Будущее ошибок в Go 2

Старт разработки Go 2 был анонсирован в июле 2017 года (на конец 2021 года разработка всё ещё продолжается).

В этом сугубо теоретическом модуле мы коснёмся новых фичей, связанных с обработкой ошибок, а по сути – кратко перескажем соответствующие draft design'ы (появившиеся через год после анонса, в августе 2018), расставляя необходимые акценты.

Для чего это нужно?

Это нужно для общего кругозора, для понимания эволюции работы с ошибками в Go, а также для знакомства с процессами, происходящими в сообществе Golang.

Не исключено, что когда-нибудь вас озарит и вы решите оставить своё предложение по работе с ошибками в языке.

Более того, мы увидим, что Go 2 – это не что-то абстрактное и далёкое, а нечто довольно осязаемое, и некоторые из представленных фич мы уже наблюдаем в нашей v1 гошке.

Прежде чем двигаться дальше, расставим точки над i, определив следующие понятия:

```
func foo() error {
    if err := bar(); err != nil {
```

```
        if err == io.EOF {
            // ...
        }
        return fmt.Errorf("do bar: %v", err)
    }
}

// ...

fmt.Printf("%+v", foo())
/*
do bar:
    some detail
    file.go:123
*/
```

Error Handling

Это весь блок, связанный с **обработкой ошибки**. Чаще всего сводится к `if ... err != nil` + `return err`:

```
if err := bar(); err != nil {
    // ...
    return fmt.Errorf("do bar: %v", err)

}
```

Error Inspection

Это непосредственно инструменты, с помощью которых мы проверяем ошибку: `==`, `.(type)`, `errors.Is` и т.д.:

```
if err == io.EOF {
    // ...

}
```

Error Printing

Это то, как наша ошибка (по сути цепочка ошибок) выводится на экран:

```
fmt.Printf("%+v", foo())
```

```
/*
do bar:
    some detail
    file.go:123

*/
```