

A journey in software development.

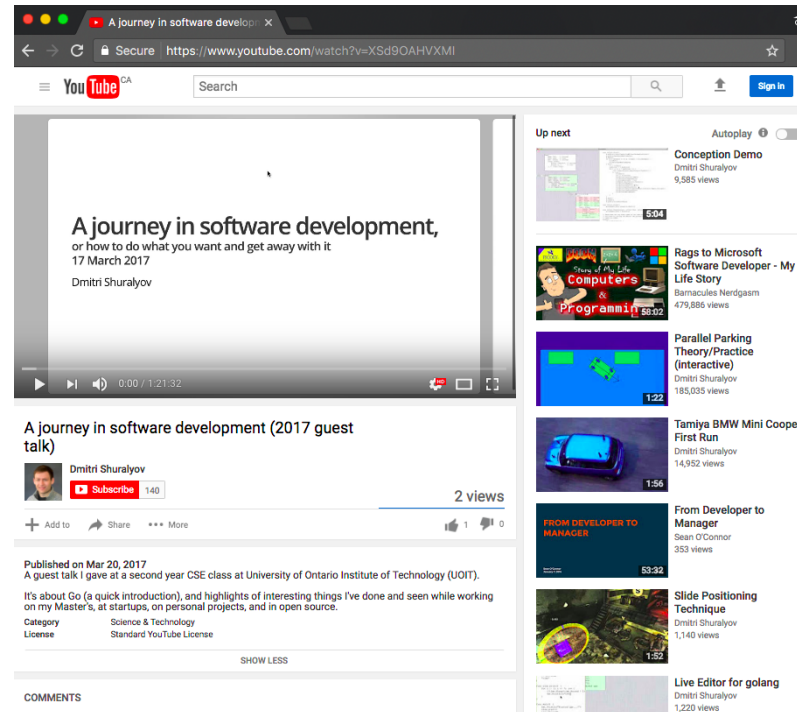
Or, how to do what you want and get away with it.

17 March 2017

Dmitri Shuralyov

Video

A video of this talk was recorded at University of Ontario Institute of Technology (UOIT) on March 17, 2017.



Video: www.youtube.com/watch?v=XSd9OAHVXMI (<https://www.youtube.com/watch?v=XSd9OAHVXMI>)

Programming languages

Which ones have you used? Heard of?

Which ones do you like?

What about frontend?

Programming languages

Preface:

- Try many languages, keep an open mind.
- Don't be afraid to use what you like.
- Go is somewhat unique. Knowing it can give you an edge.

Go

Go

You may have heard of Go.

golang.org (<https://golang.org>)

It's my favorite language. I think you'll like it, too.

What is Go?

An open source (BSD licensed) project:

- Language specification,
- Small runtime (garbage collector, scheduler, etc.),
- Two compilers (gc and gccgo),
- 'Batteries included' standard library,
- Tools (build, fetch, test, document, profile, format),
- Documentation.

Go is about composition

Go is Object Oriented, but not in the usual way.

- no classes (methods may be declared on any type)
- no subtype inheritance
- interfaces are satisfied implicitly (structural typing)

The result: simple pieces connected by small interfaces.

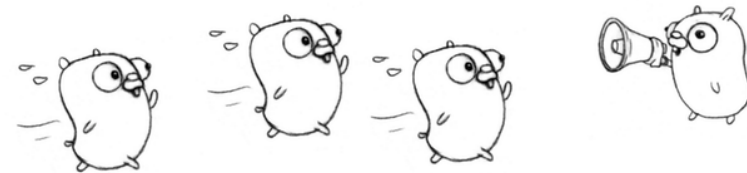
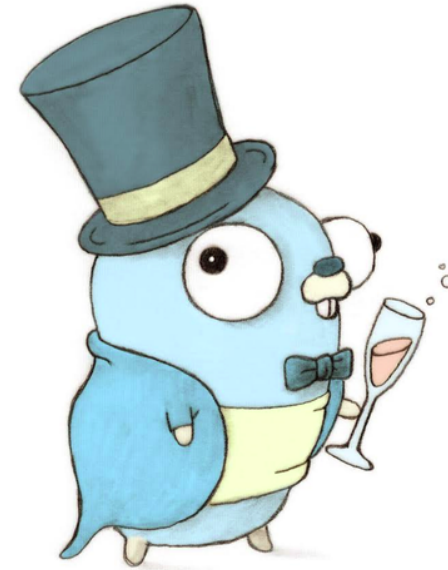
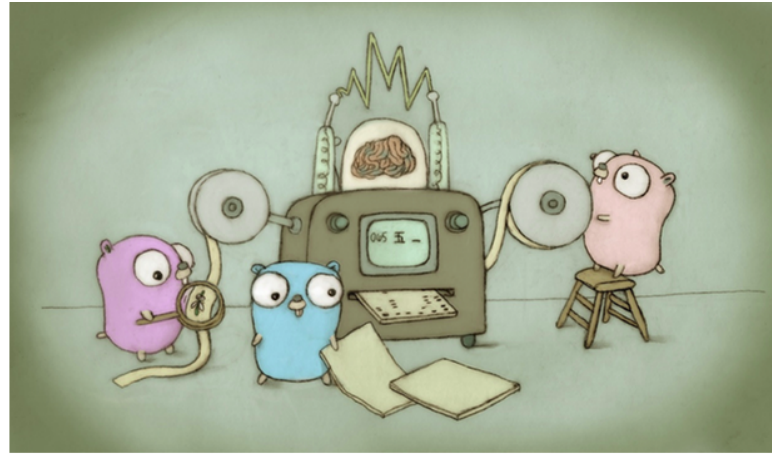
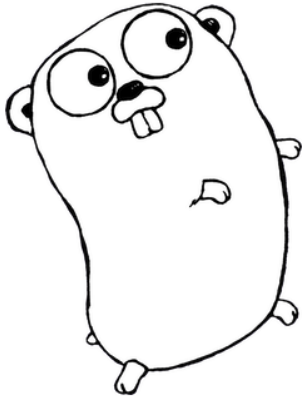
Go is about concurrency

Go provides CSP-like concurrency primitives.

- lightweight threads (goroutines)
- typed thread-safe communication and synchronization (channels)

The result: comprehensible concurrent code.

Go is about gophers



Go gopher by Renée French.

Core values

Go is about composition, concurrency, and gophers.

Keep that in mind.

Hello, Go

```
package main

import "fmt"

func main() {
    fmt.Println("Hello, Go.")
}
```

Hello, web

```
package main

import (
    "fmt"
    "log"
    "net/http"
)

const listenAddr = "localhost:4000"

func main() {
    http.HandleFunc("/", handler)
    err := http.ListenAndServe(listenAddr, nil)
    if err != nil {
        log.Fatal(err)
    }
}

func handler(w http.ResponseWriter, r *http.Request) {
    fmt.Fprint(w, "Hello, web.")
}
```

Hello, net

```
package main

import (
    "fmt"
    "log"
    "net"
)

const listenAddr = "localhost:4000"

func main() {
    l, err := net.Listen("tcp", listenAddr)
    if err != nil {
        log.Fatal(err)
    }
    for {
        c, err := l.Accept()
        if err != nil {
            log.Fatal(err)
        }
        fmt.Fprintln(c, "Hello!")
        c.Close()
    }
}
```

Interfaces

Hey neato! We just used `Fprintln` to write to a net connection.

