

크로스 플랫폼

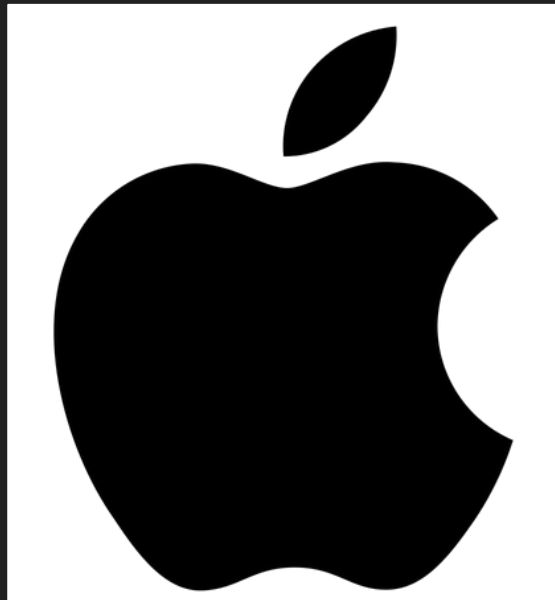
React-Native, Flutter

크로스 플랫폼이란?

크로스 플랫폼의 등장배경



Android



ios

크로스 플랫폼의 등장배경

Swift/Objective-C



Kotlin/Java

크로스 플랫폼의 등장배경

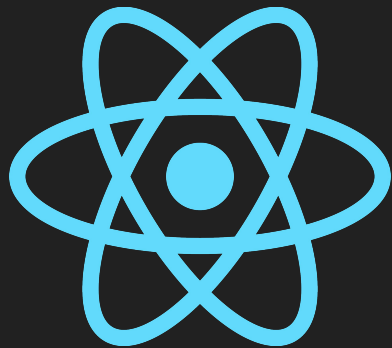
개발 인력 2배



유지 보수 두배

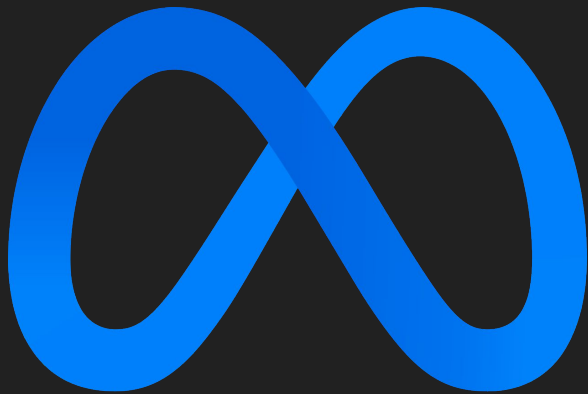
크로스 플랫폼들 소개

React-Native

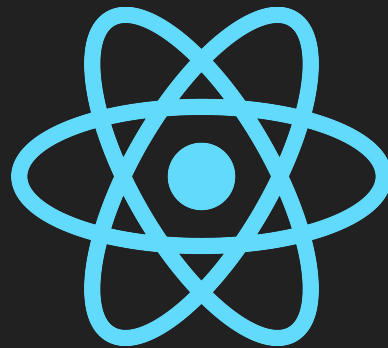


한번 배우면 어디서든 쓸 수 있다 (Learn Once, Write Anywhere)

React-Native

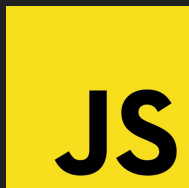
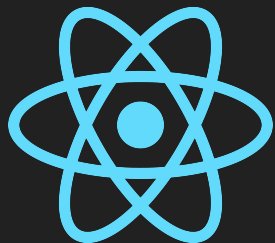


Meta



React-Native

React-Native



JS 사용



npm 과 같은 방대한 커뮤니티 사용

Flutter



모든 것이 위젯이다 (Everything is a Widget)

Flutter



Google



Flutter



Flutter



Skia 엔진 사용

어떤 서비스를 선택해야 될까?

