UNISTUDY

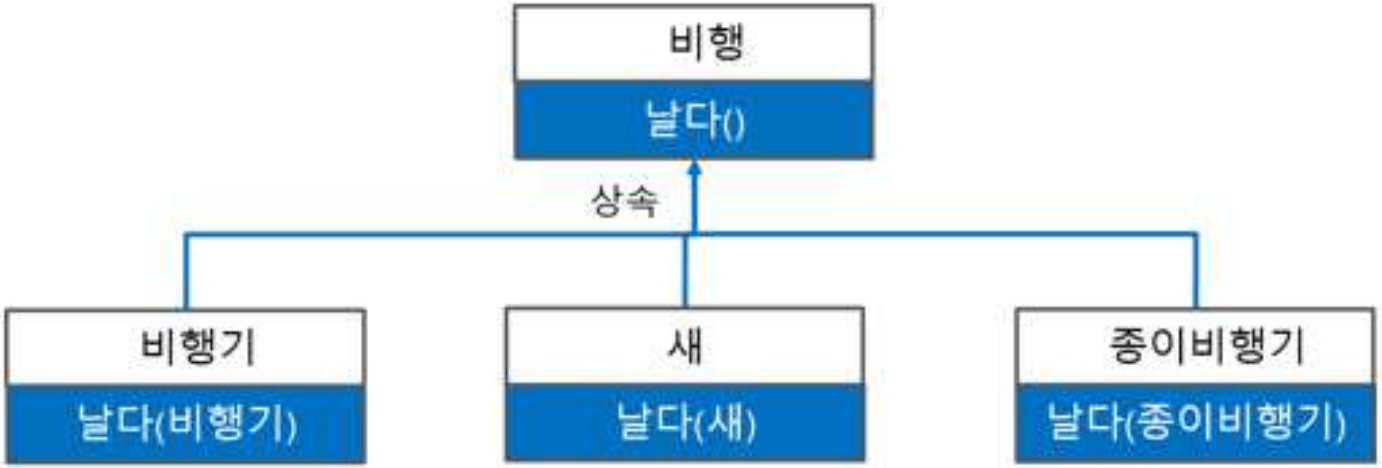
파이썬

23강. 객체지향 기법(다형성, 내장클래스)

1. 다형성

- 다형성(polymorphism)이라는 용어는 '여러 가지 형태를 가질 수 있는 능력'을 말한다. 예를 들면 파 이썬에서 플러스(+) 기호는 산술연산의 덧셈기능과 문자열 연산에서 결합기능을 갖는다. 이처럼 플러 스라는 기호 하나로 두 가지 기능을 갖는 능력을 다형성이라고 한다. 그렇다면 객체지향프로그래밍에 서 다형성은 어떤 의미를 갖고 있는지를 살펴보자. '하나의 참조변수로 여러 타입(type)의 객체를 참조 할 수 있는 것' 으로 정의된다.
- 다시 말해서 부모 객체의 참조변수로 자식 객체를 다룰 수 있다는 의미가 객체지향기법에서 다형성이 다. 따라서 다형성은 클래스의 상속관계에서만 나올 수 있는 용어이다.
- '비행' 클래스를 하위 3개의 클래스로 상속하는 과정에서 부모 클래스의 '날다()' 메 서드가 자식 클래스에서 날다(비행기), 날다(새), 날다(종이비행기) 형태로 재정의 된다. 결국 클래스 간의 상속관계를 통해서 메서드가 재정의 되고, 이를 토대로 클래스의 다형성이 구현된다.

1. 다형성



1. 다형성

chapter06.lecture.step07_polymorphism.py

```
# (1) 부모 클래스
class Flight:

    # 부모 원형 함수
    def fly(self):
        print('날다, fly 원형 메서드')

# (2) 자식 클래스 : 비행기
class Airplane(Flight) :

    # 함수 재정의
    def fly(self):
        print('비행기가 날다.')
```

Python Console

```
날다, fly 원형 메서드
비행기가 날다.
```

1. 다형성

```
# (2) 자식 클래스 : 새
class Bird(Flight) :

    # 함수 재정의
    def fly(self):
        print('새가 날다.')

# (2) 자식 클래스 : 종이비행기
class PaperAirplane(Flight) :

    # 함수 재정의
    def fly(self):
        print('종이 비행기가 날다.')
```

```
새가 날다.

종이 비행기가 날다.
```

