typing. typing? typing!

Nikita Grishko

Flo Health Inc.

```
import typing
```

typing. typing? typing!

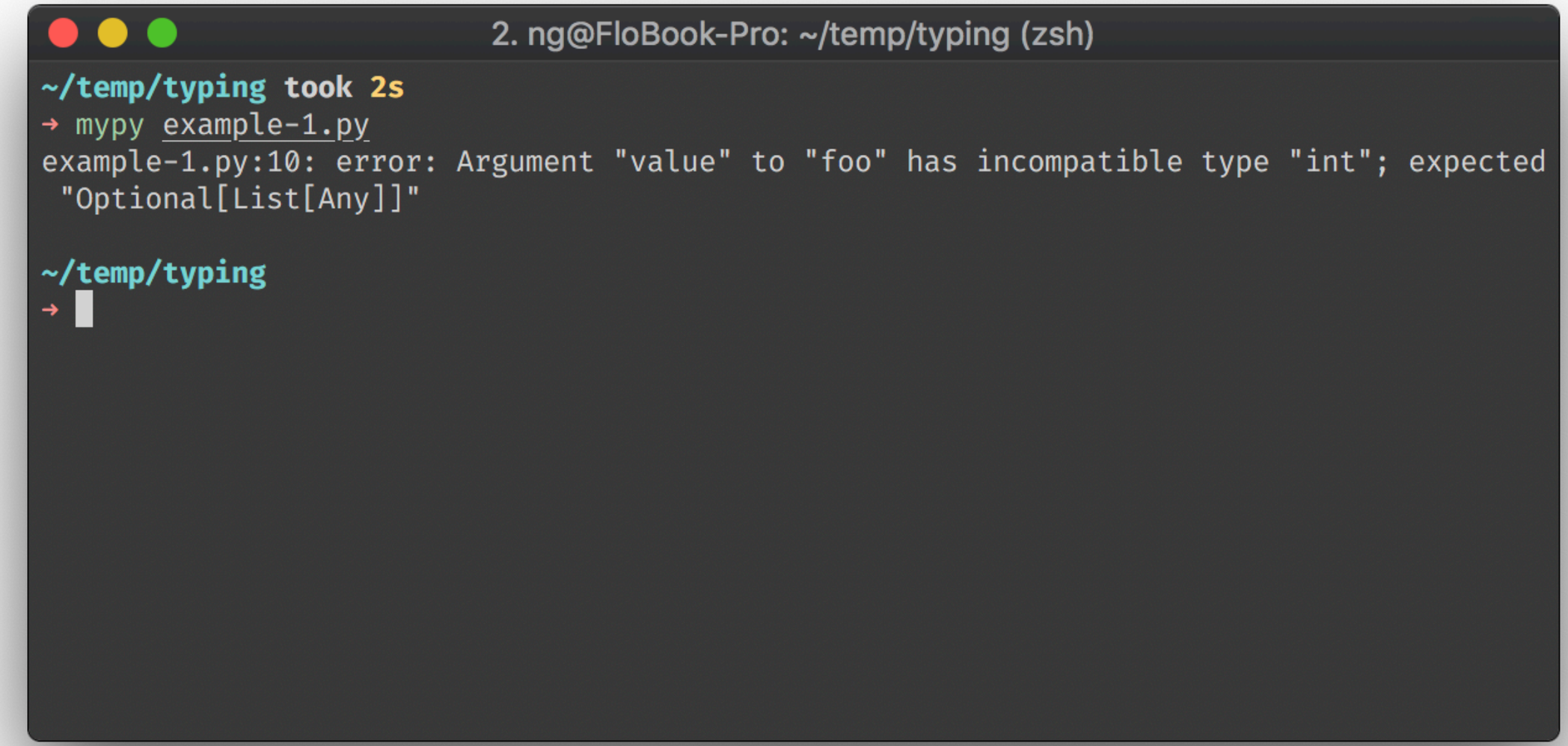
typing.

BayPiggies Talk at LinkedIn: Introducing Type Annotations for Python

https://www.youtube.com/watch?v=ZP_QV4ccFHQ

```python
def foo(*, value=None):
    value = value or []
    return "".join(value)


def bar():
    return foo(value=42)
```

```python
from typing import Optional


def foo(*, value: Optional[list] = None) -> str:
    value = value or []
    return "".join(value)


def bar() -> str:
    return foo(value=42)
```


2. ng@FloBook-Pro: ~/temp/typing (zsh)
~/temp/typing took 2s
→ mypy example-1.py
example-1.py:10: error: Argument "value" to "foo" has incompatible type "int"; expected
"Optional[List[Any]]"
~/temp/typing
→


I'M OUT

Tests!?

typing?

typing?

