**基于调度方法的类视图**

**1.** 基于方法的类视图，是根据请求的`method`来执行不同的方法的。如果用户是发送的`get`请求，那么将会执行这个类的`get`方法。如果用户发送的是`post`请求，那么将会执行这个类的`post`方法。其他的method类似，比如`delete`、`put`。

**2.** 这种方式，可以让代码更加简洁。所有和`get`请求相关的代码都放在`get`方法中，所有和`post`请求相关的代码都放在`post`方法中。就不需要跟之前的函数一样，通过`request.method == 'GET'`。

**代码演示：**

|  |
| --- |
| #定义一个基于方法调度的 类视图  **class** LoginView(views.MethodView):  **def** get(self):  **return** render\_template(**'login.html'**)  **def** post(self):  #模拟实现  #拿到前端页面传过来的 账号 和密码 去数据库做查询操作 查询到 (跳转主页面) ,反之跳转到login.html页面并给出错误提示信息  uname = request.form[**'uname'**]  pwd = request.form[**'pwd'**]  **if** uname==**"momo" and** pwd ==**"123"**:  **return** render\_template(**'index.html'**)  **else**:  **return** render\_template(**'login.html'**,error=**"用户名或者密码错误"**)  # 注册类视图  app.add\_url\_rule(**'/login/'**, view\_func=LoginView.as\_view(**'my\_login'**)) |

**改进1：**

|  |
| --- |
| #改进1  **class** LoginView(views.MethodView):  **def** get(self,error=**None**):  **return** render\_template(**'login.html'**,error=error)  **def** post(self):  #模拟实现  #拿到前端页面传过来的 账号 和密码 去数据库做查询操作 查询到 (跳转主页面) ,反之跳转到login.html页面并给出错误提示信息  uname = request.form[**'uname'**]  pwd = request.form[**'pwd'**]  **if** uname==**"momo" and** pwd ==**"123"**:  **return** render\_template(**'index.html'**)  **else**:  **return** self.get(error=**"用户名或者密码错误"**)    # 注册类视图  app.add\_url\_rule(**'/login/'**, view\_func=LoginView.as\_view(**'my\_login'**))  html页面  <**form action="/login/" method="post"**>  <**table**>  <**tr**>  <**td**>账号：</**td**>  <**td**><**input type="text" name="uname"**></**td**>  </**tr**>  <**tr**>  <**td**>密码：</**td**>  <**td**><**input type="password" name="pwd"**></**td**>  </**tr**>  <**tr**>  <**td**></**td**>  <**td**><**input type="submit" value="立即登录"**></**td**>  </**tr**>  <**tr**>  <**td colspan="2"**>  {# <font color="red">{{ error }}</font>#}  {# 优化写法 ：判断 #}  {% **if error** %}  <**font color="red"**>{{ **error** }}</**font**>  {% **endif** %}  </**td**>  </**tr**>  </**table**>  </**form**> |

**改进2:**

|  |
| --- |
| # 改进2 :基于调度方法的类视图 ，通常get()方法处理get请求,post()方法处理post请求，为了便于管理,不推荐post方法和get方法互相调用  **class** LoginView(views.MethodView):  **def** \_\_jump(self,error=**None**):  **return** render\_template(**'login.html'**, error=error)  **def** get(self, error=**None**):  **return** self.\_\_jump()  **def** post(self):  # 模拟实现  #拿到前端页面传过来的 账号 和密码 去数据库做查询操作 查询到 (跳转主页面) ,反之跳转到login.html页面并给出错误提示信息  uname = request.form[**'uname'**]  pwd = request.form[**'pwd'**]  **if** uname == **"momo" and** pwd == **"123"**:  **return** render\_template(**'index.html'**)  **else**:  **return** self.\_\_jump(error=**"用户名或者密码错误"**)  # 注册类视图  app.add\_url\_rule(**'/login/'**, view\_func=LoginView.as\_view(**'my\_login'**)) |