หลักการของการพัฒนาซอฟท์แวร์ด้วย Object-Oriented Programming

OOP หรือ Object Oriented Programming การเขียนโปรแกรมเชิงวัตถุ คือการเขียนโปรแกรมโดย

การมองว่าส่วนประกอบของโปรแกรมเป็นเสมือนวัตถุชิ้นหนึ่งที่ประกอบไปด้วยคุณสมบัติ(property) ซึ่ง

จะสามารถอธิบายได้ว่าวัตถุนี้คืออะไร และ วิธีการ(method) ซึ่งจะสามารถอธิบายพฤติกรรมของวัตถุนั้นว่า

สามารถท าอะไรได้

การเขียนโปรแกรมแบบ OOP เป็นการแบ่งซอฟท์แวร์หรือโปรแกรมออกเป็นส่วนๆ เรียกว่า class

โดยการนิยาม class และ object ก็เพื่อท าให้สามารถน าส่วนของซอฟท์แวร์หรือโปรแกรมส่วนนั้นกลับมา

เรียกใช้งานได้อีกเพื่อลดความซ้ าซ้อนและเวลาในการพัฒนาโปรแกรมลง

การท างานของ class จะถูกก าหนดโดยส่วนอินเตอร์เฟสของ method ส่วนการท างานของส่วนที่

เป็น codeจะไม่ถูกค านึงถึงมากนักในการออกแบบภาษา OOP สนใจเฉพาะข้อมูลที่จะถูกประมวลผล

มากกว่าฟังก์ชั่นที่ท าการประมวลข้อมูลนั้น ๆ

หลักส าคัญของ OOP

• Class and Subclass

• Encapsulate

• Inheritance

• Polymorphism

• Abstract Data Type

Class

-คือกลุ่ม (category) ของ objects ที่มีคุณสมบัติและพฤติกรรมที่เหมือนกัน โดยclass จะต้อง

ประกอบไปด้วย data, behavior และ interface

-คือต้นแบบ (prototype) หรือพิมพ์เขียว ที่ก าหนดตัวแปรและวิธีการเพื่อน าไปใช้ได้ในทุก object

ของ class

Objects

คือ สิ่งใดๆก็ตาม ซึ่งมีคุณลักษณะ (State) บ่งบอกถึงความเป็นตัวของมันเองในขณะนั้น และ

สามารถแสดงพฤติกรรม (Behavior) ของตัวเองออกมาได้ เช่น รถยนต์สีน้ าเงิน : มีความหมายคือ วัตถุ

ประเภทรถยนต์มีคุณลักษณะของสีเป็นสีน้ าเงิน และมีพฤติกรรมที่แสดงถึงการเคลื่อนที่ และหยุดได้

หรือกล่าวได้ว่า object ก็คือข้อมูลของ class (เป็น entities ของ class) ซึ่งทุกๆ อย่างจะจัดเป็น

objects โดยต้องประกอบไปด้วย- ชื่อ (Identity)

- สถานะ (State) คุณสมบัติ หรือค่าของข้อมูล ซึ่งแทนด้วย value

- พฤติกรรม (Behavior) ที่ระบุว่าสามารถท าอะไรได้บ้าง ซึ่งแทนด้วย method

Method

คือ function ที่บ่งบอกพฤติกรรมของ object ว่าท าอะไรได้บ้าง ก าหนดไว้ใน class โดยต้อง

ประกอบด้วย ชื่อของ method เรียกว่า Identifier ตามด้วยเครื่องหมายวงเล็บ () โดยในวงเล็บอาจมี

parameter list อยู่หรือไม่ก็ได้เช่น

getBalance()

raiseSalary( float Salary, float Percent )

Constructor Method

คือเมธอดที่ใช้ส าหรับสร้าง instance object ของคลาสนั้นๆ โดยที่ชื่อเมธอดนี้ต้องเหมือนกับชื่อ

คลาส และใช้ส าหรับ initialize ข้อมูลให้กับ instance variable โดยจะไม่มีการถ่ายทอดให้กับ subclass และ

ไม่มีการ return ค่า

Message

คือค าสั่งหรือข้อความที่จะให้ข้อมูลหรือตัวแปรใดท างาน ก็คือ parameter ในภาษาอื่นที่ไม่ใช่ OOP

คือใช้เพื่อน าส่งค่าข้อมูลระหว่าง object โดยใน message นั้นต้องประกอบด้วย

- Destination ก็คือชื่อของ object

- Method

-Parameters

Accessibility

-public : เข้าถึงได้ในทุกที่

-private : เข้าถึงได้เฉพาะภายใน class เท่านั้น ไม่รวม sub class

-protected : เข้าถึงได้เฉพาะภายใน class และ sub class ที่สืบทอดกันมา (Inherit)

-default : ถ้าไม่ระบุ จะเข้าถึงข้อมูลภายใน class และอยู่เพ็กเกจเดียวกัน

Encapsulate

คือการปิดบัง หรือจ ากัดการเข้าถึงข้อมูลบางอย่าง (Information hiding) ที่ไม่จ าเป็นต้องให้ส่วนอื่น

