

Занятие 2

Атрибуты классов и объектов



**Атрибуты
класса**

**Атрибуты
объекта**

**Получение
атрибутов**

**Добавление
атрибутов**

**Удаление
атрибутов**


Атрибуты

Переменные внутри класса называют **свойства** или **атрибуты** класса.
Чтобы обратиться к атрибуту класса, используется запись: **<Класс>.<атрибут>**

Вывод:

```
class Point:  
    x = 0  
    y = 0  
    color = 'black'
```

```
print(Point.x, Point.y)  
print(Point.color)  
0 0  
black
```

```
✓  Point = {type} <class '__main__.Point'>  
  01 color = {str} 'black'  
  01 x = {int} 0  
  01 y = {int} 0
```

Атрибуты

Получить набор всех атрибутов класса можно с помощью специальной переменной `__dict__`.

```
class Point:  
    x = 0  
    y = 0  
    color = 'black'  
  
print(Point.__dict__)
```

Вывод:

```
{'__module__': '__main__', 'x': 0, 'y': 0,  
'color': 'black', '__dict__': <attribute  
'__dict__' of 'Point' objects>,  
'__weakref__': <attribute '__weakref__' of  
'Point' objects>, '__doc__': None}
```

Атрибуты класса

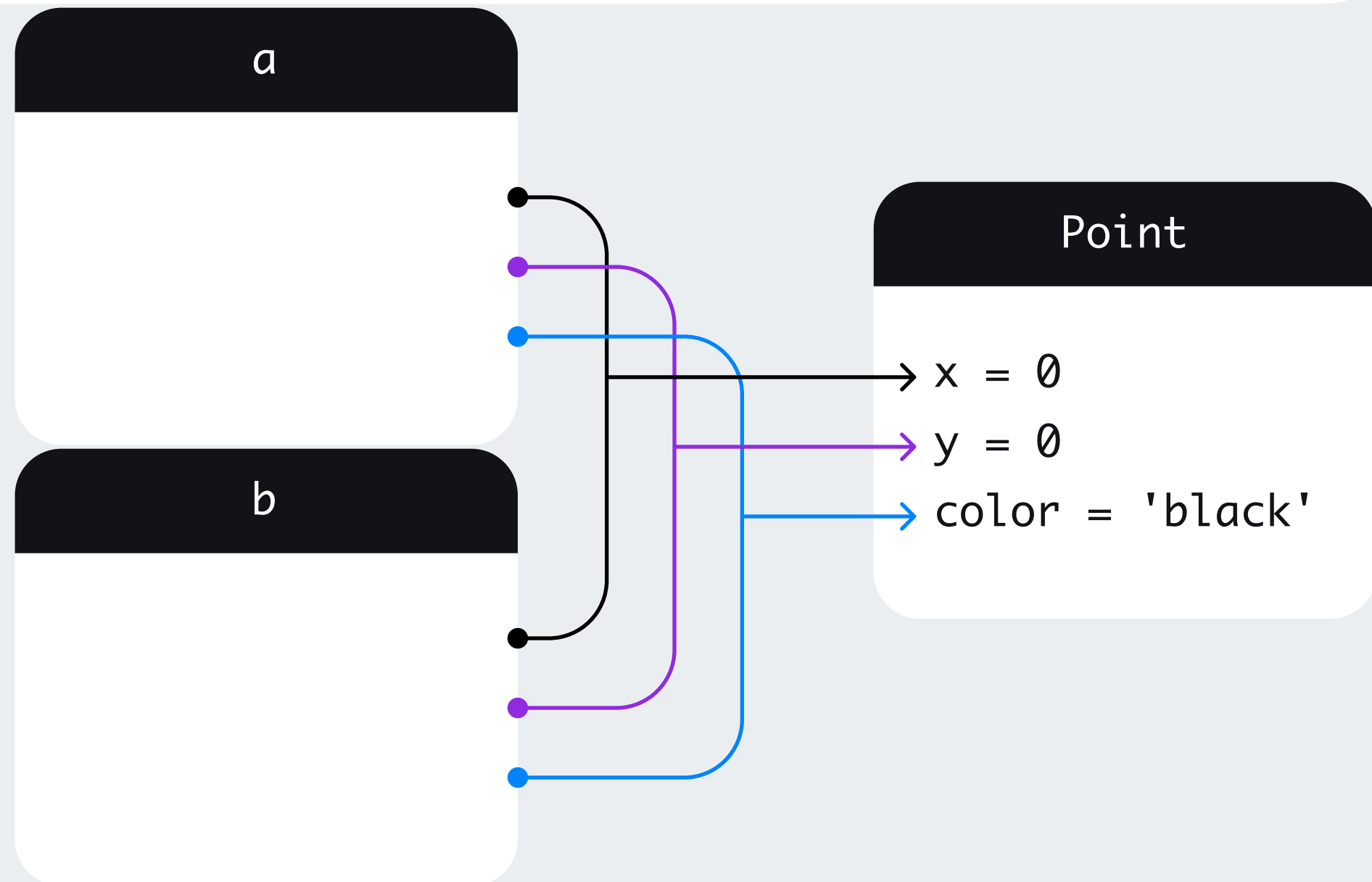
Переменные внутри класса — это атрибуты класса. Атрибутов `x`, `y`, `color` у объектов `a` и `b` не существует, они ссылаются на атрибуты класса, при этом атрибуты класса общие для всех экземпляров.

```
class Point:  
    x = 0  
    y = 0  
    color = 'black'
```

```
a = Point()  
b = Point()  
print(a.__dict__)  
print(b.__dict__)
```

