SAMSUNG SW ACADEMY FOR YOUTH

對外秘

# 프로젝트 명세서

Google Flutter 공식문서로 시작하는

크로스 플랫폼 앱 개발 입문

對外秘

목차

# 1. 프로젝트 개요

전통적으로 개발자가 학습을 처음 시작하는 단계에서 고민 하는 것은 바로 “어떠한 SW 개발 언어를 선택하여 학습을 시작 할까?” 였다.

하지만 시대가 변하면서 어떠한 언어를 배울 것인가 보다는 어떠한 Framework 을 사용해야 하는지가 우선적으로 정해지고 해당 Framework 에서 가장 효과적인 언어를 사용하는 방식으로 패러다임이 바뀌었다.

최근에는 굴지의 글로벌 기업들이 자사의 개발 생태계 구축을 위해서 저마다 경쟁적으로 자사의 도구가 “**보다 쉽고 빠르며 효율적**” 임을 내세워 홍보함과 동시에 웹사이트 등에 Reference Guide 즉, “**공식문서**” 를 통해서 유사한 기능의 Framework 간의 Migration 을 어렵지 않게 가이드 하고 있다. 이는 비단 Framework 뿐만 아니라 새로운 기술이나 언어 또한 마찬가지 이다. 이런 상황이다 보니 공식문서를 보고 신기술을 적용 및 응용 할 수 있는 경험과 능력 그리고 자신감 또한 개발자들에게 매우 중요한 요소로 여겨지고 있고 취업시장에서도 자신을 어필하기 좋은 소재로 활용하기 좋다.

본 명세서에서는 공식문서를 보면서 새로운 Framework 을 하루 동안 가볍게 진행해 보는 연습을 해 보는데 주안점을 두고 작성 되었고 이 목적에 걸맞은 새로운 형태의 Framework 으로 상대적으로 많이 알려지지 않은 Google Flutter 를 선정 하였다.

## 1-1. Google Flutter

스마트폰의 생태계는 현재 구글 중심의 안드로이드와 애플 중심의 iOS 로 나뉘어 있고 각각의 생태계에서 실행되는 어플리케이션 즉, 앱(App) 또한 Google 과 Apple 중심으로 완벽하게 갖추어 있다. 그렇기 때문에 안드로이드앱은 구글의 Android Studio 에서 Kotlin 을 사용하는게 가장 좋은 방법이고 iOS 앱은 애플의 Xcode 에서 Swift 언어로 개발하는 것이 가장 좋은 생산성과 안정성을 보장해 준다고 할 수 있다.

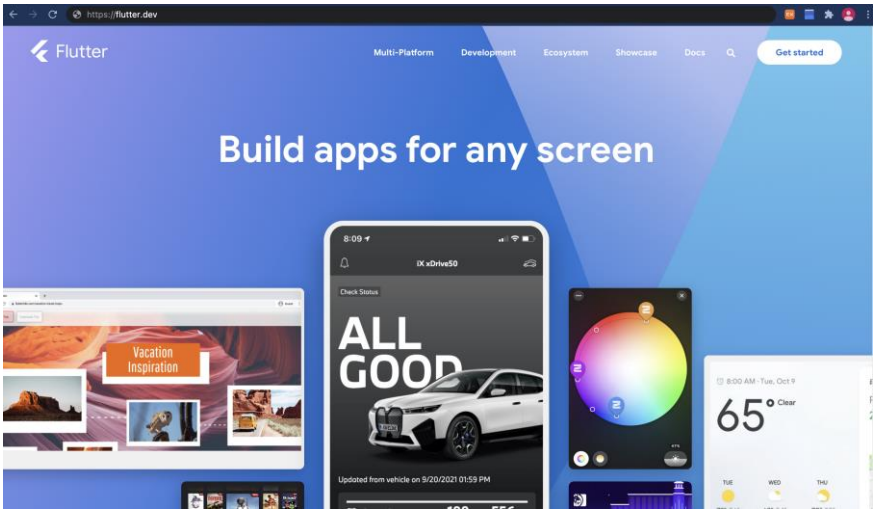
하지만 다음과 같은 상황을 가정해 보자.

