**Flask\_before\_first\_request和before\_request详解：**

**1.before\_first\_request：**处理项目的第一次请求之前执行。

|  |
| --- |
| **from** flask **import** Flask,request,session,current\_app,url\_for,g  **import** os  app = Flask(\_\_name\_\_)  @app.route(**'/'**)  **def** hello\_world():  print(**"hi"**)  **return "hello world "**  @app.before\_first\_request  **def** first\_request():  print(**'hello world'**)  **if** \_\_name\_\_ == **'\_\_main\_\_'**:  app.run(debug=**True**) |

**2.before\_request：**在每次请求之前执行。通常可以用这个装饰器来给视图函数增加一些变量。

请求已经到达了Flask，但是还没有进入到具体的视图函数之前调用。一般这个就是在视图函数之前，

我们可以把一些后面需要用到的数据先处理好，方便视图函数使用。

|  |
| --- |
| **from** flask **import** Flask,request,session,current\_app,url\_for,g  **import** os  app = Flask(\_\_name\_\_)  app.config[**'SECRET\_KEY'**]=os.urandom(24) *#加盐 混淆原数据的作用*  @app.route(**'/'**)  **def** hello\_world():  print(**"hi"**)  session[**'uname'**]=**"momo"**  **return "hello world "**  @app.route(**'/li'**)  **def** mylist():  print(**"mylist"**)  *# print("直接取出",g.user)*  **if** hasattr(g,**"user"**):  print(**"条件取出"**, g.user)  **return "hello world "**  @app.before\_request  **def** before\_request():  *# print('在视图函数执行之前执行的钩子函数')*  *# 场景：若用户已经登录了，验证时把用户名放入session中,之后取出来，放入钩子函数，以后访问的视图函数中可直接取出来使用*  uname = session.get(**'uname'**)  print(uname)  **if** uname:  g.user = uname  **if** \_\_name\_\_ == **'\_\_main\_\_'**:  app.run(debug=**True**) |