Header Parameters

You can define Header parameters the same way you define `Query`, `Path` and `Cookie` parameters.

Import `Header`

First import `Header`:

Python 3.10+

```
from typing import Annotated

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(user_agent: Annotated[str | None, Header()] = None):
    return {"User-Agent": user_agent}
```

▸ 🤓 Other versions and variants

Python 3.9+

```
from typing import Annotated, Union

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(user_agent: Annotated[Union[str, None], Header()] = None):
    return {"User-Agent": user_agent}
```

Python 3.8+

```
from typing import Union

from fastapi import FastAPI, Header
from typing_extensions import Annotated

app = FastAPI()


@app.get("/items/")
async def read_items(user_agent: Annotated[Union[str, None], Header()] = None):
    return {"User-Agent": user_agent}
```

Python 3.10+ - non-Annotated

Tip

Prefer to use the `Annotated` version if possible.

```
from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(user_agent: str | None = Header(default=None)):
    return {"User-Agent": user_agent}
```

Python 3.8+ - non-Annotated

Tip

Prefer to use the `Annotated` version if possible.

```
from typing import Union

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(user_agent: Union[str, None] = Header(default=None)):
    return {"User-Agent": user_agent}
```

Declare `Header` parameters

Then declare the header parameters using the same structure as with `Path` , `Query` and `Cookie` .

You can define the default value as well as all the extra validation or annotation parameters:

Python 3.10+

```
from typing import Annotated

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(user_agent: Annotated[str | None, Header()] = None):
    return {"User-Agent": user_agent}
```

▶ 🤓 Other versions and variants

Python 3.9+

```
from typing import Annotated, Union

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(user_agent: Annotated[Union[str, None], Header()] = None):
    return {"User-Agent": user_agent}
```

Python 3.8+

```
from typing import Union

from fastapi import FastAPI, Header
from typing_extensions import Annotated

app = FastAPI()


@app.get("/items/")
async def read_items(user_agent: Annotated[Union[str, None], Header()] = None):
    return {"User-Agent": user_agent}
```

Python 3.10+ - non-Annotated

Tip

Prefer to use the `Annotated` version if possible.

```
from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(user_agent: str | None = Header(default=None)):
    return {"User-Agent": user_agent}
```

Python 3.8+ - non-Annotated

Tip

Prefer to use the `Annotated` version if possible.

```
from typing import Union

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(user_agent: Union[str, None] = Header(default=None)):
    return {"User-Agent": user_agent}
```

Technical Details

`Header` is a "sister" class of `Path` , `Query` and `Cookie` . It also inherits from the same common `Param` class.

But remember that when you import `Query` , `Path` , `Header` , and others from `fastapi` , those are actually functions that return special classes.

Info

To declare headers, you need to use `Header` , because otherwise the parameters would be interpreted as query parameters.

Automatic conversion

`Header` has a little extra functionality on top of what `Path` , `Query` and `Cookie` provide.

Most of the standard headers are separated by a "hyphen" character, also known as the "minus symbol" (`-`).

But a variable like `user-agent` is invalid in Python.

So, by default, `Header` will convert the parameter names characters from underscore (`_`) to hyphen (`-`) to extract and document the headers.

Also, HTTP headers are case-insensitive, so, you can declare them with standard Python style (also known as "snake_case").

So, you can use `user_agent` as you normally would in Python code, instead of needing to capitalize the first letters as `User_Agent` or something similar.

If for some reason you need to disable automatic conversion of underscores to hyphens, set the parameter `convert_underscores` of `Header` to `False` :

Python 3.10+

```python
from typing import Annotated

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(
    strange_header: Annotated[str | None, Header(convert_underscores=False)] = None,
):
    return {"strange_header": strange_header}
```

▸ 🤓 Other versions and variants

Python 3.9+

```python
from typing import Annotated, Union

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(
    strange_header: Annotated[
        Union[str, None], Header(convert_underscores=False)
    ] = None,
):
    return {"strange_header": strange_header}
```

Python 3.8+

```python
from typing import Union

from fastapi import FastAPI, Header
from typing_extensions import Annotated

app = FastAPI()


@app.get("/items/")
async def read_items(
    strange_header: Annotated[
        Union[str, None], Header(convert_underscores=False)
    ] = None,
):
    return {"strange_header": strange_header}
```

Python 3.10+ - non-Annotated

Tip

Prefer to use the `Annotated` version if possible.

```
from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(
    strange_header: str | None = Header(default=None, convert_underscores=False),
):
    return {"strange_header": strange_header}
```

Python 3.8+ - non-Annotated

Tip

Prefer to use the `Annotated` version if possible.

```
from typing import Union

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(
    strange_header: Union[str, None] = Header(default=None, convert_underscores=False),
):
    return {"strange_header": strange_header}
```

Warning

Before setting `convert_underscores` to `False`, bear in mind that some HTTP proxies and servers disallow the usage of headers with underscores.

Duplicate headers

It is possible to receive duplicate headers. That means, the same header with multiple values.

You can define those cases using a list in the type declaration.

You will receive all the values from the duplicate header as a Python `list`.

For example, to declare a header of `X-Token` that can appear more than once, you can write:

Python 3.10+

```
from typing import Annotated

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(x_token: Annotated[list[str] | None, Header()] = None):
    return {"X-Token values": x_token}
```

▸ 🤓 Other versions and variants

Python 3.9+

```
from typing import Annotated, List, Union

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(x_token: Annotated[Union[List[str], None], Header()] = None):
    return {"X-Token values": x_token}
```

Python 3.8+

```
from typing import List, Union

from fastapi import FastAPI, Header
from typing_extensions import Annotated

app = FastAPI()


@app.get("/items/")
async def read_items(x_token: Annotated[Union[List[str], None], Header()] = None):
    return {"X-Token values": x_token}
```

Python 3.10+ - non-Annotated

Tip

Prefer to use the `Annotated` version if possible.

```
from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(x_token: list[str] | None = Header(default=None)):
    return {"X-Token values": x_token}
```

Python 3.9+ - non-Annotated

Tip

Prefer to use the `Annotated` version if possible.

```
from typing import Union

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(x_token: Union[list[str], None] = Header(default=None)):
    return {"X-Token values": x_token}
```

Python 3.8+ - non-Annotated

Tip

Prefer to use the `Annotated` version if possible.

```
from typing import List, Union

from fastapi import FastAPI, Header

app = FastAPI()


@app.get("/items/")
async def read_items(x_token: Union[List[str], None] = Header(default=None)):
    return {"X-Token values": x_token}
```

If you communicate with that *path operation* sending two HTTP headers like:

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

The response would be like:

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

Recap

Declare headers with `Header`, using the same common pattern as `Query`, `Path` and `Cookie`.

And don't worry about underscores in your variables, **FastAPI** will take care of converting them.