1. 하나의 소스를 가지고 여러가지 플랫폼에서 동시에 서비스를 런칭해야 하나 아직 다양한 플랫폼에 대응할 만한 인력이 갖추어 지지 않았다.
2. 볼륨이 크지 않은 서비스이거나 앱 자체가 주된 비즈니스 모델이 아니라 홍보성 및 단발성의 성격이 짙다.
3. 다중플랫폼을 지원하기위해 하이브리드앱 형태로 개발을 하였으나 실행속도가 매우 느려 사용성에 문제가 발생했다.

안드로이드와 iOS 용 앱을 동시에 개발하고 유지보수 하는 일은 가장 높은 수준의 산출물을 얻을 수 있는 방법이라고 할 수 있으나 동시에 기업에게는 비용적으로 많은 부담을 가중시키는 요소이기도 하다.

이런 이유로 하나의 소스코드로 다양한 하드웨어에서 앱을 구동 시킬 수 있는 기술들이 꾸준히 연구되어 왔고, Flutter 는 이러한 Cross-Platform 기반 App Framework 들 중에서 최근 가장 각광받고 있는 Framework 이라 할 수 있다.

Flutter 는 Dart 라는 언어를 사용하며 Android, iOS, Linux, Windows, OSX 및 웹 브라우저에서 모두 동작하는 앱 개발을 위해 고안 되었다.



그러나 Flutter 는 만능이 아니다. 안드로이드와 iOS 앱 개발 경험이 없는 개발자가 처음부터 Flutter 로 앱 개발을 시작 하는 것은 매우 바람직하지 않을 수 있다. 또한 Flutter 를 사용 하더라도 iOS 앱을 빌드하고 테스트 하려면 여전히 Apple 의 컴퓨터와 개발자 라이센스를 필요로 한다.

즉, Flutter 는 대상이 되는 플랫폼 모두를 개발 경험해 보고 환경을 갖춘 개발자가 하나의 소스를 가지고 다양한 Platform 으로 동시에 개발 할 수 있도록 해주는 Framework 이라고 할 수 있다. 따라서 iOS 나 안드로이드 개발 경험이 없다면 오늘 하루는 공식문서를 따라 가볍게 학습을 하되 이후 부터는 Flutter 가 아닌 iOS 와 안드로이드 각각 개발을 해보고 그 이후에 Flutter 를 본격적으로 사용하는 것을 추천 한다.

교육생들은 본 명세서를 통해 하루 정도 가볍게 Flutter 에 대한 학습을 진행해 보고 차 후 본격적인 앱 프로젝트 진행 시 이번에 얻게 된 경험을 토대로 크로스 플랫폼 앱을 개발해 활용한다면 좋은 결과가 있을 것으로 생각 된다.

본 프로젝트의 목표는 다음과 같다.

1. 기존에 사용해 보지 못한 새로운 Framework 을 **공식문서**를 보고 학습하는 경험을 해 본다.
2. Google 의 Flutter App Framework 으 기본 튜토리얼 코드들을 실행해 보는 경험을 한다.
3. Flutter 의 근간이 되는 Dart 언어에 대해 알아 본다.
4. Flutter 와 같은 Cross-Platform 개발 Framework 의 장단점을 생각해 본다.

# 2. 기본과제

본 명세서의 기본 과제는 Flutter 공식문서를 참고하여 기본 개발 환경을 구성 한 후 몇가지 기본 튜토리얼을 실행하도록 가이드 하고 있다.

