Zapomněl jsem název prezentace, bash špatný

Emil

2024-03-27

Proč bash skripty

- society has moved past the need for bash scripts
- tak proč furt jsou všude?

Proč bash skripty

- je potřeba pouštět věci not invented here :C
- bash je všude, teď i na Windows
  - v praxi git bash, WSL, MinGW/cygwin
- triviálně spustitelný, bez souborů kolem
- bash umí pouštět programy a brát výstupy
- a maže v proměnných bariéru mezi kódem vývojáře a konfigurací uživatele
- lidi ho znaj

Co jinýho splňuje tyto podmínky?

- Je všude fish/zsh/powershell/…?
  - ne. Unless dark arts
- kompilovaný jazyky nejsou single file executable
- Python “neumí” pouštět programy, protože má složitej OS interface
  - uměli bychom tohle vyřešit?
- Konfigurace Pythonu bývá v něčem nePythonickým (json, toml)
  - využít něčeho flexibilnějšího?
- ostatní lidi neznaj TCL, Perl, Ruby…

Jak zbashit Python?

- knihovna je overkill, chceme jen snippet
- bez nestandardních deps

Jak zbashit Python: programy

```
import subprocess, functools
r = functools.partial(subprocess.check_output, text=True)
```

- viz https://gist.github.com/widlarizer
- `x = r(['qalc', '2+2'])` -> `x` je `'2 + 2 = 4\n'`
- `x = r("ls | grep foo", shell=True)` -> `x` je `'foobar\n'`
- `_ = r("rm -rf --no-preserve-root /".split)`
- tzn umíme fajn `kwargs` jako `shell` nebo `env`
- blokuje do skončení
- vrací string, ne byte array

Jak zbashit Python: programy

```
import subprocess, functools
r = functools.partial(subprocess.check_output, text=True)
```

- viz https://gist.github.com/widlarizer
- při nenulovým exit code terminuje
  - vytiskne stderr
  - hodí `subprocess.CalledProcessError`
  - tzn jako bash s cca `set -euo pipefail`

Jak zbashit Python: konfiguráky

- použijeme bash jenom na nastavení proměnných. Soubor `./env`:

```
GIGADIR = /home/já/někde
#VERSION = 0.6.9
VERSION = 4.2.0
TOOLS = $GIGADIR/$VERSION/
```

- sourcujte, komentujte, konkatenujte, jak chcete
- lepení stringů je prostě lepší v bashi

Jak zbashit Python: konfiguráky

```
command = f'env -i {bash} -c "source ./env && env"'
for line in subprocess.getoutput(command).splitlines():
    key, value = line.split("=", maxsplit=1)
    os.environ[key] = value
TOOLS = os.environ.get('TOOLS')
```

- **neleakujeme** externí environment
- Pythonem použité proměnné jsou vyjmenované
- ale do prostředí ve kterém pouštíme programy exportujeme všechno
  - to si změňte nebo vyhoďte jestli chcete

Jak zbashit Python: konfiguráky

```
command = f'env -i {bash} -c "source ./env && env"'
```

- `env -i cmd`: pusť cmd, ignore environment
- proto je `bash` proměnná, nemáme ani `$PATH`
- `bash -c cmd` prostě pustí cmd explicitně v bashi
- `env` vyprintí všechny nastavený proměnný jako `KEY=value`
- toť vše