1.接口以及项目背景简介

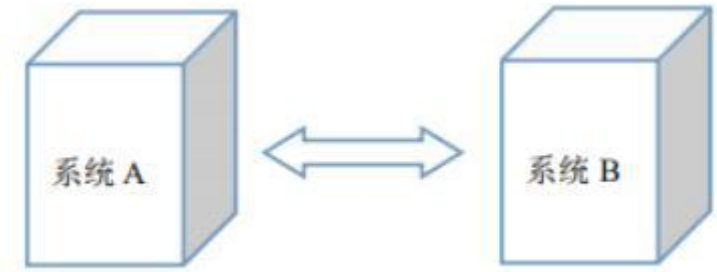
1.1 接口测试定义

接口测试是测试系统组件间接口的一种测试。接口测试主要用于检测外部系统与系统之间以及内部各个子系统之间的交互点。测试的重点是要检查数据的交换，传递和控制管理过程，以及系统间的相互逻辑依赖关系等。

1.2 哪些场景属于接口测试

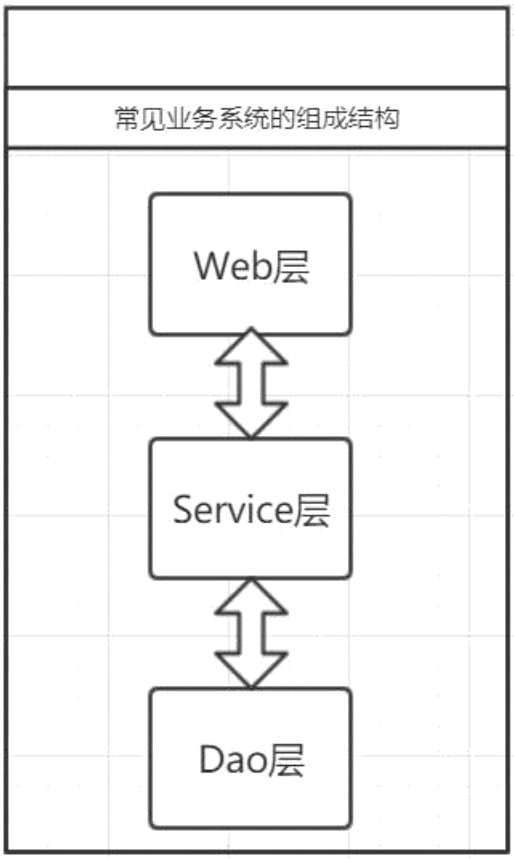
那么在我们的实际工作中都有哪些情况属于接口测试呢？大致分为以下三类：

【1】独立系统 A 与独立系统 B 之间的数据交互过程



例如：如微信，微博所提供的第三方登录接口，是不同公司之间系统接口的调用；

【2】一个系统之间的不同层，我们知道常见的 java-web 项目都是分为：Web 层、dao-数据层以及 service-服务层。这三层之间就是通过接口进行数据交互；



