实验 5　Postman 接口测试

四川大学计算机学院 杨秋辉 yangqiuhui@scu.edu.cn

1 前言

本次实验进行接口测试的学习和实践，让学生掌握 Postman 工具的基本使用，熟悉简单的接口测试流程。学生根据 Github 代码托管平台功能需求，对 Github API 接口进行测试。

任务：完成 Github API 接口测试用例的编写及运行，同时完成实验报告。

本实验个人独立完成，用 2 次课（4 学时）完成。

2 测试工具简介

Postman 是一种常用的接口测试工具。Postman 是为接口测试而设计的，它可以模拟大多 HTTP 请求，向服务器发送。它提供了一种多窗口和多选项卡的页面用于发送和接收接口请求。其主要功能包括：

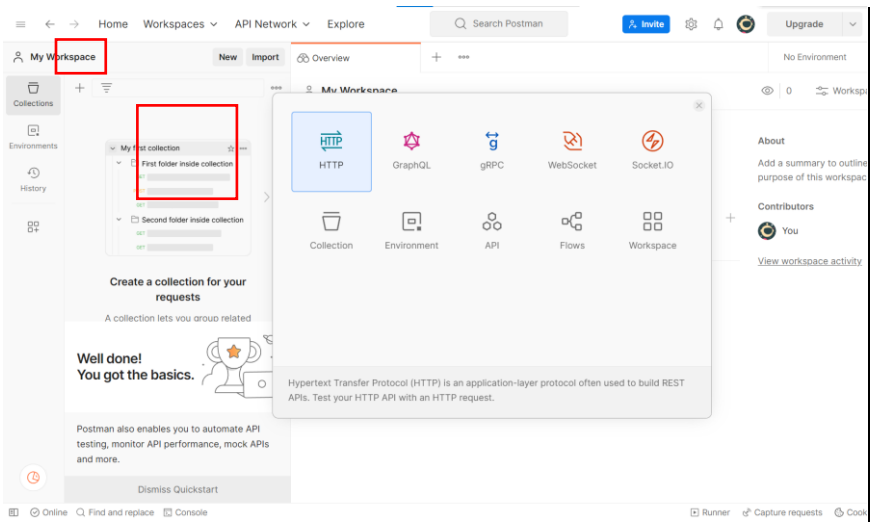
1) 网页调试、HTTP 请求发送及接口测试运行；
2) 模拟各种 HTTP 请求方式（比如：GET、POST、DELETE、PUT 等）；
3) 请求中可以发送文件（图片、文本文件等）、额外添加 header 等，实现特定的功能；
4) 更高效地帮助后台独立进行接口测试。

关于 Postman 的详细介绍，见 https://learning.postman.com/docs/introduction/overview/。

2.1 测试工具安装

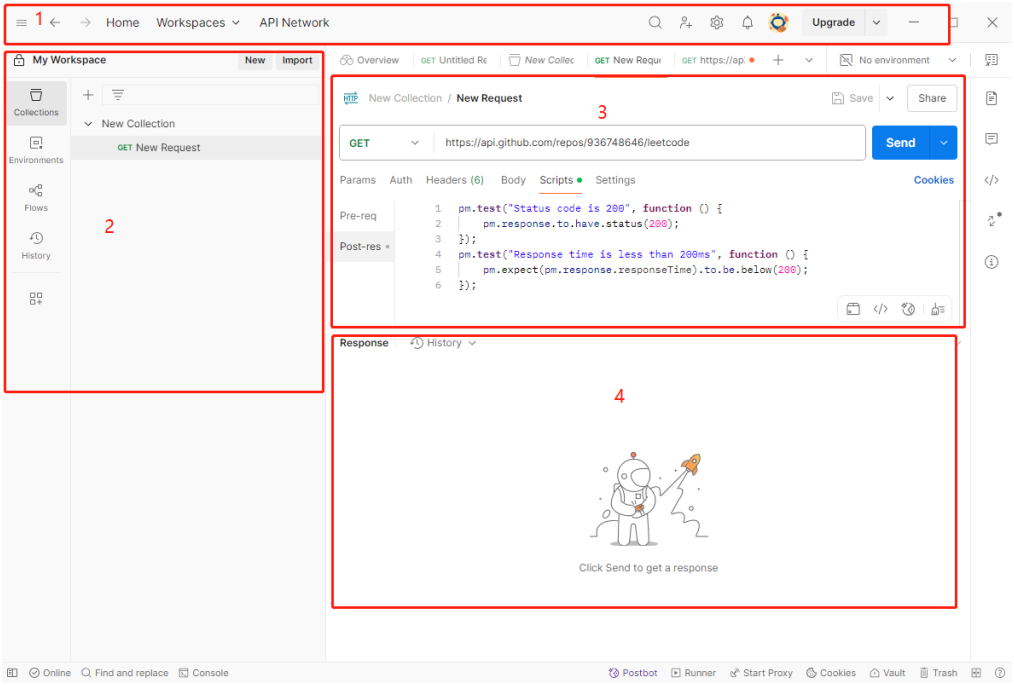
进入 Postman 的官网 https://www.postman.com/downloads/，选择好对应的版本进行下载和安装。安装成功后打开，创建账号并完成个人信息填写。（后续实验步骤要用到 GitHub，因此可用 GitHub 账号登录）

