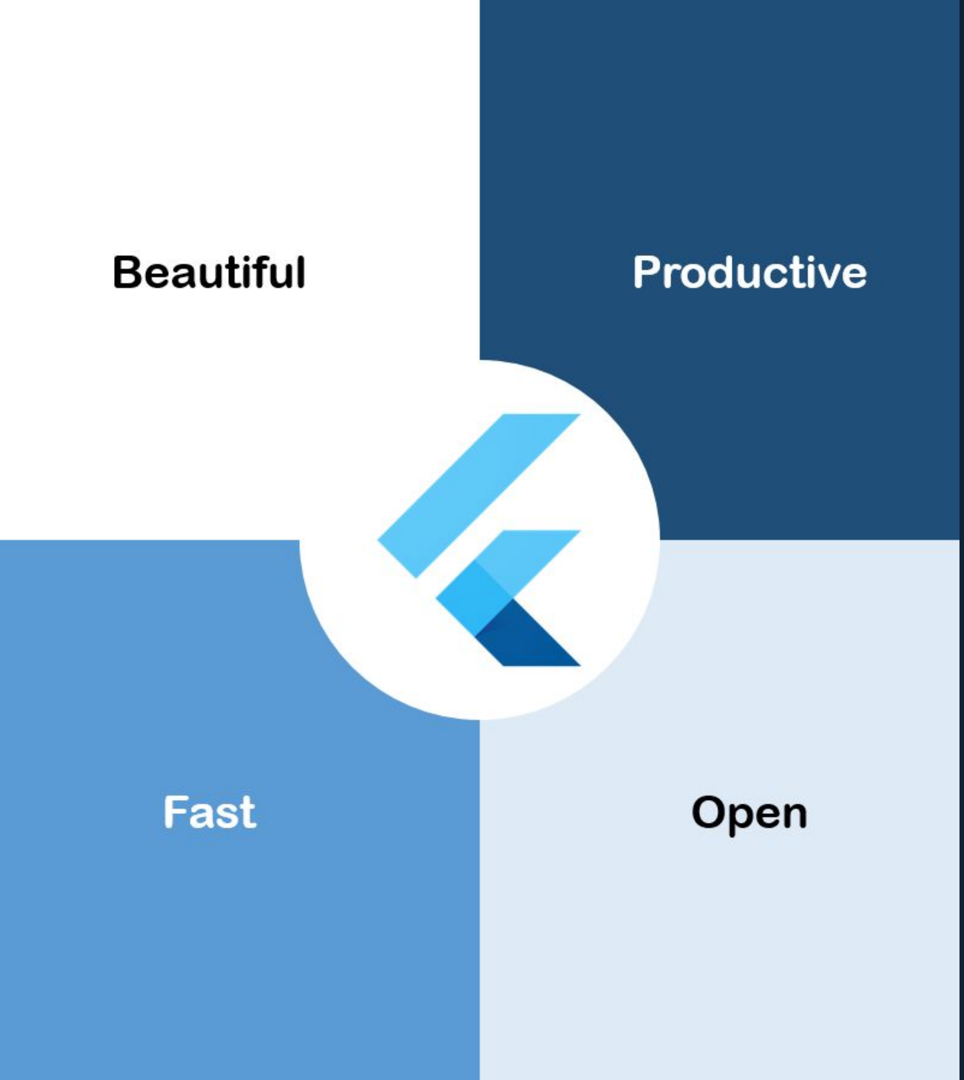
VentureDive

Flutter Training

Setup and Dart basics

VentureDive

Why Flutter?



Key Characteristics

Beautiful: Builtin support for material and cupertino widgets

Fast: Widgets are rendered onta a Skia canvas

Productive: Stateful hotreload

Open: Opensource and thousands of packages

VentureDive

Learning A New Language!

DartPad

Was initially created to let you play with Dart.

Now uses Flutter web support to let you play with Flutter code in your browser.

