Реферат

Тема: Спадкування

Спадкування передбачає те, що дочірній клас містить усі атрибути батьківського класу, причому деякі з них можуть бути перевизначені або додані в дочірньому.

Спадкування дозволяє створювати новий клас на основі вже існуючого класу. Поряд з інкапсуляцією успадкування є одним із наріжних каменів об'єктно-орієнтованого дизайну.

Ключовими поняттями успадкування є підклас та суперклас. Підклас успадковує від суперкласу всі публічні атрибути та методи. Суперклас ще називається базовим (base class) або батьківським (parent class), а підклас – похідним (derived class) або дочірнім (child class).

Синтаксис для успадкування класів виглядає так:

class подкласс (суперкласс):

методы\_подкласса

Наприклад, було створено клас Person, який представляє людину. Припустимо, нам потрібний клас працівника, який працює на деякому підприємстві. Ми могли б створити з нуля новий клас, наприклад клас Employee. Однак він може мати ті ж атрибути та методи, що й клас Person, тому що співробітник – це людина. Тому немає сенсу визначати у класі Employee той самий функціонал, що й у класі Person. І в цьому випадку краще застосувати успадкування.

class Person:

def \_\_init\_\_(self, name, age):

self.\_\_name = name

self.\_\_age = age

@property

def age(self):

return self.\_\_age

@age.setter

def age(self, age):

if age in range(1, 100):

self.\_\_age = age

else:

print("Недопустимый возраст")

@property

def name(self):

return self.\_\_name

def display\_info(self):

print("Имя:", self.\_\_name, "\tВозраст:", self.\_\_age)

class Employee(Person):

def details(self, company):

# print(self.\_\_name, "работает в компании", company) # так нельзя, self.\_\_name - приватный атрибут

print(self.name, "работает в компании", company)

tom = Employee("Tom", 23)

tom.details("Google")

tom.age = 33

tom.display\_info()

Клас Employee повністю переймає функціонал класу Person і на додаток до нього додає метод details().

Варто звернути увагу, що для Employee доступні через ключове слово self всі методи та атрибути класу Person, крім закритих атрибутів типу \_\_name або \_\_age.

Під час створення об'єкта Employee ми фактично використовуємо конструктор класу Person. І крім того, цей об'єкт ми можемо викликати всі методи класу Person.

1. Що таке наслідування?
2. Навіщо потрібно наслідування?
3. Чи потрібно наслідування?
4. Яку проблему вирішує наслідування?
5. Чи є клас, від якого наслідуються усі класи?
6. Чому особливий клас object?
7. Чому у темі про наслідування обов’язково згадується клас object?
8. Чи можна наслідуватись від декількох класів?
9. Чи можна наслідуватись від інших класів?
10. Чи можна вирішити усі проблеми з дублікацією коду за допомогою наслідування?
11. Чи може існувати наслідування від наслідуваного класу?
12. Чи потрібно підключати бібліотеки для наслідування?
13. Чи можна перевантажувати методи у наслідуваного класу?
14. Як вказати наслідуваний клас?
15. Що вказується у круглих дужках після назви класу при описі класу?
16. Що таке базовий клас?
17. Як створити базовий клас?
18. Як зробити клас базовим?
19. Що позначає слово self?
20. Що таке child клас?
21. Чи можна звернутися до властивостей базового класу з child класу?
22. Чи можна звернутися до прайвет властивостей базового класу з child класу?
23. Чи необхідно викликати конструктор базового класу?
24. Чи викликається конструктор базового класу?
25. Конструктор базового класу буде викликаний перед конструктором child класу?

Джерела:

https://uk.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B0%D1%81\_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F)

https://metanit.com/python/tutorial/7.3.php