KMITL

PROGRAMMING FUNDAMENTAL

Prachya Sangkharat (!CE)
Microsoft Learn Student Ambassadors
Department of Computer Science, KMITL

KMITL

Built-in Class

KMITL

String

String

KMITL

Method	ค่าที่ได้
`length()`	ความยาวของข้อความ (จำนวนตัวอักษร)
`charAt(n)`	ตัวอักษรตำแหน่งที่ n
`replace(char1, char2)`	แทนที่ char1 ที่พบทั้งข้อความด้วย char2
`toUpperCase()`	เปลี่ยนเป็นตัวพิมพ์ใหญ่ทั้งหมด
`trim()`	ลบช่องว่างด้านหน้าและด้านหลังข้อความ
`substring(beginIndex, endIndex)`	ตัดข้อความตั้งแต่ตัวที่ beginIndex จนถึงตัวก่อนหน้า endIndex
`equals(str)`	ข้อความเหมือนกันหรือไม่
`indexOf(char)`	อักขระนี้อยู่ตำแหน่งใด (ถ้ามีเยอะ จะเอาตำแหน่งแรกที่พบ)

String - ตัวอย่างการนำไปใช้

KMITL

```
String pokemon = "Pikachu";
System.out.println(pokemon.length()); // 7
// System.out.println(pokemon.charAt(9));
// ➔ Index 9 out of bounds for length 7
System.out.println(pokemon.charAt(1)); // i
System.out.println(pokemon.substring(2, 4)); // ka
String p1 = "Fushigidane";
String p2 = "Fushigidane";
System.out.println(p1.equals(p2)); //true
System.out.println(p1.indexOf('i')); //4
```

Math

Math

KMITL

Method	ค่าที่ได้
`max(int1, int2)`	ค่ามากที่สุดระหว่าง int1 และ int2
`min(int1, int2)`	ค่าน้อยที่สุดระหว่าง int1 และ int2
`sqrt(int1)`	กรณฑ์ที่สองของ int1
`abs(int1)`	ค่าสัมบูรณ์ของ int1
`random()`	สุ่มจำนวนระหว่าง 0.0 ถึง 1.0
`pow(int1, int2)`	ผลลัพธ์ของ int1 ยกกำลัง int2

Exception

Exception

KMITL

- ประเภทของ Error
- Try - Catch

ประเภทของ Error

ประเภทของ Error

KMITL

- **Syntax Error**

เขียนคำสั่งผิดไวยากรณ์ หรือรูปแบบไม่ถูก

เช่น System.out.println(“Hello World”)

- **Logical Error**

เขียนคำสั่งถูก แต่รันโปรแกรมแล้วได้ผลไม่ตรงตามที่ต้องการ

- **Runtime Error**

Error ที่เกิดตอนรันโปรแกรม (JVM จับได้)

เช่น int a = 4/0;

Try - Catch

Try - Catch

KMITL

- เมื่อเกิด Error จะให้ทำอะไรต่อ? (หยุดโปรแกรมเลย หรือทำอะไรบางอย่าง)
- try{ } จะเป็นบล็อกที่ให้ทำคำสั่งบางอย่างที่ต้องการจะจัดการในกรณีที่เกิด Error
- catch(Exception e){ } เมื่อเกิด Error ขึ้นมา จะให้ทำคำสั่งอะไรต่อ

```
public class TryCatchSlide13 {
	public static void main(String[] args) {
		int[] myNumbers = { 1, 2, 3 };
		System.out.println(myNumbers[10]); //error because size of array = 3
	}
}
```

Try - Catch

KMITL

- เมื่อเกิด Error จะให้ทำอะไรต่อ? (หยุดโปรแกรมเลย หรือทำอะไรบางอย่าง)
- try{ } จะเป็นบล็อกที่ให้ทำคำสั่งบางอย่างที่ต้องการจะจัดการในกรณีที่เกิด Error
- catch(Exception e){ } เมื่อเกิด Error ขึ้นมา จะให้ทำคำสั่งอะไรต่อ

```
public class TryCatchSlide14 {
    public static void main(String[] args) {
        try {
            int[] myNumbers = { 1, 2, 3 };
            System.out.println(myNumbers[10]);
            System.out.println("Print Here 1"); //not worked
        } catch (Exception e) {
                System.out.println("error because size of array = 3");
        }
        System.out.println("Print Here 2"); //worked
    }
}
```

Finally

KMITL

- finally {} คำสั่งที่อยู่ในบล็อกนี้ ไม่ว่าจะเกิด Exception ใด ๆ ก็จะถูกทำงานหลังจัดการ Exception เสร็จเสมอ

```
public class TryCatchSlide15 {
    public static void main(String[] args) {
        try {
            int[] myNumbers = { 1, 2, 3 };
            System.out.println(myNumbers[10]);
            } catch (Exception e) {
                System.out.println("error because size of array = 3");
        } finally {
                System.out.println("Print Here 1"); // worked
        }
        System.out.println("Print Here 2"); // worked
    }
}
```

