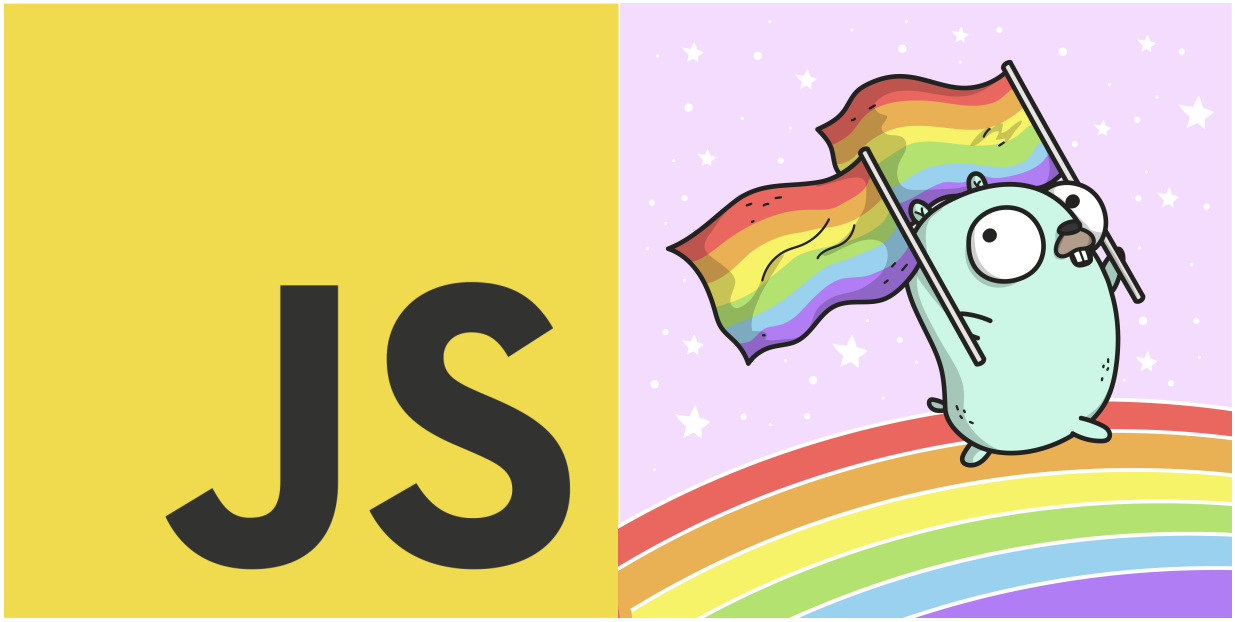
JS → GO



@CHRISBISCARDI


honeycomb.io


coffee

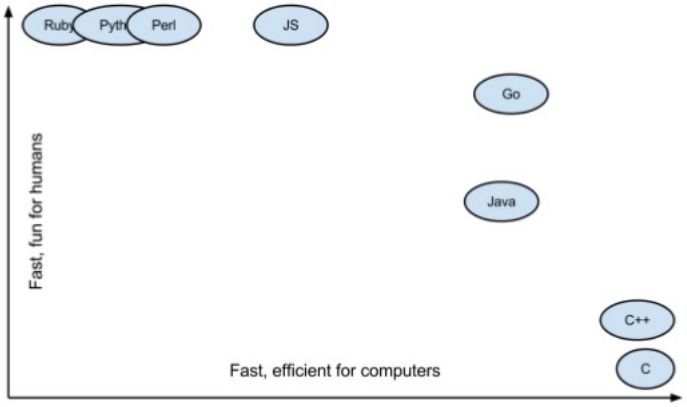

@chrisbiscardi

WHY LISTEN TO ME?

- Consulting
- UI Team @Docker
- Design Systems @Dropbox
- Product @Honeycomb

WHY

"Go: 90% Perfect, 100% of the time" -bradfitz, 2014



WHAT IS GO USEFUL FOR?