Вывод:

```
{}  
{}
```



Пример

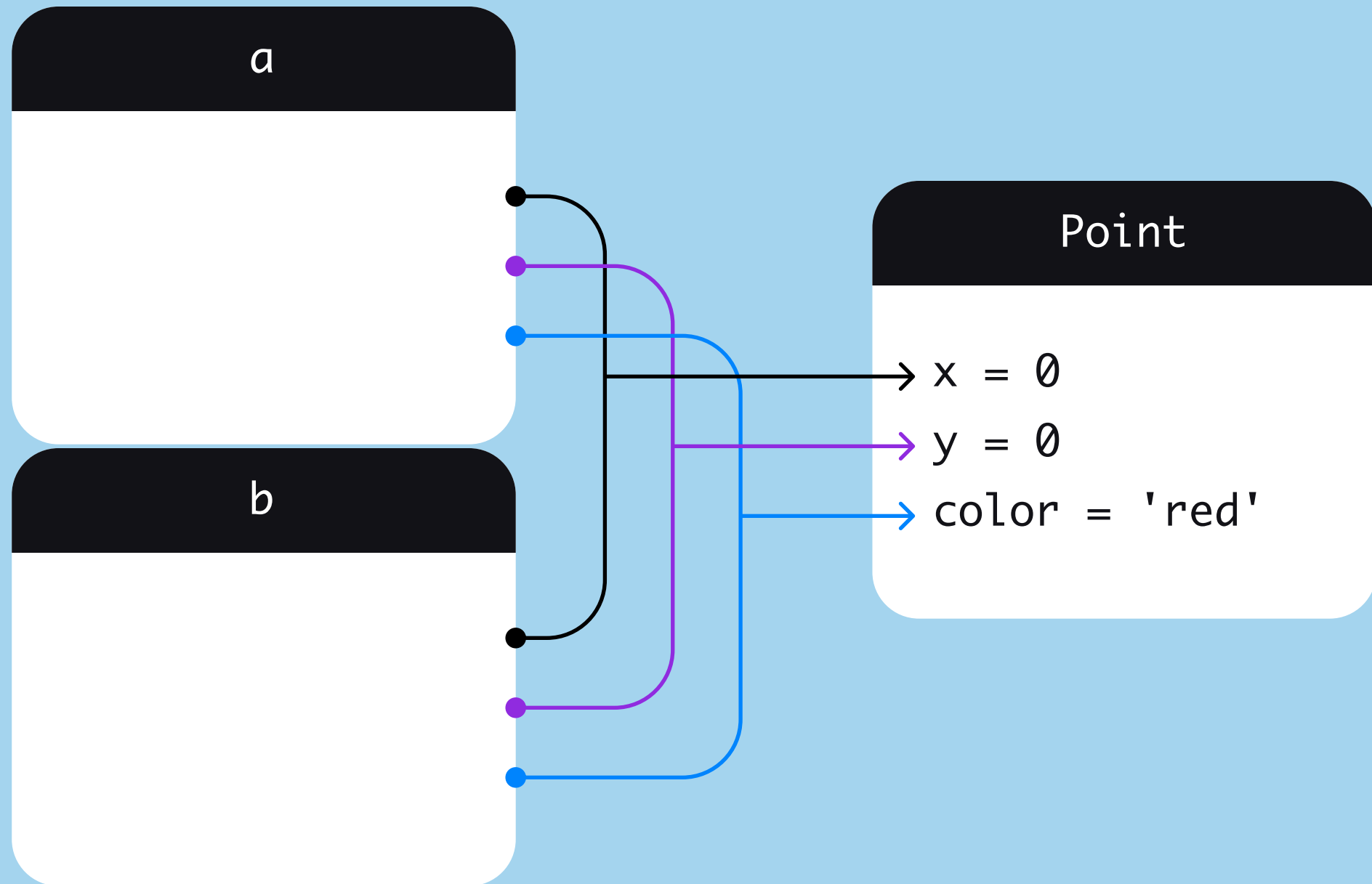


```
class Point:
    x = 0
    y = 0
    color = 'black'

a = Point()
b = Point()
print(a.color, b.color)
Point.color = 'red'
print(a.color, b.color)
```

