Tips and Tricks with Go

25 September 2013

Levi Cook
Cory LaNou

This talk will not teach you Go

But there are great resources on it, and they have great documentation.

golang.org(http://golang.org)

This talk will include:

- formatting tips
- error handling tips
- consistent function returns

Code Resources

denvergophers.com/tips-and-tricks | (This Presentation and source)(https://denvergophers.com/tips-and-tricks)

golang.org/pkg/fmt(http://golang.org/pkg/fmt)

present (what this presentation was created with)(http://godoc.org/code.google.com/p/go.tools/present)

The fmt Package

Used by importing the fmt package as follows:

import "fmt"

General:

```
%v  the value in a default format.
%+v When printing structs, adds field names
%#v a Go-syntax representation of the value
%T  a Go-syntax representation of the type of the value
%%  a literal percent sign; consumes no value
```

fmt General verbs - simple string

```
    var foo string = "This is a simple string"
    fmt.Printf("%v\n", foo)
    fmt.Printf("%T\n", foo)
```

fmt General verbs - struct (part 1)

```
type (
    Customer struct {
        Name   string
        Street []string
        City   string
        State  string
        Zip    string
    }
    Item struct {
        Id       int
        Name     string
        Quantity int
    }
    Items []Item
    Order struct {
        Id       int
        Customer Customer
        Items    Items
    }
)
```

fmt General verbs - struct (part 2)

```
    // This is my default when debugging
    fmt.Printf("%+v\n\n", order)

    // I use this when I need to know the world about that struct
    fmt.Printf("%#v\n\n", order)

    // I seldom use these
    fmt.Printf("%v\n\n", order)
    fmt.Printf("%s\n\n", order)
    fmt.Printf("%T\n", order)
```

fmt - Generating Errors with errors.New()

This is my least favorite way of seeing erors created.

```
import (
    "errors"
    "fmt"
    "log"
)

func main() {
    if err := iDunBlowedUp(-100); err != nil {
        err = errors.New(fmt.Sprintf("Something went wrong: %s\n", err))
        log.Println(err)
        return
    }
    fmt.Printf("Success!")
}

func iDunBlowedUp(val int) error {
    return errors.New(fmt.Sprintf("invalid value %d", val))
}
```

fmt - Generating Errors with fmt.Errorf()

This is how I create errors.

```
import (
    "fmt"
    "log"
)

func main() {
    if err := iDunBlowedUp(-100); err != nil {
        err = fmt.Errorf("Something went wrong: %s\n", err)
        log.Println(err)
        return
    }
    fmt.Printf("Success!")
}

func iDunBlowedUp(val int) error {
    return fmt.Errorf("invalid value %d", val)
}
```

Consistent function Returns - Bad

```
func someFunction(val int) (ok bool, err error) {
    if val == 0 {
        return false, nil
    }
    if val < 0 {
        return false, fmt.Errorf("value can't be negative %d", val)
    }
    ok = true
    return
}
```

Consistent function Returns - Good

```go
func someFunction(val int) (bool, error) {
    if val == 0 {
        return false, nil
    }
    if val < 0 {
        return false, fmt.Errorf("value can't be negative %d", val)
    }
    return true, nil
}
```

Consistent function Returns - Better (imo)

```
func someFunction(val int) (ok bool, err error) {
    if val == 0 {
        return
    }
    if val < 0 {
        err = fmt.Errorf("value can't be negative %d", val)
        return
    }
    ok = true
    return
}
```

Denver Gophers

github.com/DenverGophers - All presentations and meetup materials (https://github.com/DenverGophers)

twitter.com/DenverGophers (https://twitter.com/DenverGophers)

Google Plus Denver Gophers Group (https://plus.google.com/u/0/communities/104822260820066412402)

Questions?

Thank you

Levi Cook
Cory LaNou

@levicook (http://twitter.com/levicook)
@corylanou (http://twitter.com/corylanou)