## 2-1. Flutter 웹사이트 살펴보기

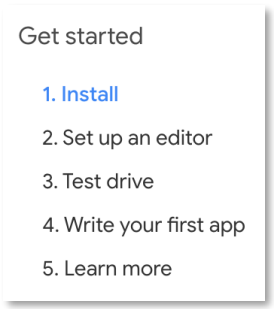
Flutter 웹사이트 (https://flutter.dev)를 방문해 보면 현재 다음과 같은 5 개의 기본 메뉴가 있다.

| Multi-Platform / Development / Ecosystem / Showcase / Docs |
|---|

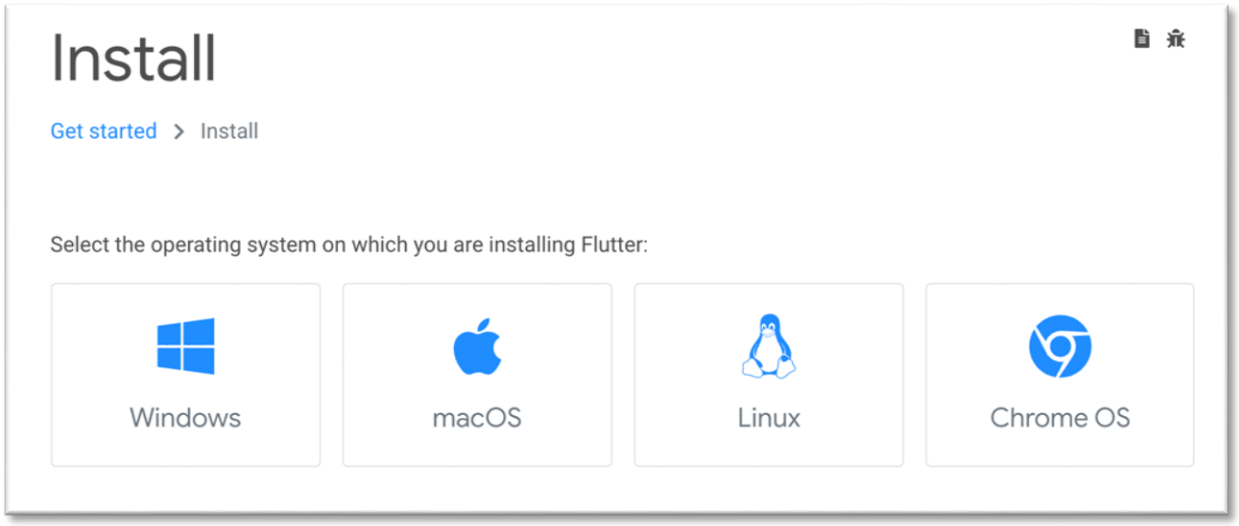
제품의 팜플렛을 읽듯 가볍게 각 메뉴들을 클릭해 읽어보면 좋다.

## 2-2. Flutter 설치

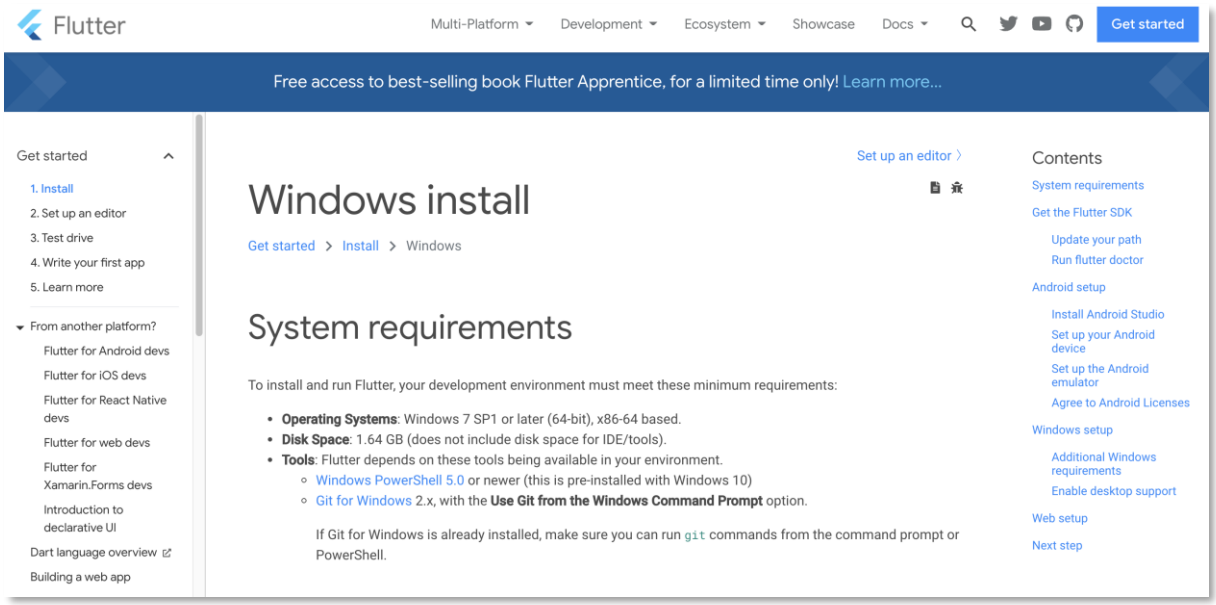
충분히 읽었다면 웹사이트 좌측 상단의 Get started 버튼을 클릭하여 Flutter 설치 페이지로 이동하여 가이드 대로 설치를 시작한다. 공식 문서의 Get started 는 다음처럼 5 개의 섹션으로 나뉘어 있으니 차근차근 따라가면 된다.



