초격차 패키지 Online.

안녕하세요,
Part 1 입문강의를
맡게 된, 유병욱입니다.

Chap1 | 오리엔테이션
Flutter / Dart 언어에 대한 소개

Chap2 | Dart Language 1
Dart 언어와 프로그래밍 언어에 대한 기초 소개

Chap3 | Flutter 개발환경 구축하기
개발을 위한 기초 환경 준비하기

Chap4 | Dart Language 2
Dart 언어에 대한 입문 소개

Chap5 | Dart 언어로 프로그래밍 시작하기
가위바위보 프로그램 만들어보기

오리엔테이션

Cross Platform, Flutter

Flutter가 무엇일까요?

Flutter, 플러터는

Google에서 개발한 크로스 플랫폼 개발 프레임워크입니다.
iOS와 Android 모바일 플랫폼에 지원 되는 것으로 현재 많이 알려져 있으며,
현재는 Windows / MacOS / Linux 등의 Desktop Application과
Web Application 또한 Flutter를 활용하여 제작 할 수 있습니다.

이번에 준비 한 강의에서는 Mobile Platform을 위주로 진행할 예정이지만,
각 Platform 개발을 위한 별도의 지식이 많이 필요 없을 정도로
간단하게 Application을 만들 수 있습니다.

지금 여러분들이 Flutter를 배워야 하는 이유

- Easy
- High Productive
- Flexible
- High Performance
- Open Source
- Simple & Beautiful

지금 여러분들이 Flutter를 배워야 하는 이유

Easy

Flutter는 다른 프레임워크에 비해 쉽습니다.

다소 난이도가 낮은 언어인 Dart,
그리고 선언형 UI 구조의 채택으로 UI 코드와 로직 코드가
하나의 파일, 함수 내에서 한 번에 써 내려갈 수 있습니다.

또한 UI 구조를 각각의 Widget 단위로 개발하여
쉽게 다른 Widget들과 조합 / 배치하여 레이아웃을 구성하기에도
상당히 용이한 구조입니다.

지금 여러분들이 Flutter를 배워야 하는 이유

High Productive

Flutter는 생산성이 매우 높습니다.

하나의 코드 베이스로 여러 Platform을 동시에 개발 할 수 있으며,
그 만큼 인적 / 물적 자원을 어느정도 절약할 수 있습니다.

또한 만들어진 코드를 재활용하고 수정하는데에 용이하도록
객체 지향 프로그래밍을 지원하고 있으며,

효과적인 프로그래밍 방법인
함수형 프로그래밍 또한 지원하고 있습니다.

지금 여러분들이 Flutter를 배워야 하는 이유

Flexible

Flutter는 다양한 디스플레이를 지원합니다.

작은 모바일부터 중간 사이즈의 태블릿 PC 그리고
Web과 Desktop을 위한 큰 사이즈의 모니터 까지
현존하는 대다수의 디스플레이에 각각의 사이즈에 알맞은 레이아웃을
제공 할 수 있습니다.

최근엔 폴더블 같은 가변형 디스플레이에
특화된 UI를 구현 할 수도 있습니다.

지금 여러분들이 Flutter를 배워야 하는 이유

High Performance

Flutter는 Native에 가까운 성능을 발휘 할 수 있습니다.

UI와 로직 코드를 모두 Native로 변환하여 작동 시키는 것이 아닌,
별도의 UI 엔진을 활용하여 더욱 효율적인 방법으로
Application을 구동 할 수 있도록 설계 되어있습니다.

기존의 SKiA 엔진부터 곧 업데이트 될 Impeller 엔진 까지
주기적으로 업데이트가 이루어지고 있으며,
특히 이 엔진들은 C / C++ 언어로 이루어져,
호환성과 성능 두 가지를 모두 지원하기 용이합니다.

지금 여러분들이 Flutter를 배워야 하는 이유

Open Source

Flutter는 오픈소스입니다.

Flutter는 Google이 만들었지만, Google만의 것은 아닙니다.

모든 소스가 Github에 공개 되어있고,
누구나 Flutter의 소스코드에 기여할 수 있습니다.

많은 사람들이 우려하는 요소 중 하나인,
Google이 Flutter를 포기하면 어떻게 될 지 걱정하지만,

사실 이미 Google 외에도 수많은 개발자들이 기여하고
발전 하고 있는 프로젝트입니다.

지금 여러분들이 Flutter를 배워야 하는 이유

Simple & Beautiful

Flutter는 심플하면서도 예쁜 디자인을 쉽게 구현할 수 있습니다.

