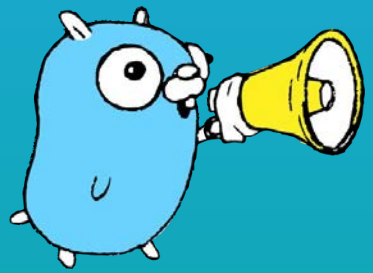
goyek

using Go for automation

Robert Pająk

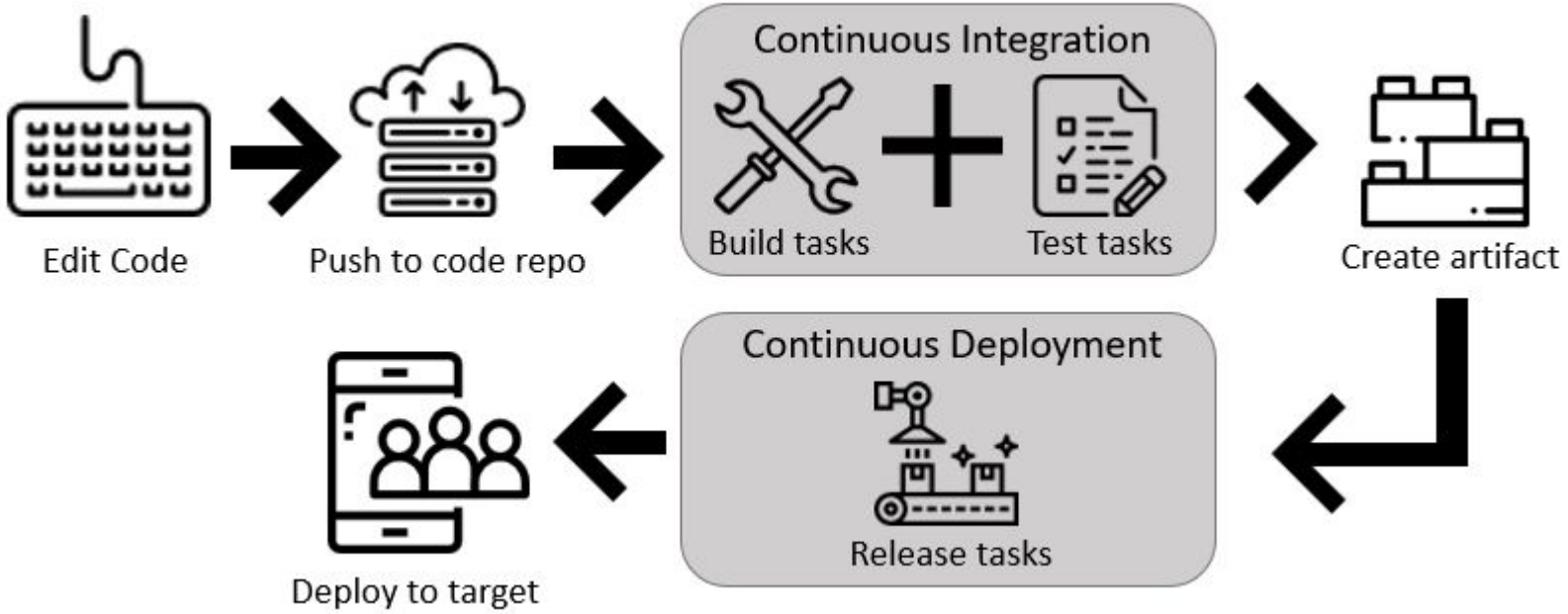
pellared @ GitHub

Splunk



Agenda

AUTOMATION


Edit Code
Push to code repo
Continuous Integration
Build tasks
Test tasks
Create artifact
Continuous Deployment
Release tasks
Deploy to target

AUTOMATION EXAMPLE (SIMPLE BUILD WORKFLOW)

all

The build workflow.
It depends on **fmt** and **test** targets/tasks:

