

A short horizontal bar with a teal segment on the left and an orange segment on the right.

Application testing

Ilija Matoski
ilijamt@gmail.com

2024-02-27



<https://matoski.com>

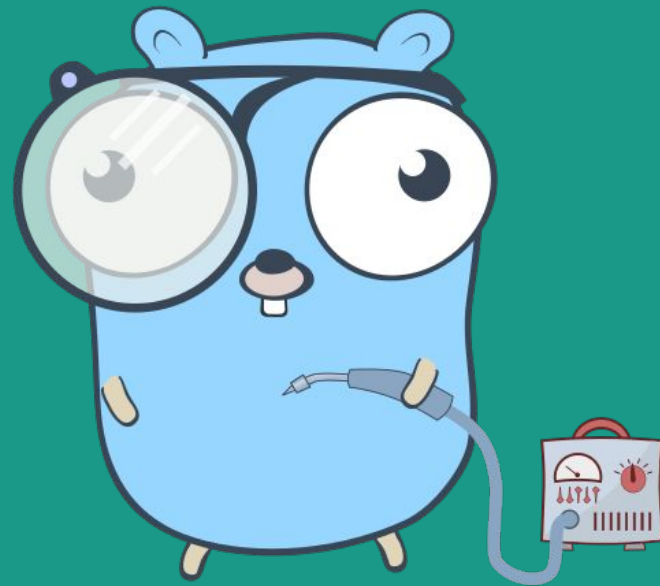


<https://linkedin.com/ilijamt>



<https://github.com/ilijamt>

-
- Collecting and reporting coverage data for Go unit tests
 - Code coverage in application testing
 - New codecov tool
 - runtime/coverage package
 - Example





go test


- › `go test -coverprofile=<filename> [package target(s)]`
- › `go tool cover -func=<covdatafile>`
- › `go tool cover -html=<covdatafile>`





How did go test work before go1.20

```
func ABC(x int) {  
    if x < 0 {  
        bar()  
    }  
}
```



```
func ABC(x int) {GoCover_0_343662613637653164643337.Count[9] = 1;  
    if x < 0 {GoCover_0_343662613637653164643337.Count[10] = 1;  
        bar()  
    }  
}
```





New changes

- Building for coverage
- Running instrumented applications
- Programs that call `os.Exit()`, or never terminate





Running instrumented applications

- › `go build -o myapp.exe -cover ...`
- › `mkdir /tmp/mycovdata`
- › `export GOCOVERDIR=/tmp/mycovdata`
- › `<run test suite, resulting in multiple invocations of myapp.exe>`
- › `go tool cover -html=/tmp/mycovdata`

`covdata.<metafilehash>.<processid>.<nanotimevalue>.out`



Programs that call `os.Exit()`, or never terminate

```
import "runtime/coverage"

var *coverageoutdir flag.String(...)

func server() {
    ...
    if *coverageoutdir != "" {
        // Meta-data is already available on program startup; write it now.
        // NB: we're assuming here that the specified dir already exists
        if err := coverage.EmitMetaDataToDir(*coverageoutdir); err != nil {
            log.Fatalf("writing coverage meta-data: %v")
        }
    }
    for {
        ...
        if *coverageoutdir != "" && <received signal to emit coverage data> {
            if err := coverage.EmitCounterDataToDir(*coverageoutdir); err != nil {
                log.Fatalf("writing coverage counter-data: %v")
            }
        }
    }
}
```



Coverage



- Coverage and modules
- Merging coverage data output files
- Differential coverage



Package selection when building instrumented applications



// Collects coverage for only the github.com/spf13/cobra package
› go build -cover -coverpkg=github.com/spf13/cobra ./cmd/example

// Collects coverage for all packages in the main module and
// all packages listed as dependencies in go.mod
› go build -cover -coverpkg=mod.deps ./cmd/example

// Collects coverage for all packages (including the Go std library)
› go build -cover -coverpkg=all ./cmd/example



Coverage instrumentation: compiler or tool or may hybrid?



```
L11: func ABC(x int) {  
L12:     if x < 0 {  
L13:         bar()  
L14:     }  
L15: }
```



Hybrid instrumentation

```
package p
```

```
L4:  func small(x, y int) int {  
L5:      v++  
L6:      // comment  
L7:      if y == 0 || x < 0 {  
L8:          return x  
L9:      }  
L10:     return (x << 1) ^ (9 / y)  
L11: }  
L12:  
L13: func medium() {  
L14:     s1 := small(q, r)  
L15:     z += s1  
L16:     s2 := small(r, q)  
L17:     w -= s2  
L18: }
```



Function metadata and counter array

| Unit index | File | Start pos | End pos | Number of statements |
|------------|------|-----------|---------|----------------------|
| 0 | F0 | L5 | L7 | 2 |
| 1 | F0 | L8 | L8 | 1 |

```
struct {  
    numCtrs uint32  
    pkgId uint32  
    funcId uint32  
    counterArray [numUnits]uint32  
}
```



