Programming in Go

Matt Holiday
Christmas 2020

Homework

Homework #2

Exercise 5.5 from *GOPL:* implement `countWordsAndImages`

Actually, given some HTML as raw text, parse it into a document and then call your counting routine to detect and count words and images (you can follow the book's example).

Don't worry about getting HTML from an HTTP query; we're not there yet.

See Homework #1 for counting words.

What happens if the HTML document is empty?

matt4biz/go-class-exer-5.5

Homework #2

```
package main

import (
    "bytes"
    "fmt"
    "os"
    "strings"

    "golang.org/x/net/html"
)
```

Homework #2

```
var raw = `
<!DOCTYPE html>
<html>
  <body>
    <h1>My First Heading</h1>
      <p>My first paragraph.</p>
      <p>HTML images are defined with the img tag:</p>
      <img src="xxx.jpg" width="104" height="142">
  </body>
</html>`
```

Homework #2

```
func main() {
    doc, err := html.Parse(bytes.NewReader([]byte(raw)))

    if err != nil {
        fmt.Fprintf(os.Stderr, "parse failed: %s\n", err)
        os.Exit(-1)
    }

    words, pics := countWordsAndImages(doc)

    fmt.Printf("%d words and %d images\n", words, pics)
}

// outputs "14 words and 1 images"
```

Homework #2

```
func countWordsAndImages(doc *html.Node) (int, int) {
    var words, images int

    visit(doc, &words, &images)

    return words, images
}
```

Homework #2

```
func visit(n *html.Node, words, pics *int) {
    // if it's an element node then see what tag it has

    if n.Type == html.TextNode {
        *words += len(strings.Fields(n.Data))
    } else if n.Type == html.ElementNode && n.Data == "img" {
        *pics++
    }

    // then visit all the children using recursion

    for c := n.FirstChild; c != nil; c = c.NextSibling {
        visit(c, words, pics)
    }
}
```