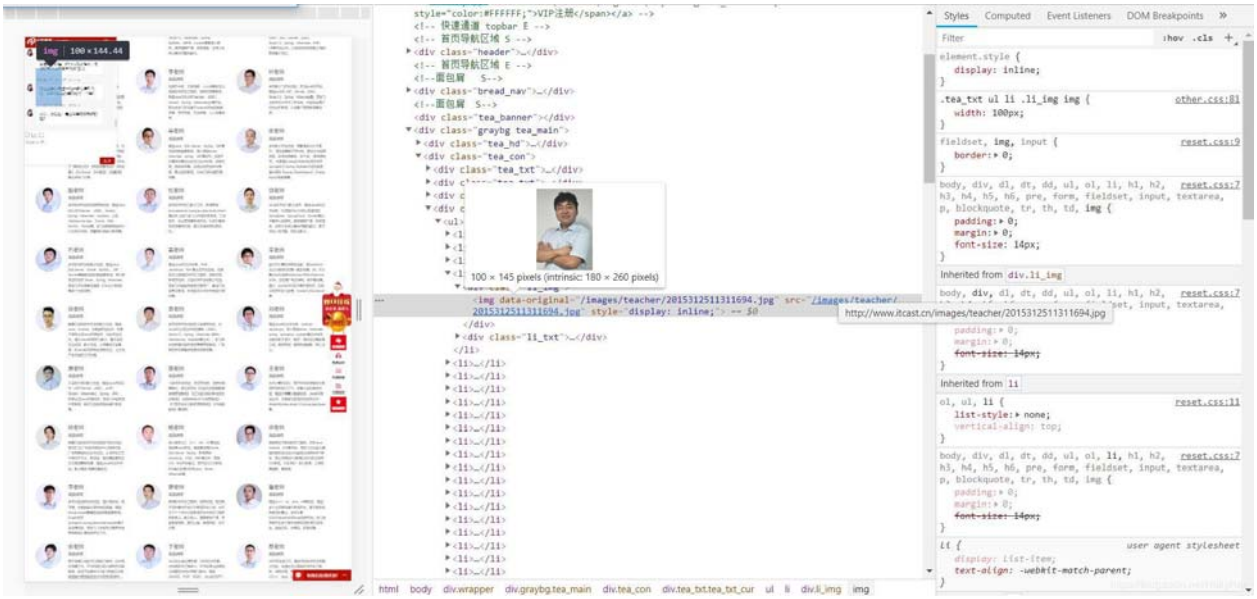
爬取的网址为传播智客 (http://www.itcast.cn/channel/teacher.shtml#ajavaee)

分析网页

审查元素，可以找到图片对应的代码



放大可查看原图

我们看到，图片的网址都是形如 `http://www.itcast.cn/images/teacher/xxxxxxxx.jpg` ，其存放位置在 `img` 标签中，称为 `data-original` 。
知晓了这些，就可以开始爬取图片了。

模块

需要用到的模块为 `requests`、`BeautifulSoup`、`os` 。

爬取图片

首先是基本的发送连接请求。

```
ul = 'http://www.itcast.cn/channel/teacher.shtml#ajavaee'
html = requests.get(ul).text
main_url = 'http://www.itcast.cn'
# print(html)
soup = BeautifulSoup(html, 'lxml')
img_ul = soup.find_all('div', {'class': 'li_img'})
```

第二行的text方法是将原始网页解码为 unicode 格式，方便读取，当然了，也可以后期对网页编码进行更改。
值得注意的是，得到的img_ul类型是ResultSet（结果集），意思是标签，并不是正儿八经的字符串，因此处理的时候需要注意。 **结果集的某个元素：** `<div class="li_img"><img data-original="/images/teacher/20154825094803449.jpg"/></div>` 。

找“肉”

创建一个文件夹保存爬取的图片。 `os.makedirs('./传播智客/', exist_ok=True) # 在当前程序的文件夹下创建一个“传播智客”的文件夹`