在选择 plans 时在界面右上角点击 Continue with free plans 进入主界面：

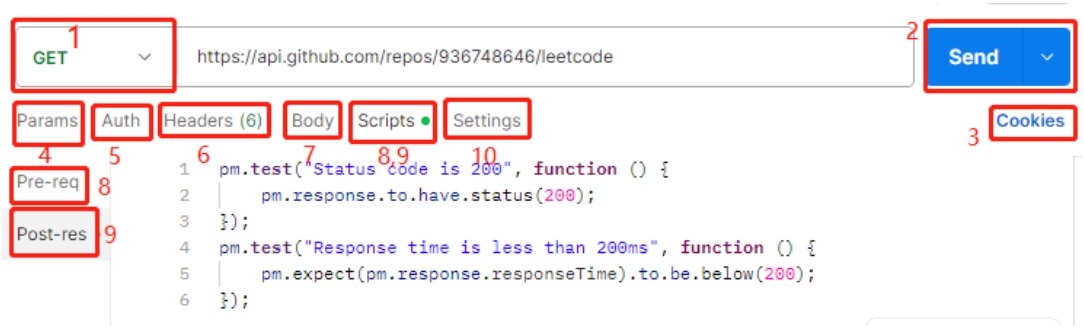


2.2 Postman APP 功能介绍

以下展示了 Postman APP 主界面的窗口布局：



1. **导航栏**：可以创建和选择工作区、搜索、邀请协作者、设置等。
2. **侧边栏**：可以导入或创建项目。
3. **请求区域**：显示对 URL 进行的请求操作，默认为 GET 方法，下方表格中的值可以直接编辑，如下图所示。

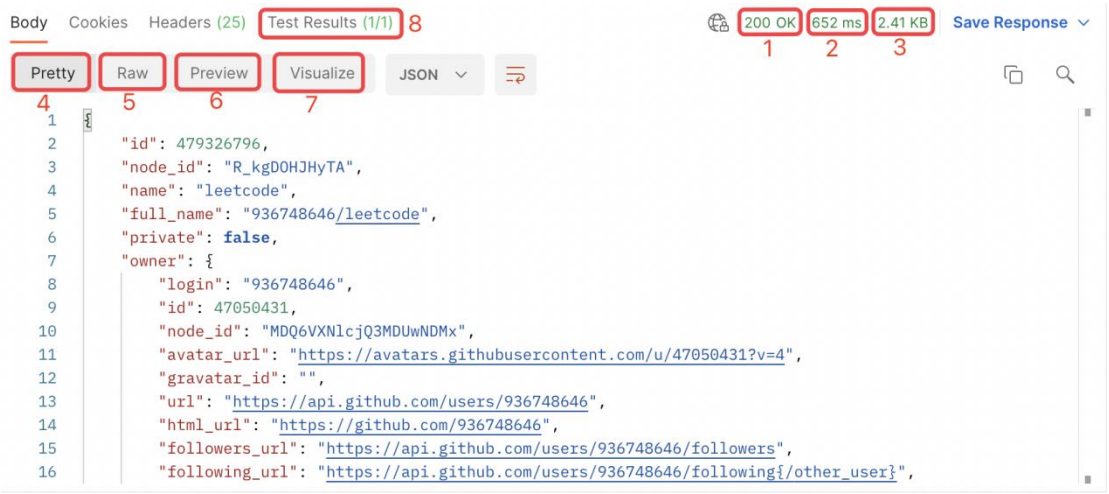


其中，各标号处的功能如下：

标号	功能和意义
1	HTTP 方法（默认为 GET），可以下拉进行选择。常用方法有：GET、POST、PUT、DELETE 等
2	发送请求
3	管理与请求相关的 Cookies
4	参数管理
5	鉴权管理
6	消息头管理，可以定义头部信息
7	消息体管理

8	发送请求前的预处理工作。
9	在请求响应后进行的其他处理（如校验等）。
10	设置。

4. **响应区域**：显示请求所得到的响应。如图所示：



其中，各标号处的意义如下：

标号	功能和意义
1	响应状态码。200 代表 HTTP 请求成功。
2	响应时间（从发出请求到返回客户端接收的时间）。
3	消息大小（包含消息头和消息体）。
4	对响应返回内容进行格式化美观显示，默认 json。
5	响应返回内容的未格式化形式。
6	预览浏览器中渲染后的情况。
7	可视化响应。
8	显示测试脚本的校验结果。

3 被测对象

GitHub 是一个面向开源及私有软件项目的托管平台，因为只支持 Git 作为唯一的版本库格式进行托管，故名 GitHub。

GitHub 于 2008 年 4 月 10 日正式上线，除了 Git 代码仓库托管及基本的 Web 管理界面以外，还提供了订阅、讨论组、文本渲染、在线文件编辑器、协作图谱（报表）、代码片段分享（Gist）等功能。作为开源代码库以及版本控制系统，Github 拥有超 900 万开发者用户，托管版本数量也非常多，其中不乏知名开源项目 Ruby on Rails、jQuery、python 等。随着越来越多的应用程序转移到了云上，GitHub 已成为管理软件开发以及发现已有代码的首选方法。

GitHub 提供了公开的 API 来让我们进行几乎一切的 GitHub 操作。GitHub API 主要包括 REST API 和 GraphQL API，相关内容可查看 GitHub 文档：https://docs.github.com/cn。