Example

```
01: package p
02:
03: var v, w, z int
04:
05: func small(x, y int) int {
06:     v++
07:     // comment
08:     if y == 0 {
09:         return x
10:     }
11:     return (x << 1) ^ (9 / y)
12: }
13:
14: func Medium(q, r int) int {
15:     s1 := small(q, r)
16:     z += s1
17:     s2 := small(r, q)
18:     w -= s2
19:     return w + z
20: }
```



```
--header-----
| size: size of this blob in bytes
| packagepath: <path to p>
| module: <modulename>
| classification: ...
| nfiles: 1
| nfunctions: 2
--file + function table-----
| <uleb128 len> 4
| <uleb128 len> 5
| <uleb128 len> 6
| <data> "p.go"
| <data> "small"
| <data> "Medium"
--func 1-----
| uint32 num units: 3
| uint32 func name: 1 (index into string table)
| <unit 0>:  F0    L6    L8    2
| <unit 1>:  F0    L9    L9    1
| <unit 2>:  F0   L11   L11    1
--func 2-----
| uint32 num units: 1
| uint32 func name: 2 (index into string table)
| <unit 0>:  F0   L15   L19    5
--end-----
```



go tool covdata

› go tool covdata

error: missing command selector

usage: go tool covdata [command]

Commands are:

textfmt convert coverage data to textual format

percent output total percentage of statements covered

pkglist output list of package import paths

func output coverage profile information for each function

merge merge data files together

subtract subtract one set of data files from another set

intersect generate intersection of two sets of data files

debugdump dump data in human-readable format for debugging purposes



go tool covdata merge



› ls windows_datadir

covcounters.f3833f80c91d8229544b25a855285890.1025623.1667481441036838252

covcounters.f3833f80c91d8229544b25a855285890.1025628.1667481441042785007

covmeta.f3833f80c91d8229544b25a855285890

› ls macos_datadir

covcounters.b245ad845b5068d116a4e25033b429fb.1025358.1667481440551734165

covcounters.b245ad845b5068d116a4e25033b429fb.1025364.1667481440557770197

covmeta.b245ad845b5068d116a4e25033b429fb

› ls macos_datadir

› mkdir merged

› go tool covdata merge -i=windows_datadir,macos_datadir -o merged



go tool covdata - Package selection

› ls somedata

covcounters.c6de772f99010ef5925877a7b05db4cc.2424989.1670252383678349347

covmeta.c6de772f99010ef5925877a7b05db4cc

› go tool covdata percent -i=somedata -pkg=mydomain.com/greetings

mydomain.com/greetings coverage: 100.0% of statements

› go tool covdata percent -i=somedata -pkg=nonexistentpackage



go tool covdata - Conversion to legacy format

➤ go tool covdata textfmt -i=final -o=covdata.txt

➤ head covdata.txt

mode: set

cov-example/p/p.go:7.22,8.2 0 0

cov-example/p/p.go:10.31,11.2 1 0

cov-example/p/p.go:11.3,13.3 0 0

cov-example/p/p.go:14.3,16.3 0 0

cov-example/p/p.go:19.33,21.2 1 1

cov-example/p/p.go:23.22,25.2 1 0

...

➤ go tool cover -func=covdata.txt | head

| | | |
|------------------------------|------|-------|
| cov-example/main/main.go:12: | main | 90.0% |
|------------------------------|------|-------|

| | | |
|-----------------------|---------|------|
| cov-example/p/p.go:7: | emptyFn | 0.0% |
|-----------------------|---------|------|

...



go tool covdata

- GO111MODULE=off
- Panics, fatal exceptions, segmentation violation,

```
> go list -f '{{if not .Standard}}{{.ImportPath}}{{end}}' -deps . | paste -sd "," > pkgs.txt
> go build -o myprogram.exe -coverpkg=`cat pkgs.txt` .
> mkdir somedata
> GOCOVERDIR=somedata ./myprogram.exe
> go tool covdata percent -i=somedata
  golang.org/x/text/internal/tag coverage: 78.4% of statements
  golang.org/x/text/language coverage: 35.5% of statements
  mydomain.com coverage: 100.0% of statements
  mydomain.com/greetings coverage: 100.0% of statements
  rsc.io/quote coverage: 25.0% of statements
  rsc.io/sampler coverage: 86.7% of statements
```



Future work, limitations

- Function-level coverage
- Tagging coverage profiles to support test “origin” queries
- Taking into account panic paths
- Intra-line coverage

```
L1: func myFunc() int {  
L2:     defer func() { cleanup() }()  
L3:     dosomework()  
L4:     mayPanic()  
L5:     morework()  
L6:     if condition2 {  
L7:         launchOperation()  
L8:     }  
L9:     return x  
L10: }
```



