Go in a Microsoft Shop

Go overview · Environment setup · Code examples

Smart Devs User Group
July 2014

harvey@documize.com
@HarveyKandola
github.com/HarveyKandola

BACKGROUND

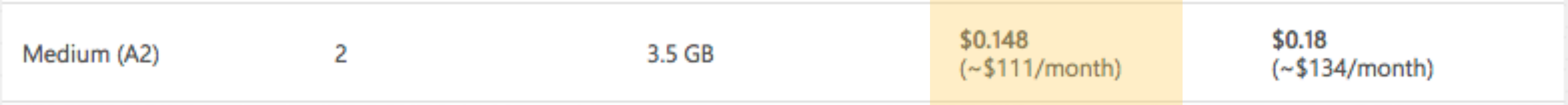
- Previously founded Countersoft
- Historically .NET ISV because we sold to enterprises, on-premise
- Software deployed to NASA, Dell, NHS, US.gov, banks, healthcare
- Background in Fortran, VB, C++, Java and.NET
- Latest start-up Documize is written Go, SaaS + on-premise
- Always seeking an edge to help us improve

Why not stick with .NET?

KEY ISSUES

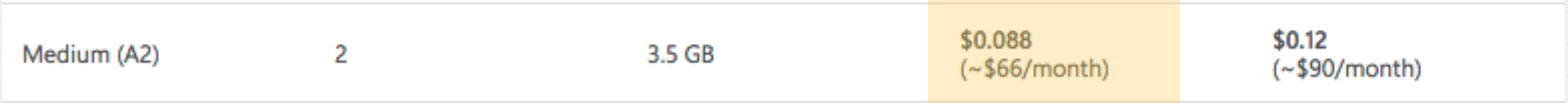
- World moving towards cloud apps, mobile apps — cost
- But on-premise still remains — deployment
- .NET start-up community — who's else is here?
- Sometimes Windows can hurt a little…

Windows VM (source: Microsoft Azure, April 14)



Medium (A2)	2	3.5 GB	$0.148 (~$111/month)	$0.18 (~$134/month)

Linux VM



Medium (A2)	2	3.5 GB	$0.088 (~$66/month)	$0.12 (~$90/month)

Microsoft

WHY GO?

- Supports Linux cloud, Windows on-premise
- +30% cheaper to run cloud stack
- Go more "safe" than Mono for cross-platform
- Better compile times for large codebases
- Better web app start-up times
- Better deployment experience
- Story for Corporate IT Police

IT POLICE FRIENDLY

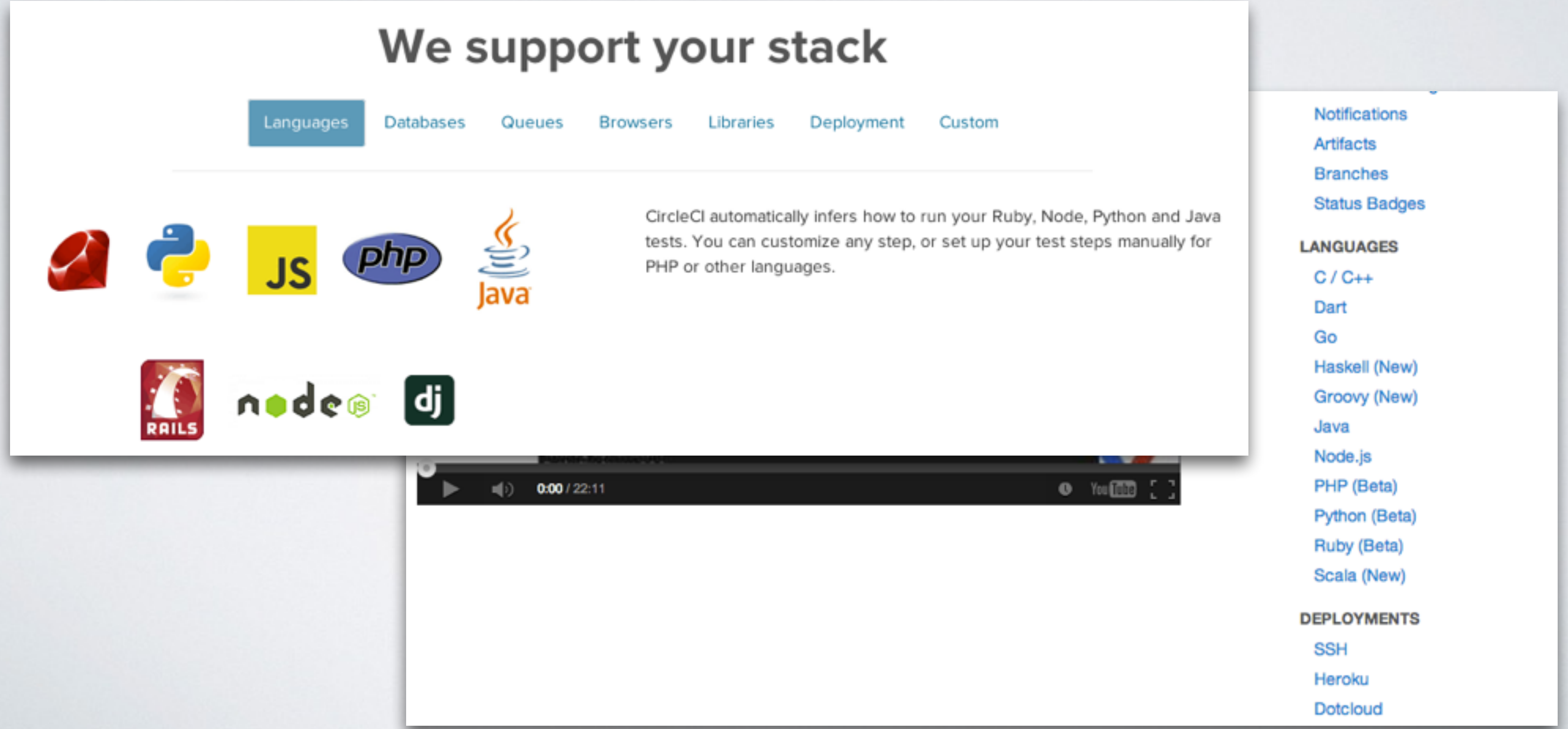
- Deploy Go on standard, corporate IT server images
- Go web apps can run under Microsoft IIS
- IIS for SSL termination
- IIS Application Request Routing