That's because a `Fprintln` writes to an `io.Writer`, and `net.Conn` is an `io.Writer`.

```
fmt.Fprintln(c, "Hello!")
```

```
func Fprintln(w io.Writer, a ...interface{}) (n int, err error)
```

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

```
type Conn interface {  
    Read(b []byte) (n int, err error)  
    Write(b []byte) (n int, err error)  
    Close() error  
    // ... some additional methods omitted ...  
}
```

An echo server

```
package main

import (
    "io"
    "log"
    "net"
)

const listenAddr = "localhost:4000"

func main() {
    l, err := net.Listen("tcp", listenAddr)
    if err != nil {
        log.Fatal(err)
    }
    for {
        c, err := l.Accept()
        if err != nil {
            log.Fatal(err)
        }
        io.Copy(c, c)
    }
}
```

Goroutines

Goroutines are lightweight threads that are managed by the Go runtime. To run a function in a new goroutine, just put "go" before the function call.

```
package main

import (
    "fmt"
    "time"
)

func main() {
    go say("let's go!", 3)
    go say("ho!", 2)
    go say("hey!", 1)
    time.Sleep(4 * time.Second)
}

func say(text string, secs int) {
    time.Sleep(time.Duration(secs) * time.Second)
    fmt.Println(text)
}
```

A concurrent echo server

```
package main

import (
    "io"
    "log"
    "net"
)

const listenAddr = "localhost:4000"

func main() {
    l, err := net.Listen("tcp", listenAddr)
    if err != nil {
        log.Fatal(err)
    }
    for {
        c, err := l.Accept()
        if err != nil {
            log.Fatal(err)
        }
        go io.Copy(c, c)
    }
}
```

"Chat roulette"

Let's look at a simple program, based on the "chat roulette" site.

In short:

- a user connects,
- another user connects,
- everything one user types is sent to the other.

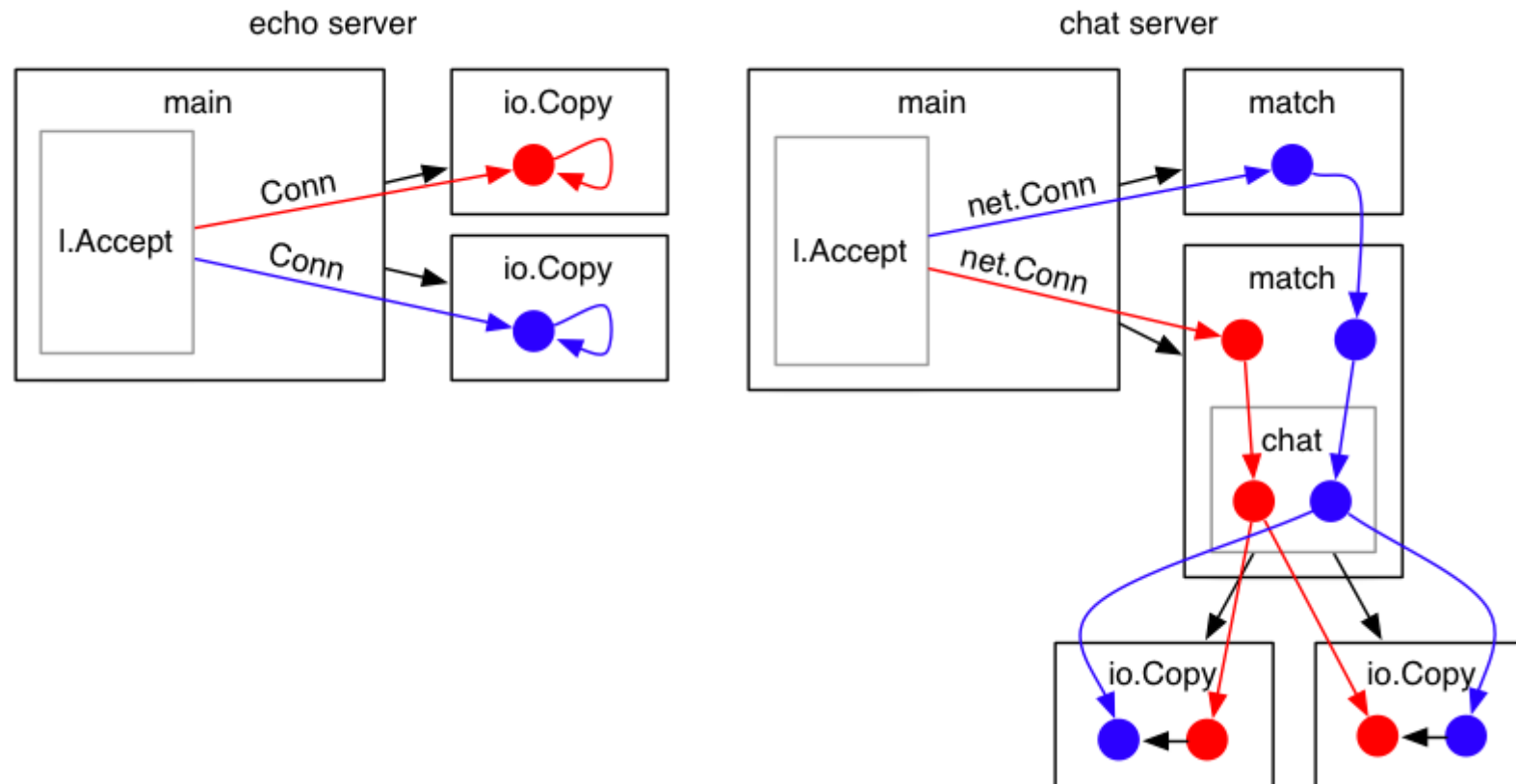
Design

The chat program is similar to the echo program. With echo, we copy a connection's incoming data back to the same connection.

For chat, we must copy the incoming data from one user's connection to another's.

Copying the data is easy. The hard part is matching one partner with another.

Design diagram



Channels

Goroutines communicate via channels. A channel is a typed conduit that may be synchronous (unbuffered) or asynchronous (buffered).

```
package main

import "fmt"

func main() {
    ch := make(chan int)
    go fibs(ch)
    for i := 0; i < 20; i++ {
        fmt.Println(<-ch)
    }
}

func fibs(ch chan int) {
    i, j := 0, 1
    for {
        ch <- j
        i, j = j, i+j
    }
}
```

Select

A select statement is like a switch, but it selects over channel operations (and chooses exactly one of them).

```
package main

import (
    "fmt"
    "time"
)

func main() {
    ticker := time.NewTicker(time.Millisecond * 250)
    boom := time.After(time.Second * 1)
    for {
        select {
        case <-ticker.C:
            fmt.Println("tick")
        case <-boom:
            fmt.Println("boom!")
            return
        }
    }
}
```

Modifying echo to create chat

In the accept loop, we replace the call to `io.Copy`:

```
for {  
    c, err := l.Accept()  
    if err != nil {  
        log.Fatal(err)  
    }  
    go io.Copy(c, c)  
}
```

with a call to a new function, `match`:

```
for {  
    c, err := l.Accept()  
    if err != nil {  
        log.Fatal(err)  
    }  
    go match(c)  
}
```

The matcher

The match function simultaneously tries to send and receive a connection on a channel.

- If the send succeeds, the connection has been handed off to another goroutine, so the function exits and the goroutine shuts down.
- If the receive succeeds, a connection has been received from another goroutine. The current goroutine then has two connections, so it starts a chat session between them.

```
var partner = make(chan io.ReadWriteCloser)

func match(c io.ReadWriteCloser) {
    fmt.Fprint(c, "Waiting for a partner...")
    select {
    case partner <- c:
        // now handled by the other goroutine
    case p := <-partner:
        chat(p, c)
    }
}
```

The conversation

The chat function sends a greeting to each connection and then copies data from one to the other, and vice versa.

