时间 2014-11-29
标签 golang goquery 天气 爬虫 栏目 Go
原文 http://blog.csdn.net/g39game/article/details/41595357

使用goquery抓取天气的demo。数据量有点多。目前按省份存储天气数据。存储到csv文件中。

```
package main

import (
 "code.google.com/p/mahonia"
 "encoding/csv"
 "fmt"
 "github.com/PuerkitoBio/goquery"
 "net/http"
 "os"
 "strings"
 "time"
)

var log = loger.Loger{
 Level: loger.DEBUG,
}

const (
 YEAR      = 2013
 SleepTime = 100 //毫秒
)

func main() {
 sc, cc := GetCity()
 var weatherInfoAll []*WeaterInfo
 for key, value := range sc {
  filePath := fmt.Sprintf("%d%s.csv", YEAR, key)
  _, err := os.Stat(filePath)
  if err == nil {
   continue
  }
  weatherInfoAll = make([]*WeaterInfo, 0, 100000)
  for _, city := range value {
   name := cc[city]
   log.Debug("get ", key, city)
   client := &http.Client{}
   weatherInfoYear := GetWeather(client, key, city, name)
   weatherInfoAll = append(weatherInfoAll, weatherInfoYear...)
  }
  SaveToCSV(key, weatherInfoAll)
 }
}

//返回数据为省份=>城市名    城市名=>拼音.html
func GetCity() (sc map[string][]string, cc map[string]string) {
 url := "http://www.tianqihoubao.com/lishi/"
 request, err := http.NewRequest("GET", url, nil)
 if err != nil {
  log.Log(err)
  return
 }
 request.Header.Add("User-Agent", "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubu
ntu Chromium/39.0.2171.65 Chrome/39.0.2171.65 Safari/537.36")
 request.Header.Add("referer", "http://www.tianqihoubao.com/")
 resp, err := http.DefaultClient.Do(request)
 if err != nil {
  log.Log(err)
```

```
  return
 }
 document, err := goquery.NewDocumentFromResponse(resp)
 if err != nil {
  log.Log(err)
  return
 }
 gbk := mahonia.NewDecoder("gbk")
 sc = make(map[string][]string)
 cc = make(map[string]string)
 document.Find(".citychk").Find("dl").Each(func(index int, s *goquery.Selection) {
  province := gbk.ConvertString(s.Find("dt").Find("b").Text())
  citys := make([]string, 0, 20)
  s.Find("dd").Find("a").Each(func(index int, se *goquery.Selection) {
   uri, exists := se.Attr("href")
   if !exists {
    return
   }
   name := gbk.ConvertString(se.Text())
   uri = strings.Replace(uri, ".html", "", -1)
   citys = append(citys, name)
   cc[name] = uri
  })
  sc[province] = citys
 })
 return
}

type WeaterInfo struct {
 Province string
 City     string
 Date     string
 Info     string
 Temp     string
 Wind     string
}

func GetWeather(client *http.Client, province, city, name string) []*WeaterInfo {
 baseUrl := fmt.Sprintf("http://www.tianqihoubao.com%s/month/%%s", name)
 weaterInfoYear := make([]*WeaterInfo, 0, 380)
 for i := 1; i <= 12; i++ {
  url := fmt.Sprintf(baseUrl, fmt.Sprintf("%d%02d.html", YEAR, i))
  weaterInfos := GetWeatherInfo(client, province, city, url)
  weaterInfoYear = append(weaterInfoYear, weaterInfos...)
  time.Sleep(time.Millisecond * SleepTime)
 }
 return weaterInfoYear
}

func GetWeatherInfo(client *http.Client, province, city, url string) (weaterInfos []*WeaterInfo) {
 request, err := http.NewRequest("GET", url, nil)
 if err != nil {
  log.Log(err)
  return
 }
 request.Header.Add("User-Agent", "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Ubu
ntu Chromium/39.0.2171.65 Chrome/39.0.2171.65 Safari/537.36")
 request.Header.Add("referer", "http://www.tianqihoubao.com/")
 resp, err := client.Do(request)
 if err != nil {
  log.Log(err)
  return
 }
 document, err := goquery.NewDocumentFromResponse(resp)
 if err != nil {
  log.Log(err)
  return
 }
 gbk := mahonia.NewDecoder("gbk")
```

```
 weaterInfos = make([]*WeaterInfo, 0, 31)
 document.Find("#content").Find("tbody").Find("tr").Each(func(index int, s *goquery.Selection) {
  //排除第一个
  if index == 0 {
   return
  }
  var date, info, temp, wind string
  s.Find("td").Each(func(index int, se *goquery.Selection) {
   if index == 0 {
    date = gbk.ConvertString(se.Find("a").Text())
   }
   if index == 1 {
    info = gbk.ConvertString(se.Text())
   }
   if index == 2 {
    temp = gbk.ConvertString(se.Text())
   }
   if index == 3 {
    wind = gbk.ConvertString(se.Text())
   }
  })
  weatherInfo := &WeaterInfo{
   Province: province,
   City:     city,
   Date:     date,
   Info:     info,
   Temp:     temp,
   Wind:     wind,
  }
  weaterInfos = append(weaterInfos, weatherInfo)
 })
 return
}

func SaveToCSV(file string, weatherInfos []*WeaterInfo) (err error) {
 filePath := fmt.Sprintf("%d%s.csv", YEAR, file)
 _, err = os.Stat(filePath)
 if err == nil {
  return
 }
 f, err := os.Create(filePath)
 if err != nil {
  log.Log(err)
  return
 }
 defer f.Close()
 f.WriteString("\xEF\xBB\xBF")   //UTF-8
 w := csv.NewWriter(f)
 w.Write([]string{"省份", "城市", "日期", "天气状况", "气温", "风力风向"})
 for i, weatherInfo := range weatherInfos {
  if i%1000 == 0 {
   w.Flush()  //刷入文件
  }
  strs := []string{TrimSpace(weatherInfo.Province), TrimSpace(weatherInfo.City), TrimSpace(weatherInfo.Date),
TrimSpace(weatherInfo.Info), TrimSpace(weatherInfo.Temp), TrimSpace(weatherInfo.Wind)}
  w.Write(strs)
 }
 w.Flush()
 return
}

func TrimSpace(value string) string {
 value = strings.Replace(value, "\n", "", -1)
 return strings.Replace(value, " ", "", -1)
}
```

