

Encapsulation & Information Hiding

Maharakham University

อาจารย์พชร พฤกษ์ศรี
สาขาวิชาวิทยาการคอมพิวเตอร์ คณะวิทยาการสารสนเทศ
Email: potchara.p@msu.ac.th

1

การสื่อสารระหว่าง Object

- การเขียนโปรแกรมแบบ OOP นั้น เมื่อมีการสร้าง Object แล้วต้องมีการทำงานร่วมกันระหว่าง Object และต้องมีการสื่อสารกันระหว่าง Object
- message** คือ คำสั่งในการร้องขอหรือกระทำไปยัง Object อื่น
- การรับส่ง message จะทำโดยการเรียกใช้เมธอด

Maharakham University

2

Encapsulation (การห่อหุ้ม)

การเข้าถึง attributes โดยตรง ตามหลัก OOP แล้วไม่ควรทำ

```

3 public class Account {
4     String accountID = "123-4-56789-0";
5     String accountName = "Peter";
6     double balance = 10000;
7 }
    
```

```

3 public class Customer {
4     public static void main(String[] args) {
5         Account acc1 = new Account();
6         System.out.println("Account ID: " + acc1.accountID);
7         System.out.println("Account Name: " + acc1.accountName);
8         System.out.println("Balance: " + acc1.balance + " Baht");
9     }
10 }
    
```

Problems @ Javadoc Declaration Console

<terminated> Customer [Java Application] C:\Program Files\

Account ID: 123-4-56789-0
Account Name: Peter
Balance: 10000.0 Baht

Maharakham University

3

4

5

6

Mutator methods

- Method แบบ Setter จะใช้ในการกำหนดค่าของ Attributes โดยปกติ Method นี้จะไม่มีการส่งค่ากลับคืน ดังนั้นส่วนใหญ่จะมี return type เป็น void และมักขึ้นต้นชื่อ Method ว่า **set**
- Method แบบ Getter จะใช้ในการเรียกค่าของ Attributes โดยปกติ Method นี้จะมีการส่งค่ากลับคืน ดังนั้นจะมี return type ที่ไม่ใช่ void และมักขึ้นต้นชื่อ Method ว่า **get**
 - แต่ถ้า Getter Method ที่มี return type เป็น boolean หมายถึง Getter Method ที่ใช้ตรวจสอบค่าใดค่าหนึ่ง มักจะขึ้นต้นชื่อ Method ว่า **is**

Mahasarakham University



7

Mutator methods

```

1 package ex02_01;
2
3 class TestModifier
4 {
5     public static void main(String[] args) {
6         Dog dog = new Dog();
7         dog.setDogDetail("DugDig", "Bangkaew", 1);
8         String text = dog.getDogDetail();
9         System.out.println(text);
10    }
11 }
12 class Dog {
13     private String name = "";
14     private String type = "";
15     private int age = 0;
16     public void setDogDetail(String dogName, String dogType, int dogAge)
17     {
18         name = dogName;
19         type = dogType;
20         age = dogAge;
21     }
22     public String getDogDetail()
23     {
24         String text = name +
25             " is a " + type + " dog " +
26             age + " years old";
27         return text;
28     }
29 }

```

Setter Method

Getter Method

University

8

this

- คำว่า "this" เป็น keyword
- this ในจาวาหมายถึง Object ของตัวเอง มักใช้ this กับสิ่งที่มีชื่อซ้ำกับ Attribute หรือ Method ของ Object
- การใช้ this อาจทำให้เกิดการสับสนในการในการเขียนโปรแกรมได้ เช่นการใช้เพื่อเรียก attributes ที่ชื่อเหมือนกัน ดังนั้น ควรตั้งชื่อให้สื่อความหมายและไม่ซ้ำกันจะดีกว่า

Mahasarakham University



9

Heart of the Northeast

this

```

1 package ex02_01;
2
3 public class DemoThis {
4     // Attribute
5     String name = "Michael";
6     public static void main(String[] args) {
7         DemoThis demo = new DemoThis();
8         demo.printName();
9     }
10    void printName()
11    {
12        // local variable
13        String name = "Peter";
14        System.out.println("Your name is " + name);
15    }
16 }

