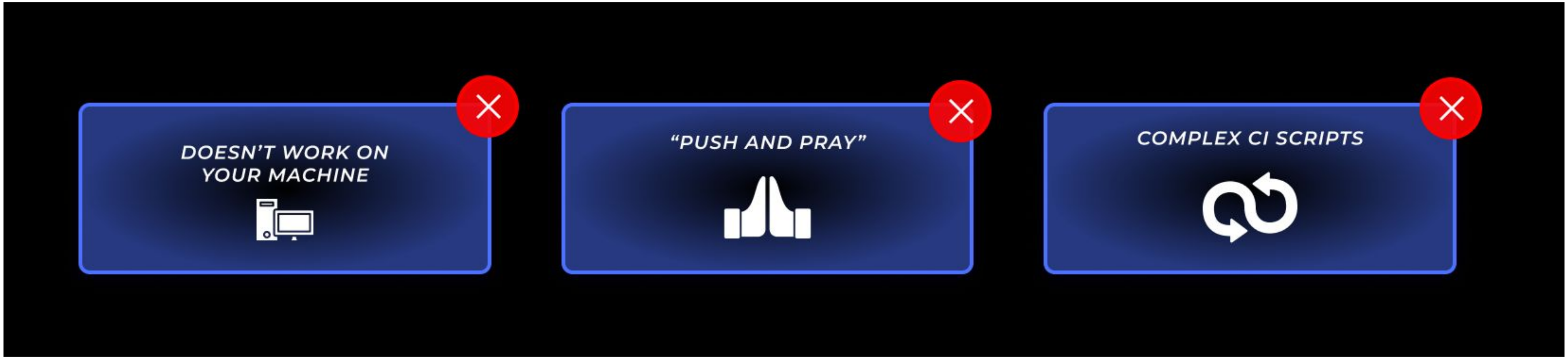
PUZZLE ITC
changing IT for the better

Dagger

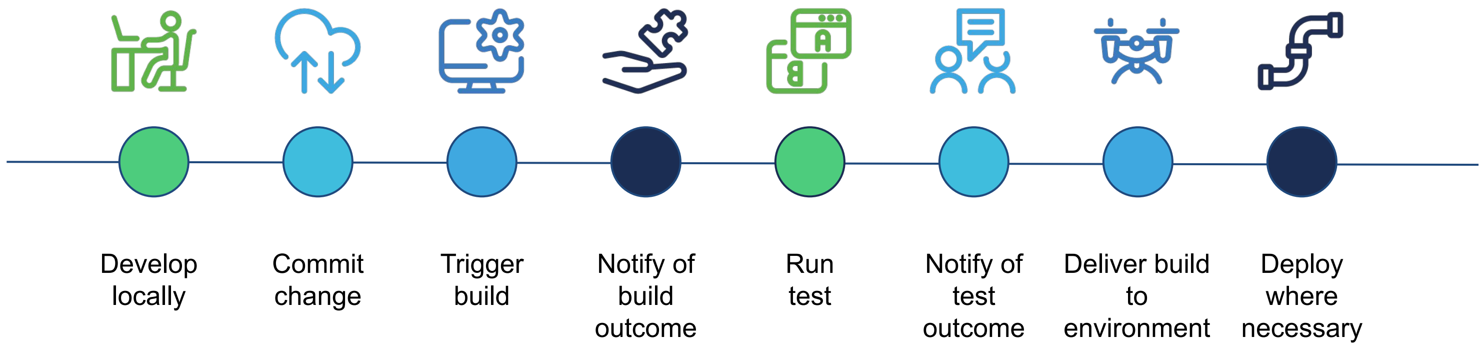
From messy CI scripts to clean code

Current CI/CD problems



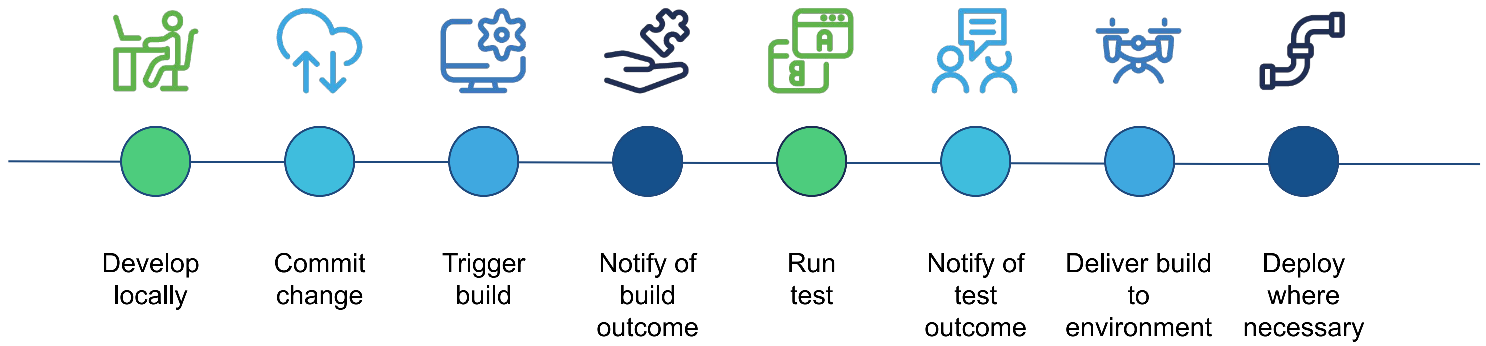
Current CI/CD problems

