1. 接口用例

思路：

正例：输入正常入参，接口能够返回数据。

反例：

鉴权反例：鉴权码为空，鉴权码错误，鉴权码过期

参数反例：参数为空，参数类型异常，参数长度异常

错误码覆盖：根据业务而定

其它：接口黑名单，接口调用次数，分页场景

1. Postman

2.1、Workspaces 工作空间

Collections：集合，项目集合

Apis：api文档

Environment：环境变量，全局变量

Mock server：虚拟服务器

History：历史记录

2.2、请求页签

Params：get请求传参

Anthorization：鉴权

Headers：请求头

Body：请求体

Form-data：既可以传键值对参数也可以传文件

X-www-from-urlencoded：只能够传键值对参数

Raw：json，text，xml，html，javascript

Binary：把文件以二进制的方式传参

Pre-request-script：请求之前的脚本

Tests：请求之后的断言

Cookies：管理cookie信息

2.3、响应页签

Body：

Pretty：可多种格式显示数据

Raw：以文本的方式查看返回的数据

Preview：以网页的方式查看返回的数据

Headers：响应头

Testresults：断言的结果

2.4、环境变量与全局变量：

环境变量：环境变量就是全局变量

全局变量：全局变量是能够在任何接口里面访问的变量

获取环境变量和全局变量的值通过：{{变量名}}

2.5、接口关联

a、使用**json提取器**实现接口关联

第一个接口：tests内实现

console.log(responseBody);

//使用json提取器提取accesss\_token值

//把返回的字符串格式的数据转换成对象的形式

var result = JSON.parse(responseBody);

console.log(result.access\_token);

//把access\_token设置成全局变量

pm.globals.set(“access\_token”,result.access\_token);

第二个接口：

{{access\_token}}

b、使用**正则表达式提取器**实现接口关联

第一个接口：

//使用正则表达式提取器实现接口关联，match匹配

var result = responseBody.match(new RegExp(‘”access\_token”:”(.\*?)”’));

console.log(result)

//设为全局变量

pm.globals.set(“access\_token”,result[1]);

第二个接口：

{{access\_token}}

2.6、内置动态参数及自定义的动态参数

内置动态参数：

{{$timestamp}} 生成当前时间的时间戳

{{$randomInt}} 生成0-1000之间的随机数

{{$guid}} 生成速记GUID字符串

自定义动态参数：可在pre-req中实现

//手动获取时间戳

var times = Date.now();

//设置为全局变量

pm.globals.set(“times”,times);

使用时使用：{{times}}

2.7、断言（tests中）

**Status code：Code is 200** 检查返回的状态码是否为200

**Response body：Contains string** 检查响应中包含指定字符串

**Response body：Json value check** 检查响应中其中的json值

**Response body：is equal to a string** 检查响应等于一个字符串

Response headers：Content-Type 检查是否包含响应头Content-Type

Response time is less than 200ms 检查请求耗时小于200ms

在断言中获取自定义动态参数的方式：

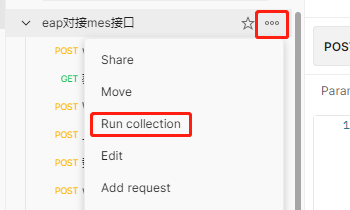
pm.globais.get(“times”)

globals[“times”]

globals.times

2.8、批跑

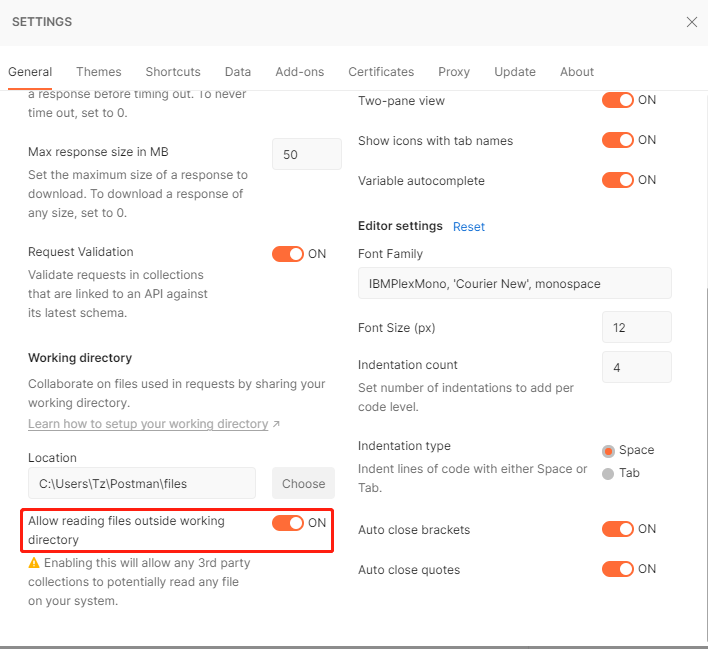
Collections中选择集合中Run collection



Iterations：批跑重复次数

Delay：每个用例之间的延迟时间

Data：可选择文件入参，需在设置中将文件读取按钮打开，数据驱动：csv文件和json文件



2.9、请求头

常见请求头：

Host：请求主机地址

Connection：连接方式

Accept：客户端接收到的数据格式

X-Requested-With：异步请求

User-Agent：客户端的用户类型

Referer：来源

Cookie：Cookie信息

Content-Type：请求内容的格式

2.10、cookie鉴权

Cookie：一小段文本，格式：key=value

Cookie鉴权原理：

1. 当客户端第一次访问服务器的时候，那么服务器就会生成cookie信息，并且在相应头的set-cookie里面把生成的cookie信息发送给客户端
2. 当客户端第2-N次访问服务器的时候，那么客户端就会在请求头的cookie带上cookie信息，从而实现鉴权

Cookie的分类：

会话cookie：保存在内存，当浏览器关闭之后就会自动化清除cookie

持久cookie：保存在硬盘，浏览器关闭之后不会清除，只有当失效时间到了才会自动清除