**2018 －2019 学年第 二 学期**

组号：G02

****

**霸笔笔记app**

**代码规范**

实验课程名称 软件工程基础

专 业 班 级 软件工程1701 1702

学 号 31701401 31701257 31701371

学 生 姓 名 高兴欣、王晨旭、倪嘉玲

实验指导老师 杨枨

|  |  |  |
| --- | --- | --- |
| 文件状态： | 文件标识： | SE-2019-G02 |
| [ ]:草稿 | 当前版本： | 1.0 |
| [ ]:正式发布 | 作 者： | 高兴欣、王晨旭、倪嘉玲 |
| [ √ ]:正在修改 | 完成日期： | 2019年06月16日 |

**版本历史**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **版本/状态** | **作者** | **参与者** | **起止日期** | **备注** |
| 1.0 | 高兴欣、王晨旭、倪嘉玲 | 高兴欣、王晨旭、倪嘉玲 | 2019-06-01  至  2019-06-14 | 初步完成代码规范 |

目录

[2018 －2019 学年第 二 学期 1](#_Toc11426665)

[1](#_Toc11426666)

[霸笔笔记app 1](#_Toc11426667)

[代码规范 1](#_Toc11426668)

[版本历史 2](#_Toc11426669)

[1.Dart语言简介 4](#_Toc11426670)

[1.1Dart语言介绍 5](#_Toc11426671)

[① 注释的方式 5](#_Toc11426672)

[② 变量的声明 5](#_Toc11426673)

[③ 字符串的声明和使用 6](#_Toc11426674)

[④ 集合变量的声明 7](#_Toc11426675)

[⑤ 数字的处理 8](#_Toc11426676)

[⑥ 循环的格式 10](#_Toc11426677)

[⑦ 抛异常的方式 11](#_Toc11426678)

[⑧ 函数的写法 12](#_Toc11426679)

[⑨ 函数的可变参数 12](#_Toc11426680)

[⑩ 构造函数 13](#_Toc11426681)

[⑩① Getter 和 Setter 15](#_Toc11426682)

[⑩② 导包 16](#_Toc11426683)

# 1.Dart语言简介

Flutter是使用Dart语言开发的。

Dart语言是基于类的纯面向对象语言。

Dart 中的所有东西都是对象，包括数字、函数等，它们都继承自 Object，并且对象的默认值都是 null（包括数字）。

Dart 中类和接口是统一的，类就是接口。

Dart语法和java比较相似，熟悉java开发或者安卓开发会更容易理解Dart语言的语法规范。官宣： Dart语言的语法比JAVA更富有表现力。

总的来说，谷歌的Flutter既然使用Dart语言开发，说明Dart语言在某些方面还是有比其他语言更突出的优势的。安卓开发者应该考虑去熟悉一下。

## 1.1Dart语言介绍

### ① 注释的方式

Dart的文档注释除了有 /\*\* \*/外 ，还有 /// 。

三斜杠一般单行文档注释使用，多行时每行有个三斜杠效果和/\*\* \*/ 一样。

推荐使用///作为注释。

Dart的文档注释中可以有markdown的标记语法，有兴趣可以百度下使用方法。

### ② 变量的声明

可以使用var声明一个变量：

var str1 = 'abc';

也可以使用具体的类型声明：

String str1 = 'abc';

用final声明终态变量，不可二次赋值：

final String str1 = 'abc';

如果声明的变量没有赋值，就始终是null，成员变量也是null，DartVM不会为成员变量赋默认值。

int number; 声明量一个成员变量，但是此时number = null; 而不是0。因为number并没有初始化。

私有变量的声明：

变量前声明了下划线，代表是私有的，只能在本类中使用。如果未声明私有，就是公有的。

var \_list; 代表 \_list变量是私有的，只有当前class域可以调用。

需要注意的是： 变量名是 \_list，而不是list。

### ③ 字符串的声明和使用

字符串既可以用双引号声明，也可以用单引号声明。

不仅双引号中可以嵌套使用单引号，单引号中也可以嵌套使用双引号。

var str2 = '中国';

var str2 = "中国";

var str2 = ‘中国"春节"快到了’;

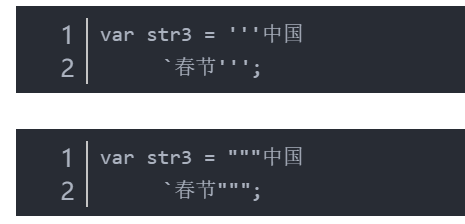
var str2 = "中国'春节'快到了";

