Java Method

Benjamas Panyangam
Matinee Kiewkanya
Computer Science, CMU

ชนิดของ Method

1. **User Defined Methods**

   หมายถึง เมท็อดที่ผู้เขียนโปรแกรมเขียนขึ้นเอง

2. **Library Method** หมายถึง เมท็อดที่ภาษาจาวามีไว้ให้ผู้เขียนโปรแกรมสามารถเรียกใช้ได้ ซึ่งจัดเก็บใน **class** หรือ **package** ต่าง ๆ เช่น

   - **Standard input/output methods** ได้แก่ **print, println, printf, read, next, nextInt** เป็นต้น
   - **Math class methods** ได้แก่ **random, round, ceil** เป็นต้น
   - **String class methods** ได้แก่ **charAt, trim, concat** เป็นต้น

เมท็อดทางคณิตศาสตร์ (Java math library methods)

เมท็อดต่าง ๆ สำหรับใช้งานทางคณิตศาสตร์ในคลาส Math

- double ceil(double x) ปัดเศษขึ้น แล้วคืนค่าเป็นเลขจำนวนจริง
  เช่น x = Math.ceil(3.04); // x จะเก็บค่า 4.0
- double floor(double x) ปัดเศษทิ้ง แล้วคืนค่าเป็นเลขจำนวนจริง
  เช่น x = Math.floor(3.99); // x จะเก็บค่า 3.0
- double rint(double x) ปัดเศษไปเป็นจำนวนเต็มที่ใกล้ค่า x มากที่สุด แล้วคืนค่าเป็นเลขจำนวนจริง เช่น
  - x = Math.rint(3.49); // x จะเก็บค่า 3.0
  - x = Math.rint(3.57); // x จะเก็บค่า 4.0
- round ปัดเศษแล้วคืนค่าเป็นเลขจำนวนเต็ม โดย มี 2 รูปแบบ
  int round(float x) จะคืนค่า (int)Math.floor(x+0.5)
  long round(double x) จะคืนค่า (long)Math.floor(x+0.5)
  เช่น
  - x = Math.round(3.49); // x จะเก็บค่า 3
  - x = Math.round (3.57); // x จะเก็บค่า 4

เมท็อดทางคณิตศาสตร์
(Java math library methods)

- exp(double a) คืนค่า e^a
- log(double a) คืนค่า natural logarithm of a=$\log_e(a)$ = ln(a)
- log10(double a) คืนค่า $\log_{10}a$
- pow(double a, double b) คืนค่า a^b
- sqrt(double a) คืนค่า square root of a
- random() คืนค่าเลขจำนวนจริงที่สุ่มได้ โดยอยู่ในช่วง [0.0, 1.0)
- sin(double a), cos(double a), tan(double a), acos(double a), asin(double a), atan(double a)
- max(a, b), min(a, b) คืนค่าสูงสุด และต่ำสุดตามลำดับ
- abs(a) คืนค่า absolute ของ a

Main Method

ในภาษาจาวา เมท็อดหลักของทุกโปรแกรม ชื่อ main

```
class ex1{
    public static void main(String [] args)
    {
        System.out.println("Hello");
    }
    Method 1
    Method 2
    Method 3
}
```

public static : แสดงว่าเป็น Method ที่สามารถเรียกใช้งานทั่วไป
void : บอกให้ทราบว่าเมื่อเรียกใช้ Method นี้แล้วจะไม่มีการส่งค่ากลับ

รูปแบบการนิยามเมท็อด

```
[modifier]  return_type   methodName ([parameter(s)]) {
        [ body ]
   }
```

โดยที่

- **modifier** คือ คำสำคัญที่ใช้ระบุระดับการเข้าถึง
- **return_type** คือ ชนิดข้อมูลที่เมท็อดจะคืนค่ากลับ
- **methodName** คือ ชื่อเมท็อดสมาชิก
- **parameter** คือ ตัวแปรที่กำหนดขึ้นเพื่อรับข้อมูลเข้าจากผู้เรียกใช้
- **body** คือ ส่วนกำหนดการทำงานของเมท็อด ที่ประกอบด้วย ตัวแปรเฉพาะที่ (Local Variable) และคำสั่งกำหนดการทำงาน