```

Problems @ Javadoc Declaration Console

<terminated> Customer [Java Application] C:\Program Files\Ja
Your name is Peter

ถ้าจาวาพบชื่อซ้ำกัน จะเลือกใช้ตัวที่ใกล้ที่สุดเสมอ

Mahosarakham University

10

Heart of the Northeast

this

```

1 package ex02_01;
2
3 public class DemoThis {
4     // Attribute
5     String name = "Michael";
6     public static void main(String[] args) {
7         DemoThis demo = new DemoThis();
8         demo.printName();
9     }
10    void printName()
11    {
12        // local variable
13        String name = "Peter";
14        System.out.println("Your name is " + this.name);
15    }
16 }

```

Problems @ Javadoc Declaration Console

<terminated> Customer [Java Application] C:\Program Files\Ja
Your name is Michael

this จะเป็นตัวอ้างถึง object นั้นคือแสดง name ที่เป็น attribute ของ object

Mahosarakham University

11

Heart of the Northeast

Information Hiding

- การซ่อนข้อมูล (Information Hiding) เป็นการปกป้องข้อมูลของ Object ใด ไม่ให้เข้าถึงได้โดยผิดวิธี และการซ่อนข้อมูลยังเข้ามาช่วยแก้ปัญหาการเปลี่ยนแปลงข้อมูลที่ไม่สัมพันธ์กันได้
- หลักการสำคัญ
 - กำหนดระดับการเข้าถึงให้กับ Attributes และ Methods ของ Object ไม่อนุญาตให้เข้าถึงได้โดยตรง
 - การใช้ Mutator methods (setter และ getter) เป็นตัวควบคุมการเข้าถึง attributes

Mahosarakham University

12

Heart of the Northeast

Information Hiding

```

3 public class Account {
4     String accountID = "123-4-56789-0";
5     String accountName = "Peter";
6     double balance = 10000;
7
8     // Getter method
9     void getDetail() {
10        System.out.println("Account ID: " + accountID);
11        System.out.println("Account Name: " + accountName);
12        System.out.println("Balance: " + balance + " Baht");
13    }
14 }

```

สร้าง getter method

```

3 public class Customer {
4     public static void main(String[] args) {
5         Account acc1 = new Account();
6         acc1.getDetail();
7     }
8 }

```

เข้าถึง attributes ผ่าน getter method

Problems @ Javadoc Declaration Console

<terminated> Customer [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\java.exe

Account ID: 123-4-56789-0
Account Name: Peter
Balance: 10000.0 Baht

13

Heart of the Northeast

Information Hiding

```

3 public class Account {
4     String accountNumber = "123-4-56789-0";
5     String accountName = "Peter";
6     double balance = 10000;
7
8     // Getter method
9     void getDetail() {
10        System.out.println("Account ID: " + accountNumber);
11        System.out.println("Account Name: " + accountName);
12        System.out.println("Balance: " + balance + " Baht");
13    }
14 }

```

เปลี่ยนแปลง attribute ของ object

```

3 public class Customer {
4     public static void main(String[] args) {
5         Account acc1 = new Account();
6         acc1.getDetail();
7     }
8 }

```

ไม่มีการเปลี่ยนแปลงค่าตั้ง

Problems @ Javadoc Declaration Console

<terminated> Customer [Java Application] C:\Program Files\Java\jdk-11.0.2\bin\java.exe

Account ID: 123-4-56789-0
Account Name: Peter
Balance: 10000.0 Baht

14

Heart of the Northeast

ข้อดีของ Encapsulation

- การซ่อนข้อมูล (Information Hiding) ทำให้ Object สามารถติดต่อกับ Object ภายนอก ผ่านทางช่องทางที่กำหนด (method) เท่านั้น ซึ่งถ้ามีการเปลี่ยนแปลง Attributes หรือ Method ใน Object จะไม่มีผลกระทบต่อ Object ภายนอกที่มาเรียกใช้
- ข้อสังเกต ถึงแม้ว่าจะเขียนโปรแกรมเพื่อให้เข้าถึงข้อมูลโดยผ่านเมธอดแล้ว แต่ก็ยังสามารถเข้าถึงข้อมูลได้โดยตรงหากรู้รายละเอียดของ attribute นั้นๆ

Mahosarakham University

15

Heart of the Northeast

Information Hiding

```

3 public class Customer {
4     public static void main(String[] args) {
5         Account acc1 = new Account();
6         acc1.getDetail();
7         System.out.println(acc1.accountName);
8     }
9 }

```

Problems Javadoc Declaration Console

<terminated> Customer [Java Application] C:\Program Files\Java\jdk-9.0.4\bin\java.exe
 Account ID: 123-4-56789-0
 Account Name: Peter
 Balance: 10000.0 Baht
 Peter