MAINTENANCE COSTS

- Code is formatted the same way, so easy to read
- Code “does exactly what it says on the page”
- The language is small, so easy for new hires

SIDE NOTE: SAAS SERVICES

Why is .NET less supported?!



We support your stack

Languages · Databases · Queues · Browsers · Libraries · Deployment · Custom

CircleCI automatically infers how to run your Ruby, Node, Python and Java tests. You can customize any step, or set up your test steps manually for PHP or other languages.

- Notifications
- Artifacts
- Branches
- Status Badges

LANGUAGES

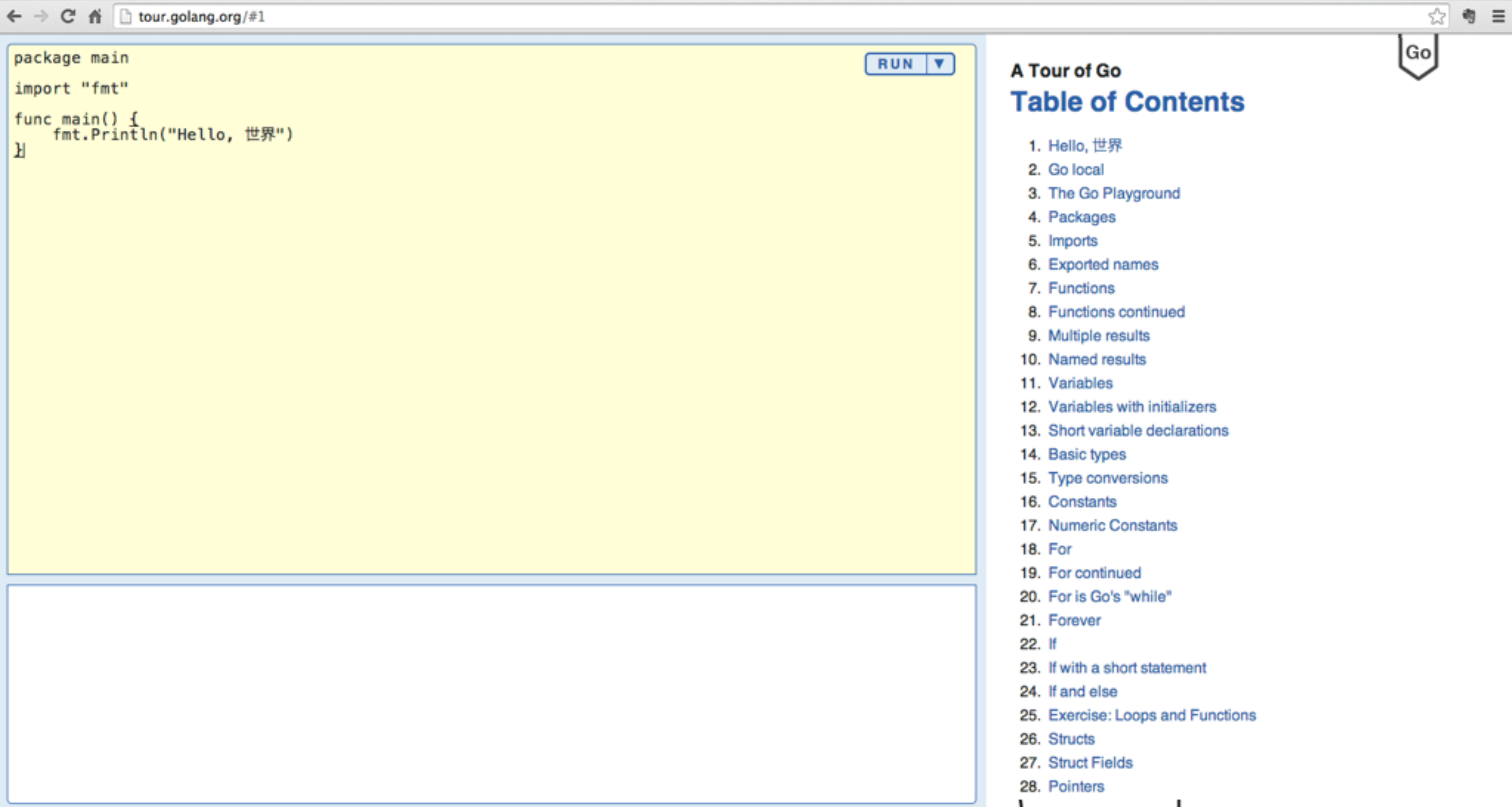
- C / C++
- Dart
- Go
- Haskell (New)
- Groovy (New)
- Java
- Node.js
- PHP (Beta)
- Python (Beta)
- Ruby (Beta)
- Scala (New)

