

공학적 문제해결 기법

10주차 2차시 사물을 클래스로 구분하기





10주차 2차시 : 사물을 클래스로 구분하기

I. 개요

객체와 클래스

II. 학습 개요

1) 학습 목표

객체와 클래스의 개념을 살펴본다. 클래스를 생성해보고,
그 클래스의 객체 (인스턴스)를 만들어 본다.

2) 학습 목차(세부 목차)

- 객체와 클래스란 무엇인가?
- 객체를 클래스에 추가하기
- 객체와 클래스를 사용하는 이유



10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 1 - 객체와 클래스란 무엇인가?

객체

- ☒ 사람들이다니는 보도, 창문, 자동차, 기린 등을 우리는 사물(things)이라고 부름
- ☒ 이러한 사물을 객체(objects)라고 부름
- ☒ 객체에 대한 개념은 컴퓨터 세계에서 매우 중요한 개념들 중의 하나
- ☒ 객체는 프로그램 내에 있는 코드를 구성하는 방법
- ☒ 복잡한 개념을 좀 더 쉽게 만들어 줌
- ☒ 앞서거북이로 그림 그리기에서 Pen이라는 객체를 사용했음



10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 1 - 객체와 클래스란 무엇인가?

객체

 객체가 어떻게 동작하는지 이해하려면 객체들의 종류를 알아야 함



기린과 보도를 예를 들어 설명



기린은 동물의 한 종류인 포유류 중의 하나, 살아있는 동물 객체



보도는 살아있지 않다는 것 말고는 더 할 말이 없음 → 무생물 객체



포유류, 동물, 생물, 무생물 이란 용어는 사물들을 구분하는 방법

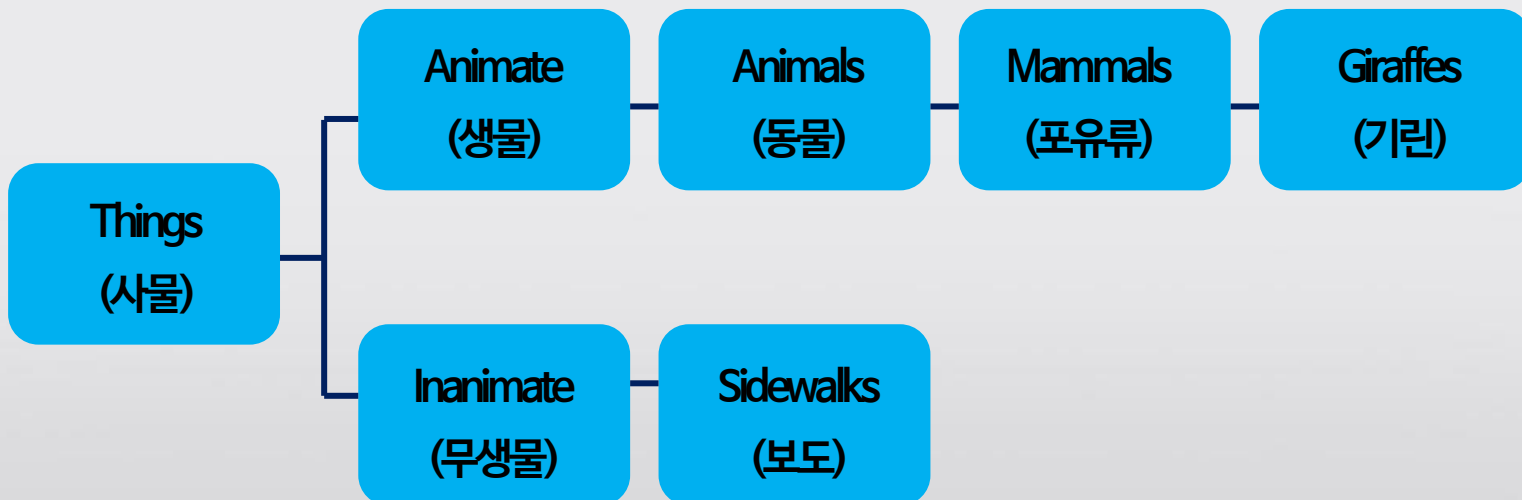


10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 1 - 객체와 클래스란 무엇인가?