【3】系统内，服务与服务之间的调用

也就是各个模块之间的相互调用，类似于如下情况：

1.3 示例接口展示

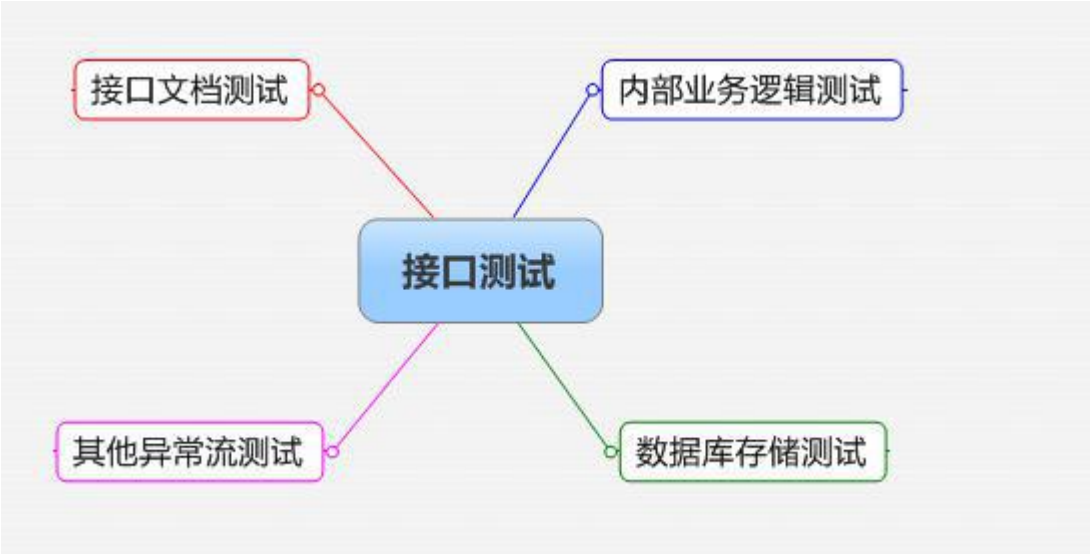
如下是我们最常见的一个接口的定义说明：

NO.	字段代码	字段名称	数据类型	是否必输	长度控制	说明/代码
	id	账户 ID	Integer	N	10	每次唯一
	userName	账户名称	Str	Y	20	不允许包含特殊

			ing			字符
	accountBalance	账户余额	String	Y	20	保留两位小数 如：100.01
	accountType	账户类型	String	Y	20	只允许传如下两个值： 01-优质客户 02-普通客户

