Python面向对象语法精讲

嵩 天

Python类的多态

嵩天

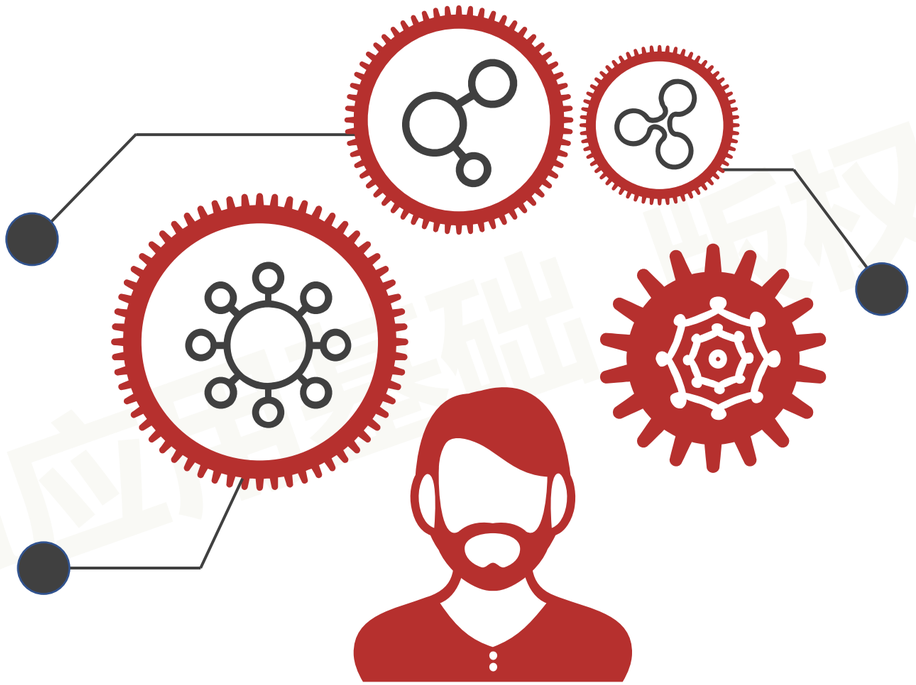
Python类的多态

单元开篇

单元开篇

(2) 参数类型的多态

(1) 多态的理解



(3) 参数形式的多态

Python类的多态

Python类的多态

多态的理解

多态的理解

多态 Polymorphism：仅针对方法，方法灵活性的抽象

- 参数类型的多态：一个方法能够处理多个类型的能力
- 参数形式的多态：一个方法能够接受多个参数的能力
- 多态是OOP的一个传统概念，Python天然支持多态，不需要特殊语法

多态的理解

多态 Polymorphism：方法灵活性的抽象

```
dc1 = DemoClass("老王")
dc2 = DemoClass("老李")
dc1.lucky()
dc1.lucky(10)
dc1.lucky("10")
dc1.lucky(dc2)
dc1.lucky(10, 100)
```

对于同一个类的方法：

方法可以接受多种类型作为其参数

方法可以接受多个参数作为其参数

Python类的多态

参数类型的多态

参数类型的多态

天然支持：Python方法无类型声明限制

- Python的函数/方法没有类型声明限制，天然支持参数类型的多态性
- Python编程理念在于：文档约束，而非语法约束
- 对不同参数类型的区分及功能，需要由程序员完成

参数类型的多态

```python
class DemoClass:
    def __init__(self, name):
        self.name = name

    def __id__(self):
        return len(self.name)

    def lucky(self, salt):
        s = 0
        for c in self.name:
            s += (ord(c) + id(salt)) % 100
        return s

dc1 = DemoClass("老王")
dc2 = DemoClass("老李")
print(dc1.lucky(10))
print(dc1.lucky("10"))
print(dc1.lucky(dc2))
```

方法参数类型的多态性

参数类型的多态

```
class DemoClass:
    def __init__(self, name):
        self.name = name

    def __id__(self):
        return len(self.name)

    def lucky(self, salt):
        s = 0
        for c in self.name:
            s += (ord(c) + id(salt)) % 100
        return s

dc1 = DemoClass("老王")
dc2 = DemoClass("老李")
print(dc1.lucky(10))
print(dc1.lucky("10"))
print(dc1.lucky(dc2))
```

需要方法能够处理不同参数类型

如：整数、字符串和DemoClass类

参数类型的多态

```python
class DemoClass:
    def __init__(self, name):
        self.name = name

    def __id__(self):
        return len(self.name)

    def lucky(self, salt):
        s = 0
        for c in self.name:
            s += (ord(c) + id(salt)) % 100
        return s

dc1 = DemoClass("老王")
dc2 = DemoClass("老李")
print(dc1.lucky(10))
print(dc1.lucky("10"))
print(dc1.lucky(dc2))
```

