go-archetype

...

https://go-archetype.dev/

@rantav

#truestory

whoami?


AppsFlyer

The Challenge

Static project template

First attempt

Solution: Use a tool

Second attempt

Solution: **use a tool**. What tools are out there?

- Maven archetype
  - Still static (plus requires installing mvn)
- Rails generators
  - Too specific for RoR
- Yeoman
  - Frontend focused
- CRA - Create React App
  - Too specific for React

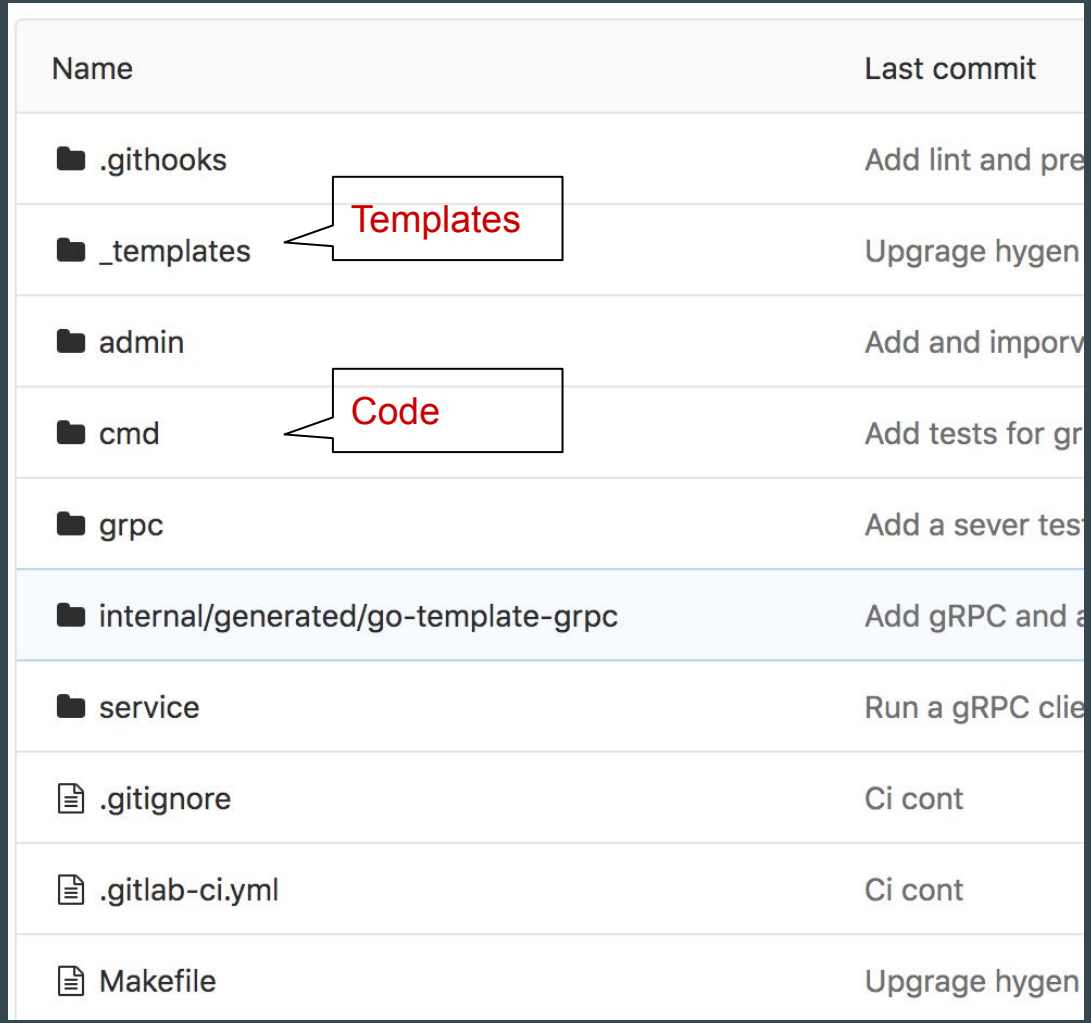
Solution: **use a tool**. OK, let's look for generic tools

