Full-Stack Service Programming

Lecture 5
Dart & Flutter 심화 기술의 이해

2023. 09. 01

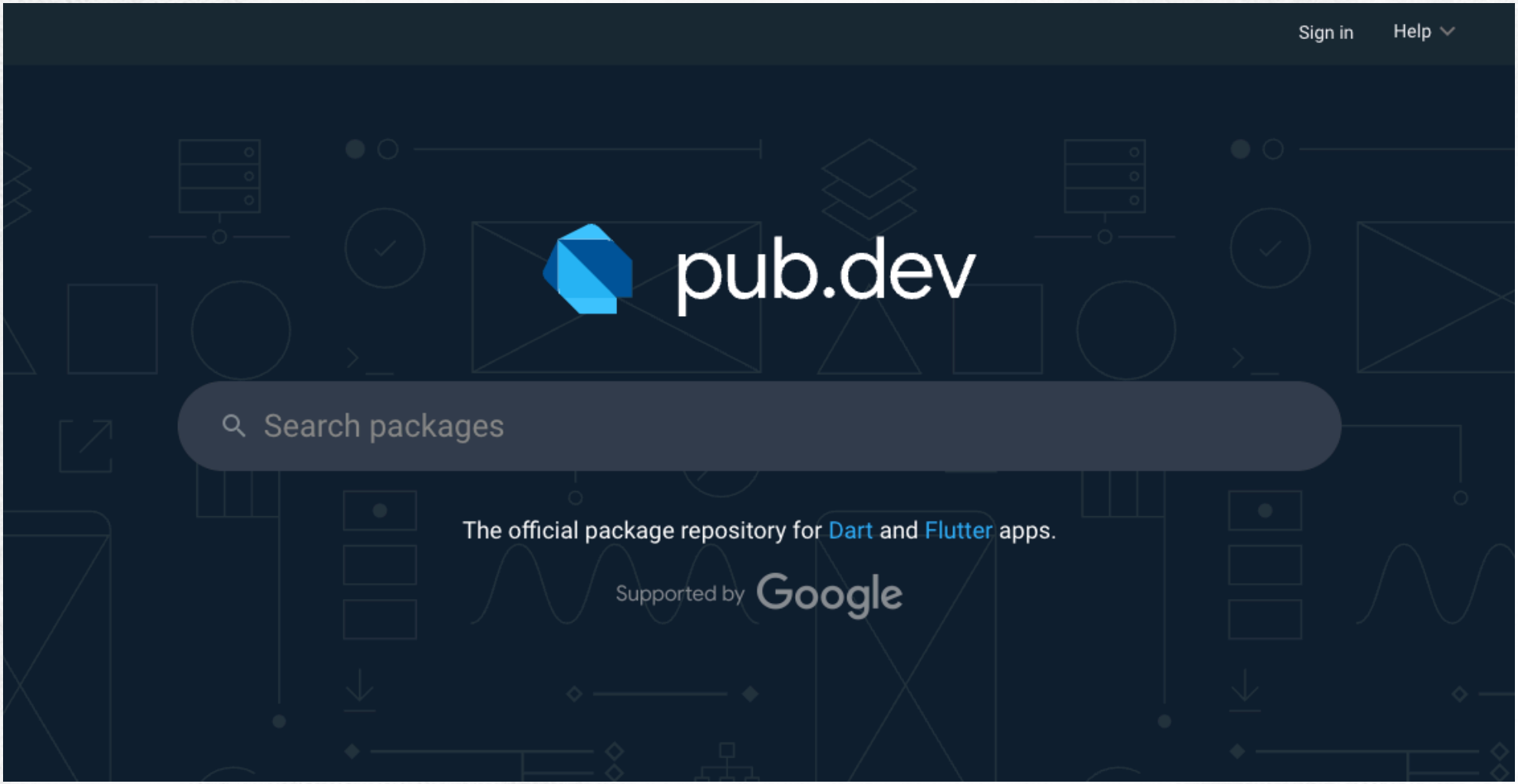
Sungwon Lee

Department of Software Convergence

경희대학교
KYUNG HEE UNIVERSITY

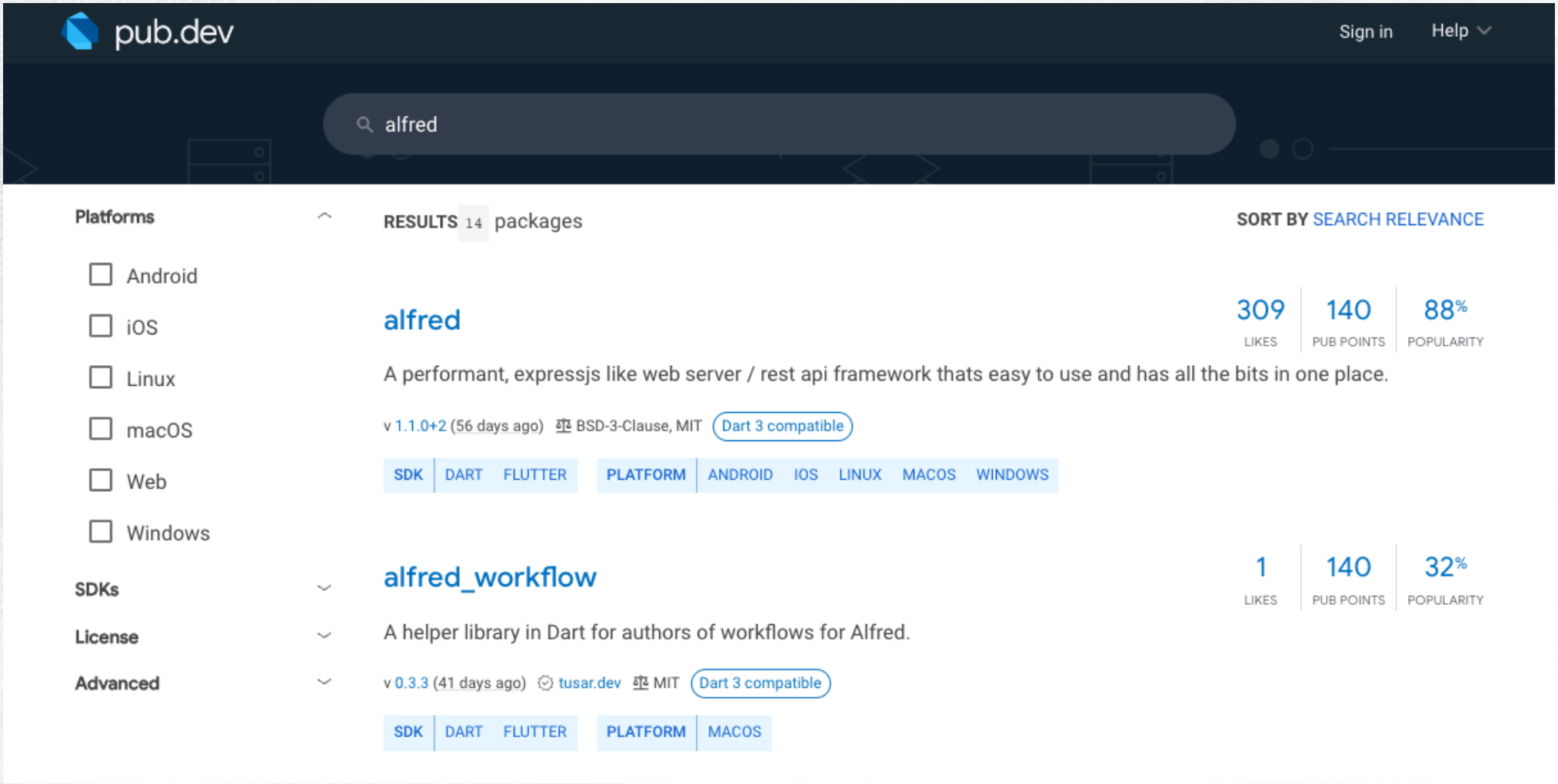
추가 패키지 활용하기

- Official package repository for Dart and Flutter apps
  https://pub.dev/



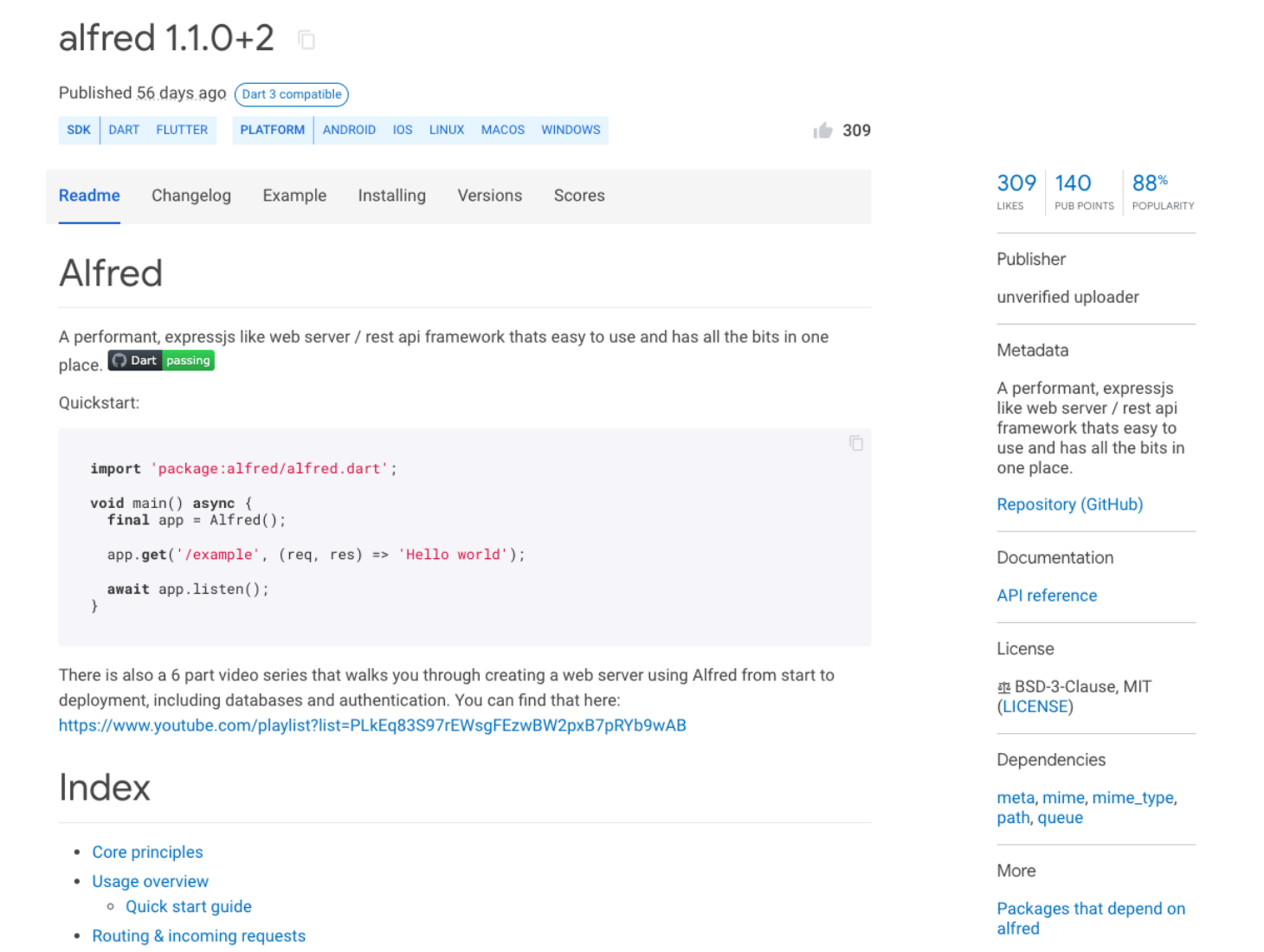
추가 패키지 활용하기

- 예시: Alfred (웹 서버 프레임워크 : 추후 수업 내용)



추가 패키지 활용하기

- 예시: Alfred (웹 서버 프레임워크 : 추후 수업 내용)



alfred 1.1.0+2

Published 56 days ago Dart 3 compatible

SDK DART FLUTTER PLATFORM ANDROID IOS LINUX MACOS WINDOWS 309

Readme Changelog Example Installing Versions Scores

Alfred

A performant, expressjs like web server / rest api framework thats easy to use and has all the bits in one place. Dart passing

Quickstart:

```
import 'package:alfred/alfred.dart';

void main() async {
  final app = Alfred();

  app.get('/example', (req, res) => 'Hello world');

  await app.listen();
}
```

There is also a 6 part video series that walks you through creating a web server using Alfred from start to deployment, including databases and authentication. You can find that here: https://www.youtube.com/playlist?list=PLkEq83S97rEWsgFEzwBW2pxB7pRYb9wAB

Index

- Core principles
- Usage overview
  - Quick start guide
- Routing & incoming requests

309 LIKES 140 PUB POINTS 88% POPULARITY

Publisher

unverified uploader

Metadata

A performant, expressjs like web server / rest api framework thats easy to use and has all the bits in one place.

