Prospects for using Go libraries in Haxe
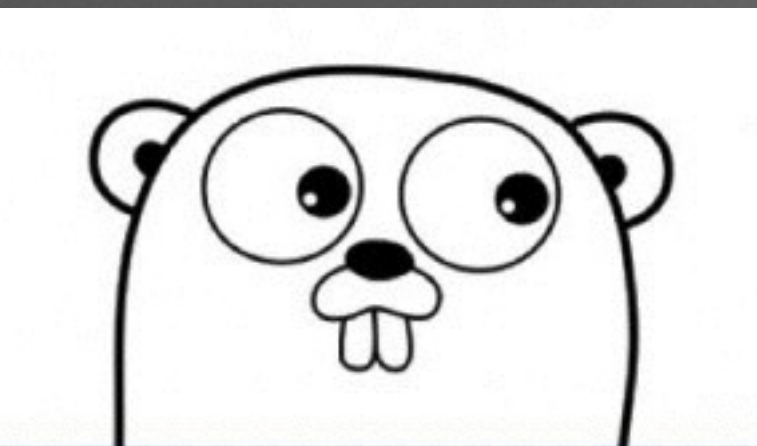
A talk by Elliott Stoneham at WWX2015

overview

- Why bother?

- TARDIS Go transpiler

- Math example

- Unicode example

- Image example

- Issues, hopes & dreams

Go vs Haxe

- Active Github repositories Q4 2014:

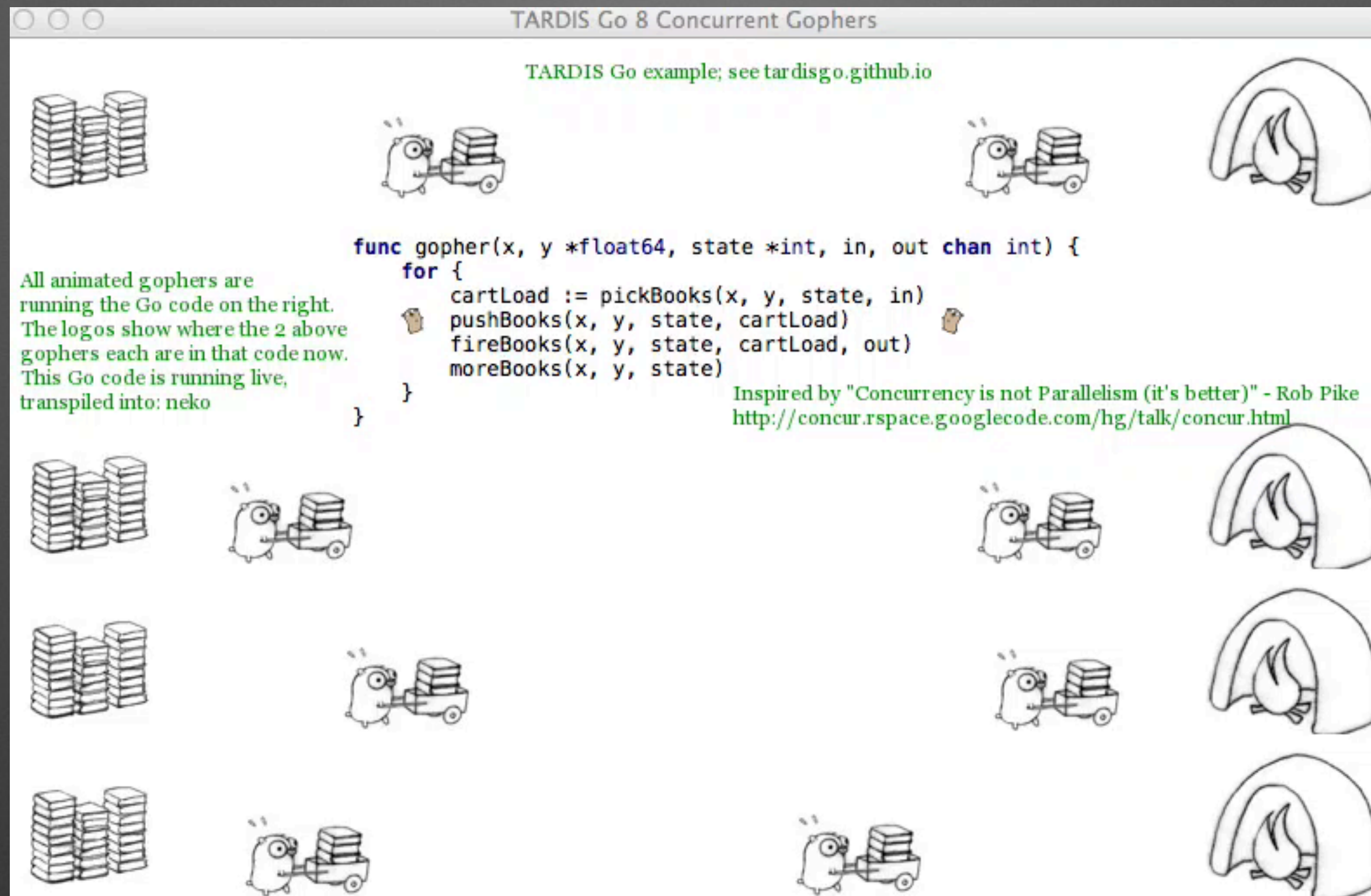- Go: 22,264

- Haxe: 1,134

- (data from githut.info)

