Build Your IT Skill

ណែនាំស្គាល់ពី DART(កំរិត *Hello World*)

រៀបរាងដោយៈ

គ្រូអាយធីចិត្តល្អ

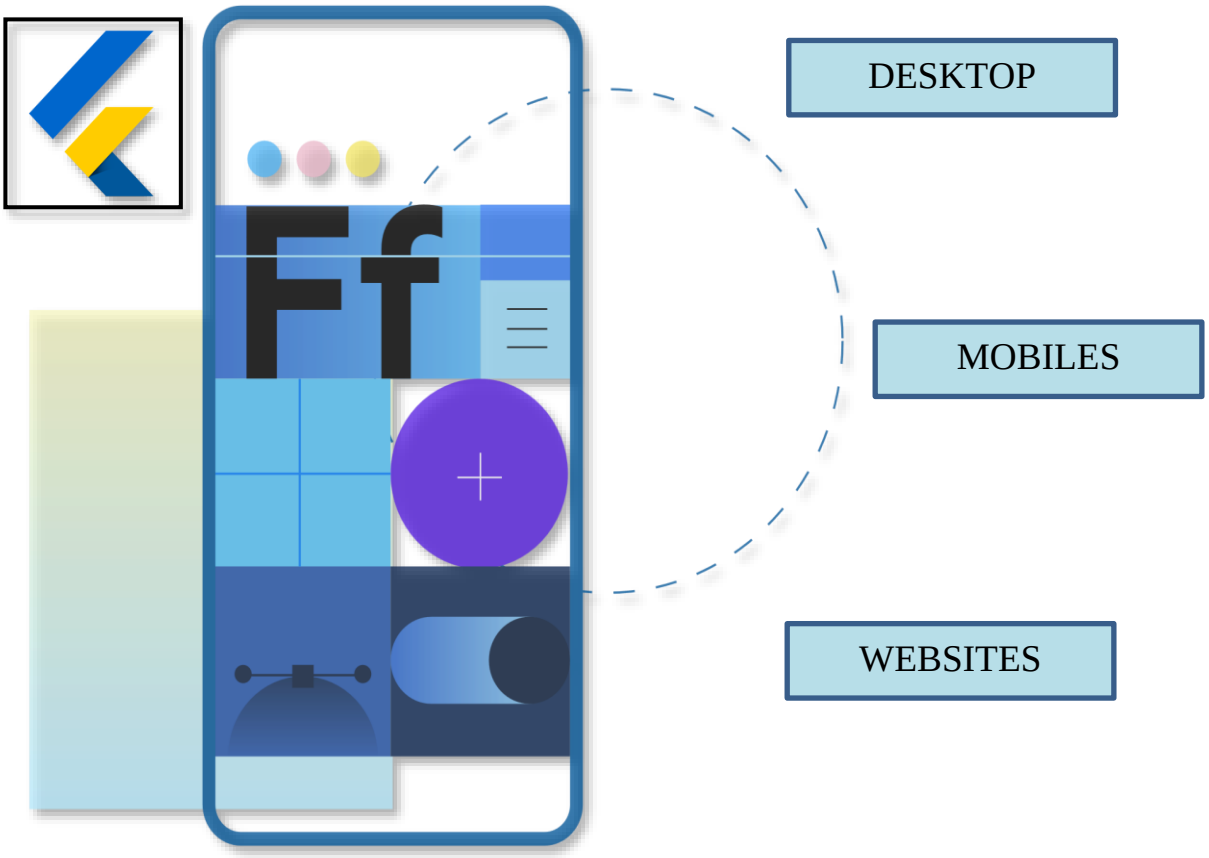
អនុបណ្ឌិត RUPP, TKU

(Software Engineering)

រៀបរៀងដោយៈ

Mr. Pha Phanna, Bs IT

របៀបតំឡើង Flutter



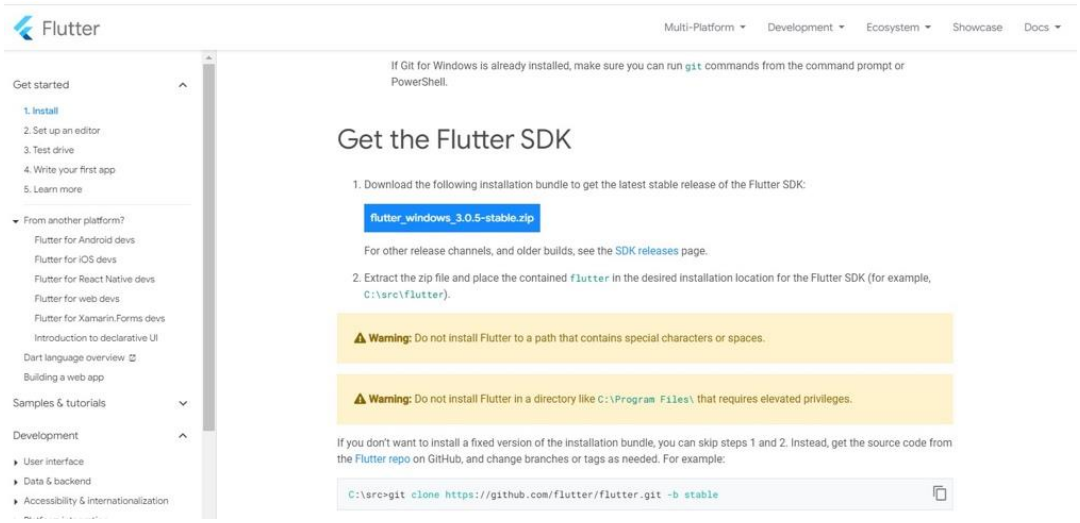
I. Requirement System hardware

- Linux , Windows , MacOS , Chrome OS
- RAM 8 GB up
- CPU Core i3 11th / core i5 8th up
- Storage drive C:/ use SSD or M.2

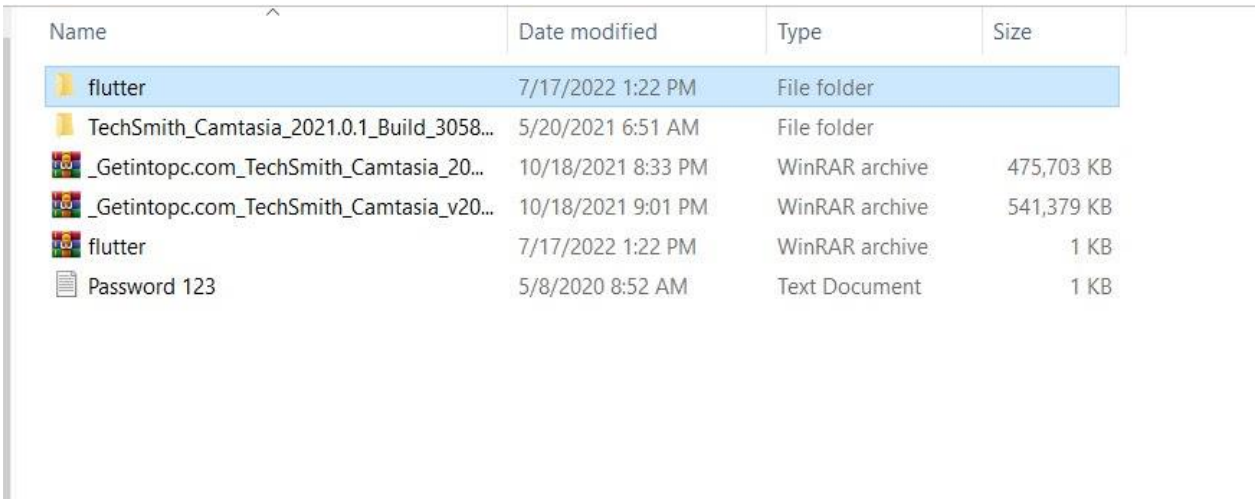
II. ដំបៀបSetup flutter on Windows

- IDE , Flutter SDK , android studio , git

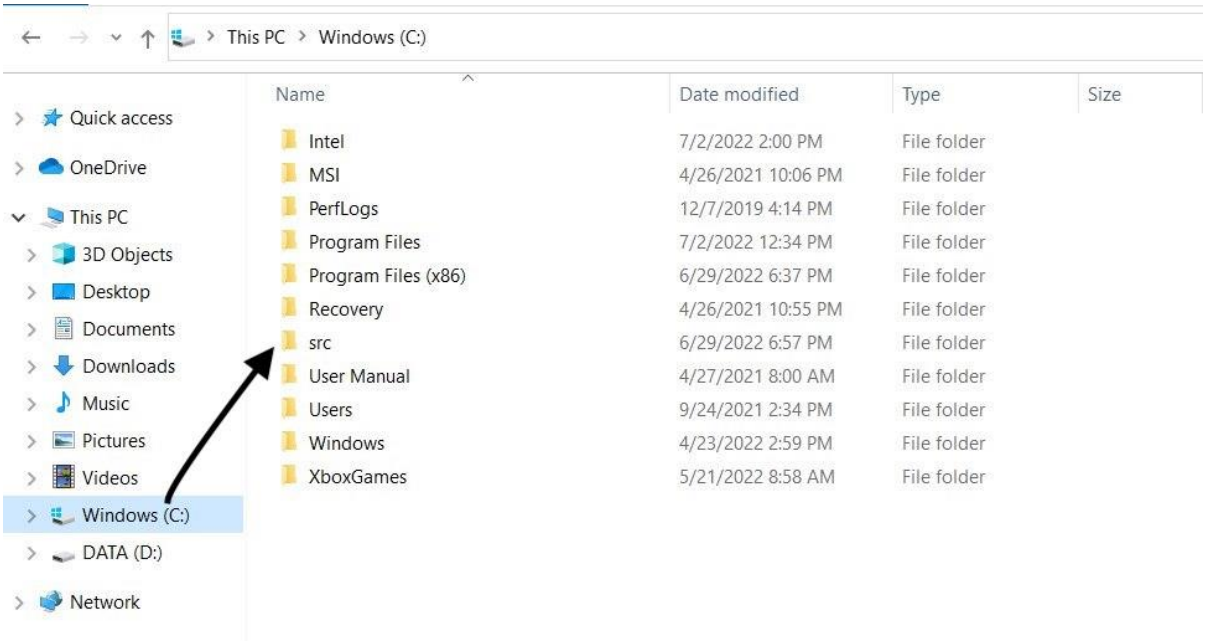
1.



