Part 6

Java Programming Language
Mr.Rungrote Phonkam
rungrote@it.kmitl.ac.th

Contents

1. Class vs. Object
2. Reference Variable
3. Instances
4. Reference Variable & Instance
5. Garbage Collector
6. Lift Time

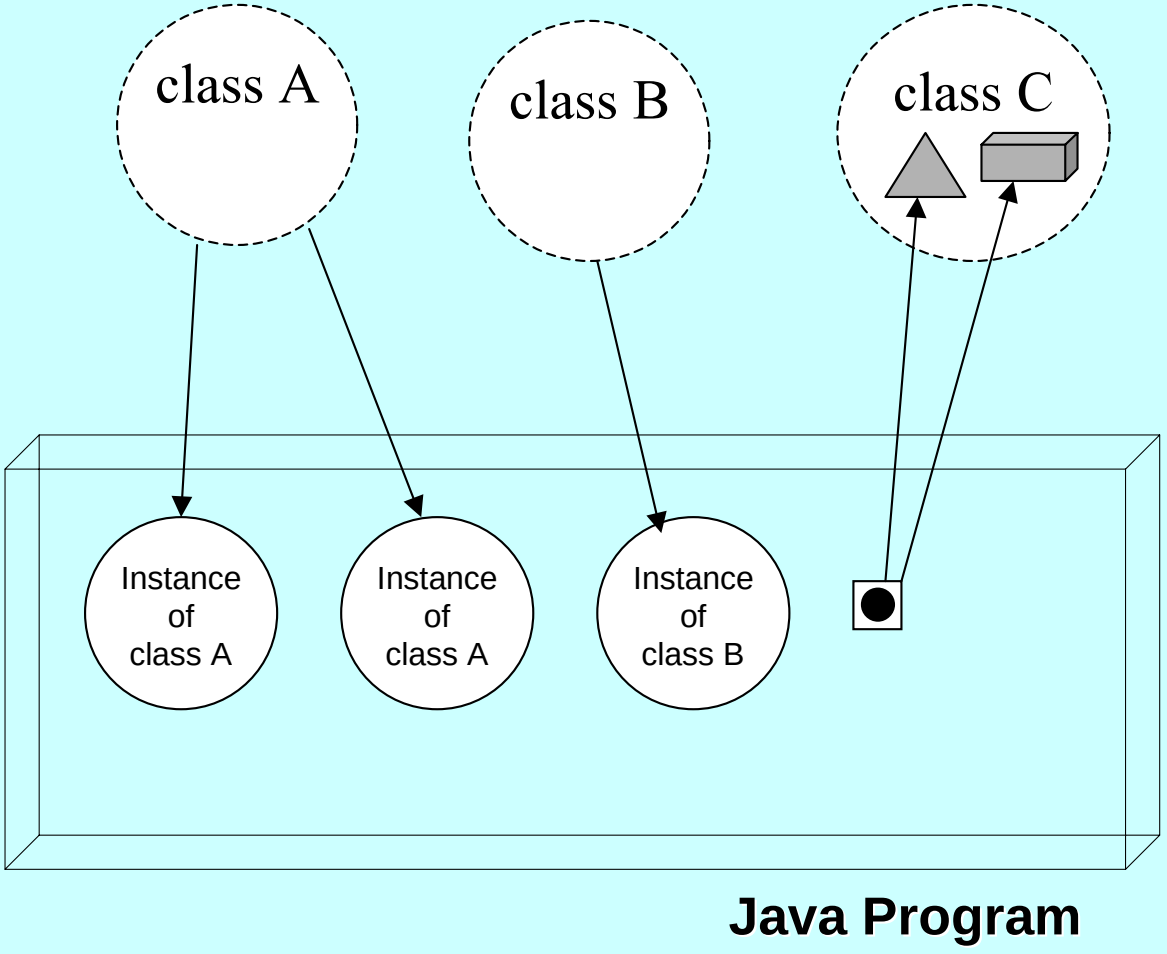
1. Class vs Object

คลาส (Classes)

