

Using Objects

การใช้วัตถุ

This chapter

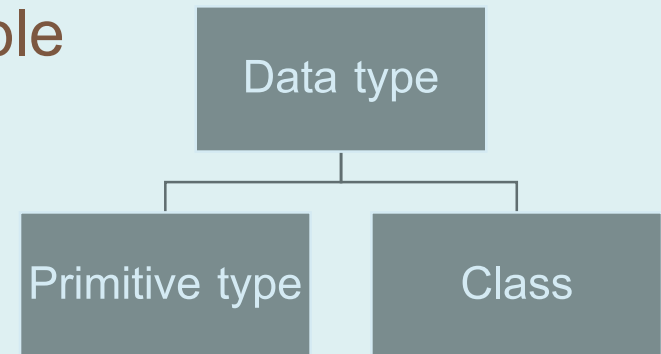
- ค่า (value) และ ชนิดข้อมูล (data type)
- ชนิดข้อมูล (data type)
 - ชนิดข้อมูลแบบ primitive
 - วัตถุ (object) และคลาส (class)
- ตัวแปร (variable)
 - การประกาศตัวแปร (variable declaration)
 - การกำหนดค่า (assignment)
- การสร้างวัตถุ
- การใช้เมทอด (method)

ค่า (value) และ ชนิดข้อมูล (data type)

- ค่าในโปรแกรมมีหลายชนิด
- Literals คือ ข้อความที่อ้างถึงค่าของข้อมูลชนิดต่าง ๆ เช่น 35, 3.14
- Literal สำหรับค่าแต่ละชนิด มีรูปแบบต่างกัน
 - Integers
 - 93, -31
 - ไม่มี . อาจขึ้นต้นด้วย + หรือ -
 - Floating-point numbers
 - 93., 3.1415, 3E-7
 - มี . หรือ E ตามด้วยเลขยกกำลังที่เป็นจำนวนเต็ม
 - Characters
 - 's', '\t'
 - Strings
 - "dept. name"

ชนิดข้อมูล (data type)

- ชนิดข้อมูลแบบ primitive
 - Integers: byte , short , int , long
 - Floating-point numbers: float , double
 - Truth values: boolean
 - Characters: char
- วัตถุ (object) และคลาส (class)
 - คลาสเป็นชนิดของวัตถุซึ่ง
 - ประกอบด้วยหลายค่า
 - มีเมทอด (method) ที่กำหนดการทำงานกับวัตถุ
 - จาวามีคลาสต่าง ๆ ให้โปรแกรมเมอร์ใช้ได้เลย
 - โปรแกรมเมอร์สร้างคลาสได้เอง



ตัวแปร (variable)

- เก็บใน RAM (random access memory)
- มีชื่อ (name) ที่ใช้อ้างถึงเลขที่อยู่ (address) ใน memory ที่เก็บค่า
- เปลี่ยนค่าที่เก็บในตัวแปรได้
- มีชนิด (type) ที่ระบุว่าเก็บค่าชนิดใด
- ชนิดของตัวแปรไม่เปลี่ยนตลอดชีวิต (lifetime) ของตัวแปร คือ เป็นตัวแปรแบบ static type

Identifier

- Identifier คือ ชื่อของตัวแปร คลาส หรือ เมทอด
- กฎการตั้ง identifier
 - ประกอบด้วยตัวอักษร (upper-case and lower-case letter) ตัวเลข (digit 0-9) ตัว `_` ตัว `$` เท่านั้น
 - ห้ามขึ้นต้นด้วยตัวเลข
 - ห้ามซ้ำกับคำสงวน (reserved word) เช่น `int` `for` `while`
- ตัวอย่าง
`age`, `Students`, `year2_student`, `$v1`, ~~`f.s.a`~~, ~~`3monthIncome`~~

Identifier

- ธรรมเนียม (convention) การตั้ง identifier
 - ชื่อควรสื่อความหมาย
 - ชื่อตัวแปรและเมทอด เริ่มต้นด้วยตัวอักษรตัวเล็ก (lower-case letter)
 - ชื่อคลาสเริ่มต้นด้วยตัวอักษรตัวใหญ่ (upper-case letter)
 - ชื่อที่สร้างอัตโนมัติจากเครื่องมือเริ่มต้นด้วย \$
 - camel case คือ การใช้ตัวอักษรตัวใหญ่ขึ้นต้นคำใหม่ในชื่อ เช่น `dailyExpense`

การประกาศตัวแปร (variable declaration)

```
type name = value; // initialize value  
type name;
```