只去库删掉了，因为看起来有点不是很好，当然，也没有说这个代码好，只是临时写着东西

日志库删掉了，因为看起来有点不是很好。当然，也没有说这个代码好。只是临时写着玩玩。

相关文章

1. 抓取天气
2. python抓取天气
3. 天气预报抓取
4. 抓取天气实例
5. 天气预报抓取天气信息模块实现（代码）
6. Python 抓取中国天气网天气数据
7. 从网上抓取历史天气数据
8. php 抓取天气情况 www.weather.com.cn
9. python练习--天气信息抓取（1）
10. PHP抓取天气预报绝对精简代码【PHP偷取天气预报】

更多相关文章...

相关标签/搜索

抓取天气 天气获取 获取天气 linuxshell 获取天气 php 抓取天猫 天气 今天天气 历年 年历 抓取 抓取 抓取 抓取 抓取 天气 天气 天气 天气 博客抓取 抓取类别 网络爬虫 Go 中华万年历天气api 万年历的天气接口 万年历加天气的 api接口 中华万年历 天气api 中华万年历 天气接口 天气预报api 抓包 iOS 取日历年 jquery获取农历节气 万年历的天气接口怎么用 python抓取天眼查

go map详细使用方法

来福马斯特 + 关注

2017.11.07 21:26* 字数 1126 阅读 9528 评论 0 喜欢 3

go map 比较深入的使用方案

参考blog： https://blog.golang.org/go-maps-in-action

现在基本上所有的编程语言都有自带的map，或者dict，主要提供一个快速的查找，插入，删除，具备与存储体量无关的O（1）的性能，并且支持key上面的唯一性，
比如java里的HashMap，python里的Dictionary，scala里的各种Map等等。

go也原生提供了一个类似的数据类型，就叫做map。首先它是个mutable的，也就是说，可以随时对其进行修改。其次，它不是线程安全的。所以等价于java里的HashMap。