ประเภทของ Method

- แบ่งตามการส่งค่า argument
  - แบบไม่มีการส่ง argument
  - แบบมีการส่ง argument
- แบ่งตามการส่งค่ากลับ
  - แบบไม่ส่งค่ากลับ
  - แบบส่งค่ากลับ
- แบ่งตามการเรียกใช้
  - **Static Method :** ไม่ต้องสร้าง object ก่อนการเรียกใช้ Method
  - **Instance Method :** ต้องสร้าง object ก่อนการเรียกใช้ Method

Static method แบบไม่ส่งค่ากลับและไม่มีการส่ง argument

```
class Example1{
    public static void main(String[] args)  {
        print_box( ) ;
        print_box( ) ;
        print_box( ) ;
}
    public static void print_box ( )   {
        System.out.println("****");
        System.out.println("*    *");
        System.out.println("****");
        System.out.println(); }
}
```

ผลลัพธ์

```
****
*   *
****

****
*   *
****

****
*   *
****
```

Static method แบบไม่ส่งค่ากลับและมีการส่ง argument

```
import java.util.Scanner;
class Example2{
    public static void main(String[] args) {
        int w, l;
        Scanner input = new Scanner(System.in);
        System.out.print( "Input width and length : " );
        w = input.nextInt();
        l = input.nextInt();
        findArea (w,l); // เรียกใช้เมท็อด findArea แบบส่ง argument
    }

    public static void  findArea (int a, int b )  {
        float area = a*b;    // ตัวแปร local อยู่ภายในเมท็อด findArea
        System.out.print( "Area = " + area);
    }
}
```

ผลลัพธ์

Input width and length : 20 10

Area = 200.0

Static method แบบส่งค่ากลับ และไม่มีการส่ง argument

```
import java.util.Scanner;
class Example3{
    public static void main(String[] args) {
      float area = findArea (); // เรียกใช้เมท็อด findArea
       System.out.print( "Area = " + area);
    }

    public static float  findArea ( )
    {
        int w, l;     // ตัวแปร local อยู่ภายในเมท็อด findArea
        Scanner input = new Scanner(System.in);
        System.out.print( "Input width and length : " );
        w = input.nextInt();
        l = input.nextInt();
       return (w*l);
    }
}
```

```
Input width and length : 20 10
Area = 200.0
```

Static method แบบส่งค่ากลับและมีการส่ง argument

```
import java.util.Scanner;
class Example4 {
    public static void main(String[] args) {
        int w, l;
        Scanner input = new Scanner(System.in);
        System.out.print( "Input width and length : " );
        w = input.nextInt();
        l = input.nextInt();
        float area = findArea (w,l); // เรียกใช้เมท็อด findArea
        System.out.print( "Area = " + area);
    }

    public static float  findArea ( int a,int b ) {
        return (a*b);
    }
}
```

ผลลัพธ์

```
Input width and length : 20 10
Area = 200.0
```

ขอบเขตของการใช้งานตัวแปร (Scope of variables)

- การใช้งานตัวแปรแต่ละตัวจะมีขอบเขตของการใช้งานขึ้นอยู่กับขอบเขตของ { } ในตำแหน่งที่ประกาศตัวแปรนั้น
- ตัวแปรส่วนกลาง (Global variable)
  - ตัวแปรที่ประกาศนอกทุกเมท็อด โดยทุกเมท็อดจะรู้จักตัวแปรนี้ว่ามี scope ภายในขอบเขตของ { } ของตำแหน่งที่ประกาศ
- ตัวแปรท้องถิ่น (Local variable)
  - ตัวแปรที่ประกาศในเมท็อด จะเป็นที่รู้จักภายในเมท็อดของตัวเองเท่านั้น เมื่อจบเมท็อด ตัวแปรนั้นจะถูกทำลาย
- กรณีที่ตัวแปรชื่อซ้ำกันทั้งในเมท็อดและนอกเมท็อด การอ้างถึงตัวแปรนั้นภายในเมท็อด จะหมายถึงตัวแปรที่เป็นตัวแปร local