1.4 针对上述接口我们需要关注的测试点

关于接口的测试点，大的方向可以概况为以下模型：

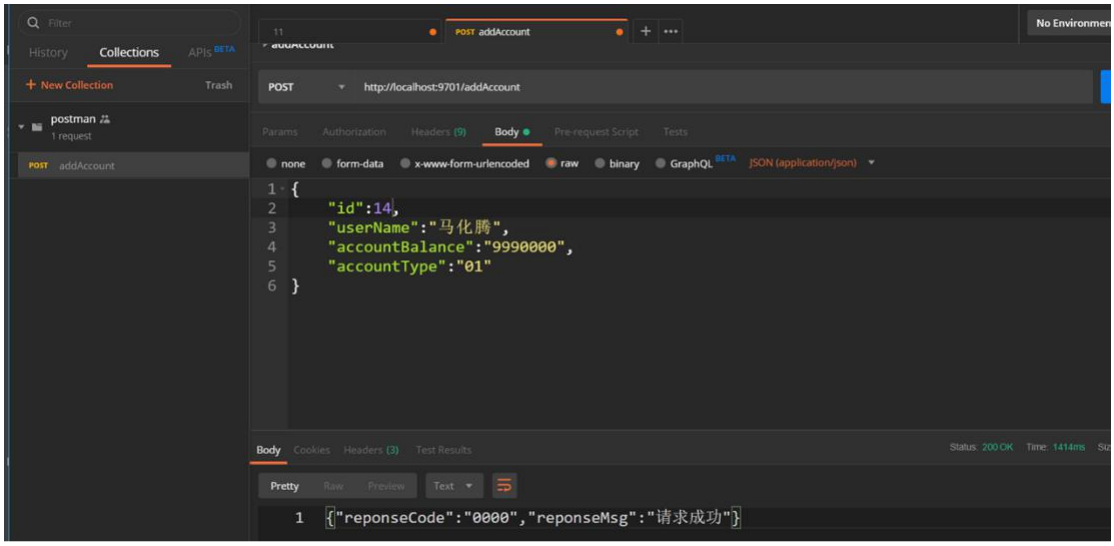


详细可观察我的博文：

https://www.cnblogs.com/haibaowang/articles/7485880.html

2. postman 在功能测试里的应用

有了上述第一部分的铺垫，那我们看下 postman 针对接口在功能测试里如何应用：

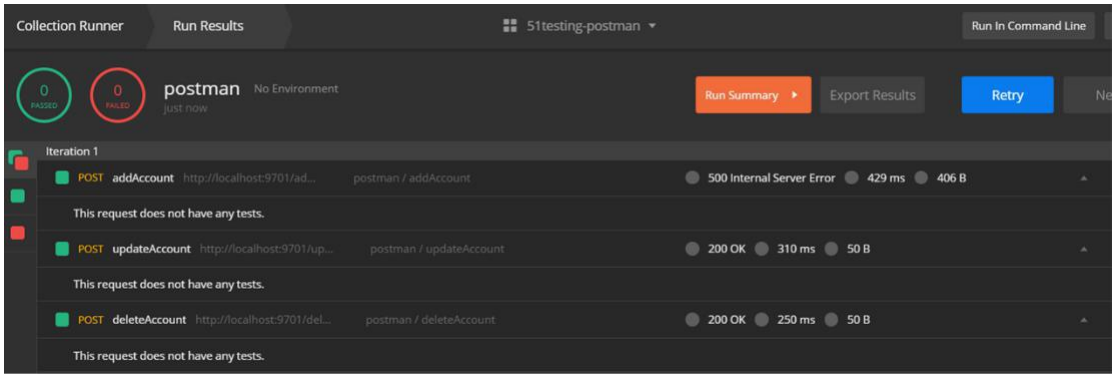


通过控制一个一个字段的输入，来实现每一个字段的校验、测试；

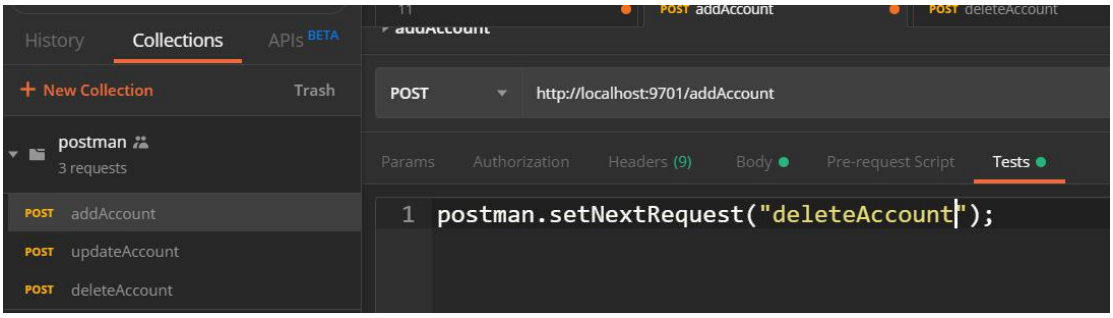
3. postman 在自动化测试里的应用

3.1 指定接口执行顺序与执行次数

Postman 可以通过 JS 脚本，指定 Runner 中各个接口间的调用顺序；例如下图中所示有三个 HTTP 接口，正常情况下，应该先调用接口 addAccount，然后调用 updateAccount，最后调用接口 deleteAccount。