รับรู้ ยกตัวอย่างเช่น เราจะไม่สนใจหรือมองเห็นได้ว่า เครื่องเล่น CD จะแปลงสัญญาณดิจิตอล ออกมาเป็น

เพลงได้อย่างไรเราใช้และติดต่อกับเครื่องแค่ควบคุมการท างานผ่านแผงควบคุม เช่น เปิด-ปิด เล่น เร่งเสียง เปลี่ยนแทร๊กไปข้างหน้า ย้อนกลับ เป็นต้น โดยเราต้องออกแบบควบคุมกฎเกณฑ์ต่างๆ ของซอฟท์แวร์ให้

สอดคล้องกับความเป็นจริง

Encapsulate เป็นคุณสมบัติของ object ซึ่งมีลักษณะดังนี้

-ก าหนดขอบเขตที่ชัดเจนให้กับ object

-ก าหนดอินเตอร์เฟสว่าจะติดต่อกับ object อื่นๆ อย่างไร

- ส่วนอิมพลิเมนต์ไม่สามารถเข้าถึงได้ภายนอกขอบเขตของ class

Inheritance

คือการถ่อยทอดข้อมูล (ซึ่งก็คือ state และ behavior) จาก class ล าดับที่สูงกว่า (super class หรือ

parent class) ไปยังล าดับที่ต่ ากว่า (subclass) โดยที่ subclass นั้นสามารถเปลี่ยนแปลง หรือแทนที่ข้อมูล

(override) ที่ได้รับการถ่ายทอดมานั้นได้

Polymorphism

การท าให้ message อันหนึ่งสามารถส่งให้ object แต่ละตัวใน class และ subclass ตอบสนองต่อ

message อันเดียวกัน ในลักษณะที่เหมาะสบกับ class ของตัวเองยกตัวอย่างเช่น method print นี้สามารถส่ง

ให้ทุก object ของ class และ subclass ที่ท าให้ object นั้นรู้จัก method print และแต่ละ object ที่ต่างกันจะ

ตอบสนองต่อ message นี้ต่างกันออกไป ตามความสามารถในการใช้

Abstract Data Type (ADT)

รูปแบบชนิดของข้อมูลที่ผู้พัฒนาเป็นผู้ก าหนดขึ้นมาเอง

เหตุผลที่ OOP มีบทบาทมากขึ้น

• ง่าย และรวดเร็ว ท าให้ลดเวลาในการพัฒนาลงไปได้

• เพิ่มปริมาณงานที่ได้ และมีความน่าเชื่อถือมากกว่า

• สามารถน า code กลับมาใช้ได้อีก (เรียกใช้ class)

• ท าต้นแบบ (Prototyping) ได้รวดเร็วกว่า

• ลดต้นทุนในการสร้าง และบ ารุงรักษาซอท์แวร์

• การเปลี่ยนแปลงแก้ไข ไม่ท าให้เกิดผลกระทบไปยังภายนอก class โครงสร้างของ Class

โปรแกรมที่สร้างจากภาษาจาวา ต้องมีออปเจ็คหรือคลาสอย่างน้อยหนึ่งตัวโดยมีรูปแบบโครงสร้าง

ดังนี้

class Class\_Name

{ Data\_Member

Method\_Member }

รูปแบบของ Data Member

[ Accessibility ] [ final ] [ static ] Data\_Type Data\_Name

เช่น private static String stu\_name ;

ดาต้าคือส่วนประกอบส่วนหนึ่งของคลาส ถูกก าหนดเพื่อใช้ส าหรับเก็บข้อมูล เช่นคลาส

Pen มีดาต้า Color ไว้เก็บข้อมูลสี

final : เป็นคีย์เวิร์ดตัวหนึ่งซึ่งใช้บอกว่าดาต้าตัวนั้นใช้ส าหรับเก็บข้อมูลโดยที่ข้อมูลจะไม่สามารถ

ท าการเปลี่ยนแปลงได้ไม่ว่าในกรณีใดๆ ปกติจะใส่คีย์เวิร์ดนี้ไว้เมื่อต้องการให้ดาต้าเก็บข้อมูลที่เป็นข้อมูลที่

คงที่(Constant) ตลอดการท างานของโปรแกรม

static : เป็นคีย์เวิร์ด ส าหรับใช้บอกถึงคุณลักษณะพิเศษในการใช้งาน เมื่อมีการก าหนดน าหน้าดาต้า

ใดๆ แล้ว ดาต้านั้นจะมีคุณลักษณะดังนี้

• ดาต้าจะถูกโหลดลงในหน่วยความจ าและพร้อมที่จะถูกใช้งานในทันทีเมื่อมีการอ้างถึง

ตามข้อก าหนดของระดับการเข้าถึง (Accessibility)

• ดาต้าจะอยู่ในหน่วยความจ าเพียงตัวเดียว ไม่มีการ copy static method ไปกับ instance

object คือจะผูกติดอยุ่กับคลาสเท่านั้น ดังนั้นจึงสามารถใช้ดาต้าเป็นที่เก็บข้อมูลรวมของกลุ่มคลาสเดียวกัน