ตัวอย่าง scope of variables

scope ของตัวแปร a และ x

```
class Example5
{
   static int a=10;

   public static void main(String[] args)  {
         int y = 30;
         x = 5;
         System.out.printf( "1:a=%d x=%d y=%d\n", a, x, y);
         a = 1;
         System.out.printf( "2:%d \n", test(20));
         System.out.printf( "3:a=%d x=%d y=%d\n", a, x, y);
   }

   static int x=50;
   public static int test(int n)
   {
        int x=2;
        return a + x + n;
   }
}
```

Scope ของตัวแปร y

Scope ของตัวแปร x และ n

ผลลัพธ์

```
1:a=10 x=5 y= 30
2:23
3:a=1 x=5 y= 30
```

ตัวอย่าง scope of variables

```
class Example6 {

 static int x=10;  //global variable


   public static void main(String[] args) {
      System.out.println("main (start) -> x : " + x );
      method1 ( );
      System.out.println("main (after method1) -> x : " + x );
      method2( x );
      System.out.println("main (after method2) -> x : " + x );
      method3 ( );
      System.out.println("main (after method3) -> x : " + x );
   }
```

ตัวอย่าง scope of variables

```
    public static void method1( )    {
            x = x + 10;
            System.out.println ( "method1 -> x : " + x );
    }

    public static void method2(int x )    {
            x = x + 10;
            System.out.println ( "method2 -> x : " + x );
    }

    public static void method3( )    {
            int x=0;
            x = x + 10;
            System.out.println ( "method3 -> x : " + x );
    }
}
```

ผลลัพธ์

```
main (start) -> x : 10
method1 -> x : 20
main (after method1) -> x : 20
method2 -> x : 30
main (after method2) -> x : 20
method3 -> x : 10
main (after method3) -> x : 20
```

ตัวแปร Global / Local

```
import java.util.Scanner;
class Example7{
   static Scanner input = new Scanner(System.in); //ตัวแปร global
   public static void main(String[] args)  {
         int score; // ตัวแปร local อยู่ภายในเมท็อด main
         char  ans;
         do {  score = getScore();
                  displayGrade(score);
                  System.out.print("\nRun Again(Y/N)? : ");
                  ans =input.next().charAt(0) ;
         } while((ans == 'y') || (ans== 'Y') );
    }

    public static int getScore( ) {
         int score; // ตัวแปร local อยู่ภายในเมท็อด getScore
         do {
                  System.out.print("Input score (0-100) : " );
                  score=input.nextInt();
         } while ((score <0 ) || (score > 100)) ;
         return(score);
   }
```

```
public static void displayGrade(int score)  {
     if (score<=50)
          System.out.println( "Your grade is U(fail)" );
     else
          System.out.println( "Your grade is S(pass)" );
   }
}
```

ผลลัพธ์

```
Input score (0-100) : 101
Input score (0-100) : 35
Your grade is U(fail)

Run Again(Y/N)? : y
Input score (0-100) : 80
Your grade is S(pass)

Run Again(Y/N)? : n
```

ตัวอย่างการใช้งาน method ที่ส่งค่ากลับเป็น Boolean

- จากตัวอย่างเดิม จะปรับแก้ส่วน main โดยให้เรียกใช้ method isYes()

```
public static void main(String[] args)  {
        int score; // ตัวแปร local อยู่ภายในเมท็อด main
        char  ans;
        do {    score = getScore();
                displayGrade(score);
                System.out.print("\nRun Again(Y/N)? : ");
                ans =input.next().charAt(0) ;
        } while( isYes(ans) );
}

public static boolean isYes(char ch) {
      if ((ch == 'y') || (ch == 'Y') )
            return(true);
      else   return(false);
}
```