

# Postman

## Маршруты и Эндпоинты

- **Маршрут (Route - роут)** — это «имя», которое отсылает работу API к определенным эндпоинтам. Если упростить, то можно сказать, что маршрут - это URL к которому можно обратиться разными HTTP методами. Маршрут может иметь несколько эндпоинтов.
- **Эндпоинт (Endpoint - конечная точка)** — это само обращение к маршруту отдельным HTTP методом. Эндпоинт выполняют конкретную задачу, принимают параметры и возвращают данные Клиенту.

## Разберем URL

`http://example.com/wp-json/wp/v2/posts/123` :

- Здесь — это маршрут, а — это базовый путь самого REST API.  
wp/v2/posts/123  
/wp-json
- Этот маршрут имеет 3 эндпоинта:
  - **GET** — запускает метод `get_item()`, и возвращает данные поста Клиенту.
  - **PUT|PATCH|POST** — запускает метод `update_item()`, обновляет данные и возвращает их Клиенту.
  - **DELETE** — запускает метод `delete_item()`, удаляет пост и возвращает только что удаленные данные Клиенту.

## Запрос к корневому маршруту

Если сделать GET запрос к корневому маршруту `http://example.com/wp-json/`, мы получим JSON ответ, в котором видно какие доступны маршруты, и какие доступны эндпоинты для каждого из них. При этом маршрут тут это `/` (корень), а при GET запросе он становится эндпоинтом (конечной точкой).

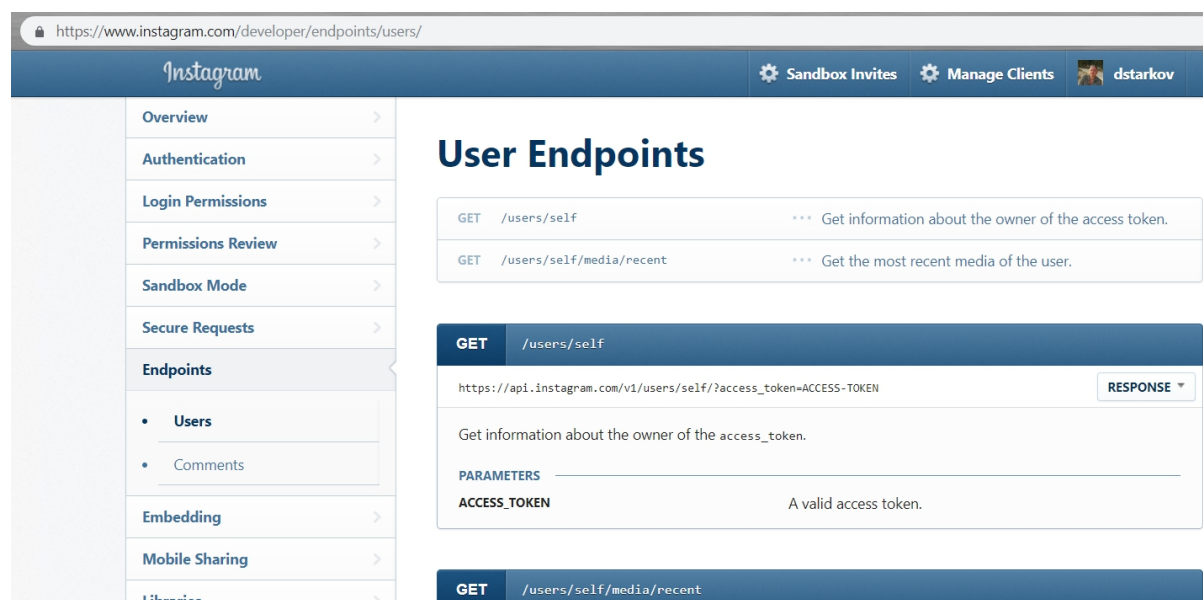
Конечные точки указывают, как получить доступ к ресурсу, а метод указывает разрешенные взаимодействия (такие как GET, POST или DELETE) с ресурсом.

Один и тот же ресурс обычно имеет множество связанных конечных точек, каждая из которых имеет разные пути и методы, но возвращает различную информацию об одном и том же ресурсе. Конечные точки обычно имеют краткие описания, похожие на общее описание ресурса, только еще короче.

Кроме того, конечная точка показывает только конечный путь URL ресурса, а не базовый, общий для всех конечных точек, путь.

## Примеры конечных точек

Вот пример конечной точки ресурса **User** API Instagram



The screenshot shows the Instagram Developer portal at <https://www.instagram.com/developer/endpoints/users/>. The left sidebar contains a navigation menu with items like Overview, Authentication, Login Permissions, Permissions Review, Sandbox Mode, Secure Requests, Endpoints (selected), Users, Comments, Embedding, Mobile Sharing, and Libraries. The main content area is titled 'User Endpoints' and lists two endpoints:

- GET /users/self: \*\*\* Get information about the owner of the access token.
- GET /users/self/media/recent: \*\*\* Get the most recent media of the user.

The first endpoint, GET /users/self, is expanded to show details:

- Method:** GET
- Path:** /users/self
- URL:** `https://api.instagram.com/v1/users/self/?access_token=ACCESS-TOKEN`
- Description:** Get information about the owner of the access\_token.
- Parameters:** ACCESS\_TOKEN (A valid access token.)

Конечная точка обычно выделяется стилизованным образом для придания ей более визуального внимания. Большая часть документации строится вокруг конечной точки, поэтому, может, имеет смысл визуально выделить конечную точку в нашей документации.

Конечная точка, возможно, является наиболее важным аспектом документации API, потому что она является тем, что разработчики будут реализовывать для выполнения своих запросов.

Сила Postman заключается в том, что он позволяет запускать собственный сценарий Javascript до и после доступа к определенному запросу, тем самым напрямую выполняя `chain request`

В результате несколько запросов могут быть объединены в один процесс для выполнения интегрированного теста. Это чрезвычайно полезно во многих операциях API, поэтому здесь необходимо суммировать некоторые часто используемые операторы.

## Pre request

Скрипт, выполняемый перед запросом, — это **скрипт, который выполняется перед каждым запросом в Postman и может изменять параметры запроса от вашего имени.**

### ИСТОЧНИК

Pre request Сценарий - это сценарий перед отправкой запроса к API, который обычно используется для динамической генерации параметров, пакетов данных JSON, адресов ссылок и т. Д.

## Написание сценариев предварительного запроса

Вы можете использовать сценарии предварительного запроса в Postman для выполнения JavaScript перед выполнением запроса. Включив код на вкладке « **Сценарий предварительного запроса** » для запроса, коллекции или папки, вы можете выполнить предварительную обработку, такую ​​как установка значений переменных, параметров, заголовков и данных тела. Вы также можете использовать сценарии предварительного запроса для отладки кода, например, выводя выходные данные в консоль.

## Пример сценария предварительного запроса

Пример использования сценария предварительного запроса может быть следующим:

- У вас есть серия запросов в коллекции, и вы выполняете их в определенной последовательности, например, при использовании средства запуска коллекции
- .
- Второй запрос зависит от значения, возвращенного из первого запроса.
- Значение необходимо обработать, прежде чем передать его второму запросу.
- Первый запрос устанавливает значение данных из поля ответа в переменную в сценарии **Tests** .
- Второй запрос извлекает значение и обрабатывает его в своем **сценарии предварительного запроса**  
, затем устанавливает обработанное значение в переменную (на которую ссылается второй запрос, например, в его параметрах).

## Сценарий перед запуском вашего запроса

Чтобы включить код, который вы хотите выполнить до того, как Postman отправит запрос:

1. Выберите « **Коллекции** » на боковой панели.
2. Откройте запрос, затем выберите вкладку « **Сценарий предварительного запроса** ».
3. Введите код JavaScript, который необходимо обработать перед выполнением запроса, затем выберите **Сохранить** .
4. Выберите **Отправить** , чтобы отправить запрос. Код будет выполнен до того, как Postman отправит запрос в API.



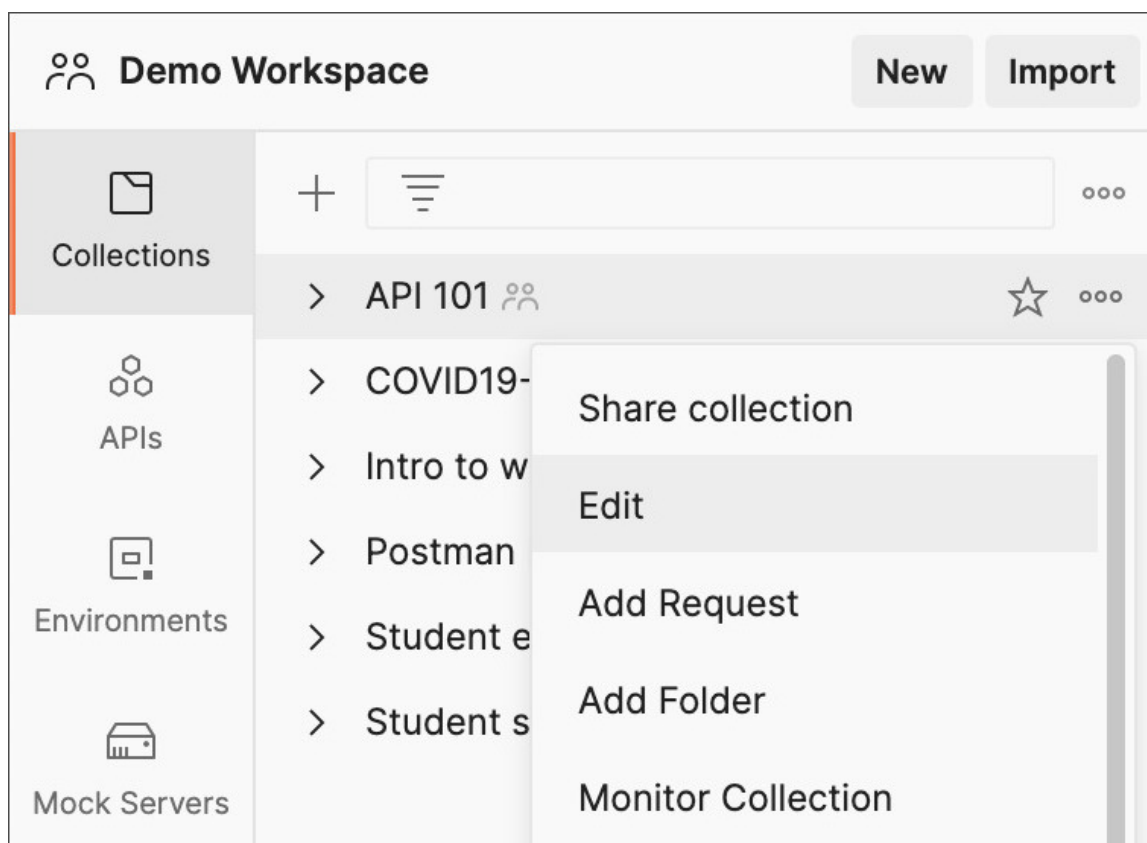
# Повторное использование сценариев предварительного запроса

Вы можете добавлять сценарии предварительного запроса ко всем коллекциям и папкам в коллекциях. В обоих случаях ваш сценарий предварительного запроса будет выполняться перед каждым запросом в коллекции или прямым дочерним запросом в папке. Это позволяет вам определить часто используемые шаги предварительной обработки или отладки, которые необходимо выполнить для нескольких запросов.

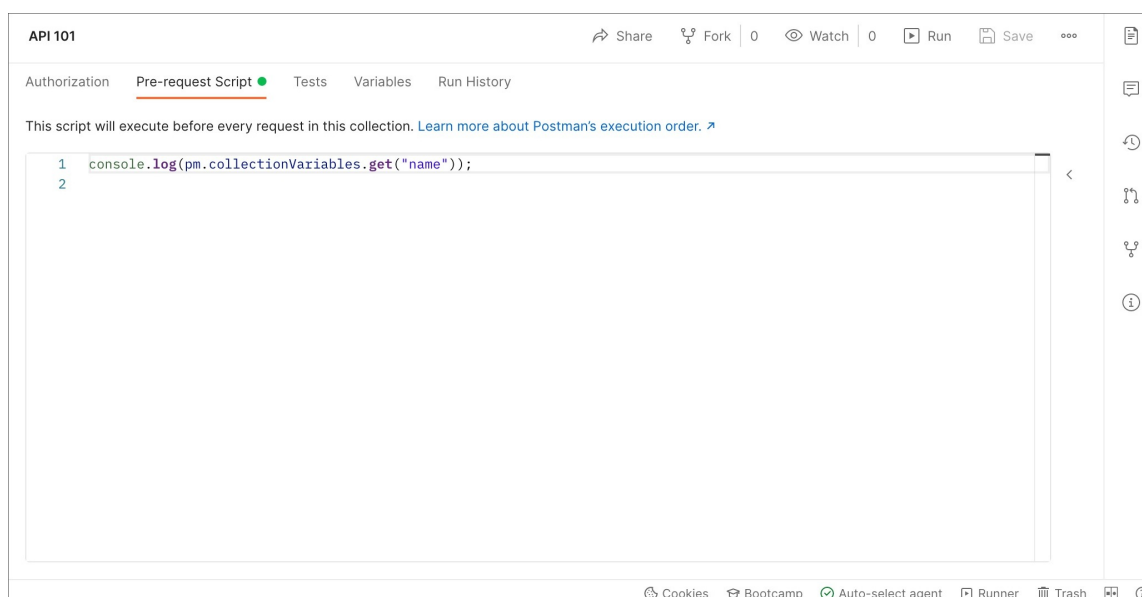
Вы можете определить сценарий предварительного запроса при первом создании коллекции или папки или в любое время после этого.

Чтобы добавить сценарии предварительного запроса в коллекцию или папку:

1. Выберите «**Коллекции**» на боковой панели.
2. Щелкните значок дополнительных действий , затем выберите **Изменить**



3. Выберите вкладку « **Сценарии предварительного запроса** ». Введите код, который будет выполняться перед каждым запросом в коллекции или прямым дочерним запросом в папке.

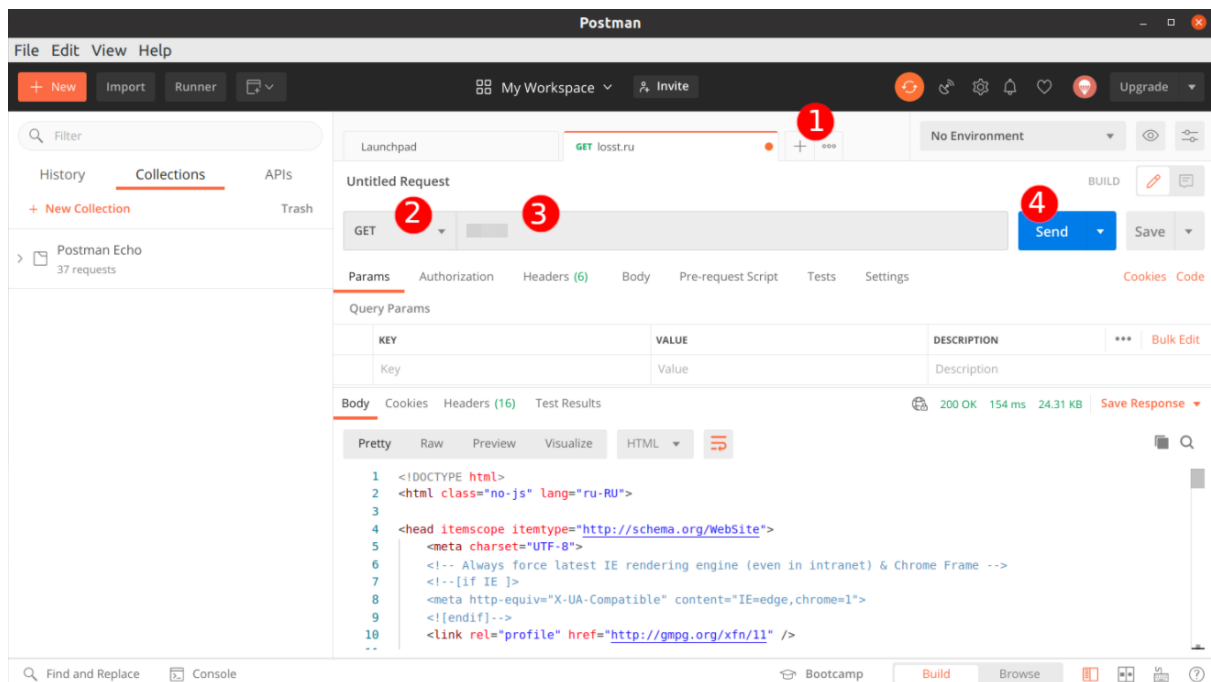


4. Выберите **Сохранить** .

Источник

## Выполнение запроса

Выполнение простого запроса, без сохранения в коллекции, осуществляется кликом по кнопке со значком <+>. В результате откроется новая вкладка, где есть возможность выбрать тип запроса (GET или POST) и внести домен, который будет открываться. Остается нажать на кнопку Send, которая и запустит процедуру проверки.



В нижней части страницы появится код страницы (HTML). Здесь имеется несколько вкладок:

**Body** – данные, содержащиеся в теле запроса.

**Cookie** – информация, записанная сервером.

**Headers** – заголовки, которые были возвращены.

На первой вкладке, где отображается тело запроса, есть выбор нескольких вариантов отображения. Так, Pretty интересна для получения JSON-данных – программа отформатирует их в достаточно удобном формате. Если выбрать режим Raw, информация будет представлена «как есть», без каких-либо изменений. Вкладка Preview отображает сайт в том виде, в котором он открывается в браузере.

**Tests** - Во этой вкладке находятся скрипты, которые выполняются во время запроса. Тесты позволяют проверить API и убедиться в том, что все работает как и было задумано.