DEPLOYMENTS

- SSH
- Heroku
- Dotcloud

GO INTRO

TOUR.GOLANG.ORG



```
package main

import "fmt"

func main() {
	fmt.Println("Hello, 世界")
}
```

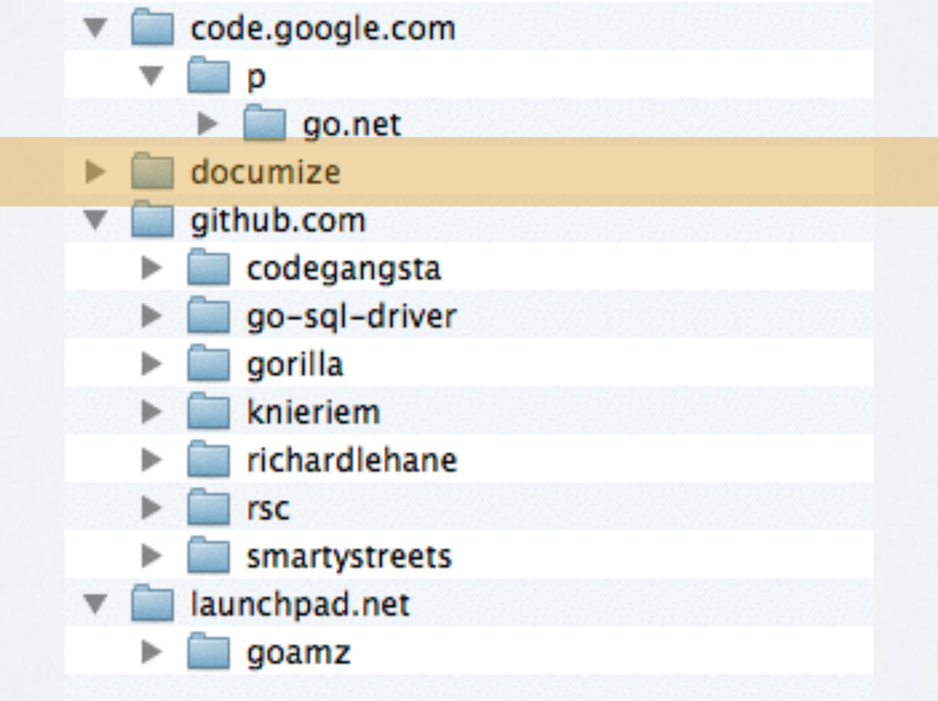
A Tour of Go

Table of Contents

1. Hello, 世界
2. Go local
3. The Go Playground
4. Packages
5. Imports
6. Exported names
7. Functions
8. Functions continued
9. Multiple results
10. Named results
11. Variables
12. Variables with initializers
13. Short variable declarations
14. Basic types
15. Type conversions
16. Constants
17. Numeric Constants
18. For
19. For continued
20. For is Go's "while"
21. Forever
22. If
23. If with a short statement
24. If and else
25. Exercise: Loops and Functions
26. Structs
27. Struct Fields
28. Pointers

