করো: Translating Code to Other (Human) Languages and Back Again

Aditya Mukerjee

Systems Engineer, Stripe

@chimeracoder

```
47  int
48  Pconv(Fmt *fp)
49  {
50          char str[STRINGSZ];
51          Prog *p;
52          char scale[40];
53
54          p = va_arg(fp->args, Prog*);
55          sconsize = 8;
56          scale[0] = '\0';
57          if(p->from.scale != 0 && (p->as == AGLOBL || p->as == ATEXT))
58                  snprint(scale, sizeof scale, "%d,", p->from.scale);
59          switch(p->as) {
60          default:
```

Go 1.3+ Compiler Overhaul

golang.org/s/go13compiler

Russ Cox

December 2013

Abstract

The Go compiler today is written in C. It is time to move to Go.

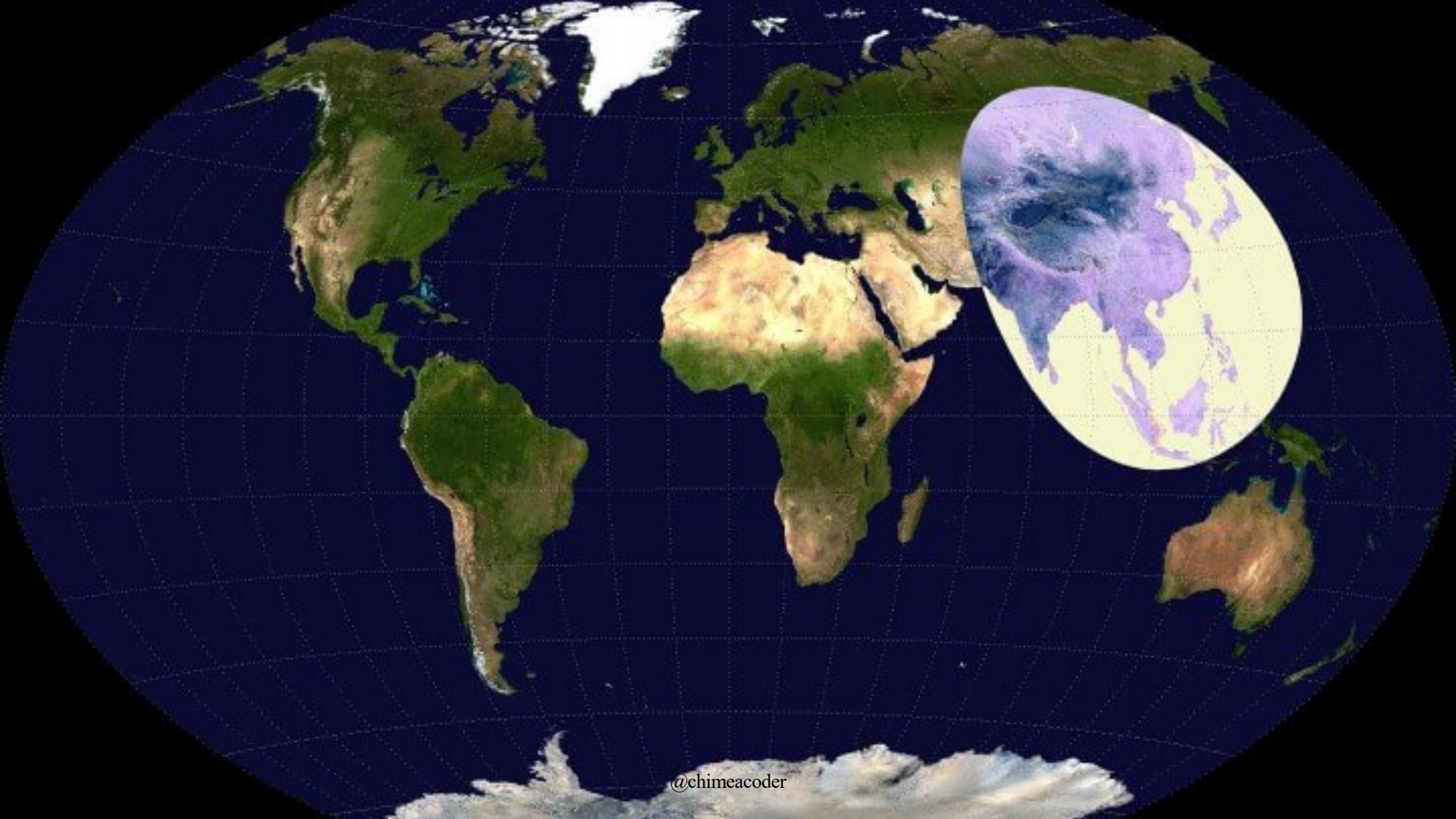
- It is easier to write correct Go code than to write correct C code.
- It is easier to debug incorrect Go code than to debug incorrect C code.
- Work on a Go compiler necessarily requires a good understanding of Go. Implementing the compiler in C adds an unnecessary second requirement.
- Go makes parallel execution trivial compared to C.
- Go has better standard support than C for modularity, for automated rewriting, for unit testing, and for profiling.
- Go is much more fun to use than C.