ตัวอย่าง

```
int age, year;  
int age=20;  
double interestRate;
```

- ต้องประกาศตัวแปรก่อนใช้
- ควรกำหนดค่าเริ่มต้น (initialize) ให้ตัวแปร

ตัวอย่าง

```
double tax, income;  
int salary;  
// next line cause error because salary has no value  
income = salary*12;  
tax = income*0.02;
```


การกำหนดค่าตัวแปร (variable assignment)

variableName = expression;

- หาค่าของ *expression* แล้วให้ตัวแปรเก็บค่านั้นไปเก็บในตัวแปร *variableName*
- ชนิดของตัวแปรต้องเหมือนกับชนิดของค่าที่นำไปเก็บ หรือเปลี่ยนค่าให้มีชนิดเหมือนตัวแปรก่อนเก็บได้ (type coercion)

ตัวอย่าง

```
double tax, income;  
int salary=20000;  
// type conversion in next line  
income = salary*12;  
tax = income*0.02;
```

การกำหนดค่าตัวแปร (variable assignment)

Python

```
name, age = "Tim", 20  
// not in Java
```

```
st = 631245723  
st = "631245723"  
// type of st is changed
```

Java

```
name = "Tim";  
age = 20;
```

```
int st=631245723;  
st = "631245723";  
// This causes error.
```

การกำหนดค่าตัวแปร (variable assignment)

Python

```
name, age = "Tim", 20  
// not in Java
```

```
st = 631245723  
st = "631245723"  
// type of st is changed
```

Java

```
name = "Tim";  
age = 20;
```

```
int st=631245723;  
st = "631245723";  
// This causes error.
```