如果要声明的字符串中有单引号，可以外层使用双引号，内层使用单引号。

如果要声明的字符串中有双引号，可以内层使用双引号，外层使用单引号。

一般用单引号声明变量，因为双引号出现在字符串中的频率高一些，所以外层使用单引号。

如果要声明多行的字符串，可以用三引号。所谓的三引号，就是三个单引号或者三个双引号。



这样声明出来的字符串就是换行的。

注意： 三引号的使用会包含编辑器自带的tab对齐标签，一般换行仍可以使用 \n 的形式。

字符串的拼接方式：

var str4 = 'hello' 'my' 'world';

var str4 = "hello" "my" "world";

var str4 = 'hello' + 'my' + “world”; // flutter中可以使用加号连接字符串。

字符串拼接除了用加号拼接，还可以直接写在一起。但是要防止有3个单引号或者3个双引号写在一起，写成了多行字符串形式，一般在要拼接的字符串中间加上空格。

字符串拼接变量的方式：

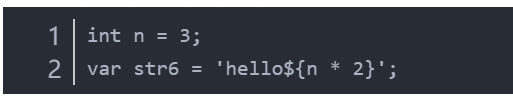
var str5 = 'hello$str1';

var str5 = ”$str1 hello“;

在字符串中，$ 符号后面加变量可以直接引用变量的值赋给字符串：var str5 的值是： helloabc。

如果字符串拼接变量的后面仍有字符串，可以加个空格隔开，也可以用{ }把 $str1 包裹起来。

字符串拼接表达式的方式：



字符串中使用表达式通过${}的方式 var str6的值是：hello6。

### ④ 集合变量的声明

#### List

List list1 = new List(); 这么写只能是空List。 而且这种写法不能在后面赋值。

List list2 = [1,3,5,7]; list.length 是4。

List<String> list = <String>['12', '13', '16']; 带泛型的List。

#### Map

Map map1 = new Map(); 空集合。

Map map2 = {'key1':3,'key2':5, 'key3':7}; Map的第二种写法。集合用 [ ] , 键值对用 { } 。

对于 map2：

map2['key4'] = 9; 如果map中没有key4，则新增了key4，map的长度加1，key4的值为9。

map2['key5'] == 0; 这是判断操作，由于map中没有key5，所以 map2['key5'] = null，表达式是false。

### ⑤ 数字的处理

#### 数字取整数值：

5.12.floor()舍的形式

5.12.round()四舍五入形式

5.12.ceil()入的形式

#### 字符串转整数：

int i = int.parse('5'); 可以转换的最大整数：-9223372036854775808 ~ 9223372036854775807 = -2^63 ~ 2^63-1

double d = double.parse('5.10'); 小数点后15位之后就不准确了。

#### 数字转字符串：

String sd = 3.14159.toStringAsFixed(2); 保留几位小数。15位后的double会有问题。

String sd2 = 13.14159.toStringAsPrecision(4); 保留几位数，包括整数部分的位数。

#### 数字的绝对值：

bool b = -8.2.abs() == 8.2; 定义布尔变量用 bool 。

#### 数字求商符号 ~/ ：

10 ~/ 4 = 2;

10 ~/ 3 = 1;

#### ??表达式：

var exp = exp1 ?? exp2; 如果exp1不是null，exp就等于exp1；如果exp1是null，exp就等于exp2。

这是对3元运算符的简写，exp ? a : b; 如果只是对exp判空操作，就可以用 ?? 表达式。

#### 重载操作符：

重载操作符就是让操作符在指定的作用域内有特殊的意义。

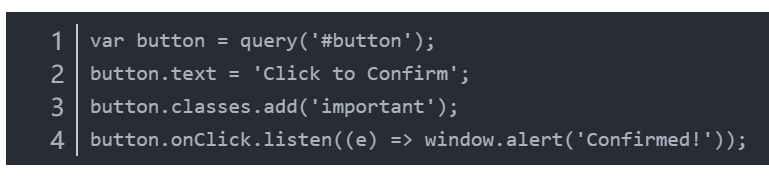
就是重新定义操作符的含义。

operator +(Vector other) {...}

(!=不可以进行重载；若重载了==，必须同时重载hashCode)

