Go and GopherJS

18 February 2015

Dmitri Shuralyov
Software Engineer, Triggit

Overview

- Go, and what if you *could* compile it to JavaScript.
- GopherJS can do that quite well.
- Demos and examples.
- Performance, code size, debugging story, etc.

Go

Motivation

- "Go is a general-purpose language for building simple, reliable, and fast software. It's fun to write and a good fit for many use cases including web apps, network services, and command-line tools."

(From mmcgrana.github.io/2012/09/getting-started-with-go-on-heroku.html (http://mmcgrana.github.io/2012/09/getting-started-with-go-on-heroku.html).)

- Great language to write libraries in, functionality becomes available everywhere, just go get and import.

Good for writing general code

Use packages to abstract out platform-specific implementation details:

- `path/filepath`
- `os`, `os/exec`
- `net`, `net/http`

Use general interfaces that can be implemented and provided:

- `io.Reader` and `io.Writer`
- `vfs.FileSystem`:

```
type FileSystem interface {
    Opener
    Lstat(path string) (os.FileInfo, error)
    Stat(path string) (os.FileInfo, error)
    ReadDir(path string) ([]os.FileInfo, error)
    String() string
}
```

Go target platforms

- OS X, Linux, Windows, arm (Raspberry Pi), arm64 (soon (https://twitter.com/davecheney/status/567621293109821440)), Android (Go 1.4), iOS (Go 1.5~), others.
- More can be added in a coherent way.
- One thing missing?

GopherJS

- GopherJS - A compiler from Go to JavaScript.
- Its main purpose is to give you the opportunity to write front-end code in Go which will still run in all browsers.
- (There are/can be more than one Go -> JS compiler, just like there's `gc` and `gccgo`. It's an implementation detail.)

GopherJS GitHub Repo

- github.com/gopherjs/gopherjs (https://github.com/gopherjs/gopherjs)
- 1344 commits
- 2136 stars, 100 watchers
- 11 open issues (139 closed)
- GopherJS is written in pure Go.
- It can compile itself, thus GopherJS Playground (http://www.gopherjs.org/play/) is possible.

What is supported?

- Nearly everything. Including channels, goroutines, select.
- See compatibility table (https://github.com/gopherjs/gopherjs/blob/master/doc/packages.md) for list of supported packages (with passing tests).
- Compiler does some heavy lifting to support goroutines, which allows for normal (idiomatic style) Go code.

Demo.

Easy to get started.

```
$ go get -u github.com/gopherjs/gopherjs

$ gopherjs build
```

Reasons to use Go in frontend

- Like Node.js, share common code/logic between frontend and backend components.
- Familiar tools. `gofmt`, `goimports`, godoc.org (https://godoc.org), `go test`, `go test -bench .`.
- Familiar types (int, uint16, []byte, string), no need for equality table, static type checking.
- Familiar compilation errors, **refactoring**.
- Familiar concurrency, goroutines, blocking receiving, instead of callbacks.
- Familiar libraries like `net/url`, `time`, `html/template`, third party ones like `blackfriday`, etc.
- Ability to start from ground up with solid foundation and build high quality, sophisticated and complicated frontend UIs and projects.

Packages that can be compiled to JavaScript

go/parser and go/printer

```
func process(input string) string {
    // Parse the AST.
    fset := token.NewFileSet()
    fileAst, parseErr := parser.ParseFile(fset, "", input, parser.ParseComments|parser.AllErrors)

    // Print the AST.
    var config = &printer.Config{Mode: printer.UseSpaces | printer.TabIndent, Tabwidth: 8}
    var buf bytes.Buffer
    err := config.Fprint(&buf, fset, fileAst)
    if err != nil {
        panic(err)
    }

    // Append parsing errors, if any.
    if parseErr != nil {
        buf.WriteString("\n---\n" + parseErr.Error())
    }

    return buf.String()
}
```

Packages that can be compiled to JavaScript

```
github.com/russross/blackfriday
github.com/microcosm-cc/bluemonday
github.com/sourcegraph/syntaxhighlight
github.com/shurcooL/go/highlight_go
github.com/shurcooL/go/highlight_diff
go/format
github.com/shurcooL/markdownfmt/markdown
```

- dmitri.shuralyov.com/projects/live-markdown/live-markdown.html

  (http://dmitri.shuralyov.com/projects/live-markdown/live-markdown.html)

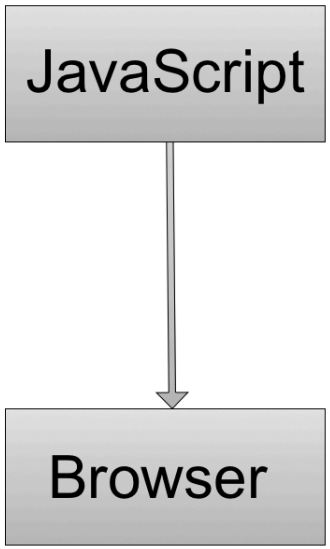
Packages that can be compiled to JavaScript

Achieving all that in the browser took minutes, because existing Go code could be reused:

```
import "github.com/shurcooL/go/github_flavored_markdown"

func run(event dom.Event) {
    output.SetInnerHTML(string(github_flavored_markdown.Markdown([]byte(input.Value))))
}

func main() {
    input.AddEventListener("input", false, run)
    input.Value = initial
    input.SelectionStart, input.SelectionEnd = len(initial), len(initial)
    run(nil)
}
```

Running Go code in the browser?

- The browser is a strange execution environment.
- Even if you can compile Go to JavaScript, it takes getting used to (but it'll be insightful).



