**Аннотация**

*Сегодня мы посмотрим, как разместить свой проект в Интернете, чтобы к нему могли получить доступ все желающие, а также изучим несколько сервисов, которые смогут нам в этом помочь.*

*Этот урок не содержит задач.*

**Введение**

Разрабатывать веб-приложения, которые «живут» только на нашем локальном компьютере, согласитесь, не очень весело. Само слово «веб» подразумевает, что к результатам нашей работы должны иметь доступ все пользователи сети Интернет, ну или те, которым мы даем на это право. Процесс размещения ресурсов в Интернете называется deploy.

Пока наше flask-приложение размещено на локальном компьютере, и мы помним, что каждый компьютер в мире думает про себя, что его IP-адрес 127.0.0.1. Как же тогда пользовательский запрос из Интернета может добраться до нашего компьютера и получить нужную веб-страницу или json-ответ от сервиса? Ведь таких машин с адресом 127.0.0.1 миллионы!

Для решения этой задачи есть несколько способов и множество разных сервисов. С некоторыми из них мы сегодня познакомимся.

Давайте сделаем простейшее веб-приложение, на примере которого мы будем проверять различные способы организации доступа к веб-приложению через Интернет.

import os

from flask import Flask

app = Flask(\_\_name\_\_)

@app.route("/")

def index():

return "Привет от приложения Flask"

if \_\_name\_\_ == '\_\_main\_\_':

port = int(os.environ.get("PORT", 5000))

app.run(host='0.0.0.0', port=port)

Из нового: os.environ пытается получить значение переменной окружения с именем PORT, а если не получается, то выставляет порт 5000. Это нам пригодится позднее.

host = 0.0.0.0 означает, что мы разрешаем подключения из любой сети по любому интерфейсу.

**Туннель в Интернет**

Очень часто при отладке веб-приложений хочется посмотреть, что происходит, когда в приложение приходят запросы из Интернета, или продемонстрировать результаты работы другу/заказчику. Не всегда удобно ради тестирования или ради одноразовой демонстрации арендовать и настраивать сервер в Интернете.

Согласитесь, было бы удобно получить какой-то временный адрес в Интернете «в один клик» для таких целей. К счастью, для этого есть сервисы, которые организуют виртуальный туннель из Интернета на ваш локальный компьютер. Один из них — [ngrok](https://ngrok.com/).

Важная его особенность: у ngrok есть бесплатный план, но даже этой ограниченной версии хватит для наших целей. Сначала надо зарегистрироваться и залогиниться на [сайте](https://ngrok.com/) сервиса.

На странице ["Setup & Installation"](https://dashboard.ngrok.com/get-started/setup) приведена подробная инструкция по настройке и запуску утилиты.

1. Необходимо скачать исполняемый файл приложения для нужной операционной системы со [страницы загрузки](https://ngrok.com/download).

2. Распаковать архив и в окне терминала перейти в папку с распакованной утилитой.

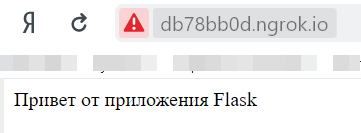
3. Скопировать со страницы "Setup & Installation" команду для настройки работы ngrok с вашей учетной записью и запустить ее в терминале. Выглядит она так ngrok authtoken 25srTxReQIIwsPMX67ZiY1E5TMH\_7KbpfVGLcg9rRDd2hwfk9

4. Настройка завершена.

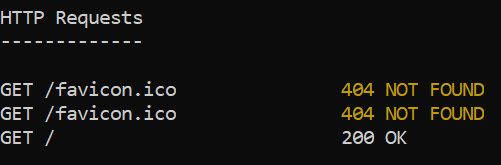
Теперь запустим наше тестовое приложение в PyCharm или через командную строку. Выполним в терминале команду:

ngrok http 5000

После этого ngrok создаст временный адрес в Интернете вида набор\_символов.ngrok.io и виртуальный туннель с этого адреса на пятитысячный порт нашего локального компьютера. Теперь все запросы по протоколу http и https, которые придут на созданный временный адрес, будут перенаправляться на наш компьютер. Давайте проверим. Перейдем по этому адресу (скопируем и вставим его в строку запроса браузера) и посмотрим на результат.



Все работает! А в консоли ngrok появилось сообщение, что GET-запрос успешно обработан:



Таким образом можно организовывать тестирование приложения в Интернете. Особенно такой подход удобен, когда для проверки функциональности приложения надо получить запрос извне. В дальнейшем мы будем использовать эту методику при отладке приложений для голосового помощника Алисы.