두 플랫폼을 선택할 때 중요성

	장점	단점
 Flutter	훌륭한 공식 라이브러리 생태계 React Native보다 더 많이 사용됨	높은 학습 비용
 React Native	Codepush 기능 사용 가능 채용에 용이한 풍부한 개발자 풀	빈약한 공식 라이브러리 생태계

시장 플랫폼 점유율 현황



App Store



1. Swift

79%



2. Flutter

14%



3. React Native

13%



4. Cordova

10%



5. MetalKit

7%



6. Unity

5%



Google Play



1. Kotlin

79%



2. Flutter

22%



3. Jetpack Compose

19%



4. React Native

18%



5. Cordova

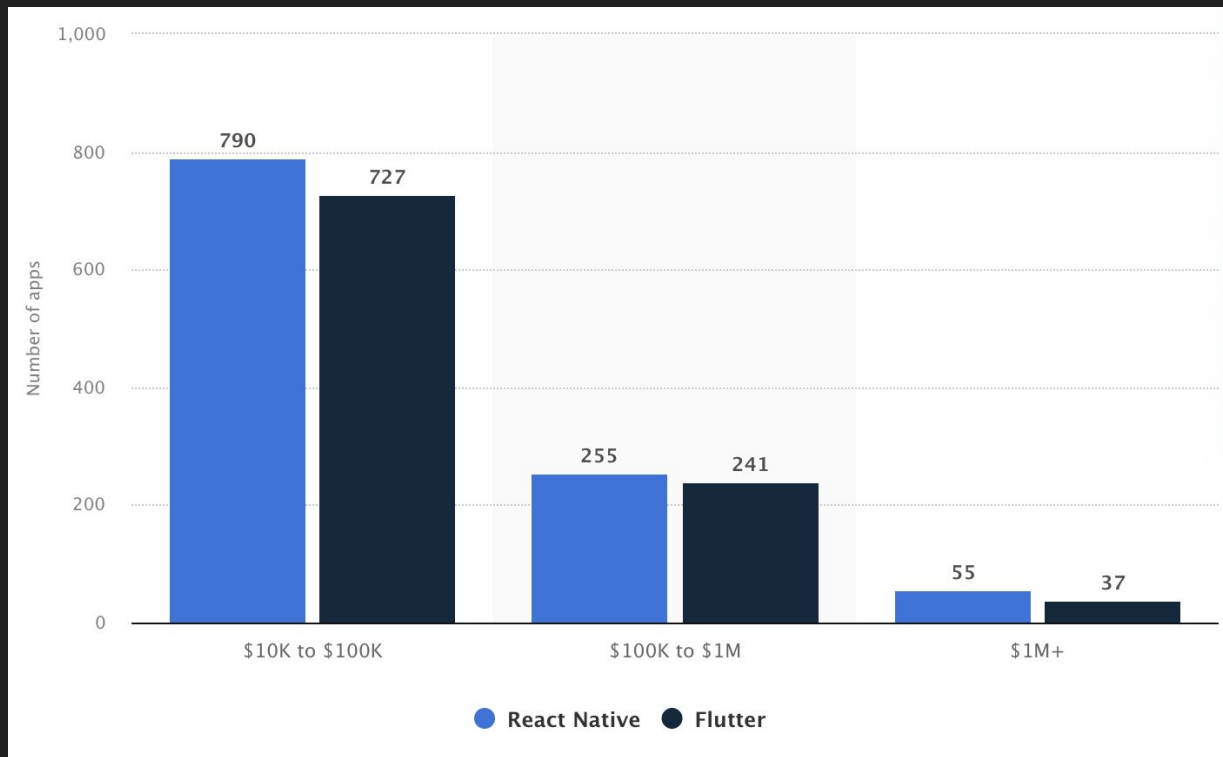
9%



6. Unity

8%

시장 플랫폼 점유율 현황



React-Native

React 와 React-Native 얼마나 유사한가?

```
import React from 'react';
import './App.css'; // CSS 파일 불러오기

function App() {
  return (
    <div className="container">
      <h1>Hello, World!</h1>
    </div>
  );
}

export default App;

/* App.css */
.container {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
  height: 100vh;
}
```