申明和初始化

```
map[KeyType]ValueType
```

这里的KeyType代表map的key类型，一定要是 comparable 的，而ValueType可以是任意的类型，甚至包括其他的内嵌的map
比如

```
var m map[string]int
```

这里的keyType是string，valueType就是int

map在go里是属于reference type，也就是作为方法的型参或者返回类型的是时候，传递也是这个reference的地址。不是map的本体。其次，这个map在申明的时候是nil map，需要如果没有初始化，那么就是nil
对于这个nil的map，可以对其进行任意的取值，返回都是（nil，err），但是如果对其设置一个新的值，就会panic
A nil map behaves like an empty map when reading, but attempts to write to a nil map will cause a runtime panic; don't do that

所以需要先初始化，方法1:

```
m = make(map[string]int)
```

方法二：

```
var m map[string]int = map[string]int{"hunter":12,"tony":10}
```

或者初始化一个空的map

```
m = map[string]int{}
```

读取

```
i := m["route"]
```

如果route存在，就返回那个值，如果不存在，返回0值，也就是说，根据这个value的类型，返回缺省值，比如string，就返回“”，int 就返回0

删除

```
delete(m,"route")
```

如果route存在，删除成功，否则什么都没有发生
因为读取在不存在key的时候返回0值，为了区分是否成功，可以采用如下手段

```
i, ok := m["route"]
```

遍历

```
for key, value := range m {
    fmt.Println("Key:", key, "Value:", value)
}
```

稍微高级点的用法

利用0值，因为当从map中读取一个不存在的key的时候，返回0值，有时候很麻烦，有时候也可以很巧妙的利用起来，参考原文英文中的例子

```
type Node struct {
    Next  *Node
    Value interface{}
}
var first *Node

func main(){
visited := make(map[*Node]bool)
    for n := first; n != nil; n = n.Next {
        if visited[n] {
            fmt.Println("cycle detected")
            break
        }
        visited[n] = true
        fmt.Println(n.Value)
    }
}
```

这是一个检测单向链表是否有环的比较笨的办法，原理就是利用map判断这个key为＊Node的值在map中是否出现过来确定是否有环。
这里的visited就是map，从这里我们可以看到，指针类型也是comparable的，所以可以作为keytype，其次，调用if语句中的visited［n］的时候，我们巧妙的利用了bool类型的0值就是false这个原理，来判断这个keytype是否已经出现。

还是原文中的例子：

```
    type Person struct {
        Name  string
        Likes []string
    }
    var people []*Person

    likes := make(map[string][]*Person)
    for _, p := range people {
        for _, l := range p.Likes {
            likes[l] = append(likes[l], p)
        }
    }

    for _, p := range likes["cheese"] {
        fmt.Println(p.Name, "likes cheese.")
    }
```

我们有一个自定义的struct，Person，里面存了人的名字和他／她的爱好，现在我们要写一个简单的小程序，把所有的people（人员）按照相同兴趣进行分类

我们这里的代码就是利用两个go里的特征，
1, range对于非nil的map，可以进行遍历，但是如果是nil的map（也就是没有初始化的map），默认按照空的map处理，也就是不运行for循环的逻辑代码
2, append支持非nil和nil 的map，都能进行成功的append。这样，就能简化代码

刚才提到map里的keytype必须是comparable的，go的文档里有明确的定义：
The language spec defines this precisely, but in short, comparable types are boolean, numeric, string, pointer, channel, and interface types, and structs or arrays that contain only those types.

Notably absent from the list are slices, maps, and functions;

these types cannot be compared using ==, and may not be used as map keys.

线程安全（goroutine）

前面提到go的map不是线程安全的，因此需要加锁，一般的方法是，定义一个embeded的struct，类似于子类

```
var counter = struct{
    sync.RWMutex
    m map[string]int
}{m: make(map[string]int)}
```

读的时候，调用读锁

```
counter.RLock()
n := counter.m["some_key"]
counter.RUnlock()
fmt.Println("some_key:", n)
```

写的时候，写锁

```
counter.Lock()
counter.m["some_key"]++
counter.Unlock()
```

读取顺序
go的map是hashmap，所以读取遍历的顺序是不保证的，如果业务需要保证key的遍历顺序，建议将key单独保存到一个slice里

```
import "sort"

var m map[int]string
var keys []int
for k := range m {
    keys = append(keys, k)
}
sort.Ints(keys)
for _, k := range keys {
    fmt.Println("Key:", k, "Value:", m[k])
}
```

来福马斯特 ♂
写了 6334 字，被 4 人关注，获得了 6 个喜欢

+ 关注

来福马斯特，走你

喜欢 | 3

更多分享

下载简书 App ▸
随时随地发现和创作内容

被以下专题收入，发现更多相似内容

程序员 go学习

Apache Spark 2.2.0 中文文档 - Spark SQL, DataFrames...