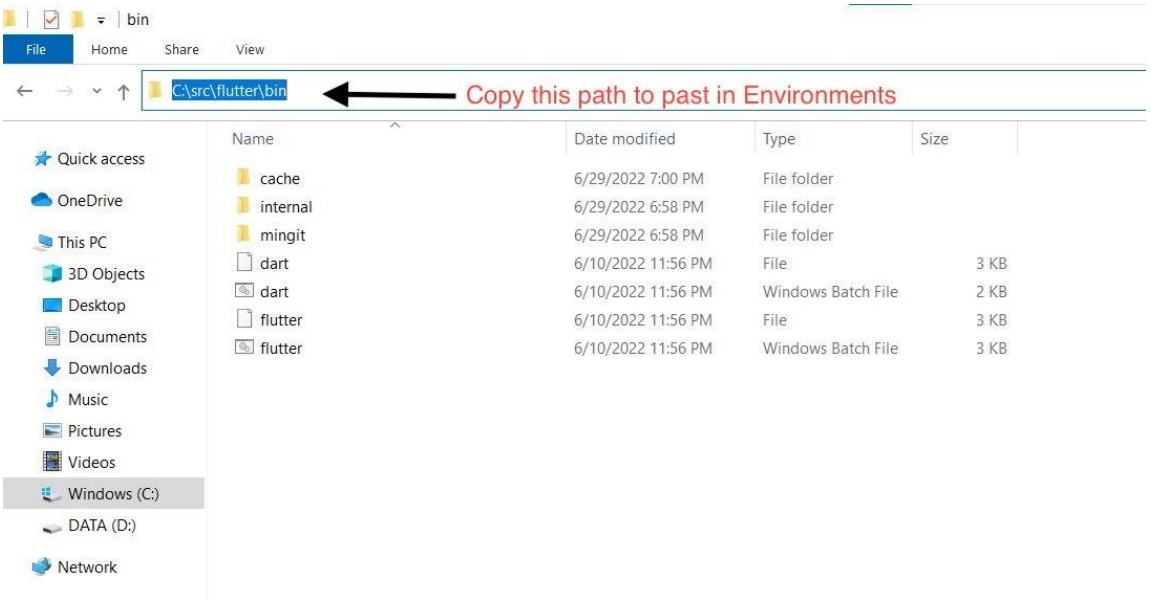
2.



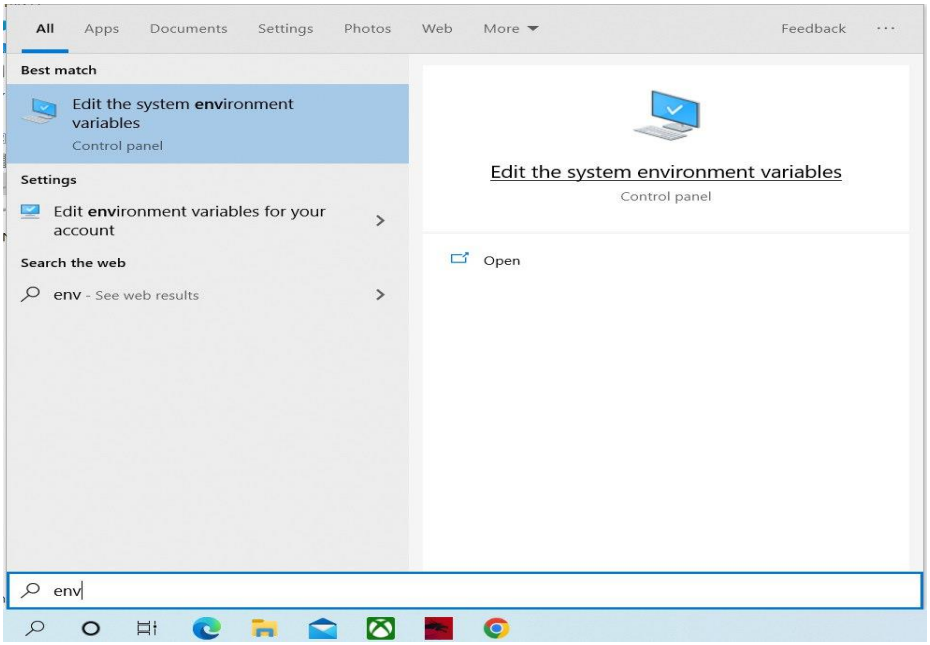
3.


This PC > Windows (C:)
Quick access
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows (C:)
DATA (D:)
Network
Name
Date modified
Type
Size
Intel 7/2/2022 2:00 PM File folder
MSI 4/26/2021 10:06 PM File folder
PerfLogs 12/7/2019 4:14 PM File folder
Program Files 7/2/2022 12:34 PM File folder
Program Files (x86) 6/29/2022 6:37 PM File folder
Recovery 4/26/2021 10:55 PM File folder
src 6/29/2022 6:57 PM File folder
User Manual 4/27/2021 8:00 AM File folder
Users 9/24/2021 2:34 PM File folder
Windows 4/23/2022 2:59 PM File folder
XboxGames 5/21/2022 8:58 AM File folder

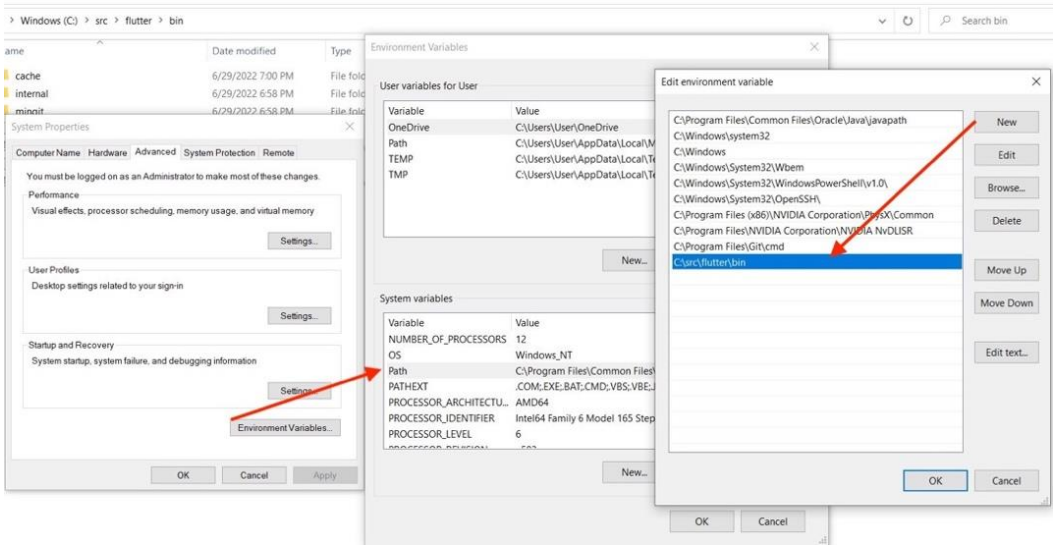
4.


bin
File
Home
Share
View
C:\src\flutter\bin
Copy this path to past in Environments
Quick access
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Windows (C:)
DATA (D:)
Network
Name
Date modified
Type
Size
cache 6/29/2022 7:00 PM File folder
internal 6/29/2022 6:58 PM File folder
mingit 6/29/2022 6:58 PM File folder
dart 6/10/2022 11:56 PM File 3 KB
dart 6/10/2022 11:56 PM Windows Batch File 2 KB
flutter 6/10/2022 11:56 PM File 3 KB
flutter 6/10/2022 11:56 PM Windows Batch File 3 KB

5.


All
Apps
Documents
Settings
Photos
Web
More
Feedback
Best match
Edit the system environment variables
Control panel
Settings
Edit environment variables for your account
Search the web
env - See web results
Edit the system environment variables
Control panel
Open
env

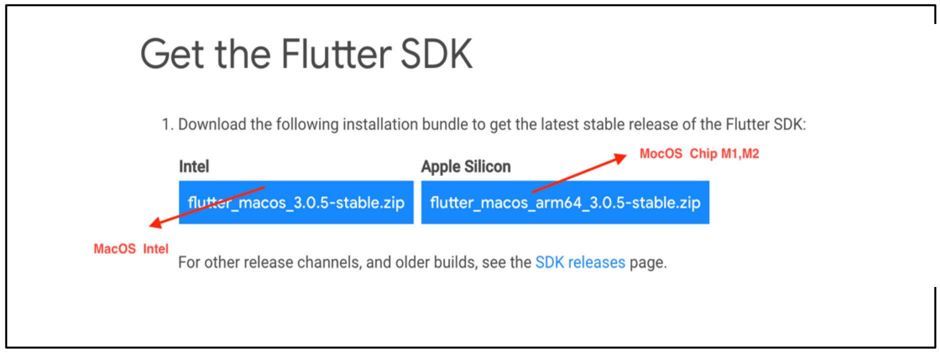
6.


Windows (C:) > src > flutter > bin
Search bin
Date modified
Type
cache
internal
System Properties
Computer Name
Hardware
Advanced
System Protection
Remote
Performance
Visual effects, processor scheduling, memory usage, and virtual memory
Settings...
User Profiles
Desktop settings related to your sign-in
Settings...
Startup and Recovery
System startup, system failure, and debugging information
Settings...
Environment Variables...
OK
Cancel
Apply
Environment Variables
User variables for User
Variable
Value
OneDrive
C:\Users\User\OneDrive
Path
TEMP
TMP
New...
System variables
NUMBER_OF_PROCESSORS
12
OS
Windows_NT
PATHEXT
PROCESSOR_ARCHITECTU...
AMD64
PROCESSOR_IDENTIFIER
PROCESSOR_LEVEL
6
Edit environment variable
C:\Program Files\Common Files\Oracle\Java\javapath
C:\Windows\system32
C:\Windows
C:\Windows\System32\Wbem
C:\Windows\System32\WindowsPowerShell\v1.0\
C:\Windows\System32\OpenSSH\
C:\Program Files\Git\cmd
C:\src\flutter\bin
New
Edit
Browse...
Delete
Move Up
Move Down
Edit text...
OK
Cancel

Finish

III. <u>របៀប Setup flutter on MacOS</u>

1.