本次接口测试实验将使用 REST API 进行演示。

4 接口测试

4.1 基本概念

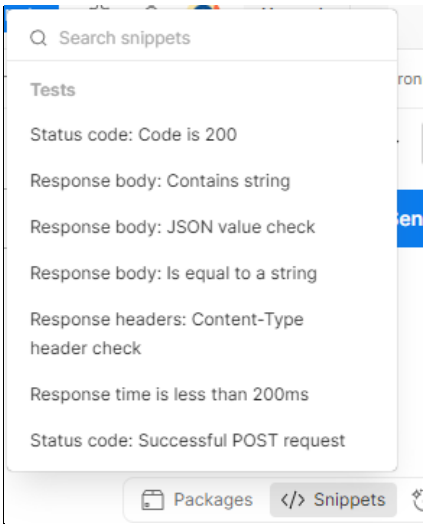
4.1.1 HTTP 请求方法

以下是几种常用 HTTP 方法的主要功能：

GET	请求指定的页面信息，并返回实体主体。
POST	向指定资源提交数据进行处理请求（例如提交表单或上传文件）。
PUT	从客户端向服务器传送的数据取代指定的文档的内容。
PATCH	是对 PUT 方法的补充，用来对已知资源进行局部更新 。
DELETE	请求服务器删除指定的页面。

4.1.2 测试脚本（Snippets）

在 Postman 中，提供了常用的各种接口数据处理操作（Snippets）。Snippets 是一组预定义的 JavaScript 代码片段，用于帮助用户快速编写和执行测试脚本。点击 Scripts->Post-response，在请求区域右下角点击 Snippets，可以看到如下：



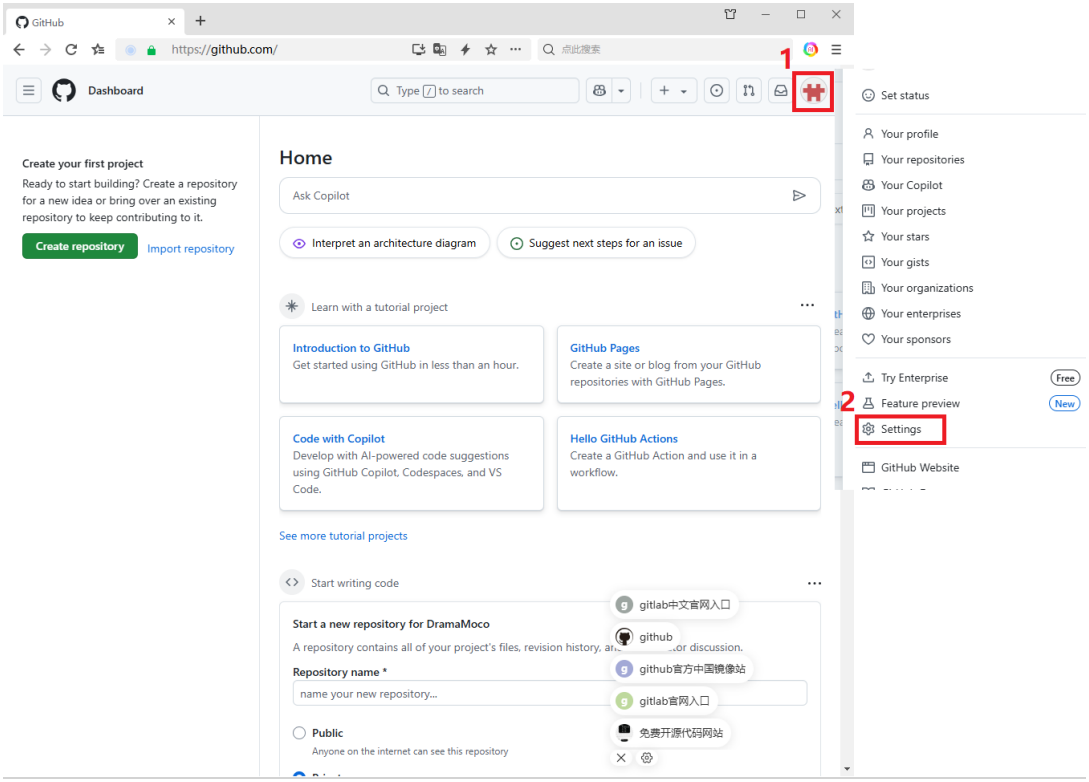
4.2 一个接口测试实例

通过实例来了解接口测试和 Postman 的基本使用方法。REST API 文档中罗列了 GitHub 许多功能点，如图所示：

本节将针对 GitHub 仓库（repository），测试 GitHub API 中常用的 HTTP 方法。

注：部分 API 测试时，要求有 GitHub Token。GitHub Token 生成步骤：

1) 登录 GitHub（没有账号需要先注册） → Settings → Developer settings → Personal Access Tokens → Tokens (classic) → Generate new token (classic)



2) 填写 Note，勾选权限：repo（全选）、user（全选）

3) 生成后保存 Token（**仅显示一次，需妥善保管**）

4.2.1 测试 API 接口

创建 Postman Collection（测试集合）。使用 Collection，可以统一管理所有测试请求。

1. **新建 Collection**

- 点击 Postman 左侧导航栏的 “+” 号 → 选择 Blank collection → 输入 Collection 名称

‘Github 仓库基本功能测试_Collection’