- Static Binaries
- Backend Services
- gRPC
- SQL DBs
- Concurrency
- Machine Code (no VM)

WHAT DOES GO LEAVE OUT?

- No Classes
- No Inheritance
- No Constructors
- No Exceptions

-CLASSES

```
class Thing {
  state = { value: 0 }
  doTheThing() {
    console.log('did it')
  }
}
```

-INHERITANCE

```
class Thing extends React.Component {
  render() {
    return <div></div>
  }
}
```

+INTERFACES

```
type Animal interface {
  Name() string
}
type Dog struct {}
func (d *Dog) Name() string {
  return “Dog”
}
func (d *Dog) Bark() {
  fmt.Println(“Woof!”)
}
```

-CONSTRUCTORS

```
class Thing {
  constructor(things) {
    super(things)
    console.log('constructed');
  }
}
const T = new Thing()
```

+FACTORIES

```
type Thing struct {
    Name  string
    Num   int
}
func NewThing(someParameter string) *Thing {
  p := new(Thing)
  p.Name = someParameter
  p.Num = 33 // <- a very sensible default value
  return p
}
```

-EXCEPTIONS

```
try {
  throw new Error('my special error')
} catch (e) {
  console.log('failed')
}
```

+ERRBACKS?

```
f, err := os.Open("filename.ext")
if err != nil {
    log.Fatal(err)
}
// do something with the open *File f
```

GO IS SMOL

break case chan const continue default defer
else fallthrough for func go goto if import
interface map package range return select
struct switch type var

FIRST GO APP

INSTALLATION

```
brew install nodejs

brew install golang
```

BINARIES

- go
- godoc
- gofmt

GODOC.ORG

https://godoc.org/github.com/honeycombio/honeycomb-go-magic

PROJECT INITIALIZATION (JS)

```
mkdir my-project && cd my-project
yarn init
touch index.js
yarn add --dev jest babel babel-preset-env rollup ...
```

```
yarn global add express-generator
express my-new-project
```

PROJECT INITIALIZATION (GO)

```
mkdir $GOPATH/src/github.com/christopherbiscardi/my-project
touch main.go
go build
go test
```

```
go get github.com/spf13/cobra/cobra
cobra init github.com/christopherbiscardi/my-new-cli
```

go

```
build       compile packages and dependencies
clean       remove object files and cached files
doc         show documentation for package or symbol
env         print Go environment information
bug         start a bug report
fix         update packages to use new APIs
fmt         gofmt (reformat) package sources
generate    generate Go files by processing source
get         download and install packages and dependencies
install     compile and install packages and dependencies
list        list packages
run         compile and run Go program
test        test packages
tool        run specified go tool
version     print Go version
vet         report likely mistakes in packages
```

dep

```
$ brew install dep
$ dep init
$ ls
Gopkg.toml Gopkg.lock vendor/
$ dep ensure
$ dep ensure -add github.com/pkg/errors
```

OUR FIRST API

EXPRESS

```
var app = express();

// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'jade');

app.use(logger('dev'));
app.use(express.json());
app.use(express.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));

app.use('/', indexRouter);
app.use('/users', usersRouter);

// catch 404 and forward to error handler
app.use(function(req, res, next) {
  next(createError(404));
});
```

go net/http

```
package main

import (
    "fmt"
    "log"
    "net/http"
)

func handler(w http.ResponseWriter, r *http.Request) {
    fmt.Fprintf(w, "Hi there, I love %s!", r.URL.Path[1:])
}

func main() {
    http.HandleFunc("/", handler)
    log.Fatal(http.ListenAndServe(":8080", nil))
}
```

gorilla/mux

```
import ( "github.com/gorilla/mux" )

func main() {
  r := mux.NewRouter()
  r.HandleFunc("/books/{title}/page/{page}", func(w
http.ResponseWriter, r *http.Request) {
    vars := mux.Vars(r)
    title := vars["title"]
    page := vars["page"]

    fmt.Fprintf(w, "requested book: %s on page %s
", title, page)
  })

  http.ListenAndServe(":80", r)
}
```

