Основы HTTP, REST, JSON

Цель работы

- Ознакомиться с HTTP запросами
- Поработать с REST API и JSON

Дополнительные материалы

Теоретический материал

Postman

Основное предназначение приложения — создание коллекций с запросами к вашему API. Любой разработчик или тестировщик, открыв коллекцию, сможет с лёгкостью разобраться в работе вашего сервиса. Ко всему прочему, Postman позволяет проектировать дизайн API и создавать на его основе Mock-сервер. Разработчикам больше нет необходимости тратить время на создание "заглушек". Реализацию сервера и клиента можно запустить одновременно. Тестировщики могут писать тесты и производить автоматизированное тестирование прямо из Postman. А инструменты для автоматического документирования по описаниям из ваших коллекций сэкономят время на ещё одну "полезную фичу". Есть кое-что и для администраторов — авторы предусмотрели возможность создания коллекций для мониторинга сервисов.

Введение

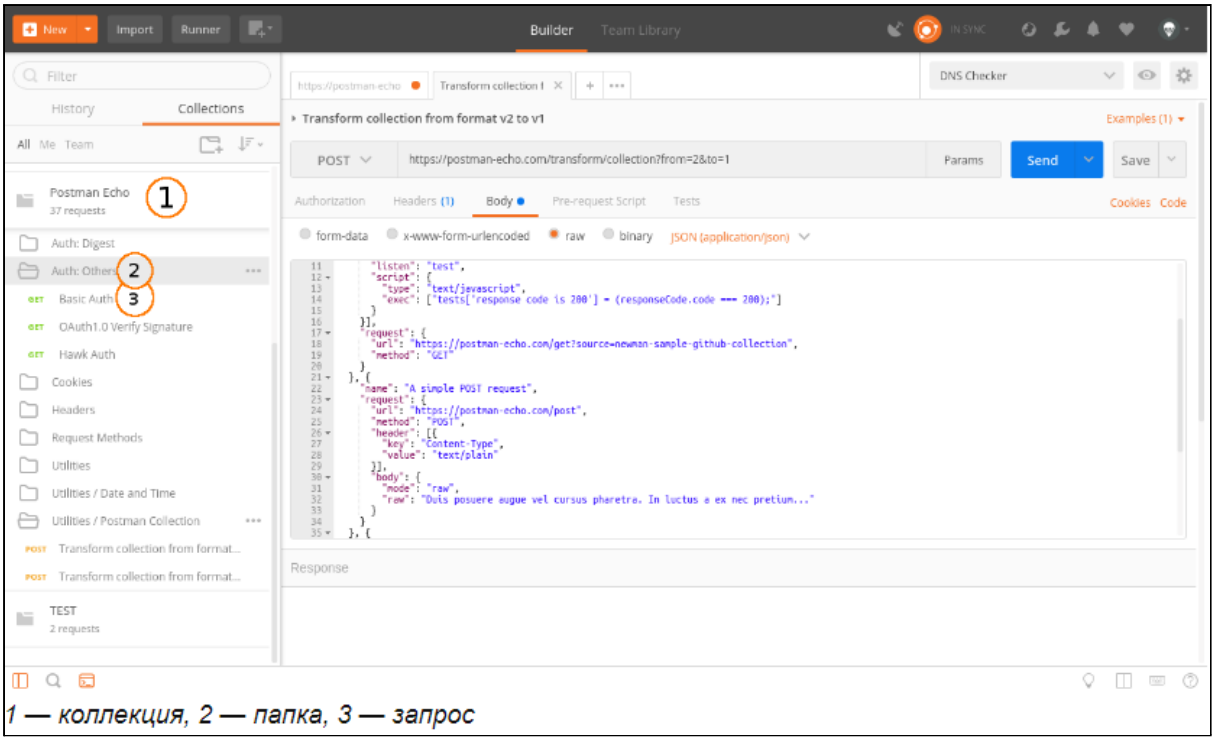


Рис.1 - Окно постмана

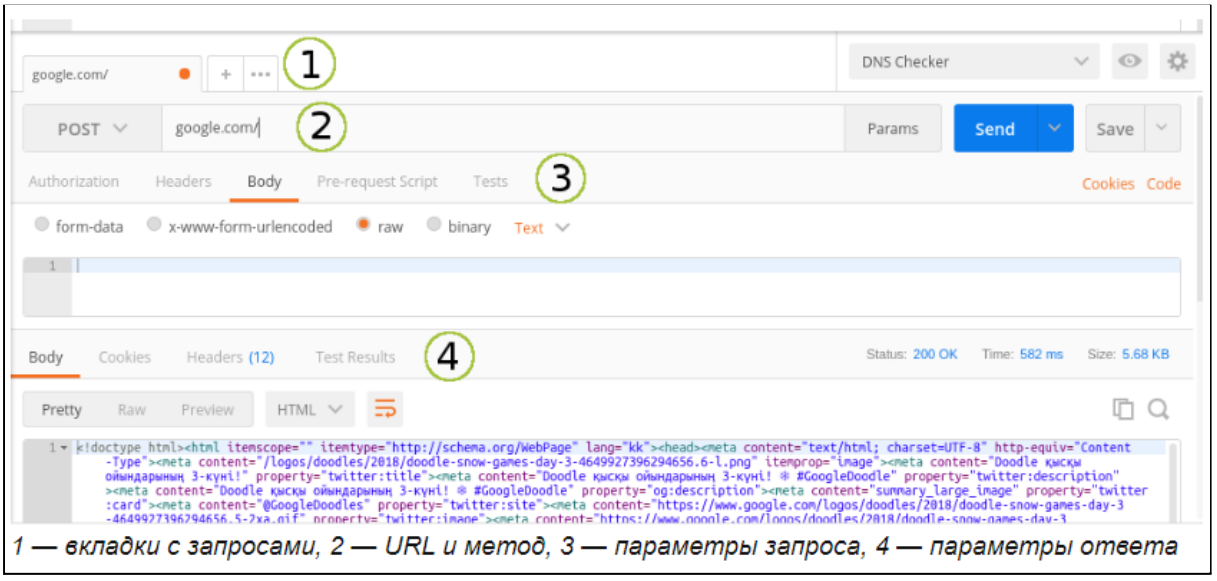
Главные понятия, которыми оперирует Postman (рис. 1) это Collection (коллекция) на верхнем уровне, и Request (запрос) на нижнем. Вся работа начинается с коллекции и сводится к описанию вашего API с помощью запросов. Давайте рассмотрим подробнее всё по порядку.