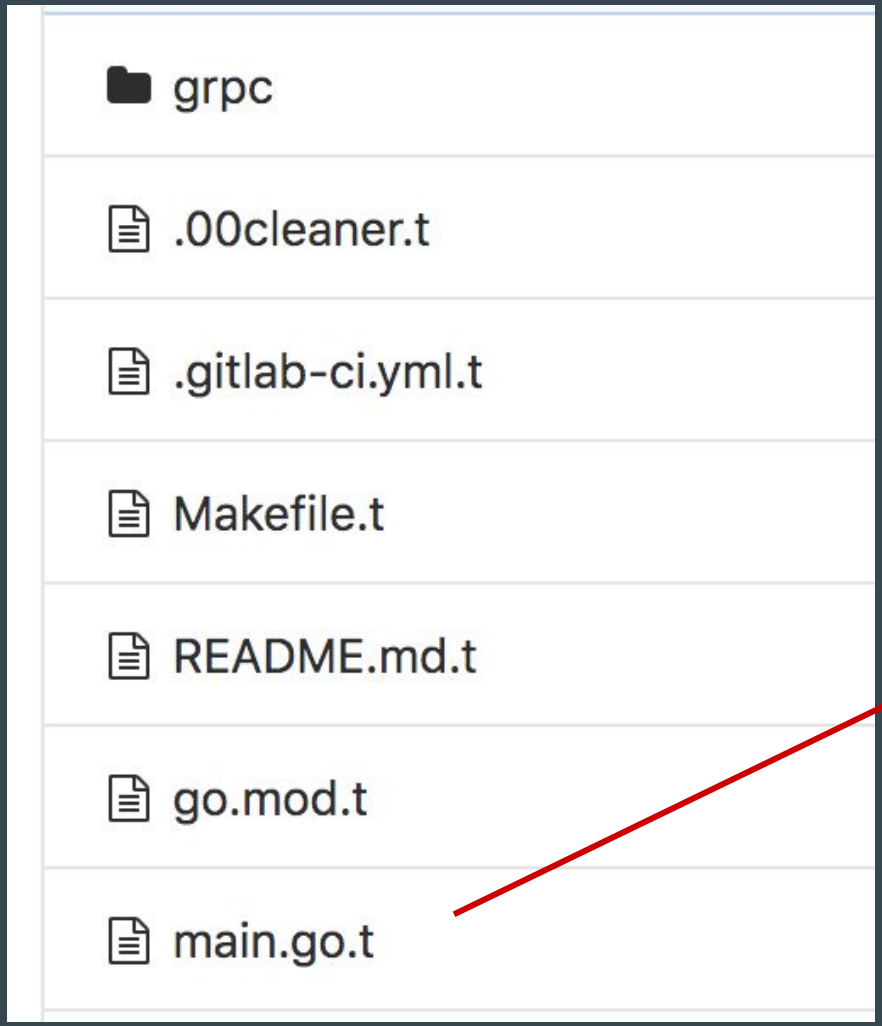
- Python Cookiecutter https://github.com/cookiecutter/cookiecutter
  - Very nice.
  - Very flexible
  - Large community
  - Jinja2 for templating
  - Even path names are templates.
  - Plus you can write python scripts for hooks etc
- Example Go template: https://github.com/lacion/cookiecutter-golang

Solution: **use a tool**. Yet another generic tool

- Hygen http://www.hygen.io/
  - Written by my friend Dotan
  - Also very capable
  - Templating via jsx (like React)

The Problem


Name
Last commit
.githooks
Add lint and pre
_templates
Templates
Upgrage hygen
admin
Add and imporv
cmd
Code
Add tests for gr
grpc
Add a sever tes
internal/generated/go-template-grpc
Add gRPC and a
service
Run a gRPC clie
.gitignore
Ci cont
.gitlab-ci.yml
Ci cont
Makefile
Upgrage hygen



main.go.t 113 Bytes

```
---
to: main.go
---
package main

import (
        "<%= repo_path %>/<%= name %>/cmd"
)

func main() {
        cmd.Execute()
}
```


WHAT DO WE WANT?
WRITE IN OUR NATIVE LANGUAGE
WHAT MORE DO WE WANT?
DRY
imgflip.com

go-archetype

The Solution

```
➜  af-go-template git:(master) .tools/bin/go-archetype-0.1.11/go-archetype trans
form --transformations transformations.yml --source=. --destination .tmp/go/my-g
o-project
```

```
func createCmd(log logger.Logger) *cobra.Command {
    var (
        cmd = &cobra.Command{
            Use:   "test-client",
            Short: "Test the gRPC or HTTP client, connect
            Long: `First run the server using the "serve"
Then run this test-client command to connect to this serv
        }
        // BEGIN __INCLUDE_GRPC__
        grpcServerAddress = cmd.Flags().String("grpc-serv
            "Network address of the gRPC server. Could us
        // END __INCLUDE_GRPC__
        // BEGIN __INCLUDE_HTTP__
        httpServerAddress = cmd.Flags().String("http-serv
            "Network address of the HTTP server.")
        // END __INCLUDE_HTTP__
    )
```

Markers

Markers

transformations.yml

```
- name: include grpc - parts of files
  type: include
  region_marker: __INCLUDE_GRPC__
  condition: .include_grpc
  files: ["Makefile", "**/*.go", "deployments/*"]
- name: include tracing - parts of files
  type: include
  region_marker: __INCLUDE_TRACING__
  condition: .include_tracing
  files: ["**/*.go", "deployments/*"]
- name: include http - parts of files
  type: include
  region_marker:   INCLUDE_HTTP
```

