

Beautiful IO

A tour through standard library `pkg/io` and various implementations of its interfaces.

Golab 2019, 2019–10–21, Florence

Martin Czygan

About me

SWE [@ubleipzig](#) working mostly with Python and Go.

Taming data – open source – writing.

| [Explore IO](#) workshop at Golab 2017.

Background

- Go Proverbs (2015)

| The bigger the interface, the weaker the abstraction.

Prominent examples are `io.Reader` and `io.Writer`.

The IO package

- contains basic, widely used interfaces (within and outside standard library)
- utility functions

Why beautiful?

La bellezza è negli occhi di chi guarda

- small, versatile interfaces
- composable

Love and praise

This article aims to convince you to use `io.Reader` in your own code wherever you can. -- [@matryer](#)

"Crossing Streams: a love letter to Go `io.Reader`" -- [@jmoiron](#)

Which brings me to `io.Reader`, easily my favourite Go interface. --
[@davecheney](#)

What's in pkg/io?

- 25 types
- 21/25 are interfaces
- 12 functions, 3 constants, 6 errors

The concrete types are: `LimitedReader`, `PipeReader`, `PipeWriter`, `SectionReader`; functions: `Copy`, `CopyN`, `CopyBuffer`, `Pipe`, `ReadAtLeast`, `ReadFull`, `WriteString`, `LimitReader`, `MultiReader`, `TeeReader`, `NewSectionReader`, `MultiWriter`

A few Interfaces

	R	W	C	S
io.Reader	x			
io.Writer		x		
io.Closer			x	
io.Seeker				x
io.ReadWriter	x	x		
io.ReadCloser	x		x	
io.ReadSeeker	x			x
io.WriteCloser		x	x	
io.WriteSeeker		x		x
io.ReadWriteCloser	x	x	x	
io.ReadWriteSeeker	x	x		x

Missing interfaces

<https://github.com/go4org/go4/blob/94abd6928b1da39b1d757b60c93fb2419c409>

```
... 33 // A ReadSeekCloser can Read, Seek, and Close.  
34 type ReadSeekCloser interface {  
35     io.Reader  
36     io.Seeker  
37     io.Closer  
38 }  
39  
40 type ReaderAtCloser interface {  
41     io.ReaderAt  
42     io.Closer  
43 }
```

How many readers, writers are there?

```
$ guru -json implements /usr/local/go/src/io/io.go:#3309
```

I counted over 280 implementations of `io.Reader` and X of `io.Writer`.

What is a Reader?

```
type Reader interface {  
    Read(p []byte) (n int, err error)  
}
```

- at most `len(p)` bytes are read
- to signal the end of a stream, return `io.EOF`
- `n` might or might not be zero, when `io.EOF` is returned