重载了id()函数对应逻辑

参数类型的多态

```python
class DemoClass:
    def __init__(self, name):
        self.name = name

    def __id__(self):
        return len(self.name)

    def lucky(self, salt):
        s = 0
        for c in self.name:
            s += (ord(c) + id(salt)) % 100
        return s

dc1 = DemoClass("老王")
dc2 = DemoClass("老李")
print(dc1.lucky(10))
print(dc1.lucky("10"))
print(dc1.lucky(dc2))
```

```
68
188
124
```

Python类的多态

参数形式的多态

参数形式的多态

天然支持：Python方法/函数支持多种参数形式

- Python的函数/方法可以支持可变参数，支持参数形式的多态性
- Python的类方法也是函数，函数的各种定义方式均有效
- 对不同参数个数及默认值的确定，需要由程序员完成

参数形式的多态

```
class DemoClass:
    def __init__(self, name):
        self.name = name

    def __id__(self):
        return len(self.name)

    def lucky(self, salt = 0, more = 9):
        s = 0
        for c in self.name:
            s += (ord(c) + id(salt) + more) % 100
        return s

dc1 = DemoClass("老王")
print(dc1.lucky())
print(dc1.lucky(10))
print(dc1.lucky(10, 100))
```

方法参数形式的多态性

参数形式的多态

```
class DemoClass:
    def __init__(self, name):
        self.name = name

    def __id__(self):
        return len(self.name)

    def lucky(self, salt = 0, more = 9):
        s = 0
        for c in self.name:
            s += (ord(c) + id(salt) + more) % 100
        return s

dc1 = DemoClass("老王")
print(dc1.lucky())
print(dc1.lucky(10))
print(dc1.lucky(10, 100))
```

需要方法能够处理可变参数

参数形式的多态

```
class DemoClass:
    def __init__(self, name):
        self.name = name

    def __id__(self):
        return len(self.name)

    def lucky(self, salt = 0, more = 9):
        s = 0
        for c in self.name:
            s += (ord(c) + id(salt) + more) % 100
        return s

dc1 = DemoClass("老王")
print(dc1.lucky())
print(dc1.lucky(10))
print(dc1.lucky(10, 100))
```

```
166
86
68
```

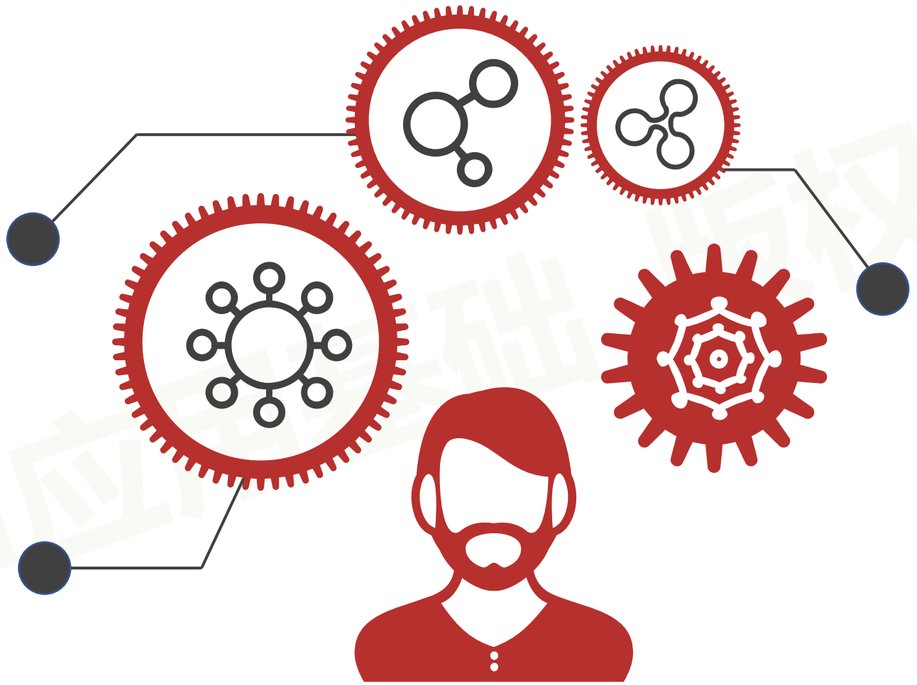
Python类的多态

单元小结

单元小结

(2) 参数类型的多态

天然支持：参数无类型限制



(3) 参数形式的多态

天然支持：函数的可变参数

(1) 多态的理解

方法灵活性的抽象

Python类的多态


Python 123
Thank you