Lecture

Getters & Setters

Getters & Setters

- Members of a class (**Methods**).
- Their purpose is to "**get**" and "**set**" the value of an instance attribute, respectively.
- They protect data by providing an indirect way to access and modify it.
- They act like "functions" for the instance that calls them.

Getters

```
class Dog:

    def __init__(self, name):
        self._name = name
```

Getters

```python
class Dog:

    def __init__(self, name):
        self._name = name
```

Should not be accessed directly

Getters

Getters

Access the attribute indirectly

Getters

```
class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name
```

Getters

```python
class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name
```

Getters

```
class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name
```

Keyword

Getters

```
class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name
```

Name

Getters

```
class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name
```

Parameters

Getters

```
class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name
```

Getters

```python
class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name
```

Return the name of the instance that called the method

Getters

get_ + <attribute>

Getters

get_ + <attribute>

get_name

Getters

get_ + <attribute>

get_name

get_age

Getters

get_ + <attribute>

get_name

get_address

get_age

Getters

get_ + <attribute>

get_name

get_address

get_age

get_id

Getters

get_ + <attribute>

get_name

get_address

get_color

get_age

get_id

Getters

```
>>> class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name

>>> dog1 = Dog("Nora")
>>> dog1.get_name()
'Nora'
```

Getters

```
>>> class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name

>>> dog1 = Dog("Nora")
>>> dog1.get_name()
'Nora'
```

Getters

```
>>> dog1 = Dog("Nora")
>>> dog1.get_name()
```

Getters

```
dog1.get_name()
```

Getters

Instance

```
dog1.get_name()
```

Getters

```
dog1.get_name()
```

Getters

Getter

```
dog1.get_name()
```

Getters

```
dog1.get_name()
```

Getters

```
dog1.get_name()
```

"self" is skipped

Setters

Lecture

Setters

Setters

Modify an attribute indirectly

Setters

Validation

Modify an attribute indirectly

Setters

```python
class Dog:

    def __init__(self, name):
        self._name = name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Setters

```python
class Dog:

    def __init__(self, name):
        self._name = name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Setters

```
class Dog:

    def __init__(self, name):
        self._name = name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Keyword

Setters

```
class Dog:

    def __init__(self, name):
        self._name = name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Name

Setters

```
class Dog:

    def __init__(self, name):
        self._name = name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Parameters

Setters

```python
class Dog:

    def __init__(self, name):
        self._name = name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Setters

```
class Dog:

    def __init__(self, name):
        self._name = name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Body

Setters

```
class Dog:

    def __init__(self, name):
        self._name = name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Condition

Setters

```python
class Dog:

    def __init__(self, name):
        self._name = name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Update

Setters

```
class Dog:

    def __init__(self, name):
        self._name = name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Alternative

Setters

set_ + <attribute>

Setters

set_ + <attribute>

set_name

Setters

set_ + <attribute>

set_name

set_age

Setters

set_ + <attribute>

set_name

set_address

set_age

Setters

set_ + <attribute>

set_name

set_address

set_age

set_id

Setters

set_ + <attribute>

set_name

set_address

set_color

set_age

set_id

Setters

```
>>> class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

```
>>> dog1 = Dog("Nora")
>>> dog1.set_name("Emily")
>>> dog1.get_name()
'Emily'
```

Setters

```
>>> dog1 = Dog("Nora")
>>> dog1.set_name("Emily")
>>> dog1.get_name()
'Emily'
```

Create the instance

Setters

```
>>> dog1 = Dog("Nora")
>>> dog1.set_name("Emily")
>>> dog1.get_name()
'Emily'
```

Change its name

Setters

```
>>> dog1 = Dog("Nora")
>>> dog1.set_name("Emily")
>>> dog1.get_name()
'Emily'
```

Access new name

Setters

```
>>> dog1 = Dog("Nora")
>>> dog1.set_name("Emily")
>>> dog1.get_name()
'Emily'
```

Successfully Modified

Setters

```
>>> class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")


>>> dog1 = Dog("Nora")
>>> dog1.set_name(5)
Please enter a valid name
>>> dog1.get_name()
'Nora'
```

```
def set_name(self, name):
    if isinstance(name, str):
        self._name = name
    else:
        print("Please enter a valid name")
```

Condition

Setters

```
>>> class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")


>>> dog1 = Dog("Nora")
>>> dog1.set_name(5)
Please enter a valid name
>>> dog1.get_name()
'Nora'
```

```
def set_name(self, name):
    if isinstance(name, str):
        self._name = name
    else:
        print("Please enter a valid name")
```

Condition

Setters

```
>>> class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")


>>> dog1 = Dog("Nora")
>>> dog1.set_name(5)
Please enter a valid name
>>> dog1.get_name()
'Nora'
```

```
def set_name(self, name):
    if isinstance(name, str):
        self._name = name
    else:
        print("Please enter a valid name")
```

Alternative

Setters

```
>>> class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")


>>> dog1 = Dog("Nora")
>>> dog1.set_name(5)
Please enter a valid name
>>> dog1.get_name()
'Nora'
```

```
def set_name(self, name):
    if isinstance(name, str):
        self._name = name
    else:
        print("Please enter a valid name")
```

Alternative

Final Class

```python
class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Getter

```python
class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Setter

```python
class Dog:

    def __init__(self, name):
        self._name = name

    def get_name(self):
        return self._name

    def set_name(self, name):
        if isinstance(name, str):
            self._name = name
        else:
            print("Please enter a valid name")
```

Now… An Example