95%

of the world doesn’t speak English as their first language

89%

of the world doesn't speak English **at all**


@chimeacoder


POLICE
RICHMOND

Software reflects the **people** who build it

*“I would have thought it would be useful to NATO, because they had the common verbs for the things they were going to do. And the nouns, they’d just have to have a dictionary for things they were referring to for inventory control…. **They’d have common nouns throughout NATO, and they could make a dictionary of common verbs and translate the program. You could write one in English and you could translate it and it could go to [the other language]**. No problem, you’d have communication. It would be a limited vocabulary.”*

- Grace Hopper

build passing

করো

করো (koro) is an extension of the Go compiler and toolchain to support programming in Bengali (বাংলা).

You can read this README in English or in Bengali.

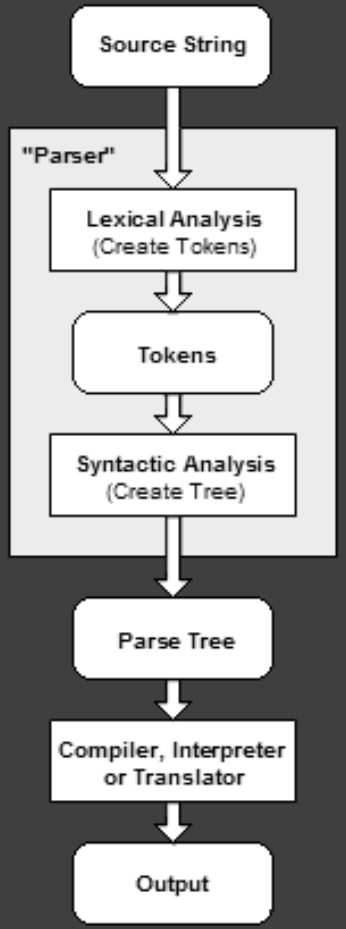
করো go কম্পাইলারের একটি এক্সটেনশন এবং বাংলা ভাষায় প্রোগ্রামিং করার টুলচেইন।

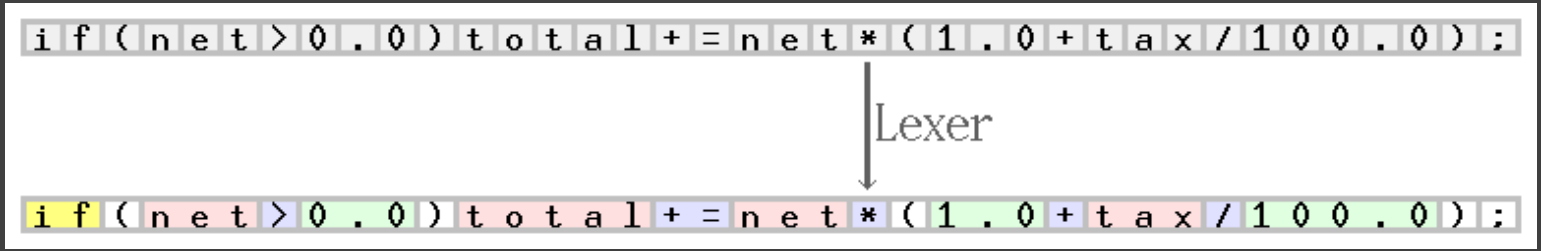
এই রিডমিটা ইংরেজি আর বাংলাতে পড়তে পারো।

