In Python, **name mangling** is a mechanism used to protect "private" attributes of a class from being accidentally overwritten by subclasses. It is a way of "hiding" or renaming attributes so they aren't easily accessible from outside the class instance.

What does "Mangling" mean?

In a general computing context, **mangling** (or name decoration) refers to the process where a compiler or interpreter changes the name of a function or variable to include extra information about its scope, type, or location.

In Python specifically, it is a way to ensure that a class member name is unique across the inheritance hierarchy to prevent **name clashes**.

How Name Mangling Works in Python

Python mangles any identifier that is prefixed with at least **two underscores** and has at most **one trailing underscore** (e.g., `__my_variable`).

The Transformation Rule

The interpreter textually replaces the attribute name by adding the class name as a prefix. The formula looks like this: $$_ \text{ClassName} __ \text{AttributeName}$$

Code Example

```
class Robot:
    def __init__(self):
        self.__energy = 100  # Mangled attribute
        self._status = "Active" # Not mangled (protected convention)

r = Robot()

# This will raise an AttributeError:
# print(r.__energy)

# This works because of name mangling:
print(r._Robot__energy)
```

Why Use It?

The primary goal of name mangling is **Safety**, not Security. It is designed to solve two main issues:

1. **Preventing Subclass Overrides:** If a subclass defines a variable with the same name as a variable in the parent class, it won't accidentally overwrite the parent's "private" data.
2. **Internal Class Integrity:** It ensures that the class's internal logic remains intact even if a user of the class adds their own attributes to an instance.

Important Distinctions

- `_single_leading_underscore`: A "weak" internal indicator. It tells other programmers "please don't touch this," but Python does nothing to the name.
- `__double_leading_underscore`: Triggers **Name Mangling**. The interpreter actively changes the name to make it harder to access.
- `__double_leading_and_trailing_underscore__`: These are "Magic Methods" or **Dunder methods** (like `__init__` or `__str__`). These are **never** mangled; they are reserved for special Python functionality.

Note: In Python, everything is ultimately accessible. Name mangling is a "speed bump," not a locked door. It is generally preferred to use a single underscore for internal variables unless you are writing a library intended for heavy inheritance where name collisions are a real risk.