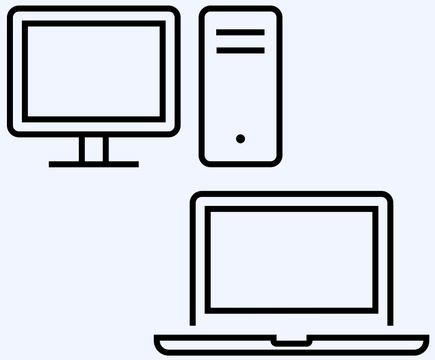
Flutter의 기본 UI Widget들은
Material Design을 기반으로 제작 되었습니다.

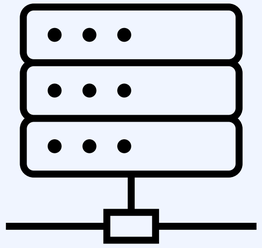
이 덕분에 디자인적 감각이 다소 부족한 개발자들도
간단한 코드 구현을 통해 수려하고 효과적인 UX를
손쉽게 구현해 낼 수 있습니다.

또한 최근 발표 된 Material 3.0에 대해서도
지원이 이루어지고 있어 개인화 된 앱에 대해서도
개발이 가능합니다.

Flutter로 개발하기 위해서 필요한 것은?

Flutter, 플러터를 사용하기 위해서는





Optional

iOS 개발의 경우엔 Mac 환경의 PC가 필요합니다.
(MacBook 또는 MacMini 등)

Next

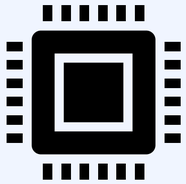
What is Dart?

프로그래밍 언어란?

Dart란 무엇일까요?

프로그래밍 언어 란?

프로그래밍을 통해 특정한 문제를 해결하기 위해,
논리적인 **알고리즘**을 코드로 구현 한 것



CPU (Device)

저급언어

고급언어

Developer (Person)

저급언어

기계가 더 잘 알아 볼 수 있는 언어로, 실행속도가 빠른것이 특징
대표적인 언어로는 0과 1로 이루어진 **기계어**나
기호코드 위주로 이루어진 **어셈블리어**

고급언어

저급언어보다 사람이 조금 더 알아보기 쉬운 언어로, Compile 등의 번역과정이 필요
흔히 알려진 프로그래밍언어인 **C, C++, Java, Kotlin, JavaScript, Swift** 등이 여기에 속하며,
이번에 배우게 될 **Dart** 또한 고급언어 중 하나

Dart가 무엇일까요?

Dart, 다트는

Google에서 개발한 고급 프로그래밍 언어입니다.
객체지향 프로그래밍 언어이며, 안정적이고 빠르게 어플리케이션을 만들기 위해 개발 되었습니다.

Dart는 다양한 플랫폼에서 실행 될 수 있도록 설계 되어있으며,
두 가지의 컴파일러(JIT, AOT)를 보유하고 있다는 것이 특징입니다.

또한 JavaScript / Python과 같은 언어와는 별개로 강력한 타입 시스템을 갖추고 있어,
코드의 안정성과 가독성을 높일 수 있습니다.

Dart의 Compiler

Dart는 다른 프로그래밍언어와는 다르게, 2가지의 Compiler를 가지고 있습니다.

JIT Compiler
(Just In Time)

코드 실행 시점에 컴파일을 수행할 수 있는 컴파일러로,
빠른 개발과 디버깅을 할 때 주로 사용되는 컴파일러.
Flutter 개발의 특징인 Hot-Reload / Hot-Restart를 구현 할 때 활용

AOT Compiler
(Ahead Of Time)

사전에 미리 컴파일을 수행하여 높은 실행속도를 필요로 할 때 활용되는 컴파일러로,
코드 내부를 보호할 수 있으며, Dart 코드를 다른 플랫폼에서 실행 할 수 있도록
컴파일 후 배포 가능한 바이너리 파일을 생성하는 것이 특징

Dart는 너무 생소한 언어

사실 Dart 가 다른 언어인 Java, Kotlin, Swift, JavaScript 만큼 유명한 언어는 아닙니다.

다만 Google에서 처음 만들었을 때에는 JavaScript에서 영감을 받아 개발한 언어로,
그 형태가 JavaScript와 유사한 점이 많으며,

익히기 또한 다른 언어들에 비해서 어렵지 않고,
타 언어 경험이 있으시다면 손쉽게 익힐 수 있는 언어입니다.

