Dart programming language

Application and Game Development for Mobile Device

What did you learn DART?

- ภาษาโปรแกรมที่พัฒนาโดย Google
- โครงสร้างภาษาคล้ายกับ Java, C, Javascript
- มีความสามารถด้าน OOP ให้ใช้งาน
- ใช้งานร่วมกับ Flutter ในการพัฒนา Mobile App

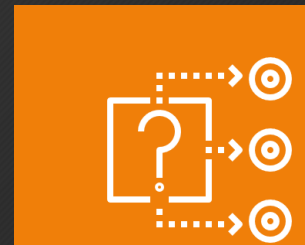
1
- [Offline] Visual Studio Code
- [Online] https://dartpad.dev

2
- เพิ่มส่วนขยายสำหรับ Visual Studio Code

3
- เริ่มต้นเขียนโปรแกรมด้วยภาษา Dart

What was your process?

Purpose

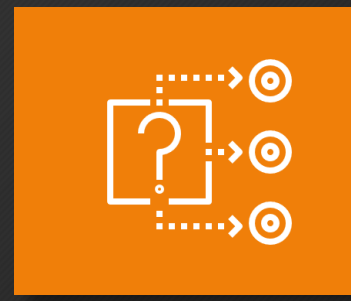


Dart Programming Language

เนื้อหา (Part 1)

- Dart เบื้องต้น
- การแสดงผลข้อมูล
- Comment
- ตัวแปรและชนิดข้อมูล
- Dynamic Type
- Constant & Final
- กฎการตั้งชื่อตัวแปร
- จัดการอักขระและข้อความด้วย String
- ตัวดำเนินการทางคณิตศาสตร์
- ตัวดำเนินการเปรียบเทียบ
- ตัวดำเนินการเพิ่มและลดค่า
- Compound Assignment
- If Statement
- If..Else
- If แบบหลายเงื่อนไข

DART : Part 1



เตรียมความพร้อมสำหรับการสร้าง Mobile Application

เริ่มต้นเขียนโปรแกรมด้วยภาษา Dart

https://dartpad.dev

Main()

main() เป็นจุดเริ่มต้นการทำงานของโค้ดโปรแกรมเป็นอันดับแรก

```
void main() {
   // คำสั่งต่างๆ
}
```

การแสดงผลข้อมูล

รูปแบบคำสั่ง

print(“ข้อความที่ต้องการแสดงผล”);

ตัวอย่างโค้ด

```
void main() {
  print("Hello Dart.");
}
```

ผลลัพธ์

```
Console

  Hello Dart.
```

การเขียนคำอธิบาย (Comment)

วิธีที่ 1 : ใช้อธิบายคำสั่งสั้นๆ บรรทัดเดียว

ใช้เครื่องหมาย Slash (//) 2 ตัว

วิธีที่ 2 : ใช้เขียนอธิบายคำสั่งยาวๆ หลายบรรทัด

ใช้เครื่องหมาย /* ... */

ข้อความอธิบาย หรือ คำสั่ง ที่อยู่ในขอบเขตของ Comment จะไม่ถูกประมวลผล

การเขียนคำอธิบาย (Comment)

ตัวอย่าง

```dart
void main() {
  // comment แสดงข้อความ
  print("Hello Dart.");

  /*
     comment ข้อความที่ต้องการอธิบาย
     หลายบรรทัด
  */
}
```

ตัวแปรและชนิดข้อมูล

ตัวแปร

คือ ชื่อที่ถูกนิยามขึ้นมา เพื่อใช้เก็บค่าข้อมูลสำหรับนำไปใช้งานในโปรแกรม

โดยข้อมูลอาจจะประกอบไปด้วย ข้อความ ตัวเลข ตัวอักษร

หรือ ผลลัพธ์จากการประมวลผลข้อมูล

ตัวแปรและชนิดข้อมูล

ชนิดข้อมูล	คำอธิบาย	รูปแบบข้อมูล
bool	ค่าทางตรรกศาสตร์	True / False
num	ตัวเลขที่ไม่มีจุดทศนิยม (int)	50
	ตัวเลขที่มีจุดทศนิยม (double)	30.15
String	ข้อความ	“Hello”
List<type>	โครงสร้างข้อมูล	[“ ”, “ ”, “ ”]
Map<index,value>	โครงสร้างข้อมูล	{“name”:“Dart”, “version”:“2”}
Dynamic	ตัวแปรเปลี่ยนค่าได้	50, 30.15, True, “Dart”