WORKSPACES

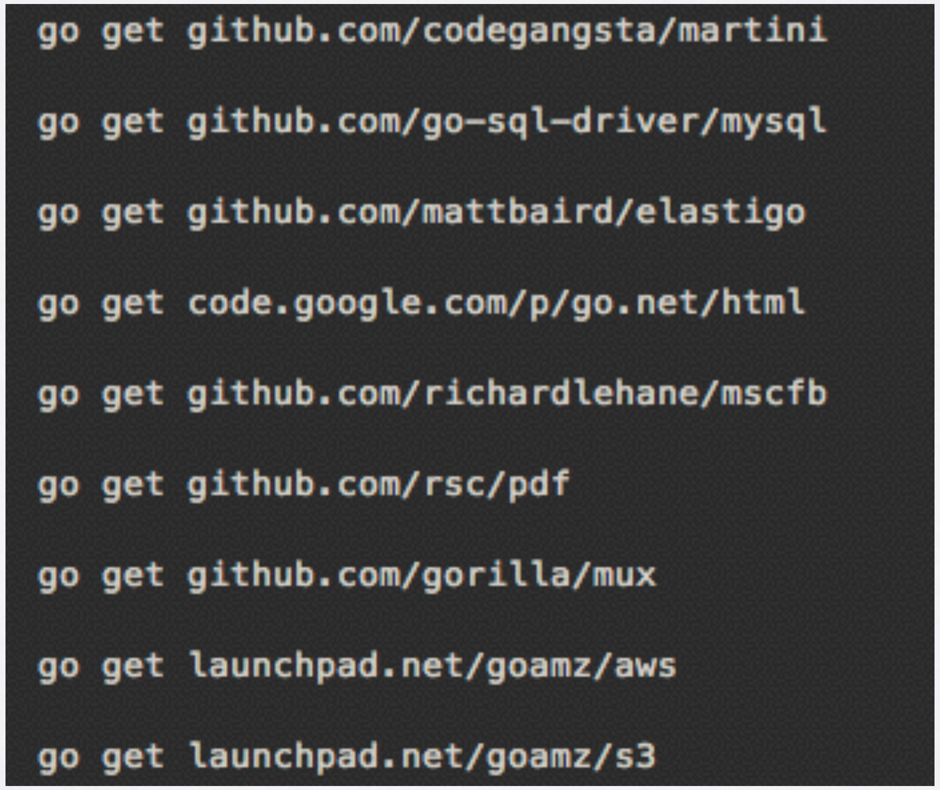
- **bin** contains compiled binaries
- **pkg** contains package related .a files
- **src** is where your code lives + 3rd party libs



```
▼ code.google.com
    ▼ p
        ▶ go.net
▶ documize
▼ github.com
    ▶ codegangsta
    ▶ go-sql-driver
    ▶ gorilla
    ▶ knieriem
    ▶ richardlehane
    ▶ rsc
    ▶ smartystreets
▼ launchpad.net
    ▶ goamz
```

THIRD PARTY LIBS

- **go get** will "download and install" 3rd party libs (nuget)
- Everything is compiled from source and fetched from repos
- No notion of assemblies, DLL's, versioning, vendoring

