In Dart with my understanding, method parameter can be given in two type.

* Required parameter
* Optional parameter (positional, named & default)

**>> Required Parameter**

Required parameter is a well know old style parameter which we all familiar with it

***example****:*

findVolume(int length, int breath, int height) {

print('length = $length, breath = $breath, height = $height');

}

findVolume(10,20,30);

***output:***

length = 10, breath = 20, height = 30

**>> Optional Positional Parameter**

parameter will be disclosed with square bracket **[ ]** & square bracketed parameter are optional.

***example:***

findVolume(int length, int breath, [int height]) {

print('length = $length, breath = $breath, height = $height');

}

findVolume(10,20,30);//valid

findVolume(10,20);//also valid

***output:***

length = 10, breath = 20, height = 30

length = 10, breath = 20, height = null // no value passed so height is null

***>> Optional Named Parameter***

* parameter will be disclosed with curly bracket { }
* curly bracketed parameter are optional.
* have to use parameter name to assign a value which separated with colan **:**
* in curly bracketed parameter order does not matter
* these type parameter help us to avoid confusion while passing value for a function which has many parameter.

***example:***

findVolume(int length, int breath, {int height}) {

print('length = $length, breath = $breath, height = $height');

}

findVolume(10,20,height:30);//valid & we can see the parameter name is mentioned here.

findVolume(10,20);//also valid

***output:***

length = 10, breath = 20, height = 30

length = 10, breath = 20, height = null

***>> Optional Default Parameter***

* same like optional named parameter in addition we can assign default value for this parameter.
* which means no value is passed this default value will be taken.

***example:***

findVolume(int length, int breath, {int height=10}) {

print('length = $length, breath = $breath, height = $height');

}

findVolume(10,20,height:30);//valid

findVolume(10,20);//valid

***output:***

length = 10, breath = 20, height = 30

length = 10, breath = 20, height = 10 // default value 10 is taken

thanks for the clear explanation given from this video link, credits to the video creator.

**Optional parameters**

Optional parameters có thể truyền theo tên hoặc theo vị trí. Nhưng không đồng thời có cả 2.

**Named parameters**

Khi gọi một function ta có thể chỉ định tên của các params, với điều kiện ta phải dùng {param1, param2, …} khi khai báo chúng. Bên cạnh đó, khi đã được chỉ định tên, thì vị trí các params có thể thay đổi. Ví dụ như:

void main() {

print("${sum(first: 1, second: 2)}"); // (1)

print("${sum(second: 2, first: 1)}"); // (2)

// both (1) and (2) are the same

}

int sum({int first, int second}) => first + second;

Mặc dù **named parameters** là một kiểu của optional parameter, tức là chúng ta có thể bỏ qua và không cần pass param đó.

sum(first: 1);

sum(second: 2);

Tuy nhiên chúng ta có thể sử dụng @required annotation để chú thích rằng param đó là bắt buộc.

sum(first: 1, second: 2);

sum(first: 1); // Warning: The parameter 'second' is required

Để dùng @required annotation, chúng ta phải có [**meta**](https://pub.dev/packages/meta) package và import package:meta/meta.dart.

**Positional parameters**

Để khai báo optional positional parameters, ta sẽ bỏ nó trong dấu []

int sum(int first, int second, [int third]) {

if(third != null) {

return first + second + third;

}

return first + second;

}

// without the optional parameter

sum(1, 2)

// with the third parameter

sum(1, 2, 3)

**Default parameter values**

Để định nghĩa default value cho named và positional parameters ta sẽ dùng dấu = sau param và trước giá trị default. *Default value phải là một compile-time constants*. Nếu ta không cung cấp giá trị default, thì nó sẽ là null.

void main() {

print(sum1(1, 2)); // print: 3

print(sum1(1, 2, 3)); // print: 6

print(sum2(1, 2)); // print: 3

print(sum2(1, 2, third: 3)); // print: 6

}

// Default parameter values with positional parameters

int sum1(int first, int second, [int third = 0]) {

return first + second + third;

}

// Default parameter values with named parameters

int sum2(int first, int second, {int third = 0}) {

return first + second + third;

}

Ví dụ về trường hợp param là một List

void main() {

print(sum()); // print: 6

}

int sum({List list = const [1, 2, 3]}) {

var sum = 0;

list.forEach((e) {

sum += e;

});

return sum;

}

Như mình đã đề cập phía trên, default value phải là một compile-time constants. Nên trong trường hợp này, khi khai báo List ta phải có const keyword. Nếu không có thì compiler sẽ báo lỗi:

Error: Constant expression expected.

**Functions as first-class objects**

Pass một function như parameter vào function khác. Ví dụ như:

void printElement(int element) {

print(element);

}

var list = [1, 2, 3];

// Pass printElement as a parameter.

list.forEach(printElement);

Hoặc gán nó như một biến như bên dưới

var loudify = (msg) => '!!! ${msg.toUpperCase()} !!!';

assert(loudify('hello') == '!!! HELLO !!!');