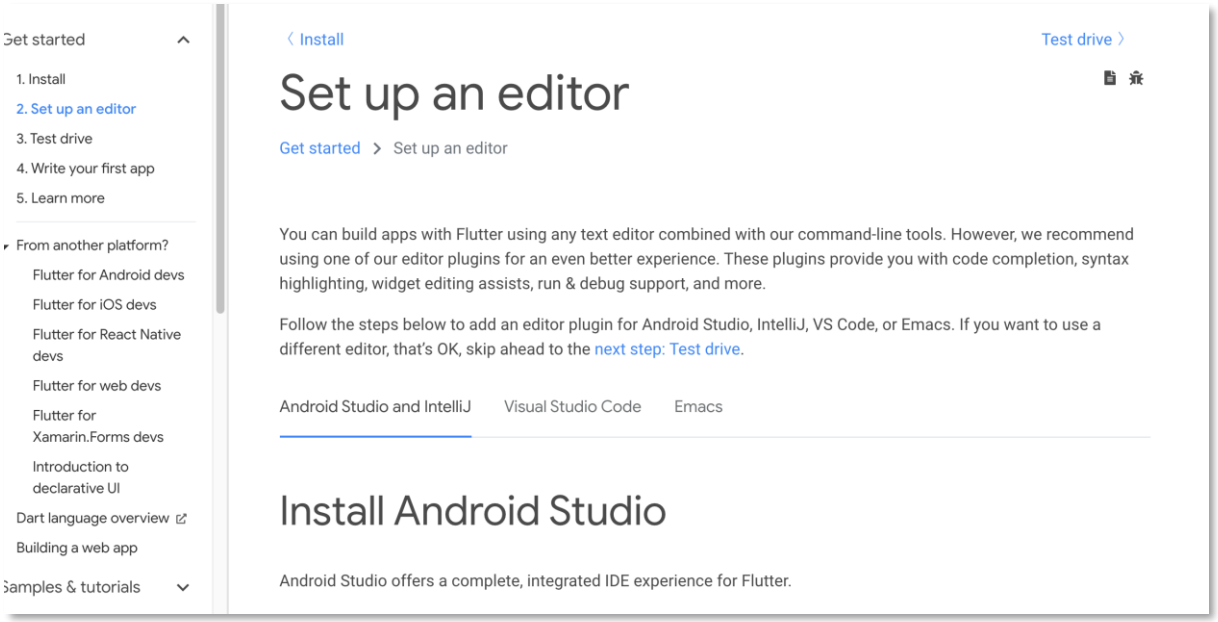
Install 섹션은 사용하는 OS 별로 Flutter 를 설치 할 수 있도록 가이드 해주고 있다. 개인별 시스템에 맞춰 Flutter 를 설치 해준다.