- คือรูปแบบ Template ของข้อกำหนดการ Encapsulation ในแนวคิด Object-Oriented
- มีการกำหนดสมาชิก Data และ Method
- สามารถกำหนดระดับการเข้าถึงแต่ละส่วนได้ใน 4 ระดับคือ

  protected, private, public และ **package**
- ในภาษาจาวาคลาสหนึ่งคลาส คือไฟล์ Bytecode หนึ่งไฟล์
- คลาสถูกใช้งานได้โดยการสร้าง Instance ขึ้นมาจากคลาส หรือบางส่วนของคลาสมีคุณสมบัติเป็น static

1. Class vs Object



1. Class vs Object

ออปเจ็ค (Object)

- คือการนำคลาสมาใช้ ด้วยวิธีการสร้างอินสแตนซ์ (Instance) ด้วยคำสั่ง new
- ประกอบจากตัวแปรอ้างถึง (Reference Variable) และอินสแตนซ์ (Instance)
- Instance แต่ละตัว จะมี Data และ Method ของตัวเอง
- ยกเว้น Data หรือ Method ที่ถูกระบุด้วย Static เท่านั้น จะมีหนึ่งเดียว และถูกใช้งานได้จาก Instance ใดๆของคลาสนั้นๆ
- การทำลายออปเจ็ค ถูกทำลายด้วยระบบ Garbage Collector โดยอัตโนมัติ

2. Reference Variable

ตัวแปรอ้างถึง (Reference Variable)

- คือตัวแปรที่ใช้เป็นทั้ง Data Member หรือ Variable ในเมธรอด หรือ Parameter
- สร้างขึ้นโดยการตั้งชื่อ จากชนิดข้อมูล(Data Type) ที่เป็นชื่อคลาส
- ใช้สำหรับเป็นชื่อเพื่ออ้างไปยัง Instance ที่สร้างจากคลาสเดียวกับที่ตัวแปรอ้างถึงสร้างขึ้นมา

เมื่อสร้างตัวแปรอ้างถึงจากคลาส Integer ก็จะถูกใช้อ้างไปยังอินสแตนซ์ที่สร้างจากคลาส Integer เช่นกัน

2. Reference Variable

รูปแบบ

```
Class_Name Reference_Name
```

เมื่อ

Class_Name	คือชื่อของคลาสซึ่งอ้างถึงด้วย Reference_Name ได้
Reference_Name	คือชื่อตัวแปรที่ถูกกำหนดขึ้นโดยผู้เขียนโปรแกรม

2. Reference Variable

ตัวอย่าง

```
Integer a;
Double d;
```

3. Instances

อินสแตนซ์ (Instance)

- การนำเอาคลาสใดๆที่สร้างขึ้นเองหรืออยู่ในไลบรารี่(Library) มาใช้งาน
- การสร้างอินสแตนซ์ใช้คำสั่ง new ในการสร้าง
- ในโปรแกรมสามารถสร้างอินแตนซ์ได้หลายๆอินสแตนซ์ ไม่ว่าจะเป็นคลาสกี่ตัวก็ตาม
- อินสแตนซ์แต่ละตัวถึงแม้จะมีโครงสร้างจากคลาสเดียวกัน แต่ก็สามารถกำหนดข้อมูลภายในให้ต่างกันได้ โดยการเรียกใช้งาน Constructor ของคลาส (จะกล่าวอีกที่ในภายหลัง)
- เมื่อคลาสใดๆถูกนำไปสร้างอินสแตนซ์ Bytecodes ของคลาสนั้นจะถูกโหลดผ่าน JVM ทำให้โปรแกรมโหลดเฉพาะ Bytecodes ที่จำเป็นในการใช้งานเท่านั้น

