ベンチマーク

TechEmpowerの独立したベンチマークでは、Uvicornの下で動作する**FastAPI**アプリケーションは、利用可能な最速のPythonフレームワークの1つであり、下回っているのはStarletteとUvicorn自体 (FastAPIによって内部で使用される) のみだと示されています。

ただし、ベンチマークを確認し、比較する際には下記の内容に気を付けてください。

ベンチマークと速度

ベンチマークを確認する時、異なるツールを同等なものと比較するのが一般的です。

具体的には、Uvicorn、Starlette、FastAPIを (他の多くのツールと) 比較しました。

ツールで解決する問題がシンプルなほど、パフォーマンスが向上します。また、ほとんどのベンチマークは、ツールから提供される追加機能をテストしていません。

階層関係はこのようになります。

- **Uvicorn**: ASGIサーバー
    - **Starlette**: (Uvicornを使用) WEBマイクロフレームワーク
        - **FastAPI**: (Starletteを使用) データバリデーションなどの、APIを構築する追加機能を備えたAPIマイクロフレームワーク

- **Uvicorn**:
    - サーバー自体に余分なコードが少ないので、最高のパフォーマンスが得られます。
    - Uvicornにアプリケーションを直接書くことはできません。つまり、あなたのコードには、Starlette (または** FastAPI **) が提供するコードを、多かれ少なかれ含める必要があります。そうすると、最終的なアプリケーションは、フレームワークを使用してアプリのコードとバグを最小限に抑えた場合と同じオーバーヘッドになります。
    - もしUvicornを比較する場合は、Daphne、Hypercorn、uWSGIなどのアプリケーションサーバーと比較してください。
- **Starlette**:
    - Uvicornに次ぐ性能を持つでしょう。実際、StarletteはUvicornを使用しています。だから、より多くのコードを実行する必要があり、Uvicornよりも「遅く」なってしまうだけなのです。
    - しかし、パスベースのルーティングなどのシンプルなWEBアプリケーションを構築する機能を提供します。
    - もしStarletteを比較する場合は、Sanic、Flask、DjangoなどのWEBフレームワーク (もしくはマイクロフレームワーク) と比較してください。
- **FastAPI**:
    - StarletteがUvicornを使っているのと同じで、**FastAPI**はStarletteを使っており、それより速くできません。
    - FastAPIはStarletteの上にさらに多くの機能を提供します。データの検証やシリアライゼーションなど、APIを構築する際に常に必要な機能です。また、それを使用することで、自動ドキュメント化を無料で取得できます (ドキュメントは実行中のアプリケーションにオーバーヘッドを追加せず、起動時に生成されます) 。
    - FastAPIを使用せず、直接Starlette (またはSanic, Flask, Responderなど) を使用した場合、データの検証とシリアライズをすべて自分で実装する必要があります。そのため、最終的なアプリケーションはFastAPIを使用して構築した場合と同じオーバーヘッドが発生します。そして、多くの場

合、このデータ検証とシリアライズは、アプリケーションのコードの中で最大の記述量になります。

- FastAPIを使用することで、開発時間、バグ、コード行数を節約でき、使用しない場合 (あなたが全ての機能を実装し直した場合) と同じかそれ以上のパフォーマンスを得られます。
- もしFastAPIを比較する場合は、Flask-apispec、NestJS、Moltenなどのデータ検証や、シリアライズの機能を提供するWEBフレームワーク (や機能のセット) と比較してください。これらはデータの自動検証や、シリアライズ、ドキュメント化が統合されたフレームワークです。