



[🏠](#) / [Design Patterns](#) / [Structural patterns](#) / [Private Class Data](#)

# Private Class Data in Python



Back to **Private Class Data** description

```
"""
Control write access to class attributes.
Separate data from methods that use it.
Encapsulate class data initialization.
"""

class DataClass:
    """
    Hide all the attributes.
    """

    def __init__(self):
        self.value = None

    def __get__(self, instance, owner):
        return self.value

    def __set__(self, instance, value):
        if self.value is None:
            self.value = value

class MainClass:
    """
    Initialize data class through the data class's constructor.
    """

    attribute = DataClass()

    def __init__(self, value):
        self.attribute = value

def main():
    m = MainClass(True)
    m.attribute = False


if __name__ == "__main__":
    main()
```

## Read next

This article is taken from our book **Design Patterns Explained Simply**.

All of the design patterns are compiled there. The book is written in clear, simple language that makes it easy to read and understand (just like this article).

We distribute it in PDF & EPUB formats so you can get it onto your iPad, Kindle, or other portable device immediately after your purchase.

 [Learn more](#)



Code examples

C#	Private Class Data in C#: Before and after

Design Patterns  
AntiPatterns  
Refactoring  
UML

My account  
Forum  
Contact us  
About us

© 2007-2018 SourceMaking.com  
All rights reserved.

[Terms / Privacy policy](#)