Repository (GitHub)

Documentation

API reference

License

BSD-3-Clause, MIT (LICENSE)

Dependencies

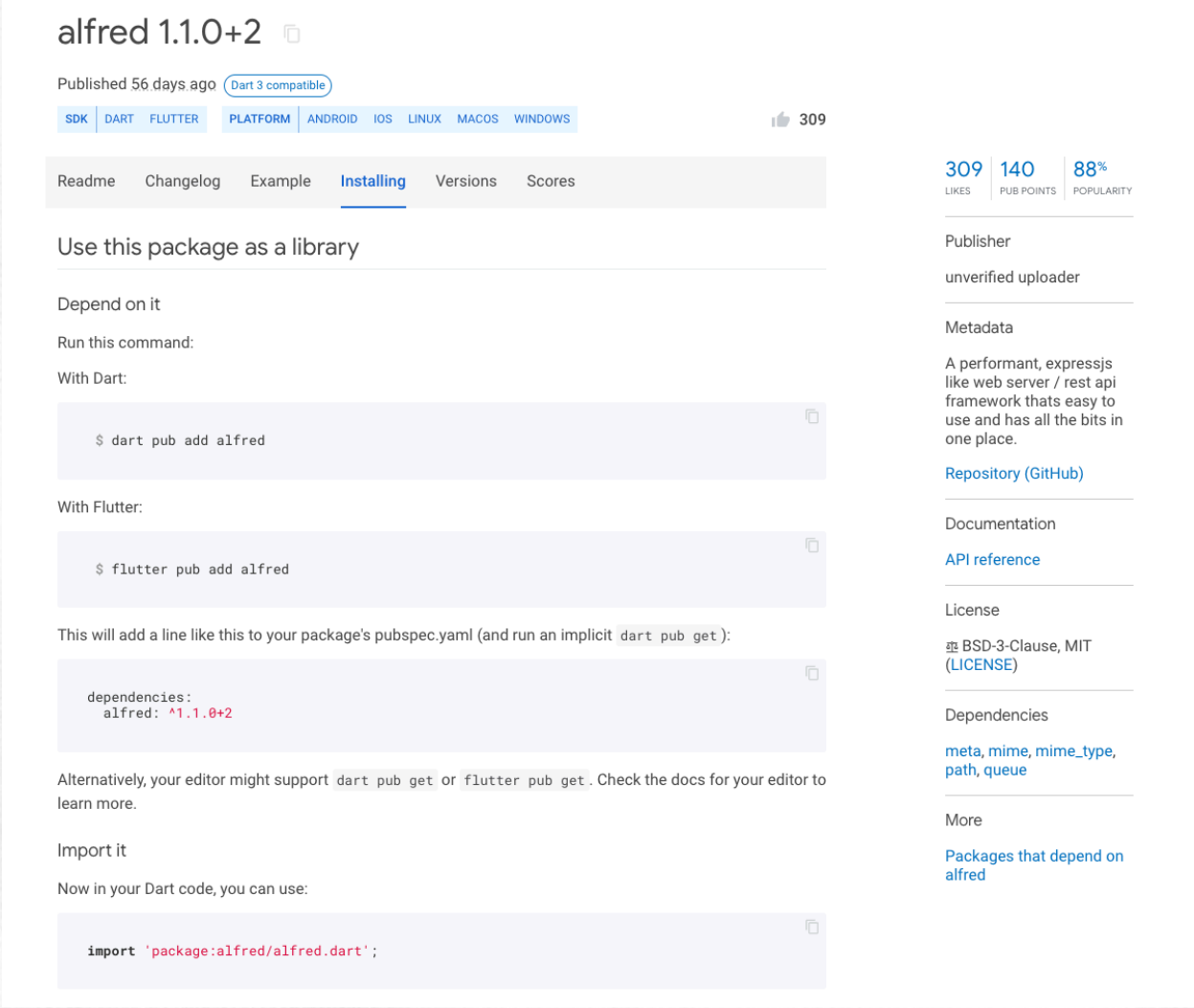
meta, mime, mime_type, path, queue

More

Packages that depend on alfred

추가 패키지 활용하기

- 예시: Alfred (웹 서버 프레임워크 : 추후 수업 내용)



alfred 1.1.0+2

Published 56 days ago Dart 3 compatible

SDK DART FLUTTER | PLATFORM ANDROID IOS LINUX MACOS WINDOWS | 309

Readme Changelog Example Installing Versions Scores

309 LIKES | 140 PUB POINTS | 88% POPULARITY

Use this package as a library

Depend on it

Run this command:

With Dart:

```
$ dart pub add alfred
```

With Flutter:

```
$ flutter pub add alfred
```

This will add a line like this to your package's pubspec.yaml (and run an implicit `dart pub get`):

```
dependencies:
  alfred: ^1.1.0+2
```

Alternatively, your editor might support `dart pub get` or `flutter pub get`. Check the docs for your editor to learn more.

Import it

Now in your Dart code, you can use:

```
import 'package:alfred/alfred.dart';
```

Publisher

unverified uploader

Metadata

A performant, expressjs like web server / rest api framework thats easy to use and has all the bits in one place.

Repository (GitHub)

Documentation

API reference

License

BSD-3-Clause, MIT (LICENSE)

Dependencies

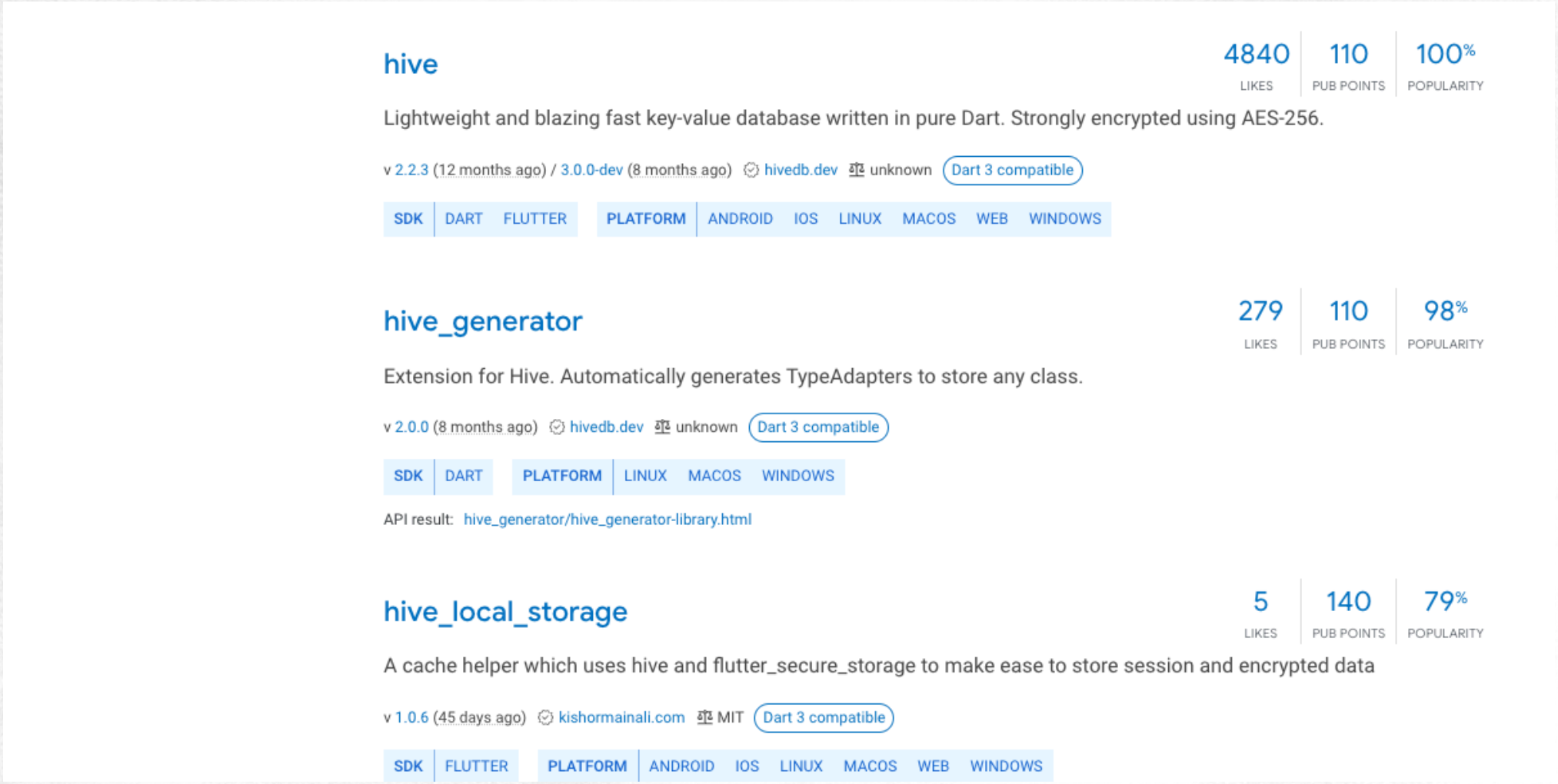
meta, mime, mime_type, path, queue

More

Packages that depend on alfred

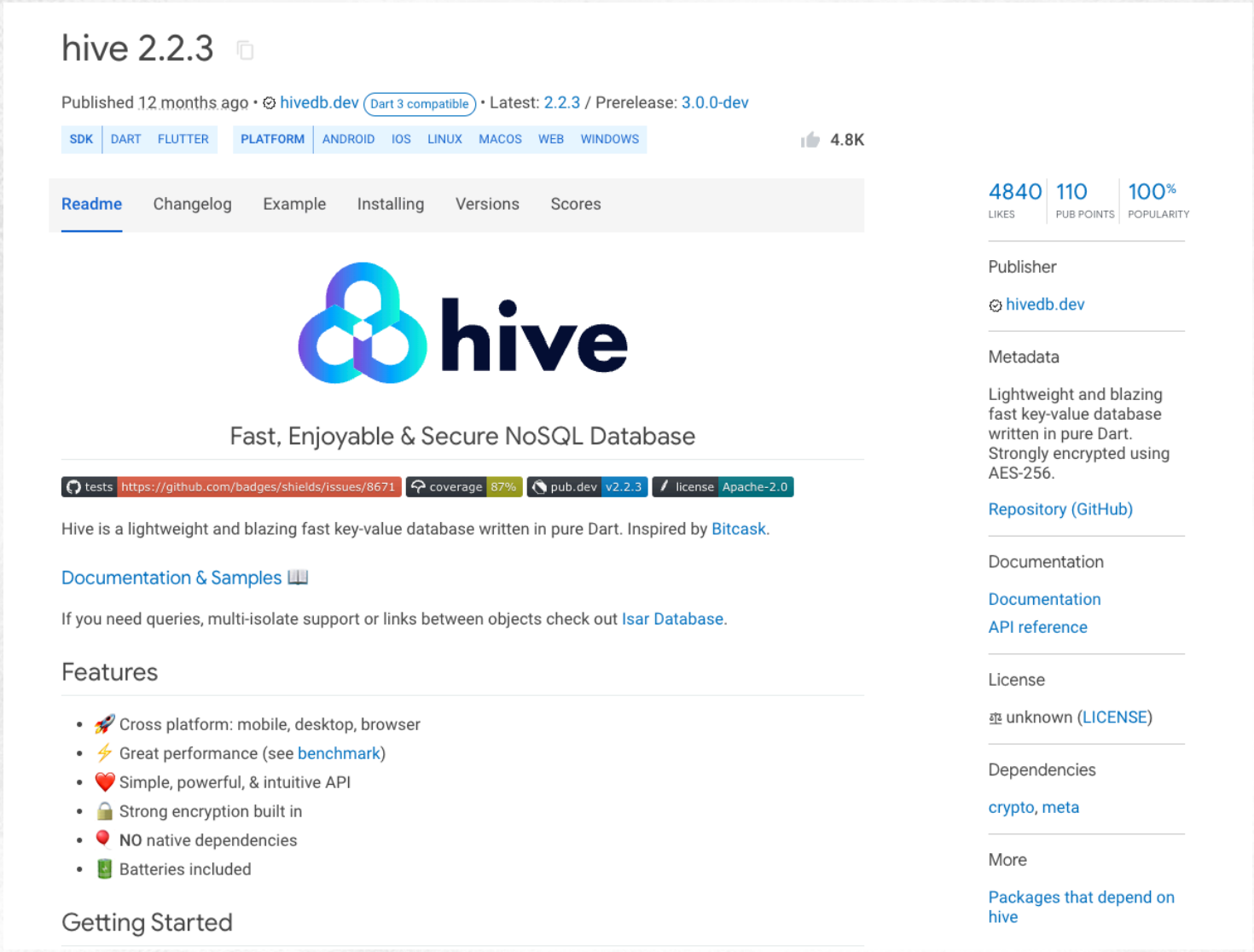
추가 패키지 활용하기

- 예시: Hive (In-App 데이터베이스 : 추후 수업 내용)



추가 패키지 활용하기

- 예시: Hive (In-App 데이터베이스 : 추후 수업 내용)



hive 2.2.3

Published 12 months ago • hivedb.dev Dart 3 compatible • Latest: 2.2.3 / Prerelease: 3.0.0-dev

SDK DART FLUTTER | PLATFORM ANDROID IOS LINUX MACOS WEB WINDOWS | 4.8K

Readme Changelog Example Installing Versions Scores

hive

Fast, Enjoyable & Secure NoSQL Database

tests https://github.com/badges/shields/issues/8671 coverage 87% pub.dev v2.2.3 license Apache-2.0

Hive is a lightweight and blazing fast key-value database written in pure Dart. Inspired by Bitcask.

Documentation & Samples 📖

If you need queries, multi-isolate support or links between objects check out Isar Database.

Features

- 🚀 Cross platform: mobile, desktop, browser
- ⚡ Great performance (see benchmark)
- ❤️ Simple, powerful, & intuitive API
- 🔒 Strong encryption built in
- 🎈 NO native dependencies
- 🔋 Batteries included

Getting Started

4840 LIKES | 110 PUB POINTS | 100% POPULARITY

Publisher

hivedb.dev

Metadata

Lightweight and blazing fast key-value database written in pure Dart. Strongly encrypted using AES-256.

Repository (GitHub)

Documentation

Documentation

API reference

License

unknown (LICENSE)

Dependencies

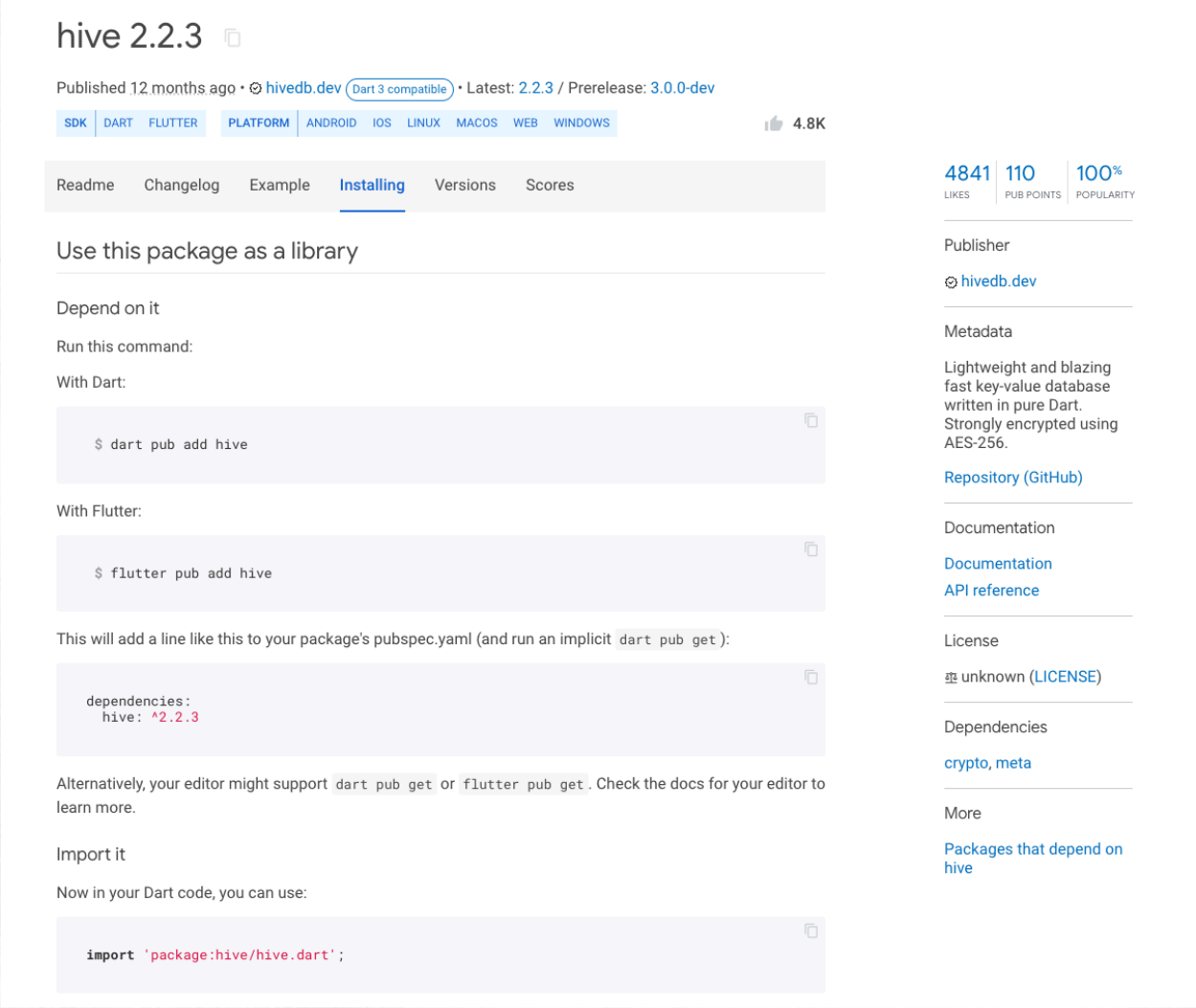
crypto, meta

More

Packages that depend on hive

추가 패키지 활용하기

- 예시: Hive (In-App 데이터베이스 : 추후 수업 내용)



hive 2.2.3

Published 12 months ago • hivedb.dev Dart 3 compatible • Latest: 2.2.3 / Prerelease: 3.0.0-dev

SDK DART FLUTTER | PLATFORM ANDROID IOS LINUX MACOS WEB WINDOWS | 4.8K

Readme Changelog Example Installing Versions Scores

Use this package as a library

Depend on it

Run this command:

With Dart:

```
$ dart pub add hive
```

With Flutter:

```
$ flutter pub add hive
```

This will add a line like this to your package's pubspec.yaml (and run an implicit `dart pub get`):

```
dependencies:
  hive: ^2.2.3
```

Alternatively, your editor might support `dart pub get` or `flutter pub get`. Check the docs for your editor to learn more.