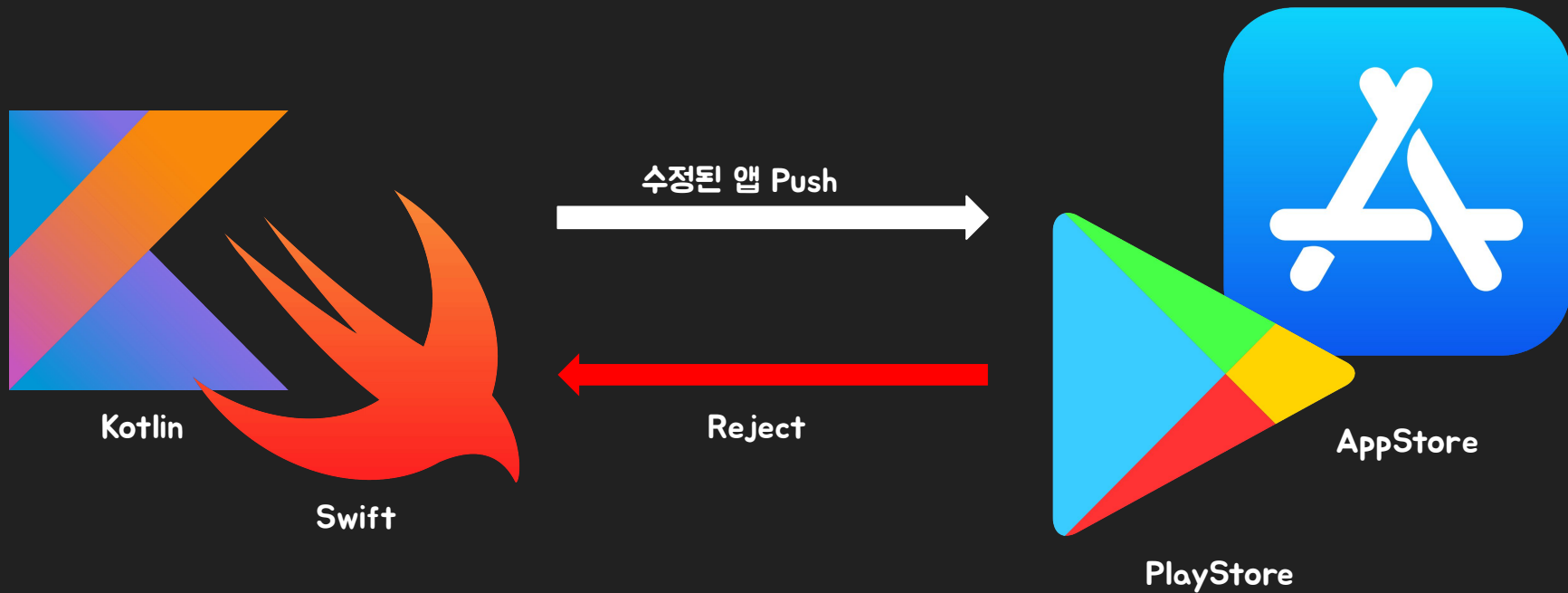
```
import React from 'react';
import { StyleSheet, View, Text } from 'react-native';

const App = () => {
  return (
    <View style={styles.container}>
      <Text>Hello, World!</Text>
    </View>
  );
};

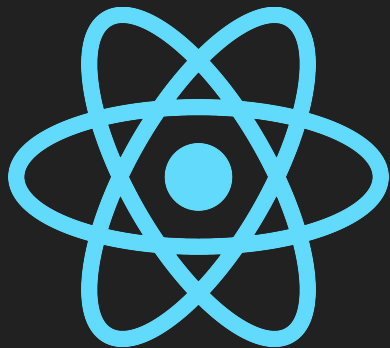
const styles = StyleSheet.create({
  container: {
    flex: 1,
    justifyContent: 'center',
    alignItems: 'center',
  },
});

export default App;
```

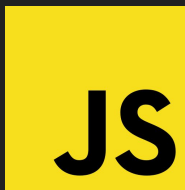
CodePush란?



CodePush란?



