

**Федеральное государственное бюджетное образовательное учреждение
высшего образования**

**«Белгородский государственный технологический университет им.
В.Г.Шухова»**

**Кафедра программного обеспечения вычислительной техники и
автоматизированных систем**

**Методические указания к лабораторной работе № 4. «Скрипты для
автоматизации тестирования REST API в Postman»**

по дисциплине «Тестирование программных систем»

Выполнил: ст. группы ПВ-202

Буйвало А.А.

Проверили:

Лебединская Анастасия Александровна

Бабенко Анастасия Александровна

Белгород, 2023

Лабораторная работа №4

«Скрипты для автоматизации тестирования REST API в Postman»

Цель работы:

- 1.1. Рассмотреть типы скриптов и порядок их запуска;
- 1.2. Рассмотреть скрипты, запускаемые перед запросом, их назначение и основные задачи, решаемые с их помощью;
- 1.3. Рассмотреть скрипты, запускаемые после получения ответа;
- 1.4. Составить коллекцию и обработать ответ от сервера с помощью простых скриптов.

Задания

Необходимо создать коллекцию и присвоить ей наименование по следующему шаблону **Группа_ФИО_ЛабораторнаяРабота_4_Задание(номер задания)**. Коллекции без правильного наименования приняты не будут.

Необходимо выполнить скрипты по следующим заданиям, т.е. должно быть три коллекции:

1. Задание № 1 или № 2;
2. Задание № 3 или № 4;
3. В задании № 5 какой-то один метод, кроме запроса токена

Необходимо оформить краткий отчет о лабораторной работе исходя из ее целей. К отчету прикрепить скриншоты вызовов с открытой вкладкой Tests.

Выполнение работы

1. Проверить ,что статус код ответа = 200 ;

```
pm.test("Status code is 200",function () {  
    pm.response.to.have.status(200);  
});
```

2. Присвоить тело ответа переменной response_JSON ;

```
var response_JSON = pm.response.json();
```

3. Проверить, что name в ответе равно name request (ввести вручную);

```
pm.test("Проверить, что name в ответе равно name request (ввести вручную);", function () {  
    pm.expect(response_JSON.name).to.eql("Nastya");  
});
```

4. Проверить, что age в ответе равно age request (ввести вручную);

```
pm.test("Проверить, что age в ответе равно age request (ввести вручную);",  
function () {  
    pm.expect(response_JSON.age).to.eql("20");  
});
```

5. Проверить, что salary в ответе равно salary request (ввести вручную);

```
pm.test("Проверить, что salary в ответе равно salary request (ввести  
вручную);", function () {  
    pm.expect(response_JSON.salary).to.eql(5);  
});
```

6. Присвоить тело запроса переменной request_JSON ;

```
var request_JSON = request.data
```

7. Проверить, что name в ответе равно name request (взять из тела запроса);

```
pm.test("Проверить, что name в ответе равно name request (взять из тела  
запроса);", function () {  
    pm.expect(response_JSON.name).to.eql(request_JSON.name);  
});
```

8. Проверить, что age в ответе равно age request (взять из тела запроса);

```
pm.test("Проверить, что age в ответе равно age request (взять из тела  
запроса);", function () {  
    pm.expect(response_JSON.age).to.eql(request_JSON.age);  
});
```

9. Проверить, что salary в ответе равно salary request (взять из тела запроса);

```
pm.test("Проверить, что salary в ответе равно salary request (взять из тела  
запроса);", function () {  
    pm.expect(response_JSON.salary).to.eql(parseInt(request_JSON.salary));  
});
```