클래스

- ✓ 객체는 객체를 그룹으로 구분할 수 있는 클래스(class)에 의해 정의
- ✓ 다음은 앞에서 정의했던 것들을 기반으로 한 기린과 보도에 맞는 클래스의 트리 다이어그램





10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 1 - 객체와 클래스란 무엇인가?

클래스

- ☒ 앞에서 배웠던 turtle 모듈을 생각해 보자
- ☒ 전진과 후진, 좌회전 그리고 우회전과 같이 turtle 모듈이 할 수 있는 모든 작업들은 Pen 클래스에 있는 함수들이다.
- ☒ 객체는 클래스의 구성원(Member)으로 생각할 수 있고
- ☒ 클래스에 객체를 몇 개라도 생성할 수 있다.
- ☒ 앞에서 봤던 트리 다이어그램과 같이 맨 위의 클래스부터 만들어보자.
- ☒ 클래스를 정의할 때는 클래스 이름에 class 라는 키워드를 사용




10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 1 - 객체와 클래스란 무엇인가?

클래스

☒ Things는 최상위 클래스이므로 이것부터 만들어 본다.



```
>>> class Things :  
    pass
```

☒ pass: 클래스나 함수를 만들었지만 그 안에 상세한 내용을 넣고 싶지 않을 경우








10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 1 - 객체와 클래스란 무엇인가?

클래스

 다른 클래스들을 추가하면서 클래스들간의 관계를 알아본다.

-  A라는 클래스가 B라는 클래스의 부분이면, A 클래스는 B 클래스의 자식(child)
-  B 클래스는 A 클래스의 부모(parent)가 됨
-  트리 다이어그램에서 어떤 클래스가 다른 클래스 위에 있으면 부모이고,
-  어떤 클래스 밑에 있으면 자식이 됨
-  Inanimate와 Animate는 Things 클래스의 자식자식이며, 이 말은 Things가 그들의 부모라는 의미



10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 1 - 객체와 클래스란 무엇인가?

클래스

- ✓ 어떤 클래스가 다른 클래스의 자식이라는 것 · 새롭게 만든 클래스 이름 다음에 괄호로 부모 클래스의 이름을 추가하면 됨

➡

```
>>> class Inanimate(Things):  
    pass
```

```
>>> class Animate(Things):  
    pass
```

- ✓ 같은 방법으로 Sidewalks 클래스를 만들어 보면, 부모 클래스를 Inanimate 클래스로 하여 Sidewalks 클래스를 생성

➡

```
>>> class Sidewalks(Inanimate):  
    pass
```

- ✓ 같은 방법으로 Animals와 Mammals, Giraffes 클래스도 만들어 보자




10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 1 - 객체와 클래스란 무엇인가?

클래스

 Animals와 Mammals, Giraffes 클래스



```
>>> class Animals(Animate):  
    pass  
  
>>> class Mammals(Animals):  
    pass  
  
>>> class Giraffes(Mammals):  
    pass
```




10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 2 - 객체를 클래스에 추가하기

클래스들에 어떤 것을 조금 넣어 보기

- ☒ 우리에게reginald라는이름의기린이있다고가정
- ☒ 그기린은 Giraffes 클래스에속한다
- ☒ 프로그래밍에서reginald라고 불리는기린을어떻게나타낼까?
- ☒ Giraffes 클래스의한객체(objects)를 reginald라고 부를 것이다.
- ☒ reginald이 Giraffes 클래스의한객체임을알려주기위한 코드

 >>> reginald = Giraffes()



10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 2 - 객체를 클래스에 추가하기

클래스들에 어떤 것을 조금 넣어 보기

- ☒ Giraffes 클래스 객체를 생성하고 변수 `reginald`에 할당
- ☒ 함수처럼 클래스 이름 다음에 괄호가 온다.
- ☒ 그렇다면, `reginald` 객체는 무슨 동작을 할까?
- ☒ 지금은 아무런 동작을 하지 않는다.
- ☒ 이 객체를 쓸모 있게 만들려면 → 클래스를 생성할 때 그 클래스의 객체를 가지고 사용할 수 있는 함수들을 정의해야 함