Коллекция — отправная точка для нового API. Можно рассматривать коллекцию, как файл проекта. Коллекция объединяет в себе все связанные запросы. Обычно API описывается в одной коллекции, но если вы желаете, то нет никаких ограничений сделать по-другому. Коллекция может иметь свои скрипты и переменные, которые мы рассмотрим позже.

Папка — используется для объединения запросов в одну группу внутри коллекции. К примеру, вы можете создать папку для первой версии своего API — "v1", а внутри сгруппировать запросы по смыслу выполняемых действий — "Order & Checkout", "User

profile" и т. п. Всё ограничивается лишь вашей фантазией и потребностями. Папка, как и коллекция может иметь свои скрипты, но не переменные.

Запрос (рис. 1) — основная составляющая коллекции, то ради чего все и затевалось. Запрос создается в конструкторе. Конструктор запросов это главное пространство, с которым вам придётся работать. Postman умеет выполнять запросы с помощью всех стандартных HTTP методов, все параметры запроса под вашим контролем. С легкостью можно поменять или добавить необходимые вам заголовки, cookie, и тело запроса. У запроса есть свои скрипты. Вкладки "Pre-request Script" и "Tests" среди параметров запроса позволяют добавить скрипты перед выполнением запроса и после. Именно эти две возможности делают Postman мощным инструментом помогающим при разработке и тестировании.



