Pierian-Data / Complete-Python-3-Bootcamp Public

Code Issues 92 Pull requests 74 Actions Projects Security Ir

43b8ab8f1c

Complete-Python-3-Bootcamp / 00-Python Object and Data Structure Basics / 01-Variable Assignment.ipynb

TiVentures added Variable Assignment

1 contributor

437 lines (437 sloc) | 8.07 KB

Variable Assignment

Rules for variable names

- names can not start with a number
- names can not contain spaces, use _ intead
- names can not contain any of these symbols:

  `:'",<>/?|\!@#%^&*~-+`

- it's considered best practice (PEP8) that names are lowercase with underscores
- avoid using Python built-in keywords like `list` and `str`
- avoid using the single characters `l` (lowercase letter el), `O` (uppercase letter oh) and `I` (uppercase letter eye) as they can be confused with `1` and `0`

Dynamic Typing

Python uses *dynamic typing*, meaning you can reassign variables to different data types. This makes Python very flexible in assigning data types; it differs from other languages that are *statically typed*.

```
In [1]: my_dogs = 2
```

```
In [2]: my_dogs
```

```
Out[2]: 2
```

```
In [3]: my_dogs = ['Sammy', 'Frankie']
```

```
In [4]: my_dogs
```

```
Out[4]: ['Sammy', 'Frankie']
```

Pros and Cons of Dynamic Typing

Pros of Dynamic Typing

- very easy to work with
- faster development time

Cons of Dynamic Typing

- may result in unexpected bugs!
- you need to be aware of `type()`

Assigning Variables

Variable assignment follows `name = object`, where a single equals sign `=` is an *assignment operator*

```
In [5]: a = 5
```

```
In [6]: a
```

```
Out[6]: 5
```

Here we assigned the integer object `5` to the variable name `a`.
Let's assign `a` to something else:

```
In [7]: a = 10
```

```
In [8]: a
```

```
Out[8]: 10
```

You can now use `a` in place of the number `10`:

```
In [9]: a + a
```

```
Out[9]: 20
```

Reassigning Variables

Python lets you reassign variables with a reference to the same object.

```
In [10]: a = a + 10
```

```
In [11]: a
```

```
Out[11]: 20
```

There's actually a shortcut for this. Python lets you add, subtract, multiply and divide numbers with reassignment using `+=`, `-=`, `*=`, and `/=`.

```
In [12]: a += 10
```

```
In [13]: a
```

```
Out[13]: 30
```

```
In [14]: a *= 2
```

```
In [15]: a
```

```
Out[15]: 60
```

Determining variable type with `type()`

You can check what type of object is assigned to a variable using Python's built-in `type()` function. Common data types include:

- **int** (for integer)
- **float**
- **str** (for string)
- **list**
- **tuple**
- **dict** (for dictionary)
- **set**
- **bool** (for Boolean True/False)

```
In [16]: type(a)
```

```
Out[16]: int
```

```
In [17]: a = (1,2)
```

```
In [18]: type(a)
```