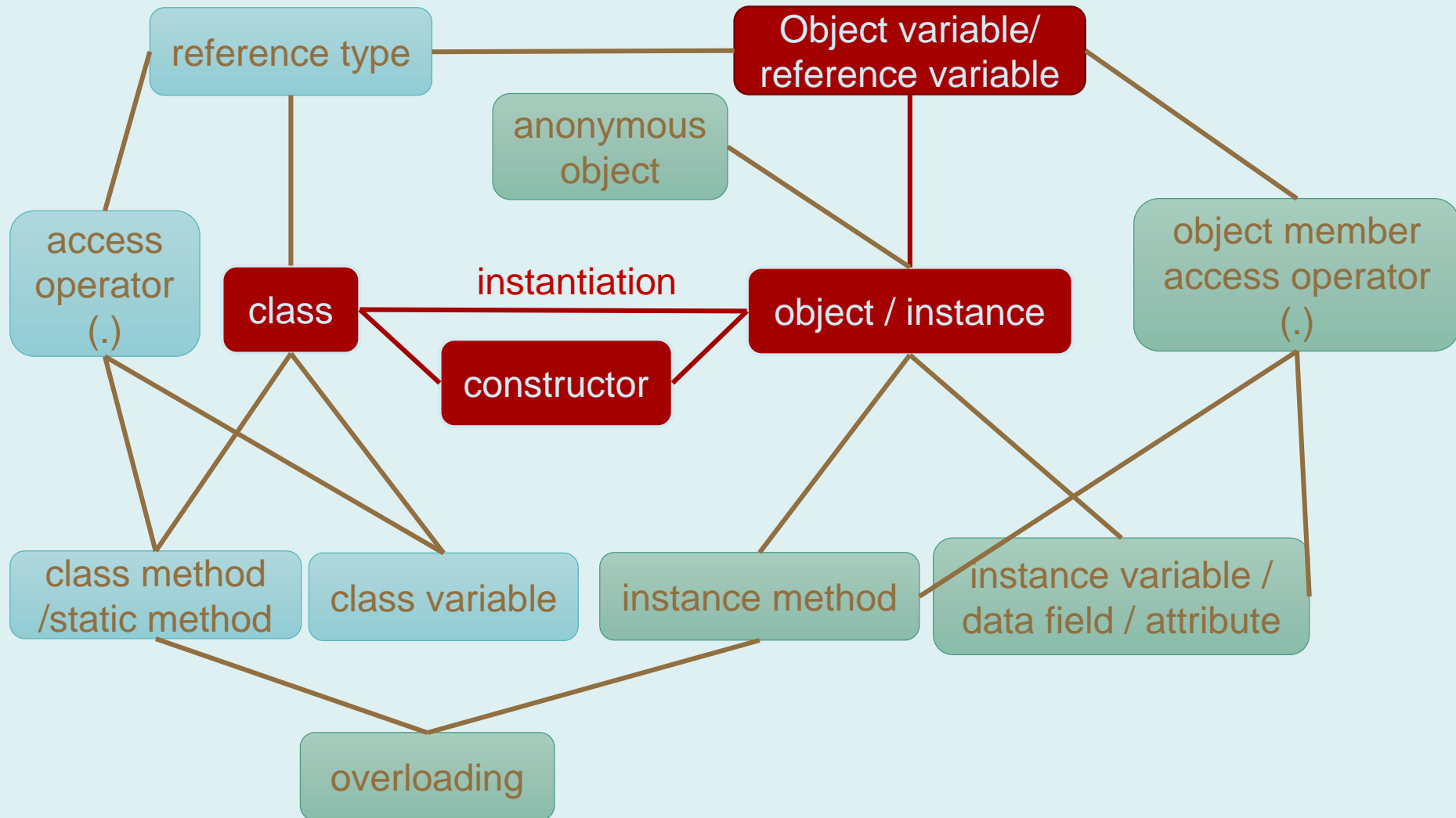
การสร้างวัตถุ

- คลาสเป็นชนิดของค่า (คล้ายกับ int, float)
- วัตถุเป็นค่าที่มีชนิดเป็นคลาสในจาวา
- ต้องมีคลาสก่อน จึงสร้างวัตถุของคลาสนั้นได้
- จาวามีคลาสที่สร้างไว้ให้ใช้ได้เลย เช่น LocalDate, Period, PrintStream
- เมื่อสร้างคลาส ต้องระบุชื่อของค่าและเมทอดที่ใช้สำหรับวัตถุในคลาสนั้น

การสร้างวัตถุ

- ตัวกระทำ **new** ใช้สร้างวัตถุโดยระบุชื่อคลาส เช่น
`Student s1 = new Student(name);`
เมื่อ `Student` เป็นคลาสที่สร้างไว้แล้ว และ