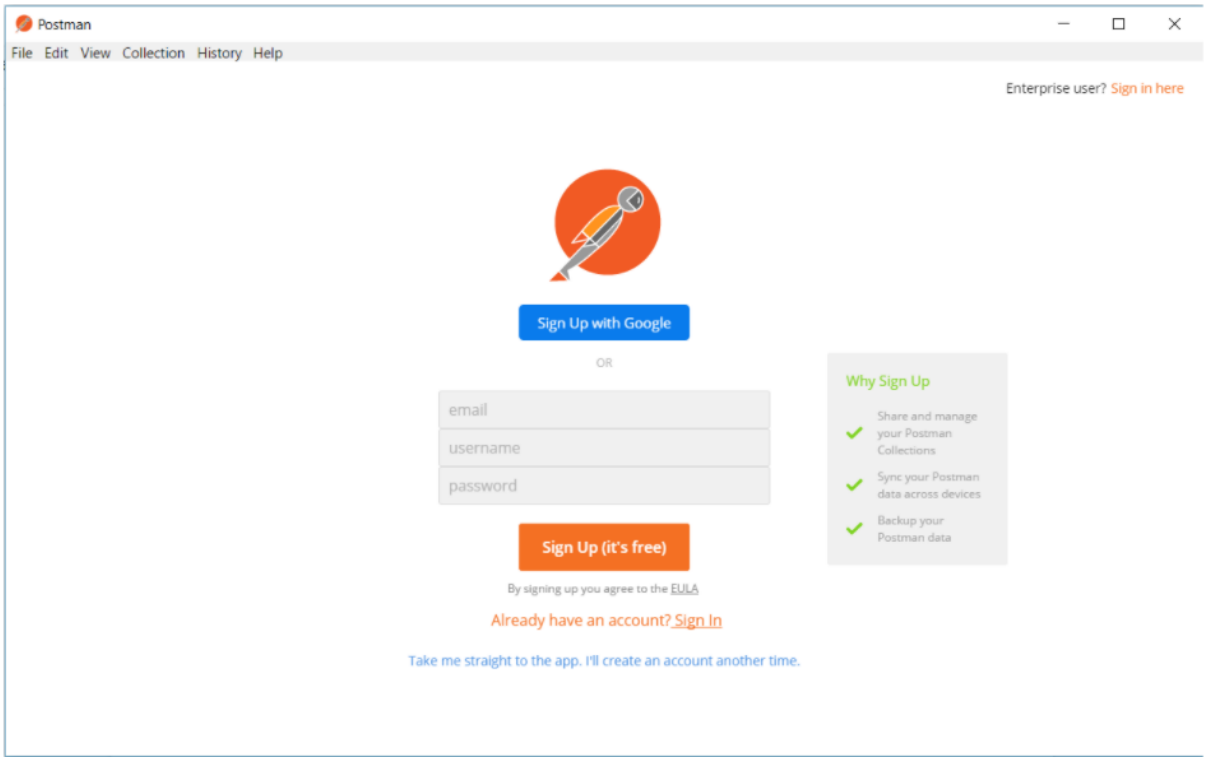
1 — вкладки с запросами, 2 — URL и метод, 3 — параметры запроса, 4 — параметры ответа

