

Lập trình Mobile



Tuần 2

Giảng viên: Trần Đức Minh

Nội dung bài giảng



- Các kiểu dữ liệu
- Biến
- Hằng
- Toán tử
- Các cấu trúc điều khiển
- Kiểu List, Map
- Hàm và Hàm Lambda

Các kiểu dữ liệu



- dartpad.dartlang.org
- Dart hỗ trợ các kiểu dữ liệu sau
 - Kiểu số
 - int : Số nguyên
 - double : Số thực 64 bits
 - Kiểu chuỗi ký tự
 - String : chuỗi ký tự của các ký tự dạng UTF-16
 - Kiểu logic
 - bool : chỉ nhận 2 giá trị true hoặc false

Biến



- Quy tắc đặt tên biến
 - Không trùng với từ khóa.
 - Có thể chứa chữ cái và chữ số.
 - Không chứa dấu cách và ký tự đặc biệt, ngoại trừ dấu gạch dưới (_) và ký hiệu đô la (\$).
 - Tên biến không được bắt đầu bằng số.
- Ví dụ:
 - Tên biến hợp lệ: **test\$123**, **test_123**
 - Tên biến không hợp lệ: **3test**, **3 test**, **for**

Biến



- Các biến trong Dart tham chiếu đến giá trị thay vì chứa giá trị.
- Sử dụng từ khóa **var** để khai báo một biến khi chưa xác định được kiểu giá trị cho nó.
 - Kiểu dữ liệu sẽ được xác định khi biến được nhận giá trị đầu tiên và sau đó kiểu dữ liệu sẽ không thay đổi được.
 - Ví dụ:
 - `var name = "Sinh vien";`
 - `name = 1; // Lỗi`
- Từ khóa **dynamic** tương tự từ khóa **var** nhưng kiểu dữ liệu của biến được phép thay đổi trong quá trình thực hiện.
 - Ví dụ:
 - `dynamic name = "Sinh vien";`
 - `name = 1; // Không báo lỗi ở đây`
- Các giá trị chưa được khởi gán ban đầu sẽ tự động nhận giá trị **null**.

Hằng



- Từ khóa **final** và **const** được sử dụng để khai báo các hằng số.
- Cú pháp:
final [<kiểu dữ liệu>] <tên hằng> = <giá trị>
const [<kiểu dữ liệu>] <tên hằng> = <giá trị>
- Ví dụ:
final pi = 3.14;
final double pi = 3.14;
const e = 2.72;
const double e = 2.72;

Toán tử



- Các toán tử số học: +, -, *, /, ~/, %, ++, --
 - ~/ : Chia lấy phần nguyên
- Các toán tử quan hệ: >, <, >=, <=, ==, !=
- Toán tử kiểm tra kiểu dữ liệu: **is**, **!is**
- Toán tử thực hiện bit: &, |, ^, ~, <<, >>
 - ^ : toán tử xor
 - ~ : toán tử not
 - << : dịch trái bit
 - >> : dịch phải bit
- Toán tử gán: =, ??=, +=, -=, *=, /=
 - ??= : chỉ gán giá trị nếu giá trị bằng null

Toán tử



- Toán tử logic: &&, ||, !
- Toán tử điều kiện
 - <điều kiện> ? <biểu thức 1> : <biểu thức 2>
 - <biểu thức 1> ?? <biểu thức 2>
 - Nếu <biểu thức 1> khác null thì trả về giá trị của nó; ngược lại trả về giá trị của <biểu thức 2>

Cấu trúc điều kiện và lặp



- Cấu trúc **if** và **switch** giống ngôn ngữ C
- Cấu trúc **for**, **while**, **do...while** giống ngôn ngữ C
- Cấu trúc **for ... in**

- Cú pháp:

```
for (<tên biến> in <đối tượng>) {  
    <lệnh thực hiện>  
}
```

- Ví dụ:

```
var obj = [12,13,14];  
for (var prop in obj) {  
    print(prop);  
}
```

Làm việc với kiểu số



- **int** : số nguyên
- **double** : kiểu số thực 64 bit theo chuẩn IEEE 754
- **num** : là kiểu kế thừa từ cả hai kiểu dữ liệu trên.
- Hàm **parse()** chuyển đổi một chuỗi thành một số.
 - Ví dụ: `num.parse("345");`

Làm việc với kiểu số



- Một số thuộc tính và hàm hỗ trợ

Sr.No	Property & Description
1	hashCode ↗ Returns a hash code for a numerical value.
2	isFinite ↗ True if the number is finite; otherwise, false.
3	isInfinite ↗ True if the number is positive infinity or negative infinity; otherwise, false.
4	isNaN True if the number is the double Not-a-Number value; otherwise, false.
5	isNegative ↗ True if the number is negative; otherwise, false.
6	sign ↗ Returns minus one, zero or plus one depending on the sign and numerical value of the number.
7	isEven ↗ Returns true if the number is an even number.
8	isOdd ↗ Returns true if the number is an odd number.