`name` เป็นค่าที่ส่งไปเป็นพารามิเตอร์สำหรับสร้างวัตถุ
- `s1` เป็น object variable หรือ reference variable ที่เก็บ object reference
- เมื่อสร้างคลาสขึ้นมา ต้องกำหนดเมทอดที่ใช้สร้างวัตถุ (constructor)
- สามารถกำหนดเมทอดอื่นที่ใช้ทำงานกับวัตถุในคลาสนั้น
- กรณียกเว้น
 - สร้างวัตถุในคลาส `String` ได้โดยใช้ literal เช่น `"name"`

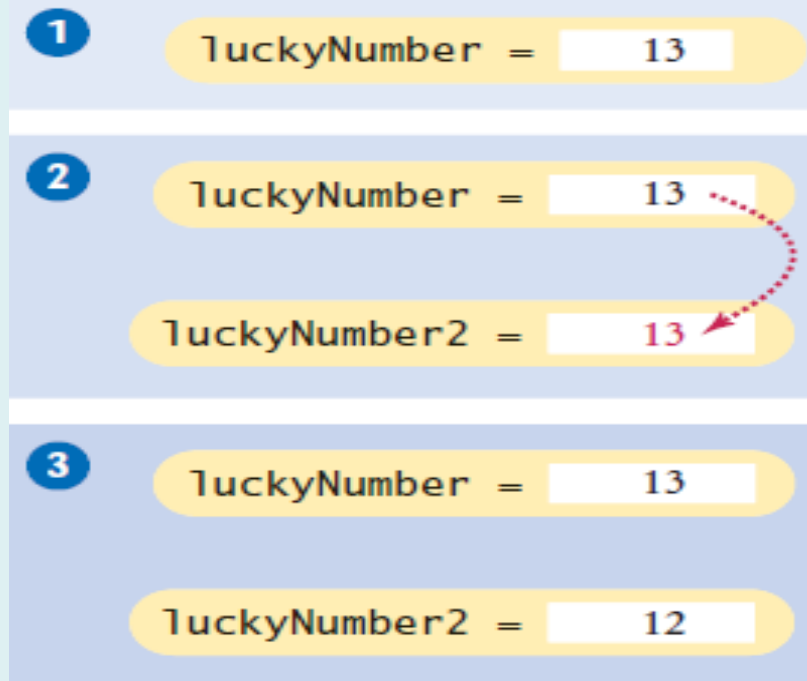
การสร้างวัด



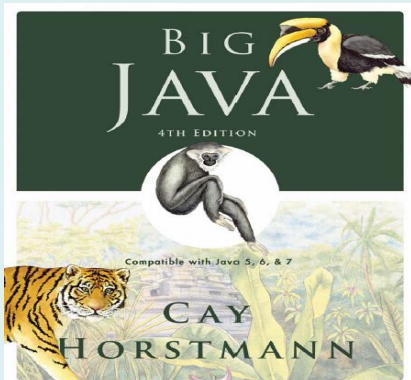
ความแตกต่างระหว่างตัวแปรที่อ้างถึง primitive data และ object