2. add flutter path

this step recommended follow step in official website (Flutter.dev)

https://docs.flutter.dev/get-started/install/macos

7. Install X-code from Appstore

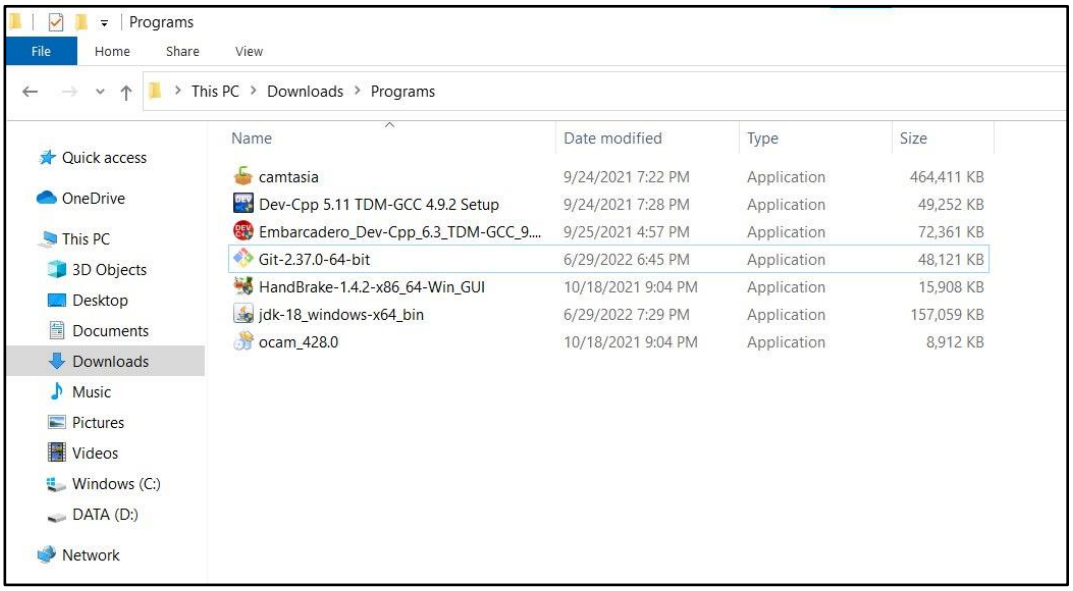
IV. <u>IDE</u>

we can use VS code , Android studio, IntelliJ IDEA

V. <u>Extension</u>

Flutter , Dart , code runner (Vs code) ..etc.

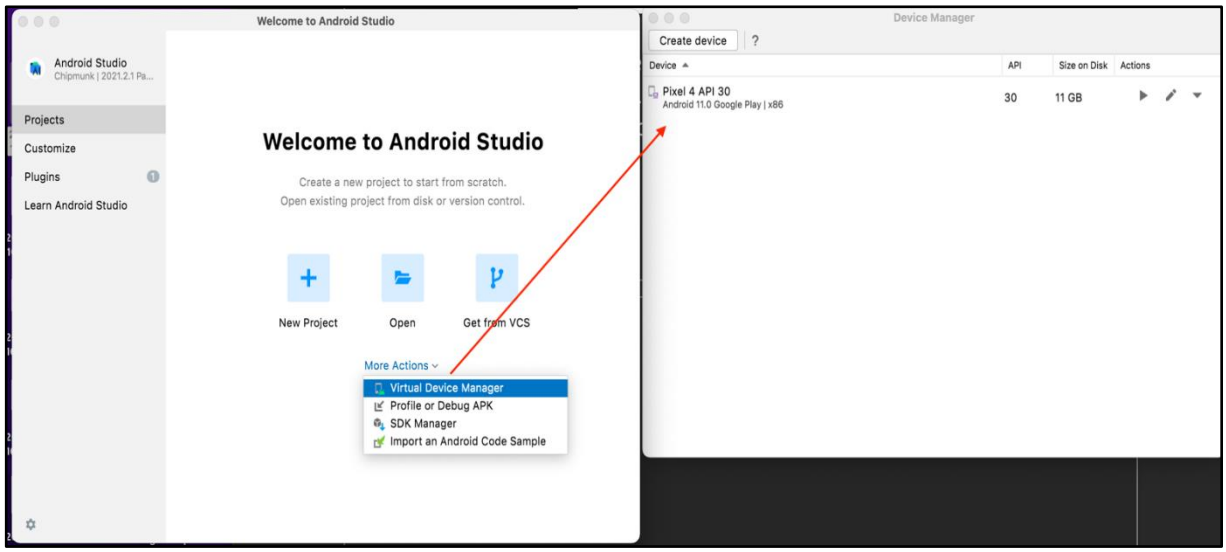
VI. <u>Set up git</u>



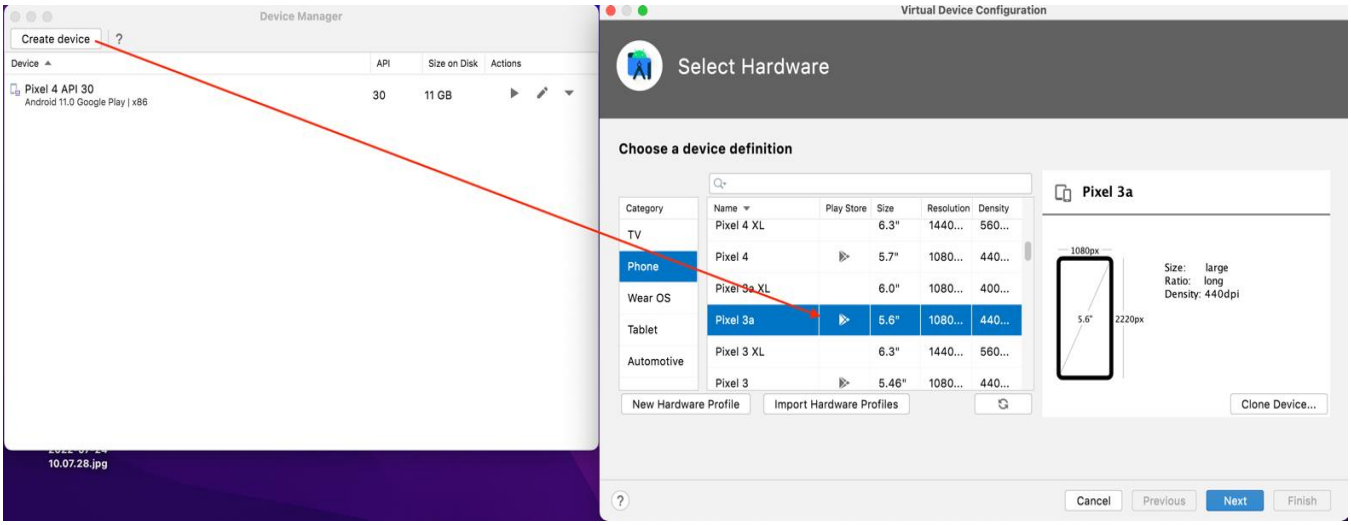
For next step Install , you can press [next .....] until done setup.

VII. Setup Android studio (emulator)

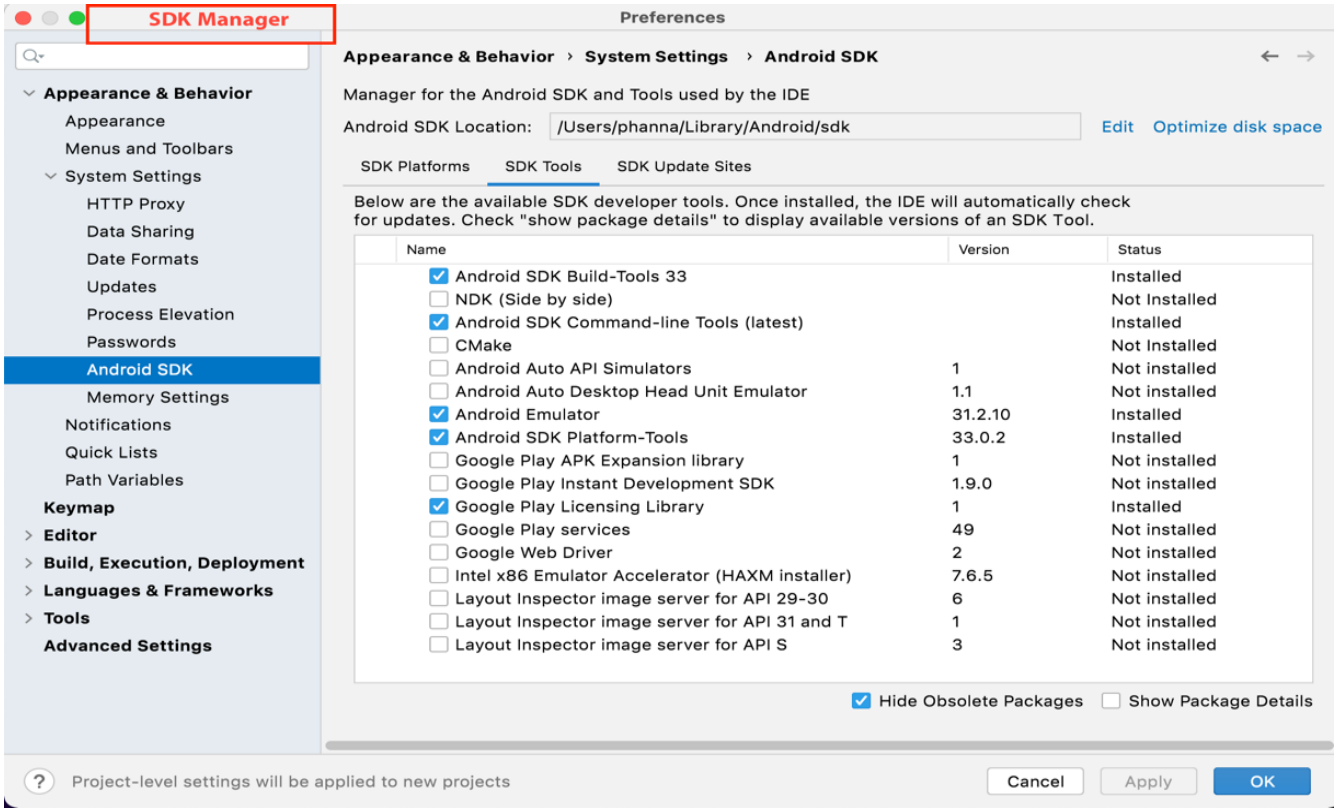
1.



