PRAGMATIC APPROACH CONCENTRATING ON OBJECT ORIENTED DESIGNING CONCEPTS IN ANY INNOVATIVE TECHNOLOGY

RPRAGNESHREDDY NANDYALA

****

UNDER GUIDANCE OF

ROBERT SCHAEFER

INDEX:

# 1. ABSTRACT 3

# 2 .INTRODUCTION 3-4

# 3. HISTORY 5

4. CLASS AND OBJECT 5-6

5. ABILITY TO COMPARE

INTERFACE AND IMPLEMENTATION 6-8

6. CONSTRUCTOR 8-13

# 7. ERROR HANDLING TECHNIQUES 13-15

# 8. PERIPHERALS OF AN OBJECT 15-23

# 9. DESIGNING A CLASS 23-25

# 10. CONCLUSIONS

# 11. REFERENCES 25-26

ABSTRACT:

Pragmatic approach concentrating on object oriented designing concepts in any innovative technology. To understand present innovative technologies one should thoroughly understand object oriented designing provide object oriented designing concepts with related examples still I felt difficulty to understand them thoroughly. So, I came up with tangible examples followed by simple code in different programming languages. The idea is to reinterpret the concepts and clear vague in understanding Object oriented designing concepts. The concepts cannot be passed through a compiler to check how they work.

So, programming languages such as Java, object c, .NET help to understand this concepts to a grate extend.

INTRODUCTION:

In order to understand UML (unified modeling language) one should know Object Oriented Design.

I will provide an example to the above statement.

Let me consider an electric circuit design without knowing electricity one cannot understand the electric circuit.

Let me consider a central processing unit in a computer. If a person do not know basic idea of Computer he cannot analyze the components of central processing unit.

Example: RAM, ROM, Cache etc.

Present software systems rely most on object oriented design.

Example:

Programming in java, python, C#, C++ etc.

Beside Object oriented design other implementation of software programming was procedural programming

Example of Procedural Programming is C language.

A programmer should have a good knowledge on Object Oriented Design to have comprehensive programming. Otherwise judging the incomprehensive code is difficult to decide whether it is object oriented or a procedural programming.

HISTORY:

Object oriented design is present in 1960’s but Object Oriented designing concepts are used in technology since from mid1990’s. Although C++ is present.

Object oriented designing concepts used to develop new technologies.

Concepts are compared with evolution as they are same since mid1990’s till to day. Technology is drastically changing from primitive web browsing to hybrid apps (app can be used in both mobile and web).

So, the concepts are core part which are not changed from past 1990’s to till date.

DIFFERENCE BETWEEN PROCEDURAL PROGRAMMING AND OBJECT ORIENTED PROGRAMMING:

According to (Matt, 2013) Procedure programming sends data and operations separately.

Object oriented combine data or attributes or fields with operations or behaviors or methods or functions into an object.

Let me consider a network. In a network only data is transferred to the network and operations are present prior in the network.

An advantage For Object oriented data and operations are combined together and passed through a network.

Example: web browser with an object passed executes both data and behavior.

OBJECT:

To define an object I should consider a class because I feel difficult in explaining object without a class and it seems to be an unclear definition.

So, I can say that class is a design and implementation or creation or bringing up a sense to the design is an object.

Objects are considered building blocks of object oriented programming.

Object is a combination of both data field and method or behavior

Example: Let’s consider a Laptop.

It has data fields and behavior.

Data fields:

It has a color,

It has a brand,

No of keys.

Behavior.

CLASS**:**

A class is a template to build an object.

Let us consider a sample design of an engine. The design of an engine will be my class. Building up of engines or manufacturing of tangible engines is our object. So, clearly when it comes to whether class comes first or an object definitely a template or design comes first that is a class.

Programming example particularly in java:

Let me consider an example

class Engine {

Int force;

Int distance;

Public void action () {

Int Torque = force \* distance

}

Public static void main (String [ ] args)

{

Engine myEngine = new Engine();

}  
}

So, When I create an object for an engine those values will be stored in the engine object.

Primary point is to use this concept in object oriented programming language.

METHOD OR FUNCTION OR BEHAVIOR OR OPERATION OR SUBROUTINE:

What the object can do is considered as behavior. One of the most important advantage object oriented design is data is a part of package it is not separated from the code.

Without using all concepts in a programming language it can never be object oriented.