the ideal setup



Current CI/CD problems

BUT



- More complex system
- Drift between local and cloud environment
- Missing documentation

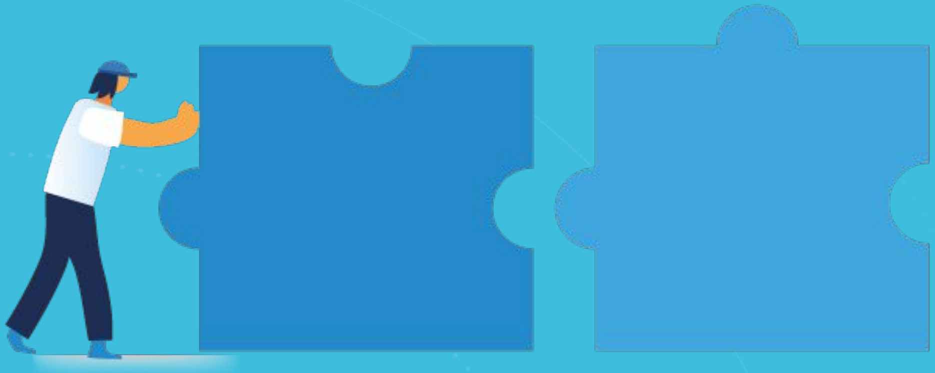
- A lot of yaml/groovy scripts to debug
- endless log files