Import it

Now in your Dart code, you can use:

```
import 'package:hive/hive.dart';
```

4841 LIKES | 110 PUB POINTS | 100% POPULARITY

Publisher

hivedb.dev

Metadata

Lightweight and blazing fast key-value database written in pure Dart. Strongly encrypted using AES-256.

Repository (GitHub)

Documentation

Documentation

API reference

License

unknown (LICENSE)

Dependencies

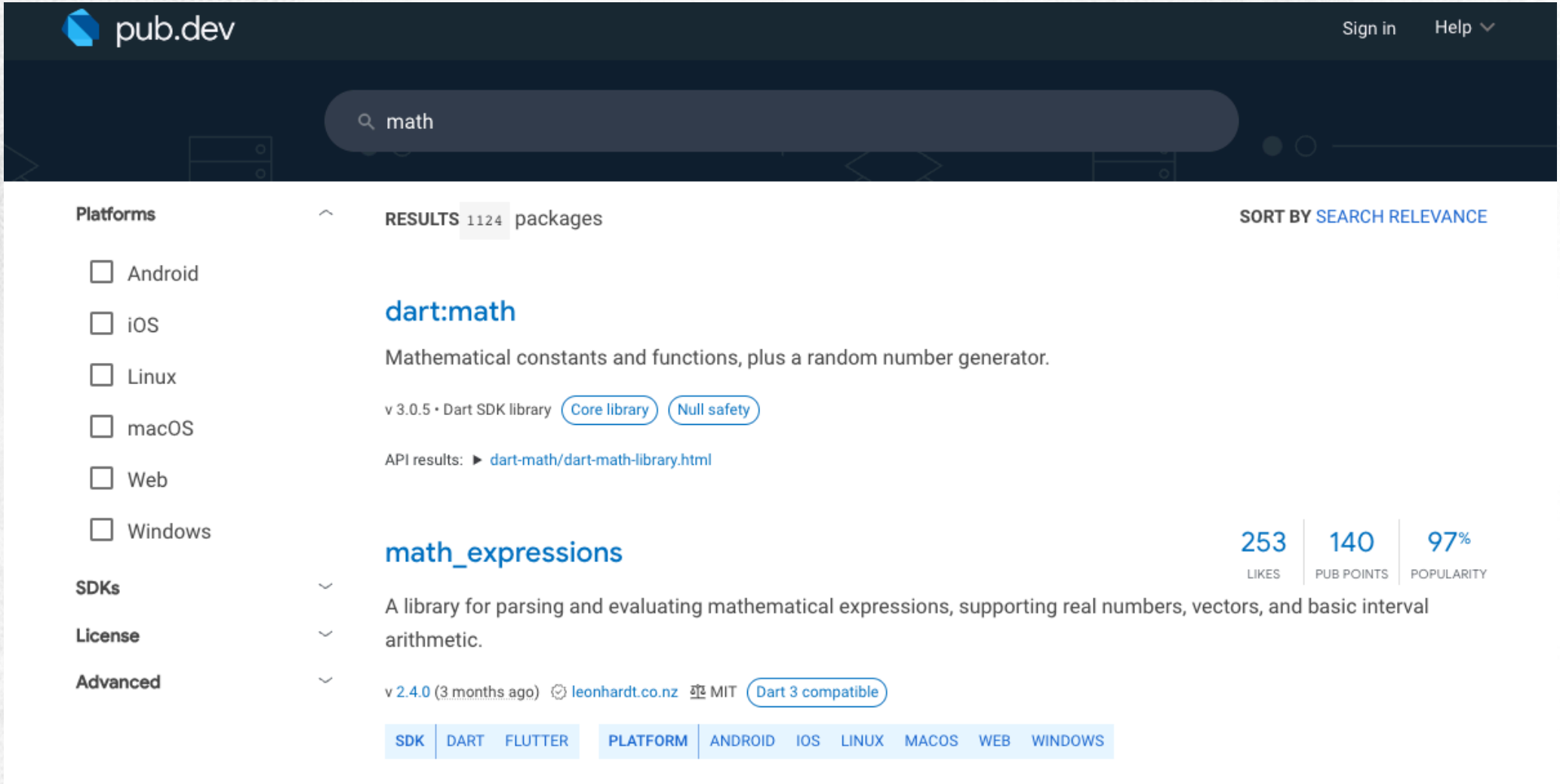
crypto, meta

More

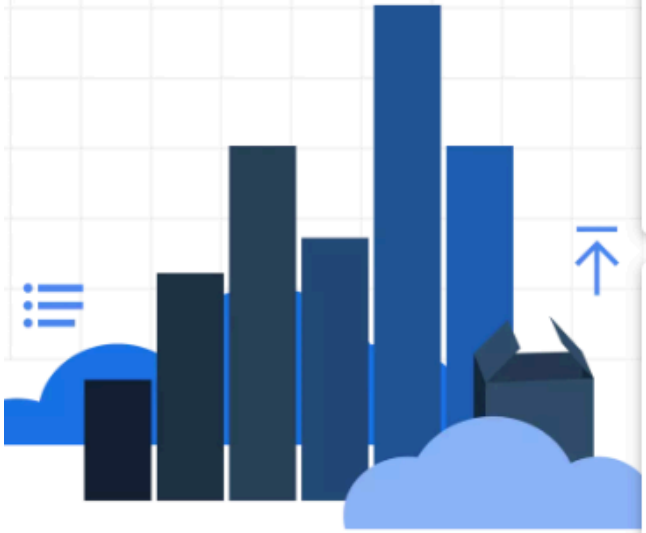
Packages that depend on hive

표준 패키지 확인하기

- 예시: Math (dart:math)



추가 패키지 활용하기



Most popular packages

Some of the most downloaded packages over the past 60 days

cupertino_icons
Default icons asset for Cupertino widgets based on Apple styled icons
flutter.dev

get_it
Simple direct Service Locator that allows to decouple the interface from a concrete
fluttercommunity.dev

badges
A package for creating badges. Badges can be used for an additional marker for any
yako.io

go_router
A declarative router for Flutter based on Navigation 2 supporting deep linking, data-
flutter.dev

font_awesome_flutter
The Font Awesome Icon pack available as Flutter Icons. Provides 1600 additional icons
fluttercommunity.dev

sse
Provides client and server functionality for setting up bi-directional communication
tools.dart.dev

VIEW ALL

추가 패키지 활용하기

Top Flutter packages

Some of the top packages that extend Flutter with new features

font_awesome_flutter

The Font Awesome Icon pack available as Flutter Icons. Provides 1600 additional icons

fluttercommunity.dev

flutter_image_comp...

Compress Pictures. Can effectively reduce the size of the transmission.

image_picker

Flutter plugin for selecting images from the Android and iOS image library, and taking

flutter.dev

mask_text_input_for...

The package provides TextInputFormatter for TextField and TextFormField which format the input by a given mask

provider

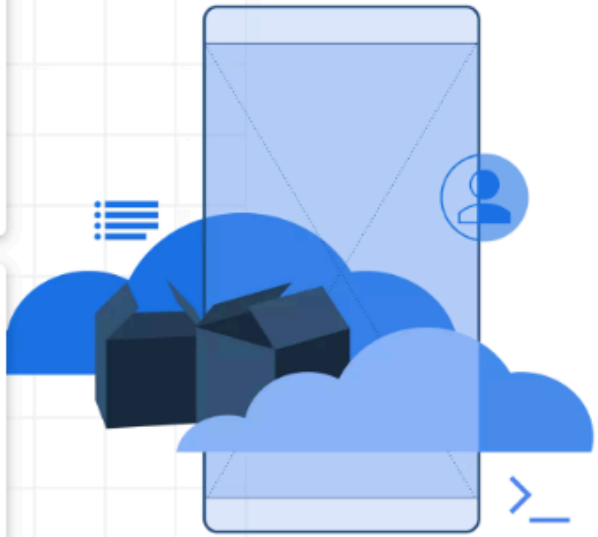
A wrapper around InheritedWidget to make them easier to use and more

dash-overflow.net

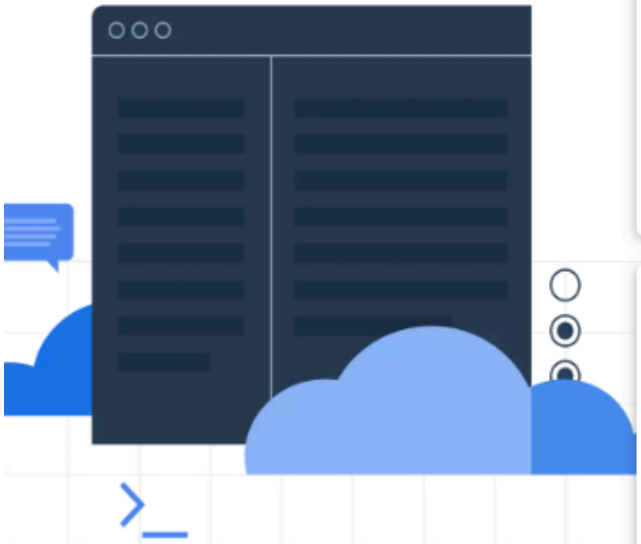
sizer

A flutter plugin for Easily make Flutter apps responsive. Automatically adapt UI to different screen sizes. Responsiveness made

VIEW ALL



추가 패키지 활용하기



Top Dart packages

Some of the top packages for any Dart-based app or program

uuid
RFC4122 (v1, v4, v5) UUID Generator and Parser for all Dart platforms (Web, VM,
yuli.dev

built_collection
Immutable collections based on the SDK collections. Each SDK collection class is split into a
google.dev

email_validator
A simple (but correct) dart class for validating email addresses

shelf_router
A convenient request router for the shelf web-framework, with support for URL-parameters,
tools.dart.dev

injectable
Injectable is a convenient code generator for get_it. Inspired by Angular DI, Guice DI and
codeness.ly

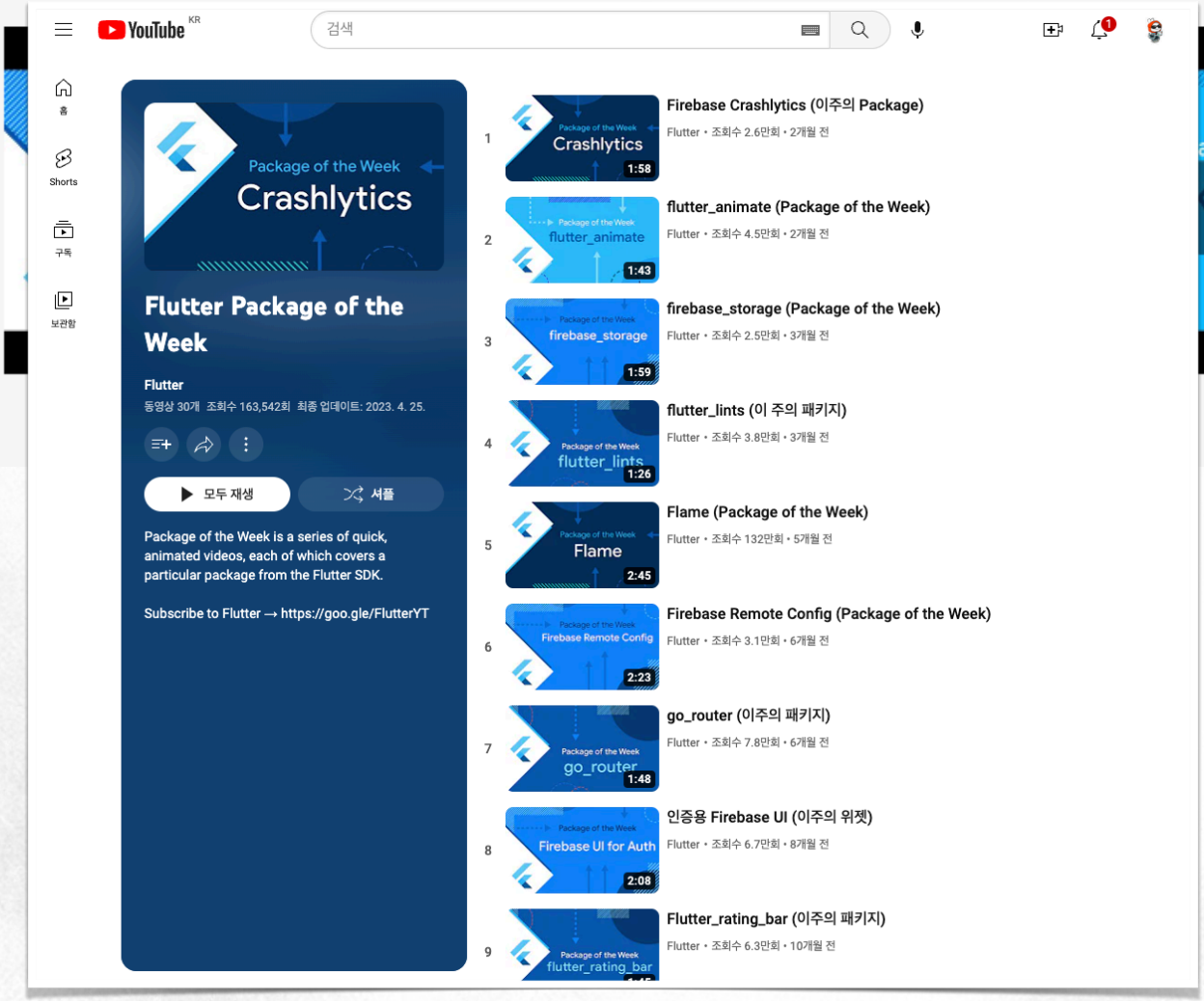
english_words
Utilities for working with English words. Counts syllables, generates well-sounding word
filiph.net