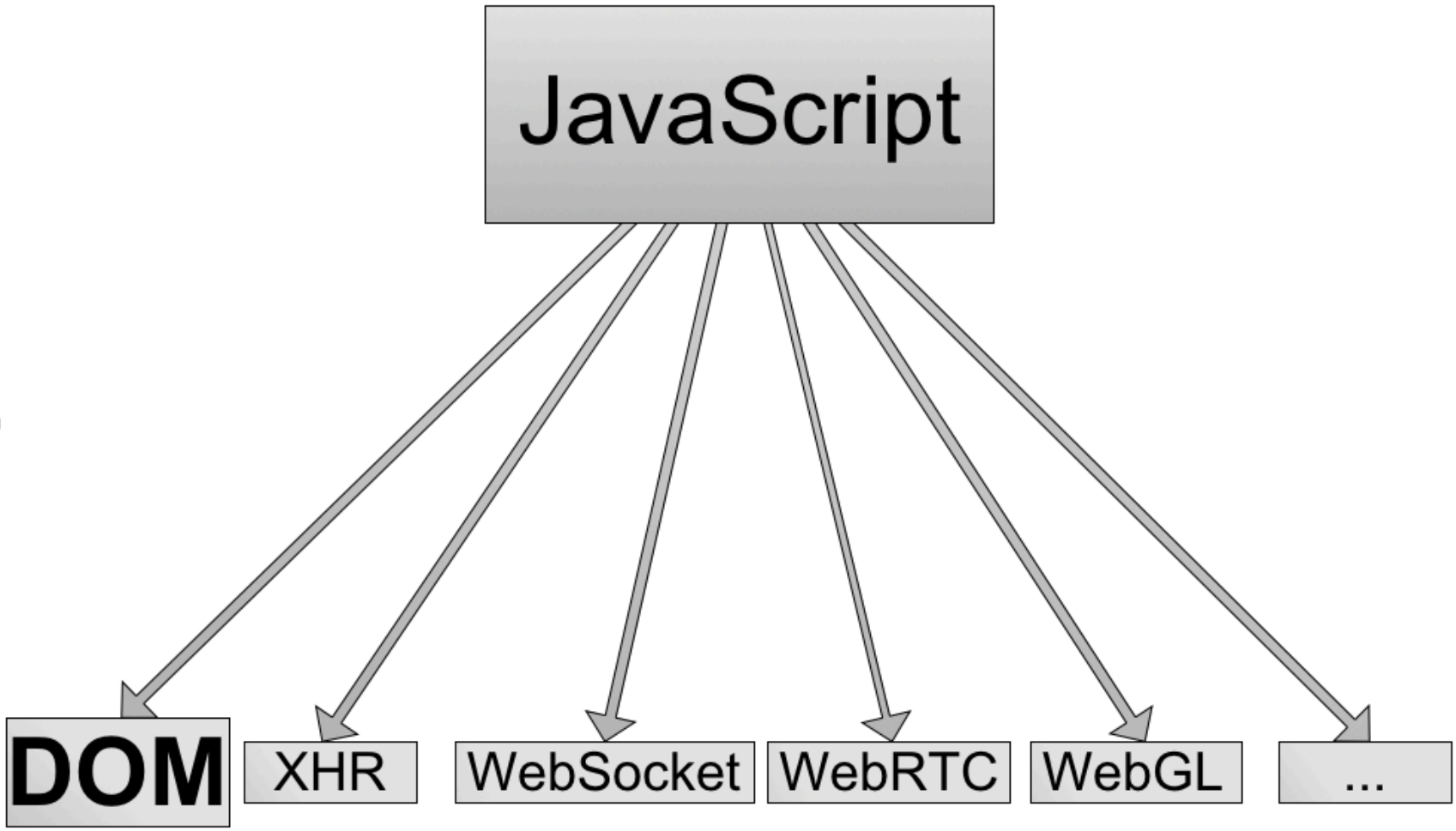
Running Go code in the browser?

- To do interesting things, you need to be able to have side effects (other than printing to console).

APIs in the browser

- DOM API
- XMLHttpRequest API
- WebSocket API
- WebRTC API
- WebGL API
- Geolocation API
- Gamepad API
- Notification API
- local file storage, full screen mode, mouse lock APIs
- ...

APIs in the browser



GopherJS and JavaScript

godoc.org/github.com/gopherjs/gopherjs/js (http://godoc.org/github.com/gopherjs/gopherjs/js)

Accessing native JavaScript APIs in Go code:

```
// document.write("Hello world!");

js.Global.Get("document").Call("write", "Hello world!")
```

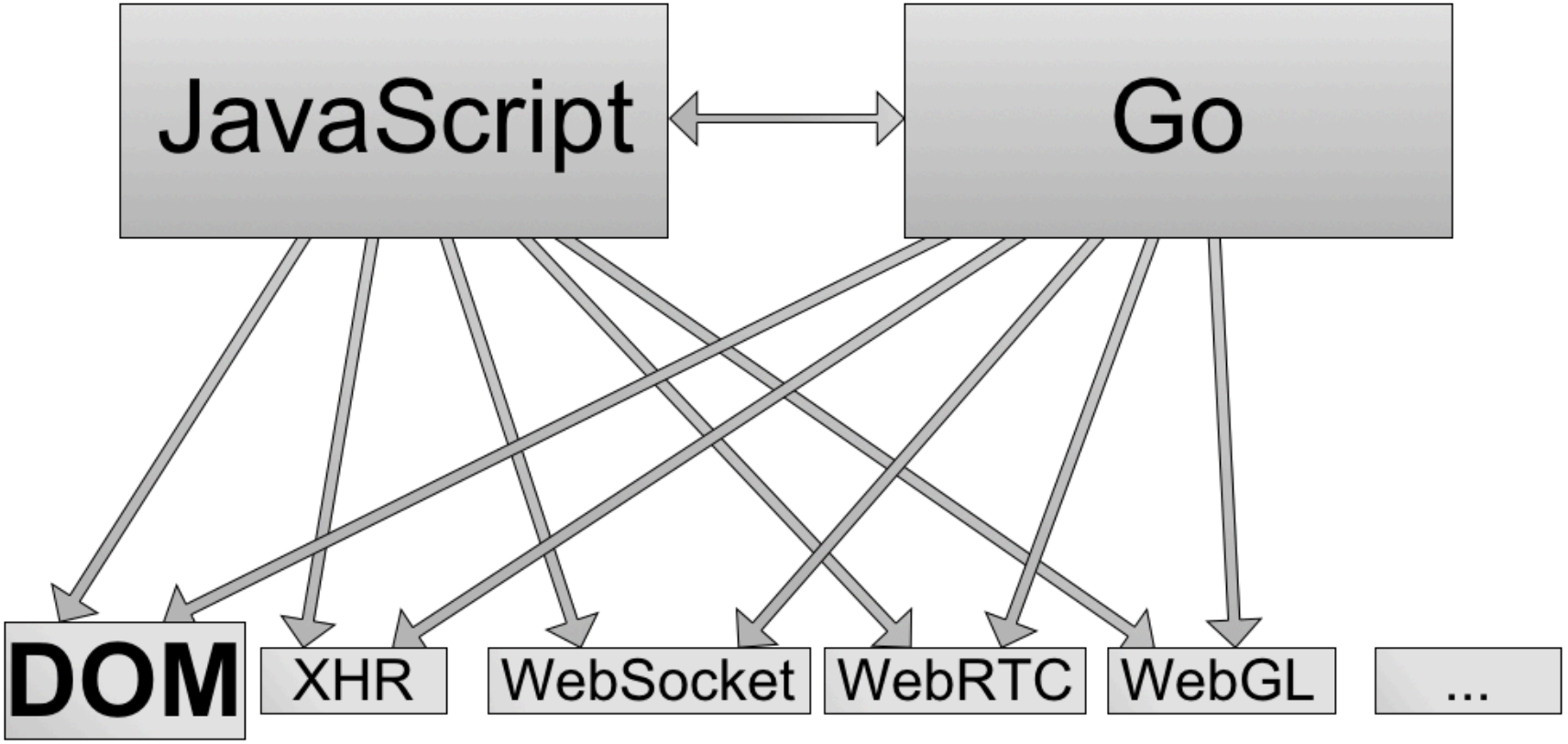
Providing Go functionality to other JavaScript code:

```
someGoFunc := func() {
    ...
}
js.Global.Set("SomeFunction", someGoFunc)
```