Вывод:

```
black black
red red
```



Изменение и добавление атрибута класса

Если обратиться к атрибуту класса и присвоить ему новое значение, то можно изменить атрибут класса, если же такого атрибута нет, то он будет добавлен.

```
class Point:  
    x = 0  
    y = 0  
    color = 'black'
```

```
Point.color = 'red'  
Point.size = 5  
print(Point.__dict__)
```

Вывод:

```
{'__module__': '__main__', 'x': 0, 'y': 0,  
'color': 'red', '__dict__': <attribute  
'__dict__' of 'Point' objects>,  
'__weakref__': <attribute '__weakref__' of  
'Point' objects>, '__doc__': None, 'size': 5}
```

Добавление атрибута объекта

Если обратиться к атрибуту объекта, а не класса, то можно добавить атрибут объекту.

```
class Point:  
    x = 0  
    y = 0  
    color = 'black'
```

```
a = Point()  
b = Point()  
a.color = 'yellow'  
print(a.__dict__)  
print(a.color)  
print(b.color)
```

Вывод:

```
{'color': 'yellow'}  
yellow  
black
```


Пространство имен класса



В общем смысле **пространство имен** — это совокупность определенных в настоящий момент имен и информации об объектах, на которые они ссылаются. Можно сказать, что это словарь, в котором имена объектов являются ключами, сами объекты значениями.



Набор атрибутов класса или объекта в некотором смысле тоже образует пространство имен. Важно понимать о пространствах имен то, что между именами атрибутов в разных пространствах имен нет связи, т.е. объекты и классы могут иметь одинаковые атрибуты, но для каждой сущности эти атрибуты будут разные.

Пример



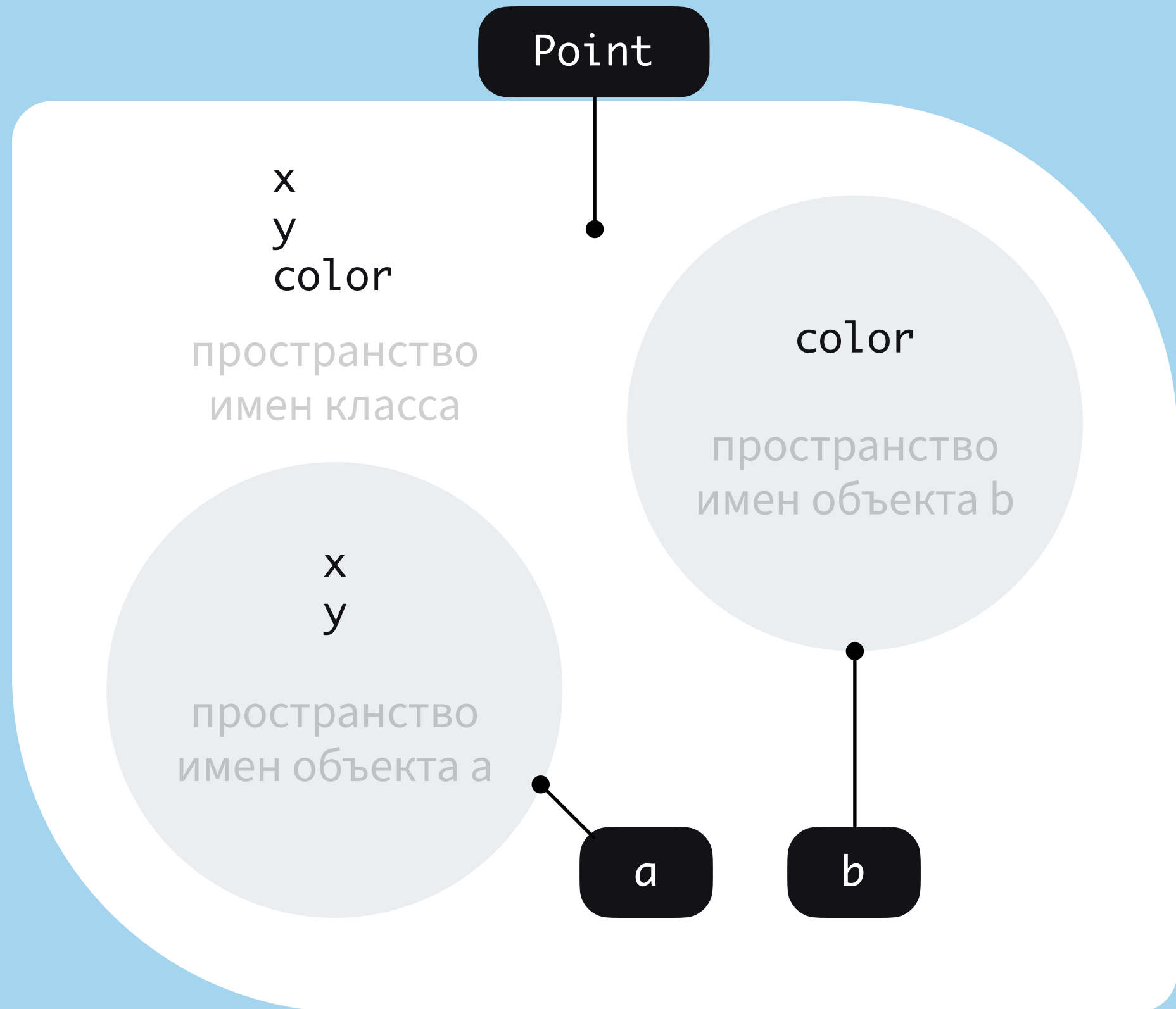
Поиск идет внутри пространства имен экземпляра, если не находит, то ищет внутри класса.

```
class Point:  
    x = 0  
    y = 0  
    color = 'black'
```

```
a = Point()  
b = Point()  
a.x, a.y = 1, 2  
b.color = 'green'  
print(a.x, a.y)  
print(a.color)  
print(b.x, b.y)  
print(b.color)
```