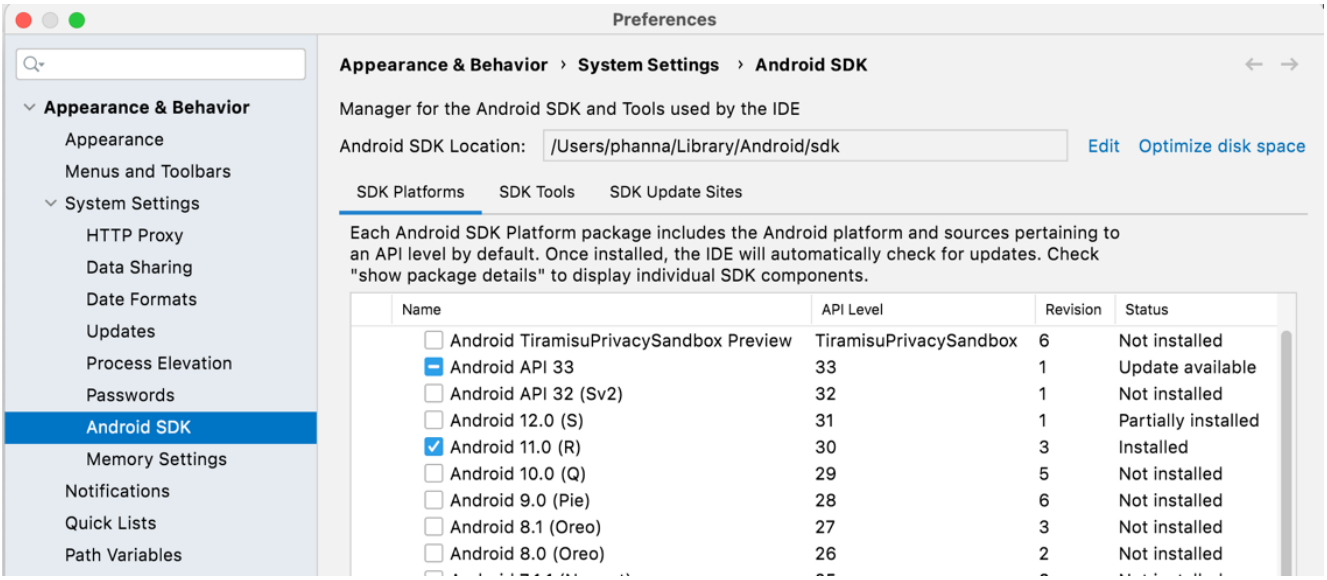
2.


Device Manager
Create device
Device
API
Size on Disk
Actions
Pixel 4 API 30
Android 11.0 Google Play | x86
30
11 GB
Virtual Device Configuration
Select Hardware
Choose a device definition
Category
TV
Phone
Wear OS
Tablet
Automotive
Name
Play Store
Size
Resolution
Density
Pixel 4 XL
Pixel 4
Pixel 3a XL
Pixel 3a
Pixel 3 XL
Pixel 3
New Hardware Profile
Import Hardware Profiles
Pixel 3a
1080px
2220px
5.6"
Size: large
Ratio: long
Density: 440dpi
Clone Device...
Cancel
Previous
Next
Finish

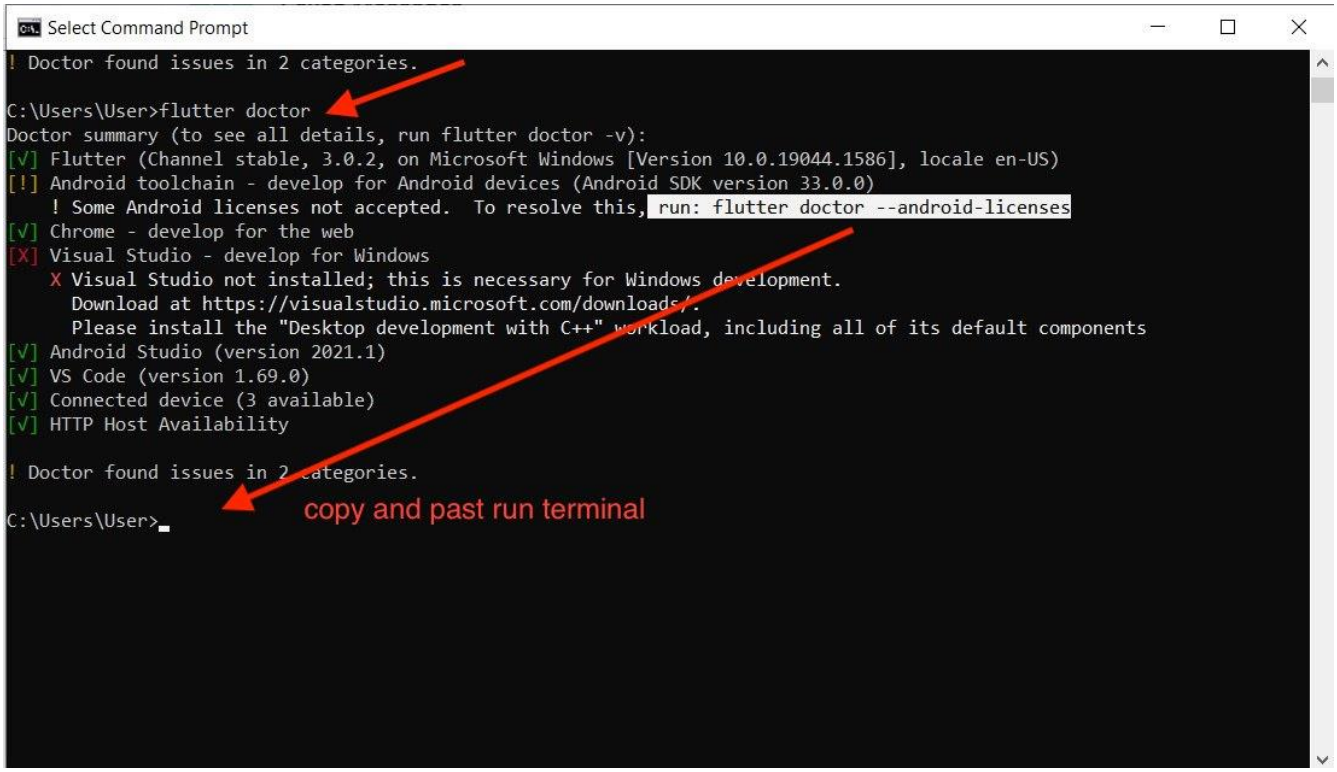
3.


SDK Manager
Preferences
Appearance & Behavior
Appearance
Menus and Toolbars
System Settings
HTTP Proxy
Data Sharing
Date Formats
Updates
Process Elevation
Passwords
Android SDK
Memory Settings
Notifications
Quick Lists
Path Variables
Keymap
Editor
Build, Execution, Deployment
Languages & Frameworks
Tools
Advanced Settings
Appearance & Behavior › System Settings › Android SDK
Manager for the Android SDK and Tools used by the IDE
Android SDK Location: /Users/phanna/Library/Android/sdk
Edit
Optimize disk space
SDK Platforms
SDK Tools
SDK Update Sites
Below are the available SDK developer tools. Once installed, the IDE will automatically check for updates. Check "show package details" to display available versions of an SDK Tool.
Name
Version
Status
Android SDK Build-Tools 33 Installed
NDK (Side by side) Not Installed
Android SDK Command-line Tools (latest) Installed
CMake Not Installed
Android Auto API Simulators 1 Not installed
Android Auto Desktop Head Unit Emulator 1.1 Not installed
Android Emulator 31.2.10 Installed
Android SDK Platform-Tools 33.0.2 Installed
Google Play APK Expansion library 1 Not installed
Google Play Instant Development SDK 1.9.0 Not installed
Google Play Licensing Library 1 Installed
Google Play services 49 Not installed
Google Web Driver 2 Not installed
Intel x86 Emulator Accelerator (HAXM installer) 7.6.5 Not installed
Layout Inspector image server for API 29-30 6 Not installed
Layout Inspector image server for API 31 and T 1 Not installed
Layout Inspector image server for API S 3 Not installed
Hide Obsolete Packages
Show Package Details
Project-level settings will be applied to new projects
Cancel
Apply
OK

4.


