

The background is a complex, abstract composition. The left side features a dark blue/black area with a stylized blue bird-like shape at the top, followed by horizontal bands of blue and yellow with various patterns like wavy lines and circles. The right side is dominated by a large, colorful geometric pattern of triangles and polygons in shades of orange, blue, and grey, creating a 3D effect. Faint, white, dashed lines swirl across the entire background.

# Object- Oriented Programming In Python

+  
By Shane Killen

# What is Object-Oriented Programming

- +Classes
- +Objects
- +Blueprint
- +Attributes
- +Functions(methods)

```
3 class Car:
4     def __init__(self, type, color, engine, doors):
5         self.type = type
6         self.color = color
7         self.engine = engine
8         self.doors = doors
9
10    def display_car(self):
11        print("Type: ", self.type, " Color: ", self.color, " Engine: ",
12              self.engine, " Doors: ", self.doors)
13
14    car = Car("truck", "white", "v8", "2")
15    car2 = Car("sedan", "red", "I4", "4")
16    car.display_car()
17    car2.display_car()
```

# What is an object

+ Everything

```
car = Car(car_type, car_color, car_engine, car_doors)
car.display_car()
```

```
input("This is an object -->").lower()
```

```
list = [1, 2, 3, 4]
```

```
list.append(5)
```

```
list.pop(2)
```

```
list.sort()
```

```
x = 5
```

```
y = 10
```

```
string = "String"
```

```
num = 10.25
```

# Benefit of OOP

```
3 class Car:
4     def __init__(self, type, color, engine, doors):
5         self.type = type
6         self.color = color
7         self.engine = engine
8         self.doors = doors
9
10    def display_car(self):
11        print("Type: ", self.type, " Color: ", self.color, " Engine: ",
12              self.engine, " Doors: ", self.doors)
13
14    num_car = int(input("How many cars would you like to make"))
15    |
16    for car in range(num_car):
17        car_type = input("enter the type of car: ")
18        car_color = input("enter the color of the car: ")
19        car_engine = input("enter the engine the car has: ")
20        car_doors = input("enter how many doors the car has: ")
21        car = Car(car_type, car_color, car_engine, car_doors)
22        car.display_car()
23
```

```
1 car_doors = 4
2 car_color = "red"
3 car_engine = "I4"
4 car_wheels = "winter"
5 car_rims = "17inch"
6 car_type = "sedan"
7
8 car_type2 = "truck"
9 car_color2 = "white"
10 car_engine2 = "v8"
11 car_wheels2 = "Off-road"
12 car_rims2 = "19inch"
13 car_doors2 = 2
14
15 print(car_type)
16 print(car_doors)
17 print(car_color)
18 print(car_engine)
19 print(car_wheels)
20 print(car_rims)
21
22 print(car_type2)
23 print(car_doors2)
24 print(car_color2)
25 print(car_engine2)
26 print(car_wheels2)
27 print(car_rims2)
```

# Classes

Def `__init__(self, x, y, z):`  
    `Self.x = x`  
    ...

What is

- `__init__`
- `Self.x`
- `(self)`
- Setters and getters

```
3 class Car:
4     def __init__(self, type, color, engine, doors):
5         self._type = type
6         self._color = color
7         self._engine = engine
8         self._doors = doors
9
10    def get_type(self):
11        return self._type
12
13    def get_color(self):
14        return self._color
15
16    def get_engine(self):
17        return self._engine
18
19    def get_doors(self):
20        return self._doors
21
22    def set_type(self, type):
23        self._type = type
24
25    def set_color(self, color):
26        self._color = color
27
28    def set_engine(self, engine):
29        self._engine = engine
30
31    def set_doors(self, doors):
32        self._doors = doors
33
34    def display_car(self):
35        print("Type: ", self._type, " Color: ", self._color, " Engine: ",
36              self._engine, " Doors: ", self._doors)
37
38
```

```
car = Car("Truck", "Red", "V8", "4")
car.set_color("Blue")
car.set_engine("V6")
print("The truck got repainted to: ", car.get_color())
print("The trucks engine is a ", car.get_engine())
car.display_car()
```

```
The truck got repainted to: Blue
The trucks engine is a V6
Type: Truck Color: Blue Engine: V6 Doors: 4
```



# Your Turn

- + Create a pizza class
- + Ask the user for
  - + Number of pizzas they want
  - + The size of the pizza
  - + Topping of the pizza



# Next Step in OOP

- + Class Libraries
- + Special Methods
- + Super class
- + Inheritance
- + Polymorphism

<code>__str__</code>	<code>__nonzero__</code>	<code>__getitem__</code>	<code>__add__</code>	<code>__neg__</code>
<code>__repr__</code>	<code>__getattr__</code>	<code>__setitem__</code>	<code>__sub__</code>	<code>__abs__</code>
<code>__lt__</code>	<code>__setattr__</code>	<code>__delitem__</code>	<code>__mul__</code>	<code>__int__</code>
<code>__le__</code>	<code>__delattr__</code>	<code>__reversed__</code>	<code>__floordiv__</code>	<code>__long__</code>
<code>__eq__</code>	<code>__get__</code>	<code>__contains__</code>	<code>__mod__</code>	<code>__float__</code>
<code>__ne__</code>	<code>__set__</code>	<code>__setslice__</code>	<code>__divmod__</code>	<code>__oct__</code>
<code>__gt__</code>	<code>__delete__</code>	<code>__delslice__</code>	<code>__pow__</code>	<code>__hex__</code>
<code>__ge__</code>	<code>__call__</code>	<code>__lshift__</code>	<code>__and__</code>	<code>__index__</code>
<code>__cmp__</code>	<code>__len__</code>	<code>__rshift__</code>	<code>__xor__</code>	<code>__enter__</code>
<code>__hash__</code>	<code>__iter__</code>	<code>__div__</code>	<code>__or__</code>	<code>__exit__</code>