Current CI/CD problems: What we want

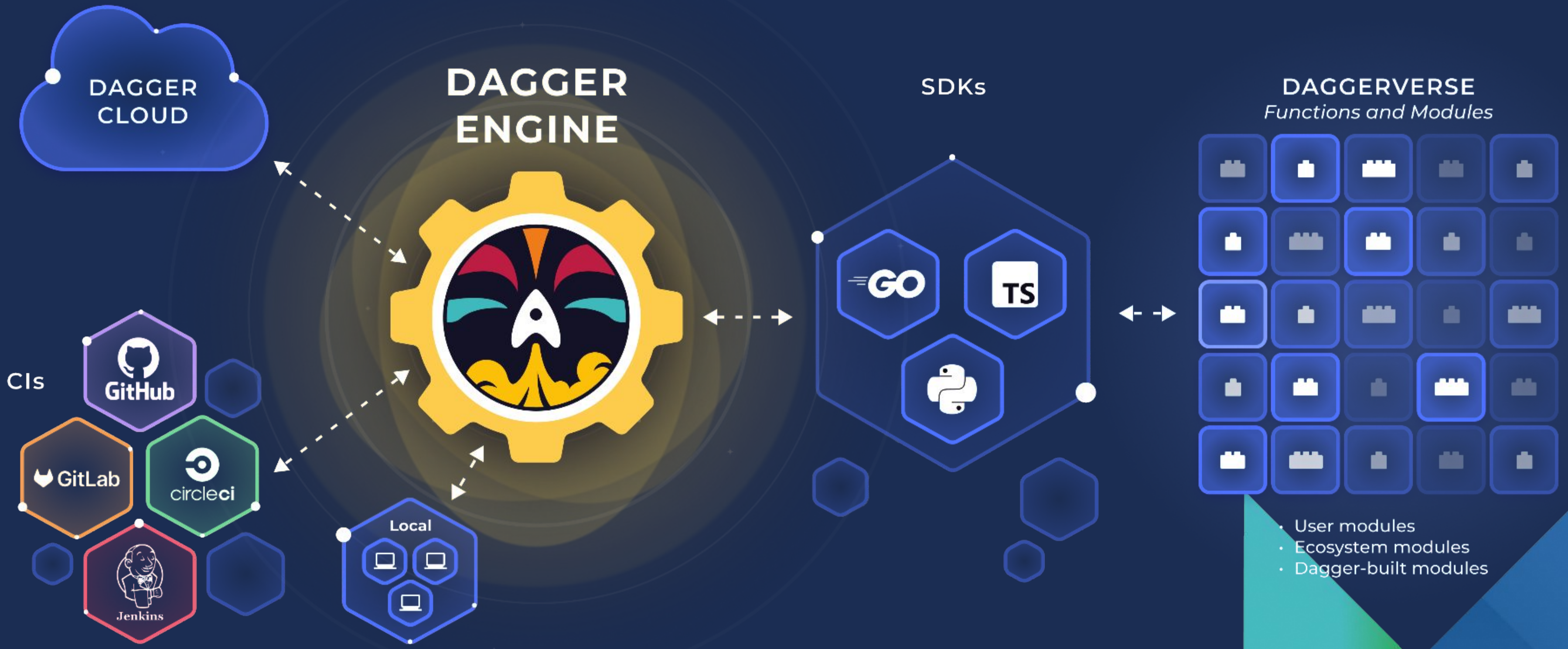
WORKS ON YOUR MACHINE
NO MORE "PUSH AND PRAY"
PROGRAM IN THE LANGUAGE OF YOUR CHOICE
TS
GO

Let’s update our pipelines

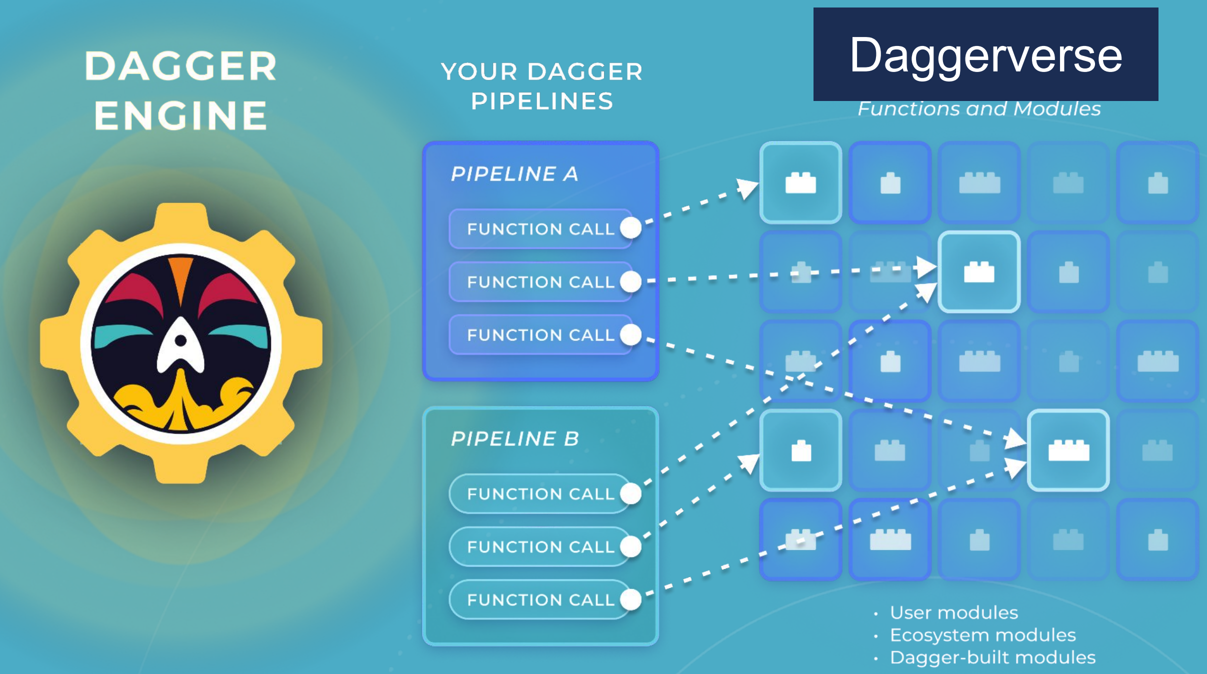
- expressive programming langs
  - code > YAML
  - Copilots/IDEs/ChatGPT
- Containers
  - Isolation, caching
- Existing CI servers
- powerful dev machines



