

1.9 实践项目—爬取学生信息

1.9.1 项目简介

设计一个 Web 服务器 `server.py`，它读取 `students.txt` 文件中的学生数据，以表格的形式呈现在网页上，其中 `students.txt` 的格式如下：

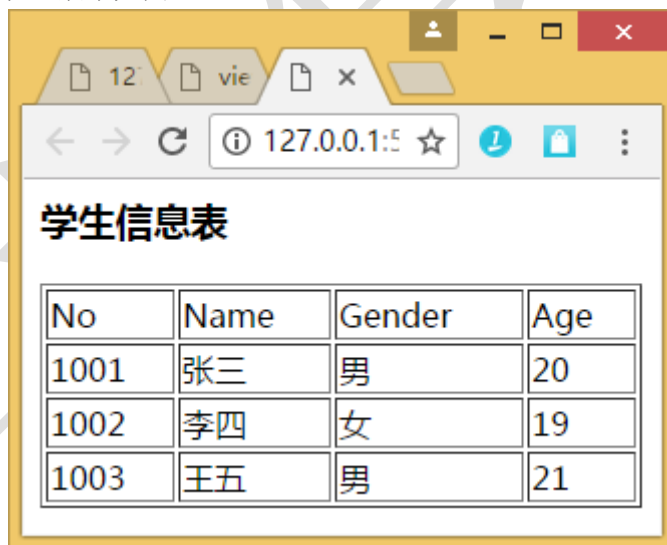
```
No,Name,Gender,Age
1001,张三,男,20
1002,李四,女,19
1003,王五,男,21
```

第一行是学生表格的标题，有学号 `No`、姓名 `Name`、性别 `Gender`、年龄 `Age`，每个学生占一行，各个数据之间用逗号分开。

设计一个客户端的爬虫程序，它从这个网页上爬行学生的这些信息，存储到数据库中。学生数据库可以使用 `Sqlite` 数据库 `students.db`。

1.9.2 服务器程序

服务器程序首先读取同一个目录下的 `students.txt` 文件，然后组成一张 `<table>` 的 HTML 表格用网页的形式呈现，效果如图 1-8-1。



The screenshot shows a web browser window with the address bar displaying `127.0.0.1:5`. The page title is "学生信息表" (Student Information Table). The table contains the following data:

No	Name	Gender	Age
1001	张三	男	20
1002	李四	女	19
1003	王五	男	21

图 1-8-1 学生信息表网页

程序先检查是否有 `students.txt` 文件存在，如果有就打开读取，读出的的一行的数据是用逗号分开的，因此使用 `split(",")` 函数拆分开，然后把一行组织在 `<tr>...</tr>` 的行中，把每个数据组织在 `<td>...</td>` 的单元格中，程序如下：

```
from flask import Flask,request
import os

app=Flask(__name__)

@app.route("/")
def show():
```

```

if os.path.exists("students.txt"):
    st="<h3>学生信息表</h3>"
    st=st+"<table border='1' width='300'>"
    fobj=open("students.txt","rt",encoding="utf-8")
    while True:
        #读取一行，去除行尾部"\n"换行符号
        s=fobj.readline().strip("\n")
        #如果读到文件尾部就退出
        if s=="":
            break
        #按逗号拆分开
        s=s.split(",")
        st=st+"<tr>"
        #把各个数据组织在<td>...</td>的单元中
        for i in range(len(s)):
            st=st+"<td>" + s[i] + "</td>"
        #完成一行
        st=st+"</tr>"
    fobj.close()
    st=st+"</table>"
    return st

if __name__=="__main__":
    app.run()

```

运行服务器程序，默认的网址是 <http://127.0.0.1:5000/>

1.9.3 客户端程序

客户端程序访问 <http://127.0.0.1:5000/>的网址，从中下载其 HTML 网页，这个网页的结果如下：

```

<h3>      学      生      信      息      表      </h3><table      border='1'
width='300'><tr><td>No</td><td>Name</td><td>Gender</td><td>Age</td></tr><tr><td>1001<
/td><td> 张 三 </td><td> 男 </td><td>20</td></tr><tr><td>1002</td><td> 李 四 </td><td> 女
</td><td>19</td></tr><tr><td>1003</td><td>王五</td><td>男</td><td>21</td></tr></table>

```

序要从这个 HTML 网页爬取数据，只要分解出第一行：

```
<tr><td>No</td><td>Name</td><td>Gender</td><td>Age</td></tr>
```

