Usman Institute of Technology

Department of Computer Science Fall 2022

Name: Muhammad Waleed

Roll no: 20B-115-SE

Course: Software Design and Architecture (SE-308)

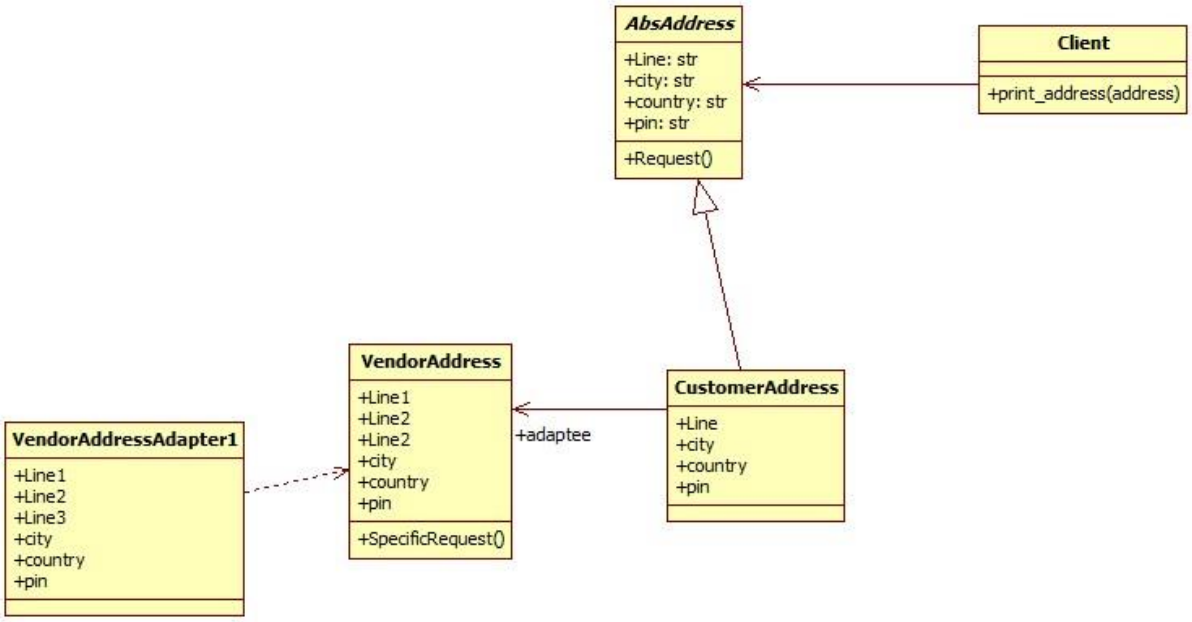
Course Instructor: Misbah ud Din

Date: 24-Nov-2022

Muhammad Waleed
20b-115-se
Lab#07
Sir Misbah ud Deen

Lab Tasks:

AddressAdapter:



Code:

```
from abc import ABC

class AbsAddress(ABC):
    line: str
    city: str
    country: str
    pin: str

class VendorAddress:
    def __init__(self, line1, line2, line3, city, country, pin):
        self.line1 = line1
        self.line2 = line2
        self.line3 = line3
        self.city = city
        self.country = country
        self.pin = pin

class CustomerAddress(AbsAddress):
    def __init__(self, line, city, country, pin):
        self.line = line
```

Muhammad Waleed
20b-115-se
Lab#07
Sir Misbah ud Deen

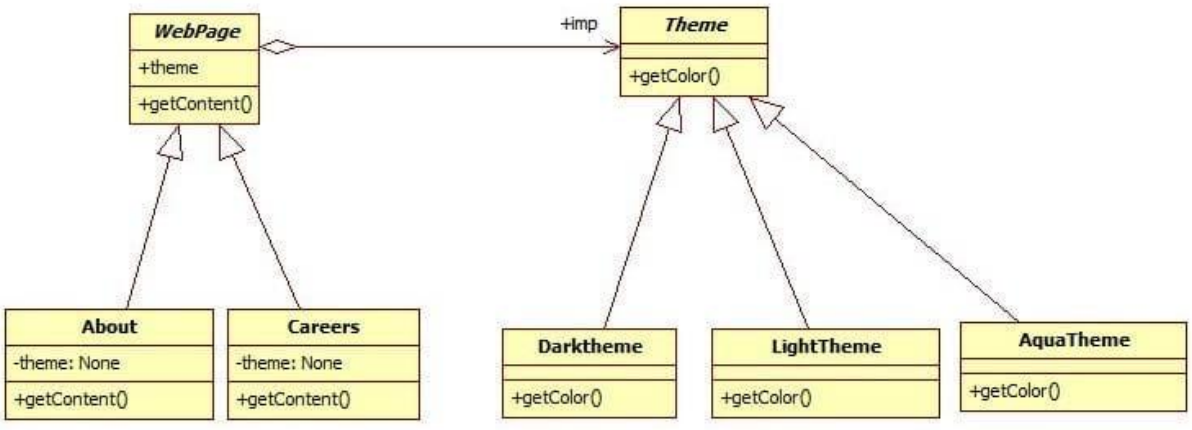
```
        self.city = city
        self.country = country
        self.pin = pin

class VendorAddressAdapter:
    def __init__(self, vendor_address):
        self.line = f'{vendor_address.line1}, {vendor_address.line2},
{vendor_address.line3}'
        self.city = vendor_address.city

        self.country = vendor_address.country
        self.pin = vendor_address.pin


# client
def print_address(address):
    print(f'{address.line}, {address.city}, {address.country}, {address.pin}')


if __name__ == '__main__':
    cust_address = CustomerAddress("House No", "A. B C Road", "Karachi",
74550)
    vend_address = VendorAddress("Home # 1", "Apartment 1", "Street 4", "A. B C
Road", "karachi", 45700)
    vend_address_adapt = VendorAddressAdapter(vend_address)

    for address in [cust_address, vend_address_adapt]:
        print_address(address)
```