React-Native



JS 만 수정

PASS



PlayStore



AppStore

CodePush의 장점

- 배포 주기 단축 : 앱 스토어의 긴 심사 과정을 생략하고, 즉시 업데이트를 배포
- 즉각적인 버그 수정 : 심각한 버그가 발생했을 때, 사용자들이 앱을 재설치할 필요 없이 빠르게 패치를 적용
- 점진적 배포 : 모든 사용자에게 한꺼번에 배포하지 않고, 특정 사용자 그룹에 먼저 업데이트를 테스트
- 롤백 기능 : 업데이트에 예기치 않은 문제가 생기면, 즉시 이전 버전으로 되돌려 사용자 경험을 보호

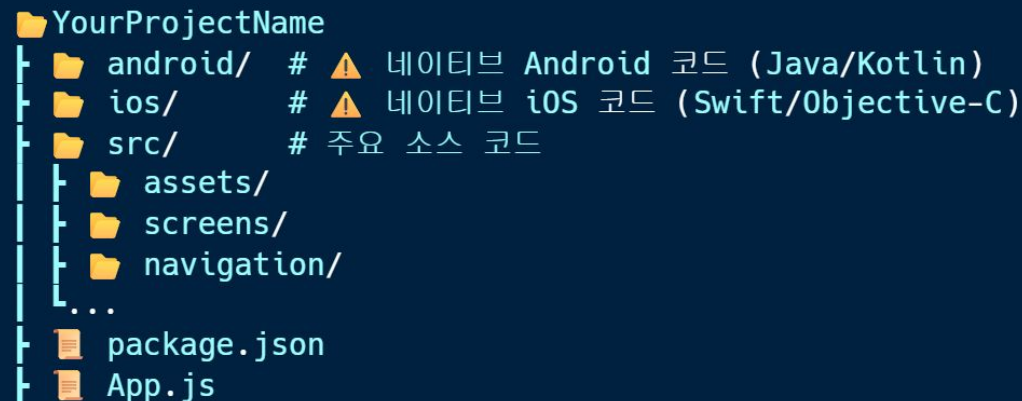
React-Native

CLI & Expo

React-Native CLI (Command Line Interface)



React-Native CLI (Command Line Interface)

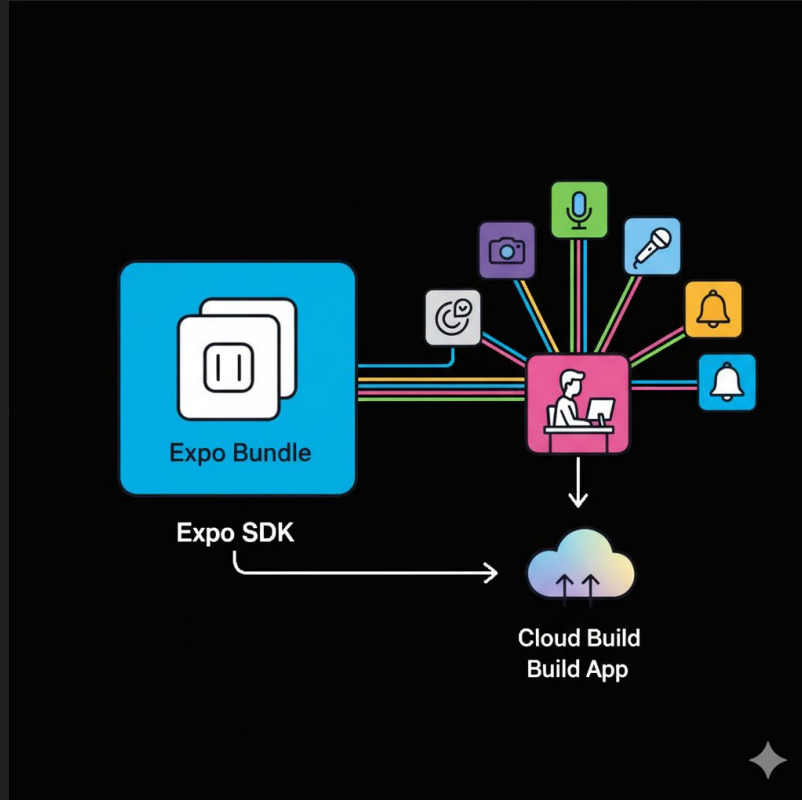


```

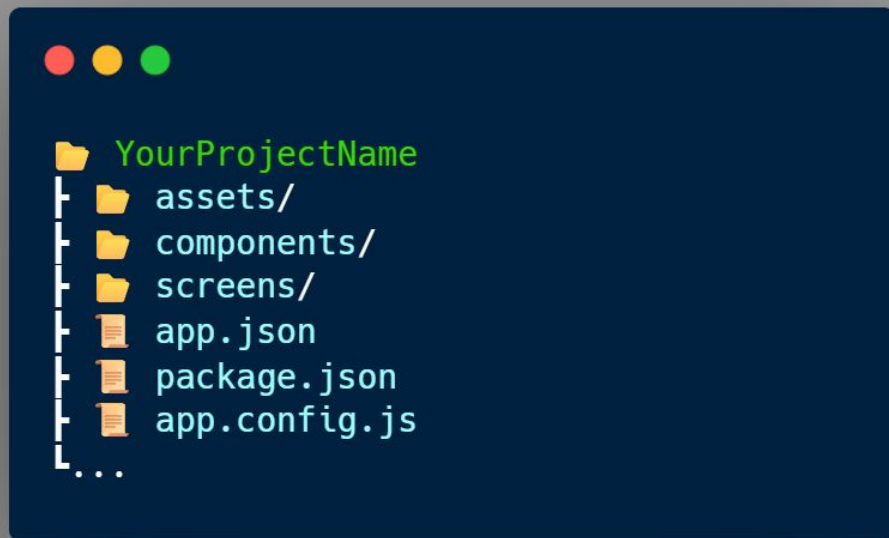
YourProjectName
├── android/ # ⚠ 네이티브 Android 코드 (Java/Kotlin)
├── ios/      # ⚠ 네이티브 iOS 코드 (Swift/Objective-C)
├── src/      # 주요 소스 코드
│   ├── assets/
│   ├── screens/
│   ├── navigation/
│   └── ...
├── package.json
└── App.js