Preferences
Appearance & Behavior
Appearance
Menus and Toolbars
System Settings
HTTP Proxy
Data Sharing
Date Formats
Updates
Process Elevation
Passwords
Android SDK
Memory Settings
Notifications
Quick Lists
Path Variables
Appearance & Behavior › System Settings › Android SDK
Manager for the Android SDK and Tools used by the IDE
Android SDK Location: /Users/phanna/Library/Android/sdk
Edit
Optimize disk space
SDK Platforms
SDK Tools
SDK Update Sites
Each Android SDK Platform package includes the Android platform and sources pertaining to an API level by default. Once installed, the IDE will automatically check for updates. Check "show package details" to display individual SDK components.
Name
API Level
Revision
Status
Android TiramisuPrivacySandbox Preview
TiramisuPrivacySandbox
6
Not installed
Android API 33
33
1
Update available
Android API 32 (Sv2)
32
1
Not installed
Android 12.0 (S)
31
1
Partially installed
Android 11.0 (R)
30
3
Installed
Android 10.0 (Q)
29
5
Not installed
Android 9.0 (Pie)
28
6
Not installed
Android 8.1 (Oreo)
27
3
Not installed
Android 8.0 (Oreo)
26
2
Not installed

5.


Select Command Prompt
! Doctor found issues in 2 categories.
C:\Users\User>flutter doctor
Doctor summary (to see all details, run flutter doctor -v):
[√] Flutter (Channel stable, 3.0.2, on Microsoft Windows [Version 10.0.19044.1586], locale en-US)
[!] Android toolchain - develop for Android devices (Android SDK version 33.0.0)
! Some Android licenses not accepted. To resolve this, run: flutter doctor --android-licenses
[√] Chrome - develop for the web
[X] Visual Studio - develop for Windows
X Visual Studio not installed; this is necessary for Windows development.
Download at https://visualstudio.microsoft.com/downloads/.
Please install the "Desktop development with C++" workload, including all of its default components
[√] Android Studio (version 2021.1)
[√] VS Code (version 1.69.0)
[√] Connected device (3 available)
[√] HTTP Host Availability
! Doctor found issues in 2 categories.
C:\Users\User>
copy and past run terminal

Finish

Build Your IT Skill

ណែនាំស្គាល់ពី Dart(កំរិត *Hello World*)

រៀបរាងដោយៈ

គ្រូអោយធីចិត្តល្អ

អនុបណ្ឌិត RUPP, TKU

(Software Engineering)

រៀបរៀងដោយៈ

Mr. Pha Phanna, Bs IT

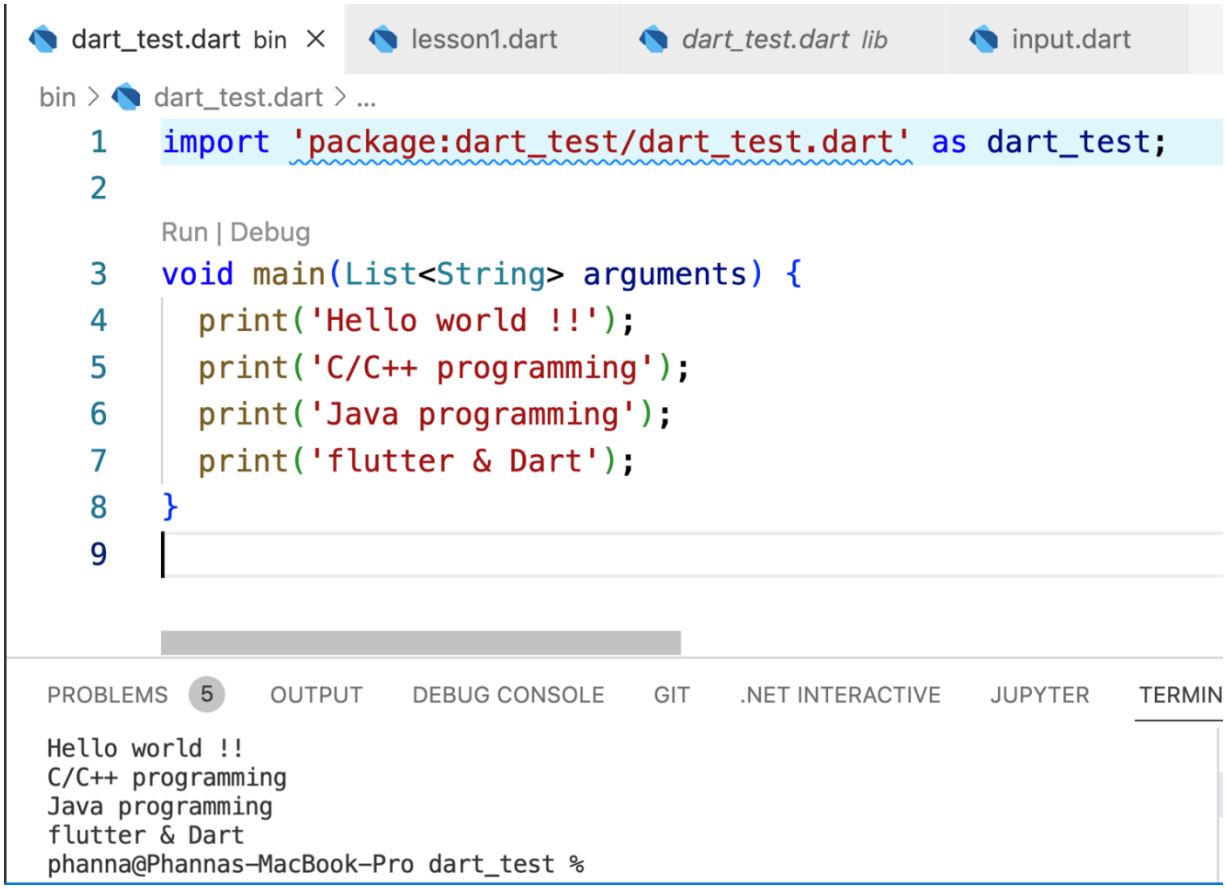
ណែនាំស្តាល់ពី Dart Language

I. Introduction Dart

Dart Programming ត្រូវបានបង្កើតដោយ Google នៅឆ្នាំ 2011 ដែលដំបូងប្រើសម្រាប់សរសេរ Web Service តែក្រោយមកទៀតគេបាន Develop អោយប្រើប្រាស់ សម្រាប់ build Web , Mobile ,Desktop ជាមួយ Flutter SDK

II. ការបង្ហាញទិន្នន័យ និង បញ្ចូល របស់ Dart I/O

ឧទាហរណ៍ ១ ៖



ឧទាហរណ៍ ២

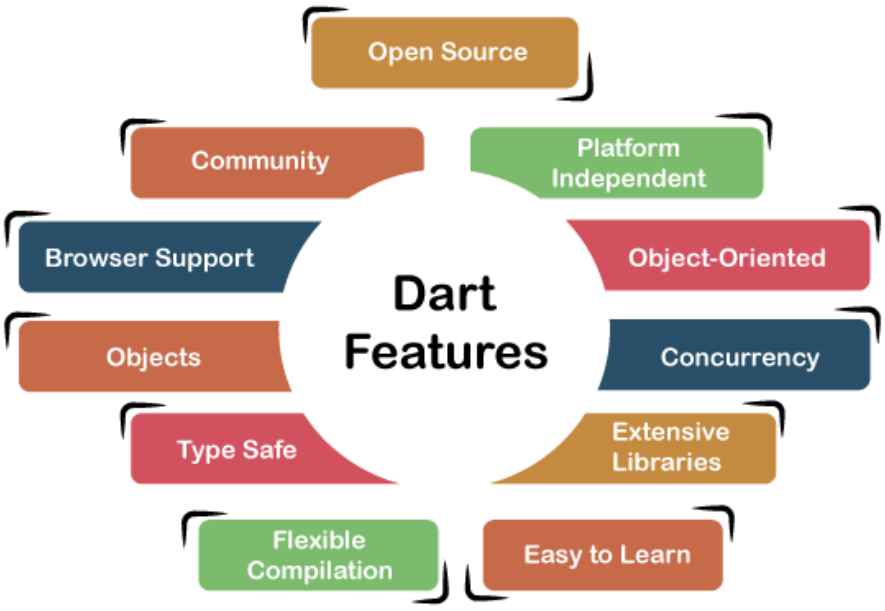
bin > lesson1.dart > main

Run | Debug

```dart
void main(List<String> args) {
  int id = 12;
  String name = "Thida";
  String sex = "Female";
  dynamic positoin = "Software Developer";
  var salary = 400;
  print('ID :$id');
  print('Name :$name');
  print('Gender :$sex');
  print('Position:$positoin');
  print('Salary:$salary');
}

```

PROBLEMS 5 OUTPUT DEBUG CONSOLE GIT .NET INTERACTIVE JUPYTER

```
Name :Thida
Gender :Female
Position:Software Developer
Salary:400
phanna@Phannas-MacBook-Pro dart_test %
```



ឧទាហរណ៍ ៣

input.dart

```dart
import 'dart:io';

Run | Debug
void main(List<String> args) {
  int? id;
  String? name;
  double? score;

  stdout.write('Input Id:');
  id = int.parse(stdin.readLineSync()!);
  stdout.write('Input Name:');
  name = stdin.readLineSync();
  stdout.write('Input score:');
  score = double.parse(stdin.readLineSync()!);
  print('========= Output ==========');
  print('Id=$id');
  print('Name=$name');
  print('Score=$score');
}
```

លំហាត់អនុវត្តន៍

១)ចូរសរសេរកូដអោយគេអាចបញ្ចូលពត៌មានដូចជា៖ id(String), name(String), sex(String) *និង* Address(String) *បន្ទាប់បង្ហាញពត៌មានទាំងនោះចេញមកក្រៅវិញ?*

២) ចូរសរសេរកូដបញ្ចូលពត៌មានដូចជា៖ Code(int), Name(String), qty(int), price(float) *បន្ទាប់មក គណនារកតំលៃ* totald(float) *និង* totalr(float) *បន្ទាប់បង្ហាញពត៌មានទាំងនោះចេញមកក្រៅវិញ?*

៣) ចូរសរសេរកូដបញ្ចូលពត៌មានដូចជា៖ score1(float), score2(float), score3(float), score4(float), score5(float)*បន្ទាប់មក គណនារកតំលៃ* total(float) *និង* average(float) *បន្ទាប់បង្ហាញពត៌មាន ទាំងនោះចេញមកក្រៅវិញ?*