github.com/ChimeraCoder/koro

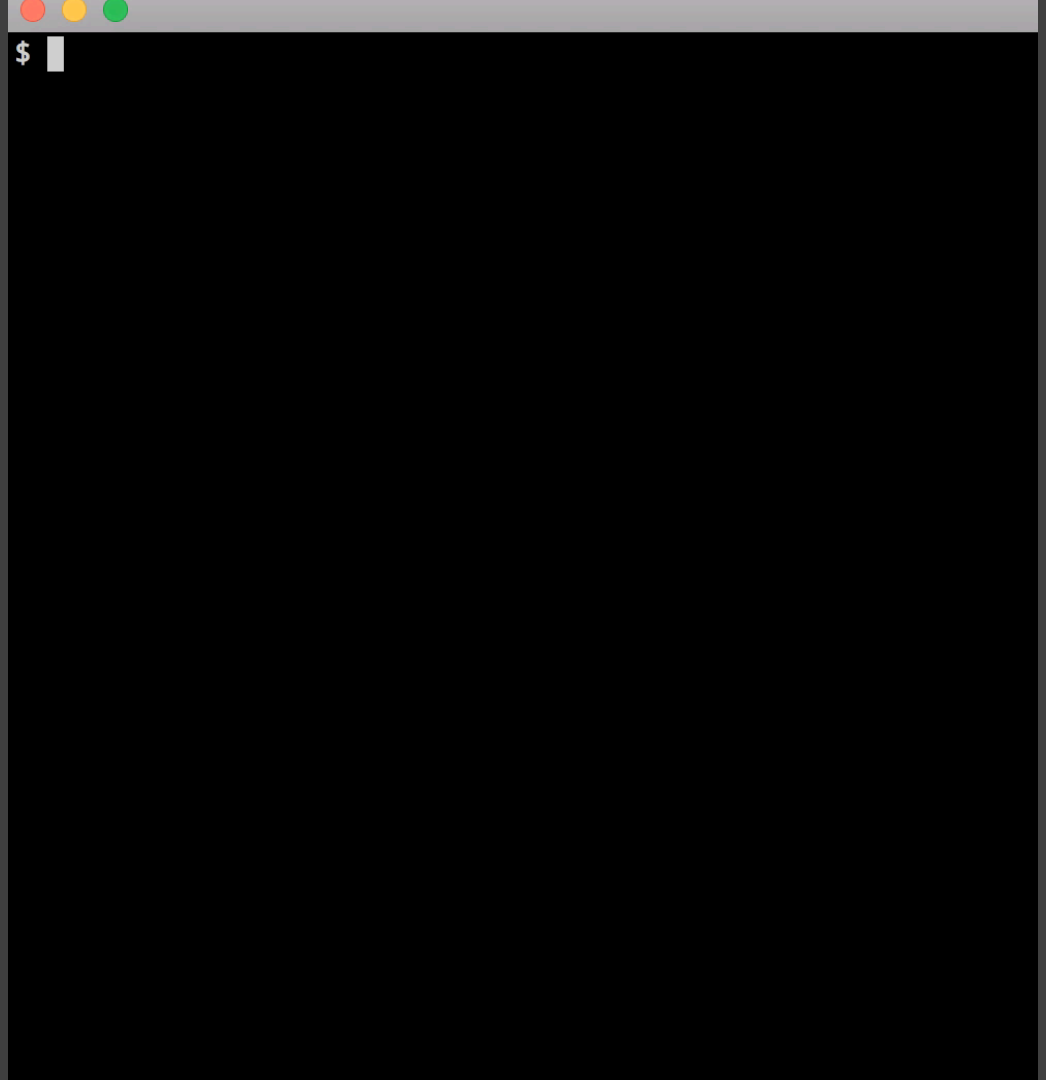
```
package main

import "fmt"

func main() {
        if true {
                fmt.Printf(“hello, world!\n")
        }
}
```

```
প্যাকেজ main

আমদানি "fmt"

ফ main(){
  যদি true {
    fmt.Println("ওহে বিশ্ব!\n")
  }
}
```


Source String
"Parser"
Lexical Analysis
(Create Tokens)
Tokens
Syntactic Analysis
(Create Tree)
Parse Tree
Compiler, Interpreter
or Translator
Output

