Mittuniversitetet
MID SWEDEN UNIVERSITY

DT094G – Klasser i Python

Lennart Franked

13 mars 2023

Institutionen för Data- och Elektroteknik (DET)
Mittuniversitetet

Mittuniversitetet
MID SWEDEN UNIVERSITY

1. Klasser

Läsanvisningar

Denna sektion behandlar [1, Kap 9.3, 9.4], läs relevanta delar och studera exempelkod.

- Klass
- Metod
- Variabel
- Instans
- Objekt

Klasser

- En klass brukar liknas med en ritning.
- Definierar upp specifikationer som ska gälla.
- Variabler, metoder och uppbyggnad.

Listing 1: Funktion

```
1  class ClassName:
2    def __init__(self):
3      self.test = ''
4
5    def setTest(self,text):
6      self.test = text
7
8    def getTest(self):
9      return self.test
10   ...
11   ...
12   ...
```

Objekt

- Det man bygger av denna ritning kallar vi för ett objekt.
- Varje objekt är unikt och namnges olika för att särskilja dem.
- När man skapar ett objekt kallar vi detta att instantiera ett objekt.

```
1 object1 = KlassNamn()
2 object1.setTest('My first object')
3 object2 = KlassNamn()
4 object2.setTest('My second object')
5 ...
```

Objekt forts.

- Vi kan arbeta med varje objekt genom att anropa dess namn och metod
- `objekt1.metod()`

```
1 print(object1.getTest())
2 My first object
3 print(object2.getTest())
4 My second object
```

self

- Ett scope är ett område där en variabel eller funktion existerar och är åtkomstbar.
- För att få åtkomst till variabler inom en klass så använder man en referens till det egna objektet (self)
- `self` anges i __init__ metoden, vilket är den metod som initieras vid skapandet av ett objekt och bör alltid vara med.
- `self` kan bytas ut mot något annat namn, men det är starkt rekommenderat att använda just `self`.

Scopes

```
1  class MyClass():
2    def __init__(self):
3      self.classvariable = 10
4      denominator = 2
5
6    def calculate(self):
7      localvariable = 10
8      print(self.classvariable + localvariable / denominator)
9
10 if __name__ == '__main__':
11   myclass = MyClass()
12   myclass.calculate()
13 ...
14 NameError: global name 'denominator' is not defined
```

Scopes

```
1  class MyClass():
2    def __init__(self):
3      self.classvariable = 10
4      self.denominator = 2
5
6    def calculate(self):
7      localvariable = 10
8      print(self.classvariable + localvariable / self.denominator)
9
10 if __name__ == '__main__':
11   myclass = MyClass()
12   myclass.calculate()
13 ...
14 15
```

Skriva över klassvariabler och metoder

- Inget inbyggd skydd finns för att förhindra att man skriver över en klassvariabel eller metod av misstag.

Exempel

```
1  class MyClass():
2    def __init__(self):
3      self.classvariable = 10
4      self.denominator = 2
5
6    def calculate(self):
7      localvariable = 10
8      return(self.classvariable + localvariable / self.denominator)
9
10 if __name__ == '__main__':
11   myclass = MyClass()
12   print(myclass.calculate())
13   myclass.calculate = 1
14   print(myclass.calculate)
15 ...
16 15
17 1
```

Referenser I

[1] *The Python Tutorial*. 2016. URL: https://docs.python.org/3/tutorial/index.html.

Mittuniversitetet
MID SWEDEN UNIVERSITY