1. Encapsulation

2. Inheritance

3. Polymorphism

4. Composition

Before going deep into programming language.

Increasing speed, deleting cache previous data, increasing brightness, increase touch sensitivity etc.

ABILITY TO COMPARE INTERFACE AND IMPLEMENTATION:

In structured programming any sort of data is not hidden from the user. In object oriented we provide necessary information to the user and hide the unnecessary details.

EXAMPLE:

In order to charge a laptop I need to plug in the cable into the socket. So, this is a requirement

This should be definitely provided for the user. This is known as interface.

But, as a user I am not consider about the details of the electricity like whether it is coming from natural resource or a hydro power.

User really do not bother about the details as long as he gets his work done.

So, these are the details really not necessary for the user. All such details are hidden from the user known as implementation.

EXAMPLE PROGRAM:

Calculate different arithmetic operation:

Class Operations

{

public void calculations(int x, int y)

{

int z = x+y;

int m = x\*y;

int h = x/y;

system.out.println(z+” ”+m+” “+ h);

}

Class Results extends operations{

public static void main(String [] args){

Results simple = new Results()

Simple.calculations(2,3);

}

}

INTERFACE:

Suppose user wants to open a file. User can open up a file using a right clicking on the required file. So, clearly clicking the file is an interface provided to the user.

The user should be provided with an option to open a file. This is one of the important consideration for requirements.

The program beneath which provides user a required file is considered as Implementation.

Let me take an example of a News Paper as long as the user can see and read the letters on the newspaper he bothers whether it is readable or not.

He will not consider about type of ink and style of letters as long as it is readable.

Let’s considered database where the user requirements are need to

1. Open up a file

2. Look up a file

3. Open the next file after the one he looked.

Unified model of a class:

+openFile( int number );

+lookUpFile(int number );

+nextFile(int number );

+getTotalNumberOfRecords(int number);

+stillAnyRecordExists():boolean;

Form (Sebti Mouelhi, Khalid Agrou, Samir Chouali & Hassan Mountassir, 2015, p. 50). In interface automata Input actions are represent as public. Importance of making public input is explained above by taking different example.

CONSTRUCTOR:

Constructors are nowhere used in the structured programming. One with knowledge only in structured programming seems to find this as a new concept.

Example:

Languages with no constructors are: C and C++.

These is important to learn because this involves in implementation of object oriented design.

IDENTIFY CONSTRUCTOR IN ANY OBJECT ORIENTED PROGRAMMING LANGUAGE:

For java or c#: Constructor can be identified by comparing class name and method name.

If they both are same then we can say that a program has a constructor. Provided that method

Should never return anything not even void.

For Visual Basic .NET: NEW keyword represents a constructor.

For ObjectC: init

So, I can say although the syntax varies from one programming language to other. The thought of a construct remains same.

EXAMPLE OF A CONSTRUCTOR IN JAVA CODE:

public class Hello{

public Hello(){

system.out.println(" Constructor Invoked ");

}

public static void main(String [] args){

Hello myHello = new Hello();

}

Whenever I run this code by creating an object then it produces result as expected.

EXAMPLES OF A NON- CONSTRUCTOR IN JAVA CODE:

public class Hello {

public void Hello(){

system.out.println(" Constructor Invoked ");

}

public static void main(String [] args){

Hello myHello = new Hello();

}

Example ii:

public class Hello {

public int Hello(){

int x,

int y=2,

int z= 3;

return x = y + z;

}

public static void main(String [] args){

Hello myHello = new Hello();

}

The above two examples does not produce expected result when I create an object. As the methods doesn’t resemble a constructor.

CALLING A CONSTRUCTOR:

When an object is created for the class. Then the method gets called automatically which is my constructor.

Syntax:

Hello myObject = new Hello( );

Here new keyword allocates the memory for the object.

If I won’t create a constructor then the complier creates itself a default constructor.

To achieve this when the researcher broke the byte converted by the complier then a default

Constructor is included in the program.

When Hello class does not extends any class then it by default extends Object class calls the constructor of Object class.

Syntax: How does the program look although no constructor is provide after compilation?

public Hello()

{

super();

}

Object class is parent class for all class.

When I create a class with no constructor java compiler creates a default constructor.

Good practice of programming is always initialize your attributes by using a constructor.