Future - Intra-line coverage

```
L4:  func small(x, y int) int {  
L5:      v++  
L6:      // comment  
L7:      if y == 0 || *x < 0 {  
L8:          return x  
L9:      }  
L10:     return (x << 1) ^ (9 / y)  
L11: }
```

```
L7a:  if y == 0  
L7b:      || *x < 0 {
```



github.com/ilijamt/terraform-provider-awx

```
> go build -cover -trimpath -o terraform-provider-awx ./cmd/provider
> ./provider/terraform-provider-awx
warning: GOCOVERDIR not set, no coverage data emitted
This binary is a plugin. These are not meant to be executed directly.
Please execute the program that consumes these plugins, which will
load any plugins automatically
> mkdir coverage/terraform-plan
> GOCOVERDIR=coverage/terraform-plan terraform plan -out plan
> go tool covdata percent -i coverage/terraform-plan
    github.com/ilijamt/terraform-provider-awx/cmd/provider           coverage: 83.3% of statements
    github.com/ilijamt/terraform-provider-awx/internal/awx          coverage: 5.8% of statements
    github.com/ilijamt/terraform-provider-awx/internal/client        coverage: 64.1% of statements
    github.com/ilijamt/terraform-provider-awx/internal/helpers       coverage: 23.6% of statements
    github.com/ilijamt/terraform-provider-awx/internal/hooks         coverage: 0.0% of statements
    github.com/ilijamt/terraform-provider-awx/internal/models
    github.com/ilijamt/terraform-provider-awx/internal/provider      coverage: 81.4% of statements
    github.com/ilijamt/terraform-provider-awx/version
> go tool covdata textfmt -i coverage/terraform-plan-apply -o coverage/terraform-plan-apply.cov
```



github.com/ilijamt/terraform-provider-awx

```
> go tool covdata intersect -i coverage/terraform-plan,coverage/terraform-plan-apply
> go tool covdata textfmt -i coverage/intersect-t-p-a -o coverage/intersect-t-p-a.cov
> go tool cover -html ../goams-meetup-application-testing/awx/coverage/intersect-t-p-a.cov -o
../goams-meetup-application-testing/awx/coverage/intersect-t-p-a.html

> go tool covdata subtract -i coverage/terraform-plan,coverage/terraform-plan-apply -o coverage/subtract-t-p-a
> go tool covdata textfmt -i coverage/subtract-t-p-a -o coverage/subtract-t-p-a.cov
> go tool cover -html ../goams-meetup-application-testing/awx/coverage/subtract-t-p-a.cov -o
../goams-meetup-application-testing/awx/coverage/subtract-t-p-a.html

> go tool covdata percent -i coverage/subtract-t-p-a
    github.com/ilijamt/terraform-provider-awx/cmd/provider          coverage: 0.0% of statements
    github.com/ilijamt/terraform-provider-awx/internal/awx         coverage: 0.0% of statements
    github.com/ilijamt/terraform-provider-awx/internal/client       coverage: 0.0% of statements
    github.com/ilijamt/terraform-provider-awx/internal/helpers      coverage: 0.0% of statements
    github.com/ilijamt/terraform-provider-awx/internal/hooks        coverage: 0.0% of statements
    github.com/ilijamt/terraform-provider-awx/internal/models
github.com/ilijamt/terraform-provider-awx/internal/provider        coverage: 0.0% of statements
    github.com/ilijamt/terraform-provider-awx/version              %
```



github.com/ilijamt/vault-plugin-secrets-gitlab

```
> go tool covdata percent -i coverage/vault-general
github.com/ilijamt/vault-plugin-secrets-gitlab          coverage: 45.0% of statements
github.com/ilijamt/vault-plugin-secrets-gitlab/cmd/vault-plugin-secrets-gitlab  coverage: 77.8% of statements
> go tool covdata percent -i coverage/vault-gitlab
github.com/ilijamt/vault-plugin-secrets-gitlab          coverage: 40.6% of statements
github.com/ilijamt/vault-plugin-secrets-gitlab/cmd/vault-plugin-secrets-gitlab  coverage: 77.8% of statements
> go tool covdata intersect -i coverage/vault-general,coverage/vault-gitlab -o coverage/intersect
> go tool covdata subtract -i coverage/vault-general,coverage/vault-gitlab -o coverage/subtract
> go tool cover -html coverage/subtract.cov -o coverage/subtract.html
> go tool cover -html coverage/intersect.cov -o coverage/intersect.html
> go test -cover ./... -args -test.gocoverdir="coverage/unit"
> go tool covdata textfmt -i coverage/merged -o coverage/merged.cov
> go tool covdata percent -i coverage/merged
github.com/ilijamt/vault-plugin-secrets-gitlab          coverage: 88.2% of statements
github.com/ilijamt/vault-plugin-secrets-gitlab/cmd/vault-plugin-secrets-gitlab  coverage: 77.8% of statements
```





Questions

