Error Wrapping in Go 1.13

Jean Niklas L’orange

vipps

October 9, 2019

First Go Service

- Database
- Cache (for performance)
- 3rd party services over HTTPS
- + more?

```
{"time":"2009-11-10T23:00:00Z", "level":"error"
"error":"unexpected EOF",
"CorrelationID":"54a23a54-6e3c-46ab-863f-c15937b85b13"}
```

```
{"time":"2009-11-10T23:00:00Z", "level":"error"
"error":"unexpected EOF",
"CorrelationID":"54a23a54-6e3c-46ab-863f-c15937b85b13"}
```

But where?

```go
if err != nil {
    return nil, err
}
```

```go
if err != nil {
  return nil, fmt.Errorf("3rd party issue: %s", err)
}
```

```
{"time":"2009-11-10T23:00:00Z", "level":"error"
"error":"3rd party issue:  unexpected EOF",
"CorrelationID":"54a23a54-6e3c-46ab-863f-c15937b85b13"}
```

Catching errors

- Want to find root causes like `ErrUnexpectedEOF` without `strings.Contains`

Catching errors

- Want to find root causes like `ErrUnexpectedEOF` without `strings.Contains`
- Solution (pre 1.13): `github.com/pkg/errors`

```go
import (
    ...
    pkgerrors "github.com/pkg/errors"
)

if err != nil {
     return nil, pkgerrors.Wrap(err, "3rd party issue")
}
```

```
switch err := pkgerrors.Cause(err).(type) {
  case *io.ErrUnexpectedEOF:
  // handle error
  default:
  // some other error
}
```

Issues with `pkg/errors`

- Can only catch root causes

Issues with `pkg/errors`

- Can only catch root causes
- Does not work with Go’s standard library

```
{"time":"2009-11-10T23:00:00Z", "level":"error"
"error":"Get http://thirdparty.com:  context deadline
exceeded",
"CorrelationID":"54a23a54-6e3c-46ab-863f-c15937b85b13"}
```

Cannot catch the underlying context deadline with `github.com/pkg/errors`!

Go 1.13 error wrapping

- Wrap:
  ```
  fmt.Errorf("3rd party issue: %w", err)
  ```

Go 1.13 error wrapping

- Wrap:
  `fmt.Errorf("3rd party issue: %w", err)`
- Is the root cause this specific error?
  `errors.Is(err, context.DeadlineExceeded)`

Go 1.13 error wrapping

- Wrap:
  `fmt.Errorf("3rd party issue: %w", err)`
- Is the root cause this specific error?
  `errors.Is(err, context.DeadlineExceeded)`
- Is the error of this type?

```go
var uerr *url.Error
if errors.As(err, &uerr) {
    if uerr.Timeout() {
      // ...
    }
}
```

See https://golang.org/pkg/errors for more information