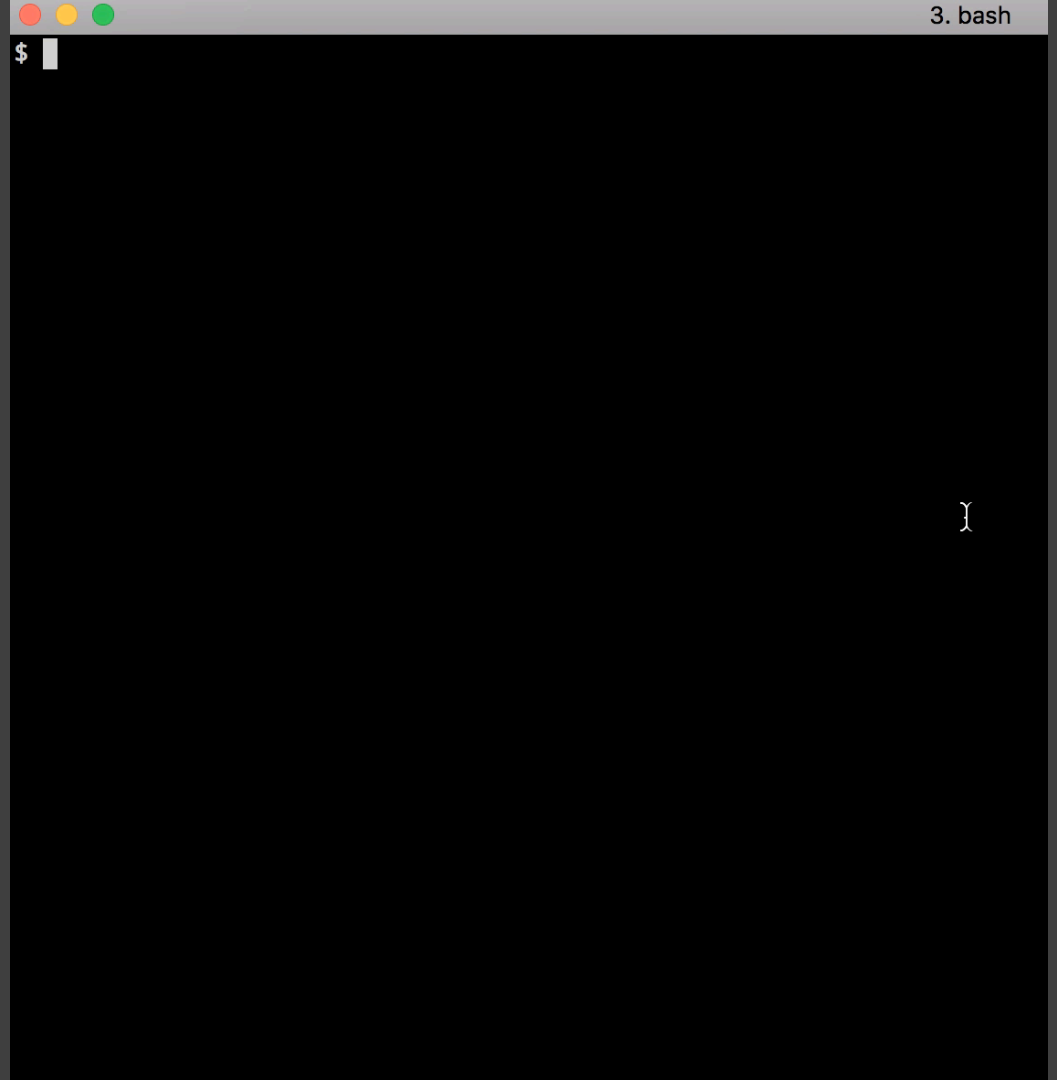

if(net>0.0)total+=net*(1.0+tax/100.0);
Lexer
if(net>0.0)total+=net*(1.0+tax/100.0);

```
+var bengaliKeywords = map[string]int32{
+       "ভাঙ্গা": LBREAK,
+       "ক্ষেত্রে": LCASE,
+       "চ্যানেল": LCHAN,
+       "ধ্রুবক": LCONST,
+       "চলো": LCONTINUE,
+       "ডিফল্ট": LDEFAULT,
+       "মুলতবি": LDEFER,
+       "অন্যভাবে": LELSE,
+       "নির্ঝর": LFALL,
+       "যখনই": LFOR,
+       "ফ":    LFUNC,
+       "কর": LGO,
```