tardisgo.github.io

For more explanation of how it works, watch my WWX2014 speech

```go
1   package main
2
3   import (
4       _ "math"
5       _ "strconv"
6   )
7
8   const tardisgoLibList = "math,strconv"
9
10  func main() {}
11  |
```

simple "math" example

Go code above generates 266 Haxe files containing 71.4k lines of code

```haxe
package ;

import tardis.*; // import the generated Go code

class Main {
    public static function main(){
        trace("Go 'math' and 'strconv' standard library trivial example");
        // for documentation see http://golang.org/pkg/math and http://golang.org/pkg/strconv
        // math.Nextafter returns the next representable float64 value after x towards y.
        // The built-in Haxe float->string conversions for cpp & cs round the results to "2",
        // so Go formatting was used to give a consistent cross-platform result.
        var na:String = Go_strconv_FFormatFFloat.hx(
            Go_math_NNextafter.hx(2.0,3.0),
            "g".charCodeAt(0),-1,64);
        trace("The next valid floating point value after 2.0 towards 3.0 is: "+na); // 2.0000000000000004
        na = Go_strconv_FFormatFFloat.hx(
            Go_math_NNextafter.hx(2.0,1.0),
            "g".charCodeAt(0),-1,64);
        trace("The next valid floating point value after 2.0 towards 1.0 is: "+na); // 1.9999999999999998
    }
}
```

```
Main.hx:7: Go 'math' and 'strconv' standard library trivial example
Main.hx:14: The next valid floating point value after 2.0 towards 3.0 is: 2.0000000000000004
Main.hx:18: The next valid floating point value after 2.0 towards 1.0 is: 1.9999999999999998
```

```
1  package main
2
3  import _ "golang.org/x/text/unicode/norm"
4
5  const tardisgoLibList = "golang.org/x/text/unicode/norm"
6
7  func main() {}
8
```

unicode normalisation example

Go code above generates 1,446 Haxe files containing 408.5k lines of code

```
package ;
import tardis.*; // import the generated Go code
class Main {
    public static function main(){
        trace("Go unicode normalization library example");
        // for documentation see https://godoc.org/golang.org/x/text/unicode/norm
        compare("aaa","aaa");                                        // aaa == aaa ? true
        compare("aaa","aab");                                        // aaa == aab ? false
        compare("a\u0300a", "\u00E0a");                              // a`a == àa ? true
        compare("a\u0300\u0320b", "a\u0320\u0300b");                 // a`_b == a_`b ? true
        compare("\u1E0A\u0323", "\x44\u0323\u0307");                 // Ḋ. == D.˙ ? true
        // A character that decomposes into multiple segments spans several iterations.
        compare("\u3304", "\u30A4\u30CB\u30F3\u30AF\u3099");         // ㌄ == イニング ? true
    }
    static function compare(a1:String,b1:String):Bool{
        // create the Go type
        var form:GoType_golang_dot_org_47_x_47_text_47_unicode_47_norm_dot_FForm;
        // set the type to translate strings to NFKD normal form
        form = Go.golang_dot_org_47_x_47_text_47_unicode_47_norm_NNFFKKDD;
        // translate
        var a2 = form._String(a1);
        var b2 = form._String(b1);
        // test if the same
        var ret = a2 == b2;
        // show that we've done it for debug purposes
        trace(a1,"==",b1,"?",ret);
        return ret;
    }
}
```

```go
package main

import (
	_ "io/ioutil"

	_ "github.com/koyachi/go-nude"
)

const tardisgoLibList = "github.com/koyachi/go-nude,io/ioutil"

func main() {}
```

nudity detection example

Go code above generates 2,044 Haxe files containing 471.3k lines of code

```haxe
static var fName:String;

public static function main(){
    trace("Starting...");
    tardis.Go_main_main.hx();
    trace("Go nude detection library ready");
    js.Browser.document.getElementById("inputFileToLoad").onchange = encodeImageFileAsURL;
}
public static function writeFile(b:haxe.io.Bytes,typ:String):Void{
    var sl=tardis.Slice.fromBytes(b); // make a Go byte slice
    trace(typ,sl.len());
    var p="temp."+typ; // the name of our file
    // write the file in the pseudo file system
    var err=tardis.Go_io_47_ioutil_WWriteFFile.hx(p,sl,438 /*0666*/);
    if(err!=null){
        trace(p," ioutil.WriteFile() had error: ",err);
    }
    trace("wrote file ",p);
    fName=p;
}
public static function isNude():Bool {
    var ret=tardis.Go_github_dot_com_47_koyachi_47_go_45_nude_IIsNNude.hx(fName);
    if(ret.r1!=null){
        trace("nude.IsNude() had error: ",ret.r1);
        return false;
    }
    return ret.r0;
}
```

Live Demo

http://tardisgo.github.io/go-nude/index.html

Issues, hopes & dreams

- **Issues:** the immaturity of TARDIS Go => large code sizes, slow execution speed, and ugly Haxe call interface

- **Hopes:** that Haxe programmers find it useful

- **Dreams:** that most Go libraries available in Haxe