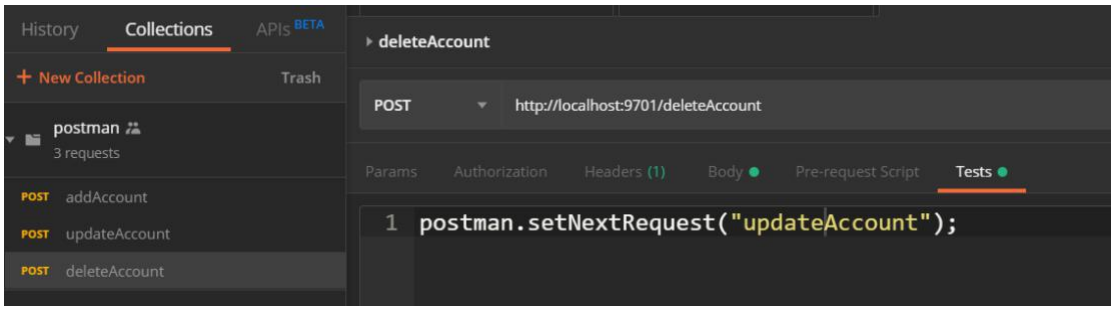


此时我们按照如下步骤，即可改变接口上述接口执行顺序，叫其按照 addAccount-deleteAccoun-updateAccount 的逻辑去执行。【1】首先在 addAccount 接口的 Tests 处添加的代码如下：

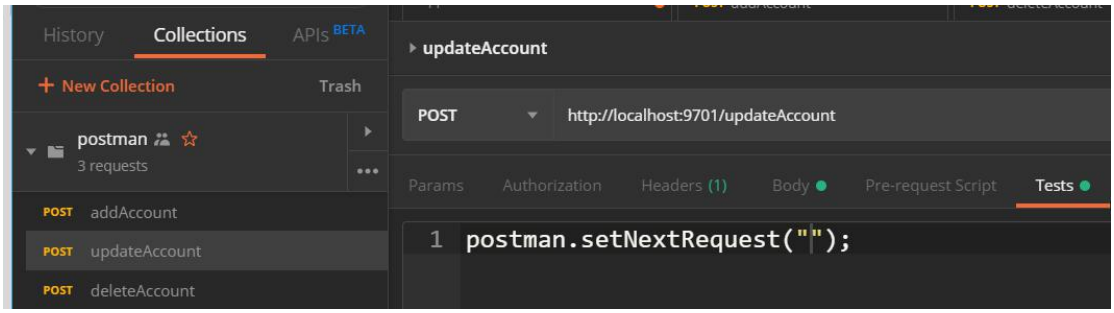


表示执行完 addAccount 接口之后，即跳转到 deleteAccount 接口；

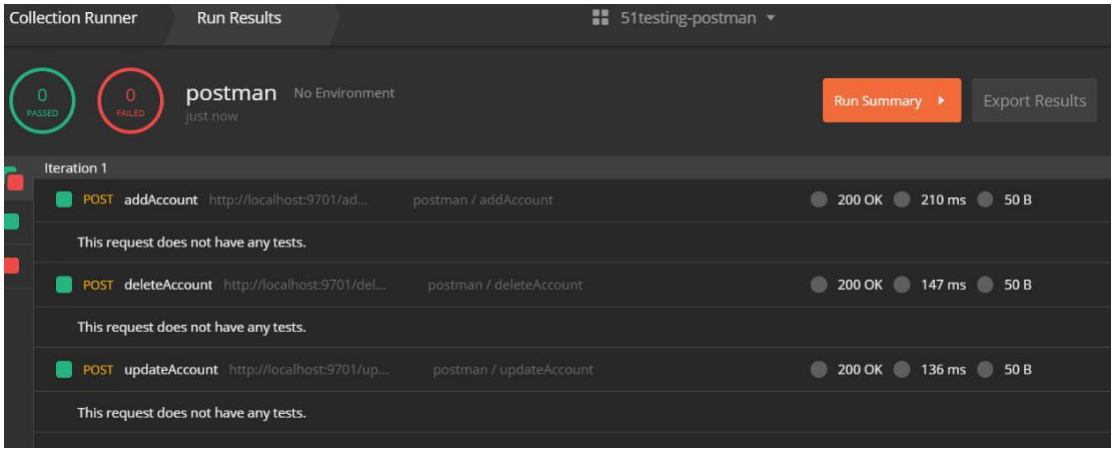
【2】然后在 deleteAccount 接口的 Tests 处添加的代码如下：