**Heroku**

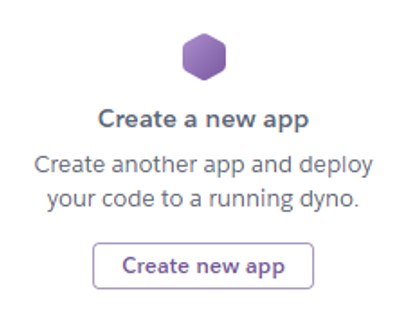
Более продвинутый способ, который (при использовании платных планов) подойдет для «боевого» разворачивания приложения в Интернете (деплоя) — аренда контейнеров для приложений на одном из множества сервисов, которые предоставляют такие услуги.

Сервисов действительно очень много, и все они разнятся как в стоимости (некоторые предлагают бесплатный тариф, который подойдет для приложения с небольшим количеством пользователей), так и в сложности настроек для запуска.

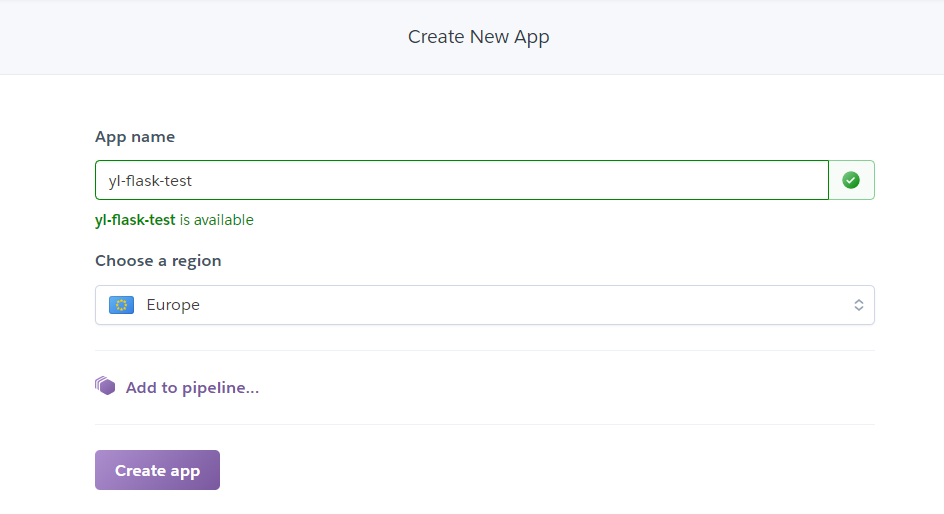
В качестве одного из самых легких стоит привести в пример [pythonanywhere.com](https://www.pythonanywhere.com/), который содержит ряд преднастроенных шаблонов для различных приложений, разработанных на Python. К недостатку этого сервиса можно отнести то, что приложения, размещенные на нем, могут обращаться только к тем ресурсам, которые были внесены в «белый список» доступных адресов.

Немного более сложным в обращении является сервис [Heroku](https://dashboard.heroku.com/apps). Он поддерживает приложения не только на Python, но и на других языках программирования, различные базы данных и разные преднастроенные сторонние приложения.

Зарегистрируйтесь на Heroku. После чего создайте новое приложение по адресу <https://dashboard.heroku.com/apps>



Придумайте приложению имя и выберите расположение.



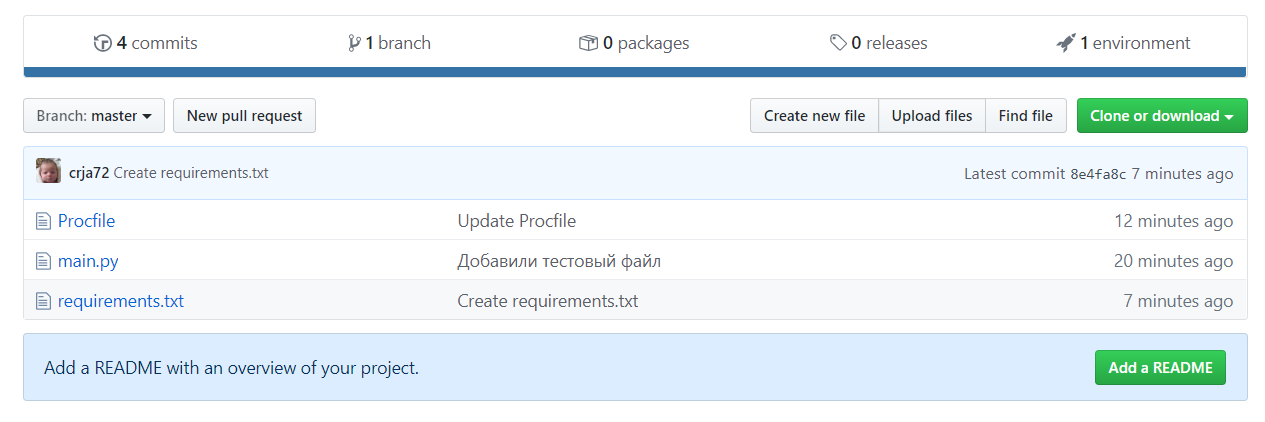
Есть несколько способов разместить свое приложение на Heroku:

* Через консольный клиент Heroku CLI
* Подключив свой репозиторий на GitHub

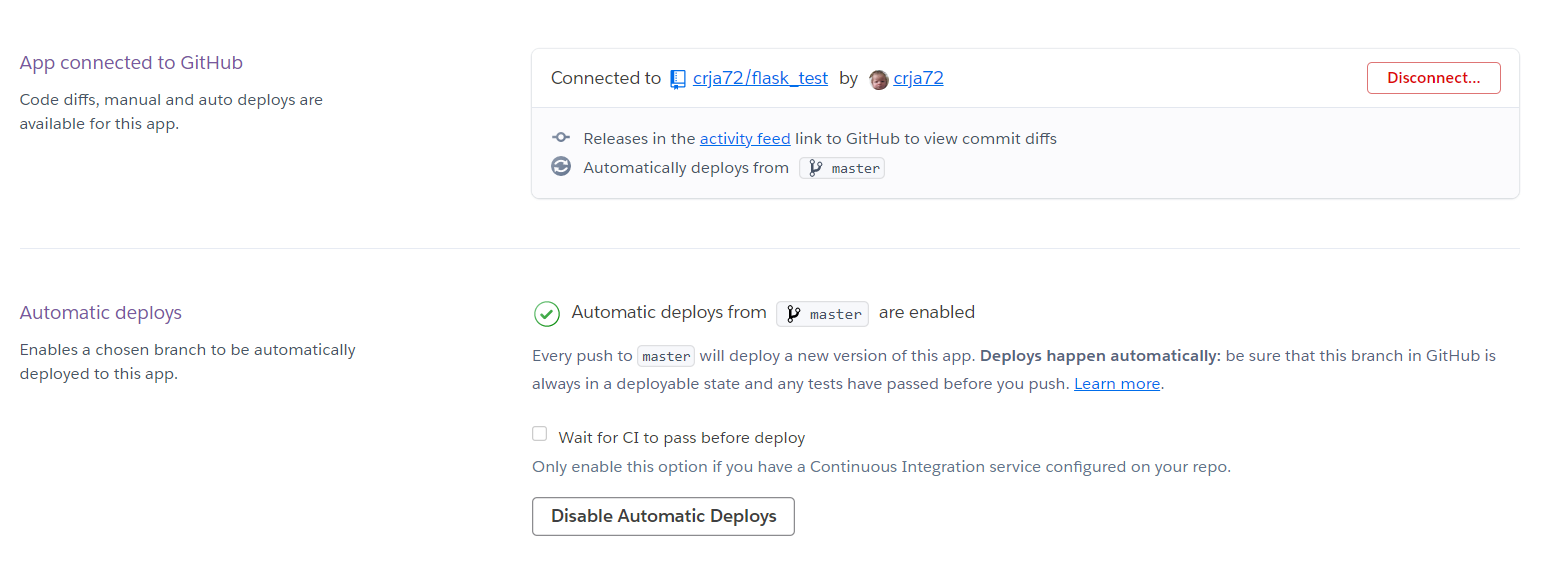
Давайте пойдем по второму пути и создадим репозиторий (не имеет значение публичный или приватный), в который **запушим** файл с нашим простейшим веб-приложением, а также еще несколько файлов. Первый из них — служебный текстовый файл Procfile (без расширения), в котором для Heroku будет указано, какой файл запускать, с текстом:

web: python main.py

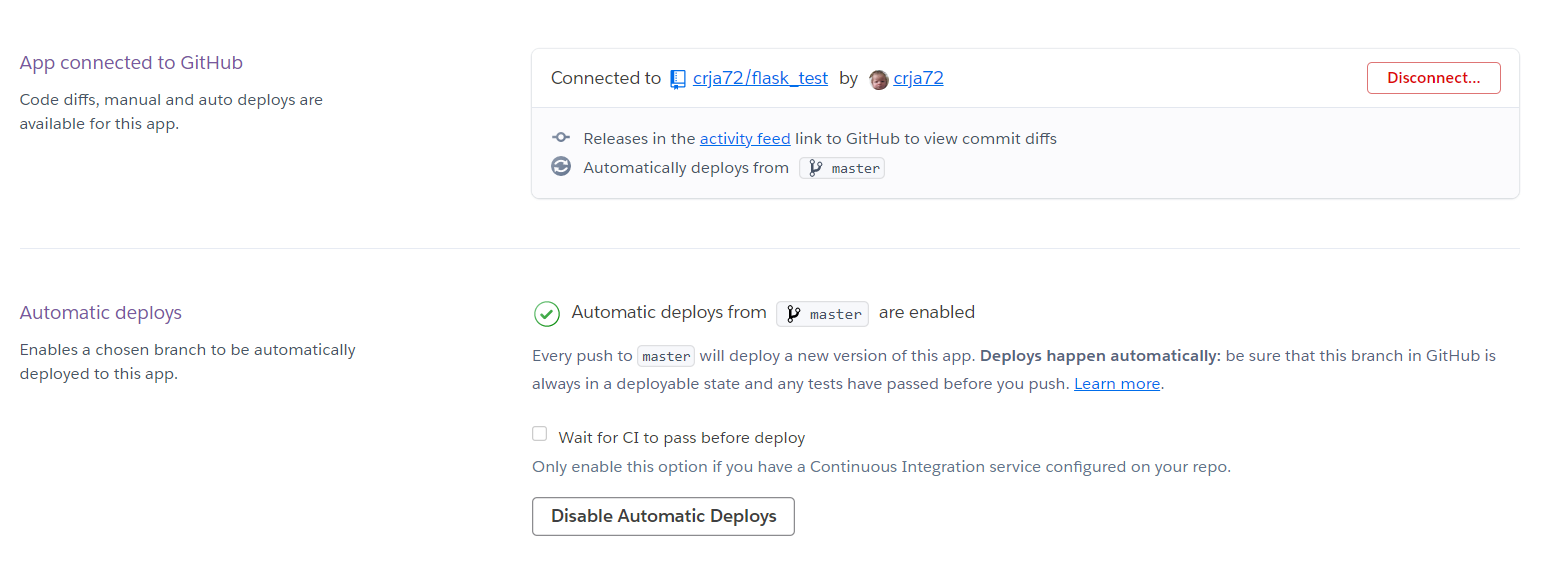
Второй — уже знакомый нам из раздела PyQT requirements.txt, в котором необходимо указать все используемые сторонние модули (flask).



Подключим GitHub к Heroku и выберем нужный репозиторий, а затем и ветку, с которой будем забирать для размещения код. Это можно сделать автоматически или в ручном режиме.



Если не включено автоматическое размещение кода, надо нажать на кнопку Delpoy branch.



Все. Можно переходить по адресу имя\_приложения.herokuapp.com и наслаждаться результатом.

При размещении приложений на Heroku очень важно указывать порт, на котором будет работать ваше приложение именно так, как мы сделали в первом примере этого урока.

if \_\_name\_\_ == '\_\_main\_\_':

port = int(os.environ.get("PORT", 5000))

app.run(host='0.0.0.0', port=port)

Когда приложение разворачивается в Heroku, для него будет выделен порт и установлен в переменную окружения. Важно, чтобы ваше приложение слушало именно этот порт, иначе оно не заработает.

**Яндекс.Облако**

Самым сложным, но одновременно самым гибким и продвинутым вариантом является размещение приложения на облачной виртуальной машине с использованием сервиса Яндекс.Облако или аналогов.

Создание и настройка таких виртуальных машин требует определенных усилий и несколько сложнее, чем сервисы, которые мы рассматривали ранее, но зато вы сможете полностью контролировать результат и реализовывать разные продвинутые сценарии разворачивания вашего приложения.

Для начала необходимо зайти на [Яндекс.Облако](http://cloud.yandex.ru/) и нажать кнопку «Подключиться» в правом верхнем углу экрана. Далее алгоритм работы можно описать так:

1. Необходимо создать виртуальную машину
2. Выбрать и установить образ операционной системы, которую вы планируете использовать. Можно использовать как «чистый» образ (только операционная система), так и подготовленный командой Яндекс.Облака или сторонними разработчиками
3. Установить все необходимое ПО: интерпретатор Python, все библиотеки, системные утилиты, которые обеспечат работу при большой нагрузке и т. д.
4. Загрузить разработанный вами сервер и ...
5. Поддерживать работу вашего ресурса

Поверьте, это далеко не просто.

Поэтому мы немного вам помогли. Мы подготовили для вас видео, в котором рассказываем, как сделать базовую настройку сервера и установить на него ваше приложение.