#### 级联操作符：

级联操作符就是对多行操作同一对象的代码进行简化。



简化成：

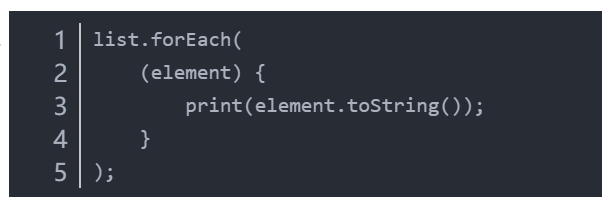


(级联操作符实验报错，请选择性相信)。

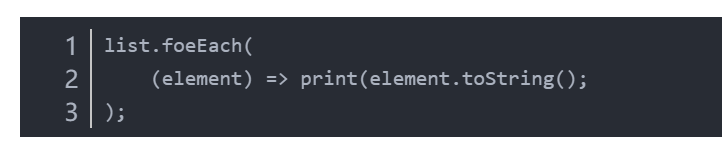
### ⑥ 循环的格式

for (int x = 0; x < list.length; x++) {...} 和java一样的格式。

for (var x in list2) {...} 简化的循环格式，遍历集合的形式。



函数中仅 return 一个表达式，则循环可以简化成：



（Switch语句中如果执行一个非空并且没有break等结束语句的case，就会报错 fall-through.）

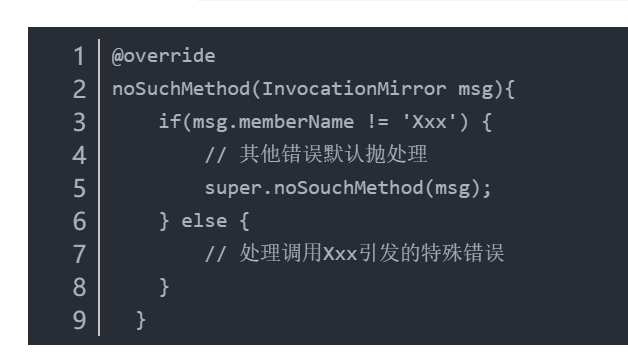
### ⑦ 抛异常的方式

throw new Exception('some Exception');

throw 'something wrong!'; 抛出一个字符串对象作为异常

手动处理 noSuchMethod 异常：

需要在类中覆盖noSuchMethod(InvocationMirror msg) { } 方法：



### ⑧ 函数的写法

函数是Function类型的对象，Function 有一个静态方法 apply 可以实现动态调用一个函数（当前版本尚未实现）。

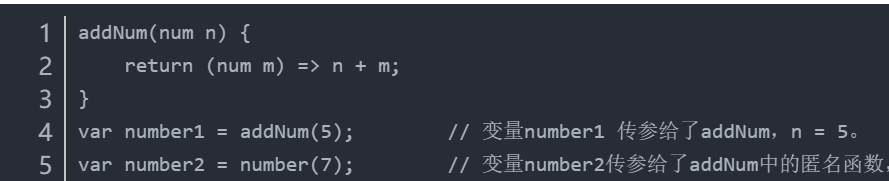
函数都有返回值，定义返回值的函数必须返回同类对象，未定义返回值的默认返回null。

如果函数中返回了一个对象，最好在函数名中声明返回类型和相关注释。

var sayHello = (name) => 'hello-$name'; 匿名函数 (name) { } 赋值给变量sayHello。

sayHello('张三'); 函数返回： hello-张三。

bool b = sayHello is Function; 判断类型用 is 关键字。所有函数都是Function对象，所以b = true；



### ⑨ 函数的可变参数

Dart语言允许参数可选。可选方式通过[]或{}指定。

固定参数: function(int a, String ,b) { } 直接 () 阔起来的参数，调用时必须传参。

可选无序参数：function(int a, {String back, int style, bool has}) { }

参数 a，为必传参数，不需要指定参数的名称。{ } 内有3个参数，可以选传，但是必须指定参数的名称，如：

function(4, back:'backcolor', has:false);。

#### 可选连续参数：

如果可选函数是用[]包裹的，就必须按照顺序指定参数值，允许后面的参数为空，但不允许参数不连续。

function(int a, [String back, int style, bool has]) { }

传参只能是：

function(4, back:'backcolor');

function(4, back:'backcolor', style:0);

function(4, back:'backcolor', style:0, has:false);