10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 2 - 객체를 클래스에 추가하기

클래스들에 어떤 것을 조금 넣어 보기

- ✓ 즉, 클래스 정의를 한 다음에 pass 키워드를 사용하지 말고 함수에 대한 정의를 추가해야 함
- ✓ 클래스와 관련된 함수를 정의할 경우
- ✓ 클래스를 정의한 곳 안쪽에 들여쓰기를 한다는 것을 제외하면
- ✓ 다른 함수를 정의하는 것과 동일한 방법을 사용
- ✓ 다음의 두 함수는 클래스에 속하는 함수이다.



```
>>> class ThisIsMySillyClass :  
    def this_is_a_class_function() :  
        print('I am a class function')  
    def this_is_also_a_class_function() :  
        print('I am also a class function. See?')
```



10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 2 - 객체를 클래스에 추가하기

클래스 특성을 함수로 추가하기

- ☒ 앞에서 정의했던 Animate 클래스의 하위 클래스를 살펴보자.
- ☒ 각각의 클래스가 무엇인지, 무슨일을 할 수 있는지를 설명하기 위한 특성(Characteristics)을 추가할 수 있다.
- ☒ 특성은 클래스의 멤버들(클래스의 자식들) 모두가 공유한다.
- ☒ 모든 동물들이 갖는 일반적인 동작은 무엇일까?
 - ➡ 모두가 숨을 쉬고, 움직이며 음식을 먹는다.
- ☒ 포유류는 어떨까?
 - ➡ 자식들에게 젖을 먹이고, 물론 숨도 쉬고 움직이며, 음식을 먹는다.






10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 2 - 객체를 클래스에 추가하기

클래스 특성을 함수로 추가하기

 기린은?

-  나무 위의 잎사귀를 먹으며
-  다른 모든 포유류들처럼 자식들에게 젖을 먹이며
-  숨을 쉬고 움직이며, 음식을 먹는다

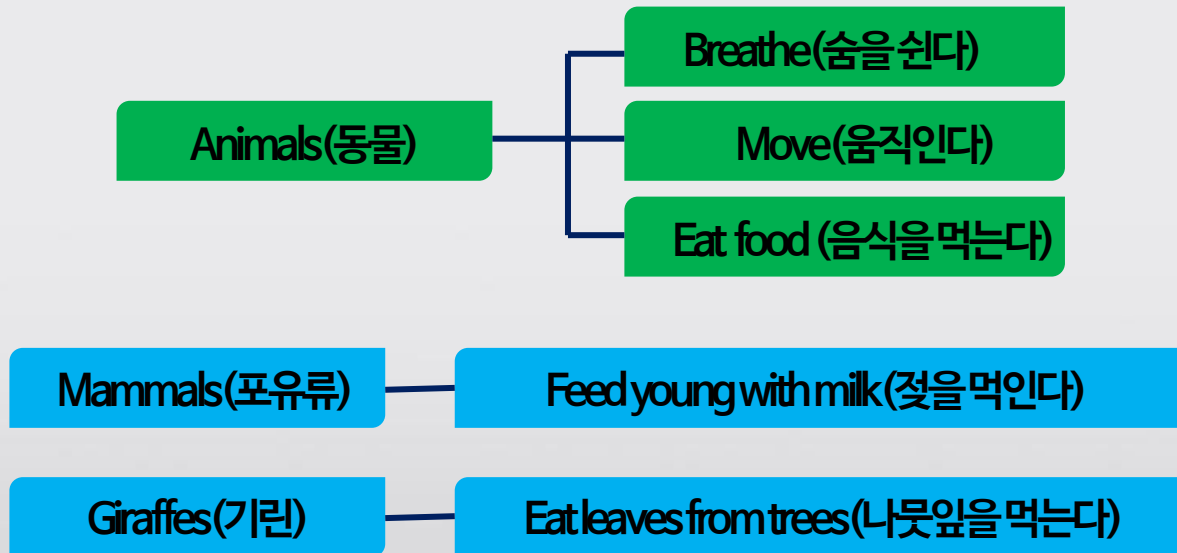


10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 2 - 객체를 클래스에 추가하기

클래스 특성을 함수로 추가하기

 이러한 특성들을 트리 다이어그램으로 나타내면





10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 2 - 객체를 클래스에 추가하기

클래스 특성을 함수로 추가하기

- ☒ 이러한 특성들을 동작, 즉 그 클래스의 객체가 할 수 있는 함수(function)처럼 생각할 수 있다.
- ☒ 클래스에 함수를 추가하려면 def 키워드를 사용
- ☒ Animals 클래스는 다음과 같을 것이다.

