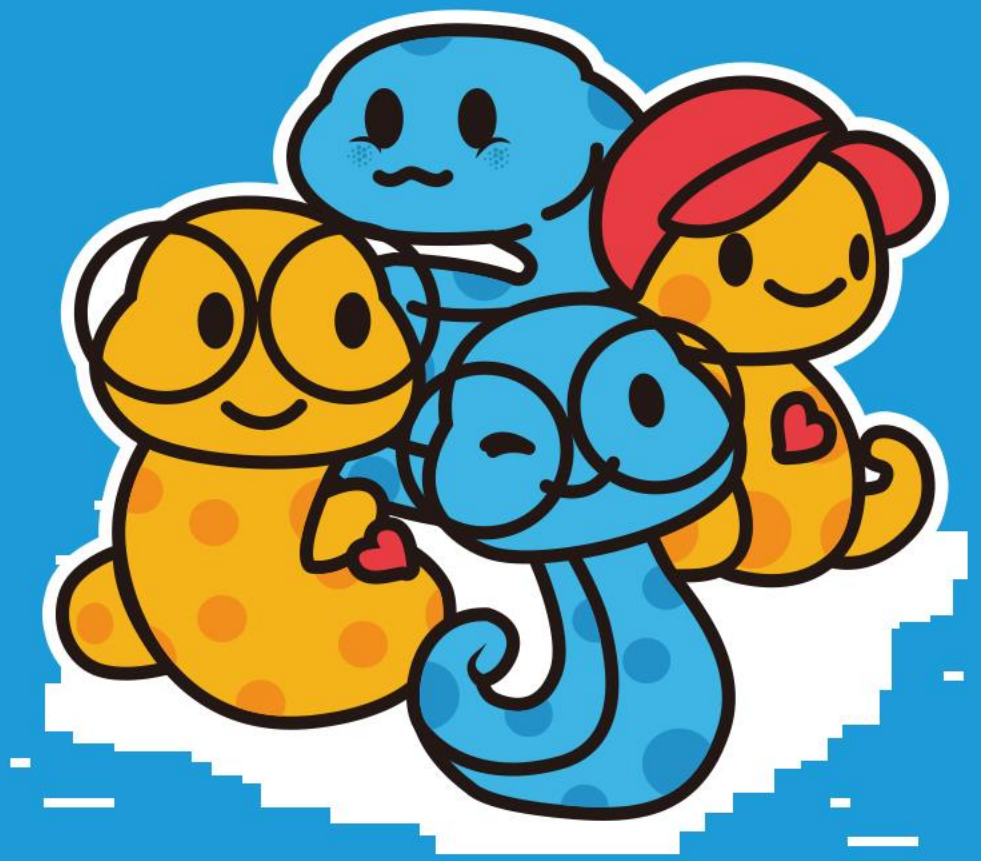
PyCon China 2024

For Good . For fun.

2024/11/23 中国 上海



PYCHINA
Python 中国社区

PyCon China 2024

>> 2024/11/23 上海

>> For good . For fun.

残缺的 Type hint 设计与错漏百出的社区代码

幽默 TypedDict

```python
class T(TypedDict):
    a: int
    b: str


t = T(a=1, b="2")
```

```
class T(
    *args: Any,
    **kwargs: Any
): ...

class T(
    *,
    a: int,
    b: str
): ...
```

```python
T = TypedDict("T", {"a-0": int, "b": str})

t = T()
```

“__init__”的重载与提供的参数不匹配
参数类型:() Pylance(reportCallIssue)

AI Fix In Chat Ctrl+Shift+E

(type) T = T

查看问题 (Alt+F8)　快速修复... (Ctrl+.)

```python
WSGIDefined = TypedDict(
    "WSGIDefined",
    {
        "wsgi.version": Tuple[int, int],  # e.g. (1, 0)
        "wsgi.url_scheme": str,  # e.g. "http" or "https"
        "wsgi.input": InputStream,
        "wsgi.errors": ErrorStream,
        # This value should evaluate true if the application object may be simultaneously
        # invoked by another thread in the same process, and should evaluate false otherwise.
        "wsgi.multithread": bool,
        # This value should evaluate true if an equivalent application object may be
        # simultaneously invoked by another process, and should evaluate false otherwise.
        "wsgi.multiprocess": bool,
        # This value should evaluate true if the server or gateway expects (but does
        # not guarantee!) that the application will only be invoked this one time during
        # the life of its containing process. Normally, this will only be true for a
        # gateway based on CGI (or something similar).
        "wsgi.run_once": bool,
    },
)


Aber, 11个月前 | 1 author (Aber)
class Environ(CGIRequiredDefined, CGIOptionalDefined, WSGIDefined):
    """
    WSGI Environ
    """
```

描述不了的 partial

```python
def add(x: int, y: int) -> int:
    return x + y


add_1 = partial(add, 1)
```

```
(variable) add_1: partial[int]
```

```python
add_1(2)        print(add_1(2))
```

```python
from cool import F

def add(x: int, y: int) -> int:
    return x + y


add_1 = F(add, 1)
```

```
(variable) x: Unknown
```

```python
x = add_1(2)
```

