django模板与上下文渲染器

一.为什么要是使用模板?

    views.py视图函数是用来写Python代码的，HTML可以被直接硬编码在views.py之中。如下：

```
    import datetime
    def current_time(request):
        now = datetime.datetime.now()
        html = "<html><body>It is now %s.</body></html>" % now
        return HttpResponse(html)
```

    对页面设计进行的任何改变都必须对Python views.py中的代码进行相应的修改，站点网页设计的修改往往比底层Python代码的修改要频繁得多，因此如果可以在不进行Python代码修改的情况下变更设计，那将会方便得多. Python代码编写和HTML设计是两项不同的工作，大多数专业的网站开发环境都将他们分配给不同的人员甚至是不同部门来完成. 设计者和HTML/CSS的编码人员不应该被要求去编辑Python的代码来完成他们的工作. 程序员编写Python代码和设计人员制作模板两项工作同时进行的效率是最高的，远胜于让一个人等待另一个人完成对某个既包含Python又包含HTML的文件的编辑工作，基于这些原因将页面的设计和Python的代码分离开会更干净简洁更容易维护，我们可以使用Django的模板系统(Template System)来实现这种模式.

二. django模板设置

2.1 修改settings.py配置文件指定模板文件路径.

```
    TEMPLATES = [
        {
            'BACKEND': 'django.template.backends.django.DjangoTemplates',
            'DIRS': [os.path.join(BASE_DIR, 'templates')], #指定模板文件路径
            'APP_DIRS': True,
            'OPTIONS': {
                'context_processors': [
                    'django.template.context_processors.debug',
                    'django.template.context_processors.request',
                    'django.contrib.auth.context_processors.auth',
                    'django.contrib.messages.context_processors.messages',
                ],
            },
        },
    ]
```

2.2 Jinjia2语法

2.2.1 读取变量

```
    {{ name }}
```

2.2.2 循环控制语句

```
    {% for item in books %}
        {{item}} <br/>
    {% endfor %}
```

2.2.3 读取字典中的内容

```
    name is  {{ dict.name }}  age is {{ dict.age }}
```

2.2.4 条件判断

```
    {% if age > 90 %}
       优秀
    {% elif age >= 70 %}
       良
    {% elif age >= 60 %}
       中
    {% else %}
       不及格
    {% endif %}
```

三. 开始使用模板

3.1 编写视图函数

```
    def hello(request):
        name = '刘硕的教程'
        books = ['Python学习手册', 'Python核心编程', 'Python CookBook']
        myDict = {"name": 'liushuo', 'age': 20}
        return render(request, 'hello.html', {'name': name, 'books': books,'dict': myDict})
```

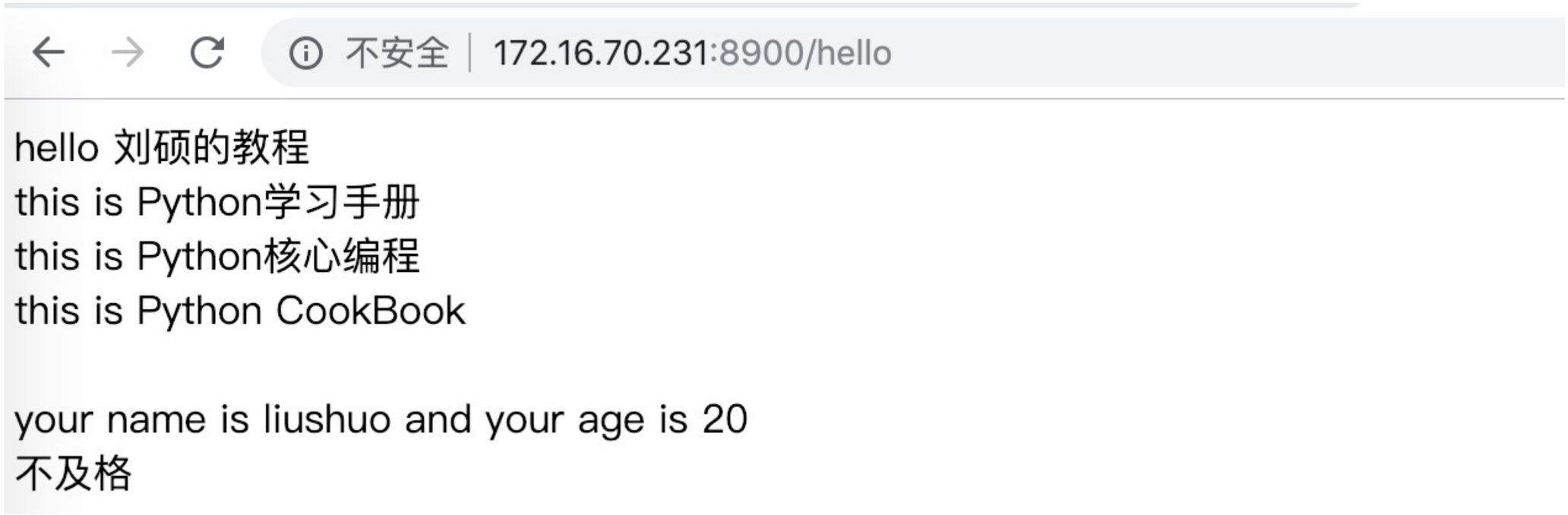
3.2 定义urls与views视图之间的映射关系

```
    from login.views import  hello
    urlpatterns = [
        path('hello', hello),
    ]
```