再次分解这一行的<td>...</td>数据，就知道这个表有哪些标题字段，这个表目前有 No、Name、Gender、Age 字段。

接下来再次分解出下一行<tr>...</tr>：

```
<tr><td>1001</td><td>张三</td><td>男</td><td>20</td></tr>
```

再次分解这一行的<td>...</td>数据，得到 No、Name、Gender、Age 的数据依次是"1001"、"张三"、"男"、"20"，把这一行的数据写入对应的数据库即可。

要分解出<tr>...</tr>只要使用 `r"<tr>"`与 `r"</tr>"`的正则表达式即可，先用 `r"<tr>"`匹配 HTML 代码，得到第一个<tr>的位置，再使用 `r"</tr>"`匹配 HTML 字符串，得到第一个</tr>的

位置，取出<tr>...</tr>的数据部分，再次使用 r"<td>"与 r"</td>"的正则表达式分解<td>...</td>的数据。

客户端程序如下：

```
import urllib.request
import re
import sqlite3

def searchWeb(html):
    rows=[]
    #查询第一个<tr>...</tr>行
    m=re.search(r"<tr>",html)
    n=re.search(r"</tr>",html)
    if m!=None and n!=None:
        #跳过第一行的标题
        html=html[n.end():]
    # 查询第二行开始的数据部分
    m=re.search(r"<tr>",html)
    n=re.search(r"</tr>",html)
    while(m!=None and n!=None):
        row=[]
        #start 是<tr>的结束位置
        start=m.end()
        #end 是</tr>的开始位置
        end=n.start()
        #t 是<tr>...</tr>包含的字符串
        t=html[start:end]
        #html[n.end():]是剩余的 html
        html=html[n.end():]
        #查询第一组<td>...</td>
        a=re.search(r"<td>",t)
        b=re.search(r"</td>",t)
        i=0
        while (a!=None and b!=None):
            start=a.end()
            end=b.start()
            #找到一组<td>...</td>的数据
            row.append(t[start:end])
            #t[b.end():]是本行剩余的部分
            t = t[b.end():]
            a = re.search(r"<td>", t)
            b = re.search(r"</td>", t)
        #增加一行数据
        rows.append(row)
        #继续查找下一行<tr>...</tr>
```

```
m = re.search(r"<tr>", html)
n = re.search(r"</tr>", html)
return rows

def saveDB(rows):
    if len(rows)==0:
        #没有数据就返回
        return
    try:
        con = sqlite3.connect("students.db")
        cursor = con.cursor()
        try:
            #如果有 students 表就删除
            cursor.execute("drop table students")
        except:
            pass
        try:
            #建立新的 students 表
            sql = "create table students (No varchar(128) primary key,Name
varchar(128),Gender varchar(128),Age int)"
            cursor.execute(sql)
        except:
            pass

        for row in rows:
            if(len(row)==4):
                #插入一条记录
                sql="insert into students (No,Name,Gender,Age) values (?, ?, ?, ?)"
                try:
                    No=row[0]
                    Name=row[1]
                    Gender=row[2]
                    Age=int(row[3])
                    cursor.execute(sql,(No,Name,Gender,Age))
                except Exception as err:
                    print(err)

            #数据库提交保存
            con.commit()
            con.close()
    except Exception as err:
        print(err);

def showWeb(rows):
    print("Showing data from Web...")
```

```
        for row in rows:
            print(row)

def showDB():
    print("Showing data from DB...")
    try:
        con = sqlite3.connect("students.db")
        cursor = con.cursor()
        #查询数据库记录
        cursor.execute("select * from students")
        rows=cursor.fetchall()
        #显示每条记录
        for row in rows:
            print(row)
        con.close()
    except Exception as err:
        print(err)

try:
    url = "http://127.0.0.1:5000"
    #访问这个网址获取 html
    resp=urllib.request.urlopen(url)
    data=resp.read()
    html=data.decode("utf-8")
    #在 html 中查找学生信息
    rows=searchWeb(html)
    #显示查找的信息
    showWeb(rows)
    #保存学生信息到数据库
    saveDB(rows)
    #显示数据库的数据
    showDB()
except Exception as e:
    print(e)
```

执行后结果中可见客户端从服务器的网页上爬取了学生信息并保存到了数据库:

Showing data from Web...

['1001', '张三', '男', '20']

['1002', '李四', '女', '19']

['1003', '王五', '男', '21']

Showing data from DB...

('1001', '张三', '男', 20)

('1002', '李四', '女', 19)

('1003', '王五', '男', 21)

值得指出的是，通过正则表达式匹配的方法来爬取网页的数据一般会比较麻烦，在后面的章节中我们将介绍更加简单高效的爬取方法。