gofmt

korofmt

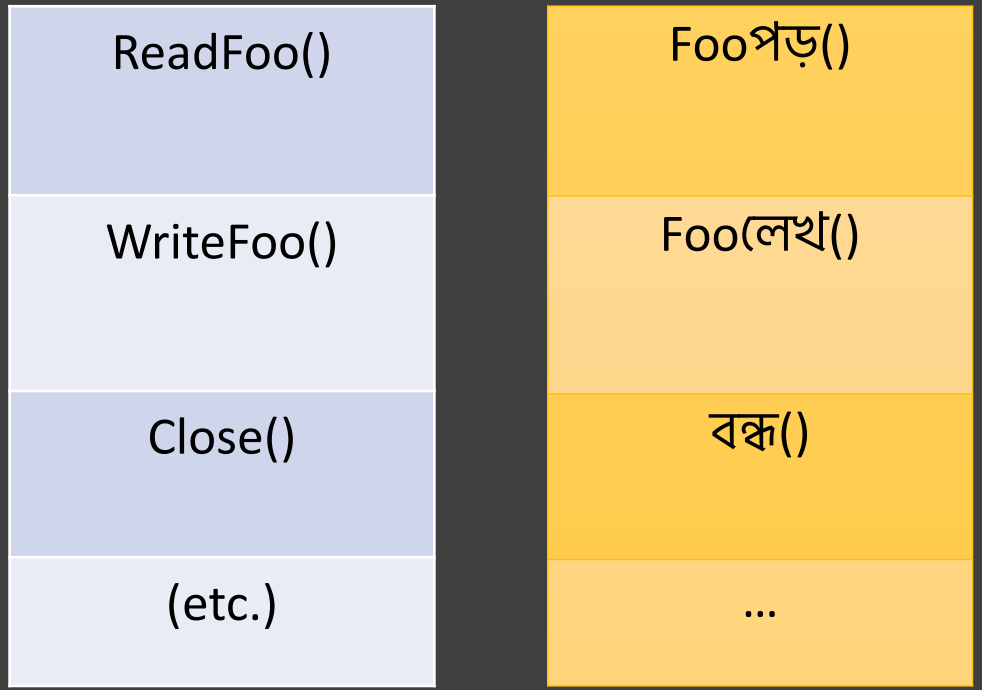

3. bash
$

Automatic source code translation

- **Bidirectional** translation layers
- Localize source code as a **commit hook**

Structuring Code

Naming Schemes



Localizing Error Messages

```
type error interface {
    Error() string
}
```

Cannot cast parameter type from type “System.String” to argument type “System.String”

The only thing worse than a cryptic error message is a cryptic error message **in a foreign language that you can't understand**

What about documentation?

The secret of automated translation….

- We don't have to bridge **all** communication between two arbitrary languages

- We just have to bridge communication **in a specific context** between two languages

Pipe creates a synchronous in-memory pipe. It can be used to connect code expecting an io.Reader with code expecting an io.Writer.

পাইপ একটি সিঙ্ক্রোনাস ইন-মেমরি পাইপ তৈরি করে। এটি একটি io.Reader কোড কোড একটি io.Writer আশা সঙ্গে কোড সংযোগ করতে ব্যবহার করা যেতে পারে।

Reads and Writes on the pipe are matched one to one except when multiple Reads are needed to consume a single Write. That is, each Write to the PipeWriter blocks until it has satisfied one or more Reads from the PipeReader that fully consume the written data. The data is copied directly from the Write to the corresponding Read (or Reads); there is no internal buffering.

পাইপের উপর লেখা এবং লিখনগুলি এক থেকে এক সাথে মেলানো হয় যখন একক লেখার জন্য একাধিক রিডগুলির প্রয়োজন হয়। যে, প্রত্যেকটি পাইপ ওয়ারিটার ব্লকগুলিতে লিখুন যতক্ষন না লিখিত ডেটা সম্পূর্ণভাবে ব্যবহার করে PipeReader থেকে এক বা একাধিক রিড সন্তুষ্ট হয়। তথ্য সংশ্লিষ্ট পাঠ্য (বা পাঠ) থেকে লিখুন থেকে সরাসরি কপি করা হয়; কোন অভ্যন্তরীণ বাফার আছে।

It is safe to call Read and Write in parallel with each other or with Close. Parallel calls to Read and parallel calls to Write are also safe: the individual calls will be gated sequentially.

এটা একে অপরের সঙ্গে বা বন্ধ সঙ্গে সমান্তরাল পড়ুন পড়ুন এবং লিখুন নিরাপদ। লেখার জন্য সমান্তরাল কলগুলি পড়ুন এবং সমান্তরাল কলগুলিও নিরাপদ: ব্যক্তিগত কলগুলি ক্রমানুসারে গেট হয়ে যাবে।

Pipe creates a synchronous in-memory pipe. It can be used to connect code expecting an io.Reader with code expecting an io.Writer.

Reads and Writes on the pipe are matched one to one except when multiple Reads are needed to consume a single Write. That is, each Write to the PipeWriter blocks until it has satisfied one or more Reads from the PipeReader that fully consume the written data. The data is copied directly from the Write to the corresponding Read (or Reads); there is no internal buffering.