3. Instances

รูปแบบ

new Class_Constructor **(** *Parameter_List* **)**

เมื่อ

new คือคีย์เวิร์ดให้มีการจัดสรรพื้นที่หน่วยความจำ

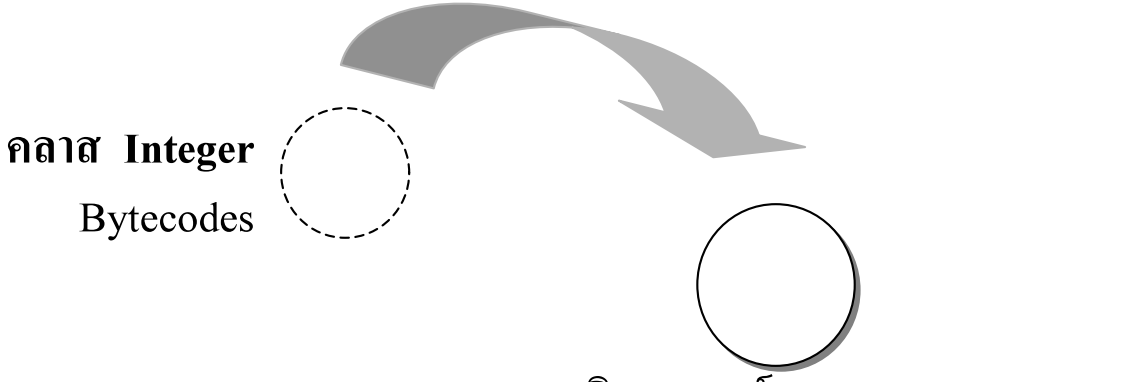
Class_Constructor คือคอนสตรักเตอร์

Parameter_List คือพารามิเตอร์ (ตัวแปรหรือค่าคงที่ใดๆ)

3. Instances

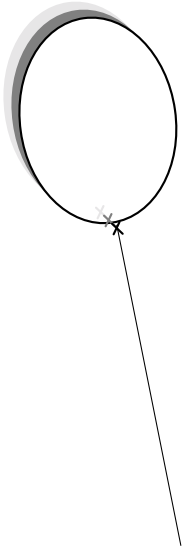
ตัวอย่าง

new Integer(105)

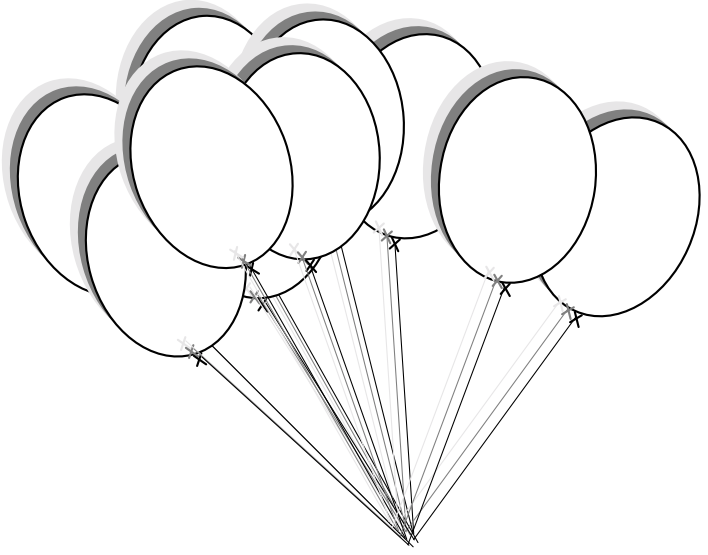


4. Reference Variable and Instance

อินสแตนซ์
(ลูกโป่ง)



ตัวแปรอ้างถึง
(เชือก)