CONFIGURATION

```
if(process.env.SQL_DB_URL) {
  // connect to db
}
```

> **Viper is a complete configuration solution for Go applications ... and can handle all types of configuration needs and formats.**

- viper docs

READ ENV VAR

```
SetEnvPrefix("spf") // will be uppercased automatically
BindEnv("id")

id := Get("id") // SPF_ID
```

```
var runtime_viper = viper.New()
runtime_viper.AddRemoteProvider("etcd",
"http://127.0.0.1:4001", "/config/hugo.yml")
runtime_viper.SetConfigType("yaml")
err := runtime_viper.ReadRemoteConfig()
runtime_viper.Unmarshal(&runtime_conf)

// open a goroutine to watch remote changes forever
go func(){
  for {
      time.Sleep(time.Second * 5) // delay after each request

          err := runtime_viper.WatchRemoteConfig()
      if err != nil {
          log.Errorf("unable to read remote config: %v", err)
          continue
      }

      runtime_viper.Unmarshal(&runtime_conf)
  }
}()
```

TALKING TO DATABASES

KNEX

```
knex('users').where({
  first_name: 'Test',
  last_name:  'User'
}).select('id')
```

```
Outputs:\n\n select `id` from `users` where `first_name` =
'Test' and `last_name` = 'User'
```

DATABASE/SQL

```
var age int64
row := db.QueryRow("SELECT age FROM users WHERE name = $1",
name)
err := row.Scan(&age)
```

JMOIRON/SQLX

```
type Person struct {
    FirstName string `db:"first_name"`
    LastName  string `db:"last_name"`
    Email     string
}
people := []Person{}
db.Select(&people, "SELECT * FROM person ORDER BY first_name
ASC")
jason, john := people[0], people[1]
fmt.Printf("%#v\n%#v", jason, john)
// Person{FirstName:"Jason", LastName:"Moiron",
Email:"jmoiron@jmoiron.net"}
// Person{FirstName:"John", LastName:"Doe",
Email:"johndoeDNE@gmail.net"}
```

BOOKSHELF

```
User.where("id", 1)
  .fetch({ withRelated: ["posts.tags"] })
  .then(function(user) {
    console.log(user.related("posts").toJSON());
  })
  .catch(function(err) {
    console.error(err);
  });
```


gorm

BUILD & DEPLOY

STATIC BUILDS

```
CGO_ENABLED=0 go build -a -ldflags '-extldflags "-static"' .
```

```
CGO_ENABLED=0 GOOS=windows GOARCH=386 go build -a -ldflags '-extldflags "-static"' .
```