User inputs

```
inputs:
  - id: name
    text: What is the project name? (e.g. my-awesome-go-proje
    type: text
  - id: repo_path
    text: What is the project repo path, without the project
    type: text
  - id: description
    text: What is a long project description? (long texts are
    type: text
  - id: include_grpc
    text: Should gRPC functionality be included?
    type: yesno
  - id: include_tracing
    text: Should Jaeger tracing functionality be included?
    type: yesno
```

User inputs

```
inputs:
  - id: name
    text: What is the project name? (e.g. my-awesome-go-proje
    type: text
  - id: repo_path
    text: What is the project repo path, without the project
    type: text
  - id: description
    text: What is a long project description? (long texts ar
```

```
→  af-go-template git:(master)  .tools/bin/go-archetype-0.1.11/go-archetype transform --transformations transformations.yml
-destination .tmp/go/my-go-project
? What is the project name? (e.g. my-awesome-go-project) hello-go
? What is the project repo path, without the project name? (e.g. gitlab.appsflyer.com/rta) gitlab.appsflyer.com/rantav
? What is a long project description? (long texts are OK, but no newlines) This is just a test project
? Should gRPC functionality be included? (y/N)
```

```
    text: Should Jaeger tracing functionality be included?
```

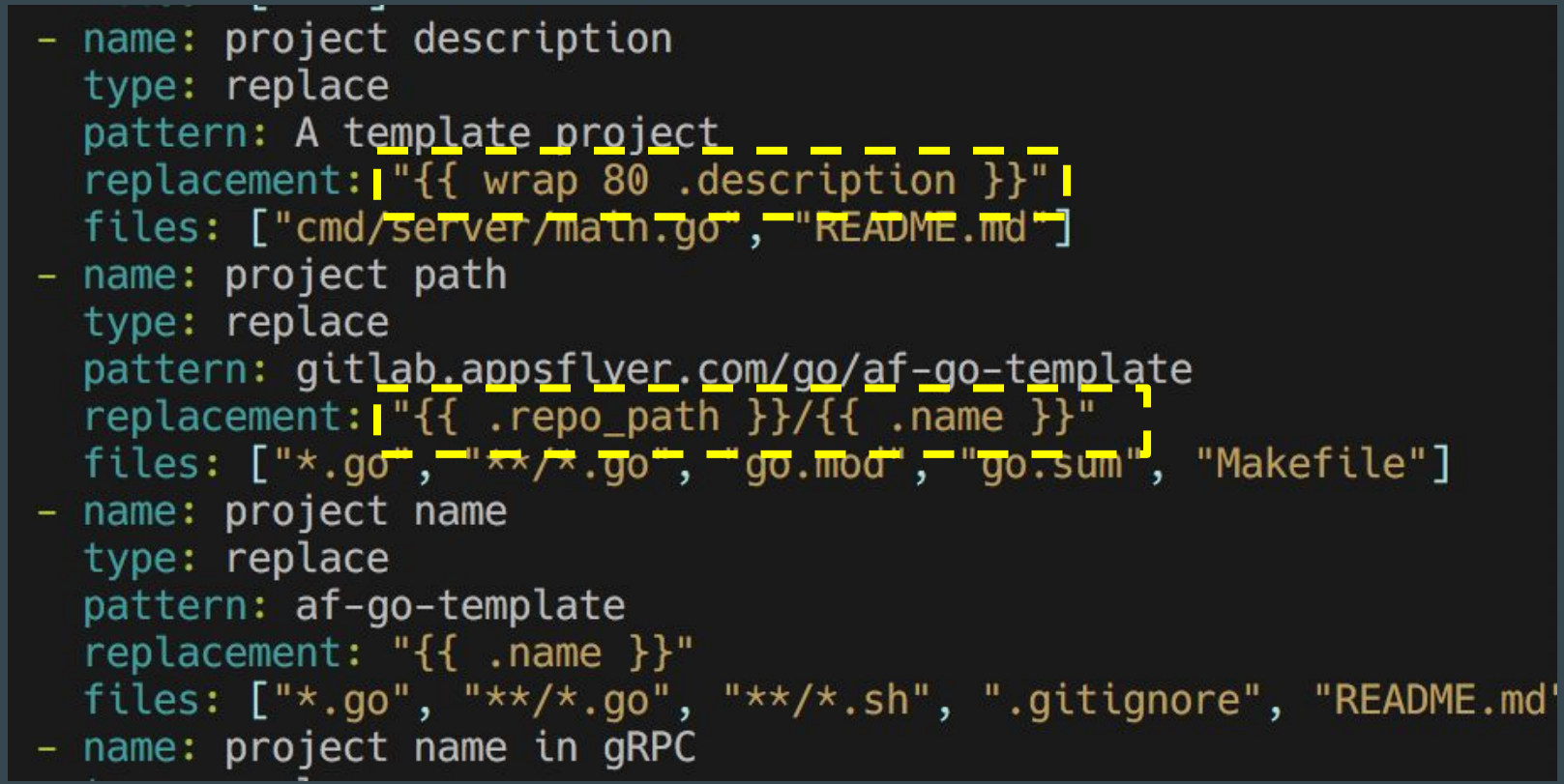
Search and replace. But with a twist

```
- name: project description
  type: replace
  pattern: A template project
  replacement: "{{ wrap 80 .description }}"
  files: ["cmd/server/main.go", "README.md"]
- name: project path
  type: replace
  pattern: gitlab.appsflyer.com/go/af-go-template
  replacement: "{{ .repo_path }}/{{ .name }}"
  files: ["*.go", "**/*.go", "go.mod", "go.sum", "Makefile"]
- name: project name
  type: replace
  pattern: af-go-template
  replacement: "{{ .name }}"
  files: ["*.go", "**/*.go", "**/*.sh", ".gitignore", "README.md"
- name: project name in gRPC
```