웹사이트의 공식문서를 참고하여 설치된 OS 에 맞게 개발 환경을 구성한다.

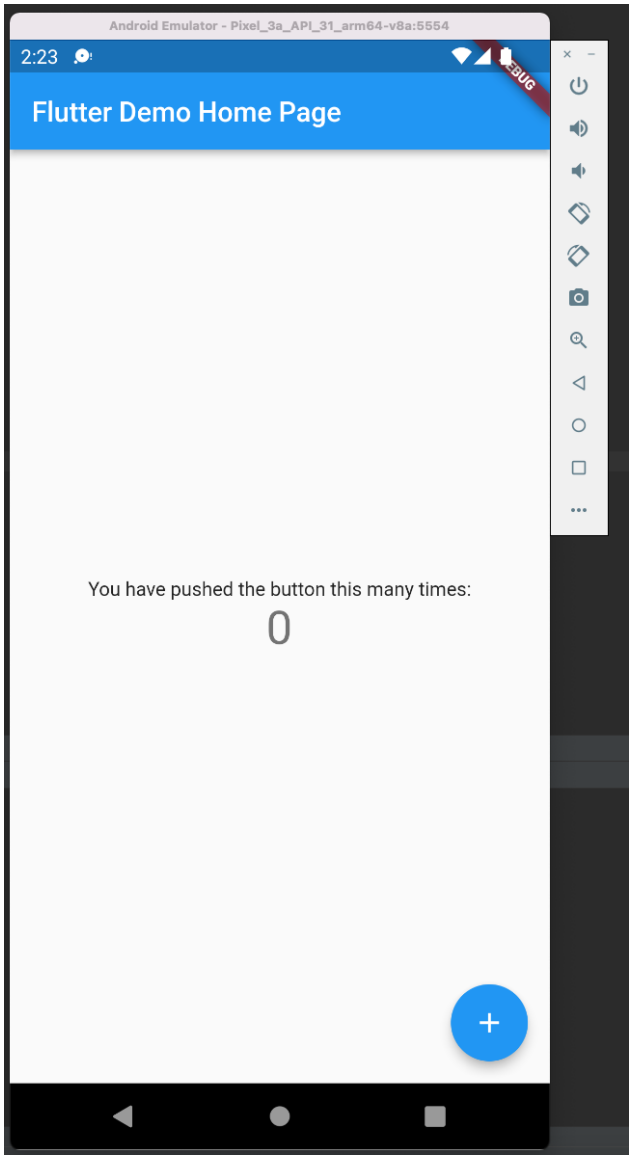


Flutter 는 자체 IDE 를 제공하고 있지는 않고 일반적으로 안드로이드 스튜디오를 설치하고 여기에 Flutter 플러그인을 추가로 설치해서 사용하는 방식으로 개발을 한다. Set up an editor 섹션을 참고하여 코딩시 사용할 에디터를 선택해 설치하면 된다. 일반적으로는 Android Studio 를 추천한다.



## 2-3. Flutter 기본 코드로 테스트 해보기

Flutter 공식문서를 따라 Install 과 Editor Setup 을 마쳤다면 Test drive 섹션을 진행해 보자. 안드로이드 스튜디오를 실행 후 'Create New Flutter Project' 를 선택 후 기본 프로젝트를 선택해 실행을 하면 다음처럼 'Flutter Demo Home Page' 앱이 네이티브 어플리케이션 형태로 빌드 되어 실행 된다.