P.S)
다만 Dart언어만의 특징인 mixin이나 일급함수 개념이 있기는 하지만,
실제 서비스 개발 시에도 이 개념을 쓰는 경우가 자주 있지는 않습니다.

Dart는 어디서 써볼 수 있을까요?

Dart Pad
(Web Browser)

Android Studio
or
IntelliJ

VS Code

Next

Hello, Dart

Dart Programing 1

1 변수와 타입
2 연산자
3 클래스와 생성자

변수란 무엇일까요?

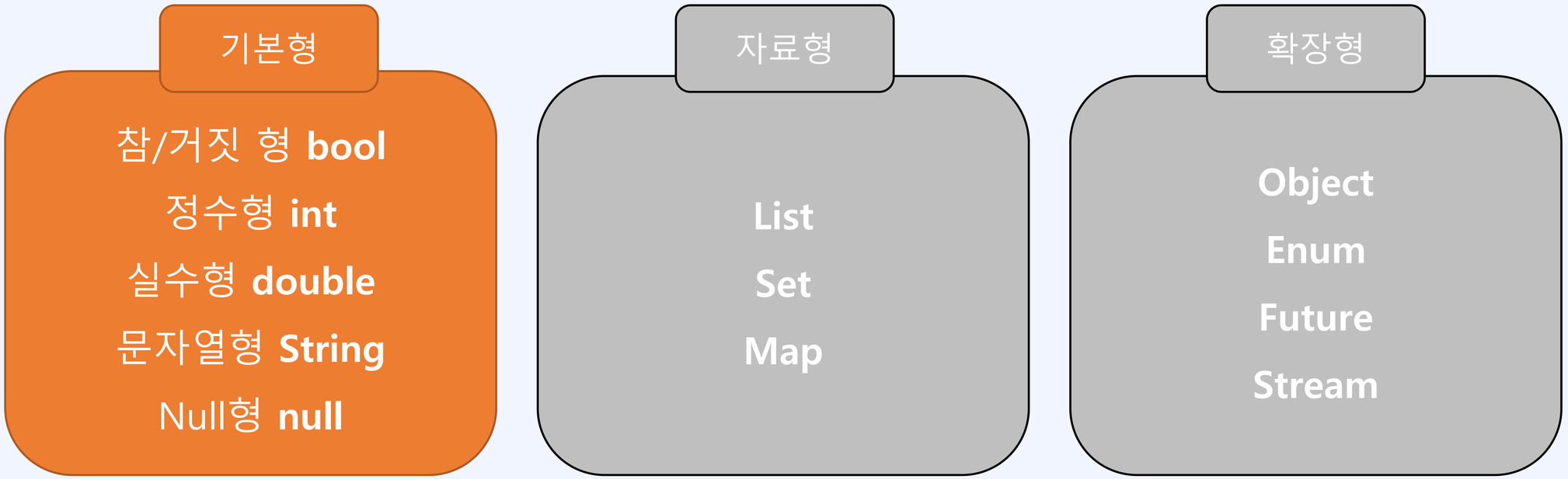
- Dart를 활용하여 프로그래밍 할 때 가장 기본이 되는 단위
- 특정한 값(데이터)를 담아두는 그릇
- 변수에는 변수명을 정하는 선언 그리고
  값을 저장하는 할당 두 가지를 가장 많이 활용하게 됩니다.
- 우선 변수를 선언 할 때에는 선언과 동시에 **타입** 을 정해 선언을 하게 됩니다.
- 그렇다면 타입이란 건 무엇일까요?

타입이란 무엇일까요?

- 데이터의 유형을 타입이라고 합니다.
- 크게는 기본형과 확장형 두 가지로 나뉩니다.

타입이란 무엇일까요?

- 데이터의 유형을 타입이라고 합니다.
- 크게는 기본형과 확장형 두 가지로 나뉩니다.



타입은 꼭 정의 해야 할까?

- 반드시 정의 할 필요는 없음.
- 하지만 프로그래밍 특성 상, 주고 받는 타입에 대한 정의가 명확해야,
  추후에 코드를 관리하고 다른 사람과 협업하는 데에 큰 도움이 됨.

가변형

var : 최초 한 번 부여 된 타입이 고정적으로 사용.

Dynamic : 타입이 코드 진행 중 에라도 언제든 변환.

변수 외에 값을 저장 할 수 있는 방법은?

