

pf-10. クラス定義, オブジェクト生成, メソッド, 属性

(Python 入門)

URL: <https://www.kkaneko.jp/pro/pf/index.html>

金子邦彦



- 
- ① ソフトウェア設計の理解
 - ② 抽象化

オブジェクトとメソッド



- ・**オブジェクト**：コンピュータでの操作や処理の対象となるもの

hero.moveDown()

hero オブジェクト
moveDown() メソッド
間を「.」で区切っている

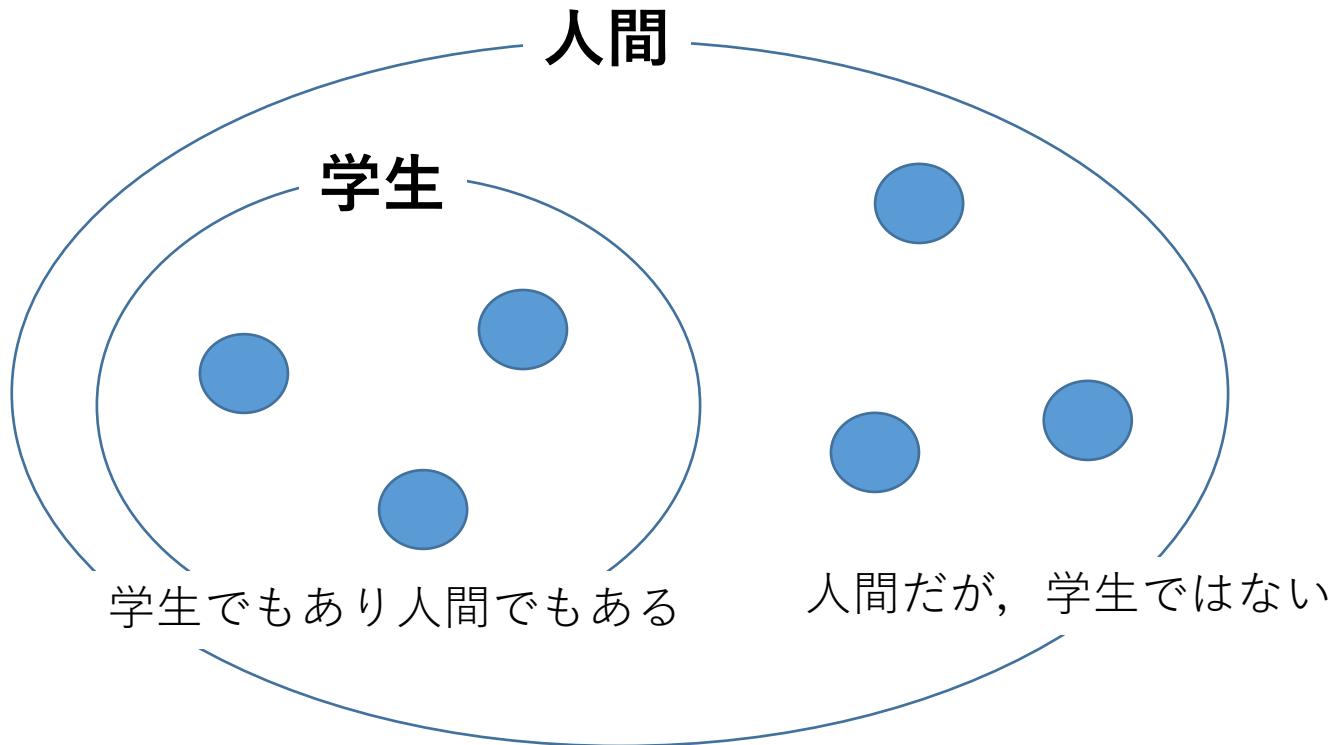
- ・**メソッド**：オブジェクトに属する機能や操作。オブジェクトがもつ能力に相当する
- ・**引数**：メソッドが行う操作の詳細に関する情報，メソッド呼び出しのときに、引数を指定できる

hero.attack("fence", 36, 26)

クラスとオブジェクト



クラスは、同じ種類のオブジェクトの集まりと考え
ることができる



同じクラスの2つのオブジェクト

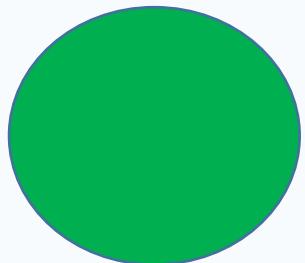


オブジェクト



半径 1, 場所 (8, 10)
色 blue

オブジェクト



半径 3, 場所 (2, 4)
色 green

クラス Ball

- オブジェクトは半径, 場所, 色などの属性を持つことができる

クラス定義の例



```
class Ball:  
    def __init__(self, x, y, r, color):  
        self.x = x  
        self.y = y  
        self.r = r  
        self.color = color  
    def printout(self):  
        print(self.x, self.y, self.r, self.color)
```