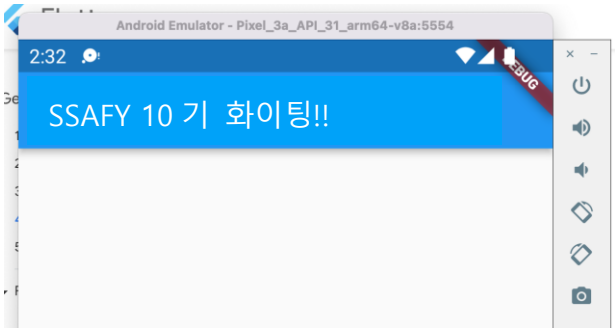
이 섹션의 주된 목적은 기본적인 Flutter 앱을 빌드해 보고 “hot reloaded” 기술을 경험해 보는 것이다. 소스코드 중 “lib/main.dart” 파일을 보면 다음처럼 “Flutter Demo Home Page” 라는 텍스트가 들어있는 부분이 있다.

```dart
class MyApp extends StatelessWidget {
  const MyApp({Key? key}) : super(key: key);

  // This widget is the root of your application.
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      title: 'Flutter Demo',
      theme: ThemeData(
        // This is the theme of your application.
        //
        // Try running your application with "flutter run". You'll see the
        // application has a blue toolbar. Then, without quitting the app, try
        // changing the primarySwatch below to Colors.green and then invoke
        // "hot reload" (press "r" in the console where you ran "flutter run",
        // or simply save your changes to "hot reload" in a Flutter IDE).
        // Notice that the counter didn't reset back to zero; the application
        // is not restarted.
        primarySwatch: Colors.blue,
      ),  // ThemeData
      home: const MyHomePage(title: 'Flutter Demo Home Page'),
    );  // MaterialApp
  }
}
```

해당 부분은 앱의 타이틀에 표시될 텍스트인데 이를 다음처럼 임의로 바꾸고 파일을 저장만 해보자

```dart
        primarySwatch: Colors.blue,
      ),  // ThemeData
      home: const MyHomePage(title:'SSAFY 10기 화이팅!!' ),
    );  // MaterialApp
  }
}
```