ตัวแปร luckyNumber
และ luckyNumber2
อ้างถึง primitive data

```
int luckyNumber=13, luckyNumber2;  
luckyNumber2 = luckyNumber;  
luckyNumber2 = 12;
```



Source:

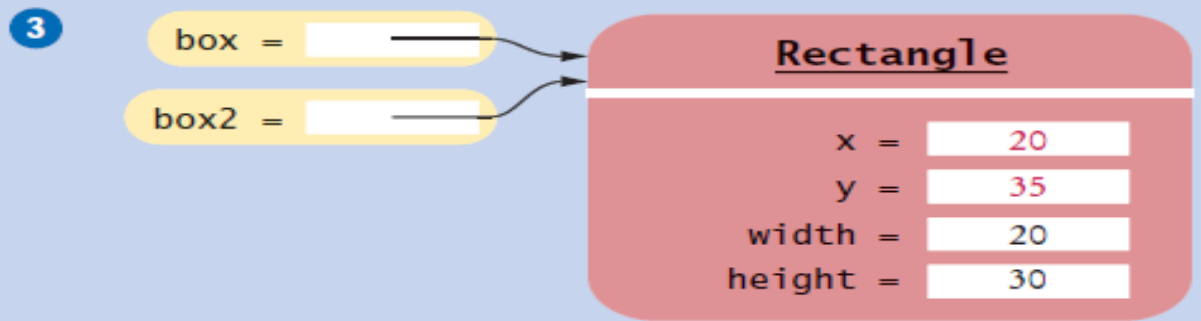
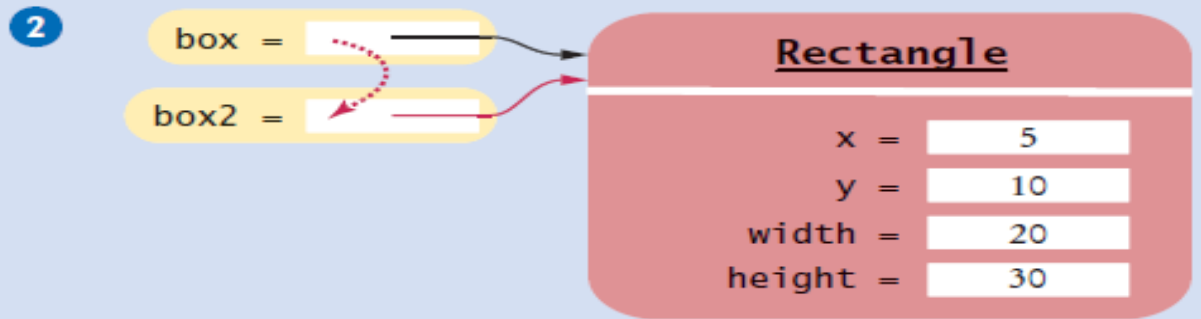
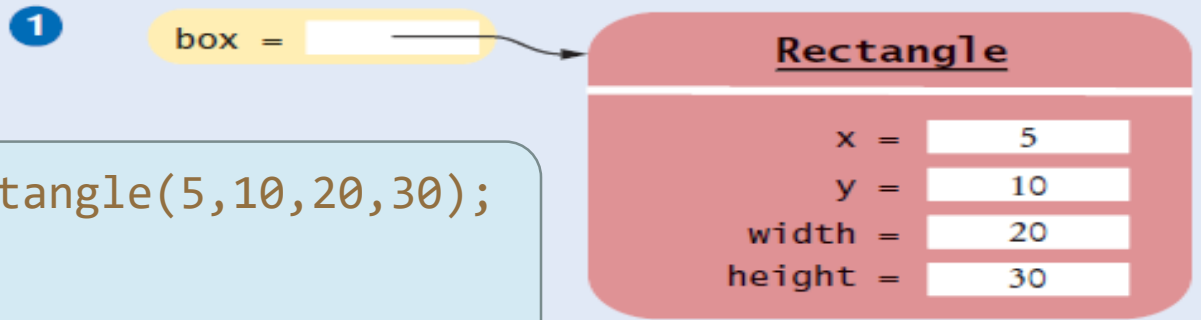
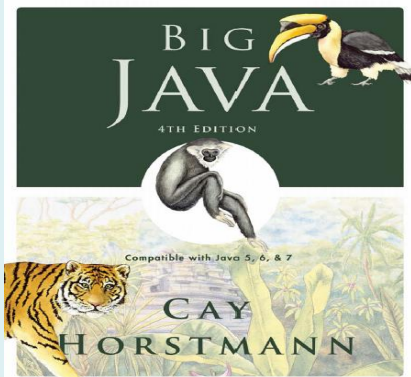