Type conversions between Go types and JavaScript types

godoc.org/github.com/gopherjs/gopherjs/js (http://godoc.org/github.com/gopherjs/gopherjs/js)

APIs in the browser

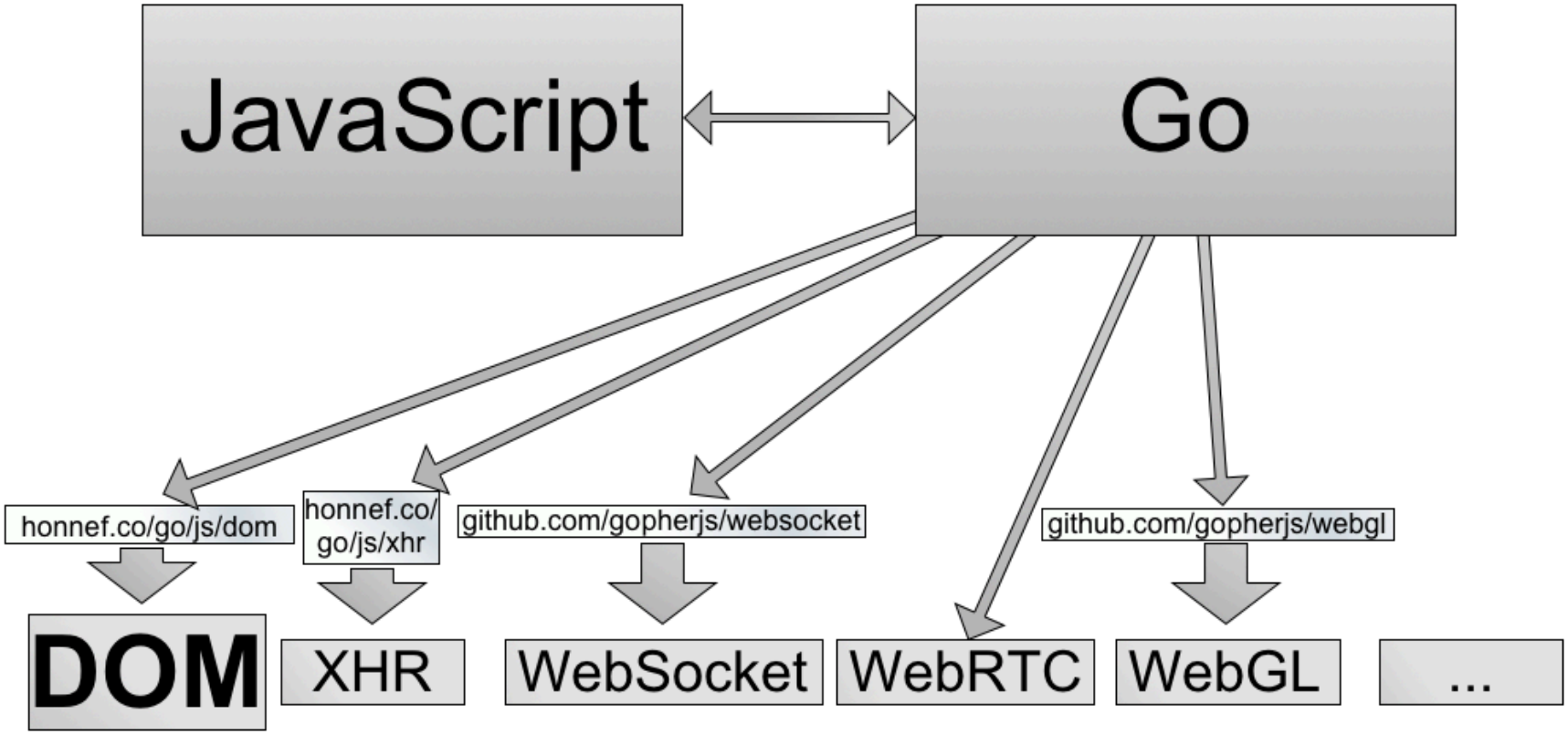


APIs in the browser

- DOM API: honnef.co/go/js/dom (https://godoc.org/honnef.co/go/js/dom)
- XMLHttpRequest API: honnef.co/go/js/xhr (https://godoc.org/honnef.co/go/js/xhr)
- WebSocket API: github.com/gopherjs/websocket (https://godoc.org/github.com/gopherjs/websocket)
- WebRTC API
- WebGL API: github.com/gopherjs/webgl (https://godoc.org/github.com/gopherjs/webgl)
- Geolocation API (part of honnef.co/go/js/dom (https://godoc.org/honnef.co/go/js/dom))
- Gamepad API
- Notification API
- local file storage, full screen mode, mouse lock APIs
- ...

Source: github.com/gopherjs/gopherjs/wiki/bindings (https://github.com/gopherjs/gopherjs/wiki/bindings)

APIs in the browser



Bindings to JS libraries

- AngularJS: github.com/gopherjs/go-angularjs (https://godoc.org/github.com/gopherjs/go-angularjs) (GopherJS Playground uses it.)
- D3: github.com/iansmith/d3 (https://godoc.org/github.com/iansmith/d3)
- jQuery: github.com/gopherjs/jquery (https://godoc.org/github.com/gopherjs/jquery)

Go Wrappers

- Can always go to JavaScript directly if you need to do something custom or for debugging.
- E.g., github.com/shurcooL/frontend/blob/d747e3d6ba5d42003950c40d3302cd6d30afdce3/sel list-view/main.go#L223-L224 (https://github.com/shurcooL/frontend/blob/d747e3d6ba5d42003950c40d3302cd6d30afdce3/select-list-view/main.go#L223-L224)

GopherJS Issue Resolution Times

- github.com/gopherjs/gopherjs/issues/150#issuecomment-69047234 (https://github.com/gopherjs/gopherjs/issues/150#issuecomment-69047234)

"Wow, awesome 1 hour fix, that was fast! Thanks!"

