Parámetros de query

Cuando declaras otros parámetros de la función que no hacen parte de los parámetros de path estos se interpretan automáticamente como parámetros de "query".

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

El query es el conjunto de pares de key-value que van después del `?` en la URL, separados por caracteres `&`.

Por ejemplo, en la URL:

```
http://127.0.0.1:8000/items/?skip=0&limit=10
```

...los parámetros de query son:

- `skip`: con un valor de `0`
- `limit`: con un valor de `10`

Dado que son parte de la URL son strings "naturalmente".

Pero cuando los declaras con tipos de Python (en el ejemplo arriba, como `int`) son convertidos a ese tipo y son validados con él.

Todo el proceso que aplicaba a los parámetros de path también aplica a los parámetros de query:

- Soporte del editor (obviamente)
- "Parsing" de datos
- Validación de datos
- Documentación automática

Configuraciones por defecto

Como los parámetros de query no están fijos en una parte del path pueden ser opcionales y pueden tener valores por defecto.

El ejemplo arriba tiene `skip=0` y `limit=10` como los valores por defecto.

Entonces, si vas a la URL:

```
http://127.0.0.1:8000/items/
```

Sería lo mismo que ir a:

```
http://127.0.0.1:8000/items/?skip=0&limit=10
```

Pero, si por ejemplo vas a:

```
http://127.0.0.1:8000/items/?skip=20
```

Los valores de los parámetros en tu función serán:

- `skip=20`: porque lo definiste en la URL
- `limit=10`: porque era el valor por defecto

Parámetros opcionales

Del mismo modo puedes declarar parámetros de query opcionales definiendo el valor por defecto como `None`:

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

En este caso el parámetro de la función `q` será opcional y será `None` por defecto.

!!! check "Revisa" También puedes notar que **FastAPI** es lo suficientemente inteligente para darse cuenta de que el parámetro de path `item_id` es un parámetro de path y que `q` no lo es, y por lo tanto es un parámetro de query.

!!! note "Nota" FastAPI sabrá que `q` es opcional por el `= None`.

```
El `Optional` en `Optional[str]` no es usado por FastAPI (FastAPI solo usará la parte `str`), pero el `Optional[str]` le permitirá a tu editor ayudarte a encontrar errores en tu código.
```

Conversión de tipos de parámetros de query

También puedes declarar tipos `bool` y serán convertidos:

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

En este caso, si vas a:

```
http://127.0.0.1:8000/items/foo?short=1
```

o

```
http://127.0.0.1:8000/items/foo?short=True
```

o

```
http://127.0.0.1:8000/items/foo?short=true
```

o

```
http://127.0.0.1:8000/items/foo?short=on
```

o

```
http://127.0.0.1:8000/items/foo?short=yes
```

o cualquier otra variación (mayúsculas, primera letra en mayúscula, etc.) tu función verá el parámetro `short` con un valor `bool` de `True`. Si no, lo verá como `False`.

Múltiples parámetros de path y query

Puedes declarar múltiples parámetros de path y parámetros de query al mismo tiempo. **FastAPI** sabe cuál es cuál.

No los tienes que declarar en un orden específico.

Serán detectados por nombre:

```
{!../../../docs_src/query_params/tutorial004.py!}
```

Parámetros de query requeridos

Cuando declaras un valor por defecto para los parámetros que no son de path (por ahora solo hemos visto parámetros de query), entonces no es requerido.

Si no quieres añadir un valor específico sino solo hacerlo opcional, pon el valor por defecto como `None`.

Pero cuando quieres hacer que un parámetro de query sea requerido, puedes simplemente no declararle un valor por defecto:

```
{!../../../docs_src/query_params/tutorial005.py!}
```

Aquí el parámetro de query `needy` es un parámetro de query requerido, del tipo `str`.

Si abres tu navegador en una URL como:

```
http://127.0.0.1:8000/items/foo-item
```

...sin añadir el parámetro `needy` requerido, verás un error como:

```
{
    "detail": [
        {
            "loc": [
                "query",
                "needy"
            ],
            "msg": "field required",
            "type": "value_error.missing"
        }
    ]
}
```

Dado que `needy` es un parámetro requerido necesitarías declararlo en la URL:

```
http://127.0.0.1:8000/items/foo-item?needy=sooooneedy
```

...esto funcionaría:

```
{
    "item_id": "foo-item",
    "needy": "sooooneedy"
}
```

Por supuesto que también puedes definir algunos parámetros como requeridos, con un valor por defecto y otros completamente opcionales:

```
{!../../../docs_src/query_params/tutorial006.py!}
```

En este caso hay 3 parámetros de query:

- `needy`, un `str` requerido.
- `skip`, un `int` con un valor por defecto de `0`.
- `limit`, un `int` opcional.

!!! tip "Consejo" También podrías usar los `Enum`s de la misma manera que con los Parámetros de path{.internal-link target=_blank}.