Mahosarakham University

16

Heart of the Northeast

Access Modifier

- จากปัญหาการเข้าถึงข้อมูลที่เกิดขึ้น การเขียนโปรแกรมแบบ OOP จึงมีการออกแบบให้มีการควบคุมการเข้าถึงข้อมูลได้ในระดับที่แตกต่างกัน โดยมีการ keyword ที่ใช้อยู่ 3 ตัว (4 ระดับ)
 - Public
 - Protected
 - Private
- ระดับการควบคุมการเข้าถึงข้อมูล เรียกว่า Accessibility หรือ Visibility โดยมีการใช้ Access Modifier เป็นตัวควบคุม

Mahosarakham University

17

Heart of the Northeast

Access Modifier

Public

- มี keyword "public"
- Attributes หรือ Methods ของ Class จะถูกเปิดเผยหรือมองเห็นได้และถูกเข้าถึงได้โดยตรงจากภายนอก ไม่มีการปกปิดใดๆ ทั้งสิ้น
- แทนสัญลักษณ์ใน Diagram ด้วยเครื่องหมาย (+)

Mahosarakham University

18


Heart of the Northeast

Access Modifier

Protected

- มี keyword “protected”
- Attributes หรือ Methods ของ Class จะไม่ถูกเปิดเผยหรือมองเห็นได้และถูกเข้าถึงได้โดยตรงจากภายนอก สามารถเข้าถึงได้จากภายในคลาสเดียวกัน, Subclass เท่านั้น
- แทนสัญลักษณ์ใน Diagram ด้วยเครื่องหมาย (#)

Maharakham University



19


Heart of the Northeast

Access Modifier

Default หรือ Package

- ไม่ต้องมี keyword
- Attributes หรือ Methods ของ Class จะไม่ถูกเปิดเผยหรือมองเห็นได้และถูกเข้าถึงได้โดยตรงจากภายนอก สามารถเข้าถึงได้จากภายในคลาส และ Package เดียวกัน เท่านั้น
- ไม่มีสัญลักษณ์

Maharakham University



20


Heart of the Northeast

Access Modifier

Private

- มี keyword “private”
- Attributes หรือ Methods ของ Class จะไม่ถูกเปิดเผยหรือมองเห็นได้และถูกเข้าถึงได้โดยตรงจากภายนอก สามารถเข้าถึงได้จากภายในคลาสเดียวกันเท่านั้น
- แทนสัญลักษณ์ใน Diagram ด้วยเครื่องหมาย (-)

Maharakham University



21

Heart of
the Northeast

Access Modifier

- ระดับการเข้าถึง Attributes และ Methods ในแบบต่างๆ

Modifier	คลาสเดียวกัน	คลาสที่อยู่ในแพคเกจเดียวกัน	คลาสที่เป็น subclass	คลาสใดๆ
public (+)	✓	✓	✓	✓
protected (#)	✓	✓	✓	
default/package	✓	✓		
private (-)	✓			

Mahasarakham University

22

Heart of
the Northeast

ตัวอย่าง public modifier

```

1 package ex02_01;
2
3 class TestModifier
4 {
5     public static void main(String[] args) {
6         Dog dog = new Dog();
7         dog.name = "DugDig";
8         dog.type = "BangKaew";
9         dog.age = 1;
10
11         System.out.println(dog.name +
12             " is a " + dog.type + " dog " +
13             dog.age + " years old");
14     }
15 }
16 class Dog {
17     public String name = "";
18     public String type = "";
19     public int age = 0;
20 }

```

สามารถเข้าถึง Attributes ได้โดยตรง เพราะไม่มีการปิดใดๆ

Attributes ถูกกำหนดเป็น public

Problems @ Javadoc Declaration Console

<terminated> Customer [Java Application] C:\Program Files\Java\JugDig is a BangKaew dog 1 years old

Mahasarakham University

23

Heart of
the Northeast

ตัวอย่าง default modifier

```

1 package ex02_01;
2
3 class TestModifier
4 {
5     public static void main(String[] args) {
6         Dog dog = new Dog();
7         dog.name = "DugDig";
8         dog.type = "BangKaew";
9         dog.age = 1;
10
11         System.out.println(dog.name +
12             " is a " + dog.type + " dog " +
13             dog.age + " years old");
14     }
15 }
16 class Dog {
17     String name = "";
18     String type = "";
19     int age = 0;
20 }