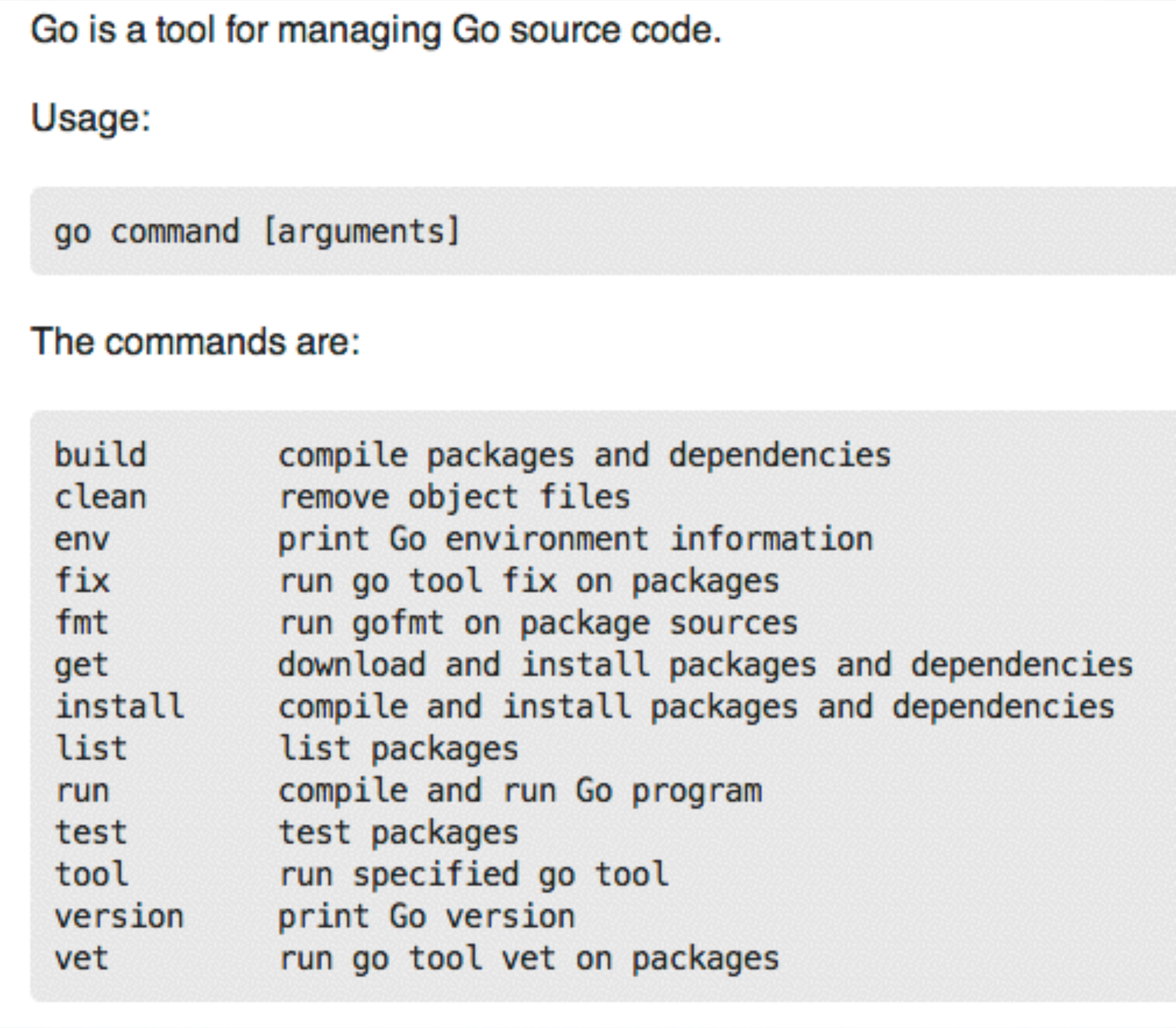


```
go get github.com/codegangsta/martini

go get github.com/go-sql-driver/mysql

go get github.com/mattbaird/elastigo

go get code.google.com/p/go.net/html

go get github.com/richardlehane/mscfb

go get github.com/rsc/pdf

go get github.com/gorilla/mux

go get launchpad.net/goamz/aws

go get launchpad.net/goamz/s3
```

WORKSPACE PATH

- **GOPATH** should always point to workspace (**bin, pkg, src**)
- (Linux) export GOPATH=**$HOME/my/code**
- (Windows) set GOPATH=**D:\my\code**

ONE TOOL



Go is a tool for managing Go source code.

