Locust框架

安装

安装：pip install locust
验证安装：locust -V

装饰器

TaskSet:模拟用户操作类继承
class ApiDemo(TaskSet)

HttpUser:模拟用户类继承
例：class AddUser(HttpUser)

wait_time方法用于模拟用户在执行任务之间应等待多长时间
wait_time=between(3,5)

@task：函数执行权重，数值越大,执行的频率越高

on_start和on_stop方法：在继承TaskSet的用户类中，用户开始和结束时执行该方法

host:主机属性，用于指定要测试的地址,继承HttpUser类中进行指定
host="http://localhost:8080"

get请求

```
import os
from locust import HttpUser,task,between,TaskSet
class ApiDemo(TaskSet):
    wait_time = between(1, 2)
    @task(1)
    #get类型接口
    def getinfo(self):
        self.client.get("/pinter/com/getSku?id=1")
class AddUser(HttpUser):
    tasks=[ApiDemo]
    host="http://localhost:8080"
if __name__ == '__main__':
    os.system("locust -f Study.py --host=http://localhost:8080 --web-host=127.0.0.1")
```

K=V类型post请求

```
import os
from locust import HttpUser,task,between
from locust import TaskSet
class ApiDemo(TaskSet):
    wait_time = between(1, 2)
    @task(2)
    #k=v类型post接口
    def post1(self):
        self.client.post("/pinter/com/login",data={"userName":"admin","password":"1234"})
class AddUser(HttpUser):
    tasks=[ApiDemo]
    host="http://localhost:8080"
if __name__ == '__main__':
    os.system("locust -f Study.py --host=http://localhost:8080 --web-host=127.0.0.1")
```

json类型post请求

```
import os
from locust import HttpUser,TaskSet,between,task
class ApiDemo(TaskSet):
    wait_time = between(1, 2)
    @task(2)
    #json类型post接口
    def post2(self):
        self.client.post("/pinter/com/register",json={"userName":"test","pwd":"1234"})
class AddUser(HttpUser):
    tasks=[ApiDemo]
    host="http://localhost:8080"
if __name__ == '__main__':
    os.system("locust -f Study.py --host=http://localhost:8080 --web-host=127.0.0.1")
```

K=json类型post请求

```
from locust import HttpUser,task,between,TaskSet
class ApiDemo(TaskSet):
    wait_time = between(1, 2)
    @task(2)
    #k=json混合类型post请求
    def post3(self):
        self.client.post("/pinter/com/buy",data={"param":{"skuId":123,"num":10}})
class AddUser(HttpUser):
    tasks=[ApiDemo]
    host="http://localhost:8080"
if __name__ == '__main__':
    os.system("locust -f Study.py --host=http://localhost:8080 --web-host=127.0.0.1")
```

设置请求头

```
from locust import HttpUser,task,between,TaskSet
class ApiDemo(TaskSet):
    wait_time = between(1, 2)
    @task(2)
    #设置请求头
    def post2(self):
        header={"content-type":"application"}
        self.client.post("/pinter/com/register",json={"userName":"test","password":"1234"},headers=header)
class AddUser(HttpUser):
    tasks=[ApiDemo]
    host="http://localhost:8080"
if __name__ == '__main__':
    os.system("locust -f Study.py --host=http://localhost:8080 --web-host=127.0.0.1")
```

断言

```
import os
import random
from locust import HttpUser,task,between,TaskSet
class ApiDemo(TaskSet):
    wait_time = between(1, 2)
    @task(1)
    #get类型接口
    def getinfo(self):
        id=random.randint(1,100)
        with self.client.get("/pinter/com/getSku",params={"id":id},catch_response=True) as respon:
            if respon.json()["code"]=="0":
                respon.success()
            else:
                respon.failure("断言失败")
class AddUser(HttpUser):
    tasks=[ApiDemo]
    host="http://localhost:8080"
if __name__ == '__main__':
    os.system("locust -f Study.py --host=http://localhost:8080 --web-host=127.0.0.1")
```

参数化

```
import os
from locust import HttpUser,TaskSet,between,task
class ApiLogin(TaskSet):
    wait_time = between(1,2)
    #用python代码读取参数化文件
    def get_data(self):
        userdata=[]
        with open("./userinfo.txt", encoding="utf-8")as f:
            result = f.readlines()
        for i in result:
            before = i.split(",")
            userinfo = {"username": before[1], "password": before[2], "message": before[3].strip("\n")}
            userdata=userdata.append(userinfo)
        return  userdata
    def on_start(self):
        self.get_data()
    @task(3)
    def login(self):
        login_url="/pinter/com/login"
        for i in range(len(self.on_start())):
            response=self.client.post(url=login_url,data={"userName":self.on_start()[i]["username"],"password":self.on_start()[i]["password"]})
            result=response.json()
        #断言操作
            if result["message"]==self.on_start()[i]["msg"]:
                print("断言成功")
            else:
                print("断言失败")
class AddUser(HttpUser):
    tasks=[ApiLogin]
    wait_time = between(1,2)
    host="http://127.0.0.1:8080"
if __name__ == '__main__':
    os.system("locust -f Demo2.py --host=http://127.0.0.1:8080 --web-host=127.0.0.1")
```

参数化还可以使用lxml直接提取网页中信息，或者队列方式

cookie关联

```
import os
from locust import HttpUser,TaskSet,between,task
#cookie关联
class ApiQuery(TaskSet):
    @task
    def login(self):
        resp=self.client.post("/pinter/bank/api/login",data={"userName":"admin","password":1234})
        print(resp.json())
    @task
    def query_cookie(self):
        res=self.client.get("/pinter/bank/api/query",params={"userName":"admin"})
        print(res.json())
    def on_start(self):
        self.login()
class AddUser(HttpUser):
    tasks=[ApiQuery]
    wait_time = between(1,2)
    host="http://127.0.0.1:8080"
if __name__ == '__main__':
    os.system("locust -f Demo3.py --host=http://127.0.0.1:8080 --web-host=127.0.0.1")
```

locust发起请求采取requests.session()，即自动处理cookie

token关联

```
import os
from locust import HttpUser,TaskSet,between,task
#token关联
class ApiQuery(TaskSet):
    @task
    def login(self):
        resp=self.client.post("/pinter/bank/api/login2",data={"userName":"admin","password":1234})
        login_token=resp.json()["data"]
        return login_token
    @task
    def query_cookie(self):
        header = {"testfan-token": self.on_start()}
        res=self.client.get("/pinter/bank/api/query2",params={"userName":"admin"},headers=header)
        print(res.json())
    def on_start(self):
        self.login()
class AddUser(HttpUser):
    tasks=[ApiQuery]
    wait_time = between(1,2)
    host="http://127.0.0.1:8080"
if __name__ == '__main__':
    os.system("locust -f Demo3.py --host=http://127.0.0.1:8080 --web-host=127.0.0.1")
```

token关联

分布式压线

主机(控制机) --master
启动主机
locust -f 待测脚本 --master

从机(负载机) --worker
启动从机
locust -f 待测脚本 --worker --master-host=主机IP地址

注意：分布式压测时间，主计算机和每个从计算机都必须具有测试脚本
建议启动模拟用户数量要大于locust类的数量*从机数量

no-web模式

命令行运行
locust -f 待测脚本 --host=服务器ip --headless -u 10 -r 2 -t 1m
--headless：命令行运行
-u：设置虚拟用户数
-r：设置每秒启动虚拟用户数
-t：设置运行时间