模板习题

1、奇偶行列表输出

使用下面字典my_dict的c的列表，在模板网页中列表ul输出多行数据

- 要求奇偶行颜色不同
- 每行有行号（从1开始）
- 列表中所有数据都增大100

```
from django.http import HttpResponse, HttpRequest
from django.shortcuts import render

def index(request:HttpRequest):
    """视图函数：请求进来返回响应"""
    my_dict = {
        'a':100,
        'b':0,
        'c':list(range(10,20)),
        'd':'abc', 'date':datetime.datetime.now()
    }
    context = {'content':'www.magedu.com', 'my_dict':my_dict}
    return render(request, 'index.html', context)
```

模板代码如下

```
<ul>
{%for line in my_dict.c %}
    <li style='color:{{forloop.counter|divisibleby:"2"|yesno:"red,blue"}}'>
        {{forloop.counter}} {{line|add:"100"}}
    </li>
{%endfor%}
</ul>
```

2、打印九九方阵

```
1*1=1 1*2=2 ... 1*9=9
...
9*1=1 9*2=18... 9*9=81
```

使用把上面所有的数学表达式放到HTML表格对应的格子中

方法一、由视图函数提供数据

为了简单，直接准备一个排好输出顺序的列表

```
from django.http import HttpResponse, HttpRequest
from django.shortcuts import render

def index(request:HttpRequest):
    data = ['{}*{}={}'.format(j, i, j*i) for i in range(1, 10) for j in range(1, 10)]
    return render(request, 'matrix.html', {'data':data})
```

matrix.html模板如下

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>九九方阵</title>
</head>
<body>
<table>
    {% for x in data %}
    {% if forloop.counter0|divisibleby:9 %}<tr>{% endif %}
        <td>{{x}}</td>
    {% if forloop.counter|divisibleby:9 %}</tr>{% endif %}
    {% endfor %}
</table>
</body>
</html>
```

方法二、内建标签 `widthratio`

widthratio 本意是计算宽度比率的。

widthratio 用法：`{% widthratio value max_value max_width %}`

举例
`{% widthratio 175 200 100 %}`
value是175，max_value是200，max_width是100。
因此，175/200=0.875得到数值比值，再利用比率计算出宽度0.875*100=87.5，四舍五入到88

{% widthratio i 1 j as product %} 别名就可以在后面引用这个变量product

```
from django.http import HttpResponse, HttpRequest
from django.shortcuts import render

def index(request:HttpRequest):
    context = {'loop':list(range(1,10))}
    return render(request, 'matrix.html', context)
```

matrix.html模板如下

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>九九方阵</title>
</head>
<body>
九九方阵表格打印--widthratio
<hr>
<table>
    {% for i in loop %}
    <tr style="background-color:{{forloop.counter|divisibleby:2|yesno:'#F0F0F0,#CCC'}}">
        {% for j in loop %}
        <td>
            {{forloop.counter}}*{{forloop.parentloop.counter}}={% widthratio forloop.counter 1
forloop.parentloop.counter %}
        </td>
        {% endfor %}
    </tr>
    {% endfor %}
</table>
<hr>
</body>
</html>
```

使用cycle标签来替换奇偶行变色代码。cycle标签，就是轮询所有其后给出的所有参数。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>九九方阵</title>
</head>
<body>
九九方阵表格打印--widthratio

<hr>
<table>
    {% for i in loop %}
    <tr style="background-color:{% cycle '#CCC' '#F0F0F0' '#FFF' %}">
        {% for j in loop %}
        <td>
            {{forloop.counter}}*{{forloop.parentloop.counter}}={% widthratio forloop.counter 1
forloop.parentloop.counter %}
        </td>
        {% endfor %}
    </tr>
    {% endfor %}
</table>
<hr>
</body>
</html>
```

方法三、自定义filter
1）构建自定义的模板的包和模块

在应用user下构建templatetags包，一定要有`__init__.py`文件。

构建自己的filter的模块，这里起名为`myfilters.py`。其中代码如下

```
from django.template import Library

register = Library()

@register.filter('multiply') # 使用装饰器注册
def multiply(a, b):# 只能1到2个参数
    return int(a)*int(b)
```

2）定义模板
matrix.html模板中要load myfilters模块，使用已经注册的filter multiply，模板内容如下

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>九九方阵</title>
</head>
<body>
九九方阵表格打印
{% load myfilters %}
<hr>
<table>
    {% for i in loop %}
    <tr style="background-color:{{forloop.counter|divisibleby:2|yesno:'#F0F0F0,#CCC'}}">
        {% for j in loop %}
        <td>
            {{forloop.counter}}*{{forloop.parentloop.counter}}=
{{forloop.counter|multiply:forloop.parentloop.counter}}
        </td>
        {% endfor %}
    </tr>
    {% endfor %}
</table>
<hr>
</body>
</html>
```

3）使用模板

```python
from django.http import HttpResponse, HttpRequest
from django.shortcuts import render

def index(request:HttpRequest):
    context = {'loop':list(range(1,10))}
    return render(request, 'matrix.html', context)
```