- Python 3.6
- PEP 484 — Type Hints
- PEP 526 — Syntax for Variable Annotations

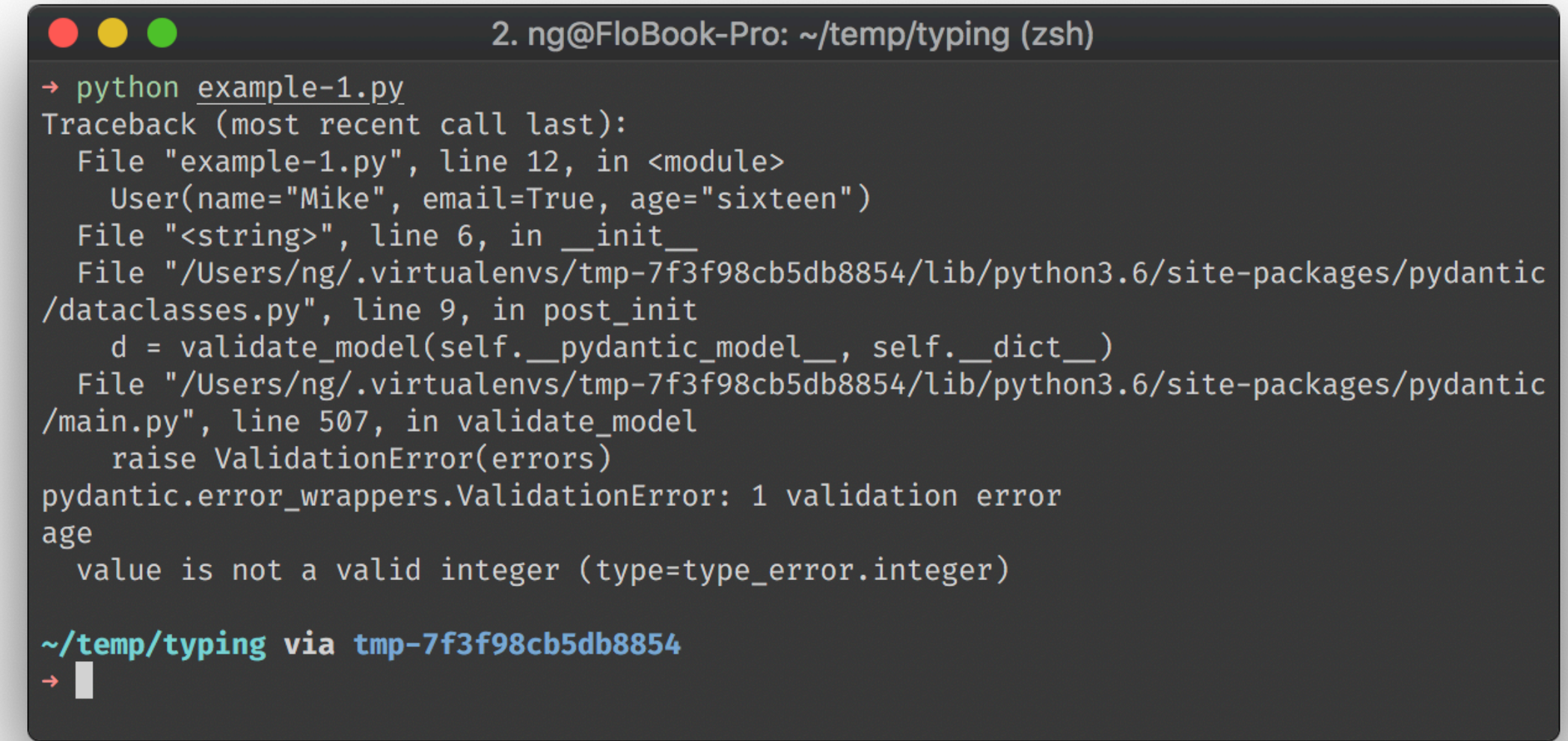
typing?