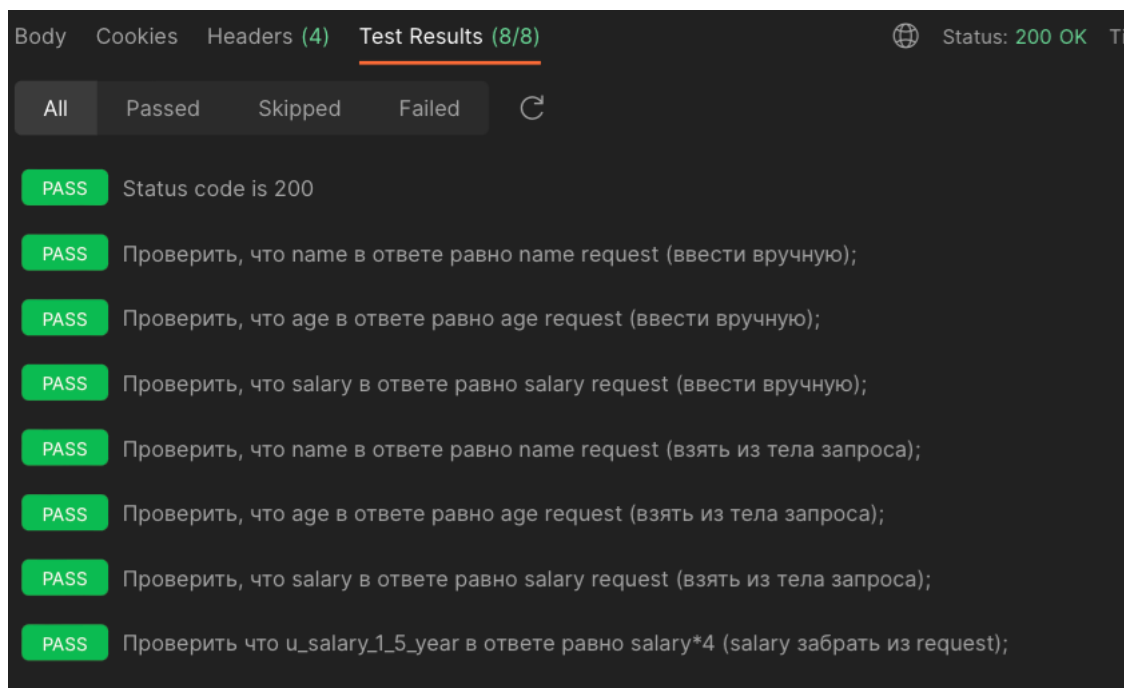
```
});
```

10. Вывести в консоль параметр family из response;

```
console.log(response_JSON.family)
```

11. Проверить что u_salary_1_5_year в ответе равно salary*4 (salary забрать из request)

```
pm.test("Проверить, что age в ответе равно age request (ввести вручную);",  
function () {  
  
pm.expect(response_JSON.family.u_salary_1_5_year).to.eql(request_JSON.sal  
ary*4);  
});
```



Задание №3

Запрос методом GET https://162.55.220.72:5005/user_info_4 параметры запроса name, age, salary присвоить самостоятельно

GET

▼

http://162.55.220.72:5005/object_info_4?name=Nastya&age=20&salary=5

Params

Authorization

Headers (5)

Body

Pre-request Script

Tests

Settings

Query Params

	Key	Value	Description
<input checked="" type="checkbox"/>	name	Nastya	
<input checked="" type="checkbox"/>	age	20	
<input checked="" type="checkbox"/>	salary	5	

1. Проверить, что статус код ответа = 200

```
pm.test("Status code is 200", function () {
  pm.response.to.have.status(200);
});
```

2. Присвоить тело ответа переменной response_JSON ;

```
var response_JSON = pm.response.json()
```

3. Присвоить тело запроса переменной request_JSON

```
var request_JSON = pm.request.url.query.toObject();
```

4. Проверить, что name в ответе равно name s request (name забрать из request.)

```
pm.test("Test response name with request data", () => {
  pm.expect(response_JSON.name).to.equal(request_JSON.name);
})
```

5. Проверить, что age в ответе равно age из request (age забрать из request.)

```
pm.test("Test response age with request data", () => {
  pm.expect(response_JSON.age).to.equal(parseInt(request_JSON.age));
})
```

6. Вывести в консоль параметр salary из request

```
console.log("request salary =", request_JSON.salary);
```

7. Вывести в консоль параметр salary из response

```
console.log("response salary =", response_JSON.salary)
```

8. Вывести в консоль 0-й элемент параметра salary из response
9. Вывести в консоль 1-й элемент параметра salary параметр salary из response
10. Вывести в консоль 2-й элемент параметра salary параметр salary из response

```
console.log("salary[0] =", response_JSON.salary[0])  
console.log("salary[1] =", response_JSON.salary[1])  
console.log("salary[2] =", response_JSON.salary[2])
```