```


React-Native Expo



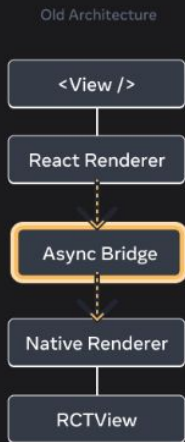
React-Native Expo



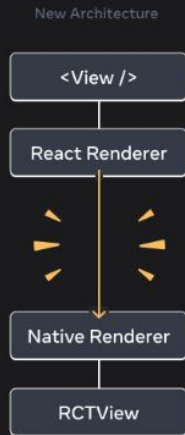
최신의 React-Native

Bridge 에서 JSI 으로

Bridge
비동기
JSON 파일 직렬화



In the old architecture, all communication between JS and Native was done through the async bridge, making all calls async.



In the new architecture, JavaScript and Native communicate synchronously, with direct bindings to C++ functions,.

JSI
동기
직접적 상호작용

Expo EAS

EAS Build : 로컬 컴퓨터가 아닌 클라우드 환경에서 iOS(.ipa)와 Android(.apk) 앱 빌드를 자동으로 처리, 복잡한 네이티브 개발 환경을 직접 구축하고 관리할 필요가 없어 시간과 노력을 크게 절약할 수 있다.

EAS Update : 앱 스토어 심사 없이 JavaScript 코드와 에셋(이미지, 폰트 등)을 사용자 기기에 실시간으로 업데이트 하고 이를 버그 수정이나 UI 변경을 즉시 배포하여 사용자에게 신속한 경험을 제공 -> **code push** 대체

EAS Submit : 빌드된 앱 바이너리(.ipa, .apk)를 Apple App Store와 Google Play Store에 자동으로 제출해주는 기능 앱 스토어 배포 과정을 자동화

Q&A

참고자료

toss https://www.youtube.com/watch?v=b_6CjuvVg8o&

SDK가 설치된 앱의 수 퍼센트 <https://makeitnew.io/cross-platform-mobile-development-trends-you-need-to-know-in-2025-a00ff6cc34f3>

매출 관련 자료 : <https://www.statista.com/statistics/1538085/reach-native-flutter-app-monthly-revenue/>