Python Fundamentals day 54

Today’s Agenda

- Extending built-in classes

Extending built-in classes

We can extend all of Python's built-in classes. This allows us to add or modify features of the data types that come with Python. This may save us from having to build a program from scratch. Let us take an example and see how to achieve it

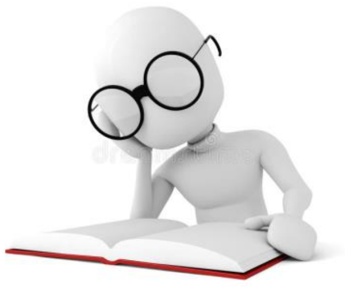
```
class Contact:
    all_contacts=[]

    def __init__(self,name,email):
        self.name=name
        self.email=email
        Contact.all_contacts.append(self)

    def display(self):
        print(self.__dict__)

def main():
    c1=Contact('Rohit','rohit@gmail.com')
    c2=Contact('Manish','mainsh@gmail.com')
    print(Contact.all_contacts)

if __name__=='__main__':
    main()
```



Output:

```
In [2]: runfile('C:/Users/rooman/OneDrive/Desktop/python/
test.py', wdir='C:/Users/rooman/OneDrive/Desktop/python')
[<__main__.Contact object at 0x000001AFF32E1088>,
<__main__.Contact object at 0x000001AFF327D3C8>]
```

We have list of two objects. Let us try to print the instances within those objects

```python
class Contact:
    all_contacts=[]

    def __init__(self,name,email):
        self.name=name
        self.email=email
        Contact.all_contacts.append(self)

    def display(self):
        print(self.__dict__)

def main():
    c1=Contact('Rohit','rohit@gmail.com')
    c2=Contact('Manish','mainsh@gmail.com')
    for i in Contact.all_contacts:
        i.display()

if __name__=='__main__':
    main()
```

Output:

```
In [3]: runfile('C:/Users/rooman/OneDrive/Desktop/python/
test.py', wdir='C:/Users/rooman/OneDrive/Desktop/python')
{'name': 'Rohit', 'email': 'rohit@gmail.com'}
{'name': 'Manish', 'email': 'mainsh@gmail.com'}
```

It would be a lot easier if we could get the above output by calling a function as shown below in the comment section.

```python
class Contact:
    all_contacts=[]

    def __init__(self,name,email):
        self.name=name
        self.email=email
        Contact.all_contacts.append(self)

    def display(self):
        print(self.__dict__)

def main():
    c1=Contact('Rohit','rohit@gmail.com')
    c2=Contact('Manish','mainsh@gmail.com')
    for i in Contact.all_contacts:
        i.display()
    #Contact.all_contacts.display_all_contacts()

    name='Rohit'
    for i in Contact.all_contacts:
        if i.name==name:
            print('contact found')
    #Contact.all_contacts.search_contact('Rohit')

if __name__=='__main__':
    main()
```

Output:

```
In [5]: runfile('C:/Users/rooman/OneDrive/Desktop/python/
test.py', wdir='C:/Users/rooman/OneDrive/Desktop/python')
{'name': 'Rohit', 'email': 'rohit@gmail.com'}
{'name': 'Manish', 'email': 'mainsh@gmail.com'}
contact found
```

Let us now see how to extend the features of built-in data types using class in the below example

```python
class ContactList(list):
    def display_all_contacts(self):
        for i in self:
            i.display()

    def search_contact(self,name):
        for i in self:
            if i.name==name:
                return 'Contact found'
            return 'Contact not found'

class Contact:
    all_contacts=ContactList()

    def __init__(self,name,email):
        self.name=name
        self.email=email
        Contact.all_contacts.append(self)

    def display(self):
        print(self.__dict__)

def main():
    c1=Contact('Rohit','rohit@gmail.com')
    c2=Contact('Manish','mainsh@gmail.com')

    Contact.all_contacts.display_all_contacts()

    print(Contact.all_contacts.search_contact('Rohit'))

if __name__=='__main__':
    main()
```

Output:

```
In [8]: runfile('C:/Users/rooman/OneDrive/Desktop/python/
test.py', wdir='C:/Users/rooman/OneDrive/Desktop/python')
{'name': 'Rohit', 'email': 'rohit@gmail.com'}
{'name': 'Manish', 'email': 'mainsh@gmail.com'}
Contact found
```

All we did is, instead of directly creating a list we created an object of the ContactList which is a list on which we can activate the extended features display_all_contacts and search_contacts which a traditional list did not have.