

## АСИНХРОННЫЙ FLASK 2.0

В ходе работы была поставлена цель продемонстрировать возможности последнего обновления микрофреймворк `aflask`, которое позволяет иначе подойти к работе в разработке веб-приложений

Flask – фреймворк для создания веб-приложений на языке программирования Python, использующий набор инструментов Werkzeug, а также шаблонизатор Jinja2. Относится к категории так называемых микрофреймворков – минималистичных каркасов веб-приложений, сознательно предоставляющих лишь самые базовые возможности.

Flask 2.0 вышел в 2021 г. и добавляет встроенную поддержку асинхронных маршрутов (routes), обработчиков ошибок (error-handlers), функций до (beforerequest) и после (afterrequest) запроса, а также обратных вызовов разрыва! Появляется возможность создавать обработчиков асинхронных маршрутов (routes), используя `async/await`. Чтобы запускать асинхронный код на Python, необходим цикл с ожиданием событий (eventloop) и запуском сопрограмм (coroutines). Flask 2.0 берет на себя включение асинхронного цикла с ожиданием событий (как обычно, это делается через `asyncio.run()` для запуска сопрограмм. При обработке функции маршрута `async` появляется новый подпоток (sub-thread). Внутри этого подпотока будет выполняться цикл с ожиданием событий для запуска обработчика маршрута (сопрограммы). До появления Flask 2.0 для выполнения асинхронного кода требовалось делать новые асинхронные циклы с ожиданием событий в каждом обработчике маршрута, что требовало запуска приложения Flask с использованием потоковых процессов-исполнителей.

Асинхронный подход действительно приносит дивиденды, когда нужно сделать ряд HTTP-запросов на внешний веб-сайт или API. Каждому запросу потребуется значительное время на получение ответа. Вследствие данного времени ожидания пользователи посчитают веб-приложение медленным или инертным.

### ЛИТЕРАТУРА

1. Кирилл Складенко- Асинхронный Flask, 2021
2. Patrick Kennedy – Flask 2.0, 2021
3. Хабр [Электронный ресурс] / Режим доступа – <https://habr.com/post/557250>