- 변수는 한 번 할당한 값을 여러 번 수정 할 수 있는 것이 특징
- 그렇다면 값을 한 번 할당하면 바꿀 수 없는 것은? 상수
- Dart에는 상수를 선언하는 방법이 두 가지가 존재

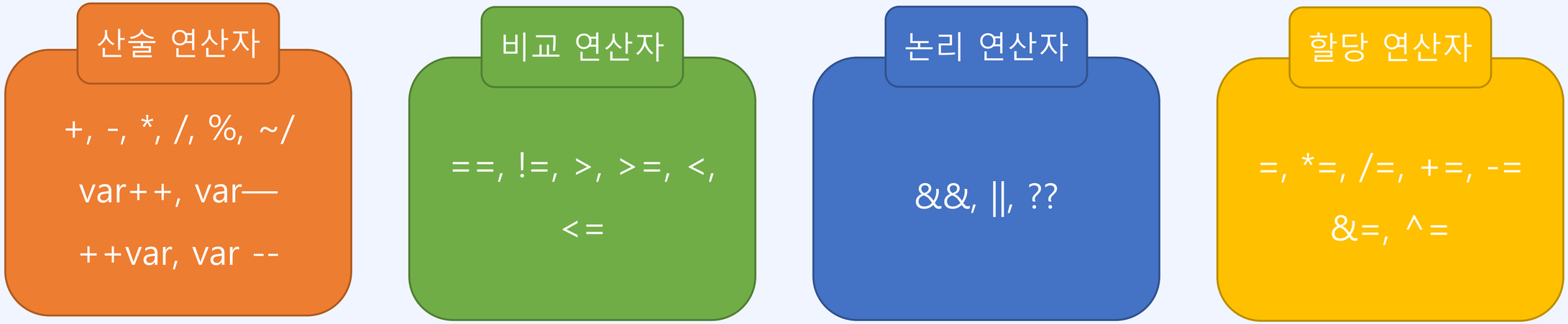
Let's Try

Dart pad

(https://dartpad.dev)

연산자란?

- 프로그래밍 언어에서 사용되는 기호 혹은 단어로,
  하나 이상의 변수나 값을 가지고 수행할 연산을 나타내는 단위



- 이외에도 삼항 연산자, 비트 연산자 등 여러 연산자가 존재

Null Safety 타입 / 연산자

- Dart는 2.12 버전부터 Null값에 대한 안정성을 위해 Null Safety 라는 개념이 도입 됨.
- 관련하여 추가 된 타입과 연산자가 존재

Nullable Type

Null을 허용하는 타입 / 변수 뒤에 ? 을 붙여 활용

Ex) int?, double?, bool?, String?...

Non-nullable Type

Null을 비허용하는 타입 / 변수 뒤에 ! 을 붙여 활용 (이 경우 해당 값이 null인 경우 에러 발생)

Ex) int!, double!, bool!, String!...

Let's Try

Dart pad

(https://dartpad.dev)

클래스, Class란?

- 클래스는 일종의 객체를 만들기 위한 Template
- 클래스를 활용하여 일종의 데이터와 코드를 그룹화해서 관련된 코드를 같이 유지하고, 객체를 쉽게 만들어 객체지향 프로그래밍을 효과적으로 활용할 수 있습니다.

Class의 구성요소

- 필드 : 클래스 내부에 선언된 데이터 (변수 / 상수 등)
- 메서드 : 클래스 내부에 선언 된 기능 (함수)
- 생성자 : 클래스 인스턴스를 생성할 때 사용되는 코드,
  생성 시 특정 작업을 지시 하는 등의 활용이 가능

생성자, Constructor란?

- 클래스의 인스턴스를 생성하는 데 사용되는 코드입니다.
- 생성자를 통해 매개변수를 전달하거나, 클래스 내 필드의 초기값을 설정하는 등의 작업을 할 수 있습니다.

- Default constructors : 기본 생성자로, 생성자를 선언하기 않을 경우 제공되는 생성자.
- Named constructors : 개발자가 필요에 의해 생성한 생성자로, 클래스에 대한 여러 생성자를 구현하거나, 추가적인 클래스의 명확성을 제공.
- Redirecting constructors : 목적이 동일한 생성자로 전달하기 위한 생성자로, 생성자의 본문은 비어 있지만, 전달 된 생성자에 대한 초기값 등을 구현할 때 활용.
- Const constructors : 상수 생성자로, 클래스가 불변의 객체를 생성하는 경우 활용.
- Factory constructors : 매번 새로운 인스턴스를 만들지 않는 생성자를 활용할 때 사용합니다.
  이미 존재하는 인스턴스를 반환하거나,
  단순한 초기값을 부여가 아닌 연산이 필요한 객체 생성 시 활용합니다.