Рис.2 - Запрос

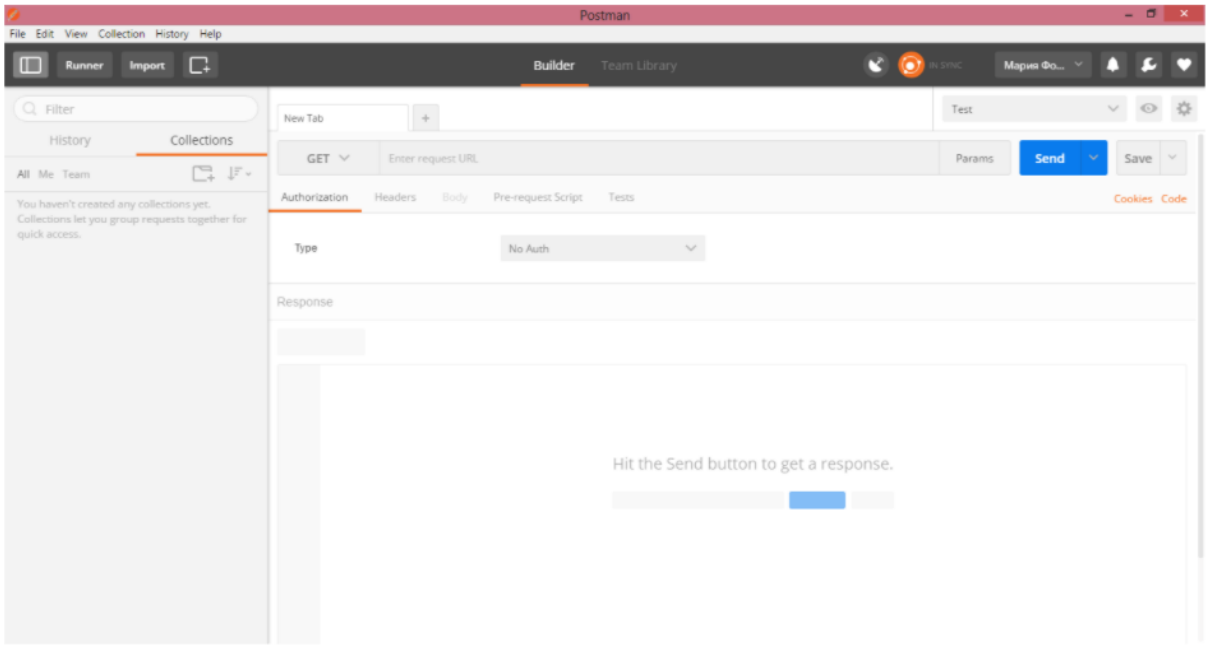
Практическая работа

Работа с Postman

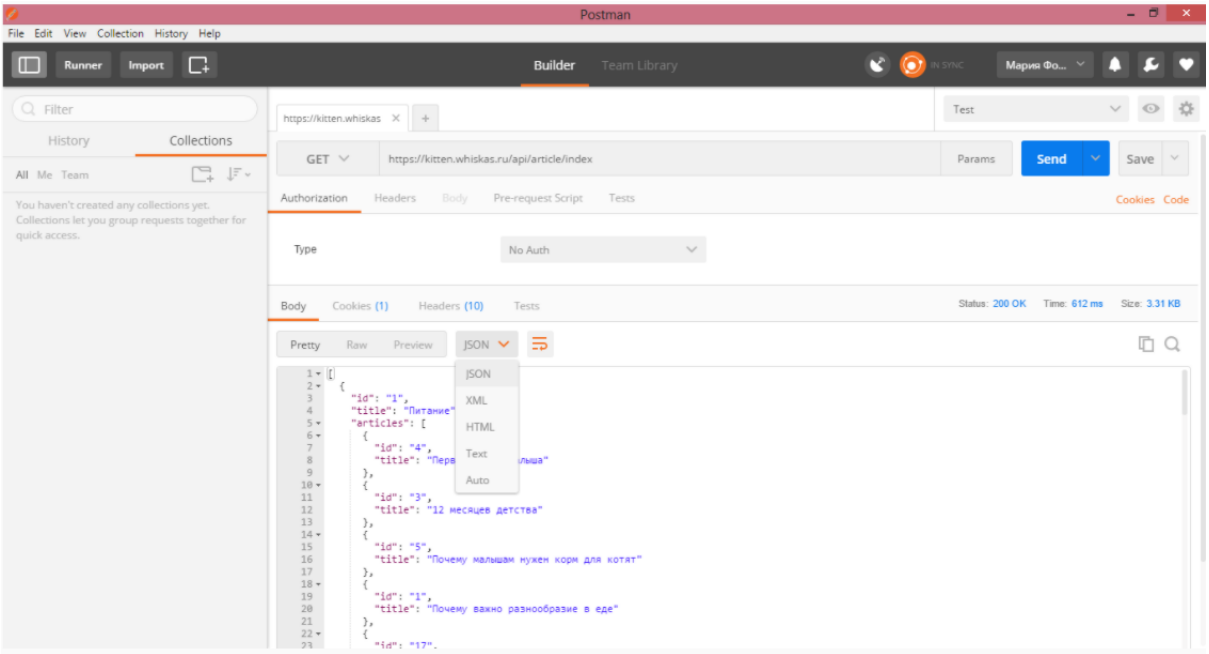
1. Для того чтобы начать использовать Postman необходимо создать аккаунт в Postman.