【3】 最后，在 updateAccount 接口的 Tests 处将下个执行的接口执行如下代码，这行代码必须要有，如果没有，Runner 将会一直按照 addAccount-deleteAccoun-updateAccount 的调用循序无限循环下去。



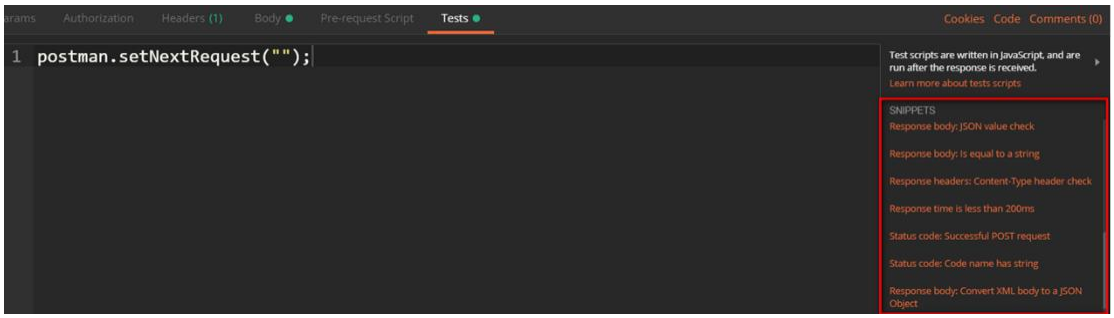
此时执行结果如下所示：



3.2 常用的 JS 脚本设置检查点

我们在上一章节，实现了 addAccount、deleteAccoun、updateAccount 三个接口的自动调用，可是现在看来是很空洞的，缺乏必要的测试点（即检查点）。

检查点也是在接口的 Tests 里进行设置，那么 Postman 支持的检查点有哪些？这个 Postman 自身已经进行了集成，不用我们自己书写。



最常见的检查点如下：

【1】设置环境变量

```
pm.environment.set("variable_key", "variable_value");
```

【2】检查 HTTP 请求的状态码是否与预期一致

```
pm.test("Status code is 200", function () {

    pm.response.to.have.status(200);

});
```

【3】 检查 JSON 响应中包含某个字段

```
pm.test("Your test name", function () {

    var jsonData = pm.response.json();

    pm.expect(jsonData.value).to.eql("字段名称");

});
```

【4】检查响应体中可以解析到某个指定字段的值

```
pm.test("Body matches string", function () {

    pm.expect(pm.response.text()).to.include("string_you_want_to_search");

});
```

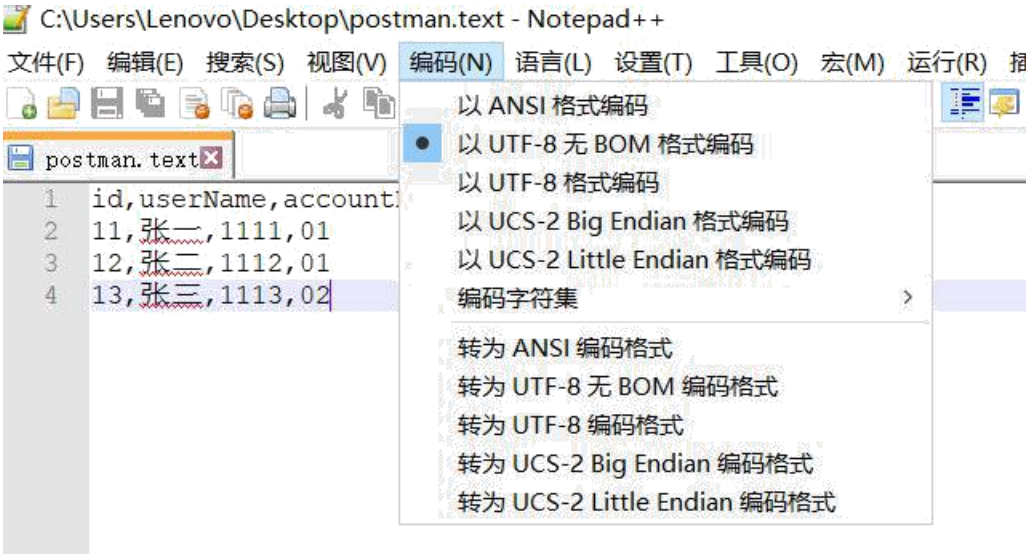
【5】检查响应时间的长短

```
pm.test("Response time is less than 200ms", function () {

    pm.expect(pm.response.responseTime).to.be.below(200);

});
```