The Dagger Platform



- User modules
- Ecosystem modules
- Dagger-built modules

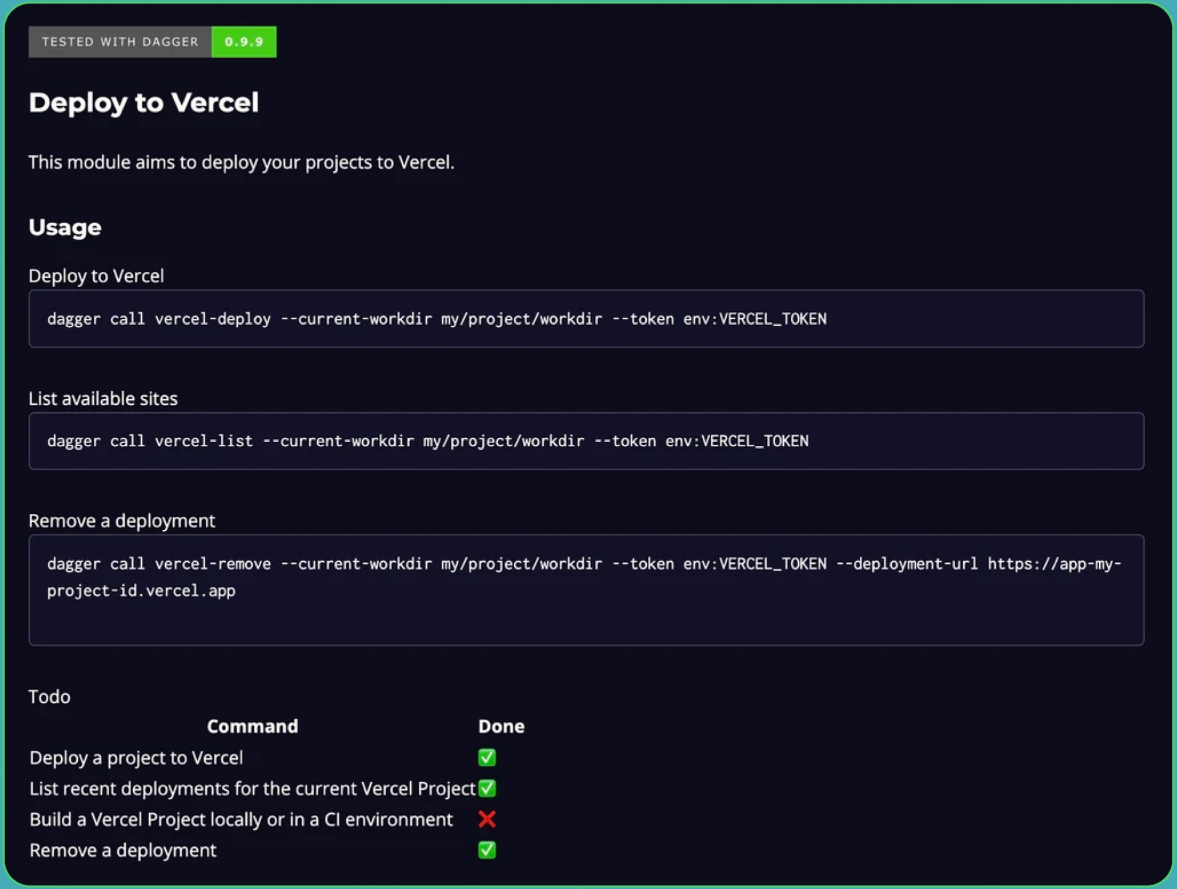


No more
YAML soup

Replace complex CI scripts
with a programmable platform

Standardized Dagger Functions

Pipelines just chain Dagger Functions - built by your team or by the community



TESTED WITH DAGGER 0.9.9

Deploy to Vercel

This module aims to deploy your projects to Vercel.