2. Наиболее удобным способом является авторизоваться через аккаунт Google. Для этого необходимо выбрать Sign Up with Google > Ввести учетные данные от аккаунта Google > Подтвердить вход. После чего откроется главное окно программы.



3. Теперь создадим запрос. Для этого нужно сделать:
   a. Выбрать тип запроса GET
   b. В поле “Enter request URL” ввести URL метода, например: https://kitten.whiskas.ru/api/article/index
   c. Нажать кнопку “Send”
4. Если все прошло успешно, то с сервера приходит ответ и Статус 200 OK. Для просмотра результата необходимо перейти во вкладку BODY и выбрать формат отображаемых данных > JSON.

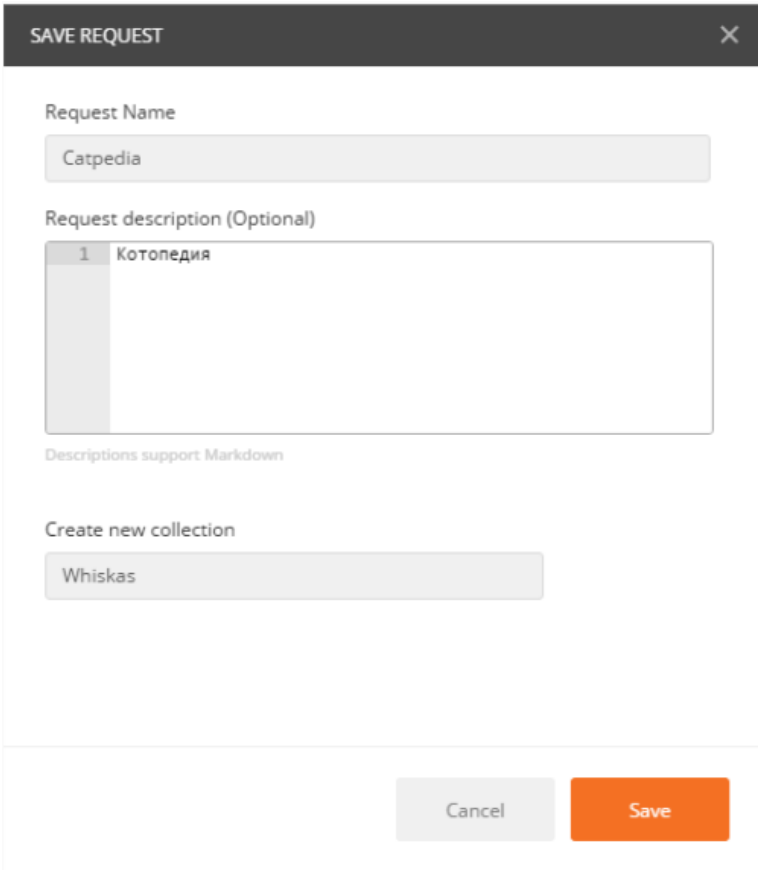


5. Для сохранения запроса необходимо нажать кнопку SAVE.