ความแตกต่างระหว่างตัวแปรที่อ้างถึง primitive data และ object

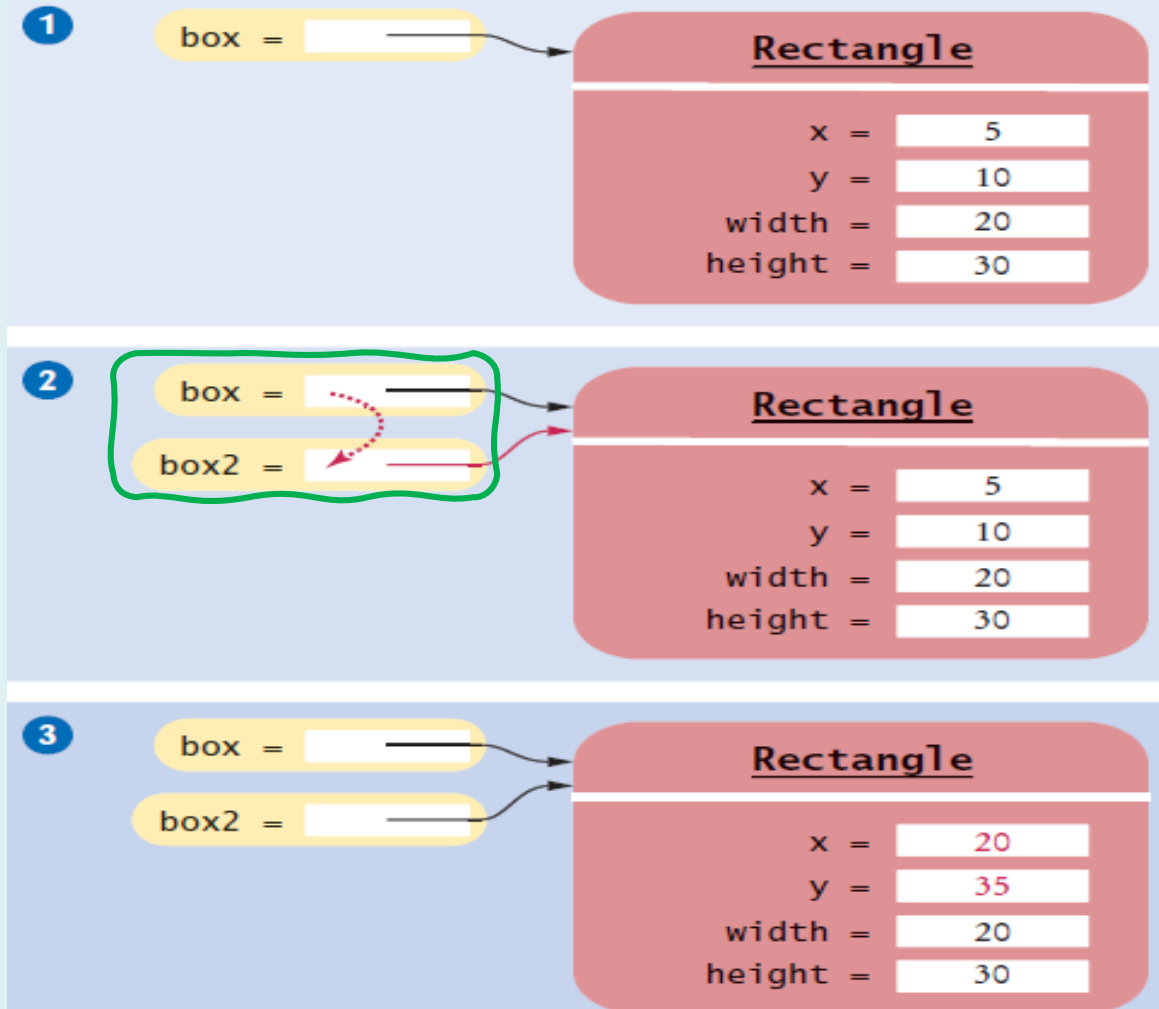
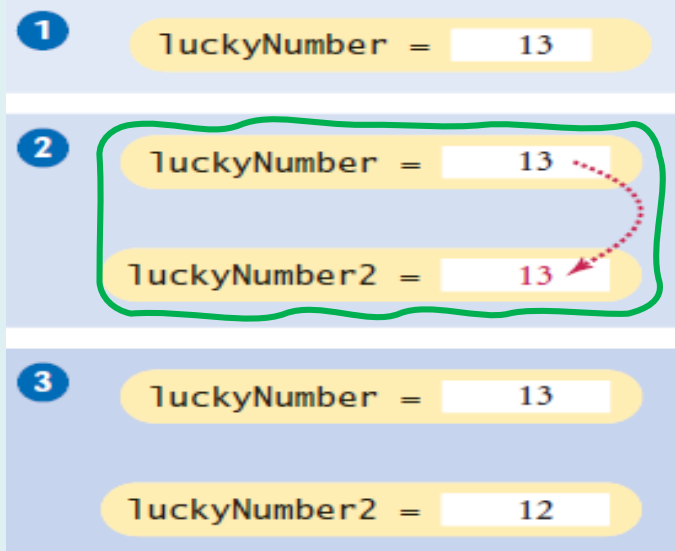
```
Rectangle box = new Rectangle(5,10,20,30);  
Rectangle box2 = box;  
box2.translate(15,25);
```