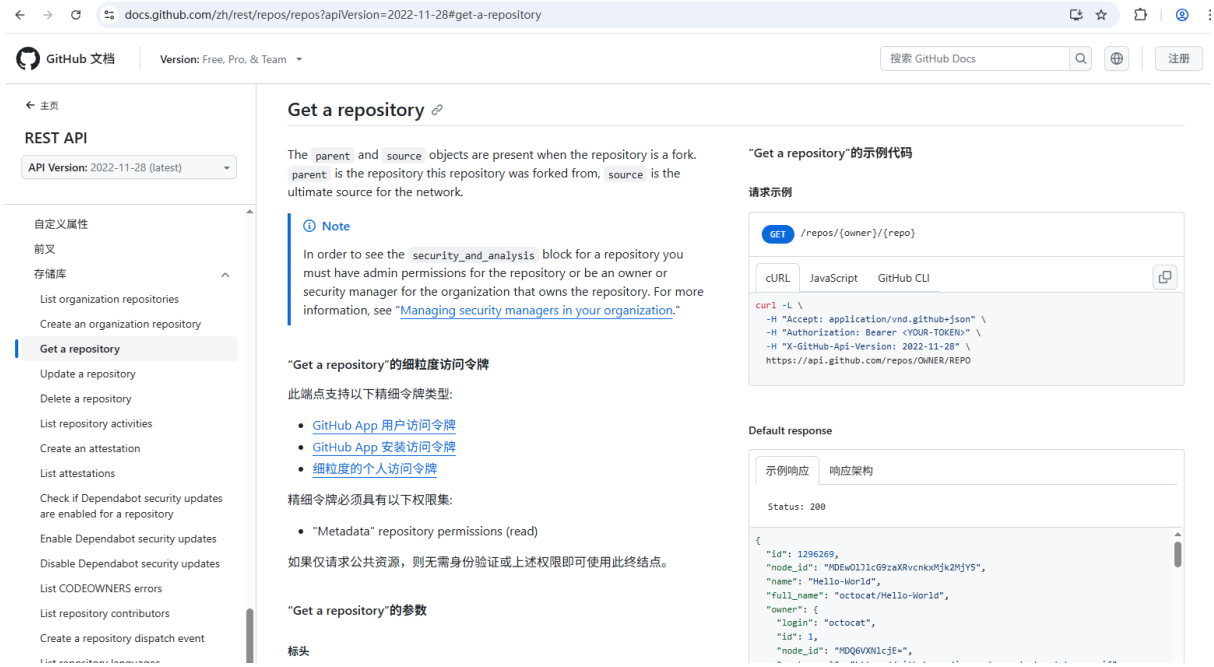
2. 将测试请求添加到 Collection

右键 Collection/或者点击 Collection 右侧的“。。。” → Add Request，创建下列各个请求：

（1）简单 GET 请求测试：获取用户信息

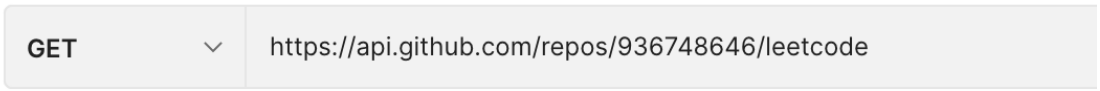
使用 GET 方法来获取 GitHub 中的一个 repository 仓库信息。

相关说明文档位于 https://docs.github.com/en：“Developers”“REST API”->左侧导航栏“存储库（Repositories）”->存储库（Repositories）-> Get a repository



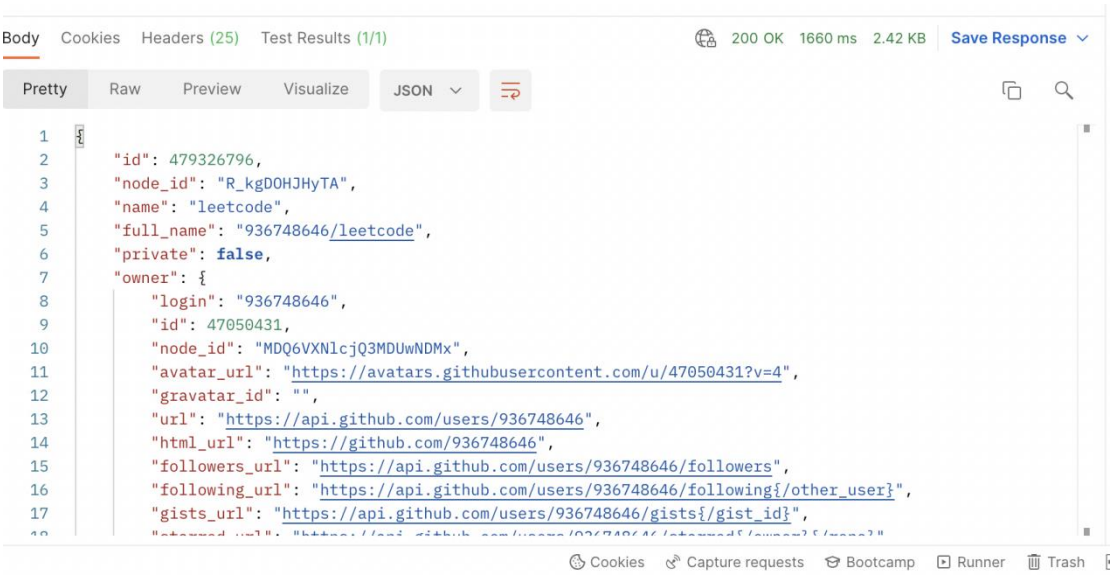
预先在 GitHub 中创建一个 repository：Your Projects->Create new…->new repository，命名为 leetcode，设为 Public。

打开 Postman，输入请求名称“GetRepoTest”，在 URL 栏里输入路径：https://api.github.com/repos/{owner}/leetcode。其中，{owner}代表用户账号,leetcode 为之前创建的 repository 名。