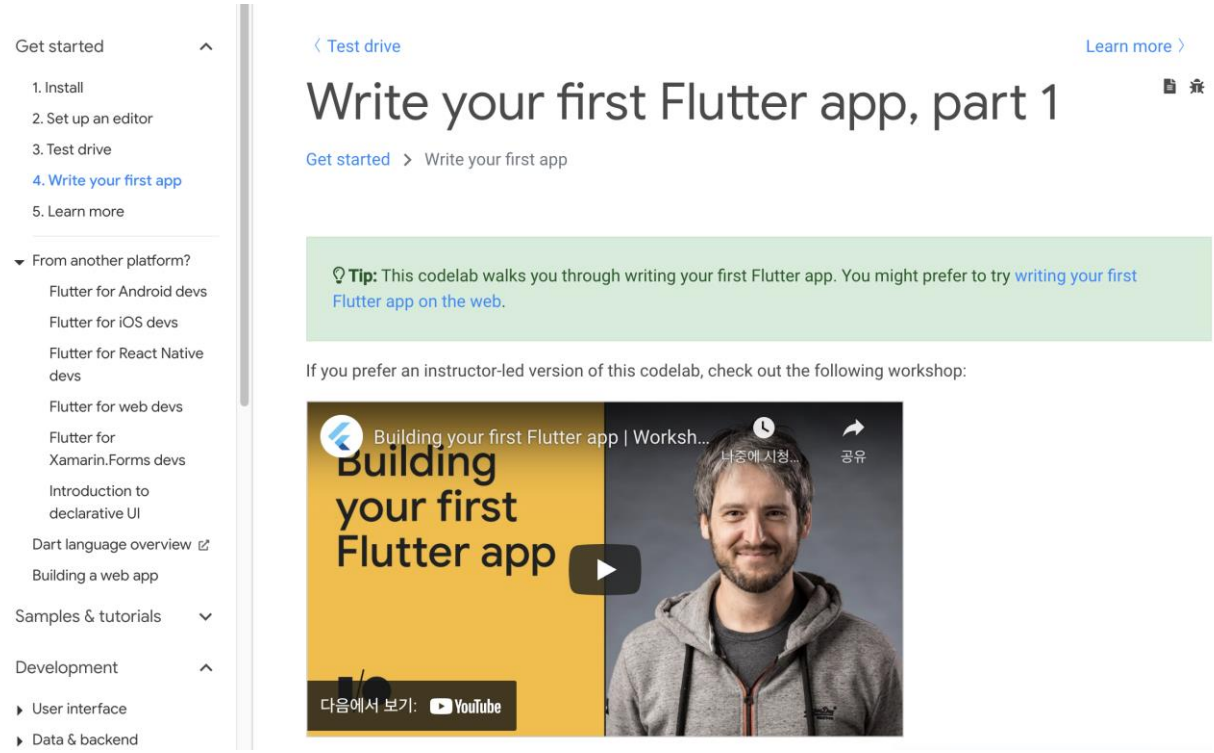

컴파일 과정을 거쳐야 하는 Native App 이 마치 웹페이지처럼 수정된 부분이 바로 반영되는 것을 확인해 볼 수 있다.

다음 Requirements 를 수행해 보고 README.md 파일에 정리 후 제출하시오.

| | |
|---|---|
| Req-1 | Flutter 의 Hot Reloaded 기술이란 무엇인가? |
| Req-2 | Flutter 는 기본적으로 Debug 모드로 실행된다. Debug 모드는 Debugging 정보를 포함해 매우 느리게 앱이 실행된다. 공식문서를 참고해 개발 완료 후 이를 해결하기 위해 어떻게 처리를 해주어야 하는지 간략히 요약하시오. |

## 2-4. Flutter 로 첫 모바일 앱 만들기

다음 섹션인 “Write your first Flutter app, part 1”에서 본격적으로 첫번째 Flutter 앱을 작성해 보자.




다음 Requirements 를 수행해 보고 README.md 파일에 정리 후 제출하시오.

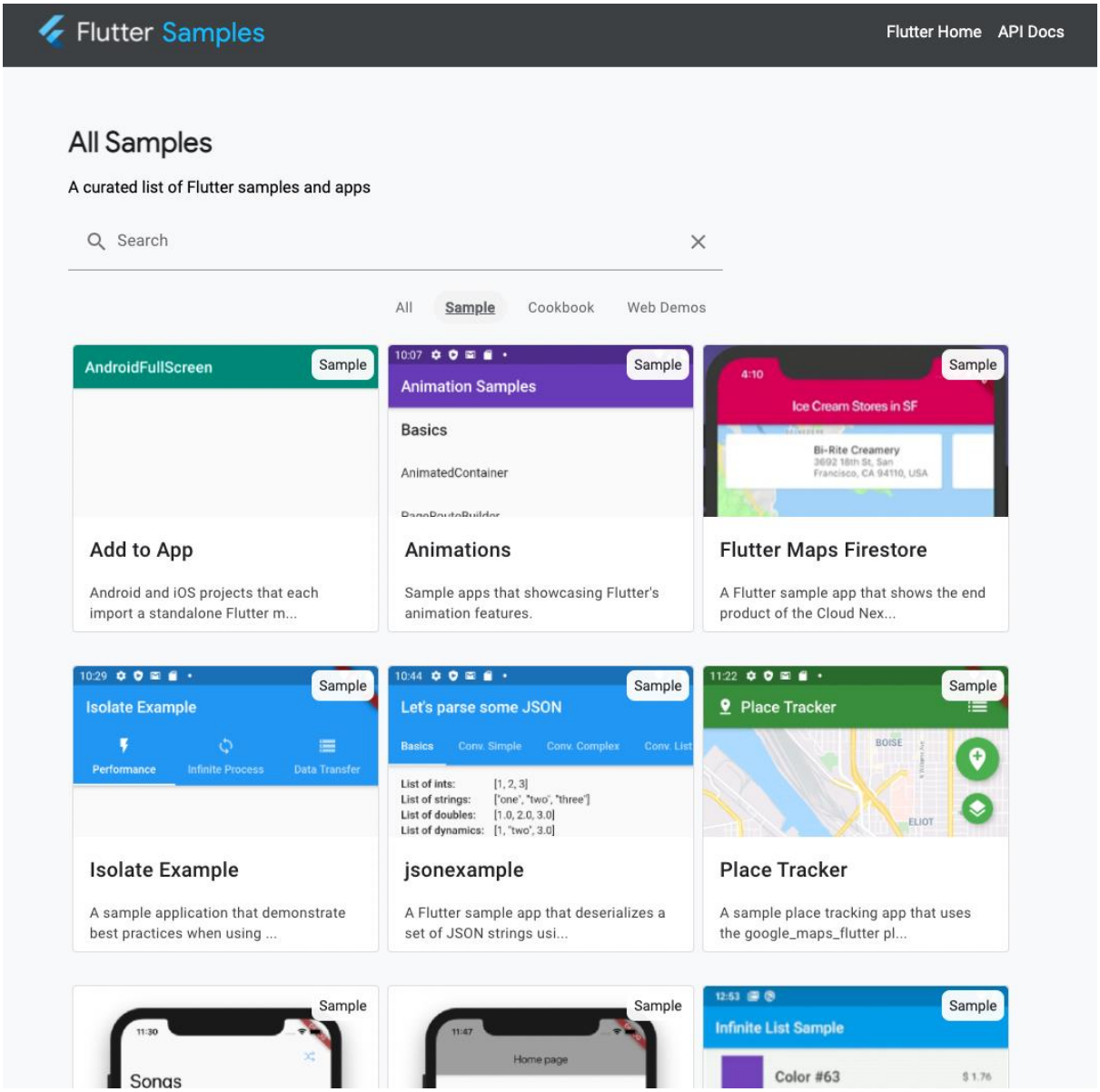
| | |
|---|---|
| Req-3 | Flutter 에서 외부 패키지는 어떤 방식으로 사용하는지 간략히 정리해 보자. |
| Req-4 | Write your first app 파트를 끝까지 구현해 보고 동작시킨 후 스크린샷을 포함에 진행과정중 느낀점을 md 파일에 정리해 보자. |