1. 다형성

```
# (3) 객체 생성
# 부모 객체 = 자식 객체(자식1, 자식2)
flight = Flight()  # 부모 클래스 객체
air = Airplane()  # 자식1 클래스 객체
bird = Bird()  # 자식2 클래스 객체
paper = PaperAirplane()  # 자식3 클래스 객체

# (4) 다형성
flight.fly() # 날다, fly 원형 메서드

flight =  air
flight.fly() # 비행기가 날다.

flight = bird
flight.fly() # 새가 날다.

flight = paper
flight.fly() # 종이 비행기가 날다.
```

1. 다형성

- 다형성 예
- # (1) 부모 클래스
- Flight 클래스는 fly() 메서드를 갖는다.
- # (2) 자식 클래스 : 비행기
- Airplane 클래스는 Flight 클래스를 상속받아서 만들어졌다. 따라서 부모 클래스의 fly() 메서드를 상속받는다. 그리고 상속 받은 메서드를 재정의 하였다.
- # (2) 자식 클래스 : 새
- Bird 클래스는 Flight 클래스를 상속받아서 만들어졌다. 따라서 부모 클래스의 fly() 메서드를
- 상속받는다. 그리고 상속 받은 메서드를 재정의 하였다.
- # (2) 자식 클래스 : 종이비행기
- PaperAirplane 클래스는 Flight 클래스를 상속받아서 만들어졌다. 따라서 부모 클래스의 fly()
- 메서드를 상속받는다. 그리고 상속 받은 메서드를 재정의 하였다.

1. 다형성

- # (3) 객체 생성
- 부모 클래스와 자식 클래스 3개를 대상으로 각각 기본생성자를 이용하여 객체를 생성한다.
- # (4) 다형성
- 부모 객체의 참조변수인 flight에 자식 객체의 참조변수를 할당해 주면 부모 객체의 참조변수를 이용하여 해당 자식 객체의 멤버를 호출할 수 있다. 다음은 다형성을 위한 참조변수의 할당 예이다.

- 부모객체 참조변수 = 자식객체 참조변수

2. 내장클래스

- 클래스는 함수와 마찬가지로 사용자가 직접 정의하는 사용자 클래스와 라이브러리 형식으로 제공되는 내장클래스로 분류된다. 파이썬에서는 풍부한 라이브러리를 통해서 수많은 함수와 클래스를 제공하고, 이를 이용하여 애플리케이션을 개발하거나 자료 분석에 활용한다.
- '제5장 모듈과 함수' 편에서 공부한 내용 중『모듈과 패키지에서 제공되는 함수를 이용하기 위해서는 먼저 import 명령어를 이용하여 해당 모듈이나 패키지를 작성중인 파일에 포함시켜야 한다.』라고 했 던 것처럼 모듈이나 패키지는 함수뿐만 아니라 클래스도 제공한다.
- 파이썬에서 제공하는 내장클래스를 사용하기 위해서는 다음과 같은 형식으로 import 명령어를 이용 하여 모듈을 포함시켜야한다. 형식1)은 해당 모듈에서 제공하는 모든 함수 또는 클래스를 가져온다. 형 식2)는 해당 모듈에서 특정 클래스만 가져온다.

2. 내장클래스

- 한편 builtins 모듈에서 제공하는 내장함수와 동일하게 내장클래스도 import 없이 사용할 수 있다.

형식

```
1)    import 모듈명
2)    from 모듈명 import 클래스명1, 클래스명2,...
```

3. builtins 모듈 내장클래스

- import가 필요 없는 builtins 모듈의 enumerate 내장클래스의 생성자를 이용하여 객체를 생성 하는 예문이다. enumerate 내장클래스는 열거형 자료를 순회하여 색인(index)과 값을 반환하는 객 체를 생성한다.

형식

```
1)    import 모듈명
2)    from 모듈명 import 클래스명1, 클래스명2,...
```

3. builtins 모듈 내장클래스

chapter06.lecture.step08_import_class.py

```
# (1) 리스트 열거형 객체 이용
lst = [1,3,5]
for i, c in enumerate(lst) :
    print('색인 : ', i, end=', ')
    print('내용 :', c)

# (2) 튜플 열거형 객체 이용
dit = {'name' : '홍길동', 'job': '회사원', 'addr':'서울시'}
for i, k in enumerate(dit) :
    print('순서 : ', i, end=', ')
    print('키 :', k, end=', ')
    print('값 :', dit[k])
```

