Postman

Postman – это платформа API, позволяющая разработчикам проектировать, создавать, тестировать и повторять свои API.

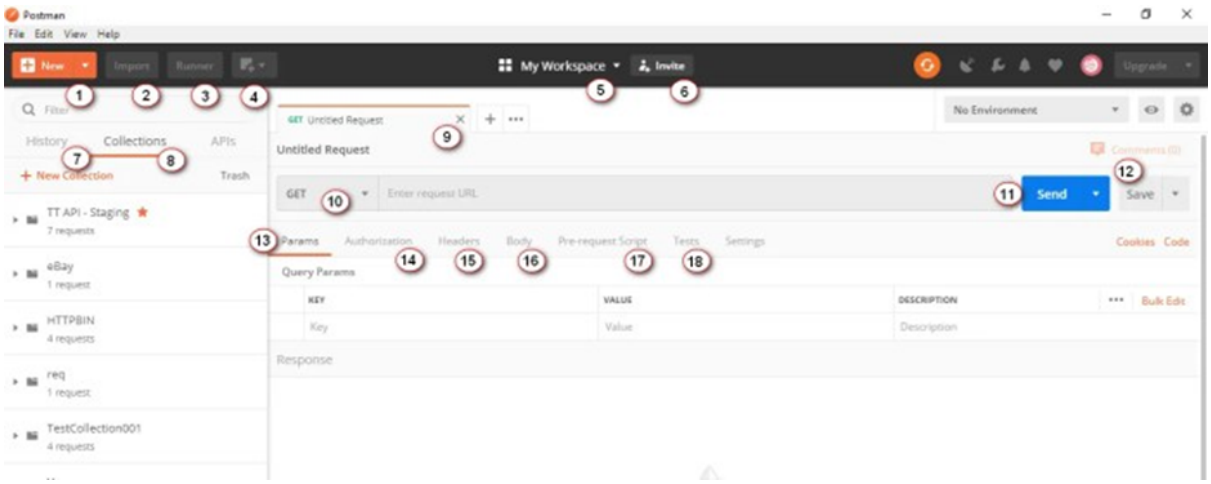
Особенности Postman

- Простой в использовании API клиент
- Функциональный и приятный UI.
- Может использоваться как для ручного, так и для автоматизированного тестирования API.
- Поддерживает интеграции с другими инструментами (например, поддерживает Swagger и RAML)
- Может быть запущен в Windows, Linux, MacOS.
- Не требует знания языков программирования.
- Предоставляет возможность легко экспортировать коллекции запросов, наборы тестов. Можно легко обмениваться этими данными с коллегами.
- Интегрируется с CI/CD инструментами (например, с Jenkins, TeamCity и т.п.)
- API Posman-а подробно документирован.
- Позволяет выполнять API автотесты.

Для работы с серверами программа использует протокол HTTP. Тестировщик отправляет тестовые запросы от клиента на сервер и получает ответ, есть ли ошибка в работе API.

Postman доступен в виде приложения для Windows, Linux и macOS, а также в web-интерфейсе.

Рабочая область Postman



1. **New:**С помощью этой кнопки можно создать новый запрос (Request), коллекцию (Collection) или окружение (Environment).
2. **Import:**С помощью этой кнопки можно импортировать коллекцию или окружение. По нажатию откроется окно, где вы сможете выбрать одну из нескольких опций для импорта: импорт из файла, папки или по ссылке. Также можно просто вставить данные для импорта в текстовое поле.
3. **Runner:**По нажатию на кнопку запускается Collection Runner, который выполняет коллекции запросов.
4. **Open New:**По нажатию открывается новое окно Postman или новое окно запуска коллекций.
5. **My Workspace:**Моя рабочая область. С помощью этой кнопки можно создать новую рабочую область (workspace). Рабочая область предоставляет общий контекст для работы с API. Может использоваться для совместной работы внутри команды (ее можно расшарить с коллегами).
6. **Invite:**С помощью этой кнопки можно пригласить других членов команды для совместной работы внутри рабочей области (workspace-а)
7. **History:**Все запросы и ответы попадают во вкладку «History» (История). Это позволяет вернуться к предыдущим запросам.