ตัวแปร box และ box2
อ้างถึง object

Source:

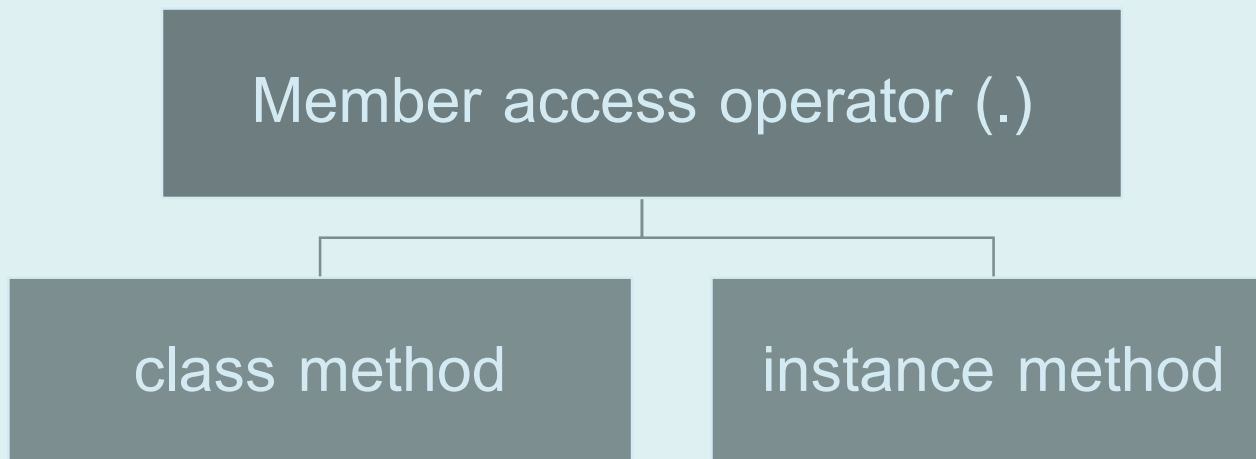


ความแตกต่างระหว่างตัวแปรที่อ้างถึง primitive data และ object

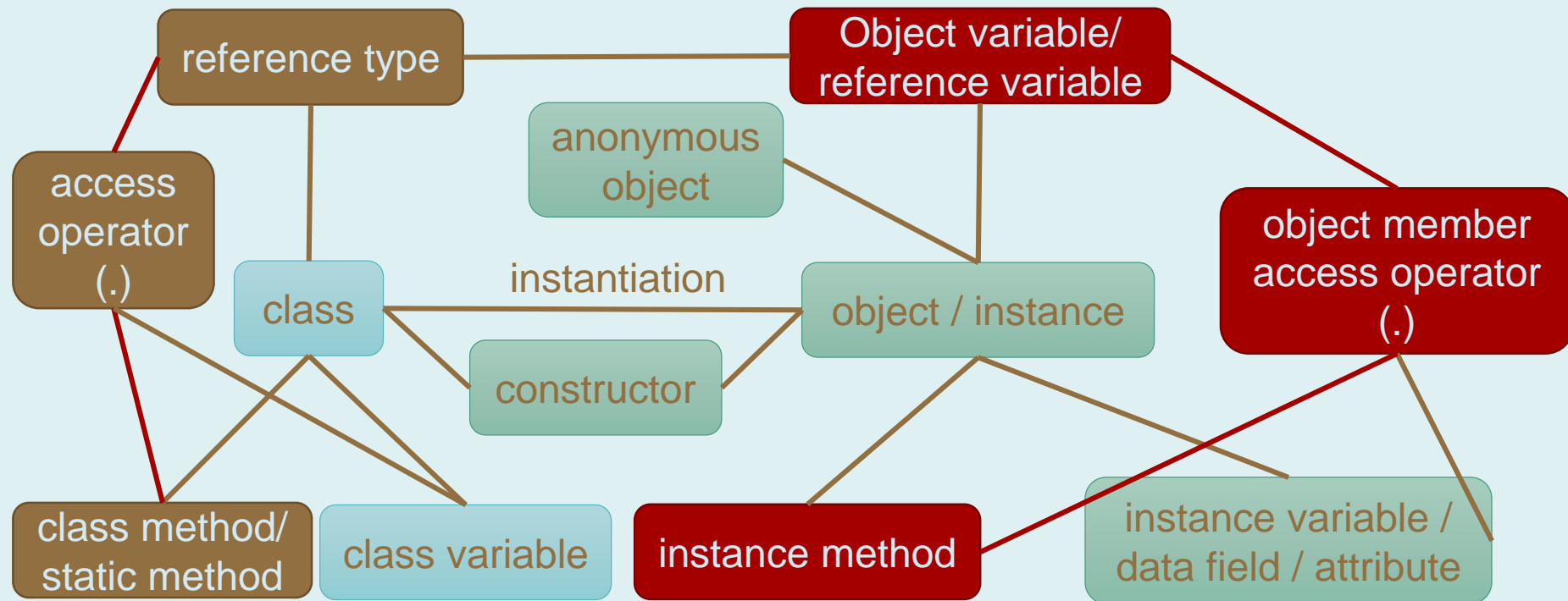
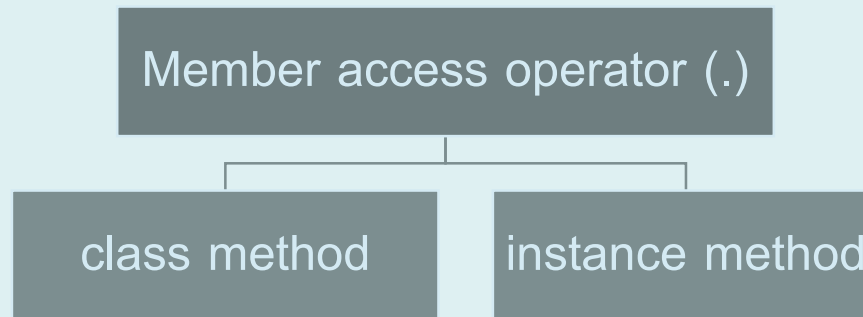


การใช้เมทอด

- เมทอดเป็นโปรแกรมย่อยที่กำหนดไว้สำหรับคลาสหนึ่ง
- เรียกใช้เมทอดโดยอ้างอิงถึงวัตถุ (instance method)
- เรียกใช้เมทอดโดยอ้างอิงถึงคลาส (class method)



การใช้เมทอด



Example

```
import java.time.LocalDate; // import the LocalDate class
import java.time.Period;    // import the Period class
public class ExampleDate {
    public static void main(String[] args) {
        int days, months;
        LocalDate today = LocalDate.now();
        LocalDate tomorrow = today.plusDays(1);

        LocalDate startDate = LocalDate.parse("2020-11-20");
        Period duration = Period.between(startDate, today);
        days = duration.getDays();
        months = duration.getMonths();
        System.out.println(today);
        System.out.println(startDate);
        System.out.println("days: "+days+", months: "+ months);
        System.out.println(Math.abs(-3.14159));
    }
}
```

การใช้เมทอดโดยอ้างอิงวัตถุ (instance method)

```
int days, months;
LocalDate today = LocalDate.now();
LocalDate tomorrow = today.plusDays(1);
LocalDate startDate = LocalDate.parse("2020-11-20");
Period duration = Period.between(startDate, today);
days = duration.getDays();
months = duration.getMonths();
System.out.println("days: "+days+", months: "+ months);
```

