Inheritance Of Constructor in Dart

by Kanjanat Sriprasomsong

Last edited 28/09/2566

URL video : https://youtu.be/pPnw_07POx0

Constructor คืออะไร?

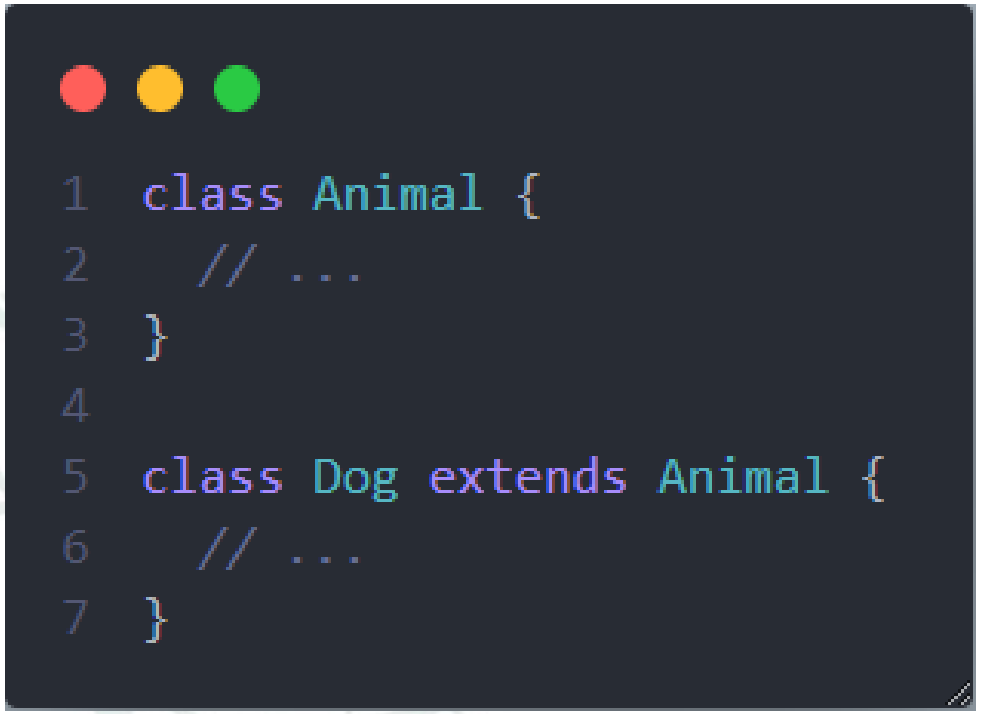
Constructor คือเมธอดเฉพาะที่ใช้ชื่อเดียวกันกับชื่อคลาส และเรียกใช้งานโดยอัตโนมัติเมื่อเราสร้างวัตถุของคลาส

```
1  class ClassName {
2    // Constructor declaration: Same as class name
3    ClassName() {
4      // body of the constructor
5    }
6  }
```

- การสร้าง Constructor ให้กับคลาส
- โดยการใช้คำสั่ง ClassName() เพื่อสร้างวัตถุของคลาสและกำหนดคุณสมบัติเริ่มต้นของวัตถุ

Inheritance คืออะไร?

Inheritance คือการสืบทอดเป็นหนึ่งในแนวคิดสำคัญในการโปรแกรมเชิงวัตถุ ซึ่งอนุญาตให้คลาสหนึ่งสามารถสืบทอดคุณสมบัติและเมธอดจากคลาสอื่น ๆ ได้ สามารถสืบทอดคุณสมบัติจากคลาสหนึ่งไปยังคลาสอื่นๆ โดยใช้คีย์เวิร์ด **extends** ดังนี้



```
1  class Animal {
2    // ...
3  }
4
5  class Dog extends Animal {
6    // ...
7  }
```

การใช้คีย์เวิร์ด extends โดยที่คลาสแม่กับคลาสลูกจะมีความสัมพันธ์กันในรูปแบบ " is-a " หรือความสัมพันธ์แบบ "เป็น“

จากตัวอย่างจะอธิบายได้ว่า " Dog" เป็นคลาสลูกของ " Animal " เราสามารถสรุปได้ว่า " Dog is an Animal. "

รูปแบบของ Constructor มีทั้งหมด 4 รูปแบบในภาษา Dart ได้แก่

1. Default Constructor

```dart
class MyClass {
  // Default Constructor ถูกสร้างโดยอัตโนมัติ
}
```

คือคอนสตรักเตอร์ที่ไม่รับพารามิเตอร์ แต่ถ้าไม่ระบุคอนสตรักเตอร์ ใด ๆ คอนสตรักเตอร์จะถูกสร้างขึ้นโดยอัตโนมัติ เพื่อใช้สร้างอ็อบเจ็กต์ของคลาสนั้น ๆ