Usage

Deploy to Vercel

```
dagger call vercel-deploy --current-workdir my/project/workdir --token env:VERCEL_TOKEN
```

List available sites

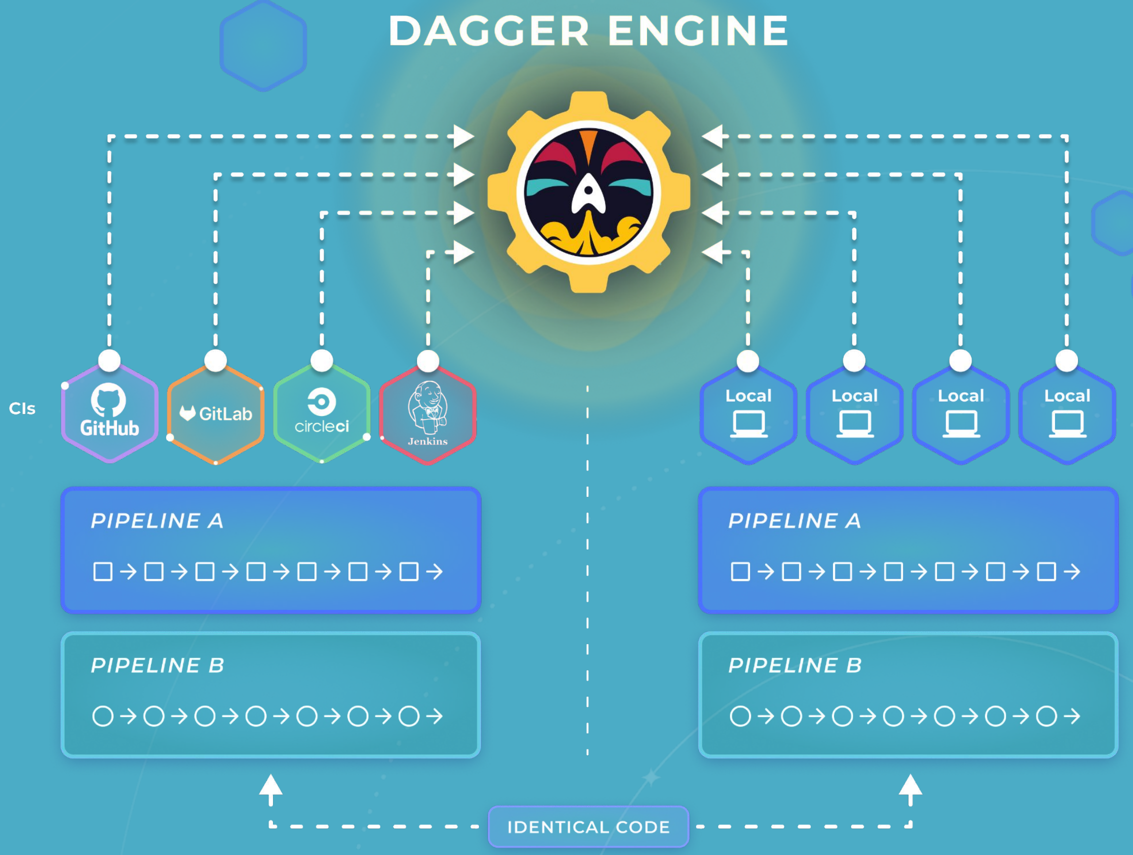
```
dagger call vercel-list --current-workdir my/project/workdir --token env:VERCEL_TOKEN
```

Remove a deployment

```
dagger call vercel-remove --current-workdir my/project/workdir --token env:VERCEL_TOKEN --deployment-url https://app-my-project-id.vercel.app
```

