

# 引き継ぎ資料 Vol.4

オブジェクトの話

---

2016/08/??

# コンセプト

“オブジェクトって何?” を解決

# 目次

1. 導入
2. クラス
3. メンバ
4. 繙承
5. まとめ

# 目次

1. 導入
2. クラス
3. メンバ
4. 繙承
5. まとめ

# 導入

Q.

オブジェクトって何?

# 導入

Q.

オブジェクトって何?

A.

データや構造としての“何か”

# 例

```
>>> int_variable = 10 # int object
>>> float_variable = 3.7 # float object
>>> str_object = "Hello" # str object
>>> none_object = None # NoneType object
>>> list_object = [] # list object
>>> dict_object = {} # dict object

>>> # numpy.ndarray object
>>> array_object = numpy.array([1, 2, 3])
```

# 例

```
>>> int_variable = 10 # int object
>>> float_variable = 3.7 # float object
>>> str_object = "Hello" # str object
>>> none_object = None # NoneType object
>>> list_object = [] # list object
>>> dict_object = {} # dict object

>>> # numpy.ndarray object
>>> array_object = numpy.array([1, 2, 3])
```

なんでもオブジェクト

# オブジェクトの中身

- ・ 関数 (正確にはメソッド)
- ・ 他のオブジェクト

# 例

```
>>> list_object = [1, 2, 3, 4] # list object
>>> # list_objectの中にあるappendメソッド
>>> list_object.append(5)
>>> # list_objectがappendという操作を受け付けて
>>> # リストの中に5を追加
>>> print(list_object)
[1, 2, 3, 4, 5]
>>> # numpy.ndarray object
>>> array_object = numpy.array([
...     [1, 2, 3],[4, 5, 6]
... ])
>>> print(a.shape) # aの中のタプルオブジェクト
(2, 3)
```

# 型

オブジェクトは様々な中身を持つことが可能  
中身のテンプレート → 型

オブジェクトの作成 = 型から物を作成



型って自分で作れないの？

型って自分で作れないの？

自作できる型 → クラス

# 目次

1. 導入
2. クラス
3. メンバ
4. 繙承
5. まとめ

# クラス

Q.

クラスって何?

# クラス

Q.

クラスって何?

A.

C の構造体 $+\alpha$

# クラス

Q.

クラスって何?

A.

C の構造体  $+ \alpha$

$+ \alpha$  の部分がとても巨大

# 復習 - 構造体 -

“車”を表す構造体

```
typedef struct Car{  
    double x;  
    double y;  
} Car;
```

- ・メンバは x と y
- ・それぞれには. でアクセス

## 復習 - 構造体 -

```
int main(int argc, char** argv){  
    Car car1, car2;  
    car1.x = 1.0;  
    car1.y = 1.0;  
    car2.x = 5.0;  
    car2.y = 5.0;  
    return 0;  
}
```

car1 と car2 は別物

# 目次

1. 導入
2. クラス
3. メンバ
4. 繙承
5. まとめ

# 目次

1. 導入
2. クラス
3. メンバ
4. 繙承
5. まとめ

# 目次

1. 導入
2. クラス
3. メンバ
4. 繙承
5. まとめ

“オブジェクト”の概念は使いこなせば便利  
データ解析の分野では作る必要は無い可能性が高い  
使えないのは危険

# Questions?