3.3 请求脚本参数化的使用

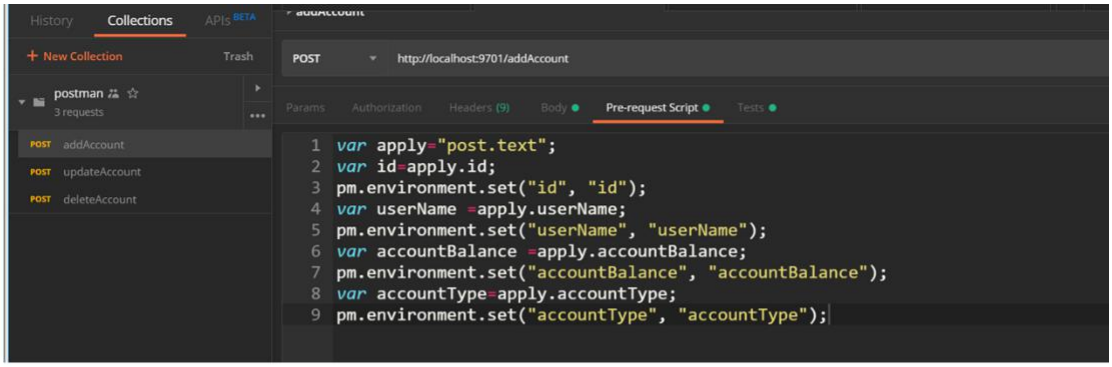
有些时候我们需要对请求中的某个值进行参数化迭代，此时可以将参数化列表写入 text 文档中，这里不推荐大家使用 CSV 与 JSON 导入的形式，不推荐 CSV 是因为每次变更，都要对其格式进行更新；不推荐 JSON 是因为其组织过程过于繁琐。

【1】例如我们对 addAccount 接口的“id”、”userName”、“accountBalance”、“accountType”字段进行参数化。参数化文件如下：



首先文件后缀名必须为 text，第一行必须为参数名称，文本内容中的标点符号必须是英文，另外文本编码格式必须为 UTF-8。最主要一点，最后一行建议用你自己喜欢的符号或者字母将其进行填充。例如我的最后一行是两个 -，其作用是为了防止 Runner 执行参数化迭代的时候，因结尾所加的换行符号导致迭代失败。

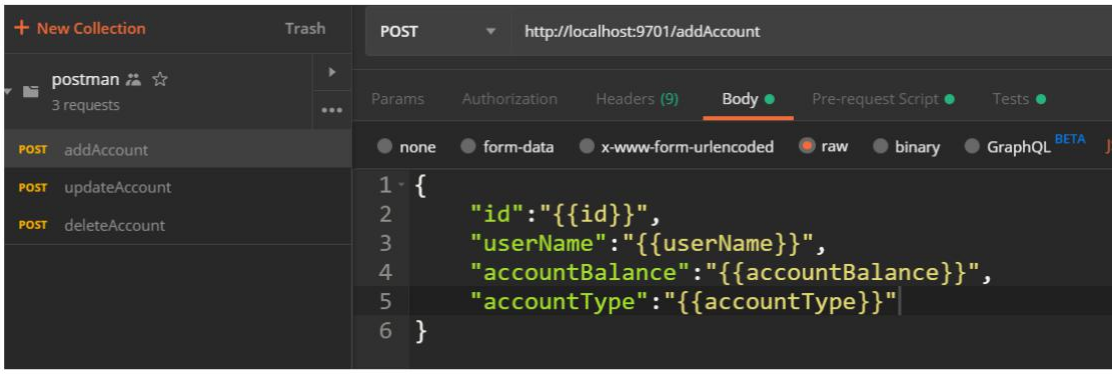
【2】在 addAccount 接口的 Pre-request Script 处写入如下脚本，读取 text 参数化文件 post.text;




```
var apply="post.text";

var id=apply.id;

pm.environment.set("id", "id");

var userName =apply.userName;
```

```
pm.environment.set("userName", "userName");

var accountBalance =apply.accountBalance;

pm.environment.set("accountBalance", "accountBalance");

var accountType=apply.accountType;

pm.environment.set("accountType", "accountType");
```

