1. <code class="language-plain">package main
3. import (
4. "./goquery"
5. "errors"
6. "fmt"
7. "io"
8. "io/ioutil"
9. "log"
10. "net/http"
11. "net/url"
12. "os"
13. "path/filepath"
14. "strings"
15. )
17. func GetNews(s string) {
19. file := "news.txt"
20. fout, err := os.Create(file)
21. defer fout.Close()
22. if err != nil {
23. fmt.Println(file, err)
24. return
25. }
26. //  str := os.Args[1]
27. str := s
28. //doc, err := goquery.NewDocument("http://news.qq.com")
29. doc, err := goquery.NewDocument(str)
30. if err != nil {
31. log.Fatal(err)
32. }
34. doc.Find(".content").Each(func(i int, s \*goquery.Selection) {
35. html, \_ := s.Html()
37. fout.WriteString(s.Text())
38. fout.WriteString(html)
39. fout.WriteString("\n")
40. })
41. }
42. func downloadPic(urls []string, dir string) error {//xiazai tupian
43. for \_, link := range urls {
44. resp, err := http.Get(link)
45. if err != nil {
46. log.Fatal(err)
47. }
48. defer resp.Body.Close()
49. if resp.StatusCode != http.StatusOK {
50. log.Fatal(resp.Status)
51. }
52. FileName := filepath.Base(link)//创建一个文件名
53. FullName := filepath.Join(dir, FileName)//创建一个存放文件的文件名
54. f, err := os.Create(FullName)
55. if err != nil {
56. log.Panic("文件创建失败")
57. }
58. io.Copy(f, resp.Body)
59. }
60. return nil
61. }
62. func fetch(url string) ([]string, error) {//分析出图片的下载路径
63. var urls []string
64. resp, err := http.Get(url)
65. if err != nil {
66. log.Fatal(err)
67. }
68. defer resp.Body.Close()
69. if resp.StatusCode != http.StatusOK {
70. return nil, errors.New(resp.Status)
71. }
72. doc, err := goquery.NewDocumentFromResponse(resp)//传入 HTTP 响应，内部拿到res.Body(实现了 io.Reader) 后的处理方式类似 NewDocumentFromReader.
73. if err != nil {
75. log.Fatal(err)
76. }
77. doc.Find("img").Each(func(i int, s \*goquery.Selection) {//迭代查询
78. link, ok := s.Attr("src")
79. if ok {
80. urls = append(urls, link)
81. } else {
82. fmt.Println("no find addr")
83. }
84. })
85. return urls, nil
86. }
87. func CleanUrls(RootPath string, PicPath []string) []string {
88. var AbsolutePath []string
89. UrlInfo, err := url.Parse(RootPath)
90. if err != nil {
91. log.Fatal(err)
92. }
93. Scheme := UrlInfo.Scheme //获取链接协议
94. Host := UrlInfo.Host//获取链接主机名
95. for \_, SoucePath := range PicPath {//处理得到绝对路径
96. if strings.HasPrefix(SoucePath, "http") {
98. } else if strings.HasPrefix(SoucePath, "//") {
99. SoucePath = Scheme + ":" + SoucePath
100. } else if strings.HasPrefix(SoucePath, "/") {
101. SoucePath = Scheme + "://" + Host + SoucePath
102. } else {
103. SoucePath = filepath.Dir(RootPath) + SoucePath
104. }
105. AbsolutePath = append(AbsolutePath, SoucePath)
106. }
107. return AbsolutePath//返回绝对路径
108. }
109. func GetPicture(str string) {
111. //RootPath :="http://www.youzu.com"
112. //  RootPath := os.Args[1]
113. RootPath := str
114. PicPath, err := fetch(RootPath)
115. if err != nil {
116. log.Fatal(err)
117. }
118. AbsolutePath := CleanUrls(RootPath, PicPath)
119. picture := "/home/sgc/go/src/pachong/pachong/"//制定下载地址
120. tmpdir, err := ioutil.TempDir(picture, "sanyue")//制定文件夹前缀，后面不知道为什么乱加一堆数
121. fmt.Println(tmpdir)
122. err = downloadPic(AbsolutePath, tmpdir)
123. if err != nil {
125. log.Panic(err)
126. }
127. }
128. func main() {
129. if len(os.Args) != 2 {//根据传入的指令，选择要做的事情
130. fmt.Println("Usage :./pachong  |-p|-picture|-n|-news|")
131. os.Exit(1)
132. }
133. options := os.Args[1]
134. if options == "-n" || options == "-news" {
135. fmt.Println("please input addr:like http://example.com")
136. var str string
137. fmt.Scan(&str)
138. fmt.Println("pa qu news")
139. GetNews(str)
141. } else if options == "-p" || options == "-picture" {
142. fmt.Println("please input addr:like http://example.com")
143. var str string
144. fmt.Scan(&str)
145. fmt.Println("pa qu picture")
146. GetPicture(str)
148. }else{
149. fmt.Println("are you kidding me")
150. fmt.Println("Usage :./pachong  |-p|-picture|-n|-news|")
151. }
153. //GetNews()
154. //GetPicture()
155. }</code>