ได้

• ไม่สามารถเรียกใช้ instance variables หรือ instance method ได้

• ไม่สามารถใช้คีย์เวิร์ดthisได้โครงสร้างของ Object

เมื่อกล่าวถึงค าว่าออปเจ็คทุกคนจะเข้าใจว่าเป็นชื่อเรียกที่ใช้เรียกส่วนที่สร้างจากโครงร่างของคลาส

ซึ่งออปเจ็คถูกสร้างโดยการใช้ค าสั่งในภาษาจาวาสร้างขึ้น จากขั้นตอนสองขั้นตอน คือ

1. ขั้นตอนประกาศชื่อตัวแปรอ้างถึง (Reference Variable)

คือขั้นตอนที่ใช้ก าหนดชื่อตัวแปรเพื่อใช้ในการอ้างถึงไปยังพื้นที่ในหน่วยความจ า โดยพื้นที่ส่วนนี้

เป็นส่วนที่จัดเก็บอินสแตนซ์ที่สร้างขึ้นมาในภายหลังโดยมีรูปแบบการก าหนดดังนี้

- Class\_Name Reference\_Name

- Class\_Name : คือชื่อของคลาสซึ่งอ้างถึงด้วย Reference\_Name ได้

- Reference\_Name : คือชื่อตัวแปรที่ถูกก าหนดขึ้นโดยผู้เขียนโปรแกรม

2. ขั้นตอนการสร้างอินสแตนซ์ (Instance of Class)

อินสแตนซ์ คือพื้นที่ในหน่วยความจ าซึ่งสร้างตามแบบโครงร่างของคลาสใดๆ ดังนั้นอินสแตนซ์

คือพื้นที่ที่สามารถเก็บข้อมูลและประมวลผลตาม ส่วนที่สามารถประมวลผลได้นั่นเอง มีรูปแบบการสร้าง

ดังนี้

new Class\_Constructor(Parameter\_List )

-new : คือคีย์เวิร์ดของภาษาจาวา เพื่อเป็นค าสั่งให้มีการจัดสรรพื้นที่หน่วยความจ า ตามโครงสร้าง

ของคลาสจากที่ก าหนดไว้แล้วในขั้นตอนที่ผ่านมา

- Class\_Constructor : คือเมธรอดของคลาสที่เป็นคอนสตรักเตอร์(เมธรอดที่เป็นคอนสตรักเตอร์มี

ชื่อเดียวกับชื่อคลาสเสมอ)โดยคลาสหนึ่งคลาสสามารถมีเมธรอดคอนสตรักเตอร์นี้ได้หลายเมธรอด(ชื่อ

เดียวกันแต่พารามิเตอร์ต่างกัน)

-Parameter\_List : คือพารามิเตอร์ (ตัวแปรหรือค่าคงที่ใดๆ) ที่ผ่านข้อมูลเข้าไปสู่เมธรอดคอนสตัก

เตอร์ซึ่งจ านวนหรือชนิดข้อมูลที่เป็นพารามิเตอร์ขึ้นอยู่ว่าผู้เขียนโปรแกรมเรียกใช้คอนสตรักเตอร์แบบ

ไหน

ตัวแปรอ้างถึงและอินสแตนซ์ (Reference variable and Instance)

ในภาษาจาวาตัวแปรอ้างถึงและอินแตนซ์มีความสัมพันธ์ซึ่งกันและกัน กล่าวคือตัวแปรอ้างถึงถูก

ก าหนดขึ้น เพื่อประโยชน์ในการอ้างไปยังต าแหน่งของอินสแตนซ์ที่อยู่ในหน่วยความจ าโดยใน

หน่วยความจ าบนเครื่องคอมพิวเตอร์บางครั้งสามารถสร้างอินสแตนซ์ได้หลายๆอินสแตนซ์ของการท างาน

แต่ละครั้งจึงจ าเป็นต้องใช้ชื่อตัวแปรอ้างถึงเพื่อให้สามารถเข้าใช้งานดาต้าและเมธรอดในอินสแตนซ์ได้

ตัวอย่างเช่น-ถ้าพิจารณาถึงลูกโป่งสวรรค์ เปรียบกับคนขายลูกโป่งสวรรค์ตามงานทั่วไป ถ้าให้ลูกโป่งแต่ละลูก

เป็นอินสแตนซ์ และให้เชือกแต่ละเส้นคือตัวแปรอ้างถึงดังนั้นเมื่อต้องการลูกโป่งลูกใดๆ ก็ให้หยิบเส้น

เชือกขึ้นมาท าให้รู้ว่าเป็นเชือกที่ผูกอยู่กับลูกโป่งลูกใด

- หรือจากค าสั่ง Integer x = new Integer(237)

หมายถึงก าหนดตัวแปรอ้างถึงชื่อ x โดยมีการอ้างถึงไปยังอินสแตนซ์ที่เกิดจากคลาสที่ชื่อ Integer และให้

อินสแตนซ์นั้นเก็บค่า 237 ไว้ โดยที่Integer คือชื่อคลาสที่ใช้ส าหรับจัดการข้อมูลตัวเลขจ านวนเต็ม

A L A S K A : รวบรวบ

----------------------------------------------------