Headers 下的 Key 为 Authorization 的一栏取消勾选

点击 Send 发送请求，请求成功则可看到下方响应区域获得了 repository 的信息，如图所示：

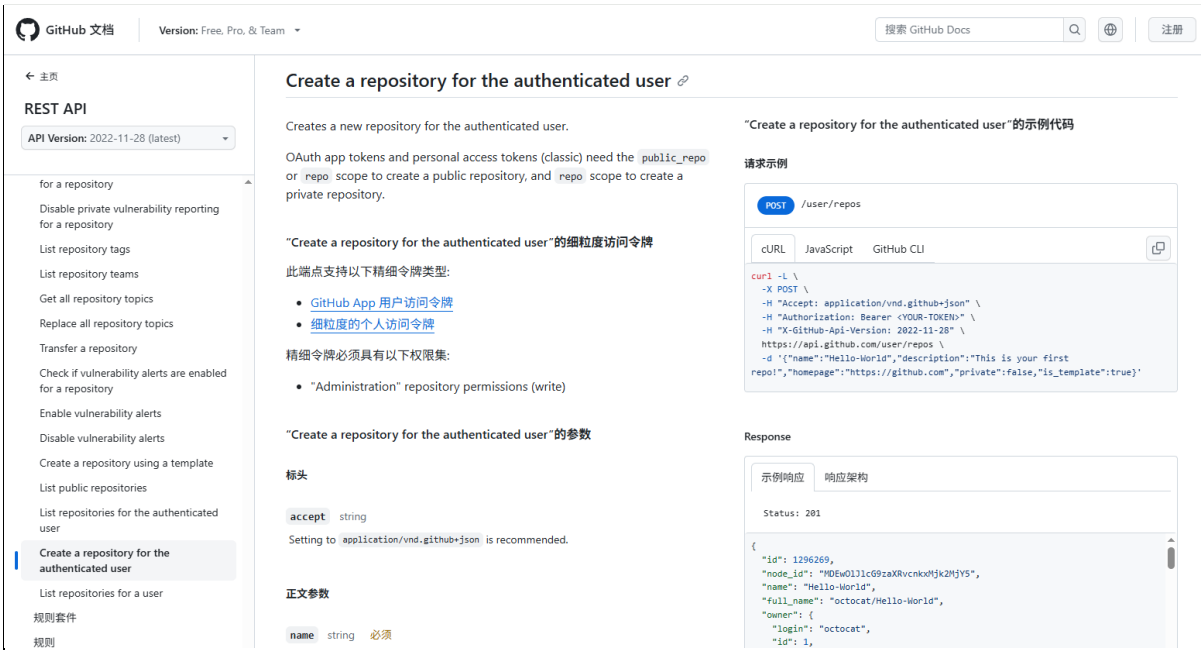


点击请求名称右侧的 Save，保存创建的请求。

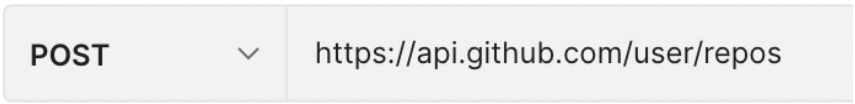
（2） 带认证的 POST 请求测试：创建仓库

使用 POST 方法来创建一个 repository 仓库。

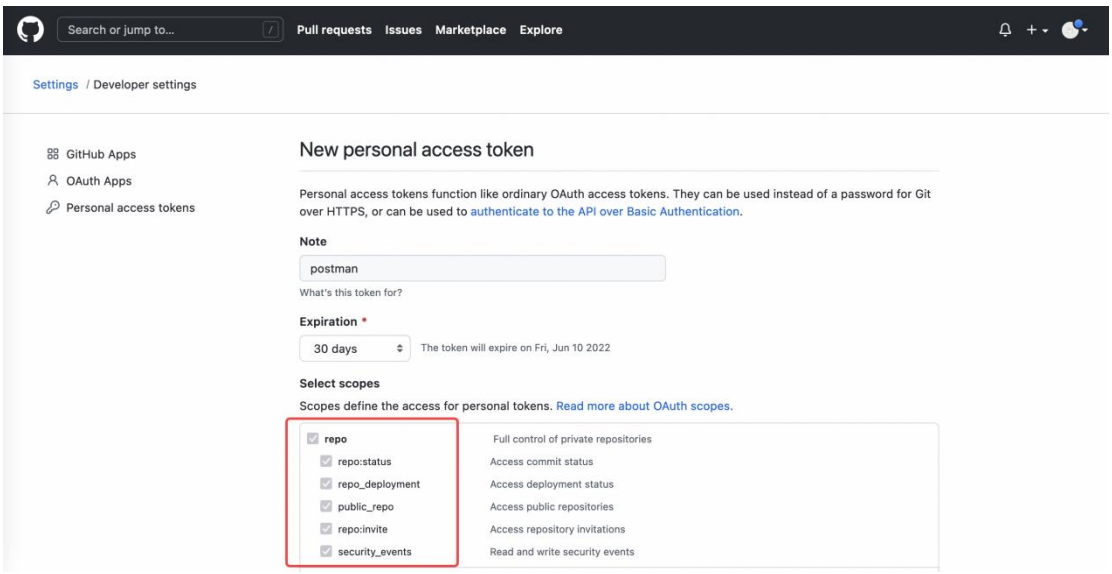
首先，查看 REST API 文档，找到“Create a repository for the authenticated user”。



