**Отчет по лабораторной работе № 4**

Тарасов А. Т.

Группа ИУ9-31Б

**Задача** Реализовать программу синтаксического разбора гипертекста инфоблоков сниппетов новостей.

Исходный код программы:

package main

import (

    "html/template"

    "log"

    "net/http"

    "golang.org/x/net/html"

)

func getChildren(node \*html.Node) []\*html.Node {

    var children []\*html.Node

    for c := node.FirstChild; c != nil; c = c.NextSibling {

        children = append(children, c)

    }

    return children

}

func getAttr(node \*html.Node, key string) string {

    for \_, attr := range node.Attr {

        if attr.Key == key {

            return attr.Val

        }

    }

    return ""

}

func isText(node \*html.Node) bool {

    return node != nil && node.Type == html.TextNode

}

func isElem(node \*html.Node, tag string) bool {

    return node != nil && node.Type == html.ElementNode && node.Data == tag

}

func isDiv(node \*html.Node, class string) bool {

    return isElem(node, "div") && getAttr(node, "class") == class

}

func readItem(item \*html.Node) \*Item {

    if a := item.FirstChild; isElem(a, "a") {

        cs := getChildren(a)

        if isText(cs[0]) {

            return &Item{

                Ref:   getAttr(a, "href"),

                Title: cs[0].Data,

            }

        } else if cs := getChildren(a); len(cs) == 2 && isElem(cs[0], "time") && isText(cs[1]) {

            return &Item{

                Ref:   getAttr(a, "href"),

                Time:  getAttr(cs[0], "title"),

                Title: cs[1].Data,

            }

        }

    }

    return nil

}

type Item struct {

    Ref, Time, Title string

}

func downloadNews() []\*Item {

    log.Println("sending request to lenta.ru")

    if response, err := http.Get("http://lenta.ru"); err != nil {

        log.Println("request to lenta.ru failed", "error", err)

    } else {

        defer response.Body.Close()

        status := response.StatusCode

        log.Println("got response from lenta.ru", "status", status)

        if status == http.StatusOK {

            if doc, err := html.Parse(response.Body); err != nil {

                log.Println("invalid HTML from lenta.ru", "error", err)

            } else {

                log.Println("HTML from lenta.ru parsed successfully")

                return search(doc)

            }

        }

    }

    return nil

}

func search(node \*html.Node) []\*Item {

    var res []\*Item

    if isDiv(node, "b-yellow-box\_\_wrap") || isDiv(node, "span4") {

        var items []\*Item

        for c := node.FirstChild; c != nil; c = c.NextSibling {

            if isDiv(c, "item") {

                if item := readItem(c); item != nil {

                    items = append(items, item)

**}**

**}**

**}**

**res = append(res, items...)**

**}**

**for c := node.FirstChild; c != nil; c = c.NextSibling {**

**if items := search(c); items != nil {**

**res = append(res, items...)**

**}**

**}**

**return res**

**}**

**//===================================================================================================**

**func main() {**

**http.HandleFunc("/", HomeRouterHandler)**

**err := http.ListenAndServe(":3000", nil)**

**if err != nil {**

**log.Fatal("ListenAndServe: ", err)**

**}**

**}**

**var pageTemplate \*template.Template**

**func HomeRouterHandler(writer http.ResponseWriter, request \*http.Request) {**

**items := downloadNews()**

**if pageTemplate, err := template.ParseFiles("index.html"); err != nil {**

**panic(err)**

**} else {**

**pageTemplate.Execute(writer, items)**

**}**

**}**

**Пример работы программы:**