Sr.No	Method & Description
1	abs ↗ Returns the absolute value of the number.
2	ceil ↗ Returns the least integer no smaller than the number.
3	compareTo ↗ Compares this to other number.
4	Floor ↗ Returns the greatest integer not greater than the current number.
5	remainder ↗ Returns the truncated remainder after dividing the two numbers.
6	Round ↗ Returns the integer closest to the current numbers.
7	toDouble ↗ Returns the double equivalent of the number.
8	toInt ↗ Returns the integer equivalent of the number.
9	toString ↗ Returns the string equivalent representation of the number.
10	truncate ↗ Returns an integer after discarding any fractional digits.

Làm việc với kiểu chuỗi



- String
- Sử dụng toán tử + để ghép 2 chuỗi
 - Ví dụ: `String testString = "Dai" + "hoc";`
- `${<biến>}` đưa giá trị của biến vào chuỗi
 - Ví dụ:
 - `int a = 8;`
 - `String str = "Gia tri cua a = ${a}";`

Làm việc với kiểu chuỗi



- Một số thuộc tính và hàm hỗ trợ

Sr.No	Property & Description
1	<code>codeUnits</code> ↗ Returns an unmodifiable list of the UTF-16 code units of this string.
2	<code>isEmpty</code> ↗ Returns true if this string is empty.
3	<code>Length</code> ↗ Returns the length of the string including space, tab and newline characters.

Sr.No	Methods & Description
1	<code>toLowerCase()</code> ↗ Converts all characters in this string to lower case.
2	<code>toUpperCase()</code> ↗ Converts all characters in this string to upper case.
3	<code>trim()</code> ↗ Returns the string without any leading and trailing whitespace.
4	<code>compareTo()</code> ↗ Compares this object to another.
5	<code>replaceAll()</code> ↗ Replaces all substrings that match the specified pattern with a given value.
6	<code>split()</code> ↗ Splits the string at matches of the specified delimiter and returns a list of substrings.
7	<code>substring()</code> ↗ Returns the substring of this string that extends from <code>startIndex</code> , inclusive, to <code>endIndex</code> , exclusive.
8	<code>toString()</code> ↗ Returns a string representation of this object.
9	<code>codeUnitAt()</code> ↗ Returns the 16-bit UTF-16 code unit at the given index.

Kiểu List



- List trong Dart tương tự kiểu mảng ở các ngôn ngữ khác.
- Khai báo:

```
var <tên list> = [<giá trị 1>, <giá trị 2>, ...];
```

```
var <tên list> = new List.filled(0, 0, growable: true);
```



Truy cập List thông qua chỉ số bắt đầu từ 0

 - Ví dụ: `list[0] = 5; list[1] = 9;`

Kiểu List



- Một số hàm hỗ trợ

Sr.No	Methods & Description
1	<code>first</code>  Returns the first element in the list.
2	<code>isEmpty</code>  Returns true if the collection has no elements.
3	<code>isNotEmpty</code>  Returns true if the collection has at least one element.
4	<code>length</code>  Returns the size of the list.
5	<code>last</code>  Returns the last element in the list.
6	<code>reversed</code>  Returns an iterable object containing the lists values in the reverse order.
7	<code>Single</code>  Checks if the list has only one element and returns it.

Kiểu List



- Các thao tác cơ bản trên List
 - Thêm:
 - `add()`, `addAll()`, `insert()`, `insertAll()`
 - Cập nhật:
 - `replaceRange()`
 - Xóa:
 - `remove()`, `removeAt()`, `removeLast()`, `removeRange()`
- Tự tra cứu cách thực hiện của các hàm trên.

Kiểu Map



- Map là một tập hợp các đối tượng có dạng `<key, value>`
- Khai báo
 - `var <tên Map> = {key1:value1, key2:value2, ...}`
 - `var <tên Map> = new Map()`
- Ví dụ:
 - `var info = {'MaSV':'A9871', 'Ten': 'Thanh'};`

Kiểu Map



- Một số thuộc tính và hàm hỗ trợ

Sr.No	Property & Description
1	Keys ↗ Returns an iterable object representing keys
2	Values ↗ Returns an iterable object representing values
3	Length ↗ Returns the size of the Map
4	isEmpty ↗ Returns true if the Map is an empty Map
5	isEmpty ↗ Returns true if the Map is an empty Map

Sr.No	Function Name & Description
1	addAll() ↗ Adds all key-value pairs of other to this map.
2	clear() ↗ Removes all pairs from the map.
3	remove() ↗ Removes key and its associated value, if present, from the map.
4	forEach() ↗ Applies f to each key-value pair of the map.

Hàm và Hàm Lamda



- Hàm trong Dart giống ngôn ngữ C
- Hàm Lamda hay còn gọi là Hàm mũi tên là một cơ chế biểu diễn hàm một cách ngắn gọn.
 - Cú pháp:
`<giá trị trả về> <tên hàm>(<tham số>) => <biểu thức>;`
 - Ví dụ:
 - `printMsg() => print("hello");`
 - `int tong(var a, var b) => a + b;`

Hết Tuần 2



Cảm ơn các bạn đã chú ý lắng nghe !!!