```

สามารถเข้าถึง Attributes ได้โดยตรง เพราะอยู่ใน package เดียวกันเข้าถึงกันได้

Attributes ถูกกำหนดเป็น default

Problems @ Javadoc Declaration Console

<terminated> Customer [Java Application] C:\Program Files\Java\JugDig is a BangKaew dog 1 years old

Mahasarakham University

24

ตัวอย่าง private modifier

```

1 package ex02_01;
2
3 class TestModifier
4 {
5     public static void main(String[] args) {
6         Dog dog = new Dog();
7         dog.name = "DugDig";
8         dog.type = "BangKaew";
9         dog.age = 1;
10
11         System.out.println(dog.name +
12             " is a " + dog.type + " dog " +
13             dog.age + " years old");
14     }
15 }
16 class Dog {
17     private String name = "";
18     private String type = "";
19     private int age = 0;
20 }

```

ไม่สามารถเข้าถึง Attributes ใดๆ เพราะปกป้องให้เข้าถึงเฉพาะใน class ตนเองเท่านั้น

Attributes ถูกกำหนดเป็น private

Heart of the Northeast

25

Accessibility with Setter and Getter methods

```

1 package ex02_01;
2
3 class TestModifier
4 {
5     public static void main(String[] args) {
6         Dog dog = new Dog();
7         dog.setDogDetail("DugDig", "BangKaew", 1);
8         String text = dog.getDogDetail();
9         System.out.println(text);
10     }
11 }
12 class Dog {
13     private String name = "";
14     private String type = "";
15     private int age = 0;
16     public void setDogDetail(String dogName, String dogType, int dogAge)
17     {
18         name = dogName;
19         type = dogType;
20         age = dogAge;
21     }
22     public String getDogDetail()
23     {
24         String text = name +
25             " is a " + type + " dog " +
26             age + " years old";
27         return text;
28     }
29 }

```

ทำงานกับ attributes ผ่าน methods

Attribute ที่กำหนดเป็น private

สร้าง Method เพื่อกำหนดค่าข้อมูล (setter)

สร้าง Method เพื่อเรียกค่าข้อมูล (getter)

Heart of the Northeast

26

สมาชิกของคลาส (Class Member)

class Dog

+ setDogDetail()

- name
- type
- age

+ getDogDetail()

Encapsulation

Mahosarakham University

27

Heart of the Northeast

การส่งผ่าน argument

- เรียกอีกอย่างว่า parameter passing
- หมายถึงการส่งข้อมูลเข้าหรือดึงออกจาก Method
- ในการส่งผ่านข้อมูลจะต้องส่งผ่าน argument หรือ parameter เข้าไปหรือส่งออกจาก Method
- argument แบ่งได้เป็น 2 ประเภท
 - argument ที่มีชนิดข้อมูลแบบพื้นฐาน (pass by value)
 - argument ที่มีชนิดข้อมูลแบบอ้างอิง (pass by reference)

Mahosarakham University

www.mahosarakham.ac.th

28

Heart of the Northeast

Pass by Value

```

1 package ex02_01;
2
3 public class PassByValue {
4
5     public static void main(String[] args) {
6         int i = 5, j = 10, k = 0;
7         PassByValue passDemo = new PassByValue();
8         k = passDemo.add(i, j);
9         System.out.println("Result of " + i +
10             " + " + j + " = " + k );
11     }
12     int add(int x, int y)
13     {
14         return x+y;
15     }
16 }

```

x = 5 และ y = 10

Problems Javadoc Declaration Console

<terminated> Customer [Java Application] C:\Program Files\Java
 Result of 5 + 10 = 15

Mahosarakham University

www.mahosarakham.ac.th

29

Heart of the Northeast

Pass by Value

```

1 package ex02_01;
2
3 public class PassByValue {
4
5     public static void main(String[] args) {
6         int i = 5, j = 10, k = 0;
7         PassByValue passDemo = new PassByValue();
8         k = passDemo.add(i, j);
9         System.out.println("Result of " + i +
10             " + " + j + " = " + k );
11     }
12     int add(int x, int y)
13     {
14         return x+y;
15     }
16 }