Notice that it launches another goroutine so that the copy operations may happen concurrently.

```
func chat(a, b io.ReadWriter) {  
    fmt.Fprintln(a, "Found one! Say hi.")  
    fmt.Fprintln(b, "Found one! Say hi.")  
    go io.Copy(a, b)  
    io.Copy(b, a)  
}
```

Demo

Error handling

It's important to clean up when the conversation is over. To do this we send the error value from each `io.Copy` call to a channel, log any non-nil errors, and close both connections.

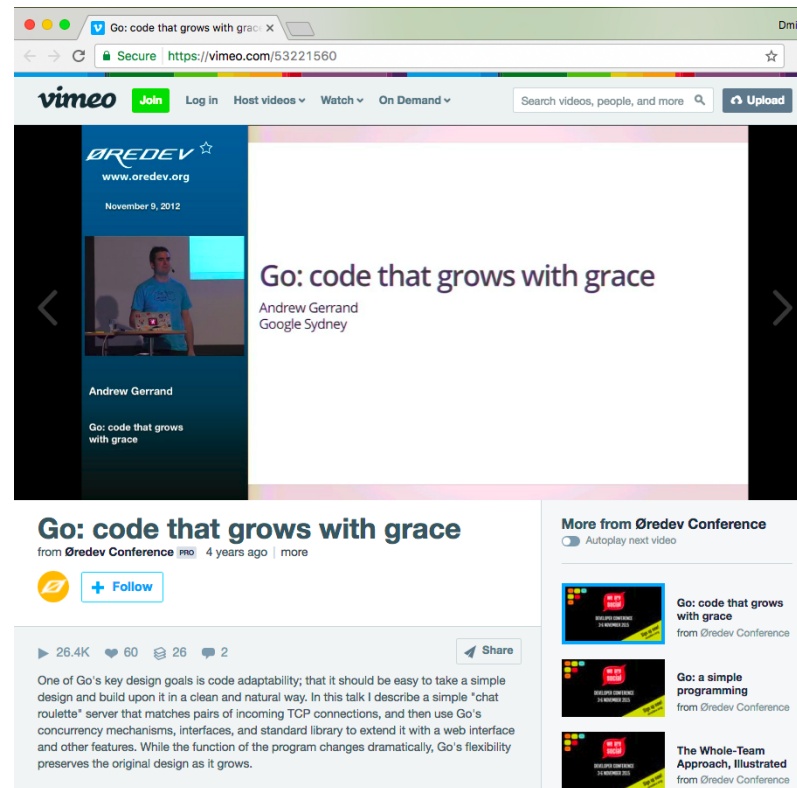
```
func chat(a, b io.ReadWriteCloser) {
    fmt.Fprintln(a, "Found one! Say hi.")
    fmt.Fprintln(b, "Found one! Say hi.")
    errc := make(chan error, 1)
    go cp(a, b, errc)
    go cp(b, a, errc)
    if err := <-errc; err != nil {
        log.Println(err)
    }
    a.Close()
    b.Close()
}
```

```
func cp(w io.Writer, r io.Reader, errc chan<- error) {
    _, err := io.Copy(w, r)
    errc <- err
}
```

Demo

Next steps

Want to see WebSockets, markov chains, learning bots, TCP and HTTP at once?



Video: vimeo.com/53221560 (<https://vimeo.com/53221560>)

Slides: talks.golang.org/2012/chat.slide (<https://talks.golang.org/2012/chat.slide>)