11. Проверить, что 0-й элемент параметра salary равен salary из request (salary забрать из request.)
12. Проверить, что 1-й элемент параметра salary равен salary*2 из request (salary забрать из request.)
13. Проверить, что 2-й элемент параметра salary равен salary*3 из request (salary забрать из request.)

```
pm.test("Test response salary[0]", () => {  
  
  pm.expect(parseInt(response_JSON.salary[0])).to.eql(parseInt(request_JSON.salary));  
})  
  
pm.test("Test response salary[1]", () => {  
  
  pm.expect(parseInt(response_JSON.salary[1])).to.eql(parseInt(request_JSON.salary * 2));  
})  
  
pm.test("Test response salary[2]", () => {  
  
  pm.expect(parseInt(response_JSON.salary[2])).to.eql(parseInt(request_JSON.salary * 3));  
})
```

14. Создать в окружении переменную name
15. Создать в окружении переменную age
16. Создать в окружении переменную salary

	Variable	Type	Initial value	Current value
<input checked="" type="checkbox"/>	name	de... ▾	5	Nastya
<input checked="" type="checkbox"/>	age	de... ▾		20
<input checked="" type="checkbox"/>	salary	de... ▾		

17. Передать в окружение переменную name


18. Передать в окружение переменную age


19. Передать в окружение переменную salary

```
pm.environment.set("name", request_JSON.name)
pm.environment.set("age", request_JSON.age)
pm.environment.set("salary", request_JSON.salary)
```

20. Написать цикл который выведет в консоль по порядку элементы списка из параметра salary

```
for(let i = 0; i < 3; i++)
  console.log(response_JSON.salary[i]);
```

Test Results ▾  200 OK

All Passed Skipped Failed 

- PASS Status code is 200
- PASS Test response age with request data
- PASS Test response name with request data
- PASS Test response salary[0]
- PASS Test response salary[1]
- PASS Test response salary[2]

Задание №5

Запрос методом POST <https://162.55.220.72:5005/login>. Необходимо залогиниться login : str , password : str . Полученный токен необходимо передавать в последующие запросы. Используйте окружение.

Отправить запрос методом POST `http://162.55.220.72:5005/test_pet_info` тело запроса отправить RAW JSON { "age" : int , "weight" : int , "name" : "string" , auth_token

: "string" }. В ответ получено тело { "name": "string", "age" : int, "daily_food" : weight *

0.012 , "daily_sleep": weight * 2.5 }. Написать тесты :

1. Проверить что статус код ответа = 200 ;
2. Проверка структуры json в ответе ;
3. В ответе указаны коэффициенты умножения weight, напишите тесты по проверке правильности результата перемножения на коэффициент.

<input checked="" type="checkbox"/>	token	default	<input type="checkbox"/>	/s34lfgbj/str/jjd909/53967kjkWpqc242...
-------------------------------------	-------	---------	--------------------------	---

```
var response_JSON = pm.response.json()
pm.environment.set("token", response_JSON.token)
```

```
1  {
2    "token": "/s34lfgbj/str/jjd909/53967kjkWpqc2423str43554evny"
3  }
```

```
pm.test("Status code is 200", function () {
  pm.response.to.have.status(200);
});

const response_JSON = pm.response.json();
const request_FORMDATA = pm.request.body.formdata;

const schema = {
  "items": {
    "age": {
      "type": "string"
    },
    "daily_food": {
      "type": "number"
    },
    "daily_sleep": {
      "type": "number"
    }
  }
}
```



```
"name": {
  "type": "string"
}
}
}

pm.test("Scheme validation", () => {
  pm.expect(tv4.validate(response_JSON, schema)).to.be.true;
})
var request_JSON = request.data
pm.test("daily_food", () => {
  pm.expect(response_JSON.daily_food).to.eql(request_JSON.weight * 0.012);
})

pm.test("daily_sleep", () => {
  pm.expect(response_JSON.daily_sleep).to.eql(request_JSON.weight * 2.5);
})
```

<input checked="" type="checkbox"/>	age	3
<input checked="" type="checkbox"/>	weight	2
<input checked="" type="checkbox"/>	name	Hi
<input checked="" type="checkbox"/>	auth_token	{{token}}

Params Authorization ● Headers (8) Body ● Pre-request Script Tests ● Settings

Body Cookies Headers (4) Test Results (4/4) 🌐 200 OK

All Passed Skipped Failed ↻

- PASS** Status code is 200
- PASS** Scheme validation
- PASS** daily_food
- PASS** daily_sleep

Вывод: в ходе выполнения данной лабораторной работы мною были изучены возможности использования скриптов в сценариях тестирования Postman. Рассмотрены скрипты перед запросом, скрипты после получения ответа.