A default constructor is only created when I do not create a constructor in the program. If I create one then complier will not create another constructor.

MULTIPLE CONSTRUCTORS:

Suppose if I want to initialize an attribute with two different numbers then I can achieve this by using a constructor.

public Hello()

{

numberOfTimesToSayHello = 0;

}

Here my class attribute

"numberOfTimesToSayHello" I am initializing it to '0' .

Now I can change the number of time to produce Hello method depending on client request using a constructor.

Public Hello (String anyMessage)

{

System.out,println(“ This is what you expected ” +anyMessage);

}

public Hello (int number )

{

numberOfTimesToSayHello = number;

}

public static void main(String [] args){

Hello myHello = new Hello (2);

Hello myHello1 = new Hello(“google”);

Hello myHello2 = new Hello();

}

The above Three constructors resembles method of over loading. Overloading is not just for constructors it works with normal methods.

I can over load a constructor as long as the method signature is different from one to other I can create as many as possible I can by differentiating a signature.

SIGNATURE OF A METHOD:

A signature is the aggregation of both method name and parameters.

EXAMPLE OF OVERLOADING CONSTRUCTOR:

public void Hello(int ProgrammingIsEasy)

public void Hello(String ProgrammingIsDifficult)

In Java and C# although I have a return type and void return for a same parameter then that is not

Considered as overloading.

EXAMPLE:

public void Hello(int programmingIsEasy )

public int Hello(int programmingIsEasy)

public class DataBaseReader {

String datab=BaseName;

int startingPosition;

public DataBaseReader (String name){

dataBaseName = name;

startingPosition = 0;

}

public DataBaseReader (String name, int pos){

dataBaseName = name;

startingPosition = pos;

}

}

IMPORTANCE OF CONSTRUCTOR:

Firstly, I can set values to appropriate number as I need. I shall start a file number by ‘1’ instead of default.

Secondly, Client cannot understand the class but by providing a constructor he known what values to be submitted and he can expect tangible outputs.

ERROR HANDLING TECHNIQUES:

1. Fix it

2. Ignore the problem by suppressing it

3. Exit the runtime in some graceful manner.

• Check for potential problems and abort the program when you find a problem.

• Check for potential problems, catch the mistake, and attempt to fix the problem.

• Throw an exception. (Often this is the preferred way to handle the situation).

Among the above three techniques I will go with the check the problem, catch the problem.

Finally, I am going to fix the problem.

class Hello{

public void Hello(int numericValue1, NumericValue2)

int result = numericValue1/numericValue2;

System.out.println(result);

public static void main(String [] args)

{

Hello MyPhone = new Hello(2,0);

}

}

Here I got an error because this value cannot be determined.

Whenever client enters a value that leads to an error then as a programmer I should look up a solution ahead of deployment and fix it.

So, Now I should Handle the exception by using try and catch block in java programming.

Try block is something where we can expect an error to occur. Catch block handle the error and provides the client tangible result instead of crashing the system.

class Hello{

public void Hello(int numericValue1, NumericValue2)

try{

int result = numericValue1/numericValue2;

}

catch (ArthematicException e){

Hello(int numericValue1, NumericValue2);

}

System.out.println(result);

public static void main(String [] args)

{

Hello MyPhone = new Hello(2,0);

}

}

This an Arithmetic Exception which occurs during runtime these a unchecked exception.

PERIPHERALS OF AN OBJECT:

LOCAL ATTRIBUTES:

If an attribute is present with in a method then we can say it as a local attribute.

Example:

public class Hello {

public void pickUp(){

int numberOfTimes = 1;

}

public void hold(){

}

public static void main(String [ ] args )

Hello MyPhone = new Hello();

MyPhone.pickUp();

}

Here the scope of the attribute is within the pickup method.

OBJECT ATTRIBUTES:

public class Hello {

int numberOfTimes;

public void pickUp(){

numberOfTimes = 1;

}

public void pressHold(){

numberOfTimes = 2;

}

Public static void main (String [ ] args)

Hello myPhone1 = new Hello();

Hello myPhone2 = new Hello();

MyPhone.pickUp();

}

The memory allocated for the attribute is shared between two methods. But, memory is not shared between the objects.

Creating new object allocate new memory for the two methods.