- github.com/gopherjs/gopherjs/issues/147#issuecomment-68966027 (https://github.com/gopherjs/gopherjs/issues/147#issuecomment-68966027)

"Wow, that was fast - thank you very much for your efforts! :)"

- github.com/gopherjs/gopherjs/issues/156 (https://github.com/gopherjs/gopherjs/issues/156)

"Great work! Thank you!"

- github.com/gopherjs/gopherjs/issues/158#issuecomment-70358592 (https://github.com/gopherjs/gopherjs/issues/158#issuecomment-70358592)

"Thanks for your prompt replies!"

GopherJS GitHub Repo

- Over a year ago, github.com/gopherjs/gopherjs/tree/2b85b2215bc59e76eaf2bd5#what-is-supported

  (https://github.com/gopherjs/gopherjs/tree/2b85b2215bc59e76eaf2bd5#what-is-supported)

github.com/shurcooL/play/95

Minor change of topic...

Demo (http://dmitri.shuralyov.com/projects/Terrain-Demo/index.html).

View Source: gotools.org/github.com/shurcooL/play/95 (http://gotools.org/github.com/shurcooL/play/95)

(Also github.com/shurcooL/play/97 and github.com/shurcooL/Hover.)

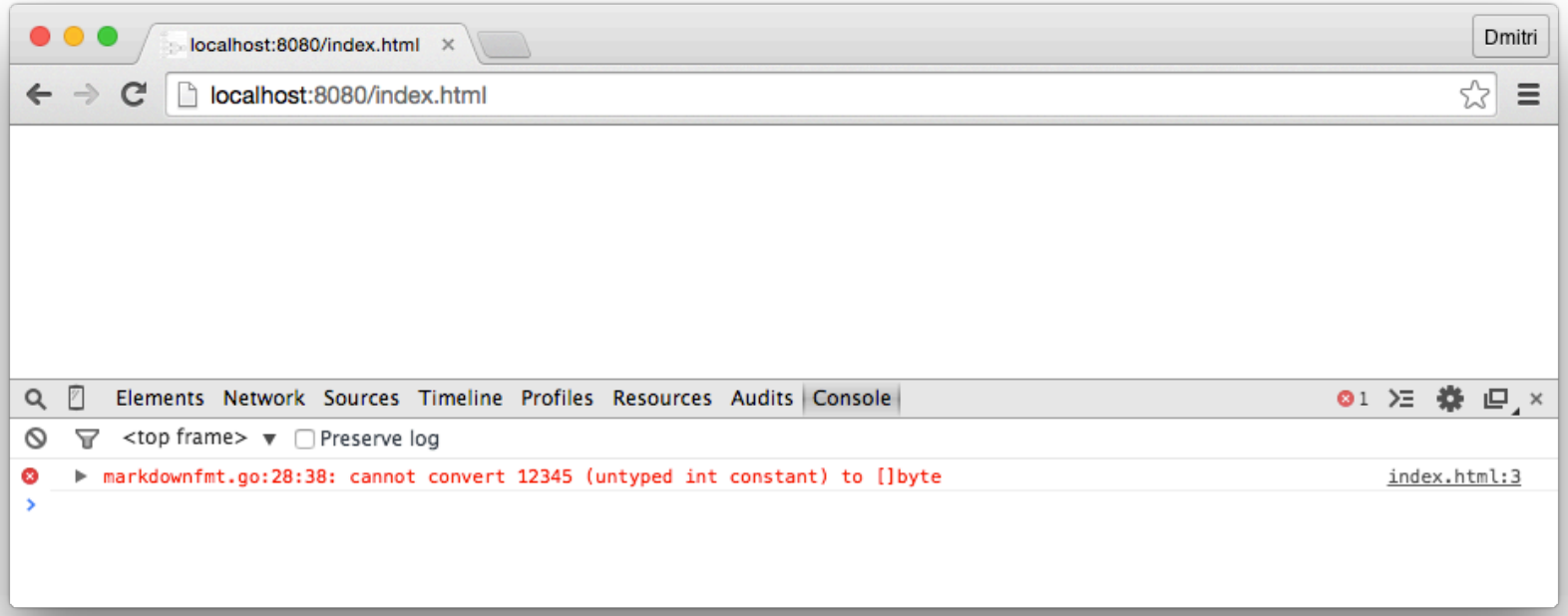
Other Demos

- tidwall.github.io/digitalrain/ (http://tidwall.github.io/digitalrain/)
- github.com/dimiro1/gopong (https://github.com/dimiro1/gopong)
- gotools.org (http://gotools.org) (Frontend functionality is written in Go.)