Todo

Command	Done
Deploy a project to Vercel	✅
List recent deployments for the current Vercel Project	✅
Build a Vercel Project locally or in a CI environment	❌
Remove a deployment	✅



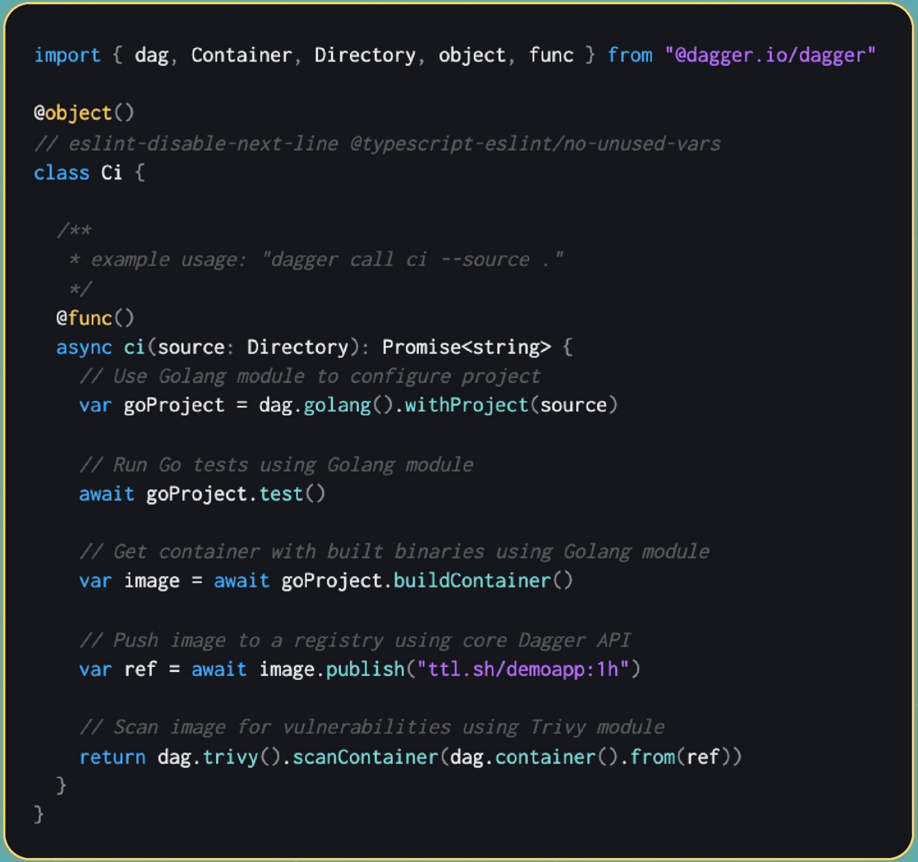
Eliminate
Push And Pray

If it works on your laptop
it'll work in CI

Cached
For Speed

Avoid unnecessary rebuilds and test reruns when nothing has changed

```go
func (g *Golang) Base(version string) *Golang {
    mod := dag.CacheVolume("gomodcache")
    build := dag.CacheVolume("gobuildcache")
    image := fmt.Sprintf("golang:%s", version)
    c := dag.Container().
        From(image).
        WithMountedCache("/go/pkg/mod", mod).
        WithMountedCache("/root/.cache/go-build", build)
    g.Ctr = c
    return g
}
```