- Python 3.7
- PEP 557 – Data Classes
- pip install "dataclasses;python_version<'3.7'"

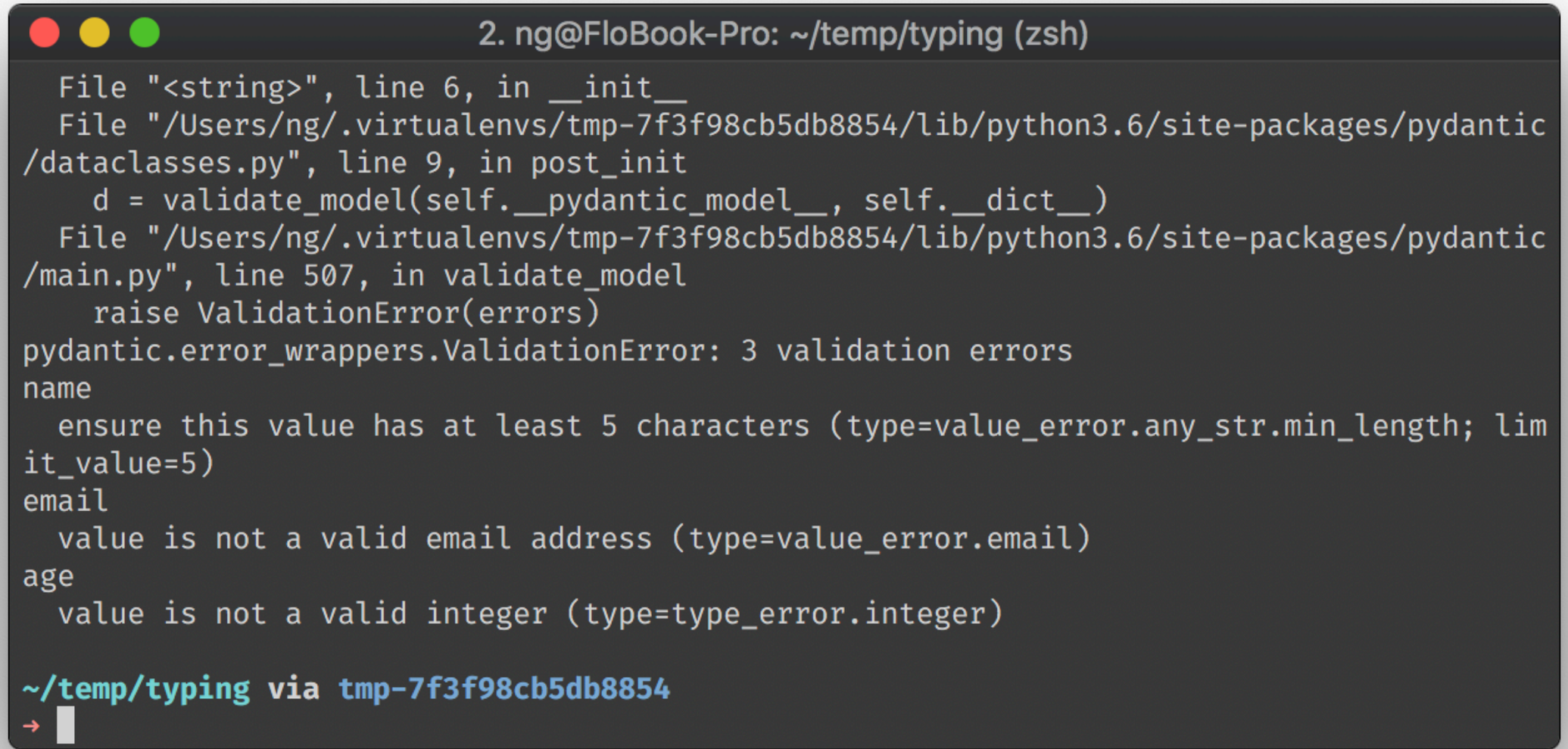
```python
from dataclasses import dataclass


@dataclass
class User:
    name: str
    email: str
    age: int
    is_active: bool = True


User(name="Walter", email="walter.mitty@mail.me", age=18)
User(name="John", email="Where is my snow?", age=21)
User(name="Mike", email=True, age="sixteen")
```

```
pip install pydantic
```

```python
from pydantic.dataclasses import dataclass


@dataclass
class User:
    name: str
    email: str
    age: int
    is_active: bool = True


User(name="Mike", email=True, age="sixteen")
```


2. ng@FloBook-Pro: ~/temp/typing (zsh)
→ python example-1.py
Traceback (most recent call last):
File "example-1.py", line 12, in <module>
User(name="Mike", email=True, age="sixteen")
File "<string>", line 6, in __init__
File "/Users/ng/.virtualenvs/tmp-7f3f98cb5db8854/lib/python3.6/site-packages/pydantic/dataclasses.py", line 9, in post_init
d = validate_model(self.__pydantic_model__, self.__dict__)
File "/Users/ng/.virtualenvs/tmp-7f3f98cb5db8854/lib/python3.6/site-packages/pydantic/main.py", line 507, in validate_model
raise ValidationError(errors)
pydantic.error_wrappers.ValidationError: 1 validation error
age
value is not a valid integer (type=type_error.integer)
~/temp/typing via tmp-7f3f98cb5db8854
→

```python
from pydantic import EmailStr, constr
from pydantic.dataclasses import dataclass


@dataclass
class User:
    name: constr(min_length=5)
    email: EmailStr
    age: int
    is_active: bool = True


User(name="Mike", email=True, age="sixteen")
```


