/ Design Patterns / Structural patterns / Private Class Data

Private Class Data in Python

Back to **Private Class Data** description

```python
"""
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**.

Code examples

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