Dynamic Typing pada Python

Python dikenal sebagai salah satu bahasa yang menerapkan dynamic typing, yakni bahasa pemrograman yang hanya mengetahui tipe variabel saat program berjalan dan dilakukan assignment.

Tentu saja, pada Python juga umumnya tidak ada deklarasi variabel, hanya berupa assignment statement. Cara ini adalah salah satu bentuk simplifikasi alokasi memori dalam bahasa Python. Anda dapat selalu memeriksa tipe variabel yang digunakan dengan fungsi **type()** dan memastikan tipe variabel yang tepat dengan metode **isinstance().** Anda akan mempelajari lebih jauh mengenai fungsi pada modul Fungsi dan mengenai class pada modul Pemrograman Berorientasi Objek.

Contoh:

```
# Ketika kita memberikan nilai 6 pada variabel x
x = 6
print(type(x))
# Kemudian Berikan string “hello” pada variabel x di baris selanjutnya
x = 'hello'
print(type(x))
```

Output:

```
<class 'int'>
<class 'str'>
```

Setiap bahasa pemrograman tentunya memiliki beberapa tipe atau kategori objek dan variabel yang dapat digunakan serta bagaimana kategori tersebut diperlakukan (behavior, fungsi, dsb). Contohnya, sebuah kategori bertipe “numerik” dengan “86” adalah sebuah objek dari tipe numerik tersebut. Pada bahasa Python, interpreter hanya mengartikan setiap baris kode saat dijalankan, dan sepanjang daur (cycle) program, dimungkinkan adanya perubahan dalam tipe/kategori variabel. Bagian ini hanya memberikan gambaran mengenai dinamisme tipe, Anda akan belajar lebih jauh tentang kondisional pada modul Percabangan. Contoh dinamisme Python dapat dilihat pada contoh berikut:

```
if False:
    9 + "satu"  # Baris ini tidak dioperasikan, sehingga tidak muncul notifikasi TypeError
else:
    9 + 1
```

```
10
```

Padahal jika kita jalankan 9 + "satu":

```
1. 9 + "satu"
```

Output:

```
TypeError: unsupported operand type(s) for +: 'int' and 'str'
```

Pada contoh pertama, cabang 9+ “satu” tidak pernah dioperasikan. Sehingga kode tidak akan diperiksa. Namun pada contoh kedua, pernyataan tersebut menimbulkan TypeError karena Anda tidak dapat menggabungkan Integer dan String. Bagian ini hanya memberikan gambaran tentang operasi antar tipe, Anda akan mempelajari lebih jauh tentang kesalahan (Error) di modul Penanganan Kesalahan.

Berikut, adalah contoh untuk assignment:

```
1. contoh = "Halo, Buchori"
2. print(type(contoh))
```

Output:

```
<class 'str'>
```

```
1. contoh = 19.7
2. print(type(contoh))
```

Output:

```
<class 'float'>
```

menerima perubahan ini pada saat program dijalankan.

Python menggunakan dynamic typing sebagai opsi utama, namun sejak Python 3.6 sudah mendukung juga static typing. Pada PEP 484 diperkenalkan type hints, yang memungkinkan Anda untuk melakukan static type checking pada Python.

Static typing seperti didefinisikan di PEP 484, 526, 544, 560, dan 563 dibangun bertahap di atas runtime Python yang ada dan dibatasi oleh perilaku sintaksis dan runtime yang ada. Tidak seperti pada bahasa lain, type hints tidak menjadikan sebuah variabel menjadi static typed, melainkan memberikan saran tipe. Terkait static type hints, dapat dibaca pada PEP 484.

Duck Typing

Python juga sering diafiliasikan dengan metode **duck typing**, yang merefleksikan pada frase:

> *"if it walks like a duck and it quacks like a duck, then it must be a duck"*

(Jika sesuatu berjalan seperti bebek dan bersuara seperti bebek, maka kemungkinan besar ia adalah bebek).

Duck typing adalah sebuah konsep, tipe atau kelas dari sebuah objek tidak lebih penting daripada metode yang menjadi perilakunya. Duck typing ini tidak terkait langsung dengan dynamic typing atau static typing, ini hanya memberikan keleluasaan pada developer untuk tidak perlu mencemaskan tentang tipe atau kelas dari sebuah objek, yang lebih penting adalah kemampuan melakukan operasinya. Sehingga untuk memeriksa dan mengetahui tipe sebuah objek, Anda cukup memastikan metode/fungsi/behavior dari objek tersebut. Misalnya fungsi len() untuk mengetahui panjang string, yang tidak berlaku pada variabel numerik (misalnya integer).

```
1. x=546
2. print(len(x))
```

Output:

```
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: object of type 'int' has no len()
```

Sebelumnya

Dicoding Space
Jl. Batik Kumeli No.50, Sukaluyu,
Kec. Cibeunying Kaler, Kota Bandung
Jawa Barat 40123

Decode Ideas

Discover Potential

Tentang Kami

Blog

Reward

Showcase

Hubungi Kami

FAQ

Penghargaan

image
click bila
belum muncul

Terms • Privacy

DIBANTU