2. Named Constructor

```dart
class MyClass {
  MyClass.namedConstructor() {
    // Constructor ที่มีชื่อ
  }
}

```

คอนสตรักเตอร์ที่มีชื่อเฉพาะ และไม่ได้ถูกเรียกโดยอัตโนมัติ โดยจะต้องเรียกชื่อคอนสตรักเตอร์ นี้เองเมื่อคุณต้องการสร้างอ็อบเจ็กต์

รูปแบบของ Constructor มีทั้งหมด 4 รูปแบบในภาษา Dart ได้แก่ (ต่อ)

3. Parameterized Constructor

```dart
class MyClass {
  int value;

  MyClass(this.value); // Constructor ที่มีพารามิเตอร์
}
```

คอนสตรักเตอร์ใด ๆ ที่มีการรับค่าพารามิเตอร์ในการสร้างอ็อบเจ็กต์ ๆ

4. Factory Constructor.

```dart
class MyClass {
  int value;

  MyClass._privateConstructor(this.value);

  factory MyClass.createInstance(int value) {
    if (value > 0) {
      return MyClass._privateConstructor(value);
    } else {
      return null;
    }
  }
}
```

คอนสตรักเตอร์ ที่ไม่สร้างอ็อบเจ็กต์โดยตรง แต่อาจจะสร้างและคืนค่าอ็อบเจ็กต์จากหลายตัวเลือกตามการตรวจสอบของคอนสตรักเตอร์ นั้น ๆ

Example1 การสืบทอดคอนสตรักเตอร์

```dart
class Laptop {

  Laptop() {
    print("Laptop constructor");
  }
}

class MacBook extends Laptop {

  MacBook() {
    print("MacBook constructor");
  }
}

void main() {
  var macbook = MacBook();
}

```

Output

```
Laptop constructor
MacBook constructor
```

Example2 การสืบทอดคอนสตรักเตอร์ที่มีพารามิเตอร์

```
class Laptop {
  // Constructor
  Laptop(String name, String color) {
    print("Laptop constructor");
    print("Name: $name");
    print("Color: $color");
  }
}

class MacBook extends Laptop {
  // Constructor
  MacBook(String name, String color) : super(name, color) {
    print("MacBook constructor");
  }
}

void main() {
  var macbook = MacBook("MacBook Pro", "Silver");
}
```

Output

```
Laptop constructor
Name: MacBook Pro
Color: Silver
MacBook constructor
```

Example3 การสืบทอดคอนสตรักเตอร์ที่มีพารามิเตอร์แบบชื่อ

```dart
class Laptop {
  // Constructor
  Laptop({String name = '', String color=''}) {
    print("Laptop constructor");
    print("Name: $name");
    print("Color: $color");
  }
}

class MacBook extends Laptop {
  // Constructor
  MacBook({String name ='', String color =''}): super(name: name, color: color) {
    print("MacBook constructor");
  }
}

void main() {
  var macbook = MacBook(name: "MacBook Pro", color: "Silver");
}
```

Output

```
Laptop constructor
Name: MacBook Pro
Color: Silver
MacBook constructor
```

เปรียบเทียบความแตกต่างระหว่าง Example2 กับ Example3

```
class Laptop {
  // Constructor
  Laptop(String name, String color) {
    print("Laptop constructor");
    print("Name: $name");
    print("Color: $color");
  }
}

class MacBook extends Laptop {
  // Constructor
  MacBook(String name, String color) : super(name, color) {
    print("MacBook constructor");
  }
}

void main() {
  var macbook = MacBook("MacBook Pro", "Silver");
}
```

```
class Laptop {
  // Constructor
  Laptop({String name = '', String color=''}) {
    print("Laptop constructor");
    print("Name: $name");
    print("Color: $color");
  }
}

class MacBook extends Laptop {
  // Constructor
  MacBook({String name ='', String color =''}): super(name: name, color: color) {
    print("MacBook constructor");
  }
}

void main() {
  var macbook = MacBook(name: "MacBook Pro", color: "Silver");
}
```

Output

Laptop constructor
Name: MacBook Pro
Color: Silver
MacBook constructor

Example4 การเรียกใช้งาน Named Constructor ของคลาสแม่

```dart
class Laptop {
  // Default Constructor
  Laptop() {
    print("Laptop constructor");
  }

  // Named Constructor
  Laptop.named() {
    print("Laptop named constructor");
  }
}

class MacBook extends Laptop {
  // Constructor
  MacBook() : super.named() {
    print("MacBook constructor");
  }
}

void main() {
  var macbook = MacBook();
}
```

