[React Native]

Facebook이 개발한 프레임워크로, JavaScript를 사용하여 iOS와 Android 애플리케이션을 개발할 수 있다.

--특징--

▶JavaScript 사용: 웹 개발자들이 이미 많이 사용하는 JavaScript를 기반으로 하고 있어 접근이 용이함.

▶React 기반: React의 컴포넌트 기반 아키텍처를 채택하여 재사용성 높은 UI를 개발할 수 있음.

▶네이티브 모듈 사용 가능: 필요할 때는 Java나 Swift 등의 네이티브 코드를 통해 성능을 최적화할 수 있음.

▶활발한 커뮤니티: 개발 커뮤니티가 크고 많은 오픈소스 라이브러리 및 도구들이 존재하여 빠르게 개발 가능.

--장점--

▶기존 웹 개발자에게 친숙: 웹 개발자들이 JavaScript와 React에 익숙하다면, React Native로 쉽게 전환할 수 있음.

▶빠른 개발 속도: 많은 오픈소스 라이브러리와 도구들이 있어 빠른 애플리케이션 개발이 가능.

▶핫 리로드(Hot Reload): 코드 수정 후 애플리케이션을 다시 빌드하지 않고도 즉시 결과를 확인할 수 있음.

▶크로스 플랫폼: 단일 코드베이스로 iOS와 Android 앱을 동시에 개발할 수 있음.

--단점--

▶성능 한계: Flutter에 비해 성능이 다소 떨어질 수 있으며, 복잡한 애니메이션 처리나 성능이 중요한 앱에서는 네이티브 코드가 필요할 수 있음.

▶네이티브 모듈 의존: 특정 기능에서는 네이티브 모듈이 필요해, 각 플랫폼별 코드를 따로 작성해야 하는 경우가 생길 수 있음.

▶UI 일관성 문제: 네이티브 컴포넌트를 사용하다 보니, 플랫폼마다 UI가 다르게 보일 수 있음.

[Flutter]

Flutter는 Google에서 개발한 오픈소스 UI 툴킷으로, Dart 언어를 사용하여 iOS와 Android 애플리케이션을 개발할 수 있다.

--특징--

▶Dart 언어 사용: Flutter는 Dart라는 언어를 사용하며, 이 언어는 학습 곡선이 있지만 최적화된 성능을 제공함.

▶위젯 기반: Flutter는 자체적으로 모든 위젯을 렌더링하여, 플랫폼 독립적인 일관된 UI를 제공.

▶우수한 성능: 네이티브 성능에 가까운 애플리케이션을 개발할 수 있음.

▶다양한 애니메이션: 복잡한 애니메이션과 그래픽을 쉽게 구현할 수 있음.

--장점--

▶일관된 UI/UX: Flutter는 모든 위젯을 자체적으로 렌더링하기 때문에, iOS와 Android에서 일관된 UI/UX를 제공.

▶우수한 성능: Flutter는 네이티브 성능에 근접한 성능을 제공하며, 특히 복잡한 UI나 애니메이션에서 뛰어난 성능을 발휘.

▶풍부한 위젯: 많은 기본 위젯과 UI 컴포넌트를 제공하여, 커스터마이징과 애니메이션 구현이 쉬움.

▶빠른 개발: Hot Reload 기능과 잘 구성된 문서 덕분에 개발 속도가 빠름.

--단점--

▶Dart 언어 학습 필요: Dart 언어는 JavaScript만큼 널리 사용되지 않아서 새로운 언어를 학습해야 함.

▶앱 크기: Flutter로 개발된 앱은 상대적으로 파일 크기가 큼.

▶네이티브 기능 제한: React Native처럼 특정 네이티브 기능을 사용할 때는 별도의 네이티브 코드 작업이 필요할 수 있음.

React Native와 Flutter 비교 분석

|  |  |  |
| --- | --- | --- |
| **항목** | **React Native** | **Flutter** |
| **언어** | JavaScript | Dart |
| **UI** | 네이티브 컴포넌트를 사용해 플랫폼 별로 UI가 다름 | 자체 위젯을 사용해 일관된 UI 제공 |
| **성능** | 성능이 약간 떨어질 수 있음 | 네이티브에 가까운 성능 제공 |
| **생태계** | 커뮤니티와 라이브러리가 풍부 | 구글의 적극적 지원과 확장성 있는 위젯 제공 |
| **개발 속도** | 빠름 | 빠름 |
| **애니메이션** | 복잡한 애니메이션 구현이 다소 복잡 | 복잡한 애니메이션도 쉽게 구현 가능 |
| **코드 재사용성** | 높은 수준의 재사용성 | 높은 수준의 재사용성 |