สามารถ Catch ได้หลาย Exception

KMITL

- เปลี่ยน Exception เป็นประเภท Exception ที่ต้องการจัดการ

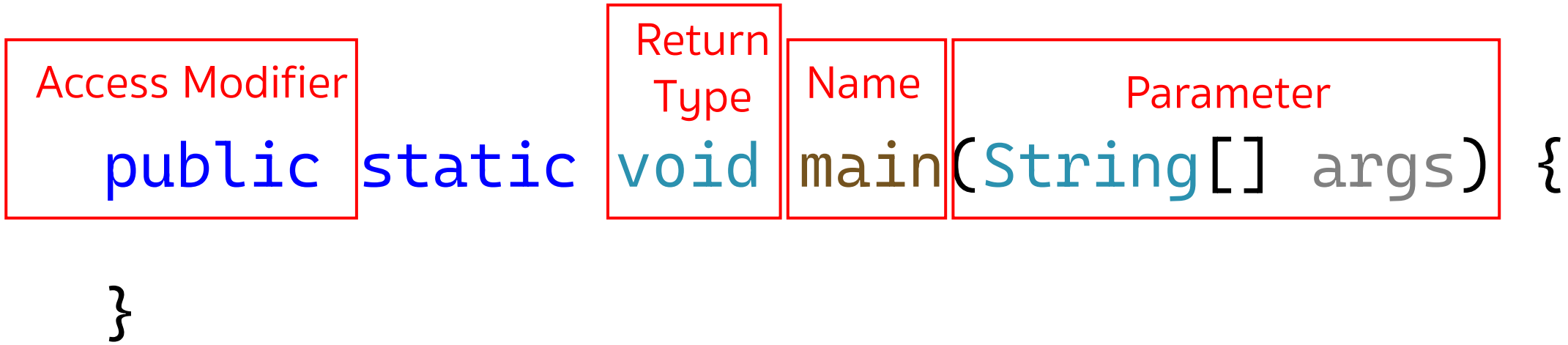
```
public class TryCatchSlide15 {
    public static void main(String[] args) {
        try {
            int[] myNumbers = { 1, 2, 3 };
            System.out.println(myNumbers[10]);
            System.out.println(myNumbers[0]/0); //Not worked

        } catch (ArithmeticException e) {
                System.out.println("cannot divide by 0");
        } catch (IndexOutOfBoundsException e) {
                System.out.println("Index out of bound");
        } finally {
                System.out.println("Print Here 1"); // worked
        }
        System.out.println("Print Here 2"); // worked
    }
}
```

Method (1)

Method

KMITL

- กลุ่มของคำสั่ง จะทำงานก็ต่อเมื่อเรียกใช้กลุ่มคำสั่งนั้น
- Method ที่เห็นกันบ่อย ๆ ➔ main()
- ทุกโปรแกรมจะมองหา Method main ก่อนเพื่อรันโปรแกรม
- ส่วนประกอบของ Method



Access Modifier

- public ➔ ทุกคลาสสามารถมองเห็นได้
- private ➔ มองเห็นได้แค่ภายในคลาสเดียวกัน
- protected ➔ มองเห็นได้เฉพาะภายใน Class และ Class ที่สืบทอดมา

Return Type

KMITL

- ชนิดของข้อมูลที่จะคืนค่าเมื่อจบการทำงาน Method
- หากไม่มีการคืนค่า จะใช้เป็น void

Parameter

- ตัวแปรที่จะรับเข้ามาทำงานใน Method
- ประกาศตัวแปร Parameter เหมือนประกาศตัวแปรทั่วไป
- เป็นแบบ Primitive หรือ Non-primitive ก็ได้
- มีกี่ตัวก็ได้

Good Programming

Good Programming

KMITL

- การเขียนโปรแกรมที่ดี ต้องแยก Method ตามการทำงานให้ชัดเจน

```
public class GoodProgrammingSlide23 {
    public static void main(String[] args) {
        int a = 3;
        int b = 10;

        // plus
        System.out.println("\na + b");
        int total = a + b;
        System.out.println(total);

        // loop a to b
        System.out.println("\nLoop a to b");
        for (int i = a; i <= b; i++) {
            System.out.print(i + " ");
        }

        // print char with unicode from a to b
        System.err.println("\nLoop a + 12 to b + 12");
        for (int i = a; i <= b; i++) {
            System.out.print(i + 12 + " ");
        }
    }
}
```

Good Programming

KMITL

- การเขียนโปรแกรมที่ดี ต้องแยก Method ตามการทำงานให้ชัดเจน

```
static void plus(int a, int b) {
    System.out.println("\na + b");
    int total = a + b;
    System.out.print(total);
}

static void loopAToB(int a, int b) {
    System.out.println("\nLoop a to b");
    for (int i = a; i <= b; i++) {
    System.out.print(i + " ");
    }
}
```

Good Programming

KMITL

- การเขียนโปรแกรมที่ดี ต้องแยก Method ตามการทำงานให้ชัดเจน

```
static void loopAPlus12ToBPlus12(int a, int b) {
    System.err.println("\nLoop a + 12 to b + 12");
    for (int i = a; i <= b; i++) {
        System.out.print(i + 12 + " ");
    }
}
```

Good Programming

KMITL

- การเขียนโปรแกรมที่ดี ต้องแยก Method ตามการทำงานให้ชัดเจน

```
public static void main(String[] args) {
    int a = 3;
    int b = 10;

    // plus
    plus(a, b);

    // loop a to b
    loopAToB(a, b);

    // loop a + 12 to b + 12
    loopAPlus12ToBPlus12(a, b);
}
```

KMITL

To be continued in Lab 2

Question? Problem?