Skripty v jednom Git repozitáři jsou fajn ale ...

- Distribuce do míst bez Gitu
- Používání v CI
- Používání v kontejneru



Řešení?

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Python



@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Python 🙄



Python jeden problém řeší, ostatní zhoršuje



Python řeší:

- Jednodušší a rychlejší vývoj než v SH / BASH
- Používání knihoven, práce s API, SDKs, ...



Python zhoršuje:

- Python není všude a když je, tak v různých verzích (2 vs 3)
- Instalace závislosti a knihoven není ideální (virtualenv ...)
- Distribuce pořád není jedna binárka



Řešení?

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika





@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



CLI tool v Go 🎉🎉🎉



Proč CLI tool v Go?

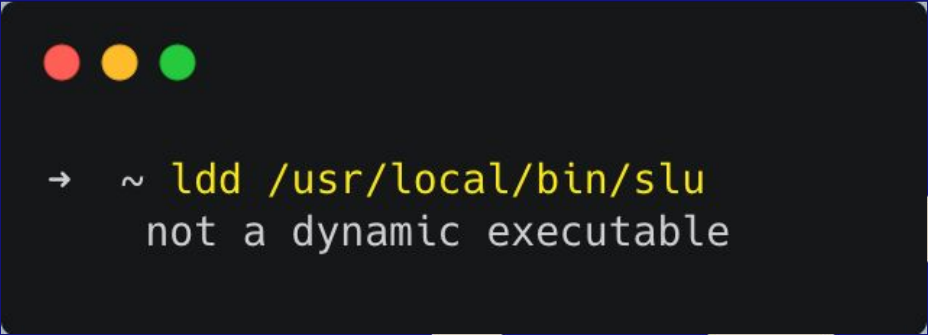


Staticky linkované binárky

- Go nepotřebuje pro svůj běh NIC (žádné závislosti)
- Velmi jednoduchý způsob jak vytvořit staticky linkovanou binárku

`CGO_ENABLED=0 go build`

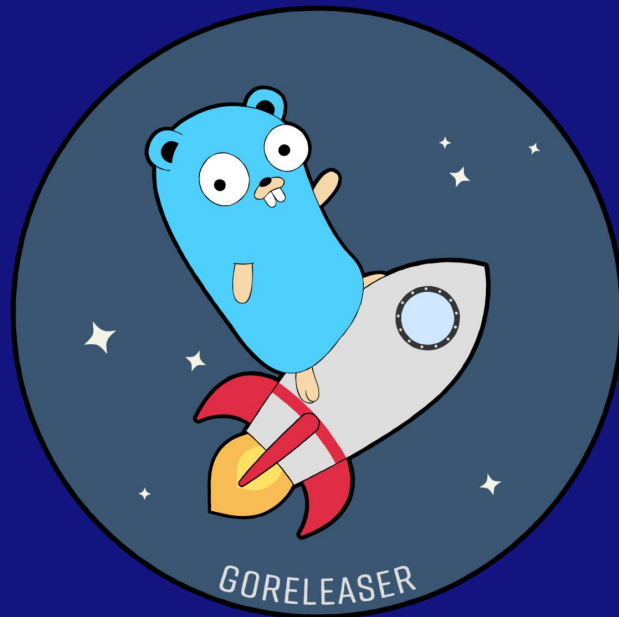
- Binárka pro každý OS a arch
 - Linux, Mac, Windows, ...
 - amd64, arm64, ...



```
→ ~ ldd /usr/local/bin/slu  
not a dynamic executable
```

Goreleaser

- Build binárek (pro všechny platformy)
- Build & push Docker image
- upload binárek na Github / Gitlab
- Vytvoří Brew a Scoop balíčky



Distribuce Go binárek

Díky Goreleaseru máme distribuci velmi jednoduchou

- `brew install my-cli-tool`
- `curl -fsSL https://.../my-cli-tool/master/install.sh | sh`
- `scoop install https://.../scoop-bucket/master/my-cli-tool.json`



Cobra & Viper

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Cobra

Cobra je skvělá knihovna pro psaní CLI nástrojů

- **Používá ji většina CLI nástrojů jako je Docker, Kubernetes, Helm, ArgoCD, ...**
- **Velmi jednoduše a příjemně se používá**
- **Automaticky generuje completion pro BASH, ZSH, Powershell, ...**
- **Generuje dokumentaci**