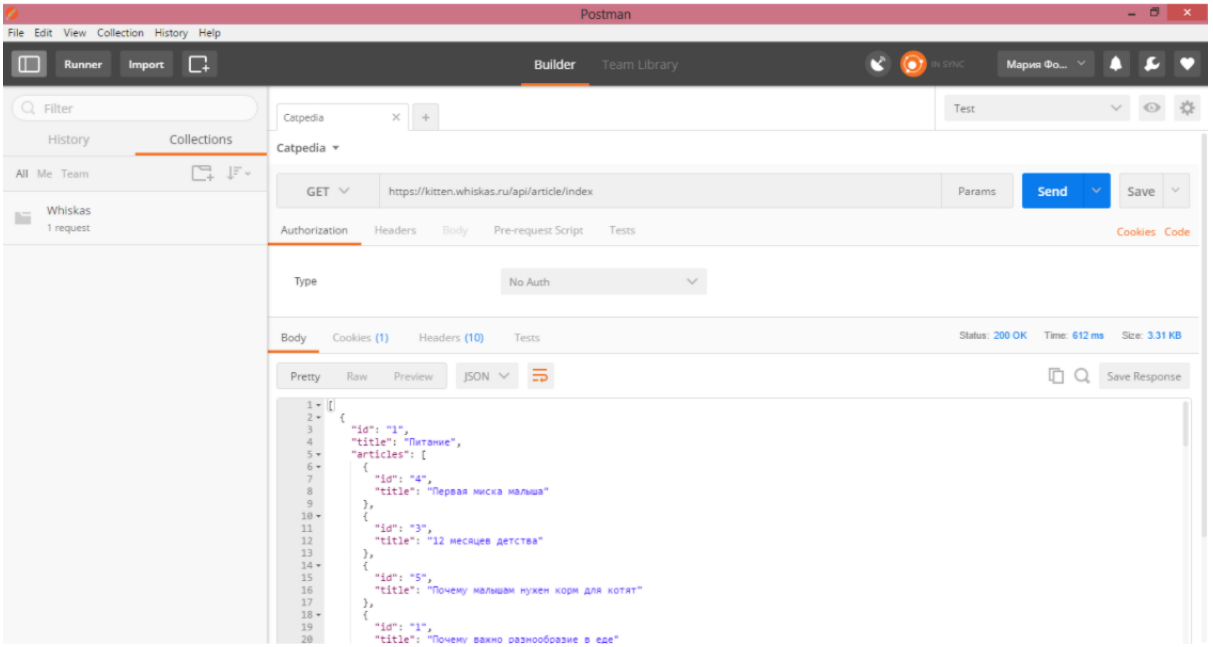
6. В открывшемся окне даем название нашему запросу, например “Catpedia”.

7. В поле Request description (необязательное поле) можно добавить описание запроса.

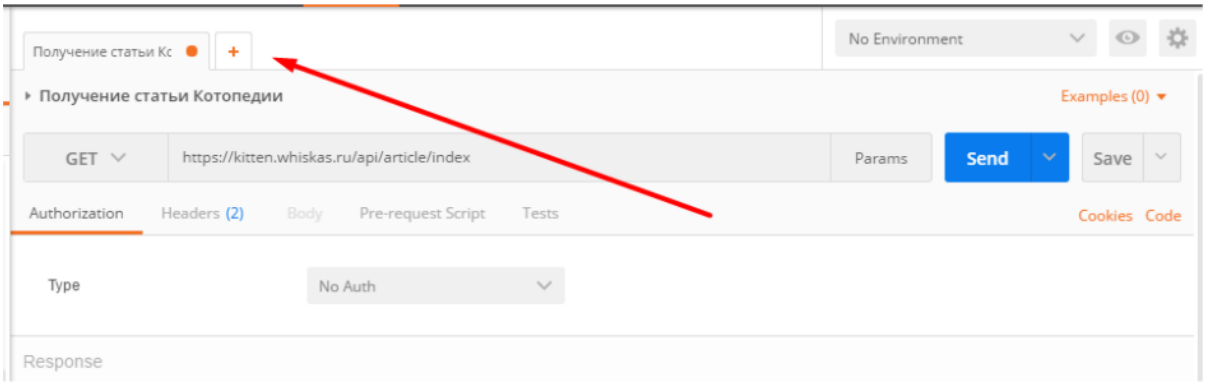
8. Если мы создаем несколько запросов для одного проекта/домена, то лучше создать коллекцию, для этого необходимо ввести имя в поле Collection Name, например Whiskas.



9. Все запросы и коллекции отображаются во вкладке Collections

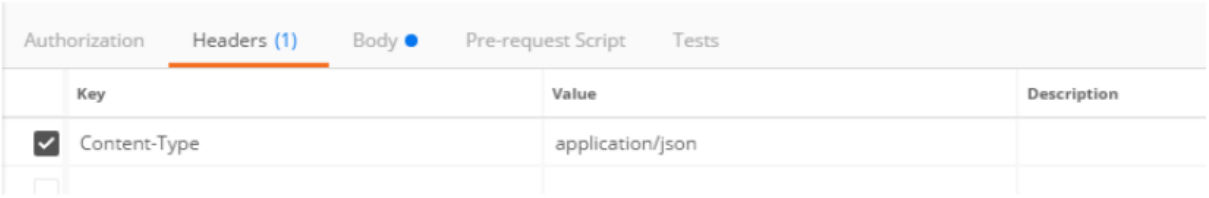


10. Теперь создадим POST запрос. Для создания нового POST необходимо нажать на “плюсик”.