# 3. 심화과제

심화과제에서는 Flutter 로 구현 가능 한 Sample 들을 추가로 찾아서 빌드 해보고 여러 Target Device 에서 실제로 실행해 보는 경험을 해보도록 하겠다.

다음 웹사이트를 방문해 보자.

- https://flutter.github.io/samples/



다음 Requirements 를 수행해 보고 README.md 파일에 정리 후 제출하시오.

| | |
|---|---|
| Req-5 | 다수의 Flutter Samples 중 관심이 가는 Sample 을 다운로드 받아서 최소한 2 개의 Platform 을 선정하여 자유롭게 빌드해서 실행해 보고 스크린 샷을 남겨보자. |
| Req-6 | 이제 Cross-Platform 앱에 대한 대략적인 개념을 잡았을 것이다. 본인이 개발하고 싶은 Cross-Platform 앱이 있을지 고민해 보고 어떤 형태로 구현하면 좋을지 자유롭고 간단하게 기획하여 md 파일로 작성해 보자. |

서두에 기술한 대로 이 명세서는 공식문서를 보고 새로운 Framework 을 경험해 보는데 그 목적이 있다. 따라서 최대한 부담없이 수행 가능하도록 작성이 되었으나 기술적인 어려움 보다는 처음 사용해보는 기술의 낯선 점이 조금은 껄끄럽지 않았을까 싶다.

오늘 경험을 바탕으로 공식문서를 읽고 새로운 기술을 접하는 자기주도 방식의 학습방법에 자신감을 갖고 2 학기 공통/특화/자율 PJT 를 수행하는데 도움이 되길 바란다.

# 4. 과제 제출방법

1) 작성위치

- {PROJECT_ROOT}/self-project/con10/rep02/**README.md**

2) 작성내용

A. 생성한 기본과제 프로젝트와 README.md 파일을 gitlab 을 통해 제출한다.

B. 스크린샷등 모든 산출물은 README.md 만 열어서 확인 할 수 있도록 정리해 제출한다.

C. 심화과제는 별도의 소스코드를 제출 할 필요없이 README.md 파일에 내용을 정리해 제출한다.

3) 제출 경로

https://lab.ssafy.com/s10-study/self-project/ 의 “산출물 제출 가이드”.docx 참조