Pattern to search

Replace with this

Search and replace. But with a twist



Usage, Tips and Tricks

Tips and Tricks - Go templates

```
- name: project path
  type: replace
  pattern: gitlab.appsflyer.com/go/af-go-template
  replacement: "{{ .repo_path }}/{{ .name }}"
```

```
- name: project description
  type: replace
  pattern: A template project
  replacement: "{{ wrap 80 .description }}"
```

Tips and Tricks - Go templates

```
- name: project name in gRPC proto
  type: replace
  pattern: af_go_template
  replacement: "{{ .name | snakecase }}"
```

```
- name: build with protoc or not
  type: replace
  pattern: "build: protoc"
  replacement: "{{ if .include_grpc }}build: protoc{{ else }}build:{{end}}"
```

Tips and Tricks - the sprig library functions

- All Sprig library functions are available.
- Examples:
  - `trim`
  - `upper`
  - `lower`
  - `title`
  - `wrap`
  - `plural`
  - `snakecase`
  - `camelcase`
  - `kebabcase`

Tips and Tricks - Global ignore list (like .gitignore)

```
1 transformations.yml
ignore:
  - .git/
  - transformations.yml
  - CHANGELOG.md
  - .tmp/
  - .gopath/
  - pkg/grpc/lib
  - .testCoverage.txt
  - scripts/init.sh
```

Tips and Tricks - Global `before` and `after` scripts

```
before:
  operations:
    - sh:
      - echo "Starting to generate proje
```

```
after:
  operations:
    - sh:
      - cd {{.destination}} && goimports -w .
      - cd {{.destination}} && gofmt -s -w .
      - cd {{.destination}} && make build
      - cd {{.destination}} && make tidy
```

Tips and Tricks - Injected variables

- All user inputs are injected to the `before` and `after` operations as well as all other templates
  - E.g. `{{ .include_grpc }}` // <- this is user input (bool)
- As well as some meta variables
  - `{{ .source }}`
  - `{{ .destination }}`

Tips and Tricks - Include/exclude entire dirs or files

```
transformations:
  - name: include grpc - whole files
    type: include
    region_marker:
    condition: .include_grpc
    files: ["pkg/grpc/**", "pkg/admin/grpc*"]
```

When region marker is empty

Tips and Tricks - Include/exclude sections inside files

```
- name: include tracing - parts of files
  type: include
  region_marker: __INCLUDE_TRACING__
  condition: .include_tracing
  files: ["**/*.go", "deployments/*"]
```

```
// BEGIN __INCLUDE_TRACING__
c.TracingProbability,
c.JaegerHost,
// END __INCLUDE_TRACING__
```

Example: Not just .go files. A Makefile

```
# BEGIN __INCLUDE_GRPC__
## protoc: run the protoc compiler
protoc: $(PROTOC_BIN) protoc-gen-go
	mkdir -p pkg/grpc/lib
	$(PROTOC_BIN) --proto_path=pkg/grpc/idl/ --go_ou
# END __INCLUDE_GRPC__
```

Summary

- go-archetype lets you easily create blueprints
  - Single code base in your native language (Go, Python, JS, Clojure)
  - Using comments as markers
  - And a single transformations.yml file
- User inputs
  - String (e.g. project_name)
  - Boolean (e.g. inclue_grpc)
- Transformations
  - Include/exclude entire file
  - Include/exclude parts of files (using inline code comments)
- Search and Replace
  - With the power of go text templates (`{{.name | snakecase}}`)

https://go-archetype.dev/

Visit and star my project 🙏