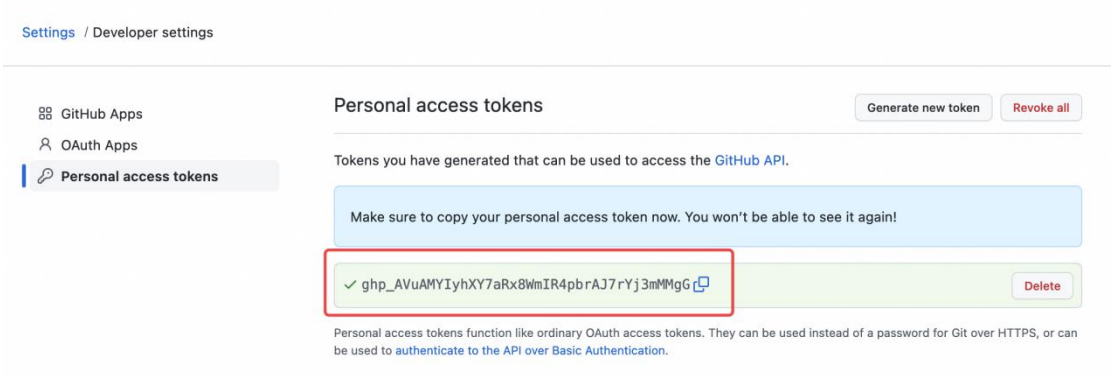
打开 Postman，输入请求名称“PostRepoTest”，选择 POST 方法，在 URL 栏里输入路径：https://api.github.com/user/repos。



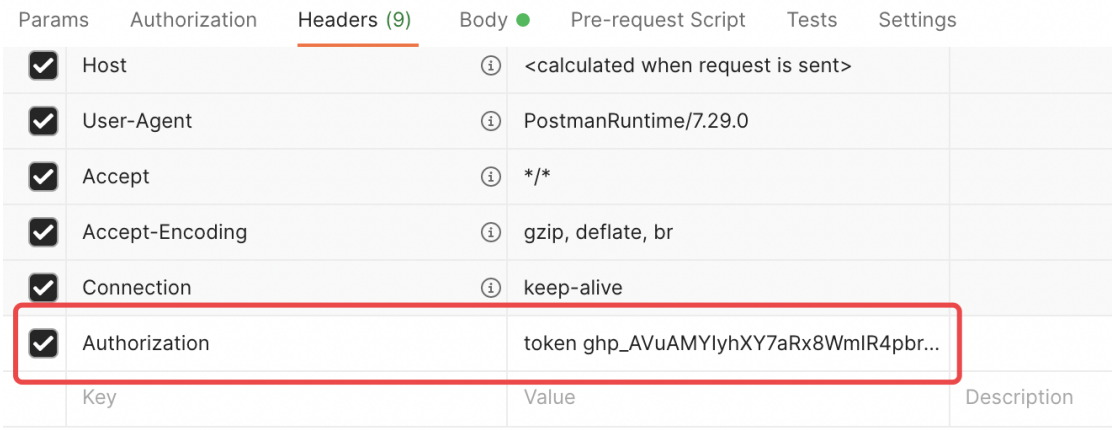
注意，要在自己的 Github 中创建仓库，必须先进行账户的授权，将代表账户登录权限的 tokens 添加到 Postman 中。打开自己的 Github，点击右上角头像->Settings->Developer settings->Personal access tokens ->Tokens(classic)->Generate new token，勾选 repo，允许对 repository 进行操作。