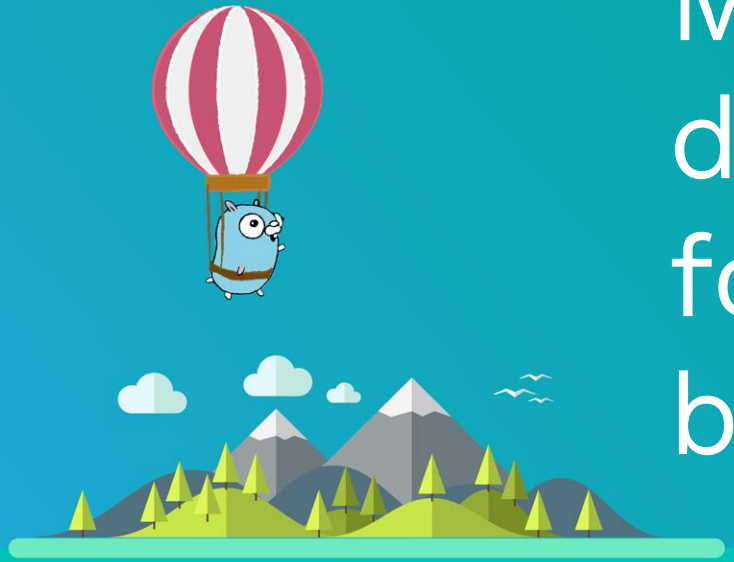
fmt

Simply runs go fmt ./....
This target is imported as it can be reused by multiple projects.

test

Runs tests and generates code coverage even if any test fails.

Repository: https://github.com/pellared/goyek-go-for-automation



“Makefile is de-facto standard for automating builds in Go”

PROBLEMS WITH MAKE

- Writing complex logic is hard
- Debugging experience
- Compatibility across different operating systems

Go for automation

WHAT ABOUT MAGE?

Mage

"Mage is a make/rake-like build tool using Go.

You write plain-old Go functions, and Mage automatically uses them as Makefile-like runnable targets."

PROBLEMS WITH MAGE

Mage is too magical 🧙

- Target discovery
- Build tags
- Import comments

Gotchas:

- Debugging
- Concurrency and logging
- Non-verbose mode
- Large API surface of the ‘sh’ package

goyek

API INSPIRATION

- testing
- spf13/cobra
- flag
- http

CHECK OUT MORE

Do you know that you can:

1. *add middlewares (task runner interceptors)?*
2. *create a standardized, reusable, customizable build pipeline?*
3. *customize printing?*
4. *integrate with spf13/viper?*

Visit https://github.com/goyek/goyek and see other examples to learn more.

RUN THE DEFAULT TASK - FAIL

```
$ go run .
===== TASK  spell
     spell.go:14: Work dir: build
     spell.go:14: Exec: go install github.com/client9/misspell/cmd/misspell
     spell.go:21: Exec: misspell -error -locale=US -w CODE_OF_CONDUCT.md README.md
README.md:40:14: corrected "folowing" to "following"
     spell.go:21: exit status 2
----- FAIL: spell (0.50s)
task failed: spell      2.204s
exit status 1
```

CODE - “SPELL” TASK (build/spell.go)

```go
package main

import (
    "strings"

    "github.com/goyek/goyek/v2"
    "github.com/goyek/x/cmd"
)

var spell = goyek.Define(goyek.Task{
    Name:  "spell",
    Usage: "misspell",
    Action: func(a *goyek.A) {
        if !cmd.Exec(a, "go install github.com/client9/misspell/cmd/misspell",
            cmd.Dir("build")) {
            return
        }
        mdFiles := find(a, ".md")
        if len(mdFiles) == 0 {
            a.Skip("no .md files")
        }
        cmd.Exec(a, "misspell -error -locale=US -w "+strings.Join(mdFiles, " "))
    },
})
```

CODE - “FIND” HELPER (build/find.go)

```go
package main

import (
    "io/fs"
    "path/filepath"

    "github.com/goyek/goyek/v2"
)

// find returns all files with the given extension in this repository.
func find(a *goyek.A, ext string) []string {
    a.Helper()
    var files []string
    err := filepath.WalkDir(".", func(path string, d fs.DirEntry, err error) error {
        if err != nil {
            return err
        }
        if filepath.Ext(d.Name()) == ext {
            files = append(files, filepath.ToSlash(path))
        }
        return nil
    })
    if err != nil {
        a.Fatal(err)
    }
    return files
}
```

remaks

MAKE

- Powerful and more concise
- Language agnostic
- You can always run Go programs instead of writing complex Bash scripts

MAGE

- Requires only Go
- Has community, 50+ contributors, 3k+ stars
- A lot of repositories are using it
- Has gotchas that are annoying me

GOYEK

- Requires only Go
- Library instead of a framework/tool
  with API inspired by popular Go packages
- Simple, yet extensible
- Works well with your IDE
- Behavior similar to “go test”
- Unpopular :)

thanks

feedback is more than welcome!

goyek: https://github.com/goyek/goyek
demo: https://github.com/pellared/goyek-go-for-automation
email: pellared@hotmail.com
linkedin: https://www.linkedin.com/in/rpajak