➡

```
>>> class Animals(Animate):  
    def breathe(self):  
        pass  
    def move(self):  
        pass  
    def eat_food(self):  
        pass
```



10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 2 - 객체를 클래스에 추가하기

클래스 특성을 함수로 추가하기

- ☒ Mammals 클래스와 Giraffes 클래스에도 함수를 추가할 수 있다.
- ☒ 각각의 클래스는 자신의 부모 클래스의 특성(함수)을 사용할 수 있다.

➡

```
>>> class Mammals(Animals):  
    def feed_young_with_milk(self):  
        pass
```

```
>>> class Giraffes(Mammals):  
    def eat_leaves_from_trees(self):  
        pass
```



10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 3 - 객체와 클래스를 사용하는 이유

클래스와 객체를 사용하는 이유가 무엇일까?

- ☒ 클래스에 함수를 추가했지만, breathe와 move, eat_food라는 일반적인 함수들을 언제 만들 수 있을까?
- ☒ Giraffes 클래스의 객체로 생성했던 reginald라는 이름의 기린을 사용해 보자.
- ☒ reginald는 객체이므로 그 클래스(Giraffes 클래스)와 부모 클래스들이 제공하는 함수를 호출할 수 있다.
- ☒ reginald에게 움직이거나 먹으라고 말하려면 다음과 같이 함수를 호출하면 됨
 - ➔


```
>>> reginald = Giraffes()  
>>> reginald.move()  
>>> reginald.eat_leaves_from_trees()
```



10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 3 - 객체와 클래스를 사용하는 이유

클래스와 객체를 사용하는 이유가 무엇일까?

- ☒ reginald에게 harold라는 이름의 기린 친구가 있다면
- ☒ harold라는 이름의 또 다른 Giraffes 객체를 생성할 수 있다.
 -  >>> harold = Giraffes()
- ☒ 클래스를 수정하여 좀 더 구체적으로 만들어 봅시다.



10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 3 - 객체와 클래스를 사용하는 이유

클래스와 객체를 사용하는 이유가 무엇일까?

 pass를 사용하던 곳에 pass 대신 print문을 각 함수마다 추가한다.

```
>>> class Animals(Animate):  
    def breathe(self):  
        print('breathing')  
    def move(self):  
        print('moving')  
    def eat_food(self):  
        print('eating food')  
  
>>> class Mammals(Animals):  
    def feed_young_with_milk(self):  
        print('feeding young')  
  
>>> class Giraffes(Mammals):  
    def eat_leaves_from_trees(self):  
        print('eating leaves')
```




10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 3 - 객체와 클래스를 사용하는 이유

클래스와 객체를 사용하는 이유가 무엇일까?

- ☒ reginald 객체와 harold 객체를 생성
- ☒ 그 객체들의 함수를 호출하면 실제로 어떤 동작이 일어나는 것을 볼 수 있다.



```
>>> reginald = Giraffes()  
>>> harold = Giraffes()  
>>> reginald.move()  
>>> harold.eat_leaves_from_trees()
```

- ☒ 만약에 이들이 컴퓨터에 있는 객체가 아니라 진짜 기린이라면
- ☒ 한 기린은 걸을 것이고, 다른 기린은 잎사귀를 먹을 것이다.



10주차 2차시 : 사물을 클래스로 구분하기

III. 학습 3 - 객체와 클래스를 사용하는 이유

 클래스와 객체를 사용하는 이유가 무엇일까?

 실행 결과

```
74 giraffes1.py - C:\Documents and Settings\Administrator\바탕 화면\파이썬W...
File Edit Format Run Options Windows Help

class Things:
    pass

class Inanimate(Things):
    pass

class Animate(Things):
    pass

class Animals(Animate):
    def breathe(self):
        print('breathing')
    def move(self):
        print('moving')
    def eat_food(self):
        print('eating')

class Mammals(Animals):
    def feed_young_with_milk(self):
        print('feeding with milk')

Python 3.3.3 Shell
File Edit Shell Debug Options Windows Help
Python 3.3.3 (v3.3.3:c3896275c0f6, Nov 18 2013, 21:18:40) on win32
Type "copyright", "credits" or "license()" for more info
>>> ===== RESTART =====
>>>
>>> moving
>>> eating leaves
>>> |
```



『 이 콘텐츠는 2014학년도 학부교육 선도대학 육성사업(ACE)에 의하여 개발한 것임 』