Python Console

```
색인 : 0, 내용 : 1
색인 : 1, 내용 : 3
색인 : 2, 내용 : 5
순서 : 0, 키 : name, 값 : 홍길동
순서 : 1, 키 : job, 값 : 회사원
순서 : 2, 키 : addr, 값 : 서울시
```

3. builtins 모듈 내장클래스

- builtins모듈 내장클래스 예
- # (1) 리스트 열거형 객체 이용
- 리스트(lst) 열거형 객체를 enumerate 생성자의 인수로 사용할 경우 열거형 객체의 원소를 순회하면 서 색인(원소의 위치)과 내용(원소)을 반환한다.
- # (2) 튜플 열거형 객체 이용
- 튜플(dit) 열거형 객체를 enumerate 생성자의 인수로 사용할 경우 열거형 객체의 원소를 순회하면서 순서(꺼내온 원소의 순서)와 내용(키)을 반환한다. 내용을 이용하여 값을 출력한다.

4. import 모듈 내장클래스

- import가 필요한 datetime 모듈의 date와 time 내장클래스의 생성자를 이용하여 객체를 생성하 는 예문이다. date 내장클래스는 연월일을 인수로 받아서 iso 형식의 날짜 객체를 생성하고, time 내 장클래스는 시분초를 인수로 받아서 iso 형식의 시간 객체를 생성하는 클래스이다.

4. import 모듈 내장클래스

chapter06.lecture.step08_import_class.py

```
# (1) 모듈 내장클래스 import
import datetime # 모듈 import
from datetime import date, time


# (2) date 클래스
help(date) # date 클래스 도움말

today = date(2019, 10, 23) # date 객체 생성
print(today) # date 객체 정보

# date 객체 멤버변수 호출
print(today.year) # 2019
print(today.month) # 10
print(today.day) # 23

# date 객체 메서드 호출
w = today.weekday() # Monday==0 ... Sunday==6
print('요일 정보 : ', w) # 요일 정보
```

Python Console

```
Help on class date in mod
ule datetime:
class date(builtins.obje
ct)
 |  date(year, month, da
y) --> date object


2019-10-23



2019
10
23


요일 정보 :  2
```

4. import 모듈 내장클래스

```
# (3) time 클래스
help(time) # time 클래스 도움말

currTime = time(21, 4, 30) # time 객체 생성
print(currTime) # time 객체 정보

# time 객체 멤버변수 호출
print(currTime.hour) # 21
print(currTime.minute) # 4
print(currTime.second) # 30

# time 객체 메서드 호출
isoTime = currTime.isoformat() # HH:MM:SS
print(isoTime)
```

```
Help on class time in mod
ule datetime:
class time(builtins.objec
t) | time([hour[, minute
[, second[, microsecond[,
tzinfo]]]]]) --> a time o
bject

21:04:30

21
4
30

21:04:30
```

4. import 모듈 내장클래스

- import 모듈 내장클래스 예
- # (1) 모듈과 내장 클래스 import
- datetime 모듈의 전체 멤버(함수 or 클래스)를 포함시키는 명령문과 datetime 모듈에서 date와 time 클래스를 포함시키는 명령문이다.
- # (2) date 클래스
- date 클래스의 도움말(help(date))을 보면 생성자 'date(year, month, day)'에 의해서 date object가 생성된다. 따라서 date(2019, 10, 23) 형식으로 년,월,일 실인수에 의해서 today 객체가 생성된다. today 객체를 출력하면 iso 형식의 날짜("YYYY-MM-DD")가 출력된다. today 객체를 이용하여 year, month, day 멤버변수를 호출할 수 있고, weekday() 메서드를 호출하여 요일 정보를 받을 수 있다. 요일 정보는 Monday == 0 ... Sunday == 6와 같이 월요일이면 0, 일요일이면 6이 반환된다.

4. import 모듈 내장클래스

- # (3) time 클래스
- time 클래스의 도움말(help(time))을 보면 생성자 'time([hour[, minute[, second[, microsecond[, tzinfo]]]]])'에 의해서 time object가 생성된다. 따라서 time(21, 4, 30) 형식으로 시,분,초 실인수에 의해서 currTime 객체가 생성된다. currTime 객체를 이용하여 hour, minute, second 멤버변수를 호출할 수 있고, isoformat() 메서드를 호출하면 iso 형식의 시간("HH:MM:SS")을 받을 수 있다.

UNISTUDY

THANK YOU