11. Обычно вместе с POST запросом мобильное приложение передает некоторые параметры серверу. Для того чтобы сервер вернул нам правильное значение метода, введем данные которые вместе с запросом будут отправляться на сервер. Для этого необходимо:

a. Выбрать метод POST.
b. Ввести URL запроса, например: https://kitten.whiskas.ru/api/account/login .
c. Во вкладке headers ввести:
   Key = “Content-Type”
   Value = “aplication/json”



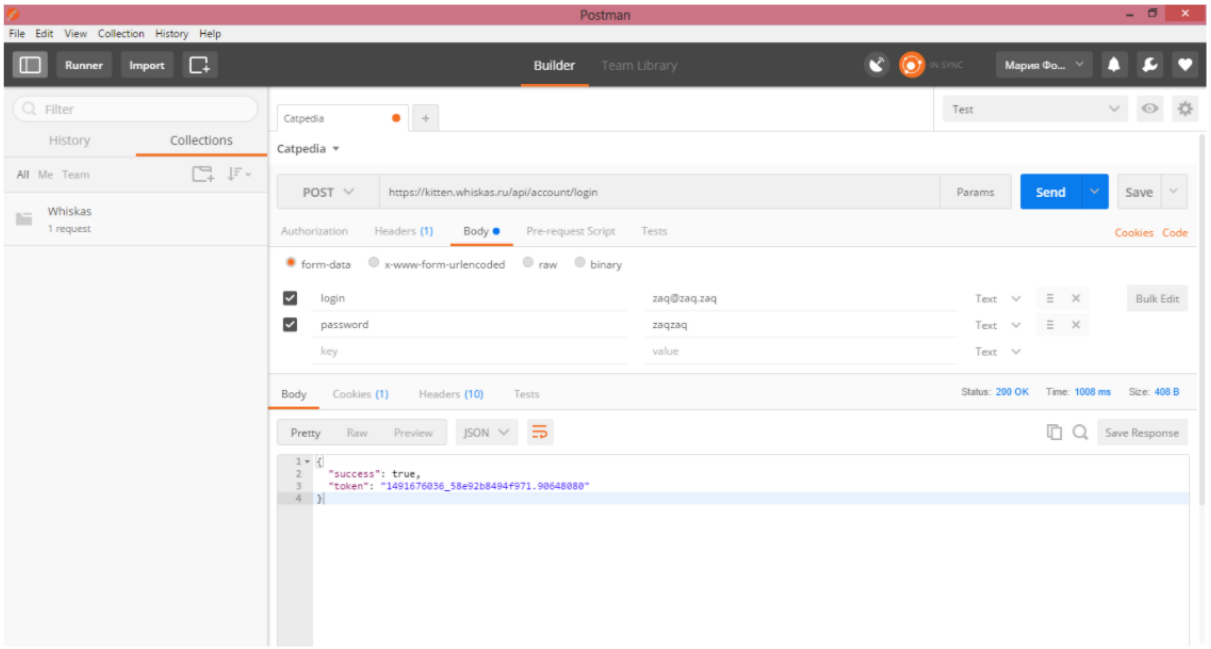
d. Для того чтобы вводить данные необходимо знать какие данные передает приложение серверу.