点击 Generate token ，完成创建并复制 token。

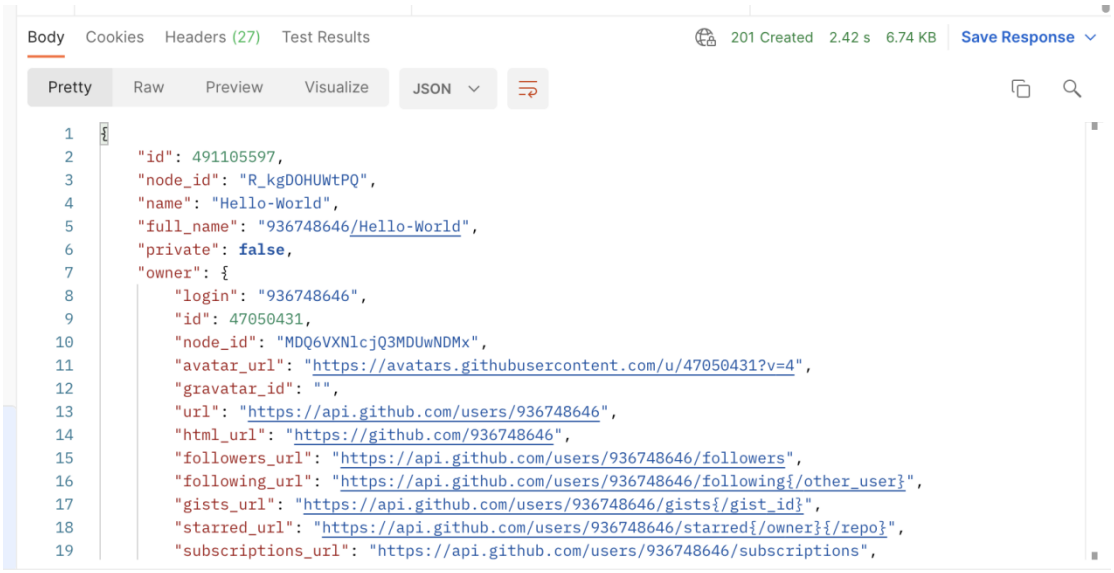


回到 Postman 中，点击 Headers，添加一条 Authorization，值为“token ”+你复制的 token。

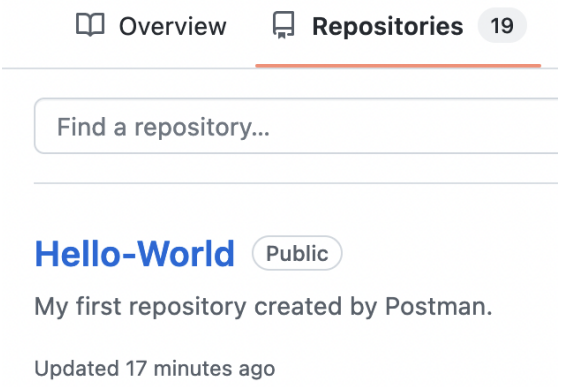


点击 Body，填写消息体。其中，“name”是必填字段。

点击 Send 发送请求，请求成功则可看到下方响应区域显示了我们刚才创建的 repository 的信息，如图所示：



打开 Github，可以看到该仓库已被成功创建。



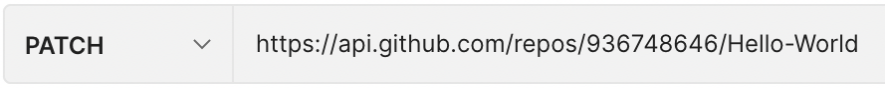
点击请求名称右侧的 Save，保存创建的请求。

（3） PATCH 方法：更新仓库

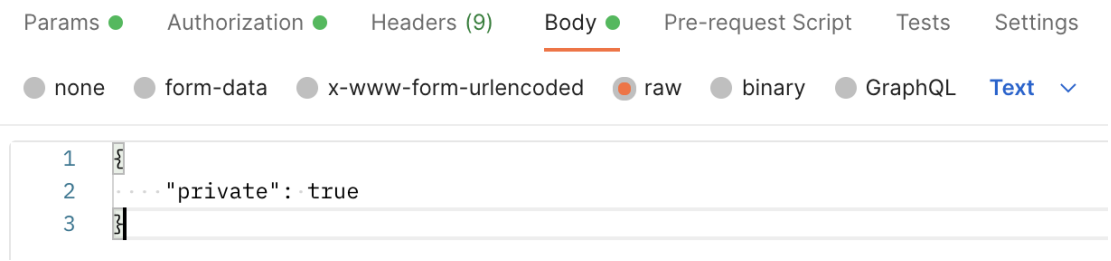
使用 PATCH 方法来修改刚才创建的“Hello-World”仓库。

首先，在文档中找到“Update a repository”。

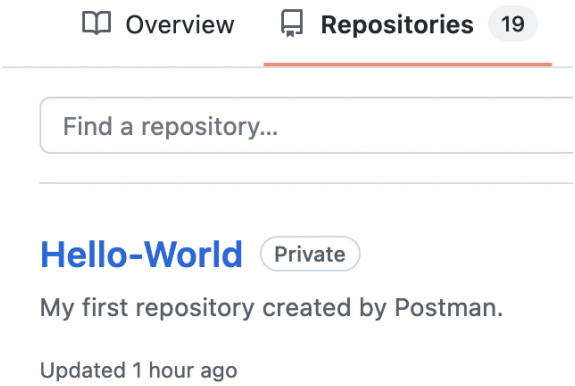
打开 Postman，输入请求名称“PatchRepoTest”，选择 PATCH 方法，在 URL 栏里输入路径：https://api.github.com/repos/{owner}/{repo}。将 owner 和 repo 进行替换，如图所示：