Let's Try

Dart pad

(https://dartpad.dev)

Flutter / Dart 개발 환경 구축하기

1 개발 환경이란?
2 Dart / Flutter 개발환경 설치하기
3 IDE 설치하기

개발 환경이란?

우선, 프로그래밍을 시작하려면,

프로그래밍 언어의 SDK (Software Development Kit)와
IDE(Integrated Development Environment)이 필요합니다.

이렇게 프로그래밍에 필요한 기본 세팅을 개발 환경이라고 하며,
Flutter로 어플리케이션을 개발하기 위해선 마찬가지로,
Flutter 개발 환경을 구축 하여야 합니다.

Flutter의 개발환경 구축을 위해서는 개발에 사용되는 OS 별로 설치하는 방법이
조금 다른 부분이 있으니,
강의를 들으시는 여러분의 환경에 따라 알맞은 내용을 찾아 설치하시기 바랍니다.

SDK란?

Software Development Kit 의 약자로,
프레임워크 등을 개발에 활용하는 데에 있어 필요한
기본 라이브러리, 도구, 문서 등을 모아둔 패키지를 말합니다.

Flutter SDK에는 기본적으로 Dart가 내장되어 있어서 Dart를 별도로 설치 할 필요없이,
Flutter SDK만 설치하는 것으로 언어와 프레임워크 모두를 한 번에 설치 할 수 있는 게 특징이며,

Flutter SDK는 Flutter 공식 홈페이지에서 내려 받을 수 있습니다.

SDK의 경우에는 사용하는 OS (운영체제)에 따라 설치 하는 방식이 차이가 있으므로,
개발에 사용할 OS를 미리 알아 두어야 합니다.

이번 강의에서는 정석적으로 Flutter SDK를 설치하는 방법.

그리고, Flutter는 버전이 자주 변하다 보니, 버전 변경을 용이하게 하고 SDK 설치도 더욱 쉽게 할 수 있는
FVM (Flutter Version Manager) 를 통한 설치 두 가지 모두를 진행 할 예정입니다.

환경변수란?

- 환경변수는 컴퓨터 프로그램이 실행되는 동안 사용할 수 있는 값을 저장하는 변수
- 환경 변수는 프로그램의 실행 환경을 설정하는 데 사용되며,
  프로그램이 다른 프로그램과 상호 작용 할 수 있도록 합니다.
- 개발에 있어서 환경변수가 중요한 이유는, 컴파일러나 SDK의 실행 파일의 위치를 지정 할 수 있습니다.
  각기 다른 디바이스에서 개발을 진행 할 경우 SDK의 설치 위치도 제각각이기 때문에,
  환경 변수로 해당 프로그램의 이름 등으로 절대 경로를 대체하여
  동일한 명령어로도 다양한 디바이스에서도 같은 동작을 할 수 있게 도와줍니다.
- 프로그래밍 개발을 시작 하기 전 환경 변수 설정이 가장 큰 벽 중에 하나입니다.
  아무래도 개발을 하면서 몇 번 접하지 않는데다,
  운영체제 설정이나 터미널에 명령어를 직접 기입 해야하다보니 초보자입장에서는 더더욱 부담입니다.
- 하지만 본질을 알고 나면 엄청 어렵거나 난해한 개념은 아닙니다.
  설정 도중 문제가 발생 시 Discord 문의 창을 통해 문의 해 주세요.

IDE란?

- IDE (통합 개발 환경) 는 프로그래머가 소프트웨어를 개발하는 데 사용할 수 있는 소프트웨어입니다.
- 소스코드 편집기, 디버거 및 빌드 시스템과 같은 개발에 필요한 다양한 도구를 통합하여 제공합니다.
- Flutter / Dart를 활용하기 위해서는 아래의 IDE가 주로 많이 사용 됩니다.
  - Android Studio

    Android Studio는 Google과 JetBrains에서 개발한 IDE입니다.
    Android를 개발하는 데에 특화되어 있지만,
    Flutter Plugin을 설치하여 Flutter IDE로도 쓸 수 있으며,
    Android 개발을 위한 기본 도구들도 함께 사용 할 수 있습니다.
  - Visual Studio Code

    VS Code는 Microsoft에서 개발한 오픈 소스 IDE로, 경량화 된 IDE의 대표주자 중 하나입니다.
    역시 Flutter Plugin을 설치하여 활용 할 수 있습니다.

Let’s Try

운영 체제에 맞게
다음 강의를 선택하여 들으세요.