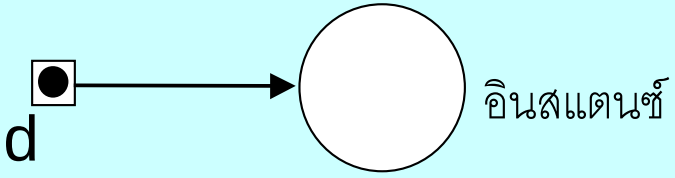
4. Reference Variable and Instance

- การกำหนดตัวแปรอ้างถึง ไปยังอินสแตนซ์

วิธีที่ 1

```
Class_Name Reference_Name
Reference_Name = new Class_Constructor ( Parameter_List )
```

```
Double d;
d = new Double(10.02);
```

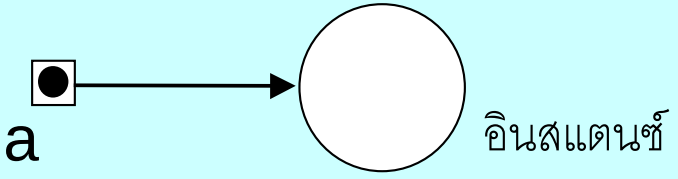


4. Reference Variable and Instance

วิธีที่ 2

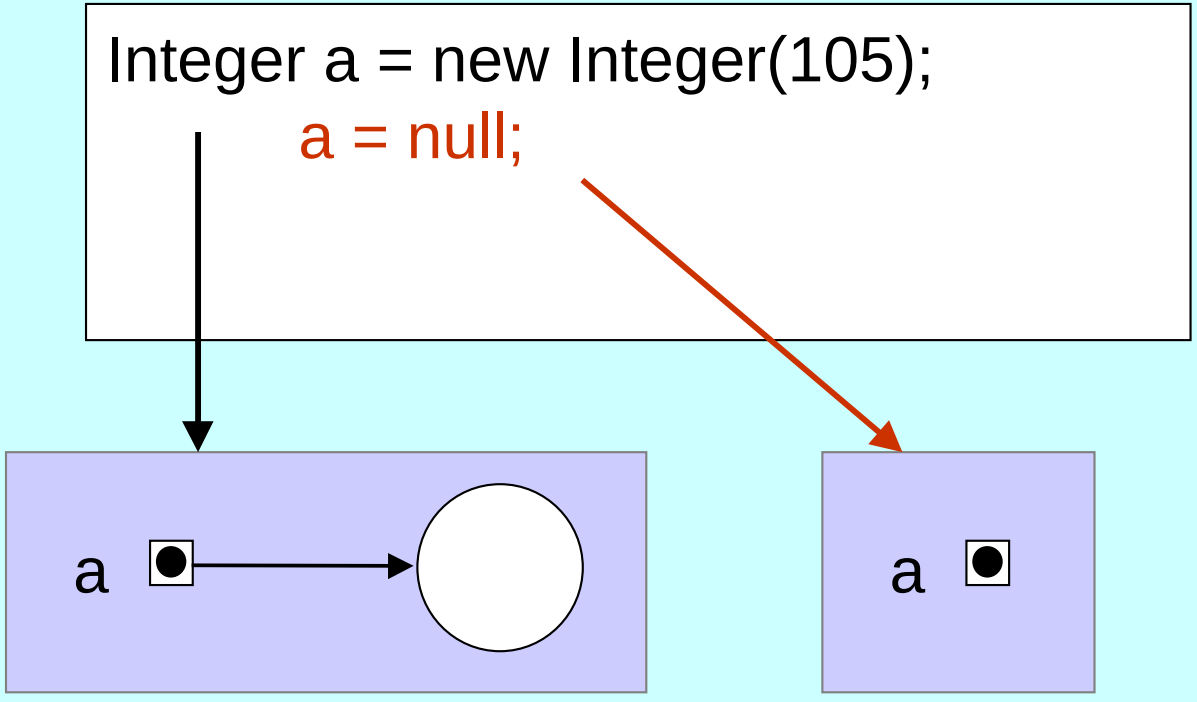
Class_Name Reference_Name = **new** Class_Constructor **(** *Parameter_List* **)**