8. **Collections:**В этой вкладке хранятся коллекции запросов. Коллекции используются для группировки запросов по каким-либо признакам. Внутри коллекции запросы можно объединить в папки, например по разным версиям API или тестируемым элементам приложения.
9. **Request Tab:**Вкладка запроса. Название вкладки по умолчанию — Untitled Project. Хорошая практика — называть вкладку по названию запроса.
10. **HTTP Request:**С помощью этого выпадающего списка можно выбрать тип запроса: GET, POST, PUT, PATCH, DELETE и т.п.
11. **Request URL (endpoint):**URL API запроса.
12. **Save:**По нажатию на кнопку Save можно сохранить запрос (или перезаписать, если запрос уже был сохранен ранее)
13. **Params:**Параметры, необходимые для выполнения запроса.
14. **Authorization:**API используют авторизацию, чтобы убедиться, что клиент имеет доступ к запрашиваемым данным. В этой секции описываются параметры авторизации: например, username, password, bearer-токен и т.п.
15. **Headers:**Для работы с некоторыми API с каждым запросом необходимо отправлять специальные хедеры. Это нужно для того, чтобы добавить дополнительные данные о типе операции, которую вы хотите провести. Хедеры можно указать в этой секции.
16. **Body:**В этой вкладке указываются данные, которые должны быть отправлены вместе с запросом. Можно отправлять различные типы данных в соответствии с API. По умолчанию стоит none – если не нужно отправлять тело с запросом.
17. **Pre-request Script:**Pre-request скрипты пишутся на JavaScript и выполняются перед отправкой запросов. Используются для того, чтобы провести какие-то действия прямо перед тем, как отправить запрос (например, добавить timestamp или какие-то вычисляемые данные в ваши запросы)
18. **Tests:**Во вкладке Tests находятся скрипты, которые выполняются во время запроса. Тесты позволяют проверить API и убедиться, что все работает так, как это было задумано.

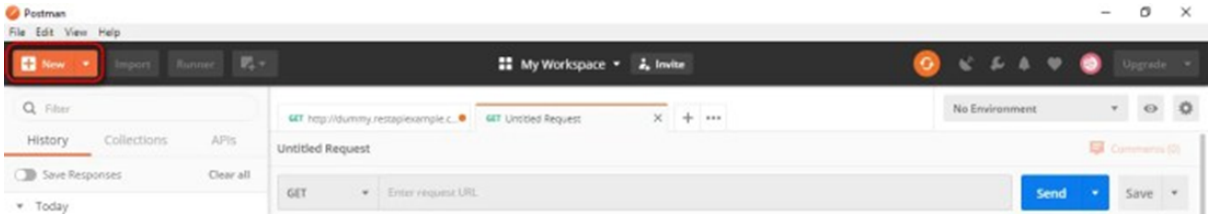
Основные сущности Postman: запросы, коллекции и окружения

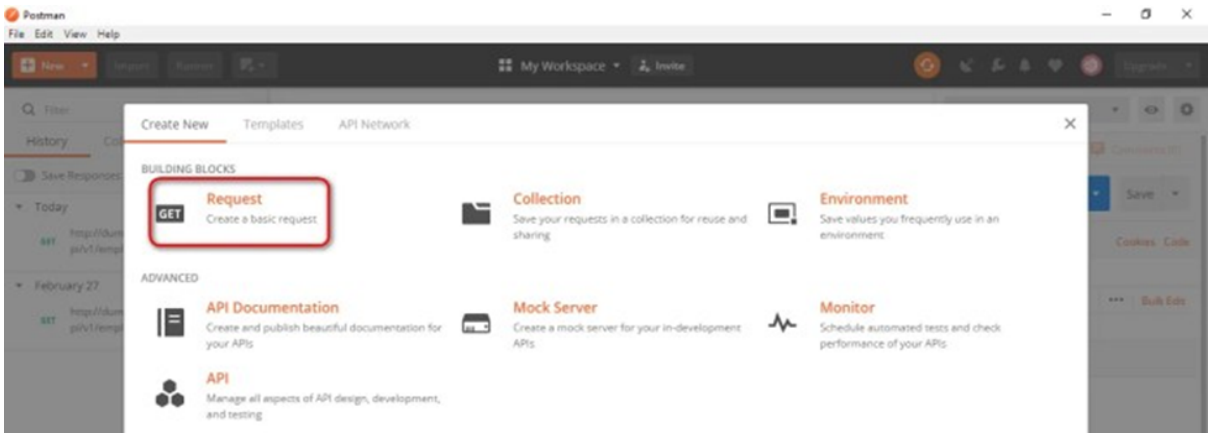
Перед тем, как приступить непосредственно к тестированию, давайте рассмотрим основные сущности, которыми оперирует Postman:

1. Запросы
2. Коллекции
3. Окружения

1. Запросы (Requests)

Запрос представляет собой комбинацию URL, хедеров и Body (тела запроса). Postman позволяет сохранять запросы и использовать их в будущем там, где вам нужно.





Postman позволяет делать запросы к API. С помощью API-запроса можно получать и отправлять данные какому-либо бэкенд-сервису. Для каждого API-запроса нужно выбрать HTTP-method.

Наиболее распространенные типы HTTP-запросов:

1. **GET:** GET-запросы используются для **получения**данных от API. GET-запросы не меняют состояние данных на сервере (не добавляют, не удаляют и не изменяют данные).
2. **POST:**POST-запросы используются для **отправки новых данных**
3. **PUT:**PUT-запросы используются для **обновления уже существующих данных**.

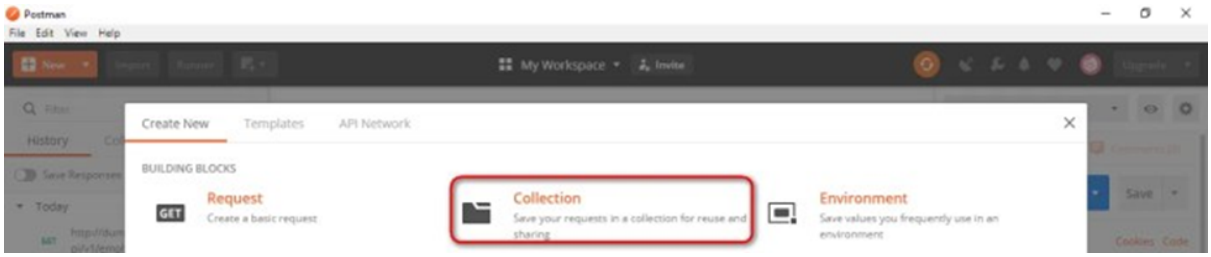
4. **PATCH:**PATCH-запросы (как и PUT) используются для **обновления уже существующих данных**. Разница в том, что с помощью PATCH запросов можно обновить несколько записей за раз.
5. **DELTE:**DELETE-запросы используются для **удаления существующих данных**.

После отправки запроса тестировщик получает ответ в виде кода статуса HTTP. Всего таких статусов 40 в пяти категориях; каждый код помогает понять, правильно ли работает API.

1xx	Информация	Информирование о статусе передачи данных
2xx	Успех	Запрос выполнен успешно
3xx	Перенаправление	Запрос может быть успешно выполнен, но по другому URL
4xx	Ошибка клиента	Указание на ошибку на стороне клиента, из-за которой запрос нельзя выполнить
5xx	Ошибка сервера	Указание на ошибку на стороне сервера, из-за которой он не может обработать запрос

2. Коллекции (Collections)

Коллекции представляют собой группы запросов. Вы можете думать о коллекциях как о папках, в которых лежат запросы.



Коллекция может содержать любое число запросов. Запустить выполнение коллекции можно двумя способами:

1. с помощью Collection Runner
2. с помощью Newman

3. Окружение (Environments)

Окружения в Postman позволяют запускать запросы и коллекции, используя разные наборы данных. Например, мы можем создавать разные окружения в Postman для Dev, QA и Production серверов. В каждом из окружений будут свои собственные настройки: например, URL, auth token-ы и пароли, API-ключи и т.п. Окружения представляют собой наборы пар «ключ-значение».