点击 Body，填写消息体。此处修改该仓库的访问权限 Public 为 Private。注意添加 Authorization（4.1.2 节）。



点击 Send 发送请求，请求成功则可看到 Hello-World 被修改为了 Private。

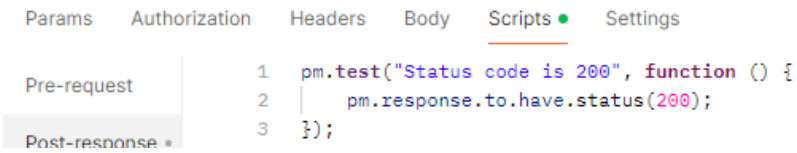


点击请求名称右侧的 Save，保存创建的请求。

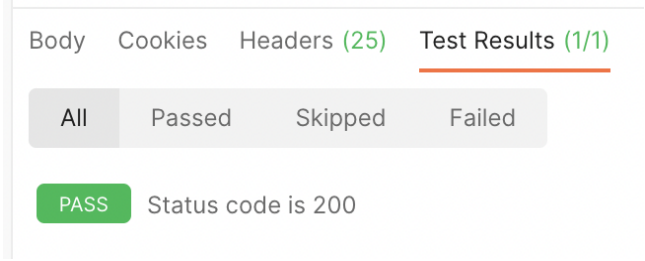
4.2.2 Snippets 的使用

在熟悉使用 API 对 GitHub 进行操作之后，我们需要验证这些 API 操作是否成功，可以通过查看返回的状态码，判断操作结果是否符合预期。

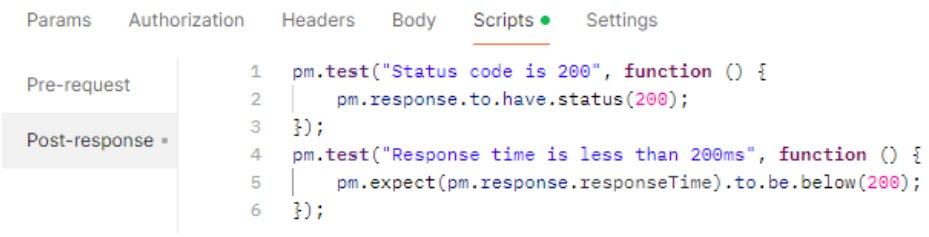
在上一节的 Get 方法中，点击 Scripts，选择 Post-response，在右下角的 Snippets 中点击“Status code: Code is 200”。可以看到系统自动编写出了测试脚本，验证返回状态 status。status 为 200 时代表返回成功：



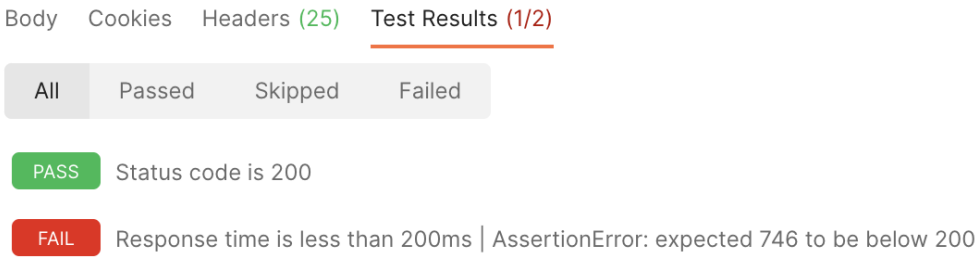
点击 Send 发送请求，开始测试。测试完成后，在下方响应区域点击 Test Results 查看测试结果。



另外，还可以检查响应时间是否符合预期。点击 Snippets 中的“Response time is less than 200ms”，测试响应时间是否小于 200ms，如图所示：



点击 Send 发送请求，开始测试。可以看到测试没有通过，因为响应时间大于了 200ms：



如果 Scripts 中自动生成的脚本不能满足你的需要，可以进行修改或自己创建。

4.2.3 导出创建的请求及测试结果

1. 导出 Postman 文件

确保 Collection 中包含所有配置好的请求、脚本及环境变量引用，按如下步骤保存：

- 保存 Collection：右键点击 Collection/或点击 Collection 右侧的。。。 → More → Export →选择 Collection v2.1→Export→保存为`Github 仓库基本功能测试.postman_collection.json`

-（可选）导出环境变量（如 Token）：Environments → Export → 保存为`Github 仓库基本功能测试.postman_environment.json`(如果没有环境变量，此项不需要导出)

注意：若使用敏感数据（如 Token），导出前清除环境变量中的实际值（保留`{{github_token}}`占位符）

2. 导出测试结果

用 Collection Runner 导出测试摘要

1) 打开 Collection Runner：点击 Postman 窗口右下角的 Runner 选项卡→点击左侧 Collections → 选择目标 Collection → 点击 Collection 名称右边的“Run”按钮
2) 勾选需要运行的请求 → 设置迭代次数（默认 1 次） → 点击“Run [Collection 名]”
3) 运行完成后，点击右上角“Export Results”→ 保存为`Github 仓库基本功能测试.postman_test_run.json`

5 实验任务

根据 GitHub API 文档，选择至少 4 个功能进行测试。一个功能是指一个有实际意义的操作序列。例如功能“Github 存储库基本操作”，包括的操作序列中有 3 个基本操作：在 Github 上创建一个存储库，修改存储库中的信息，然后获取该存储库的信息。

对于被测试的每一个功能，创建一个 Collection，每个功能要求：

1) 至少使用 4 种不同的 HTTP 请求方法；

2） 至少包括 3 个不同的操作（如第 4 节中对存储库的“创建”和“修改”就是 2 个操作）；

3） 测试脚本（Post-response）中至少使用 5 种不同的 Snippets 脚本；

6 提交内容及评分标准

根据实验资料中给出的“实验 5_Postman 接口测试_实验报告模板”撰写实验报告。要求提交的内容包括：实验报告 pdf 文件+导出的 Postman 文件+导出的测试结果文件。将所有内容打包为 zip 提交，打包文件命名规定：“姓名_学号_接口测试实验报告.zip”。

实验成果文件示例：

姓名_学号_接口测试实验报告.zip

├─── Github 仓库基本功能测试.postman_collection.json （每个功能一个）

├─── Github 仓库基本功能测试.postman_results.json （每个功能一个）

├─── ……

└─── 姓名_学号_实验报告.pdf

每个功能 25 分。选择测试的 API 的数量、测试难度、测试的完整性等，会影响最终实验分数。

7 总结

完成本实验后，学生对简单的接口测试应该有一定了解。本次实验对于学生以后进行接口测试、使用 Postman 这样的接口测试工具也会有很大的帮助。

常用的接口测试工具除了 Postman 以外，还有 SoapUI、REST-Assured、JMeter、Apifox 等。Postman 是一个强大的自动化测试工具，除了做接口测试外，现在还有人用其做简单的压力测试。在本次实验之后，学生可以在了解了 Postman 的基本功能的基础上进行更加深入的学习。