วัตถุในคลาส LocalDate

วัตถุในคลาส Period

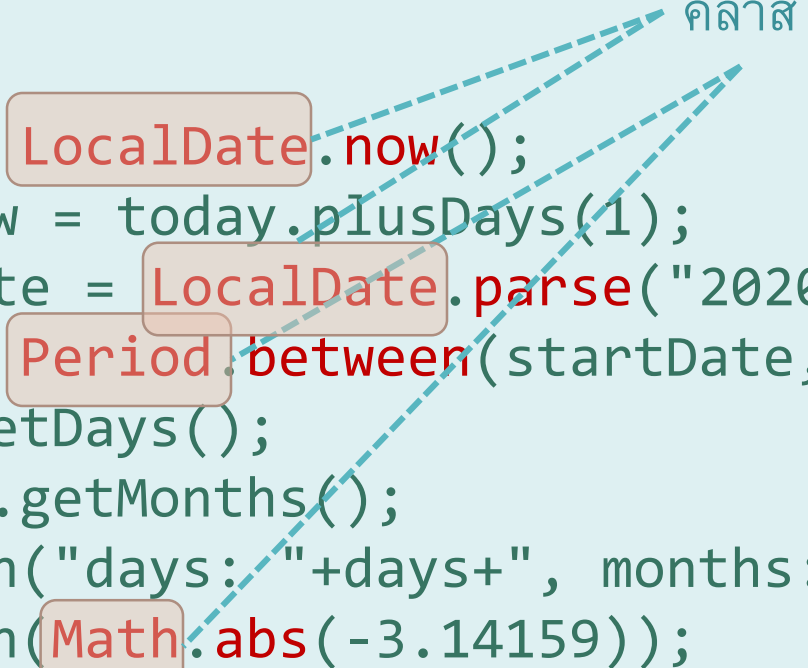
วัตถุในคลาส PrintStream

- **LocalDate** เป็น class ที่มี instance method **plusDays**
- **Period** เป็น class ที่มี instance method **getDays** และ instance method **getMonths**
- **PrintStream** เป็น class ที่มี instance method **println**

การใช้เมทอดโดยอ้างถึงคลาส (class method)

```
int days, months;
LocalDate today = LocalDate.now();
LocalDate tomorrow = today.plusDays(1);
LocalDate startDate = LocalDate.parse("2020-11-20");
Period duration = Period.between(startDate, today);
days = duration.getDays();
months = duration.getMonths();
System.out.println("days: "+days+", months: "+ months);
System.out.println(Math.abs(-3.14159));
```

คลาส



- **LocalDate** เป็น class ที่มี class method **now** และ class method **parse**
- **Period** เป็น class ที่มี class method **between**
- **Math** เป็น class ที่มี class method **abs**

พารามิเตอร์ของเมทอด

- ค่าที่ส่งเข้าไปให้เมทอด เรียกว่า พารามิเตอร์
- พารามิเตอร์ที่ใส่ในวงเล็บหลังชื่อเมทอด เรียก explicit parameter
- วัตถุที่อ้างถึงเมื่อเรียกใช้เมทอด ก็เป็นพารามิเตอร์ที่เรียกว่า implicit parameter

```
String st2 = st1.replace("o","i");  
// st1 is an implicit parameter.  
// "o","i" are explicit parameters.
```

ค่าที่เมทอดคืนมา (Return value)

- เมื่อเมทอดทำงานเสร็จแล้ว
 - อาจส่งค่าคืนมา เรียกว่า return value
 - กลับ (return) มาทำงานต่อจากคำสั่งที่เรียกเมทอด

```
String st2 = st1.replace("o","i");  
// This method replace returns a string.
```

```
int d = Math.abs(3-8);  
// This method abs returns an integer.
```

```
System.out.println("Hello");  
// This method println returns nothing.
```


เมทอดที่ overloaded

- เมทอดชื่อหนึ่งอาจรับพารามิเตอร์หลายแบบ เรียก overloaded method
- จาว่ารู้อาจจะใช้เมทอดใดจากชนิดของพารามิเตอร์ส่งไป
- อาจมีเมทอดชื่อเดียวกันในหลายคลาสได้ จาว่าบอกว่าจะใช้เมทอดในคลาสใดจาก implicit parameter

ตัวอย่าง overloaded method

- จาก <https://docs.oracle.com/javase/8/docs/api/java/lang/Math.html>

static double	abs (double a) Returns the absolute value of a double value.
static float	abs (float a) Returns the absolute value of a float value.
static int	abs (int a) Returns the absolute value of an int value.
static long	abs (long a) Returns the absolute value of a long value.

- จาก <https://docs.oracle.com/javase/7/docs/api/java/lang/String.html>

int	indexOf (int ch) Returns the index within this string of the first occurrence of the specified character.
int	indexOf (int ch, int fromIndex) Returns the index within this string of the first occurrence of the specified character, starting the search at the specified index.
int	indexOf (String str) Returns the index within this string of the first occurrence of the specified substring.
int	indexOf (String str, int fromIndex) Returns the index within this string of the first occurrence of the specified substring, starting at the specified index.

เมทอดแบบ accessor และ mutator

- Accessor method อ่านค่าของ object ที่เป็น implicit parameter แต่ไม่แก้ค่า

```
String st1 = "January";  
String st2 = st1.replace("a","i");  
// This method replace is an accessor.
```

- mutator method แก้ค่าของ object ที่เป็น implicit parameter

```
ArrayList<Integer> arrlist = new ArrayList<Integer>(5);  
arrlist.add(15);  
// This method add is a mutator.
```

การใช้คลาส Scanner

```
import java.util.Scanner;
public class ExampleScanner {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.println("Enter your name");
        String name = input.nextLine();
        System.out.println(name);

        System.out.println("Enter your age");
        int age = input.nextInt();
        System.out.println(age);

        System.out.println("Enter your grade");
        double grade = input.nextDouble();
        System.out.println(grade);

    }
}
```