If my code has a class attribute and methods have their own.

public class Hello {

int numberOfTimes;

public void pickUp(){

int numberOfTimes = 1;

}

public void pressHold(){

int numberOfTimes = 2;

}

Public static void main(String [] args){

Hello myPhone1 = new Hello();

Hello myPhone2 = new Hello();

MyPhone.pickUp();

}

}

Three different memory locations are allocated here.

CLASS ATTRIBUTES:

public class Hello {

static int numberOfTimes;

public void pickUp(){

numberOfTimes = 1;

}

public void pressHold(){

numberOfTimes = 2;

}

Main Method:

Hello myPhone1 = new Hello();

Hello myPhone2 = new Hello();

MyPhone.pickUp();

}

Here objects share the same memory present in the class attribute. So, 'n' number of objects share a single memory created by using class attribute.

DRAWBACKS OF USING SINGLE CLASS MEMORY FOR DIFFERENT OBJECTS:

Let me take an example:

By using class attribute if I calculate number of contacts in my Phone for one method.

Then immediately if I use other method and calculate the number of apps in my phone then I am going to loss number of contacts.

So, From above I covered all the advanced OO topics.

EXAMPLE CODE IN C#:

using System;

namespace TestNumber

{

class TestHello

{

public static void Main()

{

Hello myPhone1 = new Hello();

Hello myPhone2 = new Hello();

Hello number3 = new Hello();

}

}

public class Hello

{

int numberOfTimes = 0;

public void pickUp(){

{

numberOfTimes = 1;

}

public void pressHold()

{

numberOfTimes = 2;

}

}

}

Primary focus should be on initial design of the class**.**

Here firstly we come up with interface depending on client requirement and then we move towards implementation.

DOCUMENTATION:

I provided an example of a documented java code one of the important pattern in designing a model.

/\*

This class defines Hello. Hello represents a mobile and assigns a phone call with other person.

\*/

public class Hello {

// write Software used here.

Private static String nameOfSoftware = "Android";

// name of the person I am speaking with

private mobile Mymobile;

static int numberOfTimes;

/\* This is a non-default constructor used to assign the Name of the person I am speaking with and represents software used in Hello class

public Hello(String name, String SoftwareUsed)

{

nameOfThePerson = name;

nameOfTheSoftware = SoftwareUsed;

}

// This method used to change the name of the person

public void setNameOfThePerson(String fullName)

{

nameOfThePerson = fullName;

}

// This method gives name of the software used

public static String getNameOfSoftware()

{

return nameOfTheSoftware;

}

//This method gives name of the person I am speaking to

public static String getName()

{

return nameOfThePerson;

}

public void pickUp(){

numberOfTimes = 1;

}

public void pressHold(){

numberOfTimes = 2;

}

public void MakeAcall{

}

private void EnterTenNumbers()

{

}

private void pressCallButton()

{

}

Main Method:

Hello myPhone1 = new Hello();

Hello myPhone2 = new Hello();

MyPhone.pickUp();

}

Comments are important in order to define a class. From (Murat Oruc, Fuat Akal & Hayri Sever, 2016, p. 115) Documentation is one of the important criteria while designing a model. The model is passed through DesPaD (Design Pattern Detector)

From the above Hello class assigns a conversation with other person.

Now I am briefly going to describe what each represents.

Attribute: Attribute stores the position of the object. Attribute has information about an object.

Here one of the attribute is name of the software. I mentioned it as private. That means it only accessed by the object with in the same class.

All the attributes are mentioned as private. The purpose of doing this makes interfaces minimum.

As I mentioned it as static. Although I create 50 object for the class. All the object uses only one memory location for the static attributes.

Private mobile MyMobile.

To explain this Mobile is a different class. Let suppose it has an attribute called Software used.

To insert a new Software. I have created a object of Mobile class in a Hello class. This is just a reference does not contain any memory. Since I have not initialized any value.

Now consider the above Constructor. I can initialize a new Software. By this syntax:

public Hello(String name, String SoftwareUsed)

{

nameOfThePerson = name;

MyMobile = new Mobile("androidN");

}

So, clearly I have allocated memory by above syntax.

Here I have not mentioned my constructor as private because I cannot allocate memory to the Mobile Object with in the Hello object. As I mentioned them as private attribute. I need an interface to allocate a memory for Mobile class. Constructor acts as an interface.