FastAPI

```python
def jsonable_encoder(
    obj: Annotated[
        Any,
        Doc(
            """
            The input object to convert to JSON.
            """
        ),
    ],
    include: Annotated[
        Optional[IncEx],
        Doc(
            """
            Pydantic's `include` parameter, passed to Pydantic models to set the
            fields to include.
            """
        ),
    ] = None,
    exclude: Annotated[
        Optional[IncEx],
        Doc(
            """
            Pydantic's `exclude` parameter, passed to Pydantic models to set the
            fields to exclude.
            """
        ),
    ] = None,
```

Definition Search

fastapi/types.py

```
10 IncEx = Union[Set[int], Set[str], Dict[int,
```

```python
    if include is not None and not isinstance(include, (set, dict)):
        include = set(include)
    if exclude is not None and not isinstance(exclude, (set, dict)):
        exclude = set(exclude)
```

Beanie

```python
from typing import Optional

import pymo
from beanie
from pydant


class Categ
    name: s
    descrip


class Produ
    name: s
    descrip
    price: Indexed(float, pymongo.DESCENDING)
    category: Category

    class Settings:
        name = "products"
        indexes = [
            [
                ("name", pymongo.TEXT),
                ("description", pymongo.TEXT),
            ],
        ]


async def main() -> None:
    bar = await Product.find(Product.price > .5).inc({Product.price: 1})
    ⌘L to chat, ⌘K to generate
```

类型表达式中不允许使用调用表达式 Pylance(reportInvalidTypeForm)

AI Fix In Chat ⇧⌘E

```
(function) def Indexed(
    typ: FixtureFunction[SimpleFixtureFunction@FixtureFunction,
FactoryFixtureFunction@FixtureFunction] | None = None,
    index_type: int = ASCENDING,
    **kwargs: FixtureFunction[SimpleFixtureFunction@FixtureFunction,
FactoryFixtureFunction@FixtureFunction]
) -> (IndexedAnnotation | type[NewType])
```

If `typ` is defined, returns a subclass of `typ` with an extra attribute `_indexe` tuple:

```python
from typing import Optional

import pymongo
from beanie import Document, Indexed
from pydantic import BaseModel


class Category(BaseModel):
    name: str
    description: str


class Product(Document):  # This is the model
    name: str
    description: Optional[str] = None
    price: Index
    category: Ca

    class Settin
        name = "
        indexes
            [

            ],
        ]


async def main()
    bar = await Product.find(Product.price > .5).inc({Product.price: 1})
    ⌘L to chat, ⌘K to generate
```

“FindMany[Product]”不可等待
“FindMany[Product]”与协议“Awaitable[_T_co@Awaitable]”不兼容
“__await__”不存在 Pylance(reportGeneralTypeIssues)

AI Fix In Chat ⇧⌘E

```
(method) def inc(
    expression: Dict[ExpressionField | float | int | str, Any],
    session: AsyncIOMotorClientSession | None = None,
    bulk_writer: BulkWriter | None = None,
    **kwargs: FixtureFunction[SimpleFixtureFunction@FixtureFunction,
FactoryFixtureFunction@FixtureFunction]
) -> FindMany[Product]
```

Increment

Redis

```python
from redis import Redis
from redis.asyncio import Redis as AsyncRedis


async def async_main():
    r = AsyncRedis()
    v = await r.get("key")


def sync_main():
    r = Redis()
    v = r.get("key")
```

```python
from redis import Redis
from redis.asyncio import Redis as AsyncRedis


async def async_main() -> None:
    r = AsyncRedis()
    v = await r.get("key")


def sync_main():
    r = Redis()
    v = r.get("k
    ⌘L to chat, ⌘K to
```

未存取“v” Pylance

AI Fix In Chat ⇧⌘E

(variable) v: Any

没有可用的快速修复

```python
from redis import Redis
from redis.asyncio import Redis as AsyncRedis


async def async_main() -> None:
    r = AsyncRedis()
    v = await r.get("key")


def sync_main():
    r = Redis()
    v = r.get("key")
    ⌘L to chat, ⌘K to generate
```

未存取“v” Pylance

AI Fix In Chat ⇧⌘E

(variable) v: ResponseT

没有可用的快速修复

```python
KeyT = _StringLikeT  # Main redis key space
PatternT = _StringLikeT  # Patterns matched against keys, fiel
FieldT = EncodableT  # Fields within hash tables, streams and
KeysT = Union[KeyT, Iterable[KeyT]]
ResponseT = Union[Awaitable[Any], Any]
ChannelT = _StringLikeT
GroupT = _StringLikeT  # Consumer group
ConsumerT = _StringLikeT  # Consumer name
StreamIdT = Union[int, _StringLikeT]
ScriptTextT = _StringLikeT
TimeoutSecT = Union[int, float, _StringLikeT]
```

终