It is safe to call Read and Write in parallel with each other or with Close. Parallel calls to Read and parallel calls to Write are also safe: the individual calls will be gated sequentially.

The pipe creates a synchronous in-memory pipe. It can be used to connect the code with an io.Writer Expect an io.Reader code code.

Writing and writing on pipe matches one to one when multiple reads are required for single writing. Write down each pipe warrior block until one or more reads from PipeReader are satisfied using the written data completely. The information is copied directly from the corresponding text (or text); There are no internal buffers.

Read it to read parallel with each other or close and safe to enter. Read parallel calls for writing and parallel calls are also safe: Private calls will be gated in sequence.

Pipe creates a synchronous in-memory pipe. It can be used to connect code expecting an io.Reader with code expecting an io.Writer.

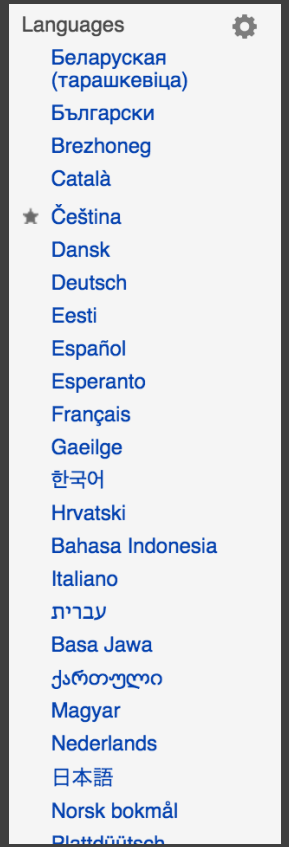
Reads and Writes on the pipe are matched one to one except when multiple Reads are needed to consume a single Write. That is, each Write to the PipeWriter blocks until it has satisfied one or more Reads from the PipeReader that fully consume the written data. The data is copied directly from the Write to the corresponding Read (or Reads); there is no internal buffering.

It is safe to call Read and Write in parallel with each other or with Close. Parallel calls to Read and parallel calls to Write are also safe: the individual calls will be gated sequentially.

Piping creates a synchronous pipeline in memory. It is a discount code io.Reader code to connect with a io.Writer hope can be used.

Writing and writing in the pipeline when the units correspond one by one with the need to write more than one ridagulira. That is, each tube will write blocks until the data using the one or more read PipeReader is satisfied. Relevant text information (or text) is copied directly from the site; There is no internal buffer.

It is read in parallel with each other or with the read and write insurance. Reading calls to write parallel and parallel calls is safe: the door will be in the order of personal calls.



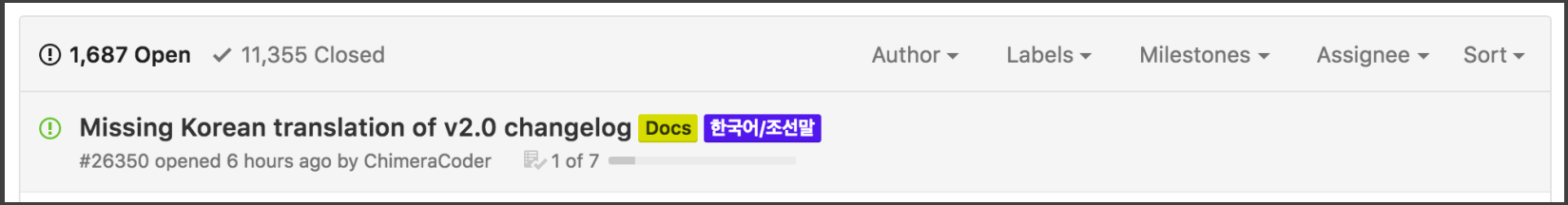
@chimeracoder

Automatic translation is not a substitute - it's an **invitation**

Growing a multilingual OSS community doesn't happen overnight

(but it doesn't happen automatically either)

It’s surprisingly easy to get community translations…
but you do have to **ask**



Localizing Live Events

We need to bring **events to the global community**

…but we also need to bring **the global community to our events**

To truly eliminate linguistic barriers, **both** are necessary

Localizing Project Communication

Encourage people to **write in their native languages**

Provide **archive links** with community translations

Where do we go from here?

95%

of the world doesn’t speak English as their first language

89%

of the world doesn't speak English **at all**

Aditya Mukerjee
@chimeracoder
https://github.com/ChimeraCoder