Output

```
Laptop named constructor
MacBook constructor
```

Example5 การสืบทอดคอนสตรักเตอร์ที่รับพารามิเตอร์แค่ในคลาสลูก และกำหนดค่าตัวแปรในคลาสแม่ผ่านคอนสตรักเตอร์ของคลาสลูก

```
class Person {
  String name;
  int age;

  // Constructor
  Person(this.name, this.age);
}

class Student extends Person {
  int rollNumber;

  // Constructor
  Student(String name, int age, this.rollNumber) : super(name, age);
}

void main() {
  var student = Student("John", 20, 1);
  print("Student name: ${student.name}");
  print("Student age: ${student.age}");
  print("Student roll number: ${student.rollNumber}");
}
```

Output

```
Student name: John
Student age: 20
Student roll number: 1
```

เปรียบเทียบการสืบทอดคอนสตรักเตอร์ในภาษา Dart กับภาษาอื่น ๆ

- คีย์เวิร์ดที่ใช้เรียกคอนสตรักเตอร์ของคลาสแม่ในคลาสลูก

```dart
class Laptop {
  // Constructor
  Laptop(String name, String color) {
    print("Laptop constructor");
    print("Name: $name");
    print("Color: $color");
  }
}

class MacBook extends Laptop {
  // Constructor
  MacBook(String name, String color) : super(name, color) {
    print("MacBook constructor");
  }
}

void main() {
  var macbook = MacBook("MacBook Pro", "Silver");
}
```

Dart

```java
class Laptop {
    // Constructor
    Laptop(String name, String color) {
        System.out.println("Laptop constructor");
        System.out.println("Name: " + name);
        System.out.println("Color: " + color);
    }
}

class MacBook extends Laptop {
    // Constructor
    MacBook(String name, String color) {
        super(name, color);
        System.out.println("MacBook constructor");
    }
}

public class Main {
    public static void main(String[] args) {
        MacBook macbook = new MacBook("MacBook Pro", "Silver");
    }
}
```

Java

เปรียบเทียบการสืบทอดคอนสตรักเตอร์ภาษาในภาษา Dart กับภาษาอื่น ๆ

- คีย์เวิร์ดที่ใช้เรียกคอนสตรักเตอร์ของคลาสแม่ในคลาสลูก

```dart
class Laptop {
  // Constructor
  Laptop(String name, String color) {
    print("Laptop constructor");
    print("Name: $name");
    print("Color: $color");
  }
}

class MacBook extends Laptop {
  // Constructor
  MacBook(String name, String color) : super(name, color) {
    print("MacBook constructor");
  }
}

void main() {
  var macbook = MacBook("MacBook Pro", "Silver");
}
```

Dart

```python
class Laptop:
    # Constructor
    def __init__(self, name, color):
        print("Laptop constructor")
        print("Name:", name)
        print("Color:", color)

class MacBook(Laptop):
    # Constructor
    def __init__(self, name, color):
        super().__init__(name, color)
        print("MacBook constructor")

macbook = MacBook("MacBook Pro", "Silver")
```

Python

เปรียบเทียบการสืบทอดคอนสตรักเตอร์ภาษาในภาษา Dart กับภาษาอื่น ๆ

- คีย์เวิร์ดที่ใช้เรียกคอนสตรักเตอร์ของคลาสแม่ในคลาสลูก

```dart
class Laptop {
  // Constructor
  Laptop(String name, String color) {
    print("Laptop constructor");
    print("Name: $name");
    print("Color: $color");
  }
}

class MacBook extends Laptop {
  // Constructor
  MacBook(String name, String color) : super(name, color) {
    print("MacBook constructor");
  }
}

void main() {
  var macbook = MacBook("MacBook Pro", "Silver");
}
```

Dart

```c
#include <stdio.h>
#include <string.h>

struct Laptop {
    // Constructor
    Laptop(const char *name, const char *color) {
        printf("Laptop constructor\n");
        printf("Name: %s\n", name);
        printf("Color: %s\n", color);
    }
};

struct MacBook {
    struct Laptop laptop;
};

int main() {
    const char *name = "MacBook Pro";
    const char *color = "Silver";
    struct MacBook macbook = {{name, color}};
    printf("MacBook constructor\n");
    return 0;
}
```

C

เปรียบเทียบการสืบทอดคอนสตรักเตอร์ภาษาในภาษา Dart กับภาษาอื่น ๆ

- เมื่อไม่มีการประกาศ Default Constructor
  - Dart และ Java
    จะสร้าง Default Constructor โดยอัตโนมัติ
  - Python
    ไม่มี Default Constructor สามารถสร้างอ็อบเจ็กต์โดยใช้คลาสตรง ๆ โดยไม่ต้องประกาศคอนสตรักเตอร์
  - C
    เป็นภาษาที่ไม่มีคอนเซปต์ของคลาสและการสืบทอดแบบตรงๆ จะใช้ struct หรือฟังก์ชันเพื่อใช้เป็นทางเลือกในการจำลองคลาสและการสืบทอด

ขอจบการนำเสนอ
ขอบคุณค่ะ

by Kanjanat Sriprasomsong
Last edited 28/09/2566