12. Откроем вкладку Body (под адресной строкой) и выберем тип данных: form-data. Введем параметры ключ - значения

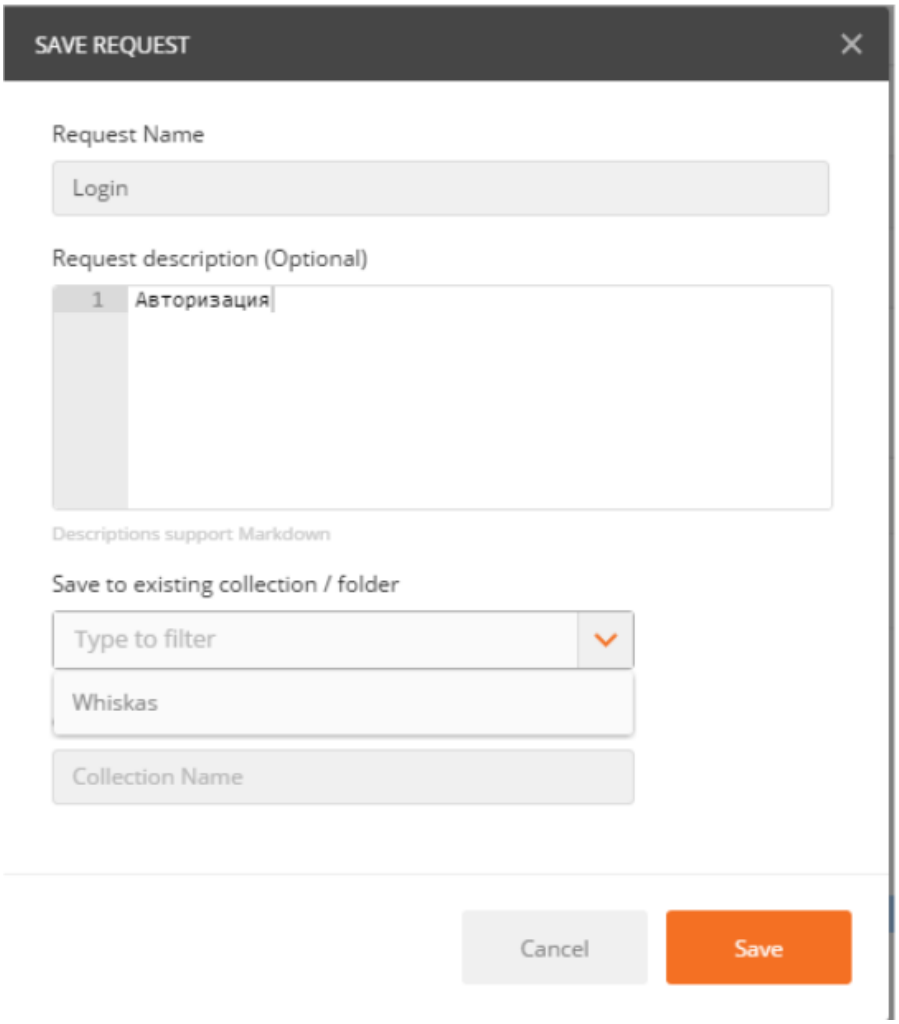
13. Также можно отправлять серверу “сырые данные” т.е. текст в формате JSON. Для этого необходимо во вкладке Body(под адресной строкой) и выбрать тип данных: raw и ввести текст в формате JSON.

```
"login": "zaq@zaq.zaq",
"password": "zaqzaq"
```

14. Нажмем кнопку “Send”



15. Для сохранения запроса необходимо выбрать в выпадающем списке рядом с кнопкой SAVE > SAVE AS.



Практические задания

1. Нужно воспользоваться несколькими сайтами с публичным API и основными методами, которые были изучены на лекции. Можно воспользоваться данным перечнем:
   - https://github.com/public-apis/public-apis
   - https://any-api.com/
   - https://reqres.in/
2. Самостоятельно создать класс для сериализация и десериализация json. После создания класса создать тесты и с помощью них провести проверку на сериализация и

десериализацию. Для это лучше использовать gradle проект.

Материал с примерами и различными аннотациями:

https://www.baeldung.com/jackson

Теоретические вопросы

1. Что такое HTTP ?
2. Из чего состоит HTTP запрос ?
3. За что отвечают методы GET, POST, PUT, DELETE ?
4. Чем отличается код состояния(код класса) 4xx от 5xx ?
5. Что такое Content-type ?
6. Что такое прокси ?
7. Что такое URL ?
8. Структура URL.
9. Что такое REST ?
10. Что такое RESTful ?
11. Кратко рассказать о 6 принципах REST.