【3】将请求报文中的参数进行替换，替换方式 {{参数名称}}。



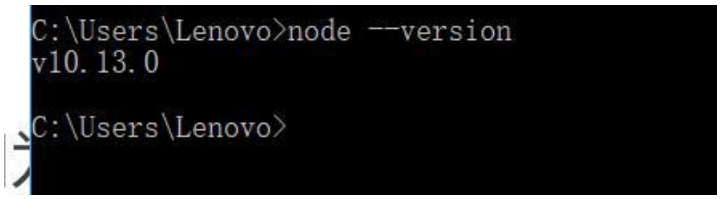
【4】在 Runner 中设置迭代次数为 3（这个次数看我们要迭代几次，迭代几次就配置几次），此时执行 Runner；

3.4 命令行自动化工具 NewMan

NewMan 是官方提供的专门用于 posman 进行自动化的命令行工具。

3.4.1NewMan 的安装

NewMan 是基于 Node.js，所以要想安装 NewMan，必须保证本地已经安装 Node.js；确定是否安装 Node.js。只需要在 cmd 下执行 node --version 即可；

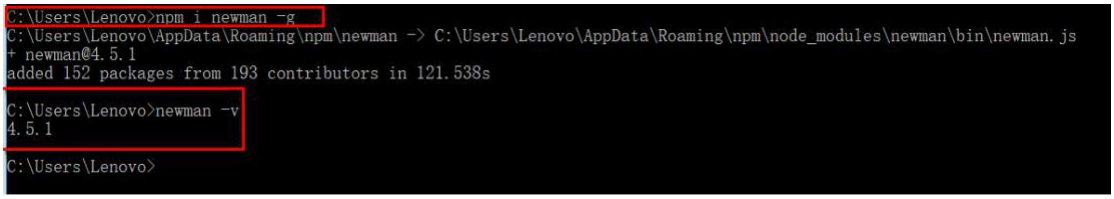


显示出版本号，即可证明 node.js 环境正常。

安装 Neweman 使用以下命令：

npm i newman -g

安装成功，输入 newman -v 来检查版本，显示版本即表示安装成功；



3.4.2NewMan 功能介绍

NewMan 命令工具的主要命令是 newman run。格式如下：

> run [options] <collection>

这里的 Collection 可以是从 Postman 中导出的 .Collection 文件，也可以是通过 Postman 账号 分享出来的在线集合的 URL 地址。

如果采用导出的集合：那么选择的集合版本为：Collection 2.1 版本。

下面介绍几个 newman 运行时常用的参数：

-e, --environment \

指定 environment 环境变量，这里环境变量需要导出为文件(json 格式)

-g, --globals \

指定 global 环境变量，这里环境变量需要导出为文件(json 格式)

-d, --iteration-data \

指定运行 Collection 时使用的 data file 文件

-n, --iteration-count \

指定 Collection run 时的迭代次数 n

-r, --reporters [reporters]