THE REASON BEHIND MAKING ATTRIBUTES AS PRIVATE INSTEAD OF PUBLIC:

Suppose, if I made my attribute software as public and if I have '' 10" different classes. Then all

'10' classes can use my attribute directly and change the information of attribute without

Contacting mobile class. So, If suppose I want to find the software used in the mobile then.

I should call attribute software using objects of all 15 classes.

But If I limit it to Mobile Class. Then any class should compulsory use Mobile class Object.

So, in order to find the software used I can simply call the public interface of Mobile class and

Know what software is used.

Let me take an example:

Suppose a Person or any individual want to position or name of upcoming software. He should

First ask the Company about upcoming software.

I can use a public interface to access this the latest software from the company.

Example:

getSoftwareUsed();

PUBLIC INTERFACE METHODS:

Public interface tend to be abstract where implementation used to concrete.

Private Implementation:

private void EnterTenNumbers()

{

}

private void pressCallButton()

{

}

CALLING IMPLEMENTATION METHOD:

Researcher explained that these methods are called in the public interface of make a call.

So, to be clear about implementation I came up with a simple example.

public void MakeAcall{

EnterTenNumbers();

pressCallButton();

}

Important thing to be noted is that all the implementation methods are called with in the same class. They cannot directly be accessed by other class Objects. From Sebti et al. (2015, p. 52). MCU (Movement Control Unit) responds OBD (On Board Device) through an isCovered method by using DEU (Data Exchange Unit). Here DEU Unit methods are hidden.

CLASS DESIGN GUIDELINES:

Primary goal of object oriented programming is to model as the way people think. Designing a class is the object oriented approach for creating a model. The object created interacts as if the real world objects.

“The interface of a well-designed object describes the services that the client wants accomplished.”

From above statement interfaces are virtually present in front of the client. To obtain this always make your attributes as private and provide minimum interface to the user.

For example:

Consider a laptop and any person. Both are objects. These two objects interact with each other using class design. The class should be designed by providing minimum interfaces and encapsulate all the data filed. By doing this both objects can access the only the interface and not the data.

If my class does not satisfy any requirement of the class. I can say that I have not build the class in a comprehensive way.

CREATE WITH MORE INTERFACE OR LESS:

Firstly, with more interfaces the user gets accessed to unnecessary data.

Secondly, although if I provide less interface. I can add interfaces by extending the class. This is nothing but inheritance. From (Jeong Yang, Young Lee, David Hicks & Kai H. Chang, 2015, p. 4). Inheritance is explain by taking Shape as a parent class Child as a Cylinder, Sphere etc. Here I provide a simple example to be clear about inheritance.

Let me explain this by an example:

Suppose if a person object need to know the brand of the laptop.

If I provide laptop with a public interface then the person knows the name of the laptop.

Java code:

class Laptop

{

private String brand = "lenovo";

public  String getBrand()

{

return brand;

}

}

class Person{

{

Person  myPerson1 = new person();

myPerson1.getBrand();

}

}

CONCLUSION:

I clearly described all basic concepts of object oriented design such as class, object, Constructor. I cover some of advanced concepts implementation, interfaces, Peripherals of an object. My idea is if I have thorough knowledge on all the basic concepts then I learn any advanced technology. Clearly, by above examples and definitions I can say Technology is transient. Object oriented design concepts are eternal.

REFERENCES:

Matt, Weisfield. (2013).*The object oriented thought process* (4th ed.). Cleveland, Ohio. Addison-Wesley Professional.

Murat Oruc, Fuat Akal & Hayri Sever. (2016). Detecting Design Patterns in Object-Oriented Design Models by Using a Graph Mining Approach. *A tool to assess the object oriented design patterns of any software. Different projects were used to assess the efficiency object oriented design used in the projects*.

Sebti Mouelhi, Khalid Agrou, Samir Chouali & Hassan Mountassir. (2015). Object-oriented component-based design using behavioral contracts: application to railway systems. *Object oriented designing concepts are used in CBTC railway system. Basic Object oriented techniques such as interface and implementation are thoroughly used in building up applications for railway systems*.

Jeong Yang, Young Lee, David Hicks & Kai H. Chang. (2015). Enhancing object-oriented programming education using static and dynamic visualization. *As it is difficult for a new student to understand the object oriented designing concepts. Base for many advanced programming languages like java. Two kinds of approaches are used in this. One is static conscience and other is dynamic conscience. That is through UML diagrams and run time environments.*”