# SPYRAI

Smart Performance Yielding
Rapid Insights with AI

# HATE SPENDING TIME WATCHING LONG TUTORIAL VIDEOS?

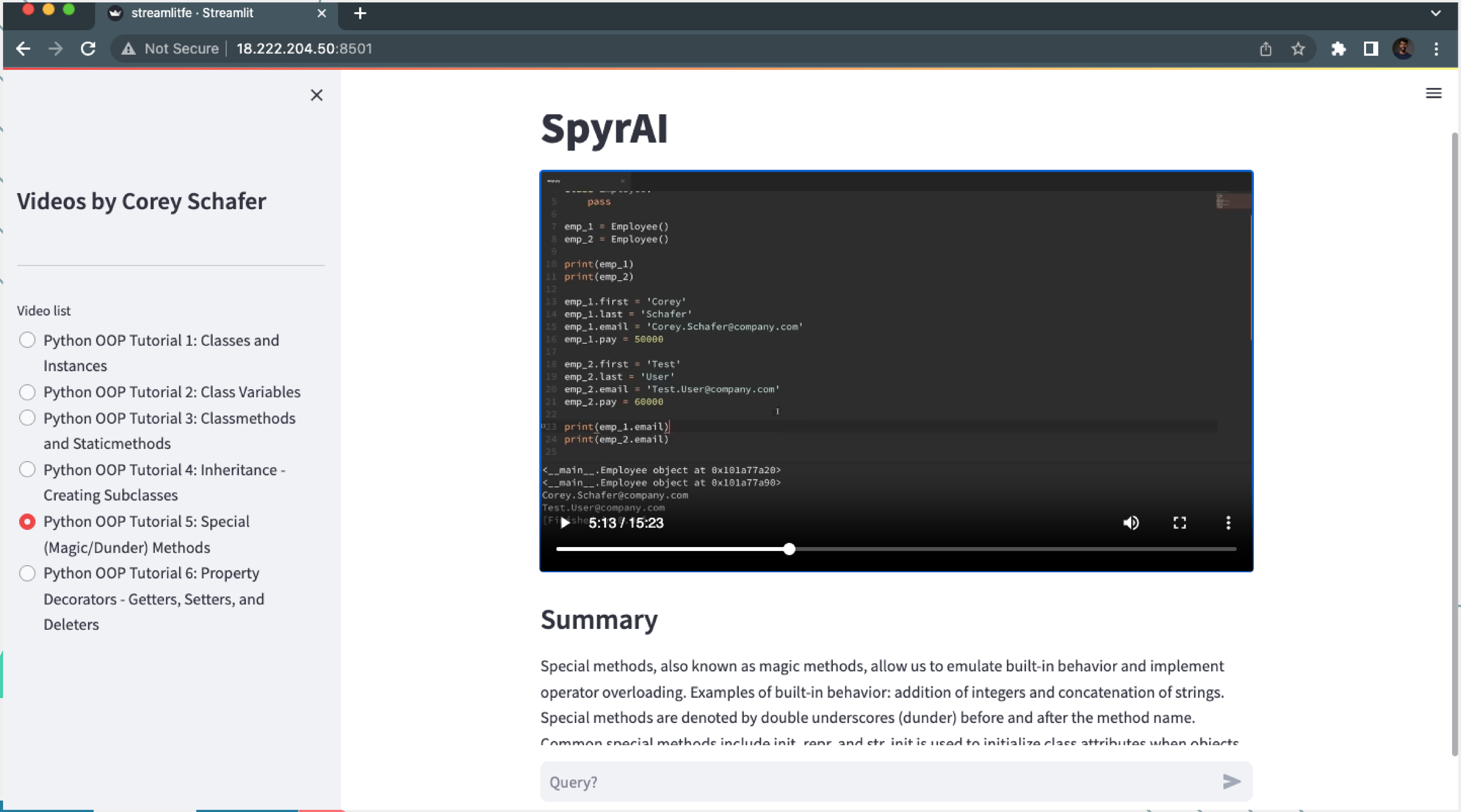
# BUT FEAR MISSING OUT ON LEARNING?

# WE GOT YOU !
# CHAT WITH YOUR VIDEOS
# ON
# SPYRAI

# WE GOT YOU !
# CHAT WITH YOUR VIDEOS
# ON
# SPYRAI

# POWERED BY

Streamlit
LangChain



# SpyrAI

```
    pass

emp_1 = Employee()
emp_2 = Employee()

print(emp_1)
print(emp_2)

emp_1.first = 'Corey'
emp_1.last = 'Schafer'
emp_1.email = 'Corey.Schafer@company.com'
emp_1.pay = 50000

emp_2.first = 'Test'
emp_2.last = 'User'
emp_2.email = 'Test.User@company.com'
emp_2.pay = 60000

print(emp_1.email)
print(emp_2.email)
```

```
<__main__.Employee object at 0x101a77a20>
<__main__.Employee object at 0x101a77a90>
Corey.Schafer@company.com
Test.User@company.com
```