Вывод:

```
1 2  
black  
0 0  
green
```



Добавление атрибута setattr

Добавить атрибут к классу или объекту можно с помощью setattr. Функция setattr добавляет объекту указанный атрибут.

★ Синтаксис: `setattr(obj, name, value)`

```
class Point:
    pass

setattr(Point, 'color', 'black')
a = Point()
setattr(a, 'x', 0)
setattr(a, 'y', 0)
print(Point.color)
print(a.x)
print(a.y)
```

Вывод:

```
black
0
0
```

Получение атрибута getattr

Получить значение атрибута можно с помощью getattr. Функция getattr возвращает значение атрибута объекта.

★ Синтаксис: `getattr(obj, name[, default])`

```
class Point:
    x = 0
    y = 0
    color = 'black'

a = Point()
print(getattr(Point, 'color'))
print(getattr(Point, 'color', 'Нет такого атрибута'))
print(getattr(Point, 'size', 'Нет такого атрибута'))
print(getattr(a, 'color', 'Нет такого атрибута'))
print(getattr(a, 'size'))
```

Вывод:

```
black
black
Нет такого атрибута
black
AttributeError: 'Point' object has no attribute 'size'
```

Удаление атрибута delattr

Удалить атрибут можно с помощью setattr или del. Функция setattr удаляет атрибут объекта.



Синтаксис: **setattr(obj, name)** или **del obj.name**

```
class Point:
    x = 0
    y = 0
    color = 'black'

a = Point()
a.size = 10
del a.size
setattr(Point, 'color')
print(a.__dict__)
print(Point.__dict__)
del a.size
```

Вывод:

```
{
  '__module__': '__main__', 'x': 0, 'y': 0, '__dict__':
  <attribute '__dict__' of 'Point' objects>, '__weakref__':
  <attribute '__weakref__' of 'Point' objects>, '__doc__': None}
AttributeError: size
```

Проверка атрибута hasattr

Проверить, содержит ли объект атрибут можно с помощью hasattr. Функция hasattr возвращает True, если объект содержит указанный атрибут, и False в противном случае.



Синтаксис: `hasattr(obj, name)`

```
class Point:
    x = 0
    y = 0
    color = 'black'

a = Point()
a.size = 10
print(hasattr(Point, 'color'))
print(hasattr(Point, 'size'))
print(hasattr(a, 'size'))
```

Вывод:

```
True
False
True
```

Документ строка docstring

Документ строка или docstring — это строка, которая идет сразу за созданием класса и является удобным способом добавления документации к классу.

Docstring также можно добавлять к модулям, функциям, методам. Доступ к docstring осуществляется через специальную переменную `__doc__`.

```
class Point:
    """Класс описывающий точки на плоскости."""
    x = 0
    y = 0
    color = 'yellow'
```

```
a = Point()
print(Point.__doc__)
print(a.__doc__)
```

Вывод:

Класс описывающий точки на плоскости.
Класс описывающий точки на плоскости.

ИТОГИ



setattr(obj, name, value) добавляет объекту указанный атрибут



getattr(obj, name[, default]) возвращает значение атрибута объекта или значение по умолчанию



delattr(obj, name) удаляет атрибут объекта



hasattr(obj, name) возвращает True если объект содержит указанный атрибут и False в противном случае



__dict__ содержит все атрибуты класса



__doc__ содержит docstring класса