Compile Errors

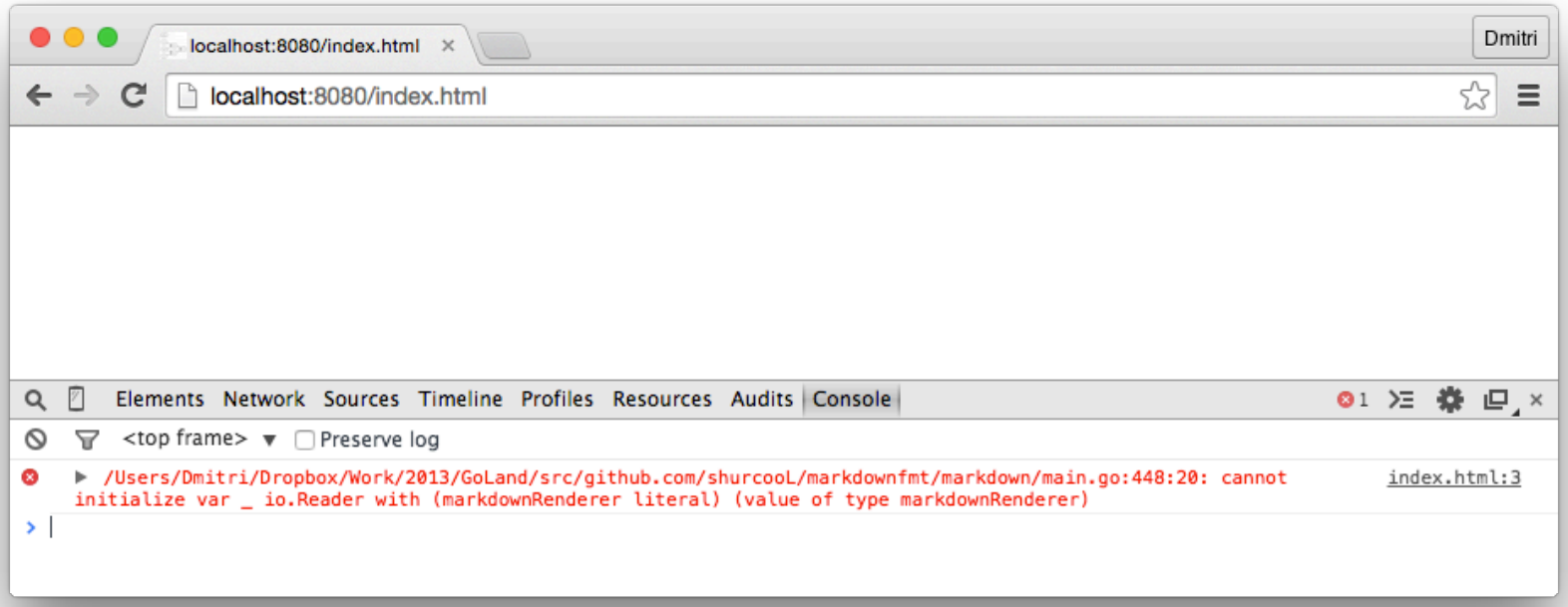
Compile Errors

```
output, err := markdown.Process("", 12345, nil)
if err != nil {
    panic(err)
}
```



Compile Errors

```
var _ io.Reader = markdownRenderer{}
```



Debugging

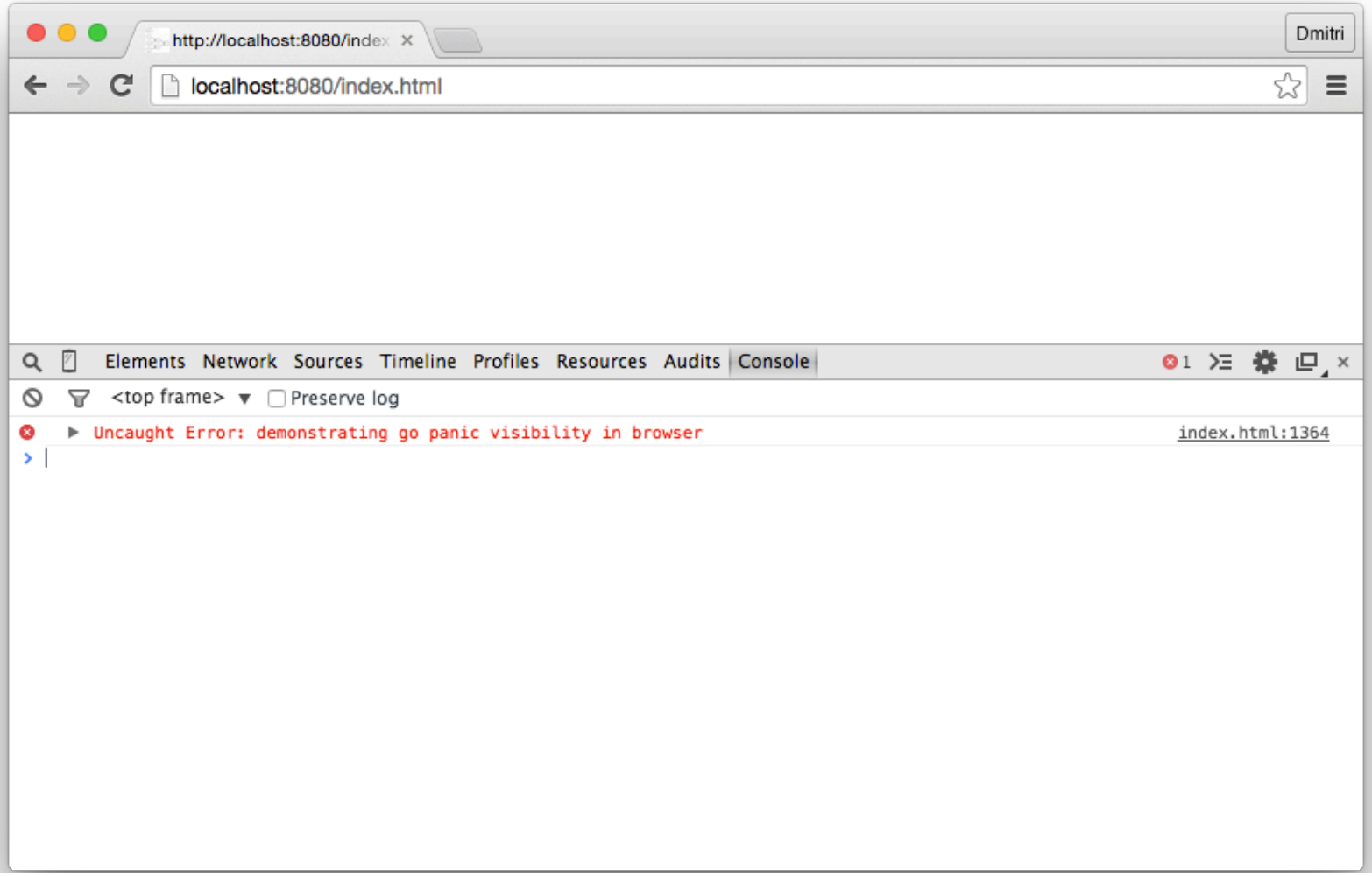
Debugging

- `fmt.Println("hello to println")`
- panic stack traces
- (static type checking, compiler errors eliminate a lot of problems)

