FastAPI vs Django vs Flask

Если несколько лет назад (до 2018) вопрос выбора фреймворка включал только Django и Flask, то с появлением FastAPI разработчики начали присматриваться к нему и выбирать FastAPI в качестве основного инструмента создания бэкенда приложений. Давайте рассмотрим особенности каждого из фреймворков.

| Django

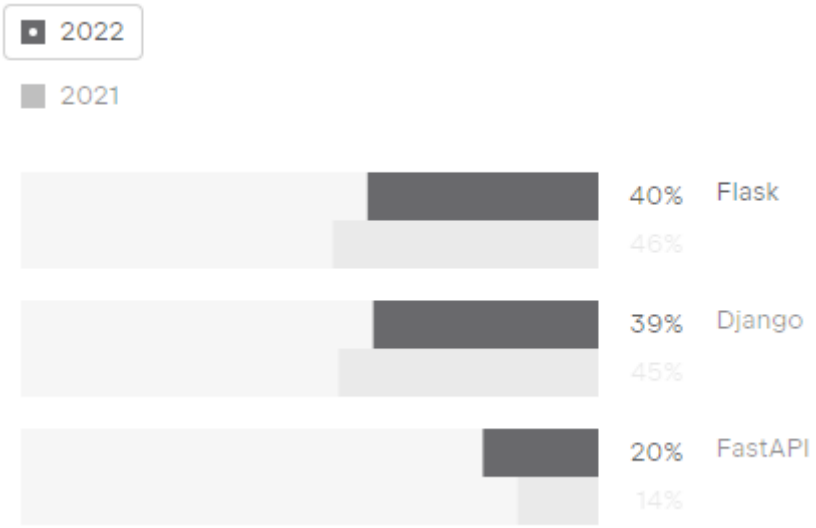
Django появился в 2005 году и быстро завоевал любовь разработчиков. В этом фреймворке есть практически все, что нужно для построения полноценного веб-приложения:

- своя библиотека (ORM) для работы с реляционными базами данных
- админка для работы с базой данных в веб-интерфейсе
- аутентификация и авторизация
- шаблонизация для построения динамических HTML страниц
- кэширование
- сериализация данных
- локализация на другие языки
- возможность писать свой middleware
- библиотека DRF для написания REST API на Django

> Если какие-то из терминов вам непонятны — не переживайте, мы рассмотрим их дальше в курсе.

Для Django существует огромное количество всевозможных плагинов для расширения функционала приложения, а также огромное коммьюнити. Вы почти всегда найдете ответ на свой вопрос, насколько легкий или сложный он бы ни был.

Согласно опросу компании JetBrains, в 2022 году Django делил с Flask 1-ое место по популярности среди Python веб фреймворков.



Django по-прежнему используется во многих компаниях, т.к. с момента главного конкурента — FastAPI, прошло еще не так много времени. Большие коммерческие проекты могут занимать десятки тысяч строк кода, и перейти с устаревающего Django на современный FastAPI для многих компаний означает потерю времени и денег.

| Flask

Перед тем как рассматривать самый быстрорастущий фреймворк последних лет — FastAPI, давайте взглянем на Flask, появившийся в 2010 году. Давайте посмотрим, что он нам предлагает:

- возможность писать API на Python
- шаблонизация для построения динамических HTML страниц (Jinja)

Как-то не густо по сравнению с Django, не правда ли? Тем не менее, отсутствие встроенных и навязываемых фреймворком библиотек и архитектуры приложения привлекло многих разработчиков. Flask давал то, чем не мог похвастаться Django — свободой выбора. Разработчики могли подключить любимый ORM для работы с базой данных (в то время большую популярность начала приобретать SQLAlchemy), а также использовать нереляционные базы данных (которые начали набирать популярность в 2010-х годах), можно было самостоятельно настраивать авторизацию, кэширование, сериализацию и валидацию данных.

Для Flask также написано большое количество сторонних библиотек/плагинов, упрощающих работу с фреймворком. Например, библиотека Flask-Admin для создания админки заменила для пользователей Flask джанговскую админку, а Flask-SQLAlchemy — знаменитый джанговский ORM.

Flask и по сей день используется на многих проектах по той же причине, что и Django — переводить старые проекты на новые фреймворки (Flask -> FastAPI) довольно долго и дорого.

| FastAPI

В конце 2018 года разработчик с никнеймом Tiangolo выпускает FastAPI — веб-фреймворк для построения RESTful API на Python. Ключевые особенности фреймворка:

- поддержка асинхронности
- простота использования и быстрота освоения
- встроенная сериализация и валидация данных через библиотеку Pydantic
- автоматически генерирующаяся документация к API в формате OpenAPI (ранее Swagger)
- отличная документация, понятная новичкам в веб-разработке
- поддержка вебсокетов, не сравнимая по скорости с Django и Flask

Казалось бы, помимо асинхронности фреймворк не предлагал чего-то совершенно нового. Почему FastAPI стал набирать популярность? На мой взгляд, основными плюсами фреймворка стали асинхронность, быстрота освоения и документация к API. Кажется, асинхронности не хватало предыдущим веб-фреймворкам, чтобы удержать свое лидерство. Она действительно дает сильный буст в производительности приложения и позволяет обрабатывать большее количество клиентов одновременно. Помимо этого, снижаются затраты на "железо", ведь теперь вместо 30 серверов (можно купить 10 и все будет работать также за счет асинхронности. FastAPI, также как и Flask, дает свободу разработчикам в выборе технологий и библиотек для создания админки, кэширования, работе с БД и пр.

Сейчас все большую популярность набирают микросервисы в противовес монолитам, когда код приложения может быть разнесен по нескольким небольшим сервисам. В

таком случае сервисы могут разрабатываться независимо разными людьми, что ускоряет разработку. Сервисы общаются между собой через HTTP или очереди сообщений (Redis, RabbitMQ). FastAPI все чаще встречается мне в вакансиях в крупные российские компании (банки, девелоперы и IT-гиганты) на новые проекты.