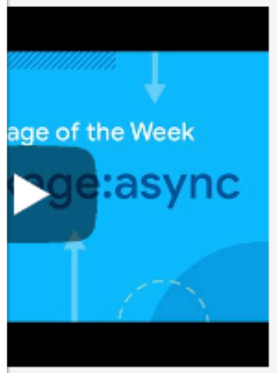
VIEW ALL

추가 패키지 활용하기

Package of the Week

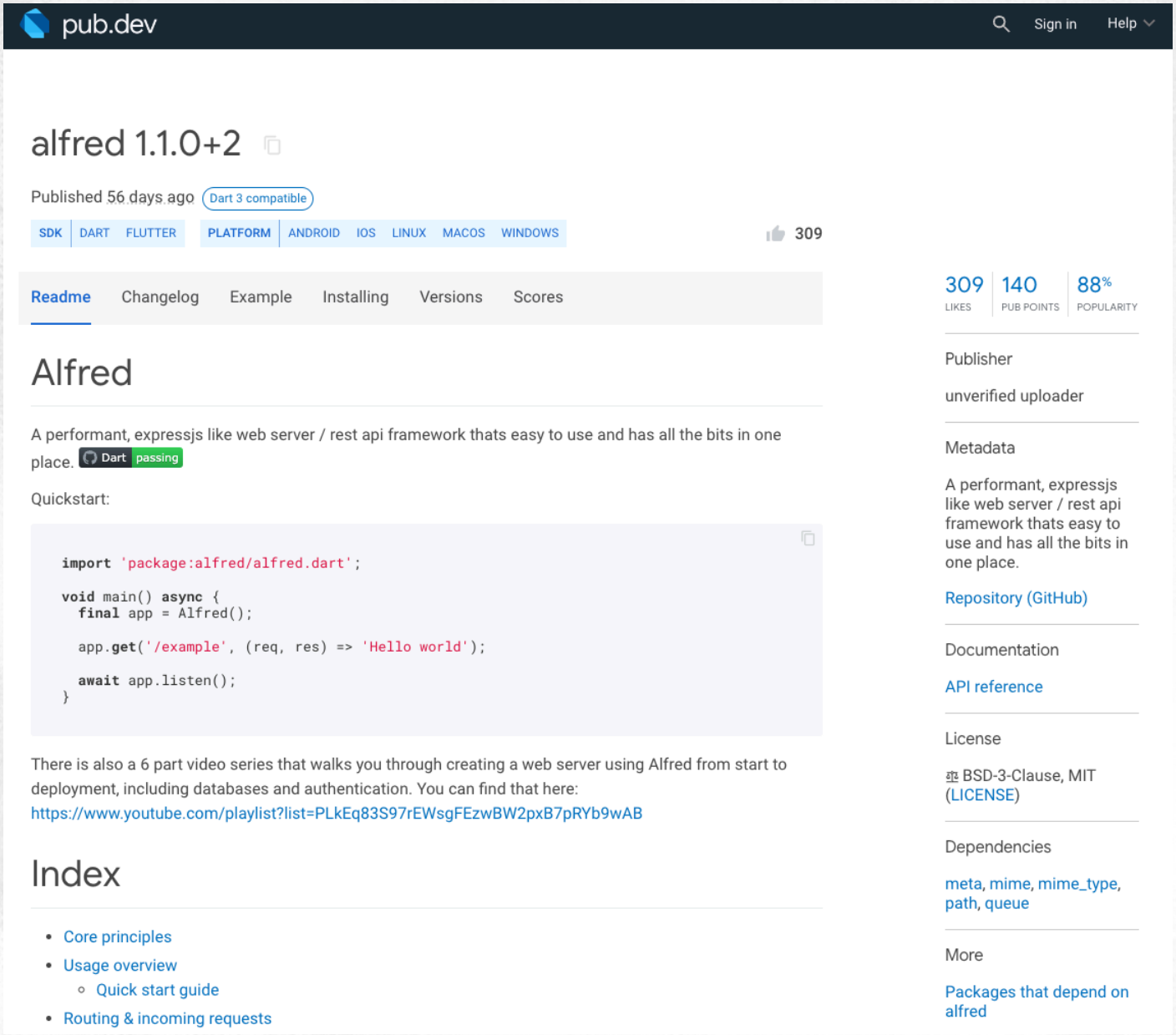
Package of the Week is a series of quick, animated videos, each of which covers a particular package





VIEW PLAYLIST

Alfred 웹 서버 프레임워크 사용하기



pub.dev

Sign in Help

alfred 1.1.0+2

Published 56 days ago Dart 3 compatible

SDK DART FLUTTER PLATFORM ANDROID IOS LINUX MACOS WINDOWS 309

Readme Changelog Example Installing Versions Scores

Alfred

A performant, expressjs like web server / rest api framework thats easy to use and has all the bits in one place. Dart passing

Quickstart:

```
import 'package:alfred/alfred.dart';

void main() async {
  final app = Alfred();

  app.get('/example', (req, res) => 'Hello world');

  await app.listen();
}
```

There is also a 6 part video series that walks you through creating a web server using Alfred from start to deployment, including databases and authentication. You can find that here: https://www.youtube.com/playlist?list=PLkEq83S97rEWsgFEzwBW2pxB7pRYb9wAB

Index

- Core principles
- Usage overview
  - Quick start guide
- Routing & incoming requests

309 LIKES | 140 PUB POINTS | 88% POPULARITY

Publisher

unverified uploader

Metadata

A performant, expressjs like web server / rest api framework thats easy to use and has all the bits in one place.

Repository (GitHub)

Documentation

API reference

License

BSD-3-Clause, MIT (LICENSE)

Dependencies

meta, mime, mime_type, path, queue

More

Packages that depend on alfred

Alfred 웹 서버 프레임워크 사용하기

직접 개발
volume-D-chapter-05-server.dart

```dart
import 'dart:io';
import 'dart:convert';

Run | Debug
Future main() async {
  var db = <dynamic, dynamic>{};

  var server = await HttpServer.bind(
    InternetAddress.loopbackIPv4, // ip address
    4040, // port number
  );

  printHttpServerActivated(server);

  await for (HttpRequest request in server) {
    if (request.uri.path.contains('/api/') == true) {
      printHttpRequestInfo(request);
      try {
        switch (request.method) {
          case 'POST': // Create
            createDB(db, request);
            break;
          case 'GET': // Read
            readDB(db, request);
            break;
          case 'PUT': // Update
            upateDB(db, request);
            break;
          case 'DELETE': // Delete
            deleteDB(db, request);
            break;
          default:
            print("\$ Unsupported http method");
        }
      } catch (err) {
        print("\$ Exception in http request processing");
      }
    } else {
      printAndSendHttpResponse(
          db, request, "${request.method} {ERROR: Unsupported API}");
    }
  }
}
```

Alfred 사용
main.dart

```dart
import 'dart:io';
import 'dart:convert';
import 'package:alfred/alfred.dart';

Run | Debug
Future main() async {
  var db = <dynamic, dynamic>{};

  final app = Alfred();

  app.post('/api/*', (req, res) => createDB(db, req));
  app.get('/api/*', (req, res) => readDB(db, req));
  app.put('/api/*', (req, res) => upateDB(db, req));
  app.delete('/api/*', (req, res) => deleteDB(db, req));

  await app.listen(4040);
}
```

CRUD 서버 개발 예제

Alfred 웹 서버 프레임워크 사용하기

직접 개발
volume-G-chapter-04
simpleHttpFileServer.dart

```dart
import 'dart:io';

Run | Debug
Future main() async {
  var server = await HttpServer.bind(
    InternetAddress.loopbackIPv4, // ip address
    4040, // port number
  );
  printHttpServerActivated(server);

  await for (HttpRequest request in server) {
    printHttpRequestInfo(request);
    try {
      switch (request.method) {
        case 'GET':
          httpGetHandler(request);
          break;
        default:
          print("\$ Unsupported http method");
      }
    } catch (err) {
      print("\$ Exception in http request processing");
    }
  }
}

void printHttpServerActivated(HttpServer server) {
  var ip = server.address.address;
  var port = server.port;
  print('\$ Server activated in ${ip}:${port}');
}

void printHttpRequestInfo(HttpRequest request) async {
  var ip = request.connectionInfo!.remoteAddress.address;
  var port = request.connectionInfo!.remotePort;
  var method = request.method;
  var path = request.uri.path;
  print("\$ $method $path from $ip:$port");

  if (request.headers.contentLength != -1) {
    print("\> content-type   : ${request.headers.contentType}");
    print("\> content-length : ${request.headers.contentLength}");
  }
}

void httpGetHandler(HttpRequest request) async {
  var fileName = request.uri.path.substring(1);
  var fileType = request.uri.path.split('.').last;

  var contentType = {
    'html': 'text/html',
    'js': 'application/javascript',
    'json': 'application/json',
    'wasm': 'application/wasm',
    'otf': 'application/x-font-opentype',
    'ttf': 'application/x-font-ttf',
    'png': 'image/png',
  };

  try {
    if (await File(fileName).exists() == true) {
      request.response.statusCode = HttpStatus.ok;
      request.response.headers.contentType =
          ContentType.parse(contentType[fileType]!);
      var file = new File(fileName);
      file.readAsBytes().then((List<int> bytes) {
        bytes.forEach((int b) => request.response.writeCharCode(b));
        request.response.close();
      });
    } else {
      var content = "File Not found";
      request.response
        ..headers.contentType = ContentType('text', 'plain', charset: "utf-8")
        ..headers.contentLength = content.length
        ..statusCode = HttpStatus.notFound
        ..write(content);
      await request.response.close();
    }
  } catch (err) {
    print("\$ Exception in httpGetHandler()");
  }
}
```

Alfred 사용

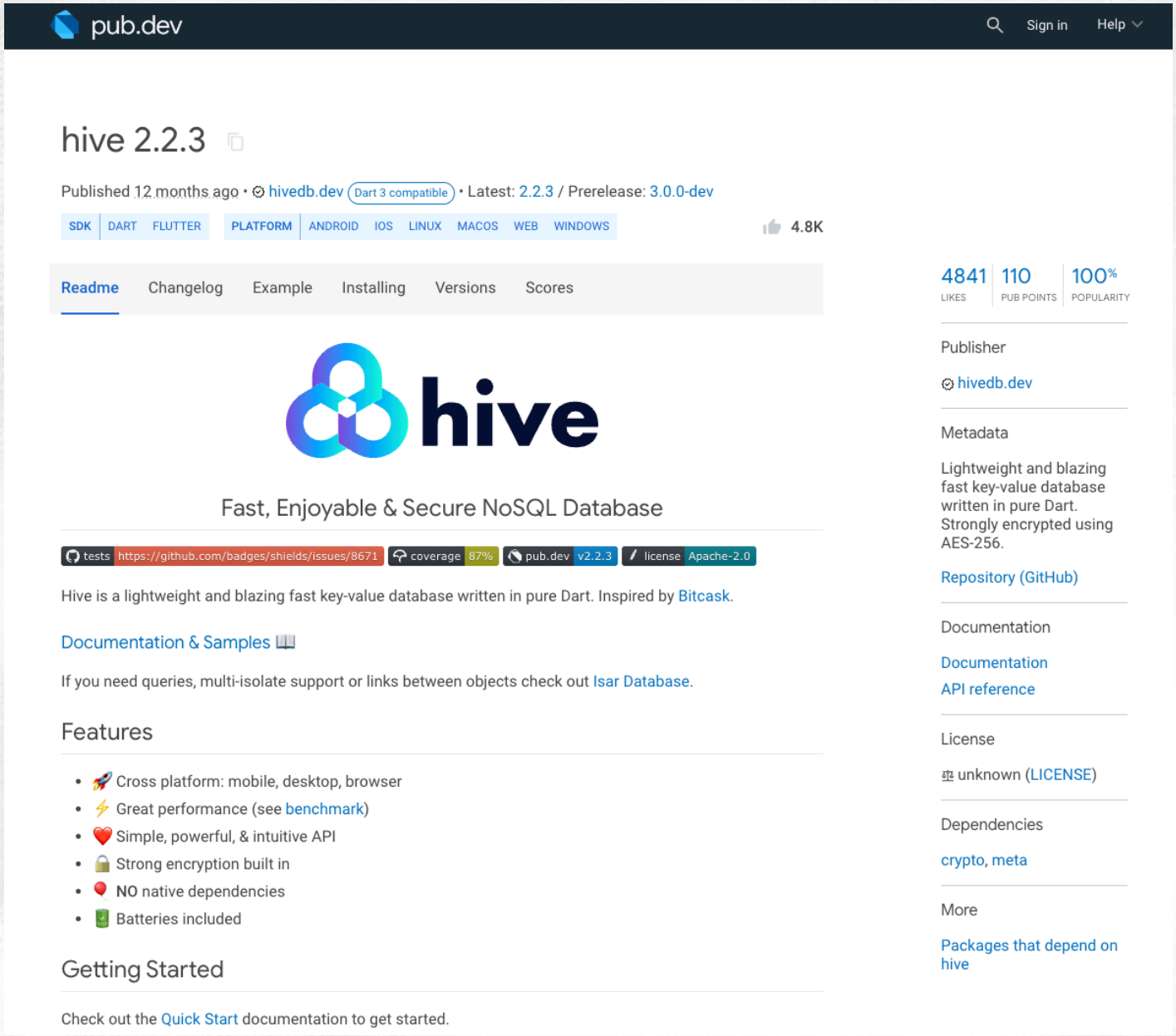
```dart
import 'dart:io';
import 'package:alfred/alfred/dart';
import "package:path/path.dart" show dirname;

void main() async {
  final app = Alfred();
  app.get('/*', (req, res) => Directory('build/web'));
  await app.listen(4040);
}
```

python -m http.server
동일 기능 프로그램 개발

Alfred 웹 서버 프레임워크 사용하기 (리뷰 및 실습)