Output:

```
House No, A. B C Road, Karachi, 74550
Home # 1, Apartment 1, Street 4, A. B C Road, karachi, 45700
```

Muhammad Waleed
20b-115-se
Lab#07
Sir Misbah ud Deen

Web Page Bridge:



Code:

```
class WebPage:
    def __init__(self, theme):
        self.theme = theme

    def getContent(self):
        pass


class About(WebPage):
    _theme = None

    def __init__(self, theme):
        self.theme = theme

    def getContent(self):
        return "About page in " + self.theme.getColor()


class Careers(WebPage):
    _theme = None

    def __init__(self, theme):
        self.theme = theme

    def getContent(self):
        return "Careers page in " + self.theme.getColor()
```

Muhammad Waleed
20b-115-se
Lab#07
Sir Misbah ud Deen

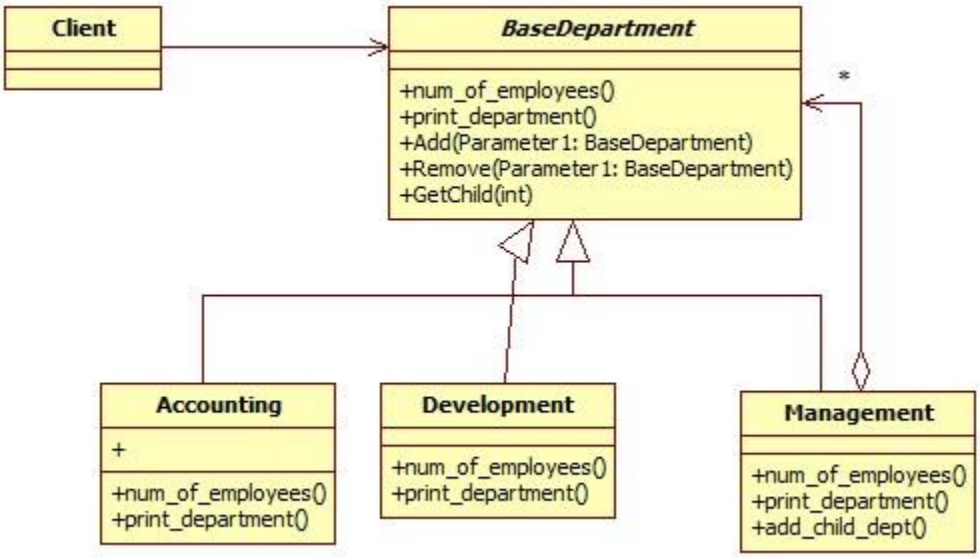
```
class Theme:
    def getColor(self):
        pass


class DarkTheme(Theme):
    def getColor(self):
        return 'Dark Black'


class LightTheme(Theme):
    def getColor(self):
        return 'Off White'


class AquaTheme(Theme):
    def getColor(self):
        return 'Light Blue'


if __name__ == '__main__':

    darkTheme = DarkTheme()
    lightTheme = LightTheme()
    about = About(darkTheme)
    careers = Careers(darkTheme)
    aboutLight = About(lightTheme)
    careersLight = Careers(lightTheme)

    print(about.getContent())
    print(careers.getContent())
    print(aboutLight.getContent())
    print(careersLight.getContent())
```

Output:

```
About page in Dark Black
Careers page in Dark Black
About page in Off White
Careers page in Off White
```

Muhammad Waleed
20b-115-se
Lab#07
Sir Misbah ud Deen

Department Composite:



Code:

```
from abc import ABC, abstractmethod

class BaseDepartment(ABC):
    @abstractmethod
    def __init__(self, num_of_employees):
        pass

    @abstractmethod
    def print_department(self):
        pass

class Accounting(BaseDepartment):
    def __init__(self, num_of_employees):
        self.num_of_employees = num_of_employees

    def print_department(self):
        print(f"Accounting employees: {self.num_of_employees}")

class Development(BaseDepartment):
    def __init__(self, num_of_employees):
        self.num_of_employees = num_of_employees

    def print_department(self):
```