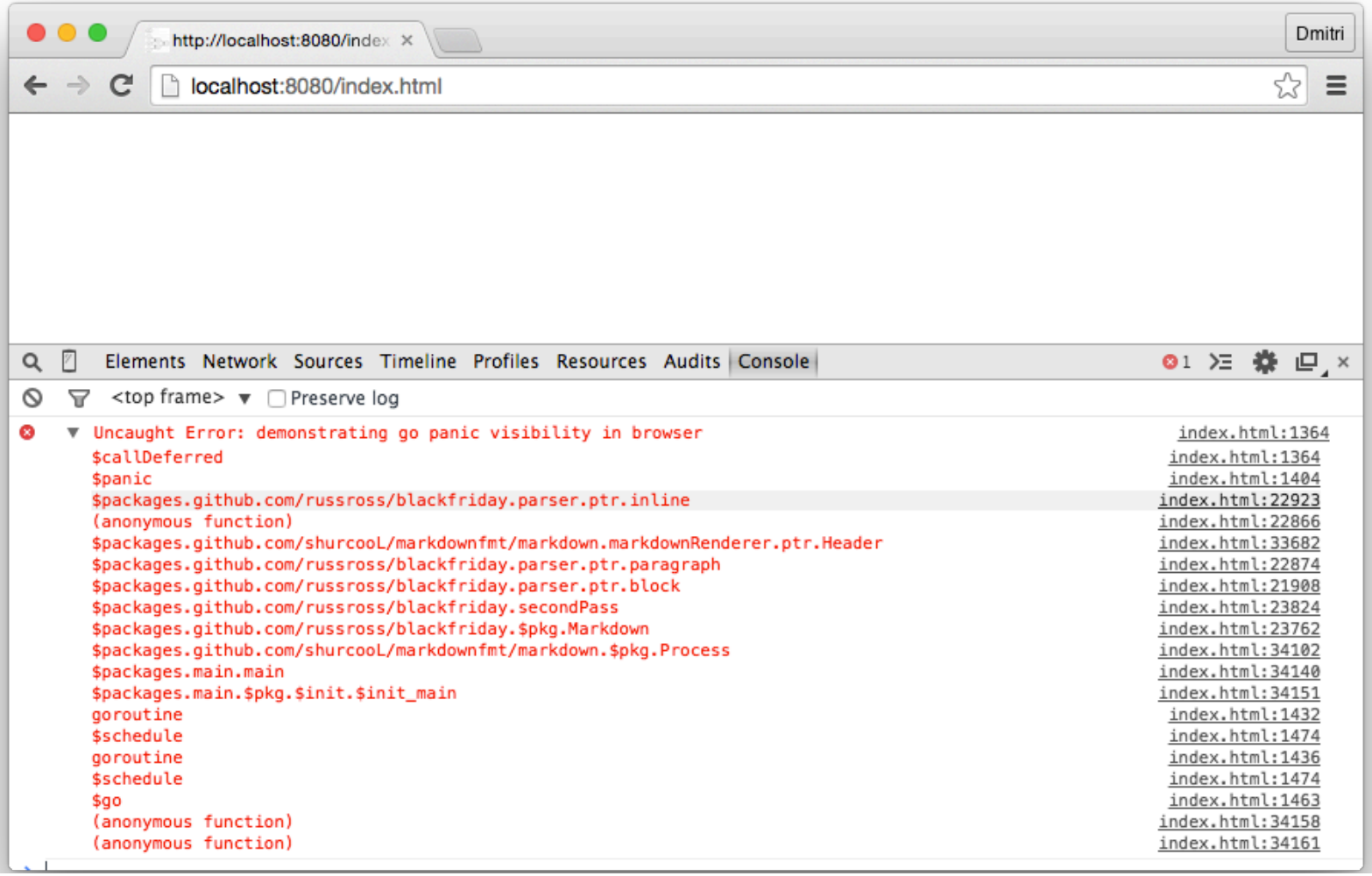
Debugging

```go
func (p *parser) inline(out *bytes.Buffer, data []byte) {
    // this is called recursively: enforce a maximum depth
    if p.nesting >= p.maxNesting {
        return
    }
    p.nesting++

    i, end := 0, 0
    for i < len(data) {
        // copy inactive chars into the output
        for end < len(data) && p.inlineCallback[data[end]] == nil {
            end++
        }

        p.r.NormalText(out, data[i:end])

        panic("demonstrating go panic visibility in browser")

        if end >= len(data) {
            break
        }
        i = end

        // call the trigger
        handler := p.inlineCallback[data[end]]
        if consumed := handler(p, out, data, i); consumed == 0 {
            // no action from the callback; buffer the byte for later
            end = i + 1
        } else {
            // skip past whatever the callback used
            i += consumed
            end = i
        }
    }

    p.nesting--
}
```