Your First Dart Code

```dart
main() {
    print("Hello World!");
}
```

Data Types

- Numbers (Integer, Double)
- Strings
- Booleans
- Lists
- Maps
- Dynamic (optionally typed language)

Variables & Constants

```
var name = 'Smith';
String name = 'Smith';
int num = 10;
dynamic x = "tom";
```

Final and Const

```
final val1 = 12;
const pi = 3.14;
```

Operators

Arithmetic Operators: + - * / ~/ % ++ --

Equality and Relational Operators: ==, !=, <=, >=, <, >

Assignment Operators: =, ??=, –=, *=, /=

Logical Operators: &&, ||, !

condition ? expr1 : expr2

expr1 ?? expr2

Loops

```
● for (var i = 0; i < 5; i++) {}
● for (var prop in obj) {}
● while(num >=1) {}
● do {
        print(n);
        n--;
    }
   while(n>=0);
```

Conditional Statements

```
if(boolean_expression){}
else if (boolean_expression2) {
   // if the expression2 evaluates
    //to true
}
else {
   // statement(s) will execute if the
    //Boolean expression is false.
}
```

```
switch(variable_expression) {
   case constant_expr1: {
      // statements;
   }
   break;

   default: {
      //statements;
   }
   break;
}
```

String Interpolation

```
void main() {

   String str1 = "hello";

   String str2 = ‘world’;

   String res = str1+str2;



   print("The concatenated string : ${res}" );

}
```

Lists

```
  var lst = new List(3);
```

```
   lst[0] = 12;
```

Growable List

```
var num_list = [1,2,3];
var lst = new List();
lst.add(12);
```

Map

Using Map Literal

```
var identifier = { key1:value1, key2:value2 [,…..,key_n:value_n] }
```

Map Constructor

```
var details = new Map();
   details['Usrname'] = 'admin';
```

Functions

```
void functionName(123,"this is a string") {

   //statements

 }
```

Optional parameters

```
test_param(n1,[s1])
```

Optional named parameters

```
test_param(n1,{s1,s2})
```

Lambda functions

printMsg()=>print("hello");

Classes

Declare a class

```
class class_name {

   <fields>

   <getters/setters>

   <constructors>

   <functions>

 }
```

Instantiating

```
var obj = new Car("Engine 1")
```

Accessing

```
//accessing an attribute

obj.field_name

//accessing a function

obj.function_name()
```

Named Constructor

```
class Car {

   Car() {
print("Non-parameterized constructor invoked");
}

   Car.namedConst(String engine) {
print("The engine is : ${engine}");
}

}
```

Objects

The cascade operator (..)

The cascade operator can be used as a shorthand in cases where there is a sequence of invocations.

```
void main() {
   new Student()
   ..testMethod1()
   ..testMethod2();
}
```

Sets and Queues

Set represents a collection of objects in which each object can occur only once.

```
Set numberSet = new  Set();
Set numberSet = new Set.from([12,13,14]);
```

A HashSet is an unordered hash-table based Set implementation.

```
Set numberSet = new  HashSet();
```

A Queue is a collection that can be manipulated at both ends.

```
 Queue queue = new Queue();
   queue.add(10);
   queue.add(20);
   queue.add(30);
```

```
numQ.addFirst(400); //at the beginning of a Queue
numQ.addLast(400);  //at the end of Queue
```

Exception Handling

The try / on / catch Blocks

```
try {
      res = x ~/ y;
  }
on IntegerDivisionByZeroException catch(e) {
     print(e);
  }
finally {
     print('Finally block executed');
   }
```

VentureDive

Setup

What you need

- Android SDK
- Xcode
- Flutter SDK

Editors

- Android Studio
- VS Code

VentureDive

Hello World!

VentureDive

Take home assignment

Read text from a file and find words that appear most in a line in the file.
(i) finding the highest frequency word(s) in each line
(ii) finding lines in the file whose "highest frequency words" is the greatest value among all lines.
Print the result in the following format:

> *The following words have the highest word frequency per line:*
> *["word1"] (appears in line #)*
> *["word2", "word3"] (appears in line #)*

Assignment starter files

Assignment submission

Upload your code on github and submit it’s link on the Google chat group.

VentureDive

Thank you

Upcoming: Flutter basics