2. ng@FloBook-Pro: ~/temp/typing (zsh)
File "<string>", line 6, in __init__
File "/Users/ng/.virtualenvs/tmp-7f3f98cb5db8854/lib/python3.6/site-packages/pydantic/dataclasses.py", line 9, in post_init
d = validate_model(self.__pydantic_model__, self.__dict__)
File "/Users/ng/.virtualenvs/tmp-7f3f98cb5db8854/lib/python3.6/site-packages/pydantic/main.py", line 507, in validate_model
raise ValidationError(errors)
pydantic.error_wrappers.ValidationError: 3 validation errors
name
ensure this value has at least 5 characters (type=value_error.any_str.min_length; limit_value=5)
email
value is not a valid email address (type=value_error.email)
age
value is not a valid integer (type=type_error.integer)
~/temp/typing via tmp-7f3f98cb5db8854
→

typing!

typing!

- How to write yet another API
- How to pack/unpack JSON
- How to write API documentation
- How to write API client

```
class UglyView(web.View):
    async def post(self):
        ...
```

```python
class UglyView(web.View):
    async def post(self):
        try:
            body = await self.request.json()
        except json.JSONDecodeError:
            ...
```

```python
class UglyView(web.View):
    async def post(self):
        try:
            body = await self.request.json()
        except json.JSONDecodeError:
            body = {}

        try:
            user = User.parse_obj(body)
        except ValidationError as e:
            return web.json_response(
                e.errors(), status=HTTPStatus.BAD_REQUEST
            )
```

```python
class UglyView(web.View):
    async def post(self):
        try:
            body = await self.request.json()
        except json.JSONDecodeError:
            body = {}

        try:
            user = User.parse_obj(body)
        except ValidationError as e:
            return web.json_response(
                e.errors(), status=HTTPStatus.BAD_REQUEST
            )

        try:
            trace_id = UUID(self.request.query.get("trace_id"))
        except (TypeError, ValueError):
            return web.json_response(
                "invalid `trace_id`", status=HTTPStatus.BAD_REQUEST
            )
```

```python
class HandsomeView(TypeView):
    async def post(self):
        ...
```

```python
class HandsomeView(TypeView):
    async def post(self, body: Body[User]):
        ...
```

```python
class HandsomeView(TypeView):
    async def post(self, body: Body[User], trace_id: QueryParam[UUID]):
        ...
```

https://github.com/Gr1N/aiohttp-typed-views

```python
from typing import Generic, TypeVar


TB = TypeVar("TB")
TQ = TypeVar("TQ")


class Body(Generic[TB]):
    def __init__(self, value: TB) -> None:
        ...


class QueryParam(Generic[TQ]):
    def __init__(self, value: TQ) -> None:
        ...
```

```
import inspect
```

```
body, query = {}, {}

for param in inspect.signature(method).parameters.values():
    param_name, param_type = param.name, param.annotation
    if param_type is inspect.Signature.empty:
        continue

    if getattr(param_type, "__origin__", param_type) is Body:
        body[param_name] = getattr(param_type, "__args__", (Any,))[0]
    elif getattr(param_type, "__origin__", param_type) is QueryParam:
        query[param_name] = getattr(param_type, "__args__", (Any,))[0]
```

```python
from pydantic import BaseModel, EmailStr, Schema


class User(BaseModel):
    name: str = Schema(
        ..., description="User name", example="Walter"
    )
    email: EmailStr = Schema(
        ..., description="User email", example="walter.mitty@mail.me"
    )


User.schema()
```

```python
class Client:
    async def create_user(
        self, body: Body[User], track_id: QueryParam[UUID]
    ):
        return await self.session.post(
            "/users",
            body=body,
            params={
                "track_id": track_id,
            },
        )
```

typing!

- https://github.com/encode/apistar (<0.6.0)
- https://github.com/timothycrosley/hug

typing!

- https://github.com/samuelcolvin/pydantic
- https://github.com/Gr1N/aiohttp-typed-views

Questions?

https://github.com/Gr1N