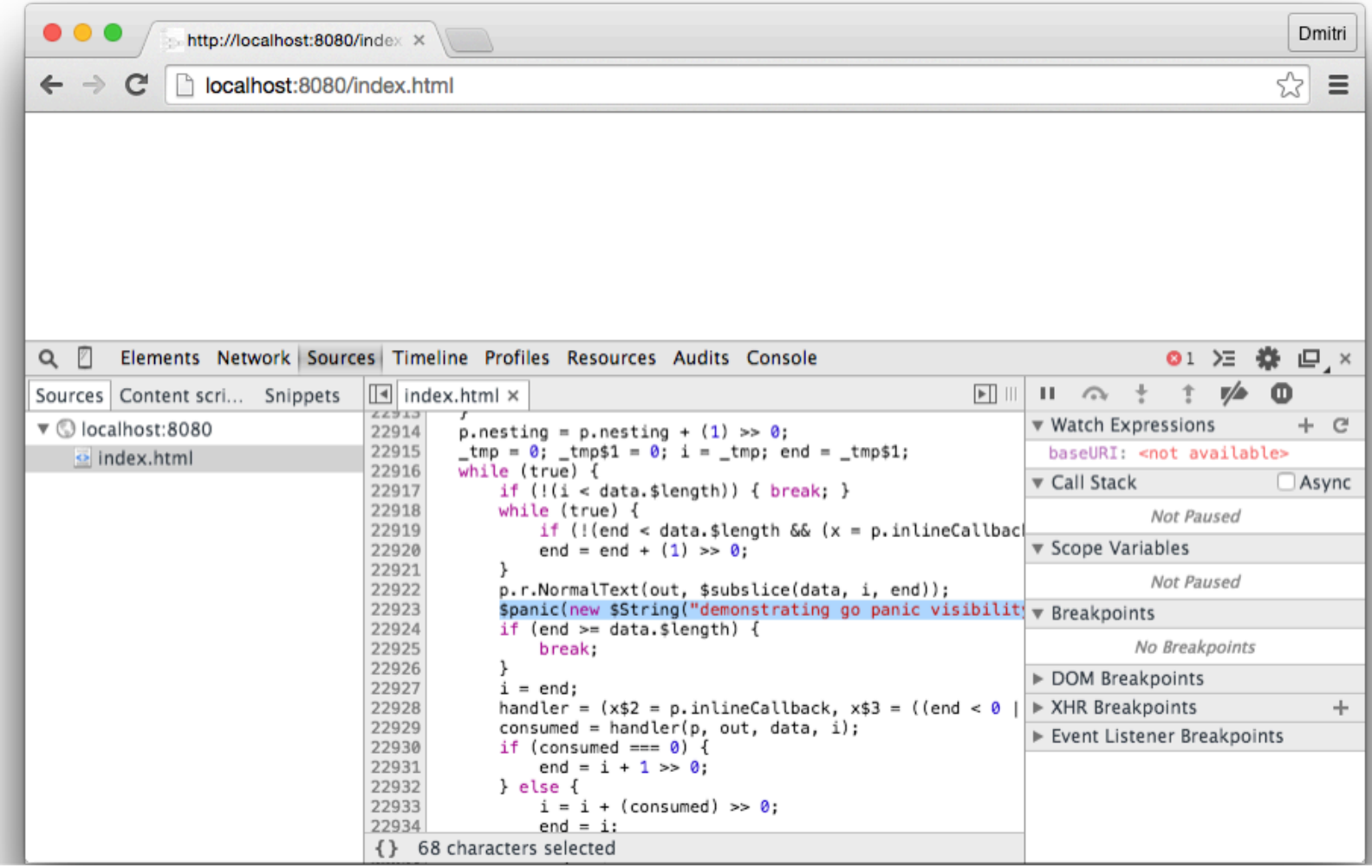
Debugging


http://localhost:8080/index
Dmitri
localhost:8080/index.html
Elements Network Sources Timeline Profiles Resources Audits Console
1
<top frame>
Preserve log
Uncaught Error: demonstrating go panic visibility in browser
index.html:1364

Debugging


http://localhost:8080/index
Dmitri
localhost:8080/index.html
Elements Network Sources Timeline Profiles Resources Audits Console
<top frame> Preserve log
Uncaught Error: demonstrating go panic visibility in browser index.html:1364
$callDeferred index.html:1364
$panic index.html:1404
$packages.github.com/russross/blackfriday.parser.ptr.inline index.html:22923
(anonymous function) index.html:22866
$packages.github.com/shurcooL/markdownfmt/markdown.markdownRenderer.ptr.Header index.html:33682
$packages.github.com/russross/blackfriday.parser.ptr.paragraph index.html:22874
$packages.github.com/russross/blackfriday.parser.ptr.block index.html:21908
$packages.github.com/russross/blackfriday.secondPass index.html:23824
$packages.github.com/russross/blackfriday.$pkg.Markdown index.html:23762
$packages.github.com/shurcooL/markdownfmt/markdown.$pkg.Process index.html:34102
$packages.main.main index.html:34140
$packages.main.$pkg.$init.$init_main index.html:34151
goroutine index.html:1432
$schedule index.html:1474
goroutine index.html:1436
$schedule index.html:1474
$go index.html:1463
(anonymous function) index.html:34158
(anonymous function) index.html:34161

Debugging


http://localhost:8080/index
Dmitri
localhost:8080/index.html
Elements Network Sources Timeline Profiles Resources Audits Console
Sources Content scri... Snippets
localhost:8080
index.html
index.html
22914 p.nesting = p.nesting + (1) >> 0;
22915 _tmp = 0; _tmp$1 = 0; i = _tmp; end = _tmp$1;
22916 while (true) {
22917 if (!(i < data.$length)) { break; }
22918 while (true) {
22919 if (!(end < data.$length && (x = p.inlineCallbac
22920 end = end + (1) >> 0;
22921 }
22922 p.r.NormalText(out, $subslice(data, i, end));
22923 $panic(new $String("demonstrating go panic visibilit
22924 if (end >= data.$length) {
22925 break;
22926 }
22927 i = end;
22928 handler = (x$2 = p.inlineCallback, x$3 = ((end < 0 |
22929 consumed = handler(p, out, data, i);
22930 if (consumed === 0) {
22931 end = i + 1 >> 0;
22932 } else {
22933 i = i + (consumed) >> 0;
22934 end = i;
{} 68 characters selected
Watch Expressions
baseURI: <not available>
Call Stack
Async
Not Paused
Scope Variables
Not Paused
Breakpoints
No Breakpoints
DOM Breakpoints
XHR Breakpoints
Event Listener Breakpoints

Debugging

```go
func (p *parser) inline(out *bytes.Buffer, data []byte) {
    // this is called recursively: enforce a maximum depth
    if p.nesting >= p.maxNesting {
        return
    }
    p.nesting++

    i, end := 0, 0
    for i < len(data) {
        // copy inactive chars into the output
        for end < len(data) && p.inlineCallback[data[end]] == nil {
            end++
        }

        p.r.NormalText(out, data[i:end])

        panic("demonstrating go panic visibility in browser")

        if end >= len(data) {
            break
        }
        i = end

        // call the trigger
        handler := p.inlineCallback[data[end]]
        if consumed := handler(p, out, data, i); consumed == 0 {
            // no action from the callback; buffer the byte for later
            end = i + 1
        } else {
            // skip past whatever the callback used
            i += consumed
            end = i
        }
    }

    p.nesting--
}
```

Performance

Performance

- Not faster than pure hand-written JavaScript
- Not much slower either, acceptable for most general use
- Often the slowest part is actual DOM manipulation, etc.

Performance

- Possible to benchmark via `gopherjs test -bench .`
- Possible to fallback to JavaScript as "assembly"; rewrite slow parts with careful hand-tuned JS
- `asm.js` support planned, not yet implemented
- (PNaCl, etc. might happen in the future, by 2050 browsers may simply support/run Go natively)

Performance

- github.com/gopherjs/gopherjs#performance-tips (https://github.com/gopherjs/gopherjs#performance-tips)
- Careful of more expensive string manipulation (Go uses utf-8, Unicode) if in a tight loop.
- Huge improvements have been made. github.com/gopherjs/gopherjs/issues/142 (https://github.com/gopherjs/gopherjs/issues/142) Still plenty of opportunity remaining.

Size of generated code

- "Hello world" will be large due to fixed size overhead (Go/JS type conversions, etc.).
- Large program with same imports will be marginally larger.
- Extremely huge programs with huge recursive imports seem to max out at 200-350 KB.