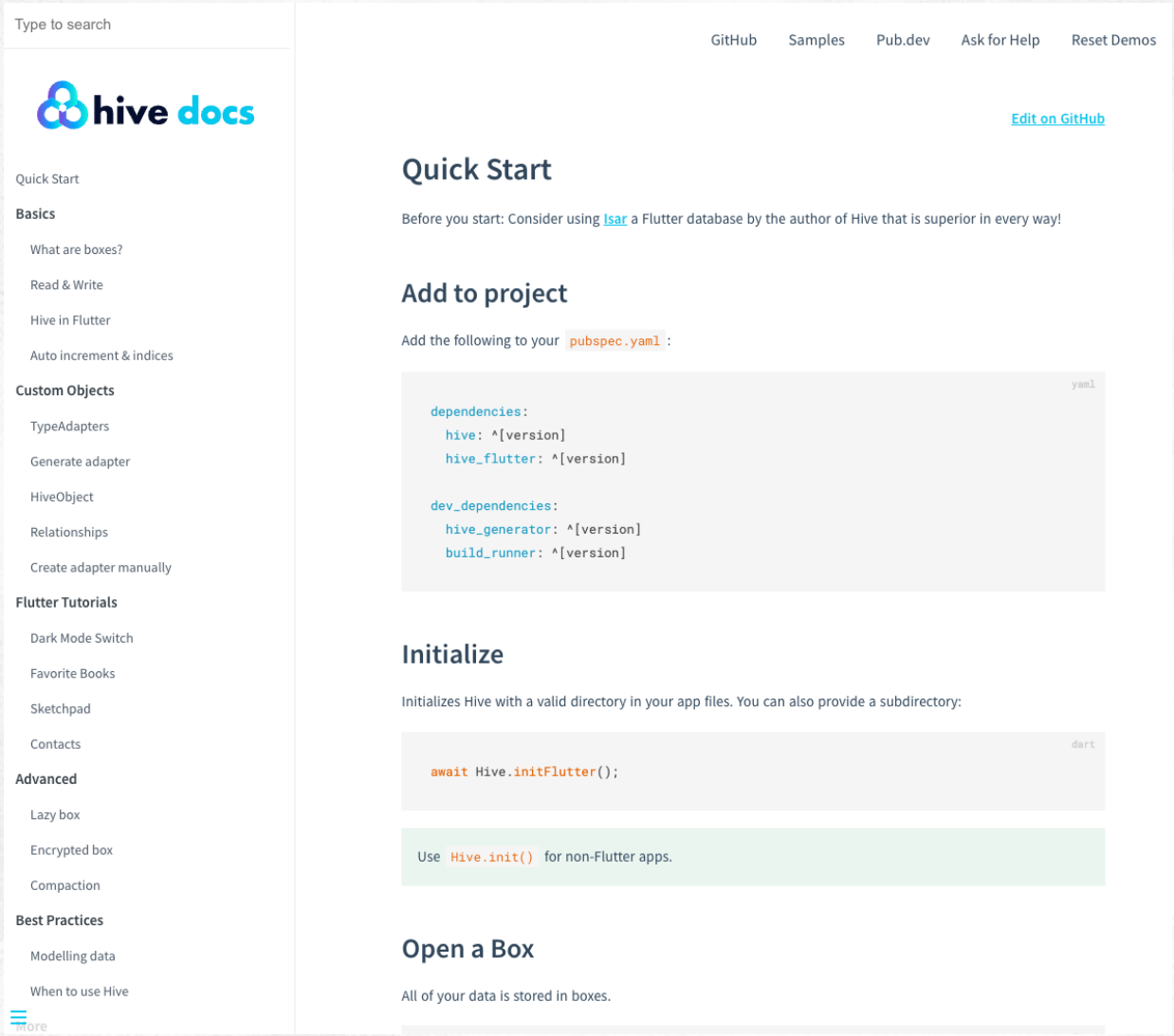
- main.dart
  (volume-D-chapter-05-server.dart의 Alfred 버전)

Hive 데이터베이스 활용하기



Hive 데이터베이스 활용하기

- Quick Start : https://docs.hivedb.dev/#/?id=quick-start

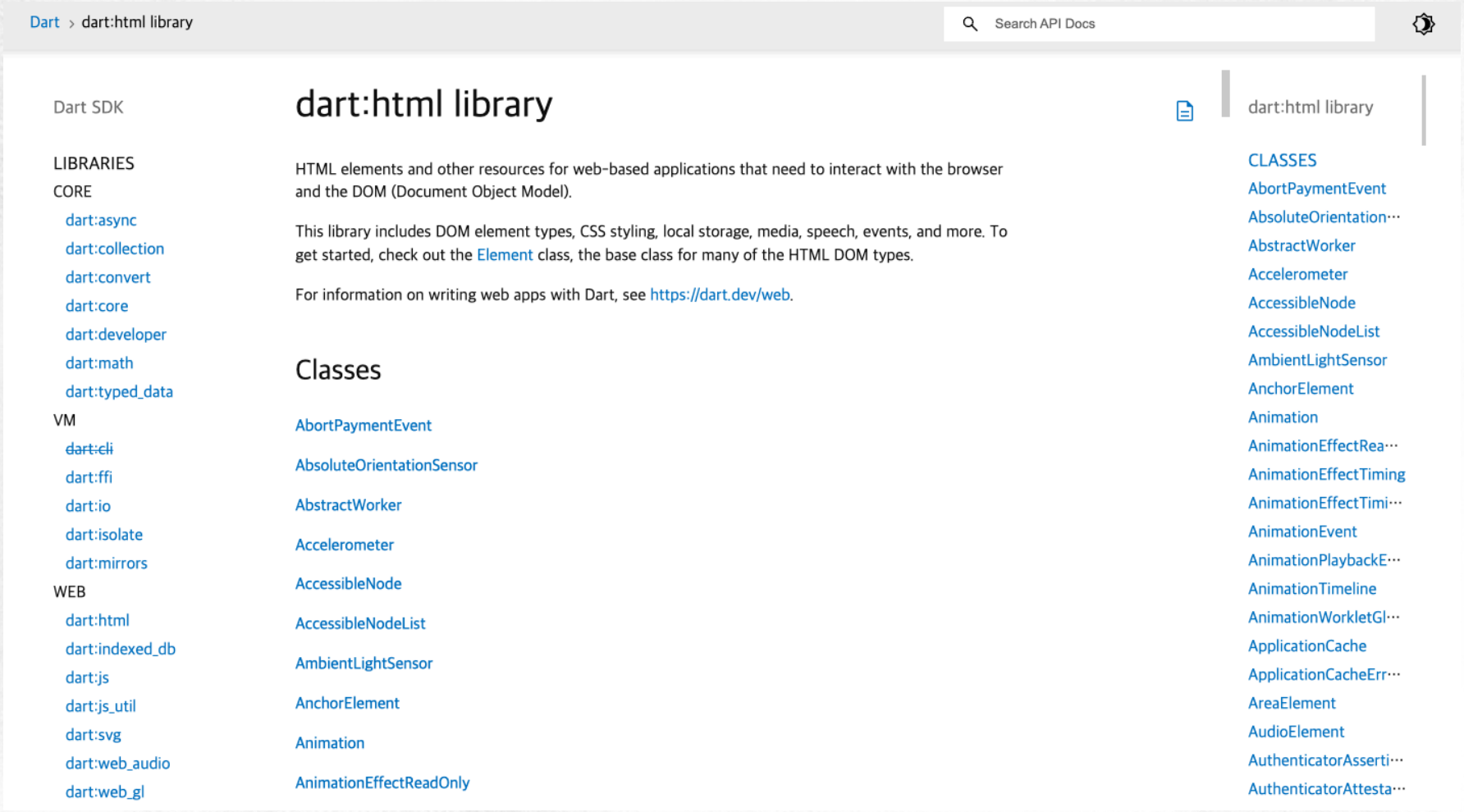


Hive 데이터베이스 활용하기 (리뷰 & 실습)

```
drsungwon~$ dart create hellohive
Creating hellohive using template console...

  .gitignore
  analysis_options.yaml
  CHANGELOG.md
  pubspec.yaml
  README.md
  bin/hellohive.dart
  lib/hellohive.dart
  test/hellohive_test.dart

Running pub get...                          3.6s
  Resolving dependencies...
  Changed 47 dependencies!

Created project hellohive in hellohive! In order to get started, run the following commands:

  cd hellohive
  dart run

drsungwon~$ cd hellohive
drsungwon~$ dart pub add hive
Resolving dependencies...
+ hive 2.2.3
Changed 1 dependency!

===>> volume-I-chapter-02-hellohive_bin.dart 화일을 {프로젝트폴더}/bin/main.dart로 복사
===>> volume-I-chapter-02-hellohive_v2_lib.dart 화일을 {프로젝트폴더}/lib/main.dart로 복사

drsungwon~$ dart run
Building package executable...
Built hellohive:hellohive.
STUD#0001 : STUD#0001: Dart - Seoul
STUD#0002 : STUD#0002: Flutter - Bundang
drsungwon~$
```

Dart for Web 활용하기

- Dart:HTML
  enables HTML/CSS/JavaScript -> HTML/CSS/Dart



Dart for Web 활용하기

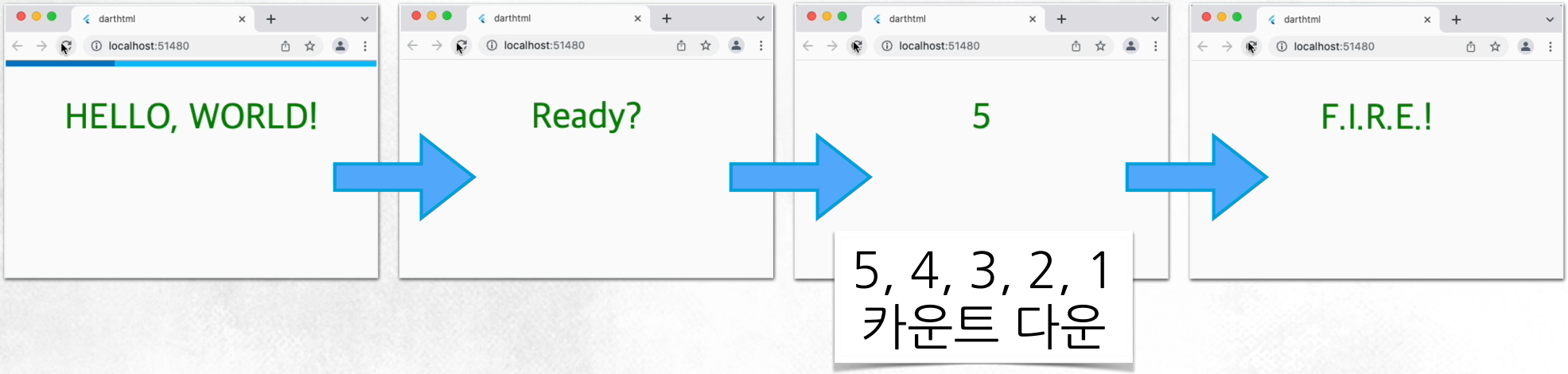
```
</script>
  <center>
    <font size ="20em" color="green">
      <p id="target">
          HELLO, WORLD!
      </p>
    </font>
  </center>
</body>
</html>
```

index.html

```
1  import 'dart:html';
2  import 'dart:core';
3
4  void main() async {
5    final header = querySelector('#target');
6    header?.text = "Ready?";
7    await Future.delayed(Duration(seconds: 2), () => {});
8
9    for (var count = 5; count > 0; count--) {
10     header?.text = "$count";
11     await Future.delayed(Duration(seconds: 1). () => {});
12   }
13
14   header?.text = "F.I.R.E.!";
15 }
```

Dart for Web 활용하기 (리뷰 및 실습)

- flutter create hellodarthtml
- {프로젝트폴더}/lib/main.dart 교체
- {프로젝트폴터}/web/index.html 교체
- flutter run -d chrome