Usage:

```
go command [arguments]
```

The commands are:

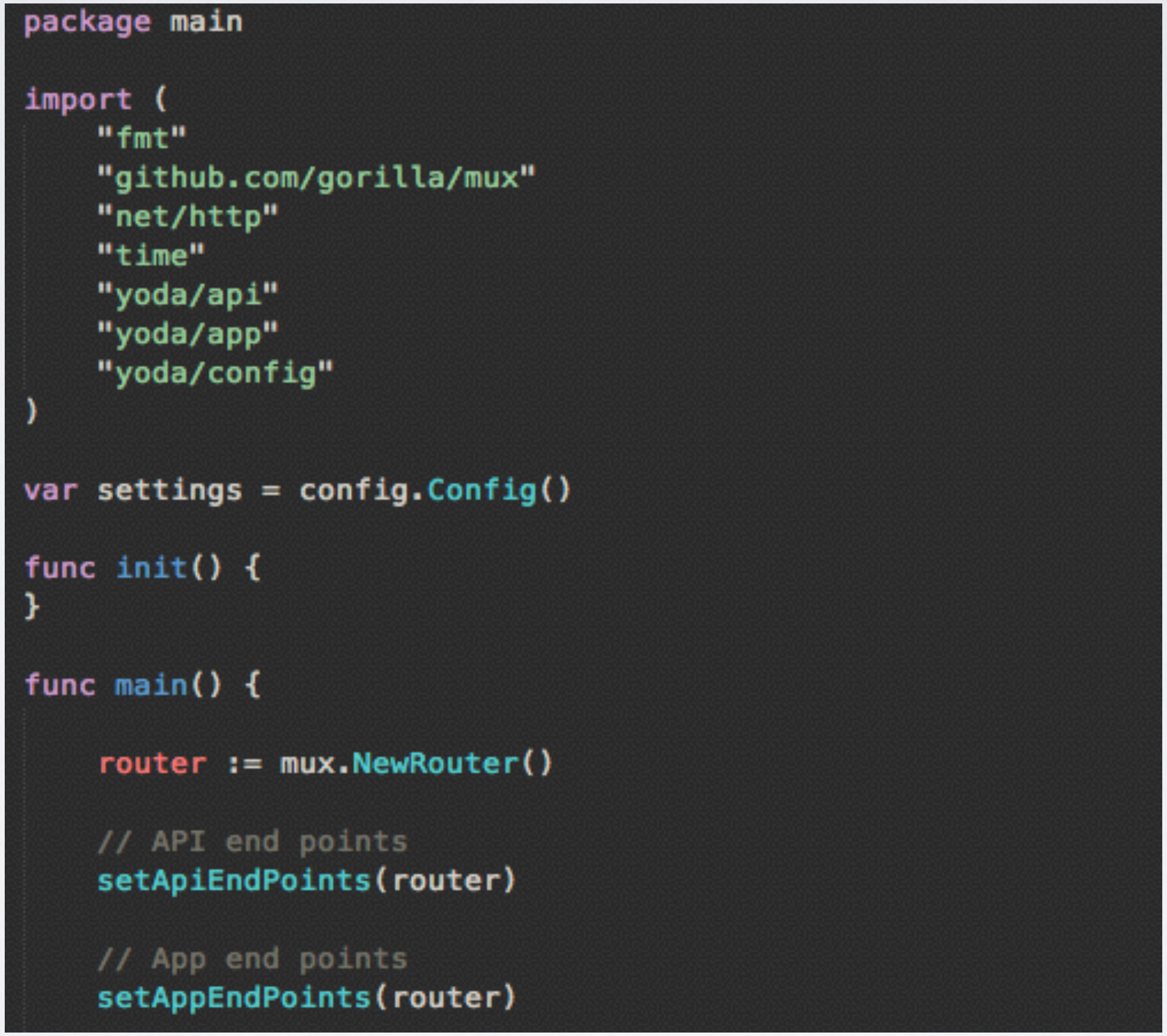
```
build       compile packages and dependencies
clean       remove object files
env         print Go environment information
fix         run go tool fix on packages
fmt         run gofmt on package sources
get         download and install packages and dependencies
install     compile and install packages and dependencies
list        list packages
run         compile and run Go program
test        test packages
tool        run specified go tool
version     print Go version
vet         run go tool vet on packages
```

BUILDING

```
go install my/hello
$GOPATH/bin/hello
```

```
go build my/hello
./hello
```

