**Beautiful Soup简介**

**施嘉伟 180809335**

1. **安装**

pip install beautifulsoup4

pip install lxml

pip install html5lib

1. **导入**

from bs4 import BeautifulSoup

import requests

1. **构造方法**

|  |  |  |  |
| --- | --- | --- | --- |
| **解析器** | **使用方法** | **优点** | **缺点** |
| **Python标准库** | **BeautifulSoup(html, “html.parser”)** | **Python的内置标准库、执行速度适中 、文档容错能力强** | **Python 2.7.3 or 3.2.2)前的版本中文容错能力差** |
| **lxml**  **HTML解析器** | **BeautifulSoup(html, “lxml”)** | **速度快、文档容错能力强** | **需要安装C语言库** |
| **Lxml XML**  **解析器** | **BeautifulSoup(html, “xml”)** | **速度快、唯一支持XML的解析器** | **需要安装C语言库** |
| **Html5lib** | **BeautifulSoup(html, “html5lib”)** | **最好的容错性、以浏览器的方式解析文档、生成HTML5格式的文档** | **速度慢、不依赖外部扩展** |

1. **测试案例**
2. **html = """ <html><head><title>The Dormouse's story</title></head> <body> <p class="title" name="dromouse"><b>The Dormouse's story</b></p> <p class="story">Once upon a time there were three little sisters; and their names were <a href="http://example.com/elsie" class="sister" id="link1"><!-- Elsie --></a>, <a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and <a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>; and they lived at the bottom of a well.</p> <p class="story">...</p> """**
3. **html = """ <html><head><title>The Dormouse's story</title></head> <body> <p class="title" name="dromouse"><b>The Dormouse's story</b></p> <p class="story">Once upon a time there were three little sisters; and their names were <a href="http://example.com/elsie" class="sister" id="link1"><!-- Elsie --></a>, <a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and <a href="http://example.com/tillie" class="sister" id="link3">Tillie</a>; and they lived at the bottom of a well.</p> <p class="story">...</p> """**
4. **html = """ <html> <head> <title>The Dormouse's story</title> </head> <body> <p class="story"> Once upon a time there were three little sisters; and their names were <a href="http://example.com/elsie" class="sister" id="link1"> <span>Elsie</span> </a> <a href="http://example.com/lacie" class="sister" id="link2">Lacie</a> and <a href="http://example.com/tillie" class="sister" id="link3">Tillie</a> and they lived at the bottom of a well. </p> <p class="story">...</p> """**
5. **html=''' <div class="panel"> <div class="panel-heading"> <h4>Hello</h4> </div> <div class="panel-body"> <ul class="list" id="list-1"> <li class="element">Foo</li> <li class="element">Bar</li> <li class="element">Jay</li> </ul> <ul class="list list-small" id="list-2"> <li class="element">Foo</li> <li class="element">Bar</li> </ul> </div> </div> '''**
6. **选择标签（假设soup是已经构造完的Beautiful Soup）**
7. **选择元素**

**soup.title**

**soup.head**

**soup.p**

1. **获取名称**

**soup.title.name**

1. **获取属性**

**soup.p.attrs['name']**

**soup.p['name']**

1. **获取内容**

**soup.p.string**

1. **嵌套选择**

**soup.head.title.string**

1. **子节点和子孙节点**

**soup.p.children（子标签）**

**soup.p.descendants（所有子子孙孙标签）**

**外：list(enumerate())把它转换为list可遍历数组**

1. **父节点和祖父节点**

**soup.a.parent（父标签）**

**soup.a.parents（所有祖先）**

1. **兄弟节点**

**soup.a.next\_siblings**

**soup.a.previous\_siblings**

1. **内置方法**

### **1、find\_all( name , attrs , recursive , text , \*\*kwarg)**

**find\_all返回所有元素,可根据标签名、属性、内容查找文档**

1. **name**

**soup.find\_all('ul')**

**soup.find\_all('ul')[0])**

1. **Attrs**

**soup.find\_all(attrs={'id': 'list-1'})**

**soup.find\_all(attrs={'name': 'elements'})**

1. **Text**

**soup.find\_all(text='Foo')**

### **find(name,attrs,recursive,text,\*\*kwargs)**

**返回单个元素**

**contents获取标签内所有内容**

1. **find\_parents()**

**返回所有祖先节点**

**find\_parent()**

**返回父节点**

1. **find\_next\_siblings()**

**返回后面所有兄弟节点**

**find\_next\_sibling()**

**返回后面第一个兄弟节点**

1. **find\_previous\_siblings()**

**返回前面所有兄弟节点**

**find\_previous\_sibling()**

**返回前面第一个兄弟节点**

1. **find\_all\_next()**

**返回节点后所有符合条件的节点**

**find\_next()**

**返回第一个符合条件的节点**

1. **find\_all\_previous**

**返回节点后所有符合条件的节点**

**find\_previous**

**返回第一个符合条件的节点**

1. **Clear**

**清空所有子标签（保留标签名）**

**soup.clear()**

1. **Decompose**

**递归删除所有标签**

1. **extract**

**递归删除所有标签，并获取标签**

1. **Decode**

**转换为字符串**

**decode\_contents（不含当前标签转换为字符串）**

1. **encode**

**转换为字节**

**encode\_contents（不含当前标签转换为字节）**

1. **has\_attr**

**检查标签是否具有该属性**

1. **get\_text**

**获取标签内部文本内容**

1. **来不及整理的**
2. **CSS选择器（select,select\_one）**
3. **追加标签 (append)**
4. **插入标签（insert、insert\_after,insert\_before）**
5. **替换标签（replace\_with）**
6. **创建标签之间的联系**
7. **包裹标签（wrap、unwrap）**
8. **判断是否为空标签**

**七、案例演示**