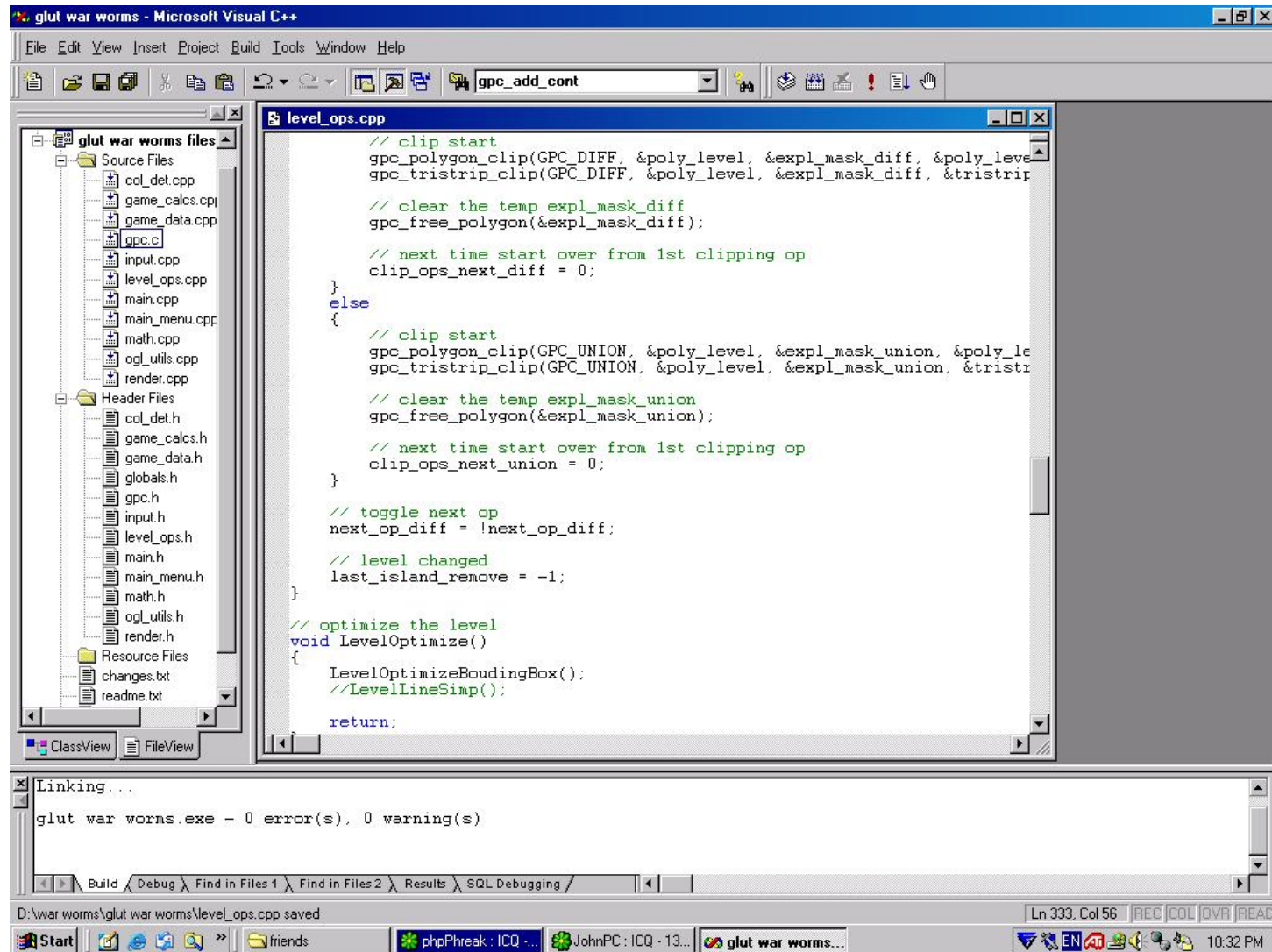
Journey

Journey

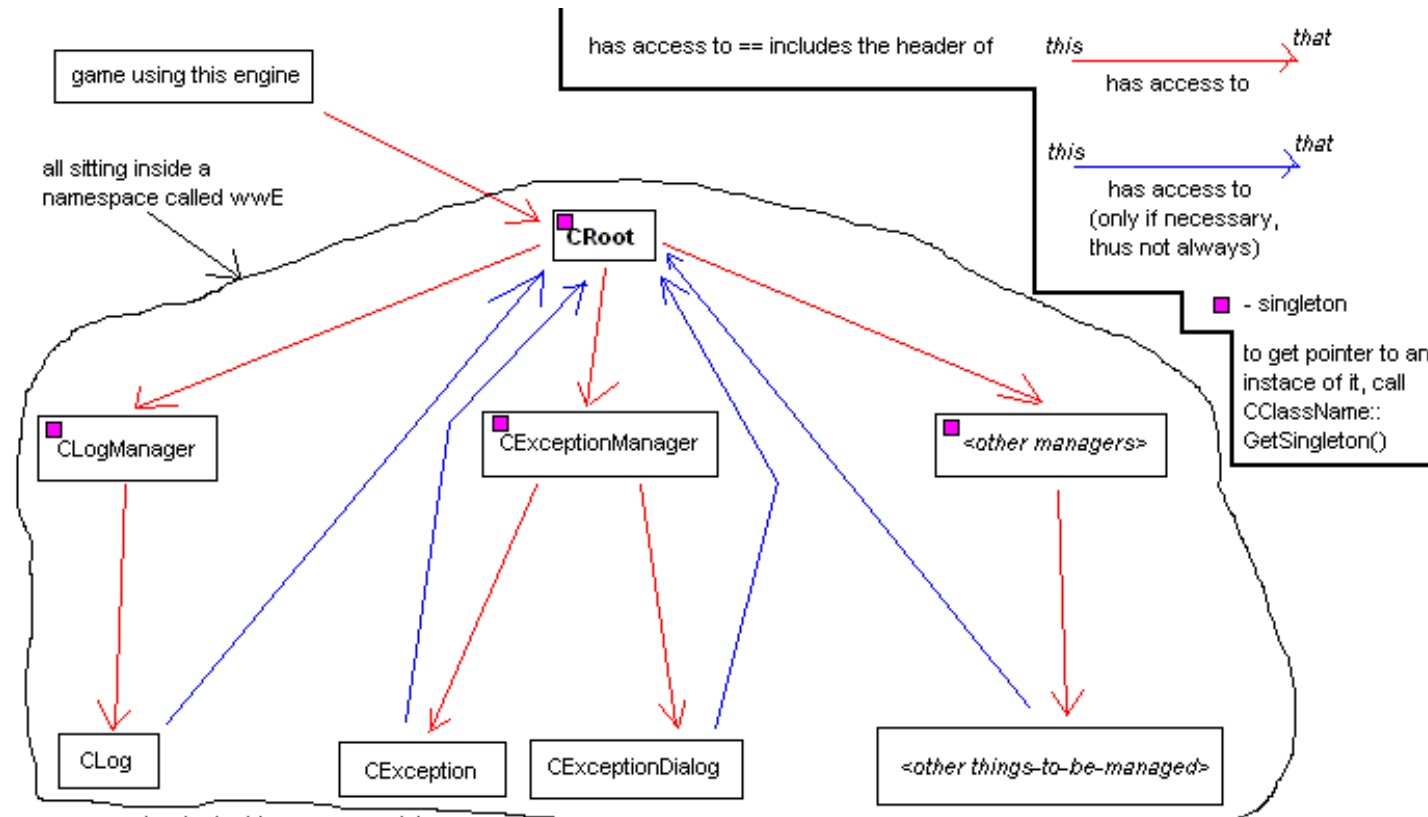
Started programming at 14.

Excited by the ability to create video games.

Journey



Journey

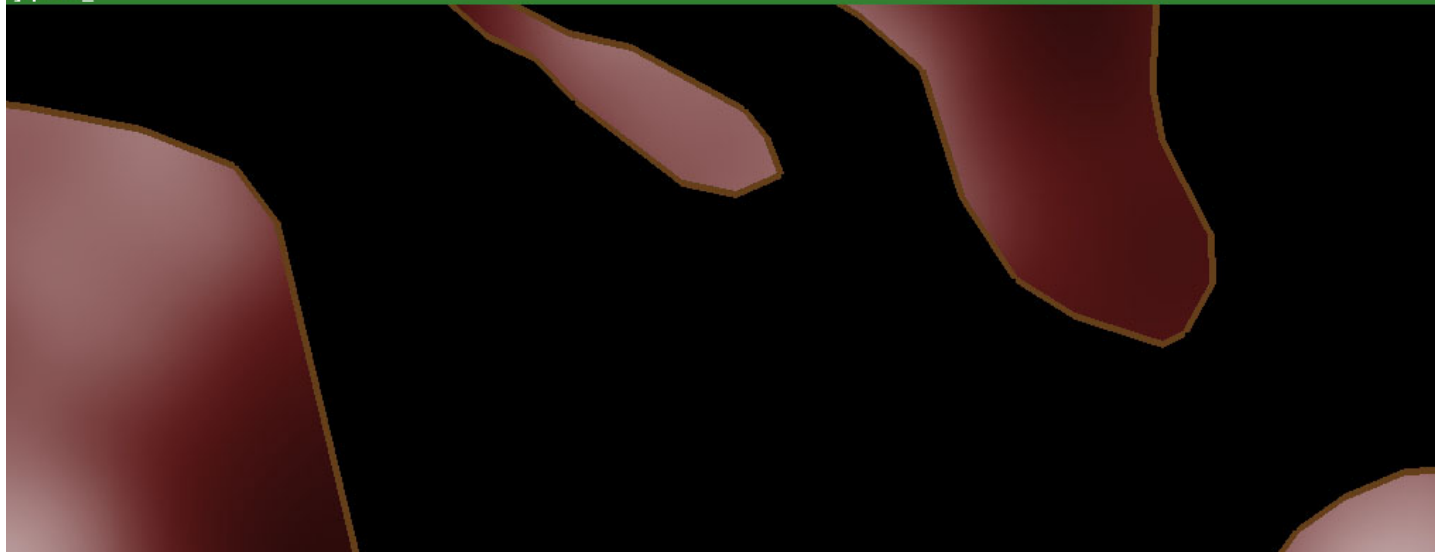


an example of why blue arrows exist:

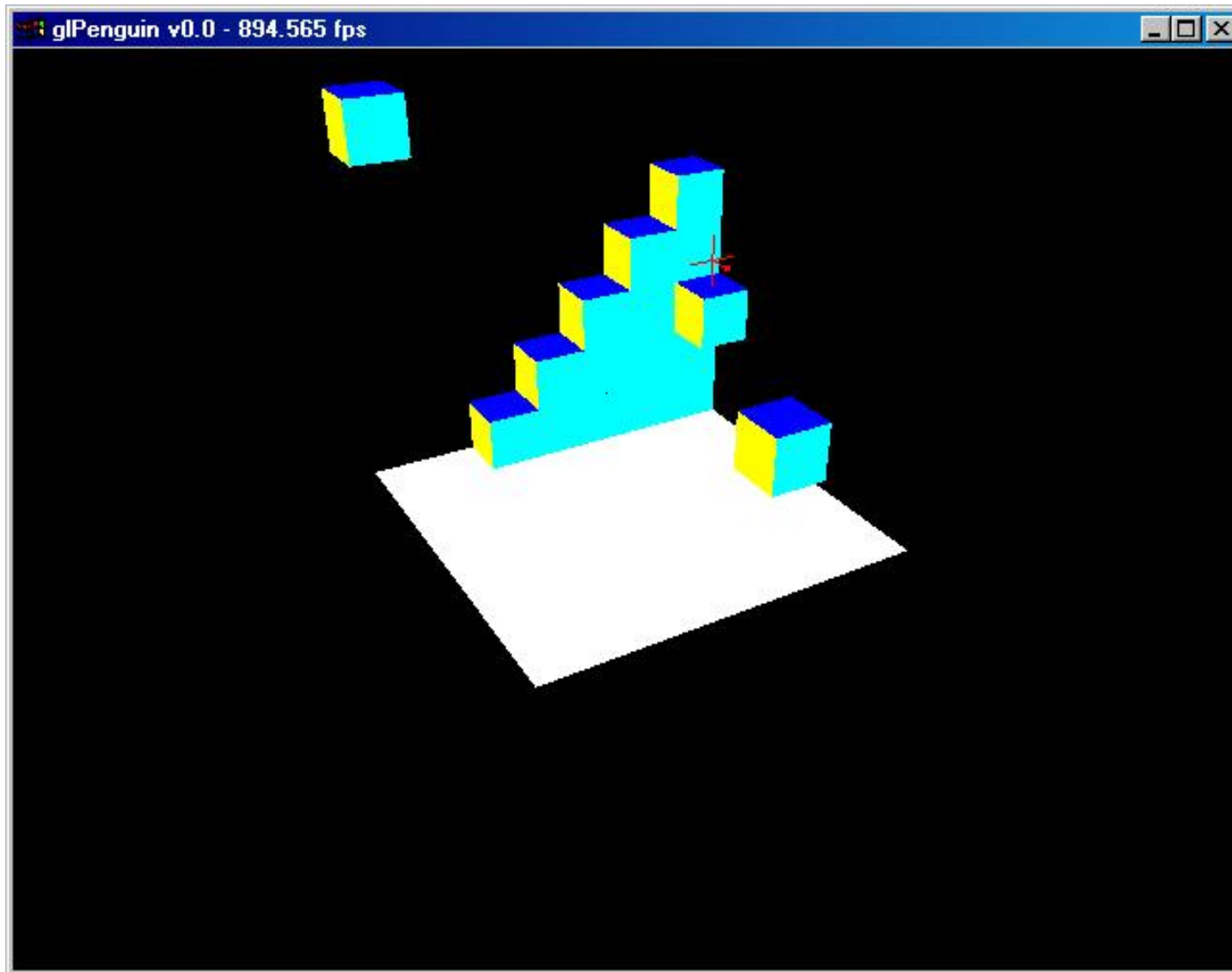
CException, so far, is the only class that actually does include CRoot header. that gives it access to CLogManager, and thus lets it log the description of the exception (i throw the CException class when an exception occurs, so its ctor logs about it).

Journey

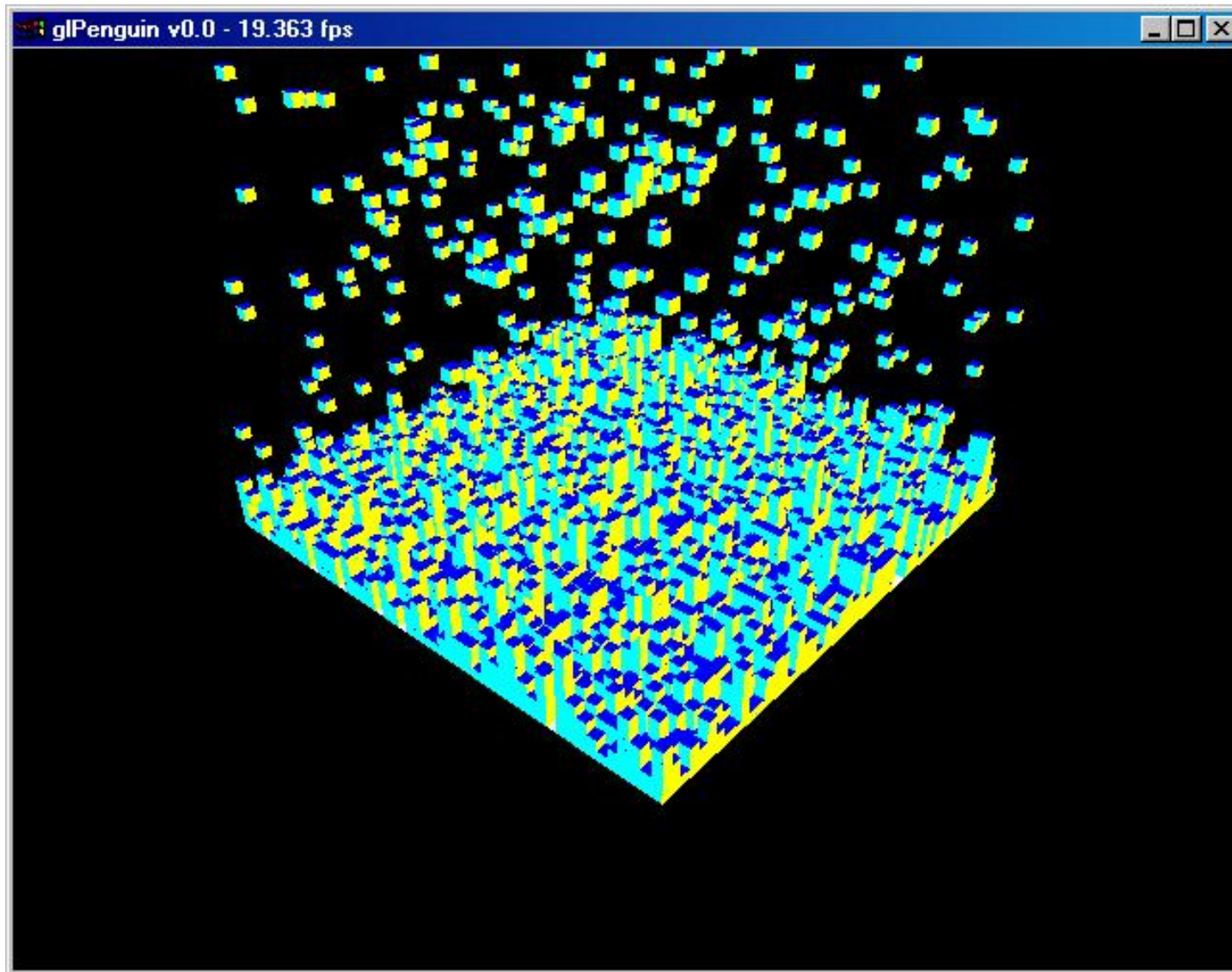
```
type 'cmdlist' to list all commands
type 'help cmdname' to get help on that command
]cmd_list
12 commands in total:
quit
version
help
cls
cmd_list
echo
disconnect
about
set_res
set_bpp
num_params
get_nth_param
]version
war worms build 0017
]echo please note that the ground texture shown below is just a temp one ;)
please note that the ground texture shown below is just a temp one ;)
]quit_
```



Journey



Journey



Journey

- wormFX
- eX0 (C++)

Journey

Moved towards C++ because it's fast, cross-platform, compiles native binaries.

Journey

Master's degree.