Muhammad Waleed
20b-115-se
Lab#07
Sir Misbah ud Deen

```
        print(f"Development employees: {self.num_of_employees}")

class Management(BaseDepartment):
    def __init__(self, num_of_employees):
        self.num_of_employees = num_of_employees
        self.childs = []

    def print_department(self):
        print(f"Management base employees: {self.num_of_employees}")
        total_emp_count = self.num_of_employees
        for child in self.childs:
            total_emp_count += child.num_of_employees
            child.print_department()
            print(f'Total employees: {total_emp_count}')

    def add_child_dept(self, dept):
        self.childs.append(dept)
    def delete_child_dept(self,dept):
        self.child.pop(dept)
#
acc_dept = Accounting(200)
dev_dept = Development(500)

management_dept = Management(50)
management_dept.add_child_dept(acc_dept)
management_dept.add_child_dept(dev_dept)

# print dept
management_dept.print_department()
```

Output:

```
Management base employees: 50
Accounting employees: 200
Total employees: 250
Development employees: 500
Total employees: 750
```

Muhammad Waleed
20b-115-se
Lab#07
Sir Misbah ud Deen

Home Tasks:

Socket Adapter:

```
class Socket:
    def __init__(self):
        self.voltage = 220
        self.type = '2Pins'
    def get_voltage(self):
        return self.voltage
    def get_type(self):
        return self.type

class Charger:
    def __init__(self):
        self.voltage = 220
        self.type = '3Pins'
    def get_voltage(self):
        return self.voltage
    def get_type(self):
        return self.type

class Adapter:
    def __init__(self, socket):
        self.socket = socket
    def get_voltage(self):
        return self.socket.get_voltage()
    def get_type(self):
        return self.socket.get_type()

class Device:
    def __init__(self, adapter):
        self.adapter = adapter
    def charge(self):
        if self.adapter.get_type() == '3Pins':
            print('Charging...')
        else:
            print('Cannot charge. Adapter type is not compatible.')

if __name__ == '__main__':
    print('Socket Type: 2Pins')
    print('Socket Voltage: 220V')

    print('Mobile Charger Type: 3Pins')
```

Muhammad Waleed
20b-115-se
Lab#07
Sir Misbah ud Deen

```
    print('Mobile Charger Voltage: 220V')
    print('Mobile Socket Type: 3Pins')

    charger = Charger()
    adapter = Adapter(charger)
    print('Adapter Type: {}'.format(adapter.get_type()))
    device = Device(adapter)
    print('Plug in the adapter to the device...')
    device.charge()
```

Output:

```
Socket Type: 2Pins
Socket Voltage: 220V
Socket Voltage: 220V
Mobile Charger Type: 3Pins
Mobile Charger Voltage: 220V
Mobile Socket Type: 3Pins
Adapter Type: 3Pins
Plug in the adapter to the device...
Charging...
```

Muhammad Waleed
20b-115-se
Lab#07
Sir Misbah ud Deen

Encryption Bridge:

```
class Encryption:
    def encrypt(self):
        pass

class AES(Encryption):
    def encrypt(self):
        print('Encrypting with AES')

class Blowfish(Encryption):
    def encrypt(self):
        print('Encrypting with Blowfish')

class EncryptionBridge:
    def __init__(self, encryption):
        self.encryption = encryption
    def encrypt(self):
        self.encryption.encrypt()

if __name__ == '__main__':
    aes = AES()
    blowfish = Blowfish()
    aesBridge = EncryptionBridge(aes)
    blowfishBridge = EncryptionBridge(blowfish)
    aesBridge.encrypt()
    blowfishBridge.encrypt()
```

Output:

```
Encrypting with AES
Encrypting with Blowfish
```

PCComposite:

```
# composite design pattern example

class PC:
    def __init__(self, name):
        self.name = name
        self.parts = []
    def add(self, part):
        self.parts.append(part)
```

Muhammad Waleed
20b-115-se
Lab#07
Sir Misbah ud Deen

```
    def remove(self, part):
        self.parts.remove(part)
    def get_price(self):
        price = 0
        for part in self.parts:
            price += part.get_price()
        return price
    def get_name(self):
        return self.name

class Part:
    def __init__(self, name, price):
        self.name = name
        self.price = price
    def get_price(self):
        return self.price
    def get_name(self):
        return self.name

if __name__ == '__main__':
    pc = PC('PC')
    cpu = Part('CPU', 100)
    ram = Part('RAM', 50)
    pc.add(cpu)
    pc.add(ram)
    print('Total price of {} is {}'.format(pc.get_name(), pc.get_price()))
```

Output:

```
Total price of PC is 150
```