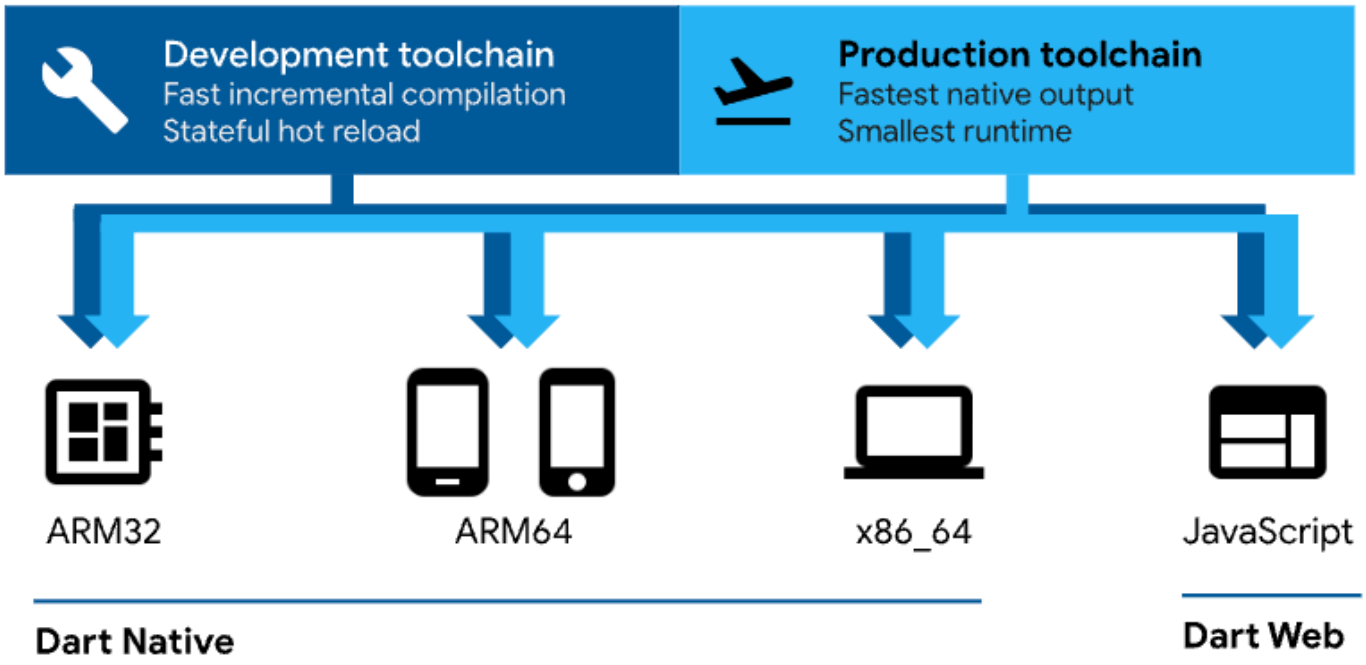
FFI로 C 언어 연결하기

- Dart for Native Platform : https://dart.dev/overview#platform

Dart: The platforms

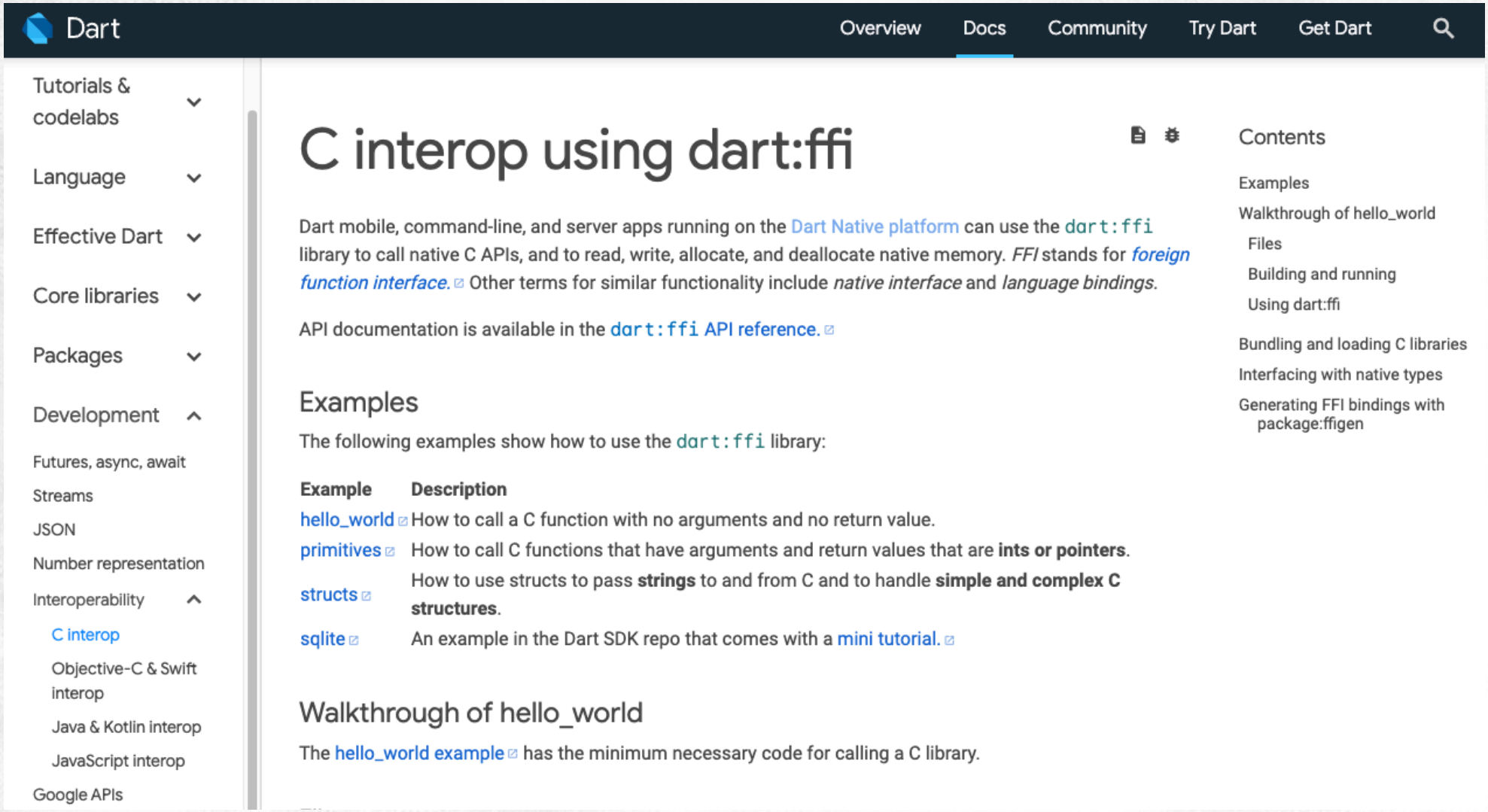
Dart's compiler technology lets you run code in different ways:

- **Native platform**: For apps targeting mobile and desktop devices, Dart includes both a Dart VM with just-in-time (JIT) compilation and an ahead-of-time (AOT) compiler for producing machine code.
- **Web platform**: For apps targeting the web, Dart can compile for development or production purposes. Its web compiler translates Dart into JavaScript.