- Slide demo

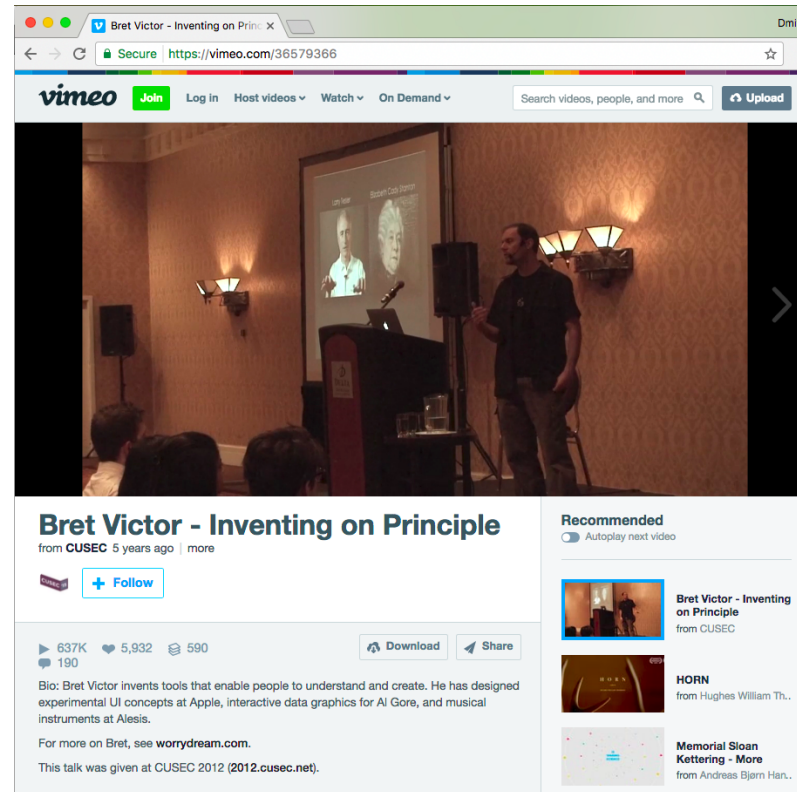
Journey

I realized I wanted to create tools that help people (more directly than games).

Also grew frustrated with C++, extremely motivated to do something about it.

Journey

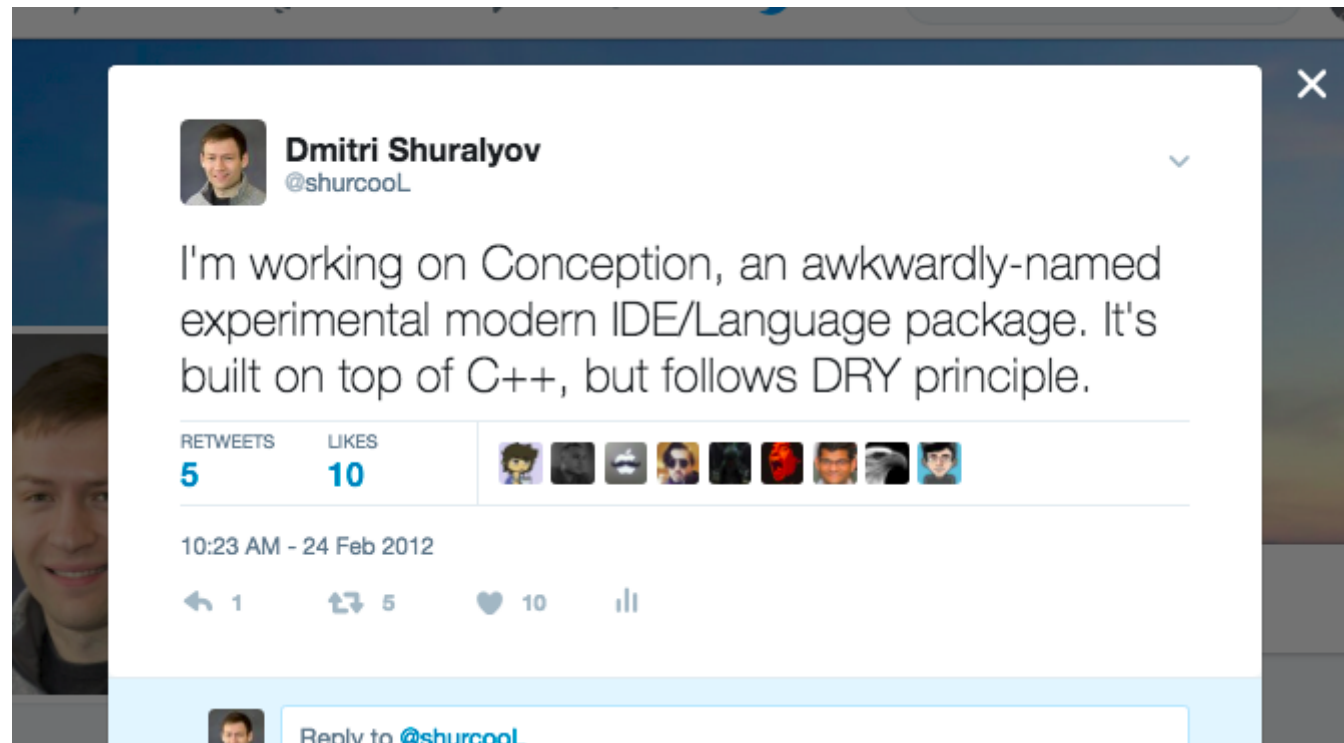
Bret Victor gave an incredible talk on following a *guiding principle*.



Video: vimeo.com/36579366 (<https://vimeo.com/36579366>)

Journey

Started to work on an experimental project.



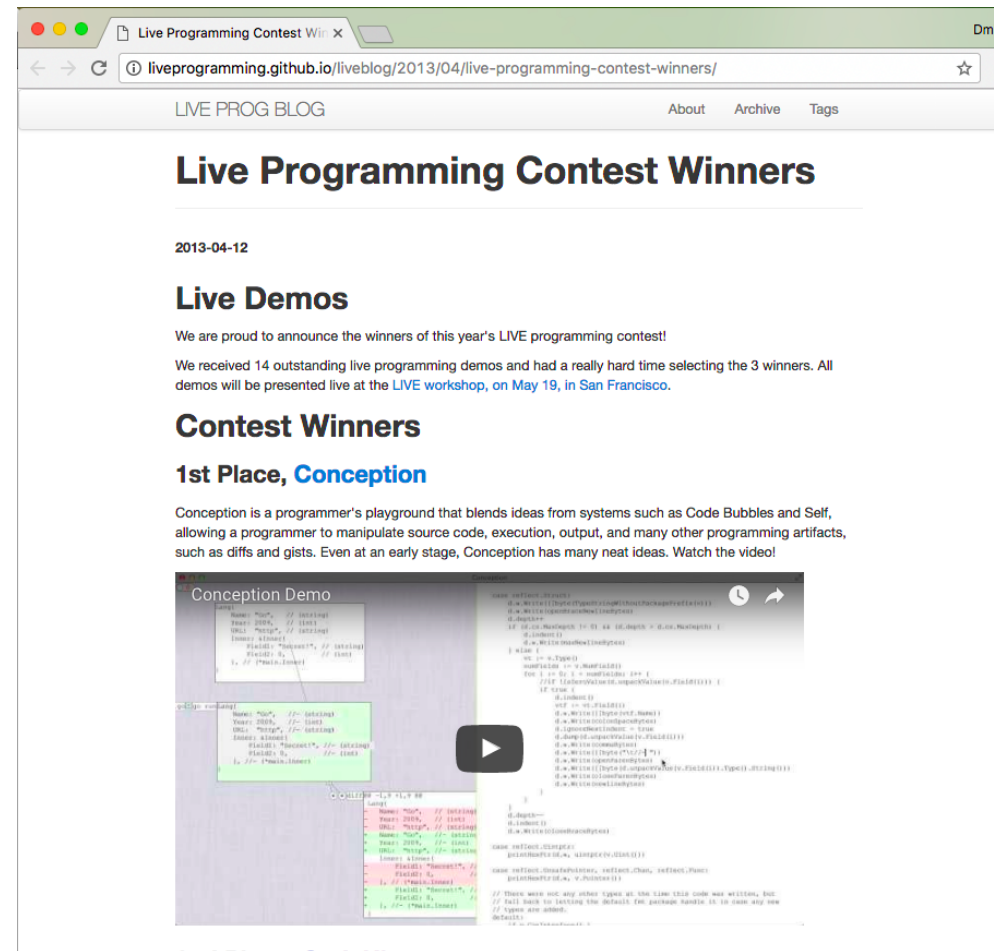
twitter.com/shurcool/status/173110768726839296 (<https://twitter.com/shurcool/status/173110768726839296>)