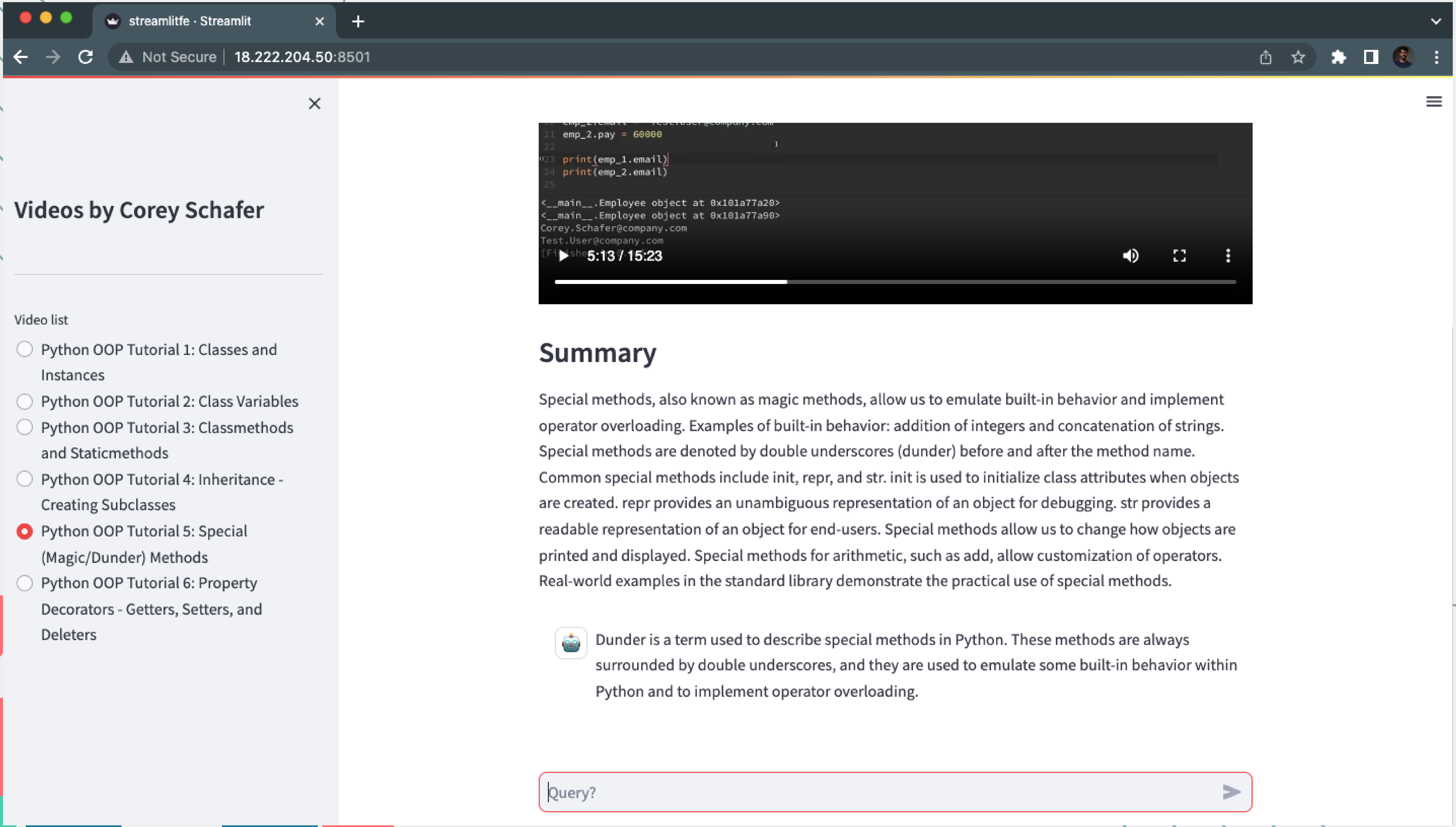
## Summary

Special methods, also known as magic methods, allow us to emulate built-in behavior and implement operator overloading. Examples of built-in behavior: addition of integers and concatenation of strings. Special methods are denoted by double underscores (dunder) before and after the method name. Common special methods include init, repr, and str. init is used to initialize class attributes when objects

## Videos by Corey Schafer

Video list

- ○ Python OOP Tutorial 1: Classes and Instances
- ○ Python OOP Tutorial 2: Class Variables
- ○ Python OOP Tutorial 3: Classmethods and Staticmethods
- ○ Python OOP Tutorial 4: Inheritance - Creating Subclasses
- ● Python OOP Tutorial 5: Special (Magic/Dunder) Methods
- ○ Python OOP Tutorial 6: Property Decorators - Getters, Setters, and Deleters



## Videos by Corey Schafer

Video list

- ○ Python OOP Tutorial 1: Classes and Instances
- ○ Python OOP Tutorial 2: Class Variables
- ○ Python OOP Tutorial 3: Classmethods and Staticmethods
- ○ Python OOP Tutorial 4: Inheritance - Creating Subclasses
- ● Python OOP Tutorial 5: Special (Magic/Dunder) Methods
- ○ Python OOP Tutorial 6: Property Decorators - Getters, Setters, and Deleters

```
emp_2.pay = 60000

print(emp_1.email)
print(emp_2.email)

<__main__.Employee object at 0x101a77a20>
<__main__.Employee object at 0x101a77a90>
Corey.Schafer@company.com
Test.User@company.com
```

## Summary

Special methods, also known as magic methods, allow us to emulate built-in behavior and implement operator overloading. Examples of built-in behavior: addition of integers and concatenation of strings. Special methods are denoted by double underscores (dunder) before and after the method name. Common special methods include init, repr, and str. init is used to initialize class attributes when objects are created. repr provides an unambiguous representation of an object for debugging. str provides a readable representation of an object for end-users. Special methods allow us to change how objects are printed and displayed. Special methods for arithmetic, such as add, allow customization of operators. Real-world examples in the standard library demonstrate the practical use of special methods.

Dunder is a term used to describe special methods in Python. These methods are always surrounded by double underscores, and they are used to emulate some built-in behavior within Python and to implement operator overloading.

# TEAM

Rushang Shah

Shaunak Galvankar

# THANK YOU

www.reallygreatsite.com