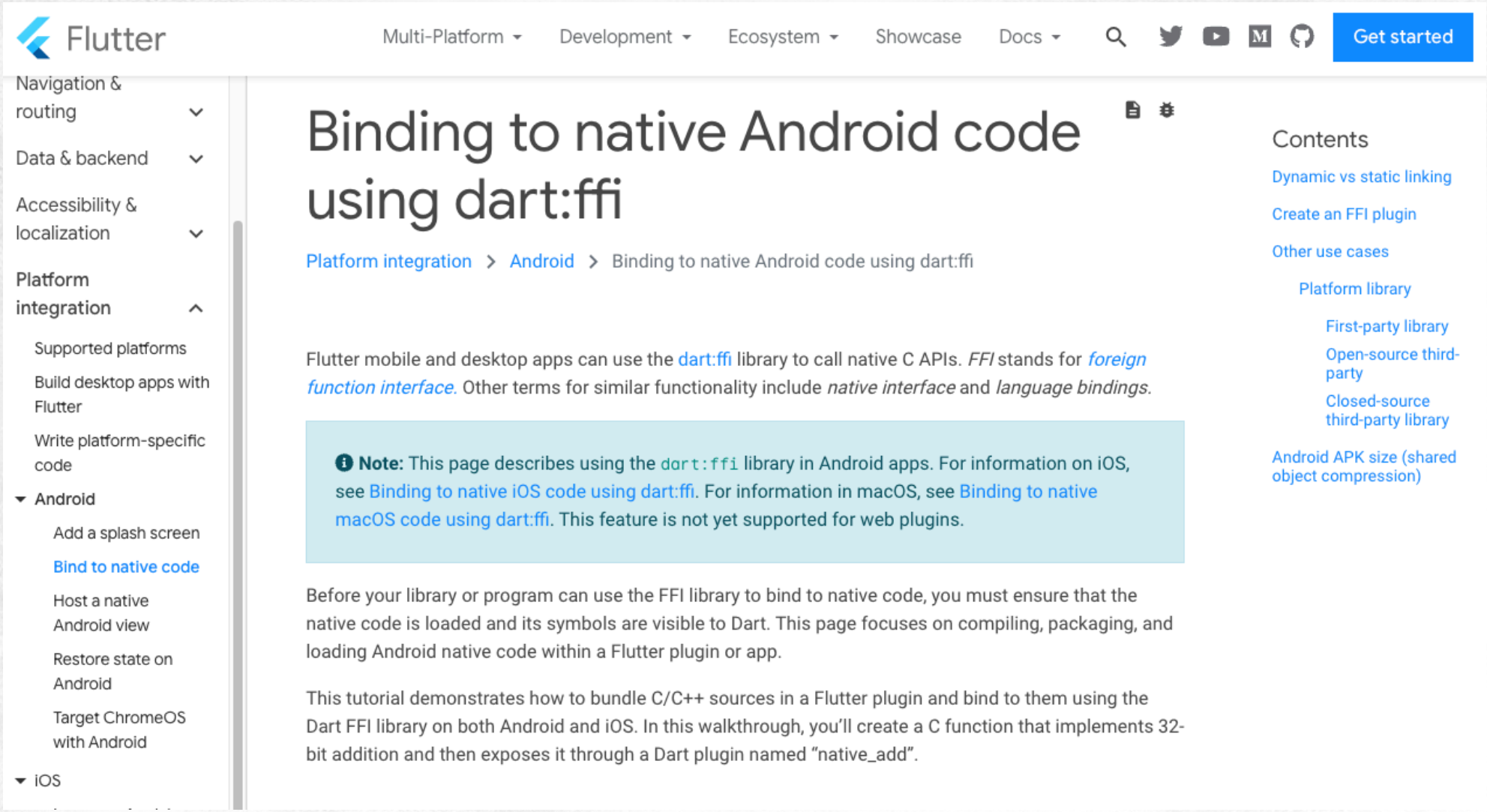
dart:ffi for C Language Interop

- https://dart.dev/guides/libraries/c-interop



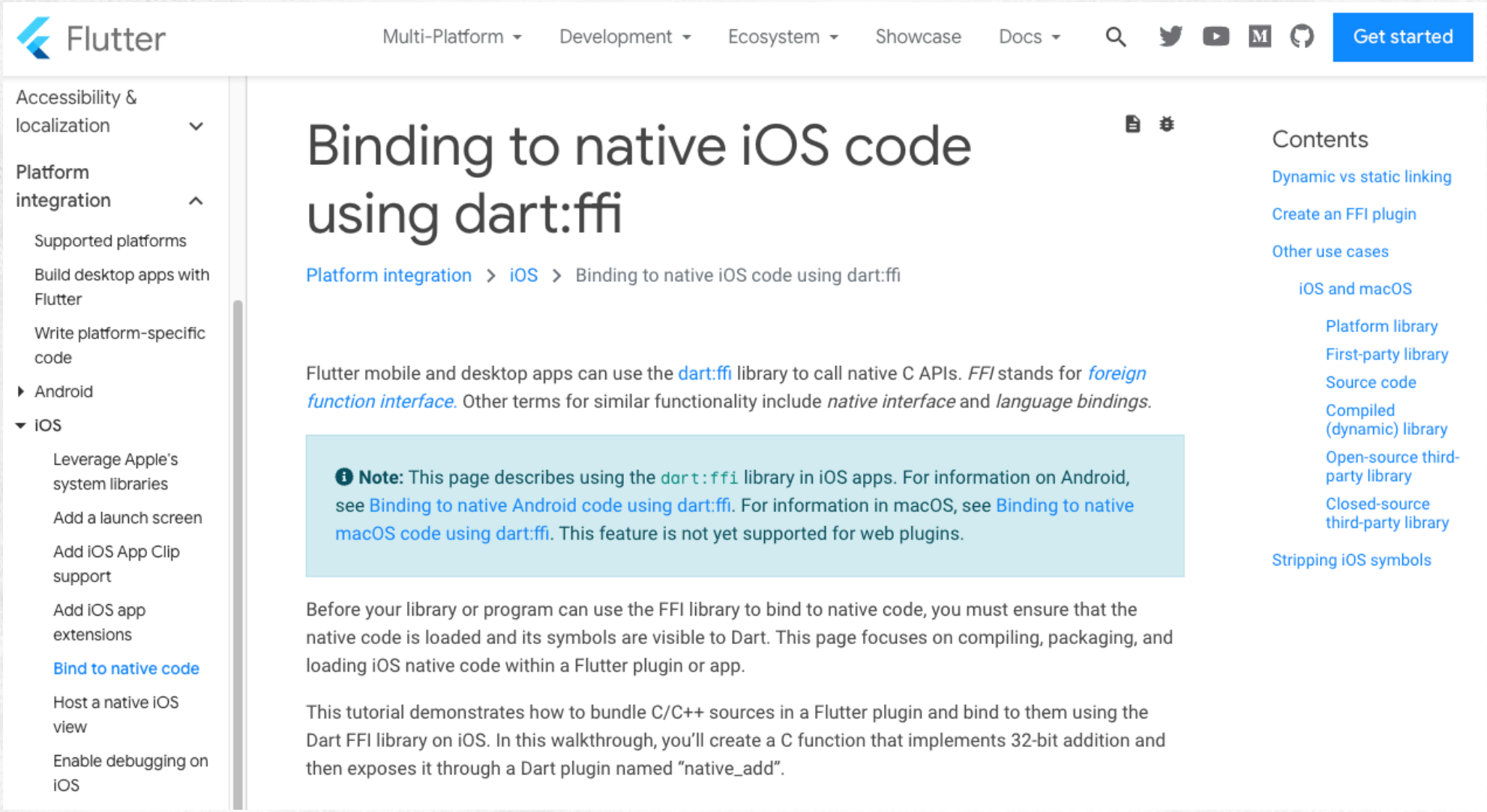
Android Case

- https://docs.flutter.dev/platform-integration/android/c-interop



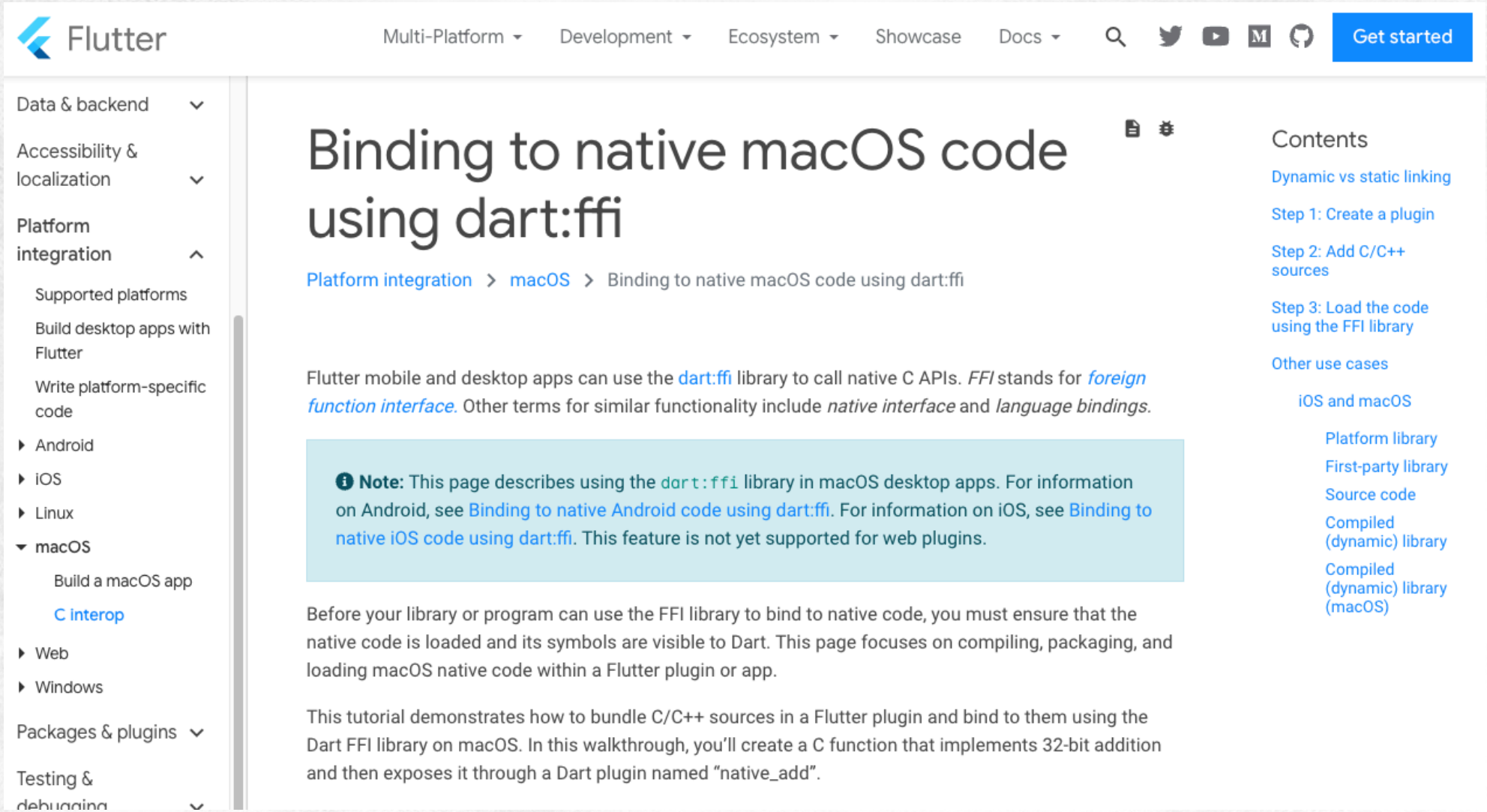
iOS Case

- https://docs.flutter.dev/platform-integration/ios/c-interop



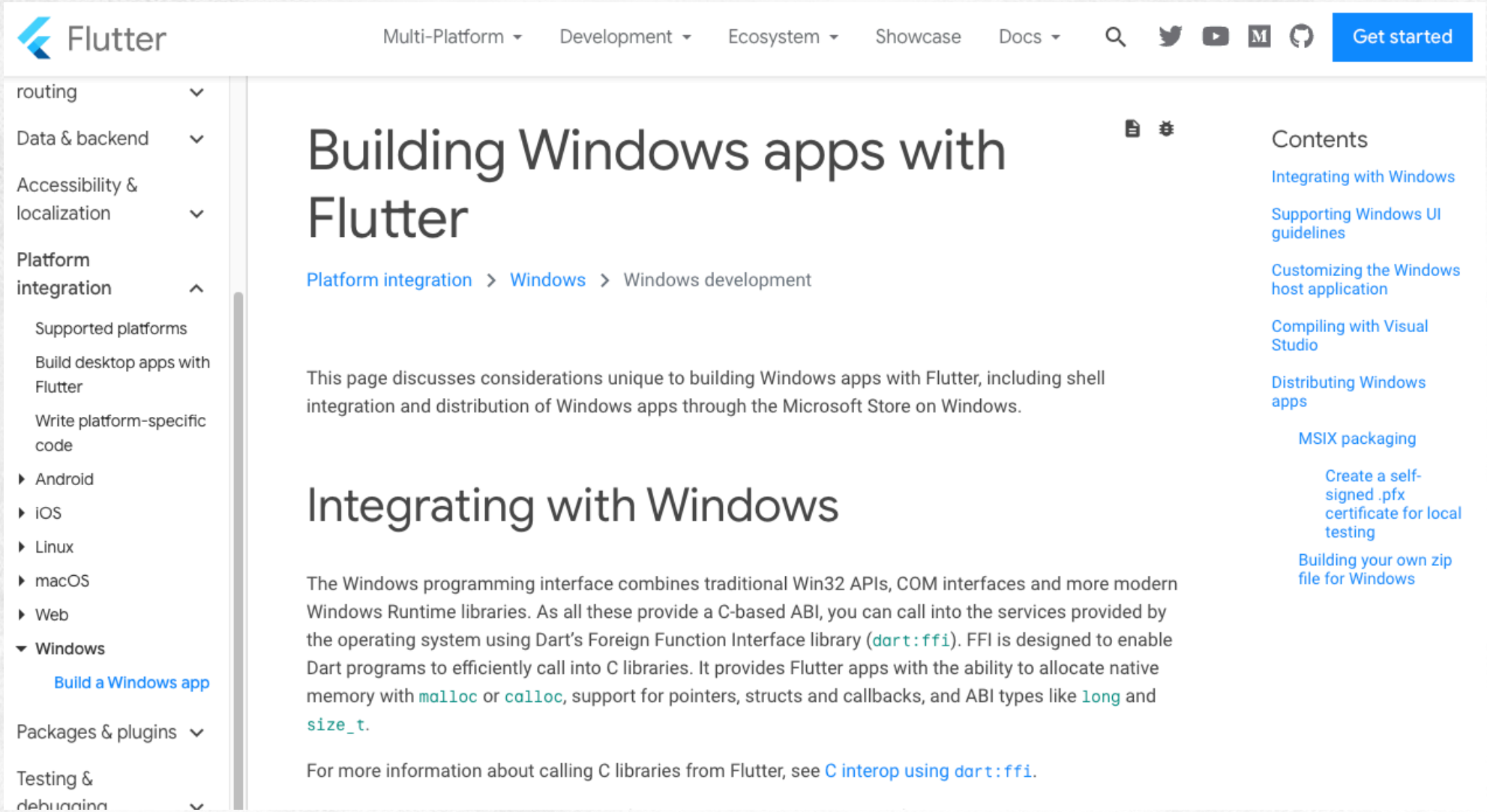
macOS Case

- https://docs.flutter.dev/platform-integration/macos/c-interop



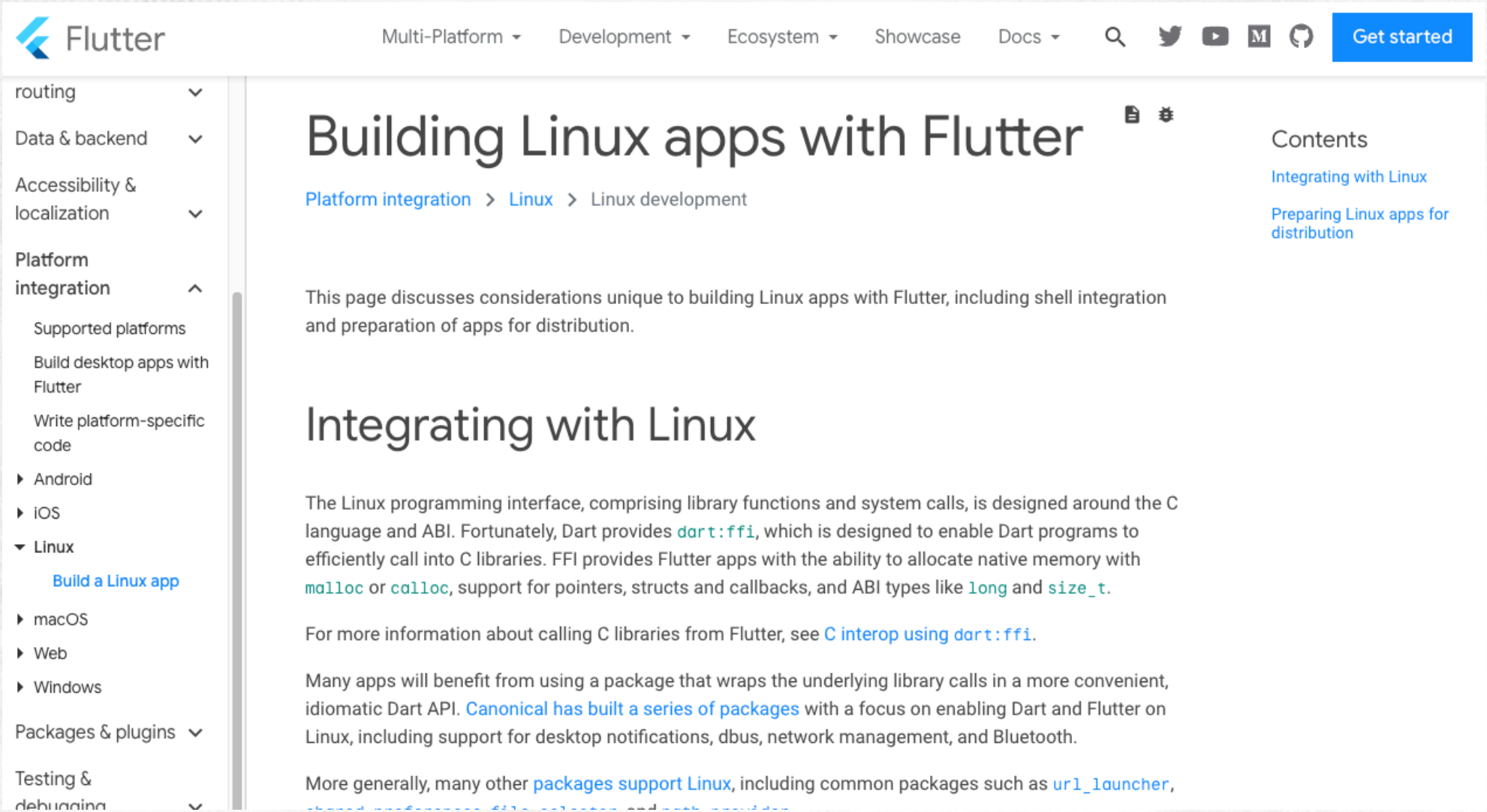
Windows Case

- https://docs.flutter.dev/platform-integration/windows/building



Linux Case

- https://docs.flutter.dev/platform-integration/linux/building



FFI로 C 언어 연결하기 (리뷰 & 실습)

- 교재에서 설명한 환경 설정 등을 수행
- volume-I-chapter-07의 sum_ffi.c를 sum_ffi.dart에서 호출

```dart
import 'dart:ffi';
import 'dart:io';

import 'package:path/path.dart' as path;

typedef SumFunc = Int32 Function(Int32 a, Int32 b);
typedef Sum = int Function(int a, int b);

Run | Debug
void main() {
  // Open the dynamic library
  var libraryPath =
      path.join(Directory.current.path, 'sum_ffi_library', 'libsum_ffi.so');
  if (Platform.isMacOS) {
    libraryPath = path.join(
        Directory.current.path, 'sum_ffi_library', 'libsum_ffi.dylib');
  }
  if (Platform.isWindows) {
    libraryPath = path.join(
        Directory.current.path, 'sum_ffi_library', 'Debug', 'sum_ffi.dll');
  }

  final dylib = DynamicLibrary.open(libraryPath);

  // calls int sum(int a, int b);
  final sumPointer = dylib.lookup<NativeFunction<SumFunc>>('sum');
  final sum = sumPointer.asFunction<Sum>();
  print('3 + 5 = ${sum(3, 5)}');
}
```

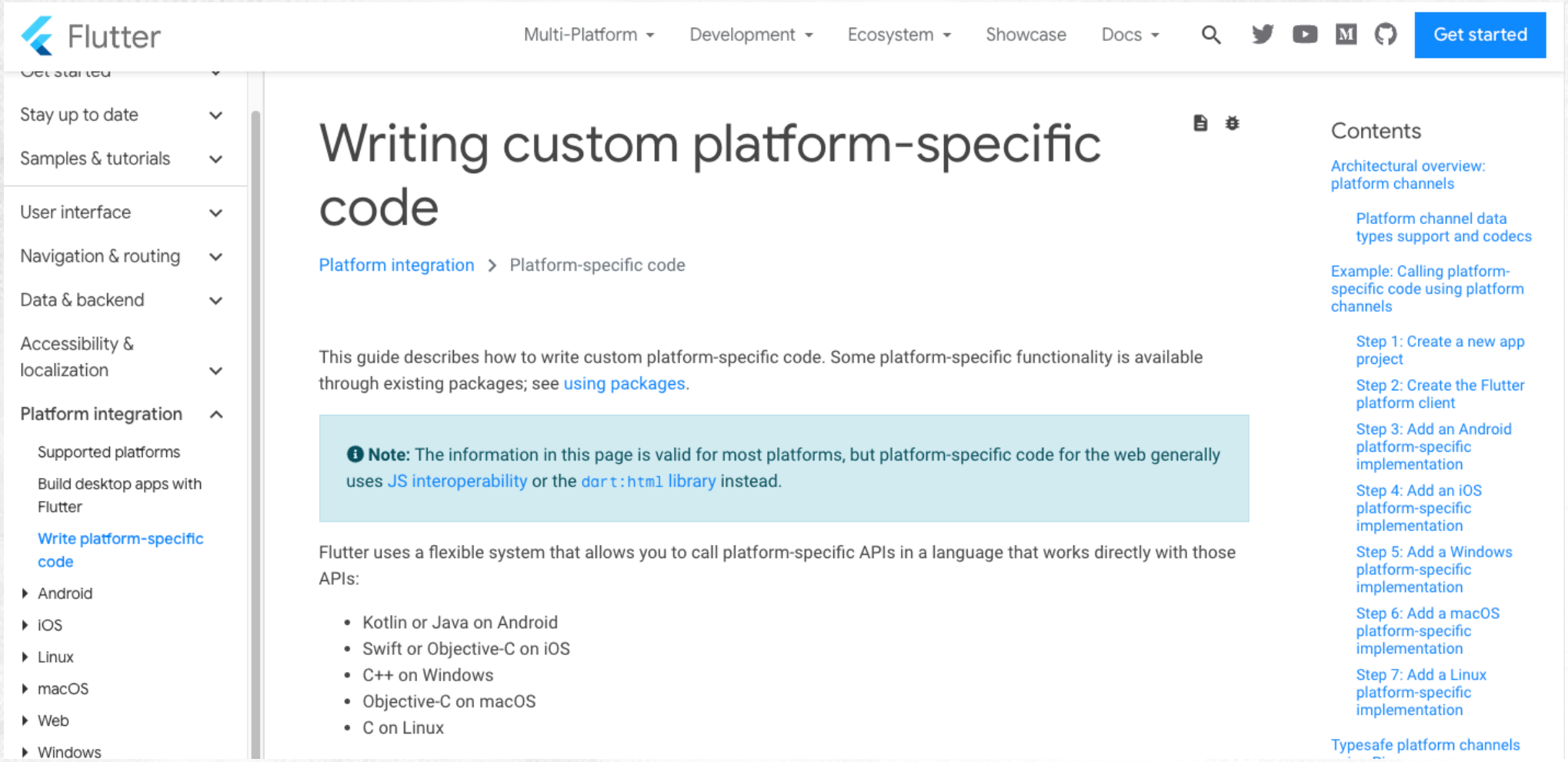
```c
#include <stdlib.h>
#include <stdio.h>
#include <stdarg.h>
#include "sum_ffi.h"

int main()
{
    printf("3 + 5 = %d\n", sum(3, 5));
    return 0;
}

int sum(int a, int b)
{
    return a + b;
}
```

참조: Platform Specific API 호출

- FFI와 별개로, Flutter는 플랫폼 API 호출을 지원함
  - Kotlin or Java on Android
  - Swift or Objective-C on iOS
  - C++ on Windows
  - Objective-C on macOS
  - C on Linux

참조: Platform Specific API 호출

- https://docs.flutter.dev/platform-integration/platform-channels?tab=type-mappings-java-tab