รูปแบบตัรแปรในภาษา Dart

ตัวแปรแบบ Dynamic Typing

คือ ชนิดตัวแปรเป็นอะไรก็ได้ตามค่าที่เก็บ โดยไม่ต้องประกาศชนิดตัวแปร

ตัวแปรแบบ Static Typing

คือ ต้องประกาศชนิดข้อมูลในตอนเริ่มต้น

เช่น int, double, string เพื่อระบุชนิดข้อมูลที่จะเป็นในตัวแปร

Static Type

ตัวอย่างโค้ด

```dart
void main() {
  // static variable
  int age = 30;
  double price = 100.50;
  String name = "DartPad";

  print("Age: $age");
  print("Price: $price");
  print("Name: $name");
}
```

ผลลัพธ์

```
Console

  Age: 30
  Price: 100.5
  Name: DartPad
```

Dynamic Type

ตัวอย่างโค้ด

```
void main() {
  // dynamic variable
  var age = 30;
  var price = 100.50;
  dynamic name = "DartPad";

  print("Age: $age");
  print("Price: $price");
  print("Name: $name");
}
```

ผลลัพธ์

```
Console

  Age: 30
  Price: 100.5
  Name: DartPad
```

dynamic มีความยืดหยุนมากกว่า var
โดยสามารถเปลี่ยนแปลงชนิดของข้อมูลภายหลังได้

Constant & Final

การนิยามค่าคงที่ (Constant)

const ชนิดข้อมูล ชื่อตัวแปร;

const ชนิดข้อมูล ชื่อตัวแปร = ค่าเริ่มต้น;

* ไม่สามารถนำค่าจากตัวแปรอื่นมาคำนวณได้

การนิยามค่าคงที่ (Final)

final ชนิดข้อมูล ชื่อตัวแปร;

final ชนิดข้อมูล ชื่อตัวแปร = ค่าเริ่มต้น;

* สามารถนำค่าจากตัวแปรอื่นมาคำนวณได้

Constant & Final

```dart
void main() {
  // constant
  const int number1 = 100;
  const number2 = 200;
  print("number1=$number1");
  print("number2=$number2");

  // final
  final int number3 = 300;
  final number4 = 400;
  print("number3=$number3");
  print("number4=$number4");
}
```

Console

```
number1=100
number2=200
number3=300
number4=400
```

```dart
// constant
const int number1 = 100;
int x = 50;
number1 = number1 + x;
```

Constant variables can't be assigned a value.

กฎการตั้งชื่อตัวแปร

- ประกอบไปด้วยตัวเลข ตัวอักษร เครื่องหมาย
- อักษรแรกห้ามขึ้นต้นด้วยตัวเลขและสัญลักษณ์พิเศษ **ยกเว้น _ (Underscore) และ $**
- ห้ามซ้ำกับคำสงวน (Keyword) เช่น final var for int
- Case Sensitive ตัวพิมพ์เล็ก ตัวพิมพ์ใหญ่ ต่างกัน

จัดการอักขระและข้อความด้วย String

การประกาศตัวแปรชนิด String ต้องกำหนดค่าข้อมูลให้อยู่ในเครื่องหมาย

' (single quote) หรือ " (double quote)

```
void main() {
  var lang = 'dartPad';
  String fname = "piyaphol";
  print("$lang : $fname");
  print(lang+' : '+fname);
  int num1 = 100, num2 = 50;
  print("num1 = "+num1.toString());
  print("ผลบวก num1+num2=${num1+num2}");
}
```

Console

```
dartPad : piyaphol
dartPad : piyaphol
num1 = 100
ผลบวก num1+num2=150
```

ตัวดำเนินการทางคณิตศาสตร์

- **ตัวดำเนินการ (Operator)**

  กลุ่มของเครื่องหมาย หรือ สัญลักษณ์ ในการเขียนโปรแกรม

- ตัวถูกดำเนินการ (Operand)

  กลุ่มของตัวเลข หรือ ตัวแปร

Operator	คำอธิบาย
+	บวก
-	ลบ
*	คูณ
/	หาร
%	หารเอาเศษ

ตัวดำเนินการทางคณิตศาสตร์

- ตัวอย่าง ตัวดำเนินการ + (บวก) , – (ลบ)

```
void main() {
  var x = 50, y = 100, z = 0;
  z = x + y;
  print("x+y = $z");

  z = y - x;
  print("y-x = $z");
}
```

```
Console

  x+y = 150
  y-x = 50
```

