[320] Object Oriented Programming

Tyler Caraza-Harter

Creating New Types

Classes and Other Types

Objects

https://www.macys.com/shop/product/martha-stewart-collection-set-of-6-cookie-cutters-created-for-macys?ID=5467270

Class Attributes

create some objects
of type **dict** for **movies**

```
m1 = {...}
m2 = {...}
```

create some objects
of type **dict** for **people**

```
p1 = {}
p2 = {}
p3 = dict()
```

set some keys/values

```
p1["Fname"] = "Joseph"
p2["fname"] = "Peyman"
p3["fname"] = "Shri Shruthi"

print(type(m1))
print(type(p1))
```

```python
class Person:
    pass

p1 = Person()
p2 = Person()
p3 = Person()

p1.Fname = "Joseph"
p2.fname = "Peyman"
p3.fname = "Shri Shruthi"

print(type(p3))
```

create a Person type/class

create some objects of type Person

set some attributes

Objects created from classes are mutable.
Attribute names are not fixed at creation.

Attribute Names/Values are like Keys/Values

Using Dict	Using `class Point:` `pass`	
`d = dict()`	`p = Point()`	`p = Point()`

`d["x"] = 3`
`d["y"] = 4`
	`setattr(p, "x", 3)`
	`setattr(p, "y", 4)`
		`p.x = 3`
		`p.y = 4`

`tot = d["x"] + d["y"]`	`tot = (getattr(p, "x")`
	`+getattr(p, "y"))`
		`tot = p.x + p.y`

`has_z = "z" in d`	`has_z = hasattr(p, "z")`	`# no equivalent`

Attribute Names/Values are like Keys/Values

Using Dict	Using `class Point:` `    pass`	
`d = dict()`	`p = Point()`	`p = Point()`

`d["x"] = 3`
`d["y"] = 4`
	`setattr(p, "x", 3)`
	`setattr(p, "y", 4)`
		`p.x = 3`
		`p.y = 4`

`tot = d["x"] + d["y"]`	`tot = (getattr(p, "x")`
	`       +getattr(p, "y"))`
		`tot = p.x + p.y`

`has_z = "z" in d`	`has_z = hasattr(p, "z")`	`# no equivalent`
	avoid this	preferred

only use attribute names that could also be variables names

Attribute Names/Values are like Keys/Values

Using Dict	Using `class Point:` `pass`	
`d = dict()`	`p = Point()`	`p = Point()`
`d["x"] = 3`	`setattr(p, "x", 3)`	`p.x = 3`
`d["y"] = 4`	`setattr(p, "y", 4)`	`p.y = 4`
`tot = d["x"] + d["y"]`	`tot = (getattr(p, "x")`	`tot = p.x + p.y`
	`+getattr(p, "y"))`	
`has_z = "z" in d`	`has_z = hasattr(p, "z")`	`# no equivalent`
	avoid this	preferred

only use attribute names that could also be variables names

Attribute Names/Values are like Keys/Values

Using Dict	Using `class Point:` `pass`	
`d = dict()`	`p = Point()`	`p = Point()`
`d["x"] = 3` `d["y"] = 4`	`setattr(p, "x", 3)` `setattr(p, "y", 4)`	`p.x = 3` `p.y = 4`
`tot = d["x"] + d["y"]`	`tot = (getattr(p, "x")` `+getattr(p, "y"))`	`tot = p.x + p.y`
`has_z = "z" in d`	`has_z = hasattr(p, "z")`	`# no equivalent`
	avoid this	preferred

only use attribute names that could also be variables names

Coding Examples: Animal Classes

Principals

- methods
- checking object type
- type-based dispatch
- self
- constructors