Journey

- [Conception demo video](https://github.com/shurcool/Conception#demonstration) (<https://github.com/shurcool/Conception#demonstration>)

Journey

Submitted it to a Live Programming Contest at LIVE 2013 Workshop. Won 1st place.



Journey

Got to visit San Francisco.



Journey



Journey



Journey

It helped me land my first job at a startup in San Francisco.



Startup life

- Surrounding yourself with better people is the best way to learn.
- Value an engineer brings vs company investing into them.
- First production program feeling.
- Diagrams and whiteboards are used during high-level discussions, planning.
- Hiring is really hard.
- 1 of 10 startups have successful outcome, luck plays a huge role.

Startup life

- Free food and perks, but you work hard.
- San Francisco has great weather, tons of events, meetups, great food, but expensive rent and homeless problem.

Hiring is hard

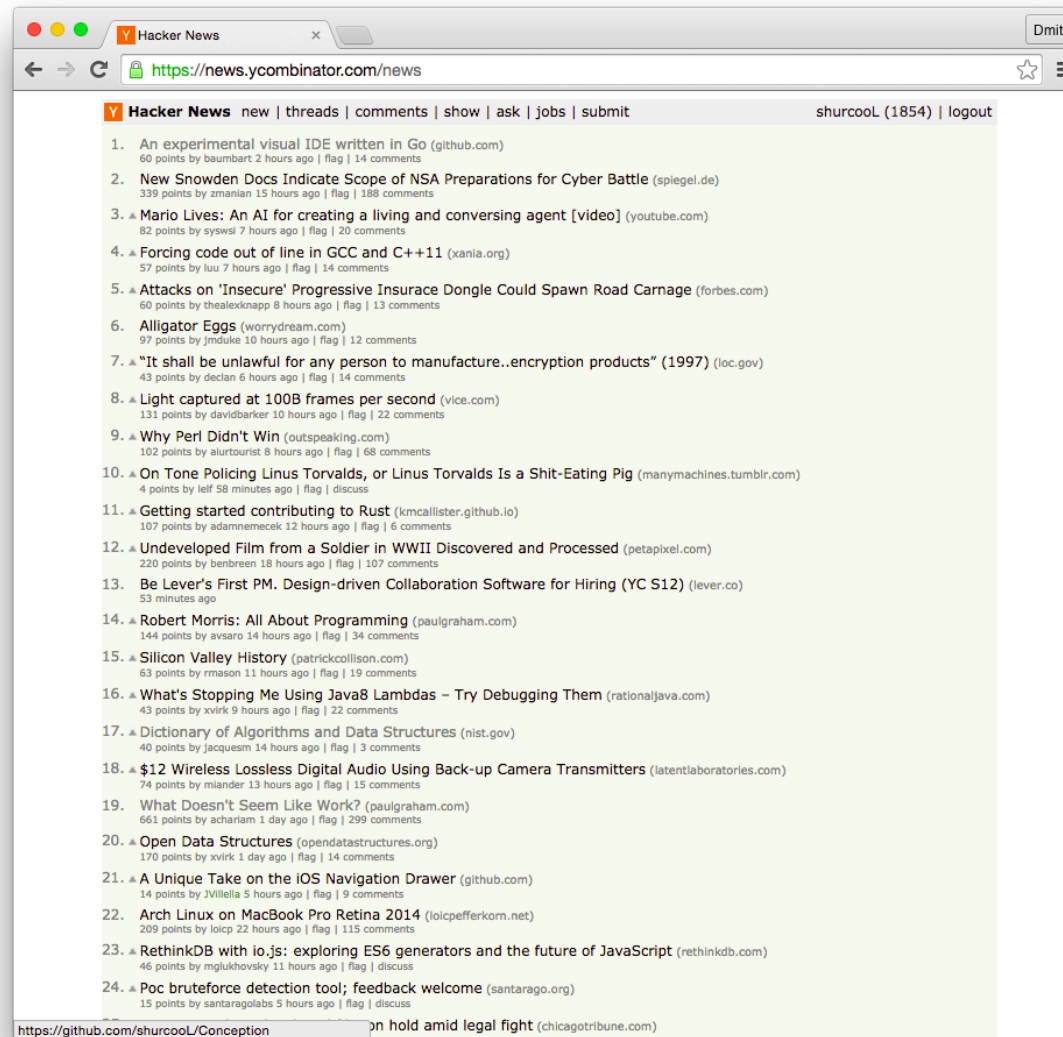
Startups with lots of raised money are bottlenecked on people.

A real-life story about a star.

Hiring is hard



Hiring is hard



Hiring is hard

The screenshot shows a web browser window displaying the GitHub repository page for 'shurcool / Conception'. The browser's address bar shows the URL 'https://github.com/shurcool/Conception/stargazers/you_know'. The repository page header includes the repository name, a search bar, and navigation links for 'Pull requests', 'Issues', and 'Gist'. Below the header, the repository name 'shurcool / Conception' is displayed, along with statistics: 50 Unwatch, 542 Stars, and 30 Forks. The 'Stargazers' tab is selected, showing a list of users who have starred the repository. The list includes: Ian Chiles (@fromAtoB), Juan Benet (Protocol Labs), Kamil Kisiel (Vancouver, British Columbia, Ca...), Dan Buch (@travis-ci), Michal Bohuslavěk (Joined on Sep 11, 2012), Minux Ma (中国), rt (Berkeley, CA), and Quinn Slack (@Sourcegraph). A red arrow points to the user 'rt'.

Stargazer	Username	Location
Ian Chiles	@fromAtoB	
Juan Benet	Protocol Labs	
Kamil Kisiel		Vancouver, British Columbia, Ca...
Dan Buch	@travis-ci	
Michal Bohuslavěk		Joined on Sep 11, 2012
Minux Ma		中国
rt		Berkeley, CA
Quinn Slack	@Sourcegraph	

Hiring is hard

JoanneHuang (Joanne Huang) x

GitHub, Inc. [US] <https://github.com/JoanneHuang>

Search GitHub Pull requests Issues Gist

Overview Repositories 8 Stars 0 Followers 3 Following 21

Joanne Huang
JoanneHuang

Follow

Block or report user

Triggitt
San Francisco
Joanne@triggitt.com
<http://triggitt.com/careers>

Popular repositories

- RubyTuesdays-ConnectFour**
Forked from [acanyon/RubyTuesdays-ConnectFour](#)
Learn Rails through ConnectFour. A Women Who Code SF project. womenwhocode.com
Ruby
- shareabouts-littlelibraries**
Forked from [HackingMadison/shareabouts-littlelibraries](#)
A Shareabouts instance for Madison's Little Library is a mapping application for crowdsourced info gathering.
JavaScript
- Spoon-Knife**
Forked from [octocat/Spoon-Knife](#)
This repo is for demonstration purposes only. Comments and issues may or may not be responded to.
Java
- nflidata**
Forked from [eljefe6a/nflidata](#)
Combining datasets with MapReduce on NFL play by play data.
Java
- madrailers.github.io**
Forked from [madrailers/madrailers.github.io](#)
madrailers home page
JavaScript
- tapme-iphone-game**
Tap Me iPhone Game from Learn to Code iOS
Objective-C

Hiring is hard


A story about a talented engineer's ups and downs.