不可以是 function(4, back:'backcolor', has:false); 因为参数不连续了。

### ⑩ 构造函数

#### 标准构造函数：

对于MyDemo类, 有默认的唯一标准构造函数：MyDemo() { }

如果构造函数没有内容，可以直接分号结束：MyDemo();

带参数的构造函数：

MyDemo(String str) {

this.str = str;

}

如果构造函数只做了对成员变量初始化操作，可以简写成：

MyDemo(String this.str);

如果构造函数需要在方法体之前执行操作，可以用冒号，

MyDemo(var key)

: this.key = key,

super(key){

print('123');

}

#### 命名构造函数：

有明明构造函数就必须要先有标准构造函数。

命名构造函数可以有多个：

MyDemo.polar(int n) { }

MyDemo.pose(String str) { }

常量构造函数：

const Person.pose(Map map);

工厂构造函数：

factory Person(Map map) { }

命名构造函数不可以简化使用this引用对成员变量赋值，赋值方式是：

Person.pose(Map map) : width = map['s1'], y = map['s2'] { }

Person.polar(String str) : this.str = str { }

调用父类构造，先执行父类构造的方法

Person.pose(Map map) : super.pose(map) { }

Dart不支持多重构造函数，可以通过命名构造函数指向另一个命名构造函数，并部分传值：

Person.pose(Map map) : this(map, 0);

### ⑩① Getter 和 Setter

Dart中的字段默认有隐式get和set方法，但是仍然通过字段名调用，getXxx调用一般不使用，使用时需要先定义，并且可被子类覆盖。如果是final 或者const 则只有get没有set。

定义成员变量字符串 str ：

String str;

则str变量默认有getStr() 和setStr(String str) 方法，如果需要对str赋值，要通过 demo.str = '333'; 赋值。

如果没有显式定义getStr() 和setStr(String str),则不可以调用getStr() 和setStr(String str)，也不会被子类覆盖。

如果定义了getStr() 和setStr(String str)，则可以调用，并且可以被子类覆盖。

显式的getStr() 和setStr(String str) 和默认的getStr() 和setStr(String str) 是不同的概念。

如果是final 或者const 的成员变量， 则只有默认get没有默认set：

final String str1;

const String str2;

(const对象不能使用 new 创建)。

### ⑩② 导包

import 用于导入一个库，library 用于定义一个库。

#### 导入Dart标准库：

import 'dart:html';

#### 导入文件：

import 'lib/unittest.dart';

#### 导入包管理系统下的库：

import 'package:mylib/mylib.dart';

#### 如果两个库中有相同的方法名，可以通过在导入时指定前缀命名空间来区分：

import 'package:lib1/lib1.dart';

import 'package:lib2/lib2.dart' as lib2; // 给 lib2.dart 定义一个前缀命名空间

var element1 = new Element();

var element2 = new lib2.Element(); // 从定义的lib2中使用一个函数。

#### 部分导包：

show关键字指定需要导入的函数：

import 'package:lib1/lib1.dart' show foo, bar; 只导入 lib1 中的 foo 和 bar 这两个函数。

hide关键字指定不需要导入的函数，剩下的全导入：

import 'package:lib2/lib2.dart' hide foo; 导入 lib2 中除了 foo 之外的所有函数。

#### export 包导出：

可以用于将多个小库导出成一个大库。

如果使用show 参数，则代表只导出show所表明的函数；

如果使用hide 参数，则代表导出除了hide 所表明的函数外的所有函数。

export 'french.dart' show hello; 只导出 hello。

export 'french.dart' hide goodbye; 导出除goodbye之外的所有函数。

#### library和part 语句：

使用 library 加上一个标示符 定义当前库的名字，

如 ： library box2d;

库允许把代码写在多个文件中，只写在一个文件不利于维护。一般库都是写在多个文件中。

如果库的代码写在多个文件中，库的主文件就只用来包含其他文件，仅仅充当一个文件管理者，不包含任何方法。通过part语句指定子文件的路径，子文件通过part of语句表明属于哪个库。import也只能写在主库文件中

注意： 所有的import和part都只能写在主库文件中。子库文件使用part of 关键字，表明属于哪个库。

如：



库文件仅用于管理import和part子文件，子文件通过part of 指定属于哪个主