Spark SQL, DataFrames and Datasets Guide Overview SQL Datasets and DataFrames 开始入门 起始点: SparkSession 创建 DataFrames 无类型的Dataset操作 (aka Dat...

片刻_ApacheCN

Go语法指南

出处---Go编程语言 欢迎来到 Go 编程语言指南。本指南涵盖了该语言的大部分重要特性 Go 语言的交互式简介，它分为三节。第一节覆盖了基本语法及数据结构，第二节讨论了方法与接口， 第三节则简单介绍了 Go 的并发原语。每节末尾都有几个练习，你可以对自己的所学进行实践。 ...

Tuberose

如何设计并实现一个线程安全的 Map？(下篇)

在上篇中，我们已经讨论过如何去实现一个 Map 了，并且也讨论了诸多优化点。在下篇中，我们将继续讨论如何实现一个线程安全的 Map。说到线程安全，需要从概念开始说起。 线程安全就是如果你的代码块所在的进程中有多个线程在同时运行，而这些线程可能会同时运行这段代码。如果每次运行...

一缕殇流化隐半边冰霜

Apache Spark 2.2.0 中文文档 - Spark SQL, DataFrames...

Spark SQL, DataFrames and Datasets Guide Overview SQL Datasets and DataFrames 开始入门 起始点:

SparkSession 创建 DataFrames 无类型的Dataset操作 (aka Dat...

Joyyx

[1/2]Clojure入门教程: Clojure – Functional Program...

//Clojure入门教程: Clojure – Functional Programming for the JVM中文版 (sinaapp.com) - 推酷 http://www.tuicool.com/articles/zMbiauJ 作者:xumingming| 可...

葡萄喃喃呓语

侘寂之美

如何打扫满是落叶的庭院？首先用草耙把地清理得一干二净。然后，摇晃其中一棵树，好让少许树叶掉落。这就是wabi-sabi（侘寂）。 侘寂与所谓的“原始艺术”有一些共同特色，像是物体均呈粗糙，不矫饰的状态，..而且都习惯依素材的天然原型来雕塑。 原来wabi-sabi容易让人联想到...

齐滇大圣

体彩排三~福彩3D

本期~分享参考 体彩胆~6~ 福彩胆~6~ 四码复试 ~5678~~1459~ 三码组合打底 680 473 287 872 319 957 265 531 678 809 280 421 842 427 624 964 735 367 245 220 226 909 53...

散文_ae38

今夜我在梦里等你

刚才打开朋友圈，一个养狗的朋友发了个状态，一个小时前，她养了七年的狗狗被车撞死了。 那字里行间掩饰不住的悲愤，那司机撞了狗之后，没有犹豫没有..减速，压死狗的车轮依然飞驰而去。那个司机开始应该有点慌乱，但几秒之后就心安理得了，就是撞了一只狗而已嘛，谁让你主人不管，你跑到公路上来...

微雨随风

对面山上三棵树

对面山上有三棵树。在这里住了快三年，我竟然才知道，还是听旁人说了以后，眯着眼睛仔细观察才辨别出来。 是我眼睛近视的太厉害？确实近视，但我配了..眼镜，看电影或者打扫房间时会戴。这也难怪，这两样都不需要远眺对面的山顶。其它时间，除了带娃，好像就是看手机。 吃完晚饭，我决定推着宝宝...

小龙真棒

我不等你了。

每个夜深人静的夜里 想到你都会想哭 青春已经过了大半 写满的却全是我等你 你还不能回到我身边 现在不能 未来几年也不能 我跟你，就像风走了几万里 早...已不问归期 我只想说 我不等你了 我随你 哪怕颠沛流离 1 最近开始校招，意味着离别已悄悄...

芝麻虽小