Open source

I like open source because people aren't locked out from being able to contribute.

- One can't be at Apple and Google at once.

Open source




Dmitri Shuralyov
shurcool

I like making and assembling well-built components in Go. I value correctness and simplicity.

Block or report user

✉ shurcool@gmail.com
🌐 <https://dmitri.shuralyov.com>

Organizations



Overview Repositories 50 Stars 880 Followers 532 Following 243

Popular repositories

Go-Package-Store

An app that displays updates for the Go packages in your GOPATH.

Go ★ 711 🍴 18

Conception

Conception was an experimental project, looking for ways to make software development more efficient.

C++ ★ 542 🍴 30

markdownfmt

Like gofmt, but for Markdown.

Go ★ 492 🍴 29

Conception-go

An unfinished Go implementation of Conception.

Go ★ 255 🍴 15

gostatus

A command line tool that shows the status of Go repositories.

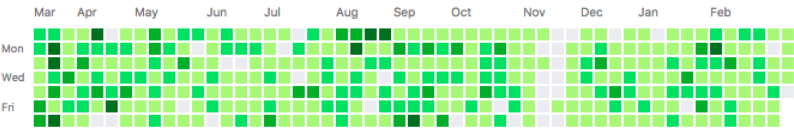
Go ★ 197 🍴 10

vfsgen



Takes an input http.FileSystem (likely at go generate time) and generates Go code that statically implements it.

Go ★ 136 🍴 8

2,657 contributions in the last year



[Learn how we count contributions.](#)

Less   More

Open source

Open source

- dmitri.shuralyov.com (<https://dmitri.shuralyov.com>) has a detailed activity feed.
- Why contribute? Because you want/need it yourself. Also to improve your skills, build a reputation, easier to find a job.
- Ways of contributing: triaging issues, reproducing issues, finding smaller repros, writing code.

Open source

It's normal to start small to test the waters.

- My 1st commit - github.com/golang/go/commit/4118665775dd21b2f244

<https://github.com/golang/go/commit/4118665775dd21b2f244> .

- My 5th commit - github.com/golang/go/commit/912ec1990bd09f8fc128

<https://github.com/golang/go/commit/912ec1990bd09f8fc128> .

Go community

Go is probably my first programming language experience with a "community".

There are conferences, meetups, talks, slack group, etc.

Go community



Go community



Now

Working full time on open source. Including personal projects (Go Package Store, etc.).

Contributing to Go, GopherJS.

First half of 2017, I'm prioritizing things I think are important (WebAssembly, etc.).

Second half, I'll be prioritizing ideas that can lead to profitability/sustainability.

Bonus: Browsers and Go

Browsers

I want to use Go, but it's hard to ignore the browser in 2017.

Go cross-compilation and wide platform support

Go already runs on many platforms.

```
# Desktop OSes.  
GOOS=darwin GOARCH=arm64 go build  
GOOS=linux GOARCH=amd64 go build  
GOOS=windows GOARCH=arm64 go build  
  
GOOS=plan9 GOARCH=amd64 go build # Plan 9.  
GOOS=linux GOARCH=s390x go build # Linux on IBM z Systems.  
  
# Mobile OSes.  
GOOS=darwin GOARCH=arm64 go build # iOS.  
GOOS=android GOARCH=arm go build # Android.
```

Go cross-compilation and wide platform support

Go already runs on many platforms.

Go cross-compilation and wide platform support

How about one more?

GopherJS

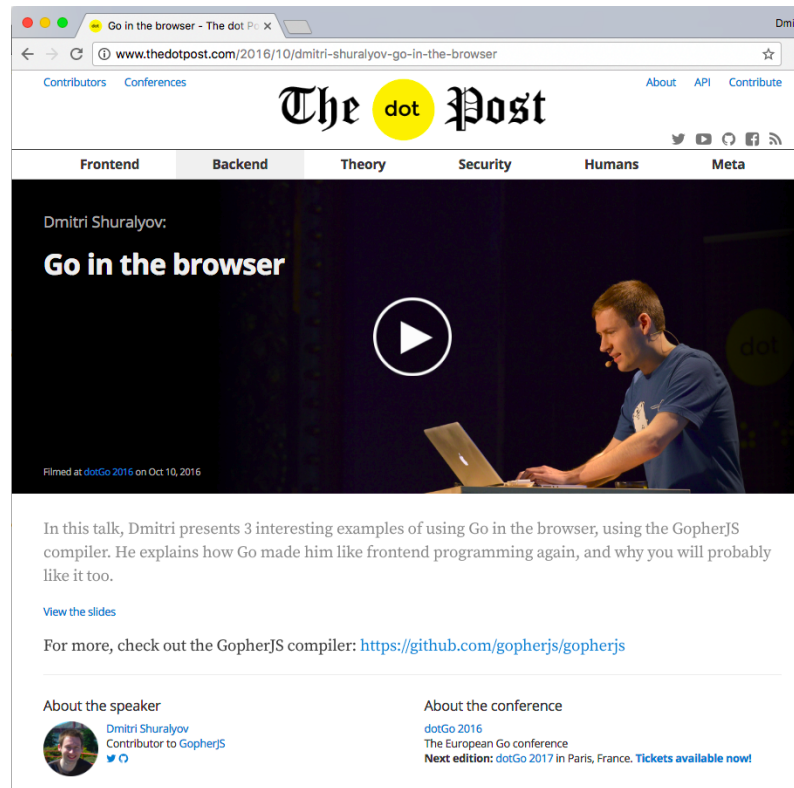
GopherJS is a compiler that compiles Go to JavaScript, which runs in browsers.

github.com/gopherjs/gopherjs (<https://github.com/gopherjs/gopherjs>)



- August 2013: Initial commit.
- 2013: Everything except goroutines, most stdlib tests passing. Can compile itself.
- 2014: Goroutines, channels, select statement, requires `//gopherjs:blocking`.
- 2015: Remove need for `//gopherjs:blocking`.
- 2016, 2017: Nothing major left to do. (Minor enhancements.)

Want to see more?



Video: www.thedotpost.com/2016/10/dmitri-shuralyov-go-in-the-browser

(<http://www.thedotpost.com/2016/10/dmitri-shuralyov-go-in-the-browser>)

Slides: dmitri.shuralyov.com/talks/2016/Go-in-the-browser/Go-in-the-browser.slide

(<https://dmitri.shuralyov.com/talks/2016/Go-in-the-browser/Go-in-the-browser.slide>)

Thank you

Dmitri Shuralyov

shurcool@gmail.com (mailto:shurcool@gmail.com)

<https://dmitri.shuralyov.com> (https://dmitri.shuralyov.com)

[@shurcool](http://twitter.com/shurcool) (http://twitter.com/shurcool)

