Python → Object-oriented programming → <u>Class</u>

Theory: Class

© 15 minutes 7 / 7 problems solved

7358 users solved this topic. Latest completion was about 2 hours ago.

Start practicing

As you already know, object-oriented programming (OOP) is a programming paradigm that is based on the concept of objects. Objects interact with each other and that is how the program functions. Objects have states and behaviors. Many objects can have similar characteristics and if you need to work with them in your program it may be easier to create a class for similar objects. Classes represent the common structure of similar objects: their data and their behavior.

§1. Declaring classes

Classes are declared with a keyword class and the name of the class:

```
# class syntax
class MyClass:
    var = ... # some variable

def do_smt(self):
    # some method
```

Generally, class name starts with a capital letter and it is usually a noun or a noun phrase. The names of the classes follow the **CapWords** convention: meaning that if it's a phrase, all words in the phrase are capitalized and written without underscores between them.

```
# good class name
class MyClass:
    ...

# not so good class name:
class My_class:
    ...

# not so good class name:
```

Classes allow you to define the data and the behaviors of similar objects. The behavior is described by methods. A method is similar to a function in that it is a block of code that has a name and performs a certain action. Methods, however, are not independent since they are defined within a class. Data within classes are stored in the attributes (variables) and there are two kinds of them: class attributes and instance attributes. The difference between those two will be explained below.

§2. Class attribute

A class attribute is an attribute shared by all instances of the class. Let's consider the class **Book** as an example:

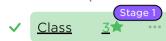
```
1  # Book class
2  class Book:
3   material = "paper"
4   cover = "paperback"
5  all_books = []
```

This class has a string variable material with the value "paper", a string variable cover with the value "paperback" and an empty list as an attribute all_books. All those variables are class attributes and they can be accessed using the dot notation with the name of the class:

```
Book.material # "paper"
Book.cover # "paperback"
Book.all_books # []
```

Class attributes are defined within the class but outside of any methods. Their value is the same for all instances of that class so you could consider them as the sort of "default" values for all objects.

Current topic:



Topic depends on:

```
Introduction to OOP

Stage 1

Stage 1
```

Topic is required for:

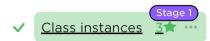


Table of contents:

↑ Class

§1. Declaring classes

§2. Class attribute

§3. Class instance

§4. Summary

Feedback & Comments

https://hyperskill.org/learn/step/6661

As for the instance variables, they store the data unique to each object of the class. They are defined within the class methods, notably, within the __init__ method. In this topic, we'll deal with the class attributes, but don't worry you'll have plenty of time to learn more about instance attributes.

§3. Class instance

Now, let's create an instance of a **Book** class. For that we would need to execute this code:

```
1  # Book instance
2  my_book = Book()
```

Here we not only created an instance of a **Book** class but also assigned it to the variable my_book. The syntax of instantiating a class object resembles the function notation: after the class name, we write parentheses.

Our my_book object has access to the contents of the class **Book**: its attributes and methods.

```
print(my_book.material) # "paper"
print(my_book.cover) # "paperback"
print(my_book.all_books) # []
```

§4. Summary

Well, those were the very basics of classes in Python. Classes represent the common structure of similar objects, their attributes, and methods. There are class attributes and instance attributes. Class attributes are common to all instances of the class.

All in all, classes are an extremely useful tool that can help you optimize your code and organize the program in a logical and readable way. We hope you'll make use of them!

Report a type

567 users liked this theory. 10 didn't like it. What about you?











Start practicing

Comments (10) Hints (1) Useful links (0) Show discussion

https://hyperskill.org/learn/step/6661