3.3 编写模板代码

```
    hello {{ name }}<br/>
    {% for item in books %}
       this is {{ item }}<br/>
    {% endfor %}
    <br/>
    your name is {{ dict.name }} and your age is {{ dict.age }}<br/>
    {% if dict.age > 90 %}
      优秀
    {% elif dict.age >= 70 and dict.age <= 80 %}
      良
    {% elif dict.age >= 60 and dict.age < 70 %}
    中
    {% else %}
    不及格
    {% endif %}
```

3.4 测试模板



四.上下文渲染器
我们想让一些内容在多个模板中都要有，可以用上下文渲染器来处理一些公共的变量，来提供给所有模板使用，比如一个网站的每个页面都会显示访问者的IP地址，类似这种需求我们可以通过上下文渲染器来实现.

4.1 开发自定义上下文渲染器代码

在项目配置文件同级目录中创建context_processor.py并写入如下代码：

```
[root@harbor-a webapps]# touch context_processor.py
```

```
def get_remote_addr(request):
    return {'remote_addr': request.META['REMOTE_ADDR']}
```

4.2 在配置文件settings.py中应用自定义的上下文渲染器

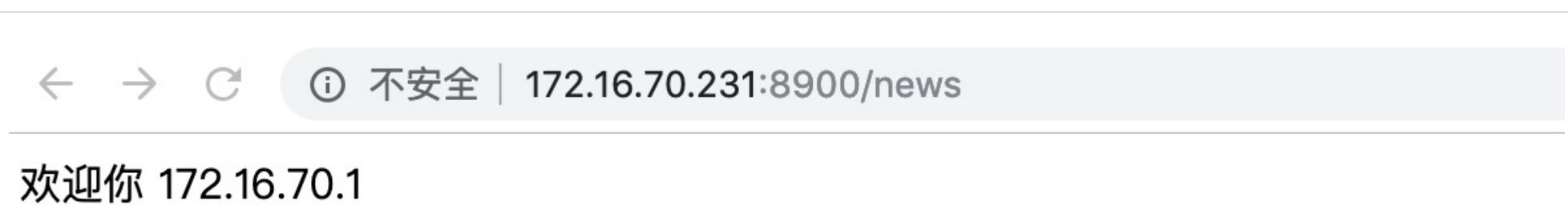
```
TEMPLATES = [
    {
        'BACKEND': 'django.template.backends.django.DjangoTemplates',
        'DIRS': [os.path.join(BASE_DIR, 'templates')],
        'APP_DIRS': True,
        'OPTIONS': {
            'context_processors': [
                'django.template.context_processors.debug',
                'django.template.context_processors.request',
                'django.contrib.auth.context_processors.auth',
                'django.contrib.messages.context_processors.messages',
                'web.context_processor.get_remote_addr', #加入自定义的上下文渲染器
            ],
        },
    },
]
```

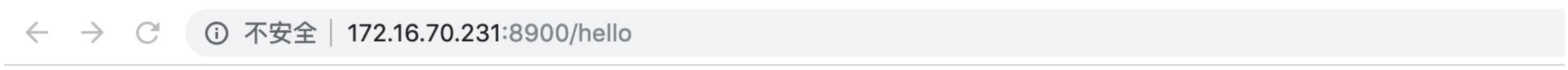
4.3 在模板中使用上下文渲染器中的内容

在templates下需要应用上下文渲染器的模板中加入以下代码：

```
欢迎你 {{ remote_addr }}
```

4.4 查看效果





hello 刘硕的教程
this is Python学习手册
this is Python核心编程
this is Python CookBook

your name is liushuo and your age is 20
不及格 欢迎你 172.16.70.1

In [ ]: