

# Документация по библиотеке Python Requests¶

**Requests** – это элегантная и простая HTTP-библиотека для Python, созданная для людей.

Узрите силу Requests:

```
>>> r = requests.get('https://digitology.tech/', auth=('user', 'pass'))
>>> r.status_code
200
>>> r.headers['content-type']
'application/json; charset=utf8'
>>> r.encoding
'utf-8'
>>> r.text
'{"type": "User"... '
>>> r.json()
{'private_gists': 419, 'total_private_repos': 77, ...}
```

Аналогичный код, без Requests:

```
import urllib2

gh_url = 'https://digitology.tech/'

req = urllib2.Request(gh_url)

password_manager = urllib2.HTTPPasswordMgrWithDefaultRealm()
password_manager.add_password(None, gh_url, 'user', 'pass')

auth_manager = urllib2.HTTPBasicAuthHandler(password_manager)
opener = urllib2.build_opener(auth_manager)

urllib2.install_opener(opener)

handler = urllib2.urlopen(req)

print(handler.getcode())
print(handler.headers.getheader('content-type'))

# -----
# 200
# 'application/json'
```

**Requests** позволяет очень легко отправлять HTTP/1.1 запросы. Нет необходимости вручную добавлять строки запроса к вашим URL-адресам или кодировать POST данные. Сохранение активности (Keep-Alive) и пулы HTTP-соединений на 100% автоматические, благодаря *urllib3*.

## Любимые функции¶

Requests готов к работе уже сейчас.

- Keep-Alive и пул подключений
- Международные домены и URL-адреса



- Сесансы с сохранением cookie файлов
- Проверка SSL в стиле браузера
- Автоматическое декодирование контента
- Базовая/дайджест-аутентификация
- Элегантные cookie с ключом/значением
- Автоматическая декомпрессия
- Автоматическая декодирование тела ответа
- Поддержка HTTP(S) прокси
- Многостраничная загрузка файлов
- Потокное скачивание
- Таймауты подключения
- Разбивка на части запоросов
- Поддержка .netrc

Requests официально поддерживает Python 2.7 и 3.6+ и отлично работает на PyPy.

## Руководство пользователя¶

Эта часть документации, в основном прозаическая, начинается с некоторой справочной информации о Requests, а затем посвящена пошаговым инструкциям по максимально эффективному использованию Requests.

- [Установка Requests](#)
  - `$ python -m pip install requests`
  - [Получить исходный код](#)
- [Быстрый старт](#)
  - [Выполнить запрос](#)
  - [Передача параметров в URL-адресах](#)
  - [Содержание ответа](#)
  - [Контент двоичного ответа](#)
  - [Содержимое JSON ответа](#)
  - [Необработанный контент ответа](#)
  - [Пользовательские заголовки](#)
  - [Более сложный POST запросы](#)
  - [POST файл с многократным кодированием](#)
  - [Коды состояния ответа](#)
  - [Заголовки ответов](#)
  - [Cookie](#)
  - [Перенаправление и история](#)
  - [Таймауты](#)
  - [Ошибки и исключения](#)
- [Продвинутое использование](#)
  - [Объекты Session](#)
  - [Объекты запроса и ответа](#)
  - [Подготовка Requests](#)
  - [Подтверждение сертификата SSL](#)

- Сертификаты на стороне клиента
- Сертификаты CA
- Рабочий процесс с содержимым body
- Кеер-Alive
- Поточковая загрузка
- Фрагментированное кодирование Requests
- POST несколько файлов с многократным кодированием
- Хуки событий
- Пользовательская аутентификация
- Стриминг Requests
- Прокси
- Согласие
- HTTP-глаголы
- Пользовательские глаголы
- Заголовки ссылок
- Транспортные адаптеры
- Блокирующий или неблокирующий?
- Порядок заголовков
- Таймауты
- Аутентификация
  - Базовая аутентификация
  - Дайджест-аутентификация
  - OAuth 1 Аутентификация
  - OAuth 2 и аутентификация OpenID Connect
  - Другая проверка подлинности
  - Новые формы аутентификации

## Руководство от сообщества ¶

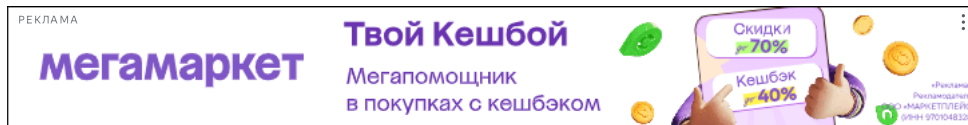
Эта часть документации, в основном прозаическая, подробно описывает экосистему и сообщество Requests.

- Рекомендуемые пакеты и расширения
  - Пакет Certifi
  - CacheControl
  - Requests-Toolbelt
  - Requests-Threads
  - Requests-OAuthlib
  - Бетамакс
- Часто задаваемые вопросы
  - Поддерживается ли декодирование данных?
  - Можно ли установить свой User-Agent?
  - Почему не HttpLib2?
  - Поддерживается ли Python 3?
  - Поддержка Python 2?
  - Что за ошибка «hostname doesn't match (имя хоста не совпадает)»?
- Интеграции
  - Python для iOS
- Статьи и обсуждения
- Служба поддержки
  - Stack Overflow
  - Зарегистрируйте проблему
- Раскрытие уязвимости

- [Процесс](#)
- [Предыдущие CVE](#)
- [Процесс и правила релиза](#)
  - [Основные релизы](#)
  - [Незначительные релизы](#)
  - [Релизы исправлений](#)
  - [Рассуждение](#)
- [Обновления сообщества](#)
- [Release History](#)

## Документация/руководство по API ¶

Если вы ищете информацию о конкретной функции, классе или методе, эта часть документации для вас.



- [Интерфейс разработчика](#)
  - [Основной интерфейс](#)
  - [Исключения](#)
  - [Сеансы Requests](#)
  - [Классы нижнего уровня](#)
  - [Классы низшего-нижнего Уровня](#)
  - [Аутентификация](#)
  - [Кодировки](#)
  - [Cookies](#)
  - [Поиск кода состояния](#)
  - [Переход на 1.x](#)
  - [Переход на 2.x](#)

## Руководство для авторов ¶

Если вы хотите внести свой вклад в проект, эта часть документации для вас.

- [Руководство для авторов](#)
  - [Будьте доброжелательны](#)
  - [Получить раннюю обратную связь](#)
  - [Пригодность вклада](#)
  - [Вклад в код](#)
    - [Шаги по отправке кода](#)
    - [Проверка кода](#)
    - [Новые участники](#)
    - [Кодовый стиль Кеннета Рейца™](#)
  - [Документация](#)
  - [Отчёты об ошибках](#)
  - [Функциональность Requests](#)
- [Авторы](#)
  - [Хранители кристаллов](#)
  - [Предыдущие Хранители кристаллов](#)
  - [Патчи и предложения](#)

Руководство завершено. Удачи.

| [следующий](#) »

Нашли ошибку? Выделите текст и нажмите **Shift+Enter**.

© Авторские права Digitology.tech

Последнее обновление: авг. 10, 2023