ตัวดำเนินการทางคณิตศาสตร์

- ตัวอย่าง ตัวดำเนินการ * (คูณ) , / (หาร)

```
void main() {
  var x = 50, y = 100, z = 0.0;
  z = (x * y).toDouble();
  print("x*y = $z");

  z = y / x;
  print("y/x = $z");
}
```

Console

```
x*y = 5000
y/x = 2
```

```
int z;
z = y ~/ x;
print("y/x = $z");
```

ตัวดำเนินการทางคณิตศาสตร์

- ตัวอย่าง ตัวดำเนินการ % (หารเอาเศษ)

```
void main() {
  var x = 50, y = 4;
  int z;
  z = x % y;
  print("(หารเอาเศษ): x%y = $z");
}
```

```
Console

  (หารเอาเศษ): x%y = 2
```

ตัวดำเนินการเปรียบเทียบ

ชนิดข้อมูล

boolean

ผลลัพธ์

True , False

Operator	คำอธิบาย
==	เท่ากับ
!=	ไม่เท่ากับ
>	มากกว่า
<	น้อยกว่า
>=	มากกว่าเท่ากับ
<=	น้อยกว่าเท่ากับ

ตัวดำเนินการเปรียบเทียบ

ตัวอย่าง

```
void main() {
  int num1 = 10, num2 = 20;

  print("เท่ากับ: " + (num1==num2).toString());
  print("ไม่เท่ากับ: " + (num1!=num2).toString());
  print(num1>num2);
  print(num1<num2);
  print(num1>=num2);
  print(num1<=num2);
}
```

Console

```
เท่ากับ: false
ไม่เท่ากับ: true
false
true
false
true
```

ตัวดำเนินการเพิ่มและลดค่า

Operator	รูปแบบการเขียน	คำอธิบาย
++ (Prefix)	++a	เพิ่มค่าให้ a ก่อน 1 ค่าแล้วนำไปใช้
++ (Postfix)	a++	นำค่าปัจจุบันใน a ไปใช้งานก่อนแล้วค่อยเพิ่มค่า 1 ค่า
-- (Prefix)	--b	ลดค่าให้ b ก่อน 1 ค่าแล้วนำไปใช้
-- (Postfix)	b--	นำค่าปัจจุบันใน a ไปใช้งานก่อนแล้วค่อย]ลดค่า 1 ค่า

ตัวดำเนินการเพิ่มและลดค่า

ตัวอย่าง การเพิ่มค่า *(ส่วนการลดค่าใช้วิธีการเดียวกัน)*

```
void main() {
  int a=10;
  print("เริ่มต้นค่า a=$a"); //แสดง a ค่าเริ่มต้น
  print("Prefix a=${++a}"); //+1 ให้ a ก่อนแสดงค่า
  print("ค่าใหม่ a=$a"); //แสดง a ค่าใหม่
  print("Postfix a=${a++}"); //แสดงค่า a ก่อนเพิ่ม +1
  print("ค่าสุดท้าย a=$a");
}
```

Console

```
เริ่มต้นค่า a=10
Prefix a=11
ค่าใหม่ a=11
Postfix a=11
ค่าสุดท้าย a=12
```

Compound Assignment

Assignment	รูปแบบการเขียน	ความหมาย
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x~/=y	x=x~/y
%=	x%=y	x=x%y

Compound Assignment

ตัวอย่าง

```
void main() {
  int x=20, y=5;
  print("x=x+y :${x+=y}"); //20+5=25
  print("x=x-y :${x-=y}"); //25-5=20
  print("x=x*y :${x*=y}"); //20*5=100
  print("x=x~/y :${x~/=y}");//100/5=20
  print("x=x%y :${x%=y}"); //20%5=0
}
```

Console

```
x=x+y :25
x=x-y :20
x=x*y :100
x=x~/y :20
x=x%y :0
```

โครงสร้างควบคุม (Control Structure)

กลุ่มคำสั่งที่ใช้ควบคุมการทำงานของโปรแกรม

- แบบลำดับ (Sequence)
- แบบมีเงื่อนไข (Condition)
- แบบทำซ้ำ (Loop)

แบบมีเงื่อนไข (Condition)