Integer a = new Integer(105)



4. Reference Variable and Instance

ค่า **null** หมายถึง การทำให้ไม่มีการอ้างหรือชี้ไปยังตำแหน่งใดๆ



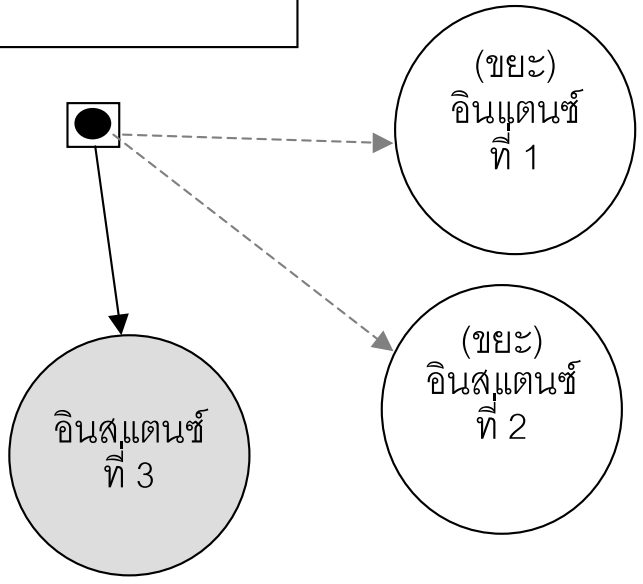
5. Garbage Collector

กลไกกำจัดพื้นที่ในระบบที่ไม่ถูกใช้งาน

- พื้นที่ที่ไม่ถูกใช้งานคือ Instance ที่ไม่ได้ถูกอ้างถึงด้วยตัวแปรอ้างถึง
- การทำงานของกลไกเป็นไปอย่างอัตโนมัติ
- ลดความยุ่งยากในการเขียนโปรแกรม เนื่องจากโปรแกรมเมอร์ไม่จำเป็นต้องดูแล และทำลายพื้นที่ที่ไม่ถูกใช้งาน
- เลี่ยงอันตรายจากการอ้างถึงหน่วยความจำ ในตำแหน่งที่ไม่เหมาะสม
- ต่างกับภาษาอื่น เช่น C หรือ C++ ที่ต้องใช้คำสั่ง free เพื่อทำลายอินสแตนซ์เอง

5. Garbage Collector

```
Date today;
today = new Date(); \\ อินสแตนซ์ตัวที่ 1
……..
today = new Date(); \\ อินสแตนซ์ตัวที่ 2
……..
……..
today = new Date(); \\ อินสแตนซ์ตัวที่ 3
```



6. Lift Time

ช่วงชีวิต

- การใช้งานได้หรือไม่ได้
- ช่วงชีวิตของ ตัวแปร
  - ถูกกำหนดโดยสโคป { }
- ช่วงชีวิตของอินสแตนซ์
  - ถูกกำหนดโดยการสร้าง และ Garbage Collector
- ช่วงชีวิตของออปเจ็ค
  - ถูกกำหนดโดยตัวแปรอ้างถึงในสโคป

6. Lift Time

```
1: class RefVarLife1
2: { public static void main(String args[])
3:   {       Double d;
4:           d = new Double(155.287);
5:           System.out.println( d.doubleValue( ) );
6:   }
7: }
```

	begin	end
Ref Variable:	3	6
Instance:	4	6

6. Lift Time

```
1: class RefVarLife2
2: { public static void main(String args[])
3:   {        Long lg;
4:            lg = new Long(150);
5:            System.out.println( lg.longValue( ) );
6:            lg = new Long(200);
7:            System.out.println( lg.longValue( ) );
8:   }
9: }
```

	begin	end
Ref Variable:	3	8
Instance ตัวแรก:	4	6
Instance ตัวที่สอง:	6	8