Filename	Source	Go	GopherJS	Minified	Min+Gzip
simple.go	67 B	624 KB	67 KB	50 KB	12 KB
fmt_simple.go	"fmt" + 85 B	1920 KB	567 KB	392 KB	89 KB
peg_solitaire_solver.go	"fmt" + 2696 B	1929 KB	570 KB	395 KB	89 KB
markdownfmt.go	11000~ LoC	3701 KB	1681 KB	1135 KB	238 KB

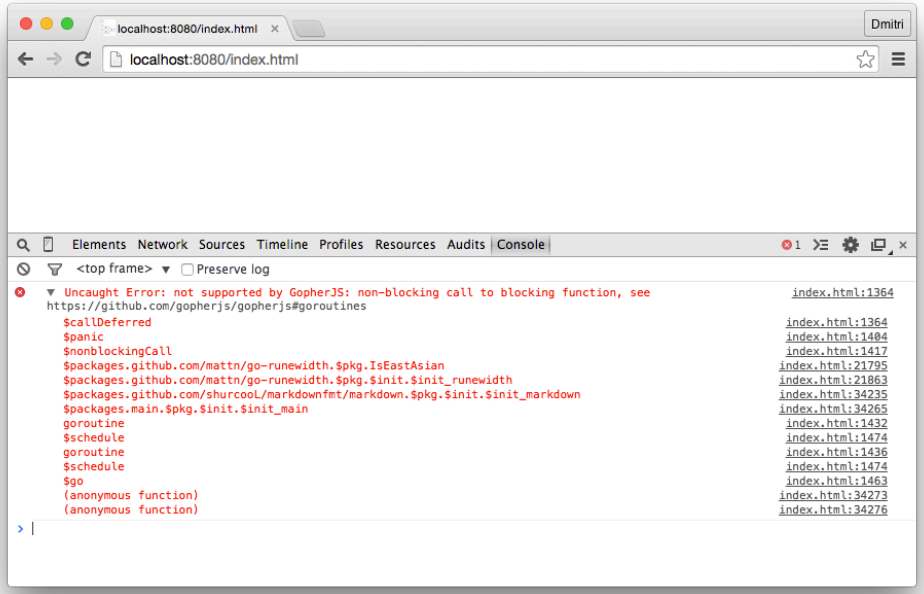
- github.com/gopherjs/gopherjs/issues/136#issuecomment-74445407

(https://github.com/gopherjs/gopherjs/issues/136#issuecomment-74445407)

Limitations

Limitations

- Need to mark blocking calls on interfaces as `//gopherjs:blocking`, or else. (Read full details here (https://github.com/gopherjs/gopherjs#goroutines).)
- (Currently ongoing work (https://github.com/gopherjs/gopherjs/issues/89) to improve blocking detection, making that work unneeded.)



Limitations

- The need to mark `//gopherjs:blocking` prevents implementing blocking `io.Reader`, `net.Conn` interfaces (since you can't easily modify Go standard library to add those comments there).
- github.com/gopherjs/gopherjs/issues/89 (https://github.com/gopherjs/gopherjs/issues/89)
- If a blocking `io.Reader` is supported, then it is possible to wrap a websocket (http://godoc.org/github.com/gopherjs/websocket) connection in a way that implements `net.Conn` interface. Doing that will allow using `net/rpc/jsonrpc` package for RPC. It will also allow creating an `http.Client` with a custom `http.Transport` that wraps around xhr (http://godoc.org/honnef.co/go/js/xhr).

Limitations

- More steps to distribute apps that use GopherJS for frontend.
- `go generate` can be helpful in pre-compiling and bundling the generated js.
- Limited to one script per html page if you don't want to pay extra price for having two Go "runtimes".
- Still young and evolving, need to be able to adapt quickly, figure out and solve problems. Often travelling a path for the first time.
- (Minor improvement/API breaking change coming to use js.Object struct pointers rather than interface. See issue 174 (https://github.com/gopherjs/gopherjs/issues/174#issuecomment-74112195).)
- Less people familiar with Go than JavaScript, less existing "frameworks".

Advantages

- Get to use Go and its ecosystem (tools, websites, libraries, errors and type checking).
- Open ended possibilities. Just you, Go code, and whatever you want to create. As simple or sophisticated as you want.
- What would you rather invest into, and deal with 2 years from now? Imagine receiving pull requests, doing code review, maintaing and developing code further, etc.

Takeaway

Which language you use in the frontend is a *choice* you have to make.

Something fun to try at home

- Think of a neat general Go package (or multiple packages) you like, see if it can be compiled with GopherJS and used on a simple web page.
- (Also try the GopherJS Playground (http://www.gopherjs.org/play/).)

Community

github.com/gopherjs/gopherjs#community (https://github.com/gopherjs/gopherjs#community)

- GitHub (repo, issues, organization)
- Google Group
- IRC channel #GopherJS
- Slack channel #GopherJS
- Follow twitter.com/GopherJS (https://twitter.com/GopherJS)

Thank you

Dmitri Shuralyov
Software Engineer, Triggit
shurcooL@gmail.com (mailto:shurcooL@gmail.com)
https://github.com/shurcooL (https://github.com/shurcooL)
@shurcooL (https://twitter.com/shurcooL)

Extras

Packages that can be compiled to JavaScript

go/parser and go/printer

```
package main

import ("bytes"; "go/parser"; "go/printer"; "go/token"; "honnef.co/go/js/dom")

var document = dom.GetWindow().Document()

var input = document.GetElementByID("input").(*dom.HTMLTextAreaElement)
var output = document.GetElementByID("output").(dom.HTMLElement)

var initial = "package main\n\n..."

func run(_ dom.Event) {
    output.SetTextContent(process(input.Value))
}

func main() {
    input.AddEventListener("input", false, run)
    input.Value = initial
    input.SelectionStart, input.SelectionEnd = 153, 153
    run(nil)
}
```

Code Samples

```
shareIcon := document.GetElementByID("share-icon")
shareIcon.AddEventListener("click", false, func(event dom.Event) {
    event.PreventDefault()
    fmt.Println("clicked!")
})
```

Setting CSS style of an element.

```
shareIcon.Style().SetProperty("display", "none", "")
```

Thank you

Dmitri Shuralyov
Software Engineer, Triggit
shurcooL@gmail.com (mailto:shurcooL@gmail.com)
https://github.com/shurcooL (https://github.com/shurcooL)
@shurcooL (http://twitter.com/shurcooL)