លក្ខណ្ឌរបស់ Dart

I. ណែនាំស្គាល់ពី Condition របស់ Dart

input.dart ex1.dart

bin > controlStatement > ex1.dart > main

```dart
Run | Debug
void main(List<String> args) {
  int? x = 54;
  int? y = 34;

  // ===== ១ លក្ខខណ្ឌ =======
  if (x > y) {
    print('X = $x is largest than Y');
  }
  // ===== ២ លក្ខខណ្ឌ =======
  if (x > y) {
    print('X =$x biggest');
  } else {
    print('Y=$y biggest');
  }
  // ===== ៣ លក្ខខណ្ឌ =======
  int number = 0;
  if (number < 0) {
    print('$number is Negative number');
  } else if (number > 0) {
    print('$number is  positive number');
  } else {
    print('number is $number');
  }
}
```

- Ternary

ternary.dart — dart_application_1

ternary.dart

bin > controlStatement > ternary.dart > ...

```dart
Run | Debug
void main(List<String> args) {
  int x = 31, y = 5;
  var max = x > y ? x : y;
  print('$max is larest');
  //============ String =======
  String code = 'basic';
  print("level code ${code.toLowerCase() == 'bsic' ? code : 'Flutter'}");
}
```

❖ Switch Statement

```
Run | Debug
void main(List<String> args) {
  int codeDay;
  stdout.write('Please Choose 1-7:');
  codeDay = int.parse(stdin.readLineSync()!);
  switch (codeDay) {
    case 1:
      print('Monday');
      break;
    case 2:
      print('Tuesday');
      break;
    case 3:
      print('Wednesday');
      break;
    case 4:
      print('Thursday');
      break;
    case 5:
      print('Friday');
      break;
    case 6:
      print('Saturday');
      break;
    case 7:
      print('Sunday');
      break;
  }
}

```

លំហាត់អនុវត្តន៏

១) ចូរសរសេរកូដសម្រាប់អោយគេអាចរក និទេស(Grade), *ពិន្ទុសរុប*(Total) *និង មធ្យមភាគ តាមលក្ខណ្ឌដូចខាងក្រោម៖*

Total Score	Grade
90-100	"A"
80-90	"B"
70-80	"C"
60-70	"D"
50-60	"E"
0-50	"F"

២) ចូរសរសេរកូដសម្រាប់អោយគេអាចរក ចំនួនសរុបប្រើប្រាស់ភ្លើង(Total) *និងតំលៃសរុប* (Payment) *តាមលក្ខណ្ឌដូចខាងក្រោម៖*

Total Use	Payment(៛)
1-10	៥០០៛
10-20	៥៥០៛
20-30	៦០០៛
30-40	៦៥០៛
40-50	៧០០៛
50-60	៧៥០៛
60-	៨០០៛

Loop របស់ Dart

ឧទាហរណ៍១(ប្រភេទនៃ Loop)

bin > loopStatement > loop1.dart > main

```dart
import 'dart:io';

Run | Debug
void main(List<String> args) {
  // ====== forloop =======
  for (int i = 1; i < 10; i = i + 2) {
    stdout.write('$i ');
  }
  print('\n========');
  for (int i = 20; i >= 1; i = i - 2) {
    stdout.write('$i ');
  }

  // ===== while loop ======
  int i = 1;
  while (i <= 10) {
    stdout.write('$i ');
    i = i + 2;
  }

  // ===== do while loop =====
  print(' do while loop ');
  i = 2;
  do {
    print(i);
    i = i + 2;
  } while (i <= 15);
}
```

ឧទាហរណ៍២ (For Each Loop)

bin > loopStatement > loop_foreach.dart > ...

Run | Debug

```dart
void main() {
  List names = ['Kim long', 'Chanthu', 'lisa', 'dalin'];
  for (var temp in names) {
    print('Name:$temp');
  }
}
```

PROBLEMS 7 OUTPUT DEBUG CONSOLE GIT TERMINAL

```
Name:Kim long
Name:Chanthu
Name:lisa
Name:dalin
phanna@Phannas-MacBook-Pro dart_application_1 %
```

Code

Code

លំហាត់អនុវត្តន៍

១. ចូរបង្ហាញទិន្នន័យដោយប្រើប្រាស់ Loop ណាមួយក៏បាន

- ✓ 0 , 3 , 5 ,…..N
- ✓ 2 , 3 , 4 ,…..N
- ✓ N,……3, 2, 1

២. ចូរគណនា Loop ដូចខាងក្រោម៖

- 1+ 2 + 3 + …..N
- 2 + 4 + 6 + ….N
- Cos(1) + Cos(2) + Cos(3) ….+ Cos(N)

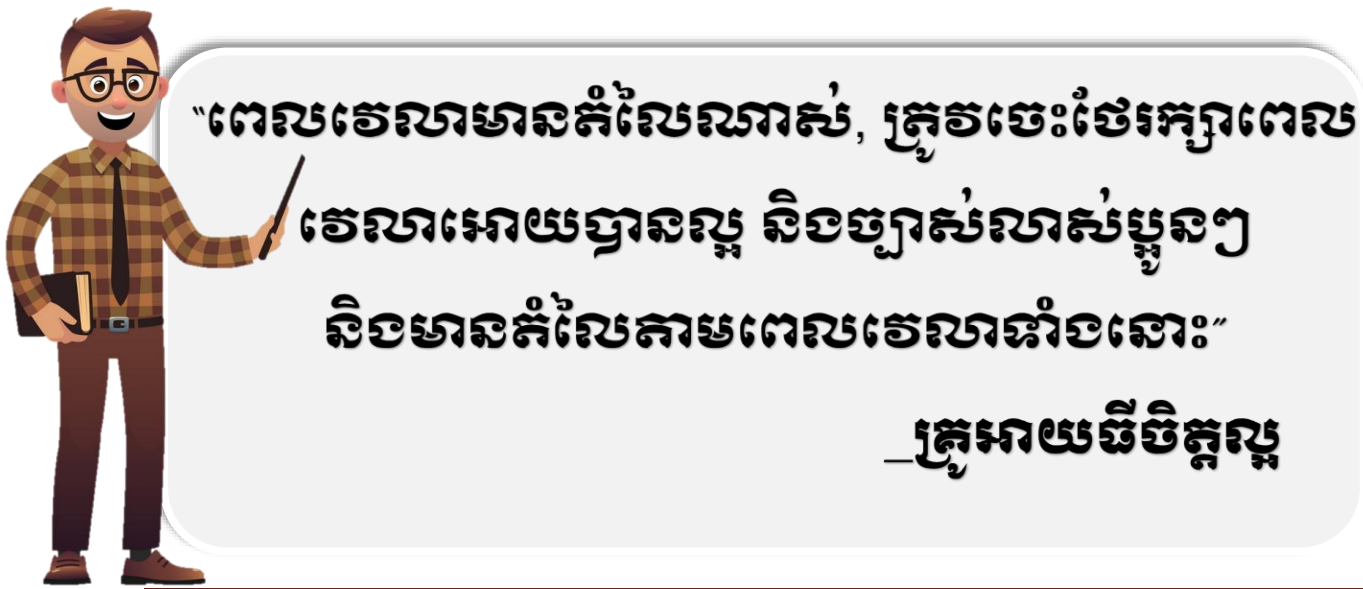
Function របស់ Dart

1. ប្រភេទ នៃ Function

   - Non- Return Function

ឧទាហរណ៍:

```
Run | Debug
void main(List<String> args) {
  sum2(32, 44);
  sum1();
}

//  Non- return funtion not parameter
void sum1() {
  int x = 34, y = 76;
  print('X+Y=${x + y}');
}

//  Non- return funtion have parameter
void sum2(int a, int b) {
  print('A+B=${a + b}');
}

```



- Return function

ឧទាហរណ៍:

```dart
void main(List<String> args) {
  var x = sum3();
  print('Sum3=$x');
  var y = sum4(34, 5);
  print('Sum4=$y');
  var sumOfFun = sum3()! + sum4(4, 8);
  print('Sum of Funtion=$sumOfFun');
}

int? sum3() {
  sum4(3, 4);
  int x = 4, y = 6;
  return x + y;
}

//return funtion have  Paramater [ int , double , bool ,class ,String ]
double sum4(int x, int y) {
  return (x + y).toDouble();
}
```

- FunctionExpressions

```dart
void main(List<String> args) {
  printText();
  printTextParam('Become to Flutter');
  print(printText2());
  print(printTextParam2('Next level is Flutter'));
}

void printText() => print('Hello Dart Programming');
void printTextParam(String text) => print(text);
String printText2() => 'Start with Dart';
String printTextParam2(String nextLevel) => nextLevel;
```

```
PROBLEMS 7   OUTPUT   DEBUG CONSOLE   GIT   TERMINAL

Hello Dart Programming
Become to Flutter
Start with Dart
Next level is Flutter
phanna@Phannas-MacBook-Pro dart_application_1 %
```

- Optional Parameter

```dart
void main(List<String> args) {
  student(43, 'Chhanon');
  print('==============');
  student(3, 'KimSeang', sex: 'Female');
}

void student(int id, String name, {String? sex = 'male'}) {
  print('ID=$id');
  print('Name=$name');
  print('Sex=$sex');
}
```

PROBLEMS 7 OUTPUT DEBUG CONSOLE GIT TERMINAL

```
ID=43
Name=Chhanon
Sex=male
==============
ID=3
Name=KimSeang
Sex=Female
phanna@Phannas-MacBook-Pro dart_application_1 %
```