クラス名: Ball

属性: x, y, r, color

メソッド: __init__, printout

オブジェクト生成の際に、メソッド __init__ が自動で実行される



クラス定義の例

```
class Ball: クラス名: Ball
    def __init__(self, x, y, r, color):
        self.x = x
        self.y = y               メソッド: __init__
        self.r = r
        self.color = color
    def printout(self):
        print(self.x, self.y, self.r, self.color)
```

クラス名: Ball

メソッド: printout

属性: x, y, r, color

メソッド: __init__, printout

__init__ は、オブジェクトが生成される際に自動的に呼び出されるメソッド

クラスの利用



```
class Ball:  
    def __init__(self, x, y, r, color):  
        self.x = x  
        self.y = y  
        self.r = r  
        self.color = color  
    def printout(self):  
        print(self.x, self.y, self.r, self.color)
```

- ・オブジェクト生成

Ball クラスのオブジェクト生成

- ・属性アクセス

属性 x, y, r, color の値の取得や変更

- ・メソッド呼び出し

メソッド printout の呼び出し

Ball クラスのオブジェクト生成



```
class Ball:  
    def __init__(self, x, y, r, color):  
        self.x = x  
        self.y = y  
        self.r = r  
        self.color = color  
    def printout(self):  
        print(self.x, self.y, self.r, self.color)
```

- Python では、「**クラス名()**」の形式で**オブジェクト生成**を行う

```
a = Ball(8, 10, 1, "blue")  
b = Ball(2, 4, 3, "green")
```

- オブジェクト生成の際に、メソッド `__init__` が自動で実行される

Ball クラスの属性アクセス



```
class Ball:  
    def __init__(self, x, y, r, color):  
        self.x = x  
        self.y = y  
        self.r = r  
        self.color = color  
    def printout(self):  
        print(self.x, self.y, self.r, self.color)
```

- Python では、「オブジェクト.属性名」の形式で属性にアクセスする

```
print(a.x, a.y, a.r, a.color)
```

- 属性値の変更は「オブジェクト.属性名 = 値や式」の形式で行う

```
a.x = 18  
a.y = 20
```

Ball クラスのメソッド呼び出し



```
class Ball:  
    def __init__(self, x, y, r, color):  
        self.x = x  
        self.y = y  
        self.r = r  
        self.color = color  
    def printout(self):  
        print(self.x, self.y, self.r, self.color)
```

- Python では、「オブジェクト. メソッド名 ()」の形式でメソッドを呼び出す

```
a.printout()
```

メソッド定義内の属性アクセス, メソッド呼び出し



- ・メソッド定義内の属性アクセスは, 「**self . 属性名**」の形式
- ・メソッド定義内のメソッド呼び出しは, 「**self . メソッド名 ()**」の形式

```
class Ball:  
    def __init__(self, x, y, r, color):  
        self.x = x  
        self.y = y  
        self.r = r  
        self.color = color  
    def printout(self):  
        print(self.x, self.y, self.r, self.color)
```

- Trinket はオンラインの Python、HTML 等の学習サイト
- 有料の機能と無料の機能がある
- 自分が作成した Python プログラムを公開し、他の人に実行してもらうことが可能（そのとき、書き替えて実行也可能）
- Python の標準機能を登載、その他、次のパッケージがインストール済み

math, matplotlib.pyplot, numpy, operator, processing, pygal,
random, re, string, time, turtle, urllib.request



trinket でのプログラム実行

- trinket は Python, HTML などのプログラムを書き実行できるサイト
- <https://trinket.io/python/0fd59392c8>

のように、違うプログラムには違う URL が割り当てられる

The screenshot shows the trinket.io web interface. At the top, there's a navigation bar with a home icon, 'My Trinkets', and a file name 's11-1'. Below the bar is the trinket logo and a 'Run' button, which is highlighted with a red box. To the left is a code editor window titled 'main.py' containing the following Python code:

```
1 age = 18
2 if age <= 11:
3     print(500)
4 else:
5     print(1800)
```

To the right of the code editor is a 'Result' panel where the output of the program will be displayed.

ソースコードの
メイン画面

実行結果

- 実行が開始しないときは、「実行ボタン」で実行
- ソースコードを書き替えて再度実行することも可能

演習

資料：16

【トピックス】

- クラス定義
- class
- オブジェクト生成
- メソッド呼び出し



① trinket の次のページを開く

<https://trinket.io/python/6fdf17af8f>

② 実行結果が、次のように表示されることを確認

このプログラムは、**オブジェクト a, b を生成する**. そして、
メソッド printout を呼び出して, 属性値を表示させる

```
class Ball:
    def __init__(self, x, y, r, color):
        self.x = x
        self.y = y
        self.r = r
        self.color = color
    def printout(self):
        print(self.x, self.y, self.r, self.color)

a = Ball(8, 10, 1, "blue")
b = Ball(2, 4, 3, "green")

a.printout()
b.printout()
```

Powered by  trinket
(8, 10, 1, 'blue')
(2, 4, 3, 'green')