Viper

Viper je knihovna na práci s konfiguračními soubory a ENV variables

- **setting defaults**
- **reading from JSON, TOML, YAML, ...**
- **reading from environment variables**
- **reading from remote config systems (etcd or Consul)**



Completion

Podporuje BASH, ZSH, FISH a PowerShell

Source in current shell

```
. <(my-cli-tool completion bash)
```

Add to .bashrc

```
echo ". <(my-cli-tool completion bash)" >> ~/.bashrc
```



Generovaná Dokumentace

☰ 60 lines (54 sloc) | 2.79 KB <> 📄 Raw Blame 🖨️ 📋 ✎ 🗑️

slu

SikaLabs Utils, v0.35.0

Options

```
-h, --help  help for slu
--json      Format output to JSON
```

🔗 SEE ALSO

- [slu argocd](#) - ArgoCD Utils
- [slu cloudflare](#) - Cloudflare Utils
- [slu completion](#) - Generate the autocompletion script for the specified shell
- [slu debug-server](#) - HTTP Server for debug & development
- [slu digitalocean](#) - DigitalOcean Utils
- [slu docker](#) - Docker Utils
- [slu du](#) - Own implemetation of "du"

Moje CLI tooly v Go

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Moje CLI tooly v Go

- **slu** - všechny možné utility, co se můžou hodit (SikaLabs Utils)
- **training-cli** - Nástroj pro zjednodušení přípravy mých školení
- **tergum** - Backup nástroj



github.com/sikalabs/slu

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika





→ ~ **slu**

SikaLabs Utils, v0.40.0

Usage:

slu [command]

Available Commands:

argocd	ArgoCD Utils
cloudflare	Cloudflare Utils
completion	Generate the autocompletion script for the specified shell
debug-server	HTTP Server for debug & development
df	System's "df" filtered for /dev devices and human readable



→ ~ `slu tls parse -a sika.io:443`

Subject Name: CN=sni.cloudflaressl.com,O=Cloudflare\,
Inc.,L=San Francisco,ST=California,C=US

Subject Common Name: `sni.cloudflaressl.com`

Issuer Name: CN=Cloudflare Inc ECC CA-3,O=Cloudflare\,
Inc.,C=US

Issuer Common Name: `Cloudflare Inc ECC CA-3`

Created: `2022-05-11T00:00:00Z`

Expiry: `2023-05-10T23:59:59Z`



→ ~ `slu loggen`

`loggen 2022/06/11 07:42:01` Logging into STDOUT

`loggen 2022/06/11 07:42:02` WARN A warning that should be ignored is usually at this level and should be actionable.

`loggen 2022/06/11 07:42:04` INFO This is less important than debug log and is often used to provide context in the current task.

`loggen 2022/06/11 07:42:07` DEBUG This is a debug log that shows a log that can be ignored.

`loggen 2022/06/11 07:42:08` DEBUG This is a debug log that

`loggen 2022/06/11 07:42:12` ERROR An error is usually an exception that has been caught and not handled.



→ ~ `slu wireguard genkey`

priv: 0Mw0nRSzBxV4cCyyTK1mD9ELT2gPzFHPwTUTGcbyrGA=

publ: Raqa9Cfw4iq/4gjqu2NiHYd7eBdLtJBPbFnCd3Ks+nY=

pres: Us40VLrPrFo7x6xxl0J8iF5R8SaDb/xTjI0uITcXNyY=

github.com/ondrejsika/training-cli

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika





→ ~ `docker pull -q sikalabs/dev`

`docker.io/sikalabs/dev:latest`

→ ~ `docker run -ti sikalabs/dev`

`root@cc58e8df57ba:/# training-cli kubernetes connect`

You are connected to my demo cluster

`root@cc58e8df57ba:/# kubectl get nodes`

NAME	STATUS	ROLES	AGE	VERSION
sikademo-clldb0	Ready	<none>	2d	v1.22.8
sikademo-cldbd	Ready	<none>	2d	v1.22.8
sikademo-cldbv	Ready	<none>	2d	v1.22.8

Shrnutí

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Shrnutí

- CLI tooly v Go jsou super, vyzkoušeno
- Skvěle se používají a distribuují
- Jednoduše a rychle se vyvíjí



Díky za pozornost

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Otázky?

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika



Email

ondrej@sika.io

Twitter

@ondrejsika

LinkedIn

/in/ondrejsika

@ondrejsika ondrej@sika.io sika.io /in/ondrejsika

