The Concept of Polymorphism in Object Oriented Programing

With illustration in Python 2.7

Presented by

Michael S. James

Tuesday, November 14, 2017 @ 12:00 PM

DEFINITION:

“Generally, the ability to appear in many forms. In object-oriented programming, polymorphism refers to a programming language's ability to process objects differently depending on their data type or class.”

Source: Vangie Bael @ https://www.webopedia.com/TERM/P/polymorphism.html

DEFINITION (cont.):

“More specifically, it is the ability to redefine methods for derived classes.”

Source: Vangie Bael @ https://www.webopedia.com/TERM/P/polymorphism.html

DEFINITION (cont.):

“For example, given a base class *Animal*, polymorphism enables the programmer to define different *animal behavior* methods for any number of derived classes, such as *dogs*, *cats* and *cows*.”

Source: Vangie Bael @ https://www.webopedia.com/TERM/P/polymorphism.html

DEFINITION (cont.):

No matter what shape an object is, applying the area method to it will return the correct results. Polymorphism is considered to be a requirement of any true object-oriented programming language (OOPL).

Source: Vangie Bael @ https://www.webopedia.com/TERM/P/polymorphism.html

CODE EXAMPLE: create base class *Animal*

```
# Create Animal() superclass object.
class Animal(object):

    # Class Object Attributes
    species = 'mammal'
    bloodtype = 'warm blooded'

    def __init__(self):
        pass

    def whoAmi(self):
        print "Animal"

    def eat(self):
        print "Eating"
```

Source: Michael S. James
@ https://github.com/msjames19702/oop-polymorphism
@ https://anaconda.org/msjames19702/oop-example-of-polymorphism-in-python-2-7/notebook

CODE EXAMPLE: create derived class *Dog*

```
# Create Dog() subclass object and functions. Make use of Inheritance.

class Dog(Animal):
    def __init__(self):
        Animal.__init__(self)
        print "Dog subclass instance created"

    def whoAmi(self):
        print "I'm a Dog"

    def sound(self):
        print "I go Woof!"

    def whatIdo(self):
        print "I play fetch the ball!"
```

Source: Michael S. James
@ https://github.com/msjames19702/oop-polymorphism
@ https://anaconda.org/msjames19702/oop-example-of-polymorphism-in-python-2-7/notebook

CODE EXAMPLE: create derived class *Cat*

```
# Create Cat() subclass object and functions. Make use of Inheritance.

class Cat(Animal):
    def __init__(self):
        Animal.__init__(self)
        print "Cat subclass instance created"

    def whoAmi(self):
        print "I'm a Cat"

    def sound(self):
        print "I go Meow!"

    def whatIdo(self):
        print "I play with yarn!"
```

Source: Michael S. James
@ https://github.com/msjames19702/oop-polymorphism
@ https://anaconda.org/msjames19702/oop-example-of-polymorphism-in-python-2-7/notebook

CODE EXAMPLE: create derived class *Cow*

```
# Create Cow() subclass object and functions. Make use of Inheritance.

class Cow(Animal):
    def __init__(self):
        Animal.__init__(self)
        print "Cow subclass instance created"

    def whoAmi(self):
        print "I'm a Cow"

    def sound(self):
        print "I go Mmmmooooooo!"

    def whatIdo(self):
        print "I just like to eat hay!"
```

Source: Michael S. James
@ https://github.com/msjames19702/oop-polymorphism
@ https://anaconda.org/msjames19702/oop-example-of-polymorphism-in-python-2-7/notebook

CODE EXAMPLE: create object instances and polymorphic functions

```
# Create object instances Animal(), Dog(), Cat() and Cow()

animalObject = Animal()
dogObject = Dog()
catObject = Cat()
cowObject = Cow()

# Functional polymorphisms

def whatAmi(animaltype):
    animaltype.whoAmi()

def soundImake(animalsound):
    animalsound.sound()

def iDo(animaldoes):
    animaldoes.whatIdo()
```

Source: Michael S. James
@ https://github.com/msjames19702/oop-polymorphism
@ https://anaconda.org/msjames19702/oop-example-of-polymorphism-in-python-2-7/notebook

CODE EXAMPLE: *dogObject* and command line results

```
# Calling objects as parameters to make use of polymorphism

whatAmi(dogObject)
soundImake(dogObject)
iDo(dogObject)
```

```
Python 2.7.12 |Anaconda 4.2.0 (x86_64)| (default, Jul  2 2016, 17:43:17)
[GCC 4.2.1 (Based on Apple Inc. build 5658) (LLVM build 2336.11.00)] on darwin

>>> whatAmi(dogObject)
I'm a Dog
>>> soundImake(dogObject)
I go Woof!
>>> iDo(dogObject)
I play fetch the ball!
>>>
```

Source: Michael S. James
@ https://github.com/msjames19702/oop-polymorphism
@ https://anaconda.org/msjames19702/oop-example-of-polymorphism-in-python-2-7/notebook

CODE EXAMPLE: *catObject* and command line results

```
# Calling objects as parameters to make use of polymorphism

whatAmi(catObject)
soundImake(catObject)
iDo(catObject)
```

```
Python 2.7.12 |Anaconda 4.2.0 (x86_64)| (default, Jul  2 2016, 17:43:17)
[GCC 4.2.1 (Based on Apple Inc. build 5658) (LLVM build 2336.11.00)] on darwin

>>> whatAmi(catObject)
I'm a Cat
>>> soundImake(catObject)
I go Meow!
>>> iDo(catObject)
I play with yarn!
>>>
```

Source: Michael S. James
@ https://github.com/msjames19702/oop-polymorphism
@ https://anaconda.org/msjames19702/oop-example-of-polymorphism-in-python-2-7/notebook

CODE EXAMPLE: *cowObject* and command line results

```
# Calling objects as parameters to make use of polymorphism

whatAmi(cowObject)
soundImake(cowObject)
iDo(cowObject)
```

```
Python 2.7.12 |Anaconda 4.2.0 (x86_64)| (default, Jul  2 2016, 17:43:17)
[GCC 4.2.1 (Based on Apple Inc. build 5658) (LLVM build 2336.11.00)] on darwin

>>> whatAmi(cowObject)
I'm a Cow
>>> soundImake(cowObject)
I go Mmmmoooooo!
>>> iDo(cowObject)
I just like to eat hay!
>>>
```

Source: Michael S. James
@ https://github.com/msjames19702/oop-polymorphism
@ https://anaconda.org/msjames19702/oop-example-of-polymorphism-in-python-2-7/notebook

Thank you !!!

Presented by

Michael S. James

Tuesday, November 14, 2017 @ 12:00 PM