II. Collection

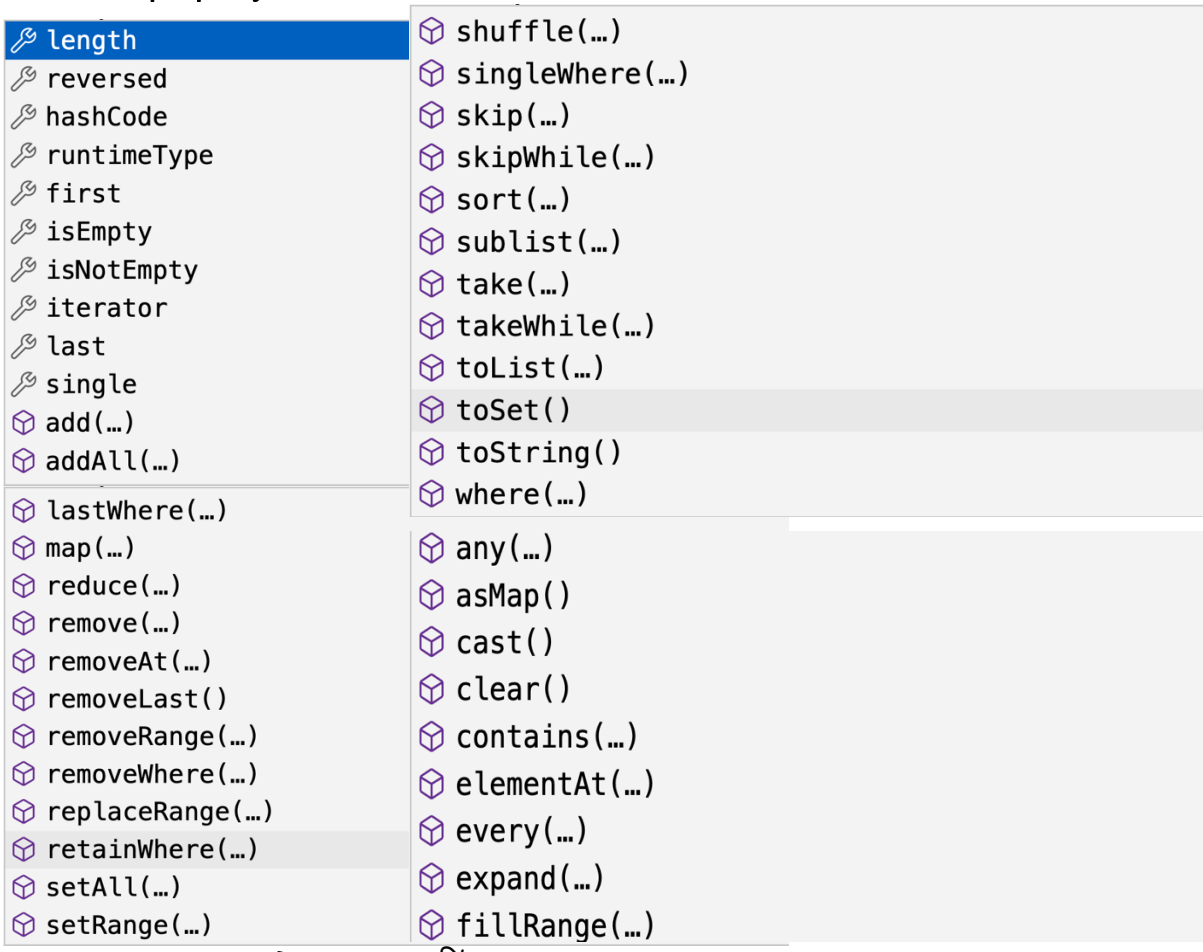
I. List

bin > collection > ex1.dart > ...

Run | Debug

```dart
void main(List<String> args) {
  // syntax 1
  List list = [12, 54, 634, 66, 34, 7, 'Animal'];
  print(list);
  // or
  for (var temp in list) {
    print('Value=$temp');
  }
  // syntax 2
  List<int> listInt = [23, 555];
  print(listInt);

  List<String> name = ['channa', 'nary', 'Davuth'];
  for (var names in name) {
    print('name=$names');
  }
}
```

Collection,property & Method :



Method & property ដែលជួបច្រើនមានដូចជា ៖

- **property**

length , first ,last ,isEmpty ,isNotEmpty ….

- **Method**

- .add(…) : add ធាតុចូលម្តងមួយ
- .addAll(…) : add ធាតុចូលម្តងទាំងអស់
- .firstWhere(…) : filter រកធាតុដែលដូច និង យកដំបូងគេ
- .map(…) : បំលែងធាតុទៅជា map ដែលប្រើ Key and Value
- .remove(…) : លុបធាតុណាមួយចេញពី list

- .sort(…) : តម្រៀបធាតុ
- .toString(…) : បំលែង data ទៅជា string
- .where(…) : រកធាតុណាមួយ
- .every(…) : យកធាតុទាំងអស់ដែលមានដូច
- .indexWhere(….) :ចាប់យក index ធាតុទាំងអស់ដែលមានដូច
- .contains(…) : យកធាតុទាំងអស់ដែលមានតួអក្សរដូច
- .clear(…) : លុបធាតុទាំងអស់ចេញពី list
- .toList(…) : បំលែងធាតុទៅជា list
- ….

II. Map

ឧទាហរណ៍ ១

```dart
void main(List<String> args) {
  Map map = {'id': 123, 'name': 'Thida', 'score': 500};

  print(map);

  Map<String, dynamic> map1 = {'id': 123, 'name': 'Thida', 'score': 500};
  print(map1);

  List<Map<String, dynamic>> list = [
    {'id': 123, 'name': 'Thida', 'score': 500},
    {'id': 143, 'name': 'Vuthy', 'score': 400}
  ];
  print(list);
  for (var temp in list) {
    temp.forEach((key, value) {
      print('$key : $value');
    });
  }
}

```

ឧទាហរណ៍ ២

bin > map > test1.dart > ...

Run | Debug

```dart
void main() {
  var details = {};
  details['Usrname'] = 'admin';
  details['Password'] = 'admin@123';
  print(details);
}
```

PROBLEMS OUTPUT DEBUG CONSOLE GIT TERMINAL

```
{Usrname: admin, Password: admin@123}
phanna@Phannas-MacBook-Pro dart_leasson %
```

ឧទាហរណ៍ ៣

bin > map > test2.dart > main

```dart
void main(List<String> args) {
  List<Map<String, dynamic>> listMap = [
    {'name': 'thida'},
    {'name': 'Sothea'},
    {'name': 'Mr.Long'},
    {'name': 'Chanra'},
    {'name': 'KimSeng'},
    {'name': 'phanit'}
  ];

  listMap.removeAt(1);

  for (var element in listMap) {
    print(element);
  }
}
```

PROBLEMS OUTPUT DEBUG CONSOLE GIT TERMINAL

```
{name: thida}
{name: Mr.Long}
{name: Chanra}
{name: KimSeng}
{name: phanit}
phanna@Phannas-MacBook-Pro dart_leasson %
```

លំហាត់អនុវត្តន៍

ចូរយក Example3 ទៅធ្វើជា Menu ដូចខាងក្រោម ៖

```
========= Menu =======
1. Add
2. Display
3. Search
4. Update
5. Remove
6. Sort
```

III. Collection With Class

ឧទាហរណ៍

bin > map > student_model.dart > Student

```dart
class Student {
  late int id;
  late String name;
  late String gender;
  late double score;
  Student(
      {required this.id,
      required this.name,
      required this.gender,
      required this.score});
  void Output() {
    print('$id\t$name\t$gender\t$score');
    print('---------------------------------');
  }
}

Run | Debug
void main(List<String> args) {
  List<Student> stuList = [
    Student(id: 1001, name: 'Angelika', gender: 'F', score: 70.9),
    Student(id: 1002, name: 'ARita', gender: 'M', score: 80)
  ];
  for (var temp in stuList) {
    temp.Output();
  }
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE GIT TERMINAL

```
1001    Angelika        F       70.9
---------------------------------
1002    ARita   M       80.0
---------------------------------
phanna@Phannas-MacBook-Pro dart_leasson %
```

III. ASYNCHRONOUS

Asyncronous គឺជា Function មួយដែលធ្វើរអាយ ដំណើរការ Code មិនធ្វើការមុន និង ក្រោយគ្នា ពោលគឺបញ្ចប់នៅ Funtion ដែលមាន asyn & await រួចហើយទើប បន្តទៅមុខទៀត។

ជាទូទៅ function នេះគេប្រើច្រើនជាមួយ Local database, cloud data, api service ,...។

❖ NOT USE ASYN & AWAIT

asyn.dart

bin > map > asyn.dart > main

```dart
Future delayedPrint(int seconds, String msg) {
  final duration = Duration(seconds: seconds);
  return Future.delayed(duration).then((value) => msg);
}

Run | Debug
main() {
  print('Life');
  delayedPrint(3, "Is").then((status) {
    print(status);
  });
  print('Good');
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE GIT TERMINAL

```
Life
Good
Is
phanna@Phannas-MacBook-Pro dart_leasson %
```

❖ USE ASYN & AWAIT

ឧទាហរណ៍១៖

asyn.dart

bin > map > asyn.dart > main

```dart
Future delayedPrint(int seconds, String msg) {
  final duration = Duration(seconds: seconds);
  return Future.delayed(duration).then((value) => msg);
}

Run | Debug
main() async {
  print('Life');
  await delayedPrint(3, "Is").then((status) {
    print(status);
  });
  print('Good');
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE GIT TERMINAL

```
Life
Is
Good
phanna@Phannas-MacBook-Pro dart_leasson %
```

ឧទាហរណ៍២៖

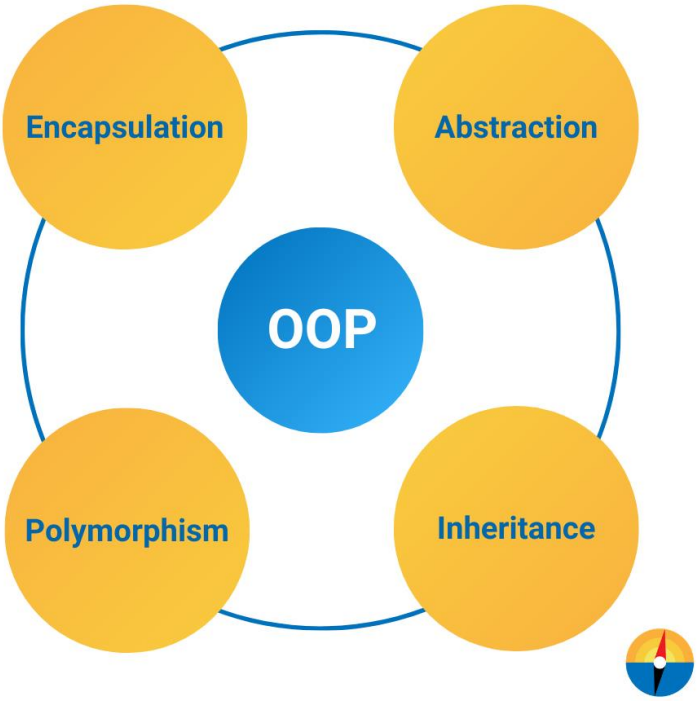
bin > map > asyn2.dart > ...