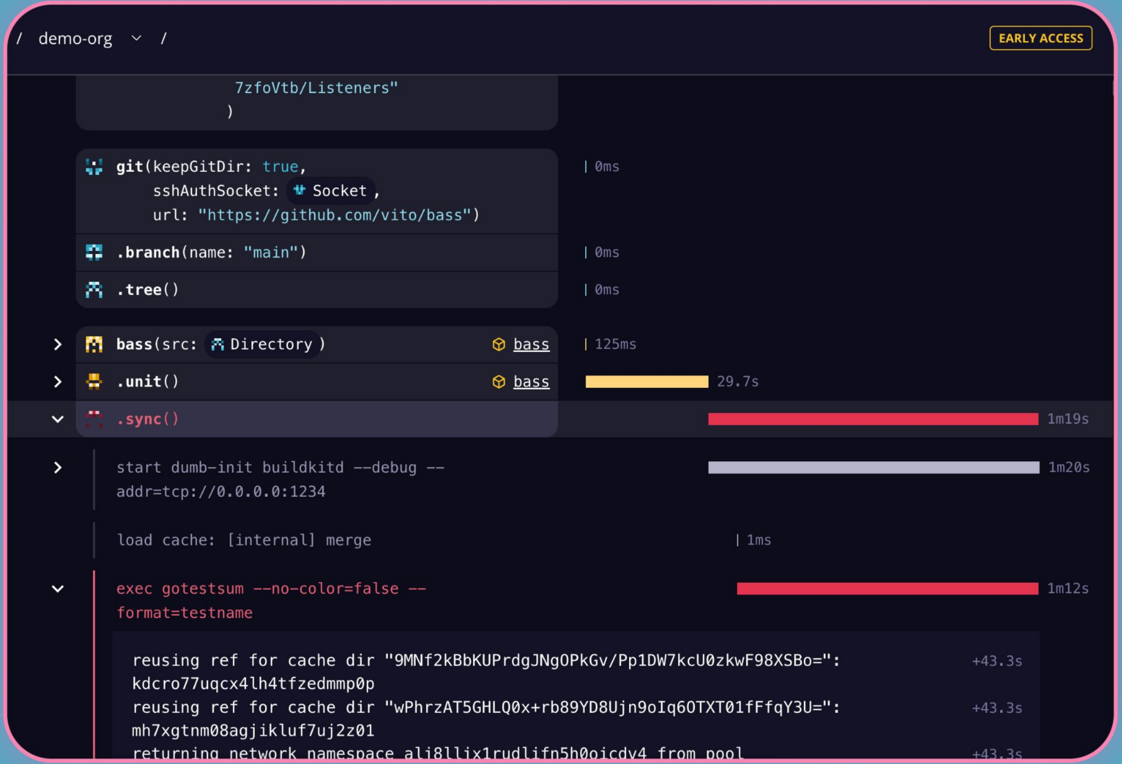
```
import { dag, Container, Directory, object, func } from "@dagger.io/dagger"

@object()
// eslint-disable-next-line @typescript-eslint/no-unused-vars
class Ci {

  /**
   * example usage: "dagger call ci --source ."
   */
  @func()
  async ci(source: Directory): Promise<string> {
    // Use Golang module to configure project
    var goProject = dag.golang().withProject(source)

    // Run Go tests using Golang module
    await goProject.test()

    // Get container with built binaries using Golang module
    var image = await goProject.buildContainer()

    // Push image to a registry using core Dagger API
    var ref = await image.publish("ttl.sh/demoapp:1h")

    // Scan image for vulnerabilities using Trivy module
    return dag.trivy().scanContainer(dag.container().from(ref))
  }
}
```

Multi-Language

Pipelines in the same language as your app. Each Dagger Function is just an API call away.

Visualize Your Pipelines

My test failed.
Is it a broken Pipeline?
Dagger gives you visibility into every aspect
of your pipelines



```
            7zfoVtb/Listeners"
          )

git(keepGitDir: true,                                  | 0ms
    sshAuthSocket: Socket ,
    url: "https://github.com/vito/bass")
.branch(name: "main")                                  | 0ms
.tree()                                                | 0ms

> bass(src: Directory )                     bass       | 125ms
> .unit()                                   bass       29.7s
v .sync()                                                         1m19s

> start dumb-init buildkitd --debug --                            1m20s
  addr=tcp://0.0.0.0:1234

  load cache: [internal] merge                            | 1ms

v exec gotestsum --no-color=false --                              1m12s
  format=testname

    reusing ref for cache dir "9MNf2kBbKUPrdgJNgOPkGv/Pp1DW7kcU0zkwF98XSBo=":   +43.3s
    kdcro77uqcx4lh4tfzedmmp0p
    reusing ref for cache dir "wPhrzAT5GHLQ0x+rb89YD8Ujn9oIq6OTXT01fFfqY3U=":   +43.3s
    mh7xgtnm08agjikluf7uj2z01
    returning network namespace ali8llix1rudlifn5h0oicdv4 from pool            +43.3s
```

Lab Time

-> your turn

https://dagger-techlab.puzzle.ch/