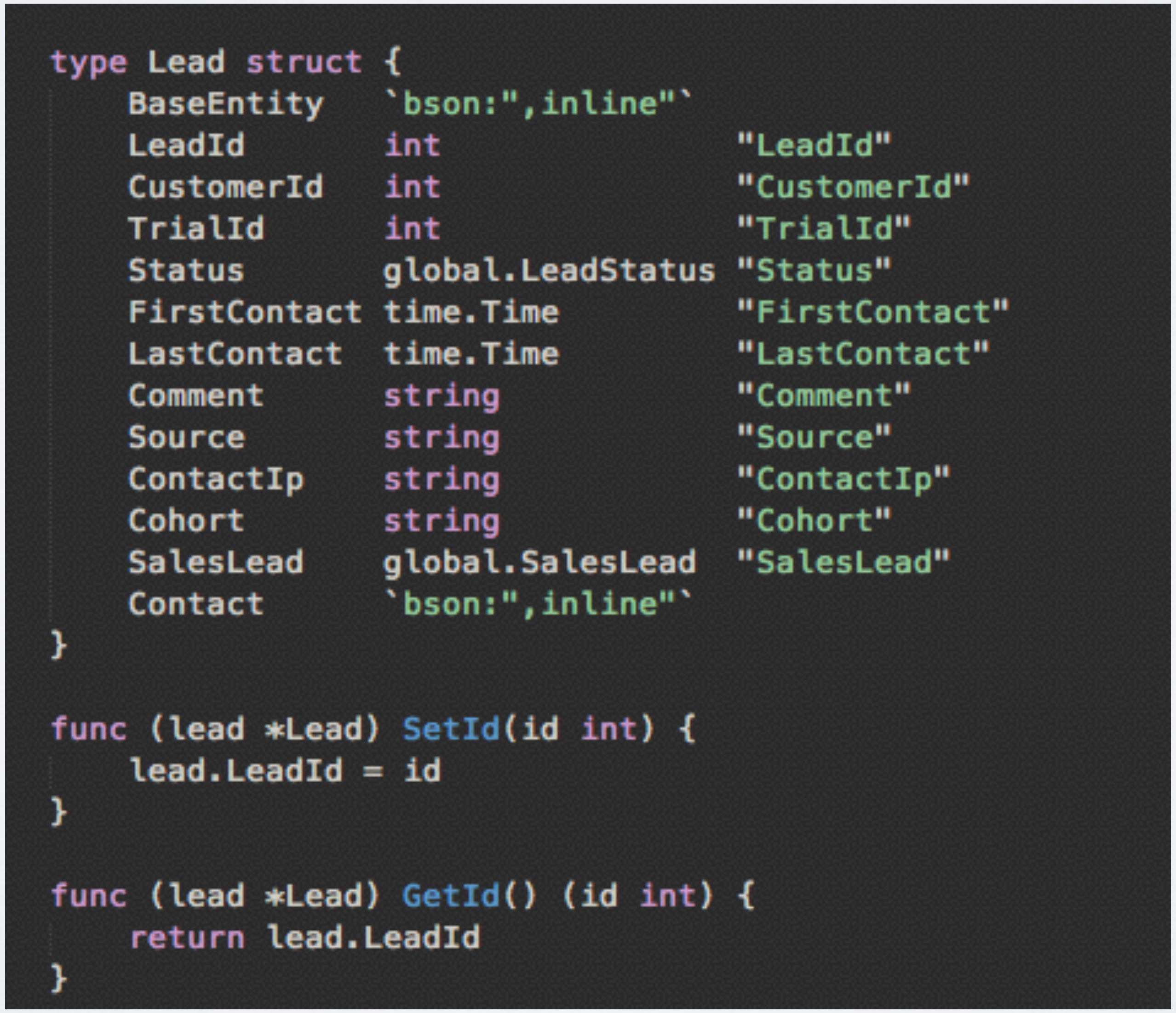
CODE STRUCTURE



```go
package main

import (
    "fmt"
    "github.com/gorilla/mux"
    "net/http"
    "time"
    "yoda/api"
    "yoda/app"
    "yoda/config"
)

var settings = config.Config()

func init() {
}

func main() {

    router := mux.NewRouter()

    // API end points
    setApiEndPoints(router)

    // App end points
    setAppEndPoints(router)
```

DIFFERENT OO APPROACH

- No inheritance, just composition
- Polymorphism via interfaces



```
type Lead struct {
    BaseEntity   `bson:",inline"`
    LeadId       int                 "LeadId"
    CustomerId   int                 "CustomerId"
    TrialId      int                 "TrialId"
    Status       global.LeadStatus "Status"
    FirstContact time.Time           "FirstContact"
    LastContact  time.Time           "LastContact"
    Comment      string              "Comment"
    Source       string              "Source"
    ContactIp    string              "ContactIp"
    Cohort       string              "Cohort"
    SalesLead    global.SalesLead  "SalesLead"
    Contact      `bson:",inline"`
}

func (lead *Lead) SetId(id int) {
    lead.LeadId = id
}

func (lead *Lead) GetId() (id int) {
    return lead.LeadId
}
```