指定运行完成后的结果报告类型，除默认命令行格式 cli 外，还支持 json、junit、progress 以及 emojitrain 几种类型。

3.4.3 通过 Newman 运行自动化脚本

如果只是运行脚本：在 cmd 下执行 newman run .json 即可；

```
D:\createFile>newman run postman.postman_collection.json
newman

postman

→ addAccount
  POST http://localhost:9701/addAccount  [200 OK, 236B, 246ms]

Attempting to set next request to deleteAccount

→ deleteAccount
  POST http://localhost:9701/deleteAccount  [200 OK, 166B, 112ms]

Attempting to set next request to updateAccount

→ updateAccount
  POST http://localhost:9701/updateAccount  [200 OK, 166B, 110ms]

┌─────────────────────────┬──────────┬──────────┐
│                         │ executed │   failed │
├─────────────────────────┼──────────┼──────────┤
│              iterations │        1 │        0 │
├─────────────────────────┼──────────┼──────────┤
│                requests │        3 │        0 │
├─────────────────────────┼──────────┼──────────┤
│            test-scripts │        3 │        0 │
├─────────────────────────┼──────────┼──────────┤
│      prerequest-scripts │        1 │        0 │
├─────────────────────────┼──────────┼──────────┤
│              assertions │        0 │        0 │
├─────────────────────────┴──────────┴──────────┤
│ total run duration: 707ms                     │
├───────────────────────────────────────────────┤
│ total data received: 219B (approx)            │
├───────────────────────────────────────────────┤
│ average response time: 156ms [min: 110ms, max: 246ms, s.d.: 63ms] │
└───────────────────────────────────────────────┘

D:\createFile>
```

如果要在指定环境下运行脚本：此时就是：

```
newman run postman.postman_collection.json -e test.json
```

3.4.4 生成 HTML 格式的测试报告

我们的脚本运行完成，肯定希望生成一个测试报告。

需要先安装一个 html 格式报告的插件，安装命令如下：、 npm install -g newman-reporter-html

```
D:\createFile>npm install -g newman-reporter-html
npm WARN newman-reporter-html@1.0.3 requires a peer of newman@4 but none is installed. You must install peer dependencies yourself.

+ newman-reporter-html@1.0.3
added 13 packages from 45 contributors in 15.176s

D:\createFile>
```

此时加上 --reporter html 参数，即可在当前目录的 newman 目录下生成一个 html 格式的报告。

即执行如下命令：

```
newman run postman.postman_collection.json --reporters html
```

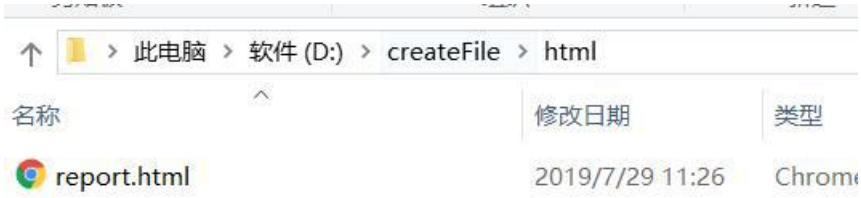
此时生成 html 格式的报告；

当然我们还可以在生成 html 报告命令基础上增加--reporter-html-export 参数指定生成的 html 文件路径，命令如下：

```
newman run postman.postman_collection.json --reporters html --
reporter-html-export D:\createFile\html\report.html
```

表示在：D 盘 createFile\html\文件夹下生成名字为 report.html 的测试报告；

```
D:\createFile>newman run postman.postman_collection.json --reporters html  --reporter-html-export D:\createFile\html\report.html

D:\createFile>
```



以上即是 postman 在自动化领域的应用；

直播专享双重优惠

原价：1388 元

原价：618 元

原价 288 元

1.买一送二活动：购买 1388 课程送价值 618 元及 288 元的课程
2.新课限时 8 折：原价 288 元的 Postman 新课，限时 230.4 元
感兴趣的同学加书雁微信:atstudy-51 进行购买（活动时间：9.1-9.6）