```

Stack Memory

y	10
x	5
k	0 → 15
j	10
i	5

Mahosarakham University

www.mahosarakham.ac.th

30

Pass by Reference

```

1 package ex02_01;
2
3 public class PassByRef {
4
5     public static void main(String[] args) {
6         PassByRef passDemo = new PassByRef();
7         Student s1 = new Student();
8         passDemo.addNewStudent(s1);
9         System.out.println(s1.id + " " + s1.name
10             + " " + s1.gpa);
11     }
12     void addNewStudent(Student object1)
13     {
14         object1.id = "1111";
15         object1.name = "Peter";
16         object1.gpa = 4.00;
17     }
18 }
19
20 class Student
21 {
22     String id = "0000";
23     String name = "Plank";
24     double gpa = 0.00;
25 }

```

Heap Memory
@0xdae25
0000Plank0.00

Stack Memory
object1 @0xdae25
s1 @0xdae25

Console
<terminated> Customer [Java Application] C:\Program Files\U
1111 Peter 4.0

31

Object Copy

- คือการคัดลอก Object ไปเป็น Object ใหม่โดยใช้เครื่องหมาย =
- เช่น ObjectA = ObjectB;
- ผลลัพธ์ที่ได้คือ ObjectA จะเก็บตำแหน่งเดียวกับ ObjectB

32

Shadow Copy

shadow copy
ObjectA = ObjectB.

A และ B จะชี้ไปที่ตำแหน่งในหน่วยความจำเดียวกัน

data

33

การเปรียบเทียบ Object

- Object มีการเก็บข้อมูลแบบอ้างอิง
- การเปรียบเทียบ Object สามารถเปรียบเทียบได้ 2 อย่าง
- เปรียบเทียบ ข้อมูลของ Object
- ต้องใช้ Method equals();
- เปรียบเทียบ ตำแหน่งในหน่วยความจำของ Object
-ต้องใช้ เครื่องหมาย ==

Maharakham
University

34

การเปรียบเทียบ Object

```

1 package ex02_01;
2
3 public class ObjectCompare {
4
5     public static void main(String[] args) {
6         Student student1 = new Student();
7         Student student2 = new Student();
8         System.out.println(student1 == student2);
9         System.out.println("student1 = " + student1);
10        System.out.println("student2 = " + student2);
11    }
12 }
13
14 class Student
15 {
16     String id = "0000";
17     String name = "Blank";
18     double gpa = 0.00;
19 }

```

Problems Javadoc Declaration Console
 <terminated> ObjectCompare [Java Application] C:\Program F
 false
 student1 = ex02_01.Student@2a139a55
 student2 = ex02_01.Student@15db9742

ผลลัพธ์เป็น false เพราะเมื่อสร้าง object (new)
 จาวาจะไปจองพื้นที่ในหน่วยความจำใหม่เสมอ

Maharakham
University

35

Overloading

- Overload ในภาษาจาวาหมายถึงความซ้ำกัน
- Method Overloading คือ การที่สามารถมี Method ที่มีชื่อเหมือนกันได้ จาวาจะแยกความแตกต่างของแต่ละ Method โดยดูจาก จำนวน และ ชนิดข้อมูล ของพารามิเตอร์

Maharakham
University

36

```

3 public class Overloading {
4
5     public static void main(String[] args) {
6
7
8     }
9     void printLine()
10    {
11        for (int i=0;i<20;i++){
12            System.out.print("-");
13        }
14        System.out.println();
15    }
16    void printLine(char ch)
17    {
18        for (int i=0;i<20;i++){
19            System.out.print(ch);
20        }
21        System.out.println();
22    }
23    void printLine(char ch, int num)
24    {
25        for (int i=0;i<num;i++){
26            System.out.print(ch);
27        }
28        System.out.println();
29    }
30    void printLine(String str)
31    {
32        for (int i=0;i<20;i++){
33            System.out.print(str);
34        }
35        System.out.println();
36    }
37 }

```

Heart of the Northeast

Maharakham University

public static void main(String[] args) {
Overloading ov = new Overloading();
ov.printLine();
ov.printLine("-");
ov.printLine("-", 20);
ov.printLine("-");
}

Problems Javadoc Declaration Console
<terminated> Overloading [Java Application] C:\Program Files

37

Overloading

- อย่าเข้าใจผิด!!!
- ถ้าจำนวนและชนิดพารามิเตอร์ของ Method Overloading เหมือนกัน
 - เปลี่ยนชื่อของพารามิเตอร์ ก็ไม่ถือว่าเป็น Overload
 - เปลี่ยน return type ก็ไม่ถือว่าเป็น Overload