```dart
import 'dart:async';

Future<String> firstAsync() async {
  await Future.delayed(const Duration(seconds: 2));
  return "First!";
}

Future<String> secondAsync() async {
  await Future.delayed(const Duration(seconds: 2));
  return "Second!";
}

Future<String> thirdAsync() async {
  await Future.delayed(const Duration(seconds: 2));
  return "Third!";
}

Run | Debug
void main() async {
  var f = await firstAsync();
  print(f);
  var s = await secondAsync();
  print(s);
  var t = await thirdAsync();
  print(t);
  print('done');
}
```

ណែនាំស្គាល់ពី OOP

(Object oriented Programming)

ណែនាំស្គាល់ OOP (OBJECT-ORIENTED PROGRAMMING)



"តស៊ូការពលំបាករៀនបន្តិចទើបលទ្ធផលទទួលបានមានតំលៃបំផុតក្នុងជីវិតអ្នករៀនម្ចាក់...!"

_គ្រូអាយធីចិត្តល្អ

I. Encapsulation

❖ Class and Object

bin > OOP > class_object.dart > Person > Person

```dart
class Person {
  int? id;
  String? name;
  String? sex;
  int? age;
  Person(
      {required this.id,
      required this.name,
      required this.sex,
      required this.age});
  Person.newInstances();
  void output() {
    print('$id\t$name\t$sex\t$age');
  }
}

Run | Debug
void main(List<String> args) {
  Person ps1 = Person.newInstances();
  ps1.output();
  print('====================');
  Person ps2 = Person(id: 12, name: 'Vuthy', sex: 'Male', age: 23);
  ps2.output();
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE GIT TERMINAL

```
null    null    null    null
====================
12      Vuthy   Male    23
phanna@Phannas-MacBook-Pro dart_leasson %
```

❖ Property get() , set()

```dart
class Product {
  int? code;
  String? name;
  int? qty;
  double? price;
  double? total() {
    return price! * qty!;
  }

  void setCode(int code) {
    this.code = code;
  }

  void setName(String name) {
    this.name = name;
  }

  void setPrice(double price) {
    this.price = price;
  }

  void setQty(int qty) {
    this.qty = qty;
  }

  void displayItem() {
    print('$code\t$name\t$qty\t$price\$\t${total()}\$');
  }
}

Run | Debug
void main(List<String> args) {
  Product pro = Product();
  pro.setCode(1001);
  pro.setName('Coca');
  pro.setQty(12);
  pro.setPrice(1.5);
  pro.displayItem();
}
```

```
PROBLEMS 1   OUTPUT   DEBUG CONSOLE   GIT   TERMINAL

1001    Coca    12      1.5$    18.0$
phanna@Phannas-MacBook-Pro dart_leasson %
```

លំហាត់អនុវត្តន៍

1). ចូរបង្កើត Class មួយដែលមានឈ្មោះ Employee ដែលផ្ទុក Data member ៣ គឺ id(String), Name(String), Sex(String) & Salary(double) និង មាន function ចំនួន ២ គឺ void Input() និង void Output() ដោយប្រើប្រាស់នូវ Console Application បន្ទាប់មកបង្កើតនូវ Object ចំនួនពីរ គឺ objemp1 and objemp2ដែលប្រើប្រាស់នូវ Class នោះបាន?

II. Inheritance

❖ super class

class_object.dart ×

bin > OOP > class_object.dart > main

```dart
class Person {
  int? id;
  String? name;
  String? sex;
  int? age;
  Person(
      {required this.id,
      required this.name,
      required this.sex,
      required this.age});
  Person.newInstances();
  void output() {
    print('$id\t$name\t$sex\t$age');
  }
}
```

❖ sub class

class_object.dart | sudent_model.dart

bin > OOP > sudent_model.dart > ...

```dart
import 'class_object.dart';

class Student extends Person {
  double? score;
  Student({this.score, int? id, String? name, String? sex, int? age})
      : super(id: id, name: name, sex: sex, age: age);
  Student.newInstances() : super.newInstances();
  @override
  void output() {
    super.output();
    print('Score:$score');
  }
}

Run | Debug
void main(List<String> args) {
  Student stu =
      Student(id: 1002, name: 'Monica', age: 20, sex: 'Female', score: 80.6);
  stu.output();
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE GIT TERMINAL

```
1002    Monica  Female  20
Score:80.6
phanna@Phannas-MacBook-Pro dart_leasson %
```

Interfaces & Abstract class

❖ Interfaces

- Not have key word **interfaces**
- Use key **implements** to implement from super class
- In super class can have data member & function member

ឧទាហរណ៍ ១៖

implement_and_abstract.dart

bin > OOP > implement_and_abstract.dart > Item

```dart
class Product {
  int? code;
  String? name;
  int? qty;
  double? price;
  void addProduct() {}
}

class Item implements Product {
  void updateProduct() {}
  void showProduct() {
    print('$code\t$name\t$qty\t$price');
  }

  @override
  int? code;

  @override
  String? name;

  @override
  double? price;

  @override
  int? qty;

  @override
  void addProduct() {}
}
```

❖ **Abstract**

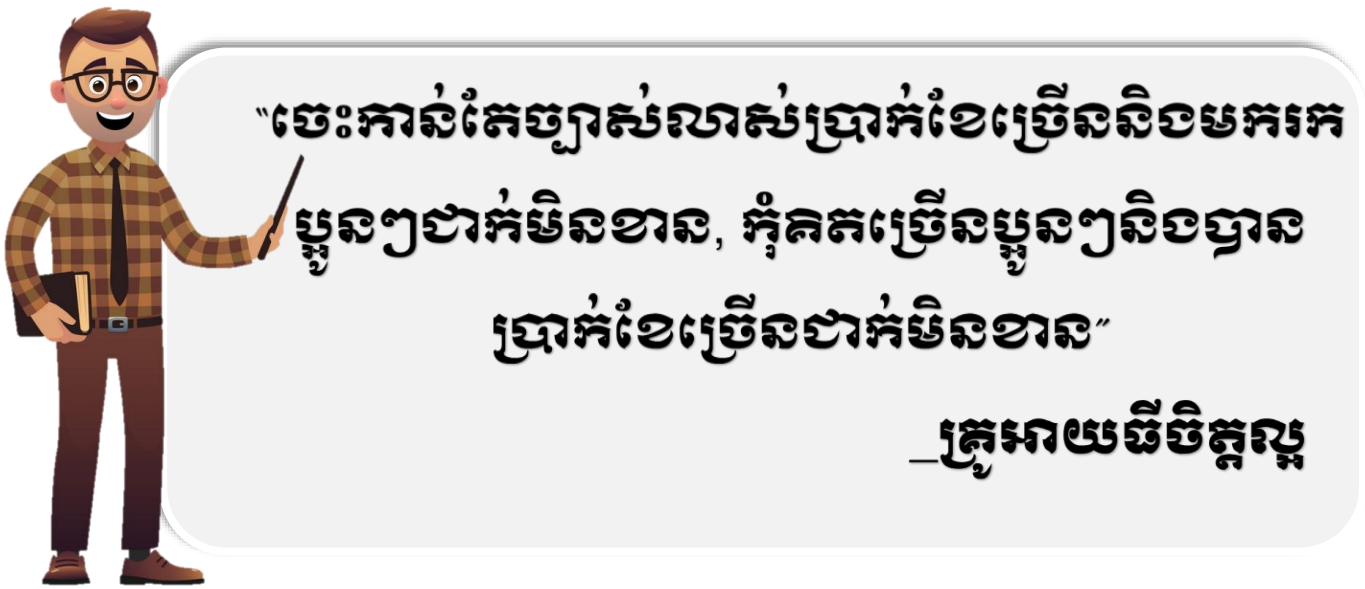
- Use key word **abstract**
- Use key **extent** to implement from super class
- In **abstract class** have abstract function , but function not has body

ឧទាហរណ៍

implement_and_abstract.dart

bin > OOP > implement_and_abstract.dart > Item

```dart
abstract class Product {
  void addProduct();
  void updateProduct();
  void showProduct();
}

class Item extends Product {
  int? code;
  String? name;
  int? qty;
  double? price;

  @override
  void addProduct() {
    // TODO: implement addProduct    TODO: implement addProduct
  }

  @override
  void updateProduct() {
    // TODO: implement updateProduct    TODO: implement updateProduct
  }

  @override
  void showProduct() {
    // TODO: implement showProduct    TODO: implement showProduct
  }
}

```



III. POLYMORPHISM (POLY + MORPHISM)

ឧទាហរណ៍ៈ

poly.dart

bin > OOP > poly.dart > main

```dart
class Car {
  void driving() {
    print("driving car 1");
  }
}

class Honda extends Car {
  //override method overrides generic driving method
  @override
  void driving() {
    print("driving car 2");
    super.driving(); //calls generic driving method
  }
}

Run | Debug
void main() {
  Honda car1 = Honda();
  car1.driving();
}
```

PROBLEMS 1 OUTPUT DEBUG CONSOLE GIT TERMINAL

```
driving car 2
driving car 1
phanna@Phannas-MacBook-Pro dart_leasson %
```

Learn more :

Dart: https://dart.dev/

Flutter : https://flutter.dev/

Habit can make you learn fast