演習

資料：18

【トピックス】

- クラス定義
- class
- オブジェクト生成
- 属性アクセス

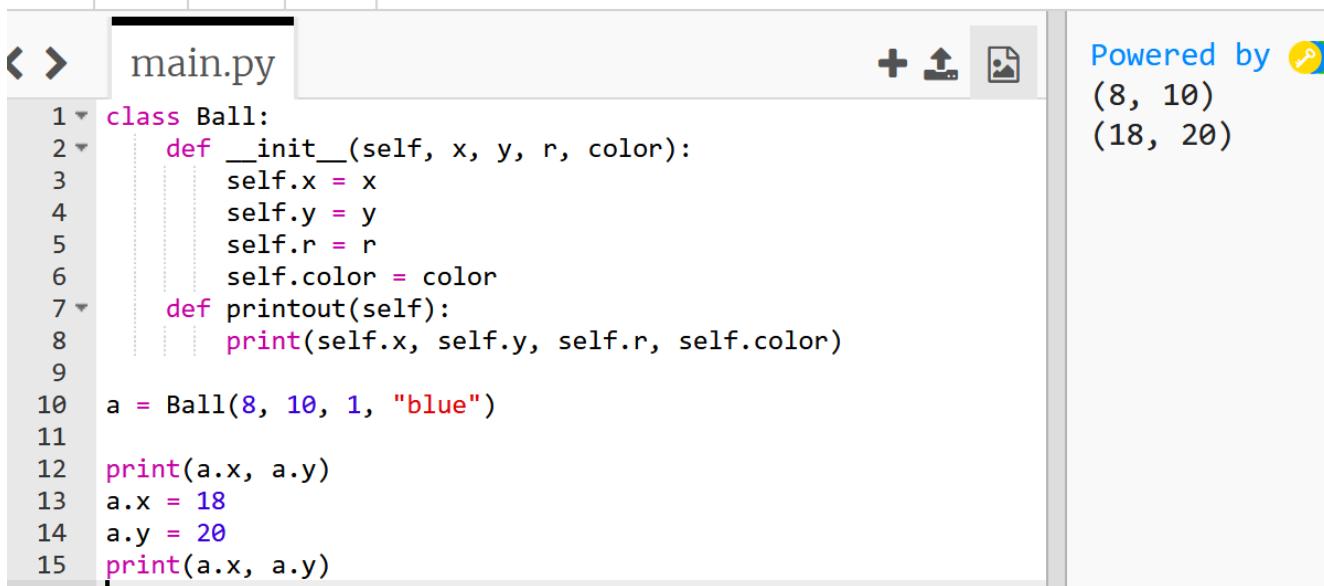


① trinket の次のページを開く

<https://trinket.io/python/2fd4a420de>

② 実行結果が、次のように表示されることを確認

このプログラムは、オブジェクト a を生成する。そして、属性 x, y にアクセスする



The screenshot shows a Python code editor window titled "main.py". The code defines a class "Ball" with an __init__ method that initializes attributes x, y, r, and color. It also defines a printout method that prints these attributes. An instance "a" is created with values (8, 10, 1, "blue"). The code then prints the attributes a.x and a.y, and changes them to 18 and 20 respectively, before printing them again. To the right of the code, the output of the execution is shown: "Powered by" followed by a logo, and then two pairs of coordinates: (8, 10) and (18, 20).

```
1  class Ball:
2      def __init__(self, x, y, r, color):
3          self.x = x
4          self.y = y
5          self.r = r
6          self.color = color
7      def printout(self):
8          print(self.x, self.y, self.r, self.color)
9
10 a = Ball(8, 10, 1, "blue")
11
12 print(a.x, a.y)
13 a.x = 18
14 a.y = 20
15 print(a.x, a.y)
```

① ソフトウェア設計の理解



クラス、メソッド、属性を使用することで、プログラムは読みやすくなり、将来必要となる変更も行いやすくなります。更には、これらの要素を適切に使用することで、全体的なソフトウェアアーキテクチャと設計への理解を深め、効率的で再利用可能なコードを作成することができます。

② 抽象化

同じ種類のオブジェクトをクラスにまとめるという抽象化を行うことで、現実世界の概念をプログラムに取り入れることが容易になるとともに、プログラムが読みやすく再利用しやすくなります。また、抽象化により、システム全体の柔軟性を向上させることができます。

全体まとめ



- ・ クラスは同じ種類のオブジェクトの集まり
- ・ 属性はオブジェクトの状態を表す
- ・ メソッドはオブジェクトに属する機能や操作.
- ・ 次のクラス定義により、「Ball」クラスのオブジェクトの生成、「color」や「x」などの属性アクセス、情報を表示する「printout」メソッドへのアクセスができるようになる

class Ball:

```
def __init__(self, x, y, r, color):
    self.x = x
    self.y = y
    self.r = r
    self.color = color
def printout(self):
    print(self.x, self.y, self.r, self.color)
```

- ・ 次のプログラムにより **x = 8, y = 10, r = 1, color = "blue"** のボールが生成され、表示が行われる
- ```
a = Ball(8, 10, 1, "blue")
a.printout()
```