กลุ่มคำสั่งที่ใช้ตัดสินใจในการเลือกเงื่อนไขต่างๆ ภายในโปรแกรมมาทำงาน

- if
- switch..case

If Statement รูปแบบคำสั่งเงื่อนไขเดียว

- เป็นคำสั่งที่ใช้กำหนดเงื่อนไขในการตัดสินใจทำงานของโปรแกรม
- ถ้าเงื่อนไขเป็นจริงจะทำตามคำสั่งต่างๆ ที่กำหนดภายใต้เงื่อนไขนั้นๆ

```
If(เงื่อนไข) {
    ชุดคำสั่งเมื่อเงื่อนไขเป็นจริง ;
}
```

If Statement ตัวอย่าง แบบเงื่อนไขเดียว

```
void main() {
  int x=20, y=20;
  if(x==y){
    print("x มีค่าเท่ากับ y");
  }
  print("จบโปรแกรม");
}
```

Console

```
x มีค่าเท่ากับ y
จบโปรแกรม
```

```
void main() {
  int x=20, y=30;
  if(x==y){
    print("x มีค่าเท่ากับ y");
  }
  print("จบโปรแกรม");
}
```

Console

```
จบโปรแกรม
```

If..else รูปแบบคำสั่ง 2 เงื่อนไข

```
If(เงื่อนไข) {
    ชุดคำสั่งเมื่อเงื่อนไขเป็นจริง ;
} else {
    ชุดคำสั่งเมื่อเงื่อนไขเป็นเท็จ ;
}
```

If Statement ตัวอย่าง แบบ 2 เงื่อนไข

```
void main() {
  int x=20;
  if(x%2==0){
    print("$x เป็นเลขคู่");
  } else {
    print("$x เป็นเลขคี่");
  }
  print("จบโปรแกรม");
}
```

Console

```
20 เป็นเลขคู่
จบโปรแกรม
```

If..else รูปแบบคำสั่งหลายเงื่อนไข

```
If(เงื่อนไขที่ 1) {
    ชุดคำสั่งเมื่อเงื่อนไขที่ 1 เป็นจริง ;
} else if(เงื่อนไขที่ 2) {
    ชุดคำสั่งเมื่อเงื่อนไขที่ 2 เป็นจริง ;
} else if(เงื่อนไขที่ 3) {
    ชุดคำสั่งเมื่อเงื่อนไขที่ 3 เป็นจริง ;
} else {
    ชุดคำสั่งเมื่อทุกเงื่อนไขเป็นเท็จ ;
}
```

If..else ตัวอย่าง แบบหลายเงื่อนไข

```
void main() {
  int score=42;
  if(score>=40){
    print("คะแนน $score/50 ดีเยี่ยม");
  } else if(score>=30) {
    print("คะแนน $score/50 ดี");
  } else if(score>=25) {
    print("คะแนน $score/50 ปานกลาง");
  } else {
    print("คะแนน $score/50 ควรปรับปรุง");
  }
}
```

Console

```
คะแนน 42/50 ดีเยี่ยม
```

Part 1 ✓

- Dart เบื้องต้น
- การแสดงผลข้อมูล
- Comment
- ตัวแปรและชนิดข้อมูล
- Dynamic Type
- Constant & Final
- กฎการตั้งชื่อตัวแปร
- จัดการอักขระและข้อความด้วย String
- ตัวดำเนินการทางคณิตศาสตร์
- ตัวดำเนินการเปรียบเทียบ
- ตัวดำเนินการเพิ่มและลดค่า
- Compound Assignment
- If Statement
- If..Else
- If แบบหลายเงื่อนไข

Part 2

- ตัวดำเนินการทางตรรกศาสตร์
- Ternary Operator
- Switch..Case
- While Loop
- For Loop
- Do..While
- Break และ Continue
- การใช้งาน Loop แต่ละแบบ
- การสร้างฟังก์ชั่น
- ฟังก์ชั่นแบบ Return ค่า
- ฟังก์ชั่นแบบส่ง และ Return ค่า
- Arrow Function
- Optional Parameter
- Named Parameter
- First-Class Function

Part 3

- โครงสร้างข้อมูล List
- List Properties & Function
- เข้าถึงสมาชิกใน List ด้วย For
- เข้าถึงสมาชิกใน List ด้วย ForEach
- ฟังก์ชั่นเพิ่มสมาชิกใน List
- ฟังก์ชั่นลบสมาชิกใน List
- โครงสร้างข้อมูล Map
- แปลง List เป็น Map
- แปลง Map เป็น List
- ทฤษฎีการเขียนโปรแกรมเชิงวัตถุ
- การสร้าง Class
- การสร้าง Object
- Public / Private
- Getter / Setter
- Constructor
- การสืบทอดคุณสมบัติ
- Overriding Method


Q
&
A