Parameterized Constructor in Dart

640710052 ธนาดล ชีวโรจน์ณรงค์

Parameterized Constructor in Dart คือ

- เป็น **constructor** ที่มีการกำหนด **parameters** และ ค่าที่รับมาจะถูกส่งไปใน constructor ขณะที่กำลังมีการสร้าง **object** อาจเพื่อใช้กำหนดค่าให้ **instance variables** หรือ ตัวแปร ของ **class**

Syntax :

```dart
class ClassName {
  int? number;
  String? name;

  ClassName(this.number, this.name);
}
```

```dart
class ClassName {
  int? number;
  String? name;

  ClassName(int number, String name) {
    this.number = number;
    this.name = name;
  }
}
```

Example 1:

Parameterized Constructor In Dart

```dart
class Student {
  String? name;
  int? age;
  int? rollNumber;
  // Constructor
  Student(this.name, this.age, this.rollNumber);
}
Run | Debug
void main() {
  // Here student is object of class Student.
  Student student = Student("John", 20, 1);
  print("Name: ${student.name}");
  print("Age: ${student.age}");
  print("Roll Number: ${student.rollNumber}");
}
```

- **constructor** จะรับ **parameter** 3 ตัวที่มีชนิดข้อมูลเดียวกับตัวแปร **name** , **age** และ **rollNumber** ตามลำดับ แล้วกำหนดค่าให้ **parameter** ทั้งสาม

OUTPUT :

```
Name: John
Age: 20
Roll Number: 1
```

Example 2: Parameterized Constructor With Named Parameters In Dart

```dart
class Student {
  String? name;
  int? age;
  int? rollNumber;

  Student({this.name, this.age, this.rollNumber});
}
Run | Debug
void main() {
  // Here student is object of class Student.
  Student student = Student(name: "John",  rollNumber: 1, age: 20);
  print("Name: ${student.name}");
  print("Age: ${student.age}");
  print("Roll Number: ${student.rollNumber}");
}
```

OUTPUT :

```
Name: John
Age: 20
Roll Number: 1
```

- ตัวอย่างนี้จากต่างจากตัวอย่างที่ 1 คือ มีการใช้ **name parameters** ใน **constructor** สังเกตจากการใส่ **{ }** ครอบ **parameter**
- ทำให้การสร้าง **object** ง่ายขึ้น เพราะ ไม่จำเป็นต้อง เรียง **argument** ที่จะส่งไปให้ตรงกับ **parameter** ของ **constructor** เพราะ มี **name parameters** คอยกำกับอยู่

Example 3: Parameterized Constructor With Default Values In Dart

```dart
class Student {
  String? name;
  int? age;

  Student({String? name = "John", int? age = 0}) {
    this.name = name;
    this.age = age;
  }
}

Run | Debug
void main() {
  // Here student is object of class Student.
  Student student = Student();
  print("Name: ${student.name}");
  print("Age: ${student.age}");
}
```

OUTPUT :

```
Name: John
Age: 0
```

- **constructor** ในตัวอย่างมีการกำหนดค่าเริ่มต้นให้ **parameters name** = "John" และ **age** = 0
- เมื่อทำการสร้าง **object** จะส่งหรือไม่ส่ง **argument** ให้ **parameter** ที่มีการกำหนดค่า ก็ได้

Parameterized Constructor in Other languages

IN C++

- การสร้าง **constructor** แบบ **Parameterized Constructor** ใน **C++** เพียงเพิ่ม **parameter** ในลักษณะเดียวกับที่เพิ่มให้กับ **functions**

OUTPUT :

```
p1.x = 10, p1.y = 15
```

```cpp
#include <iostream>
using namespace std;

class Point
{
private:
    int x, y;

public:
    // Parameterized Constructor
    Point(int x1, int y1)
    {
        x = x1;
        y = y1;
    }
    int getX() { return x; }
    int getY() { return y; }
};
int main()
{
    Point p1(10, 15);
    cout << "p1.x = " << p1.getX()
         << ", p1.y = " << p1.getY();
    return 0;
}
```

IN Java

- **Parameterized Constructor** ใน **Java** สร้างได้โดย เพิ่ม **parameter** ให้ **constructor**
- ในภาษา **Java** สามารถมี **constructor** ที่มีชื่อเดียวกันแต่ **parameter** ต่างกันได้ เรียกว่า **constructor overloading**

OUTPUT :

```
Constructor with one argument - String : Shikhar
Constructor with two arguments :  String and Integer : Dharmesh 26
Constructor with one argument : Long : 325614567
```

```java
class Geek {
    Geek(String name) {
        System.out.println("Constructor with one "
                + "argument - String : " + name);
    }
    Geek(String name, int age) {

        System.out.println(
                "Constructor with two arguments : "
                        + " String and Integer : " + name + " " + age);
    }
    Geek(long id) {
        System.out.println(
                "Constructor with one argument : "
                        + "Long : " + id);
    }
}

class GFG {
    Run | Debug
    public static void main(String[] args) {
        Geek geek2 = new Geek(name:"Shikhar");
        Geek geek3 = new Geek(name:"Dharmesh", age:26);
        Geek geek4 = new Geek(id:325614567);
    }
}
```

IN Python

- **Parameterized Constructor** ใน **python constructor** จะมี **parameter self** ที่จะ **reference** ถึง **object** ของ **class** นั้นๆที่ถูกสร้าง และ **parameter** อื่นๆ จะเป็น **parameter** ที่ **programmer** เป็นคนเขียนขึ้นมาเพิ่ม

```python
class Addition:
    first = 0
    second = 0
    answer = 0
    # parameterized constructor
    def __init__(self, f, s):
        self.first = f
        self.second = s

    def display(self):
        print("First number = " + str(self.first))
        print("Second number = " + str(self.second))
        print("Addition of two numbers = " + str(self.answer))

    def calculate(self):
        self.answer = self.first + self.second

obj1 = Addition(1000, 2000)
obj2 = Addition(10, 20)
obj1.calculate()
obj2.calculate()
obj1.display()
obj2.display()
```

OUTPUT :

```
First number = 1000
Second number = 2000
Addition of two numbers = 3000
First number = 10
Second number = 20
Addition of two numbers = 30
```

Thank you