ヘッダーのパラメータ

ヘッダーのパラメータは、`Query`や`Path`、`Cookie`のパラメータを定義するのと同じように定義できます。

`Header`をインポート

まず、`Header`をインポートします:

```
{!../../../docs_src/header_params/tutorial001.py!}
```

`Header`のパラメータの宣言

次に、`Path`や`Query`、`Cookie`と同じ構造を用いてヘッダーのパラメータを宣言します。

最初の値がデフォルト値で、追加の検証パラメータや注釈パラメータをすべて渡すことができます。

```
{!../../../docs_src/header_params/tutorial001.py!}
```

!!! note "技術詳細" `Header`は`Path`や`Query`、`Cookie`の「姉妹」クラスです。また、同じ共通の`Param`クラスを継承しています。

しかし、`fastapi`から`Query`や`Path`、`Header`などをインポートする場合、それらは実際には特殊なクラスを返す関数であることを覚えておいてください。

!!! info "情報" ヘッダーを宣言するには、`Header`を使う必要があります。なぜなら、そうしないと、パラメータがクエリのパラメータとして解釈されてしまうからです。

自動変換

`Header`は`Path`や`Query`、`Cookie`が提供する機能に加え、少しだけ追加の機能を持っています。

ほとんどの標準ヘッダーは、「マイナス記号」（`-`）としても知られる「ハイフン」で区切られています。

しかし、`user-agent`のような変数はPythonでは無効です。

そのため、デフォルトでは、`Header`はパラメータの文字をアンダースコア（`_`）からハイフン（`-`）に変換して、ヘッダーを抽出して文書化します。

また、HTTPヘッダは大文字小文字を区別しないので、Pythonの標準スタイル（別名「スネークケース」）で宣言することができます。

そのため、`User_Agent`などのように最初の文字を大文字にする必要はなく、通常のPythonコードと同じように`user_agent`を使用することができます。

もしなんらかの理由でアンダースコアからハイフンへの自動変換を無効にする必要がある場合は、`Header`の`convert_underscores`に`False`を設定してください:

```
{!../../../docs_src/header_params/tutorial002.py!}
```

!!! warning "注意" `convert_underscores` を `False` に設定する前に、HTTPプロキシやサーバの中にはアンダースコアを含むヘッダーの使用を許可していないものがあることに注意してください。

ヘッダーの重複

受信したヘッダーが重複することがあります。つまり、同じヘッダーで複数の値を持つということです。

これらの場合、リストの型宣言を使用して定義することができます。

重複したヘッダーのすべての値をPythonの `list` として受け取ることができます。

例えば、複数回出現する可能性のある `X-Token` のヘッダを定義するには、以下のように書くことができます:

```
{!../../../docs_src/header_params/tutorial003.py!}
```

もし、その*path operation*で通信する場合は、次のように 2 つのHTTPヘッダーを送信します:

```
X-Token: foo
X-Token: bar
```

このレスポンスは以下のようになります:

```
{
    "X-Token values": [
        "bar",
        "foo"
    ]
}
```

まとめ

ヘッダーは `Header` で宣言し、 `Query` や `Path` 、 `Cookie` と同じパターンを使用する。

また、変数のアンダースコアを気にする必要はありません。**FastAPI** がそれらの変換をすべて取り持ってくれます。