CONTAINERIZED

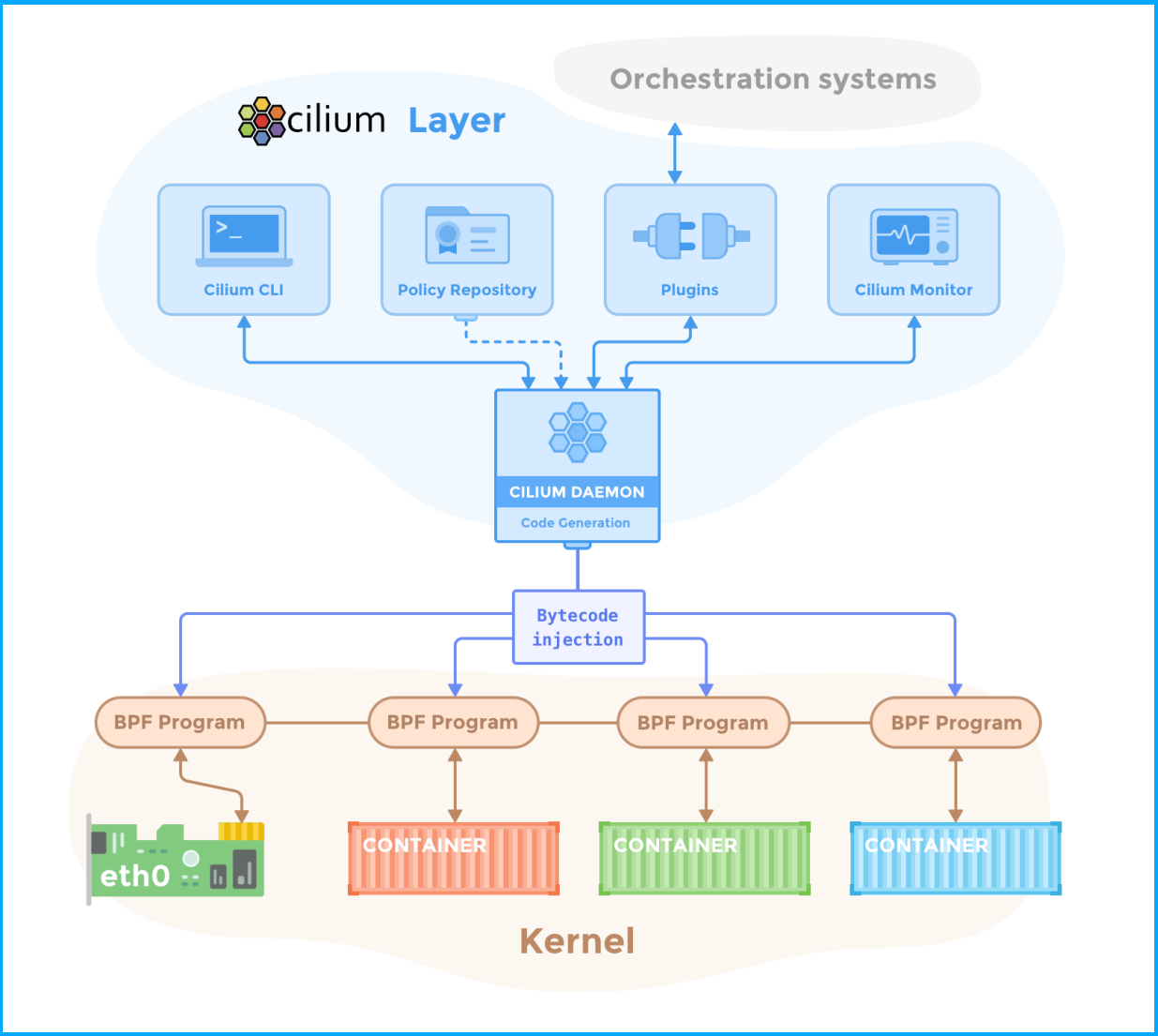
```
FROM golang:1.10

WORKDIR /go/src/github.com/christopherbiscardi/my-project
COPY . /go/src/github.com/christopherbiscardi/my-project
RUN go install
ENTRYPOINT "my-project"
```

CONTAINERIZED

```
FROM scratch
COPY ./my-project /opt/my-project
ENTRYPOINT ["/opt/my-project"]
```

```
docker build -t my-container .
docker run -itp 8080:8080 my-container
```

EXPLORATION


Orchestration systems
cilium Layer
Cilium CLI
Policy Repository
Plugins
Cilium Monitor
CILIUM DAEMON
Code Generation
Bytecode injection
BPF Program
BPF Program
BPF Program
BPF Program
eth0
CONTAINER
CONTAINER
CONTAINER
Kernel

cilium/cmd/kvstore.go

```
1. // Copyright 2018 Authors of Cilium
2. //
3. // Licensed under the Apache License, Version 2.0 (the "License");
4. // you may not use this file except in compliance with the License.
5. // You may obtain a copy of the License at
6. //
7. //     http://www.apache.org/licenses/LICENSE-2.0
8. //
```