수업에서 다루지 않은 교재 속 이야기들

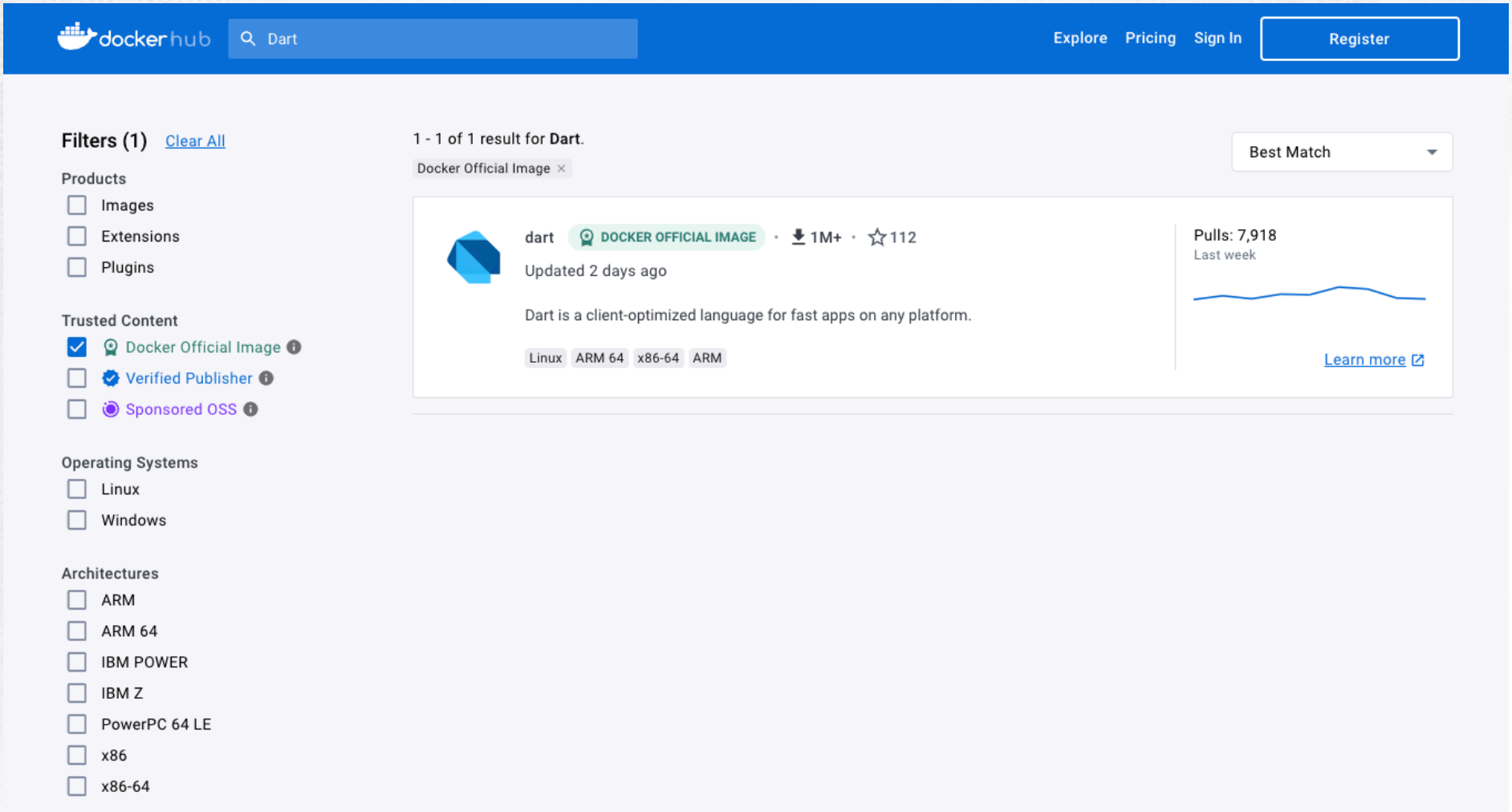
- **Volume.H 지속 가능한 개발자로 첫걸음 내딛기**
  - 01. Dart for Embedded 알아 두기
  - 03. Flutter 공식 사이트 레퍼런스와 샘플 활용하기
  - 04. Flutter CLI 명령 이해하기
  - 05. Dart와 Flutter 최신 정보 신청하기

수업에서 다루지 않은 교재 속 이야기들

- **Volume.I 알아 두면 요긴한 분야별 노하우**
  - 03. Docker를 이용한 컨테이너 기반 서비스 개발하기 서버
  - 04. 서로 다른 디바이스 간에 통신하기 네트워크
  - 05. WebSocket 기반 네트워킹 기능 개발하기 네트워크
  - 08. 라즈베리 파이에서 Dart와 Flutter 활용하기
  - 09. XD2Flutter로 디자인 개선하기 앱
  - 10. Flutter 앱 배포하기 앱
  - 11. Concurrency 기반 병렬처리 개발하기 성능
  - 12. 컴퓨터공학 전문 이론 공부하기 성능
  - 13. TensorFlow 활용하기 인공지능
  - 14. Flutter 클라우드 개발 환경 활용하기 도구

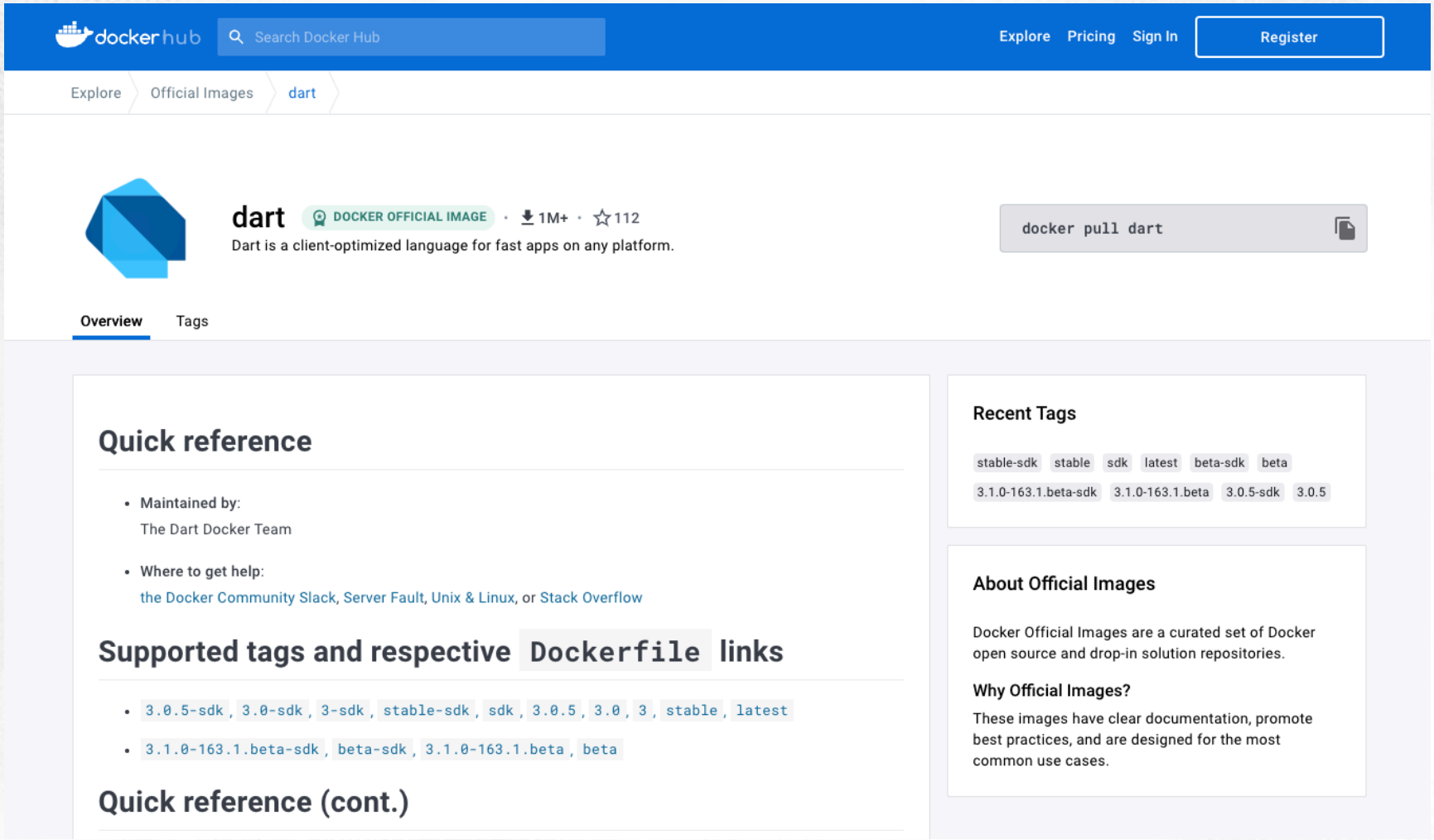
수업에서 다루지 않은 교재 속 이야기들

- **Docker를 이용한 컨테이너 기반 서비스 개발하기 서버**
  - https://hub.docker.com/search?q=Dart&image_filter=official



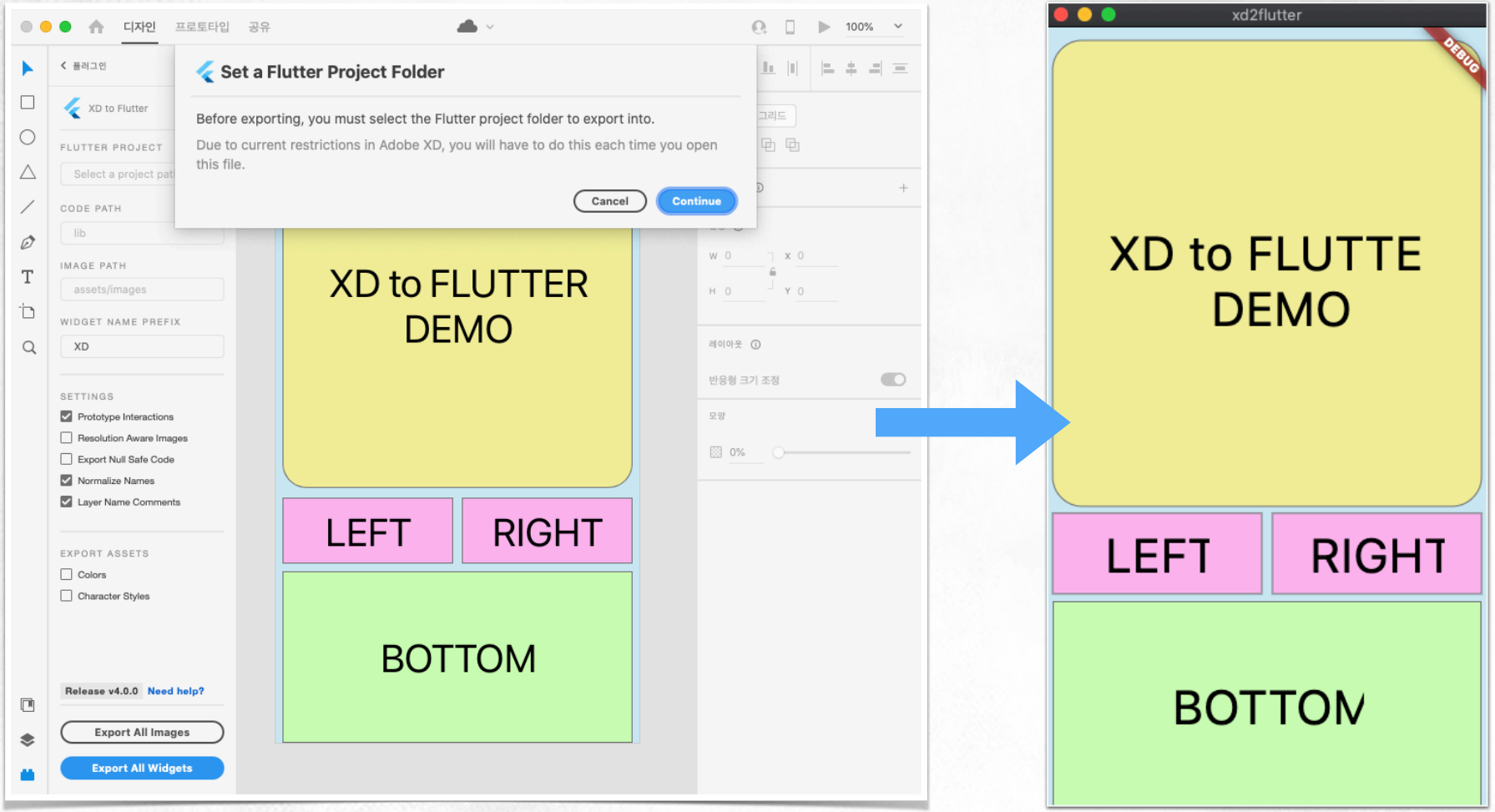
수업에서 다루지 않은 교재 속 이야기들

- **Docker를 이용한 컨테이너 기반 서비스 개발하기 서버**
  - https://hub.docker.com/search?q=Dart&image_filter=official



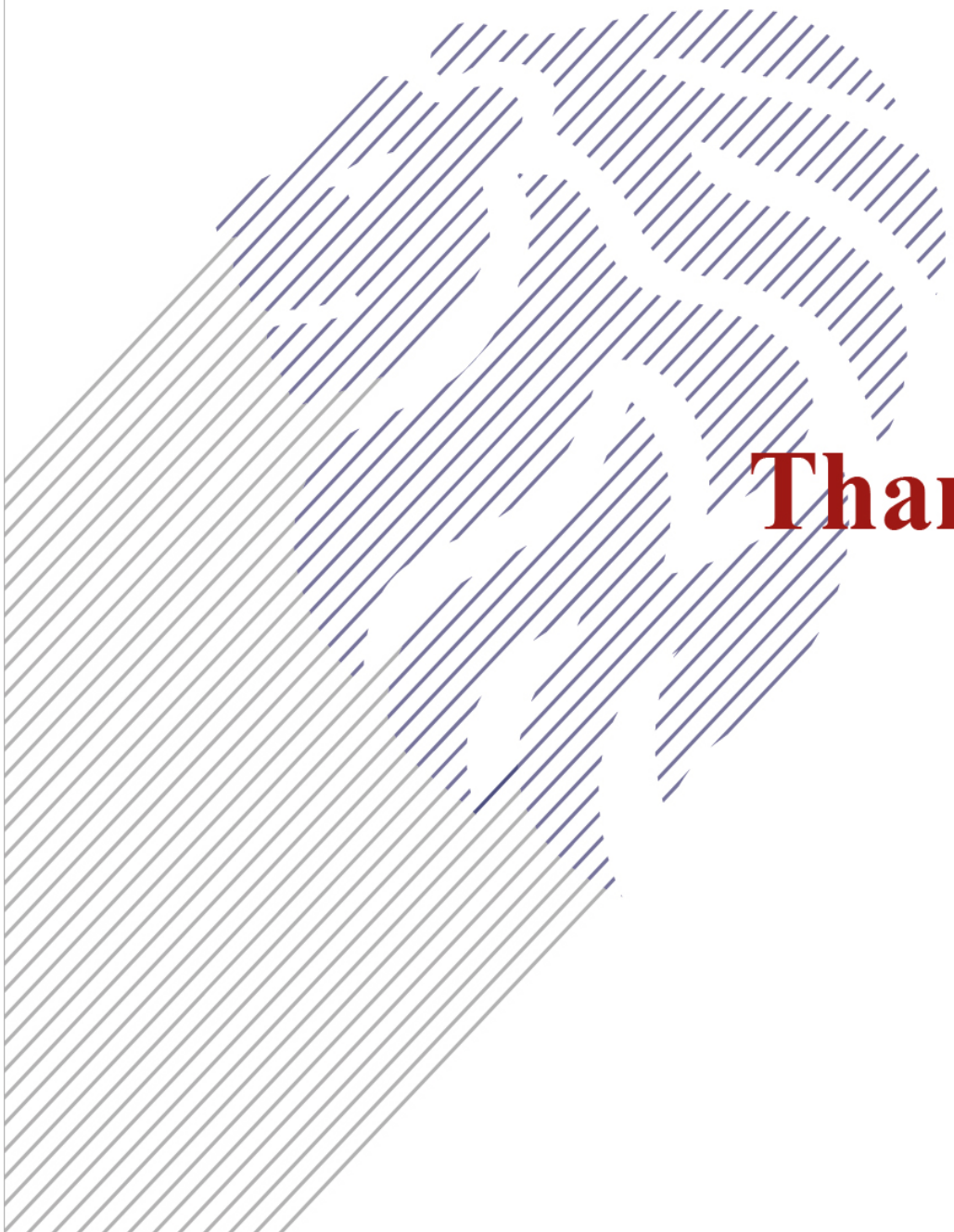
수업에서 다루지 않은 교재 속 이야기들

- **XD2Flutter로 디자인 개선하기 앱**
  - AdobeXD에서 작성한 디자인을 Flutter 코드로 자동 변환



Enjoy Dart & Flutter Life

REFERENCE: https://docs.flutter.dev/dash



Thank you

경희대학교
KYUNG HEE UNIVERSITY