Heart of the Northeast

Maharakham University

38

```

50 public static void main(String[] args) {
51     Overloading ov = new Overloading();
52     ov.printLine();
53     ov.printLine("-");
54     ov.printLine("-", 20);
55     ov.printLine("-");
56 }
57 void printLine()
58 {
59     for (int i=0;i<20;i++){
60         System.out.print("-");
61     }
62     System.out.println();
63 }
64 void printLine(char ch)
65 {
66     for (int i=0;i<20;i++){
67         System.out.print(ch);
68     }
69     System.out.println();
70 }
71 void printLine(char ch, int num)
72 {
73     for (int i=0;i<num;i++){
74         System.out.print(ch);
75     }
76     System.out.println();
77 }
78 void printLine(String str)
79 {
80     for (int i=0;i<20;i++){
81         System.out.print(str);
82     }
83     System.out.println();
84 }
85 }

```

Heart of the Northeast

Maharakham University

public static void main(String[] args) {
Overloading ov = new Overloading();
ov.printLine();
ov.printLine("-");
ov.printLine("-", 20);
ov.printLine("-");
}
void printLine()
{
for (int i=0;i<20;i++){
System.out.print("-");
}
System.out.println();
}
void printLine(char ch)
{
for (int i=0;i<20;i++){
System.out.print(ch);
}
System.out.println();
}
void printLine(char ch, int num)
{
for (int i=0;i<num;i++){
System.out.print(ch);
}
System.out.println();
}
void printLine(String str)
{
for (int i=0;i<20;i++){
System.out.print(str);
}
System.out.println();
}
}

เปลี่ยนชื่อตัวแปรไม่ได้หมายถึงการทำ overloading

39

Heart of the Northeast

Constructor overloading

```

3 public class TestAccountCon {
4
5     public static void main(String[] args) {
6         AccountCon acc1 = new AccountCon();
7         System.out.println("Account ID: " + acc1.accountID);
8         System.out.println("Account Name: " + acc1.accountName);
9         System.out.println("Account Balance: " + acc1.getBalance());
10
11         AccountCon acc2 = new AccountCon("111-1-1111-1", "Peter", 30000);
12         System.out.println("Account ID: " + acc2.accountID);
13         System.out.println("Account Name: " + acc2.accountName);
14         System.out.println("Account Balance: " + acc2.getBalance());
15     }
16 }

```

ทำงานที่ Constructor ตัวที่ 1

ทำงานที่ Constructor ตัวที่ 2

```

<terminated> TestAccountCon [Java Application] C:\Program F
Account ID: 000-0-00000-0
Account Name: Blank
Account Balance: 0.0
Account ID: 111-1-1111-1
Account Name: Peter
Account Balance: 30000.0

```

Mahasarakham University

40

Heart of the Northeast

Method toString()

- ในการเขียนโปรแกรมที่มีการสร้าง Object ใหม่ โดยใช้คำสั่ง new จะมี Method มาตรฐานที่สำคัญหลายตัว

```

public static void main(String[] args) {
    Student s1 = new Student();
    s1.
    Str
    Syst
}

```

id: String - Student
name: String - Student
equals(Object obj): boolean - Object
getClass(): Class<?> - Object
hashCode(): int - Object
notify(): void - Object
notifyAll(): void - Object
toString(): String - Student
wait(): void - Object
wait(long timeout): void - Object
wait(long timeout, int nanos): void - Object

Press 'Ctrl+Space' to show Template Proposals

Mahasarakham University

41

Heart of the Northeast

Method toString()

- Method toString();
- เป็น Method มาตรฐานที่ใช้ในการแปลงค่าของ Object ให้เป็นข้อมูลที่มีชนิดเป็น String
- หากมีการเรียก Object ตรงๆ Object จะทำงานที่ method toString();
- สามารถเปลี่ยนแปลงหรือแก้ไข method มาตรฐาน ได้โดยการทำ overriding (สืบทอดหน้า)

Mahasarakham University

42

ตัวอย่าง Method toString()

```
1 package ex02_01;
2
3 public class OverridingTest {
4
5     public static void main(String[] args) {
6         Student s1 = new Student();
7         System.out.println(s1);
8     }
9 }
10
11 class Student
12 {
13     String id = "0000";
14     String name = "Blank";
15     double gpa = 0.00;
16     @Override
17     public String toString() {
18         return "ID: " + id + " Name: " + name + " GPA: " + gpa;
19     }
20 }
```

Problems @ Javadoc Declaration Console

<terminated> OverridingTest [Java Application] C:\Program Fil
ID: 0000 Name: Blank GPA: 0.0

Mahasarakham University