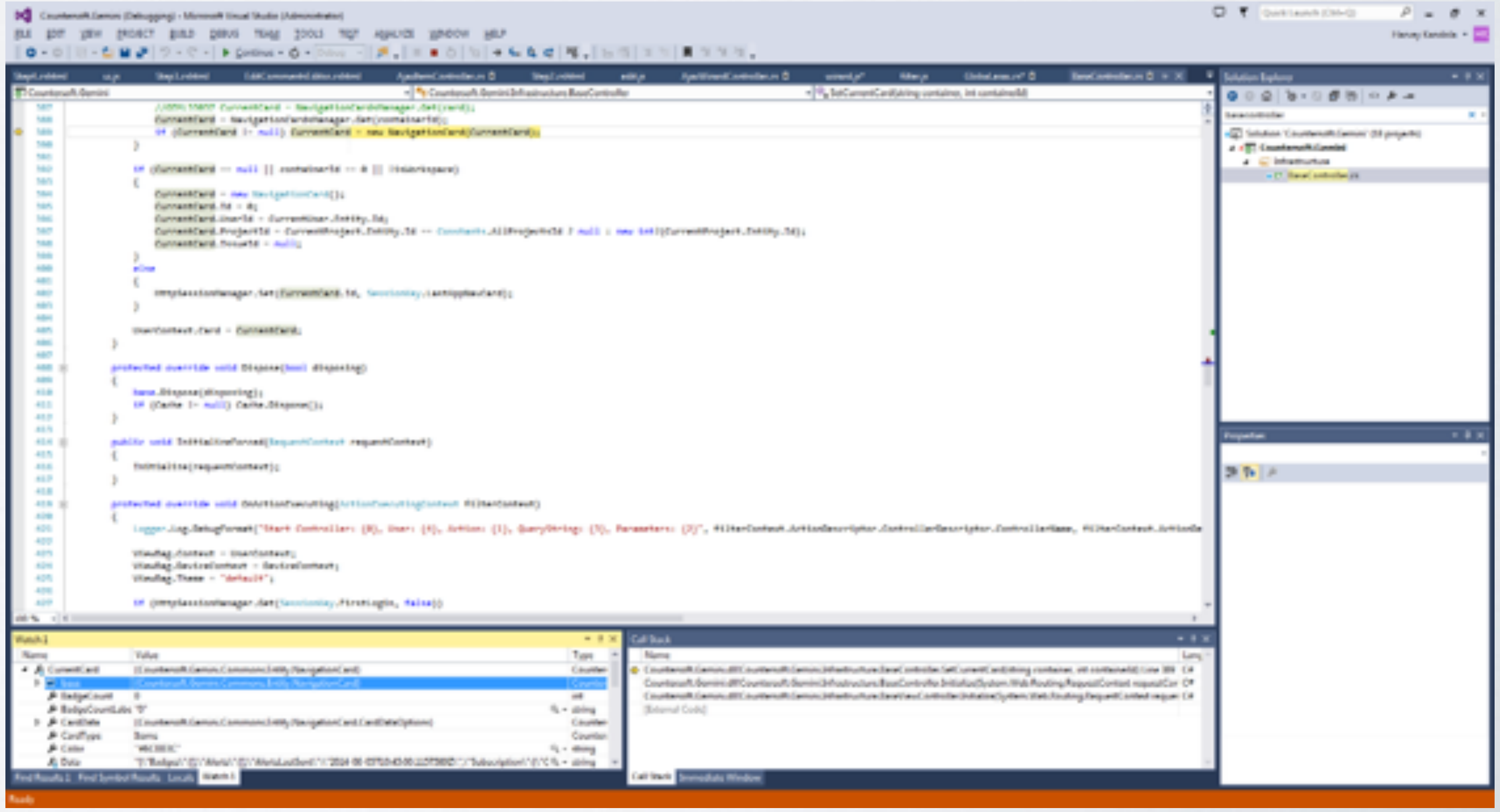
HANDLING ERRORS!

- No try-catch!
- Handle error or chuck it up

```
if err != nil {
    log.Fatalf("it all went horribly wrong: %v", err)
}
```

SOME INITIAL PAIN

- Miss JetBrains ReSharper refactoring
- 3rd party library dependency issue (vendor'ing)
- GOPATH & workspaces mind-set
- Visual Studio debugger



NOT JUST GO

- **Angular.js** for client-side
- **Node + Grunt.js** for building-zipping-shipping
- **Drone.io** for continuous delivery
- **Dash** for help docs (offline!)
- **Sublime Text** for "IDE"

Build stateless web apps with Go + Angular

CODE!