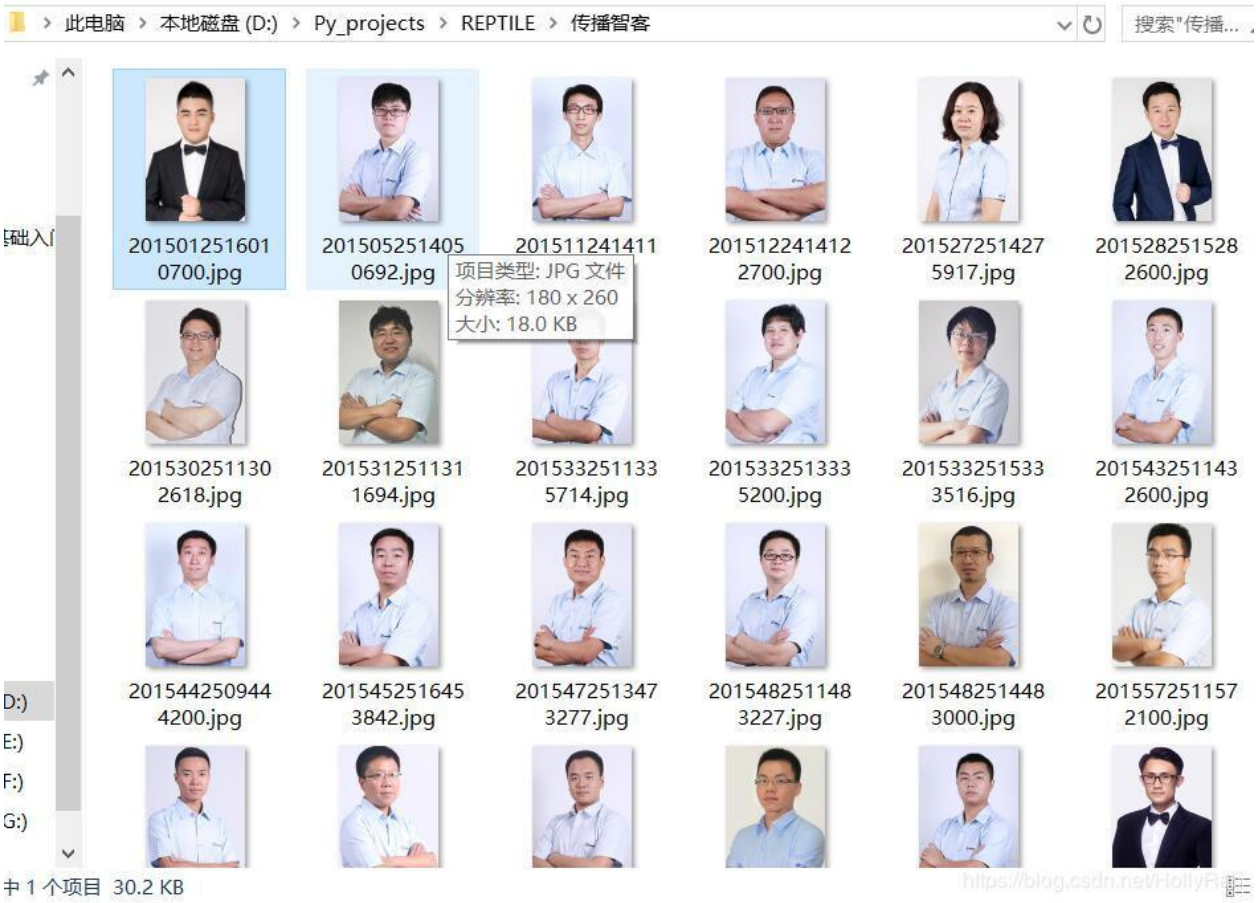
```
for ul in img_ul:
    imgs = ul.find_all('img')
    # print(imgs)
    for img in imgs:
        url = img['data-original']
        img_name = url.split('/')[-1]
        req = requests.get(main_url+url, stream=True)
        with open('./传播智客/%s' % img_name, 'wb') as f:
            for chunk in req.iter_content(chunk_size=128):
                f.write(chunk)
        print('Saved %s' % img_name)
```

因为找的是 `img_ul` 中的 `<div class="li_img">` 里面的元素，找到的 `imgs` 都是形如 `[<img data-original="/images/teacher/20154825094803449.jpg"/>]` 的列表。，因此就解释了 `url = img['data-original']` 这句话，是为了找到图片的源地址。
这里为了简洁，存取的时候使用了最后的一串数字作为图片的名称，因而将网址分割后取了最后一个元素。
注意

```
with open('./传播智客/%s' % img_name, 'wb') as f:
            for chunk in req.iter_content(chunk_size=128):
                f.write(chunk)
```

这是一种比较保险的做法，意思是内存每存满128字节(byte)的数据就将其写入硬盘，而不是一次性将数据载入内存，再一次性写入硬盘，对于本次爬取图片没什么区别，但是如果是一个20G的视频，显然前者的做法更安全一些。

结果展示



共计259张图片。

再单独展示一张

爬取结果展示