

development for eve

ts you create truly native apps and doesn't compromise on performance. It provides a core set of platform agnostic native components that map directly to the platform's native UI building blocks.



React Native를 활용한 모바일 앱 개발

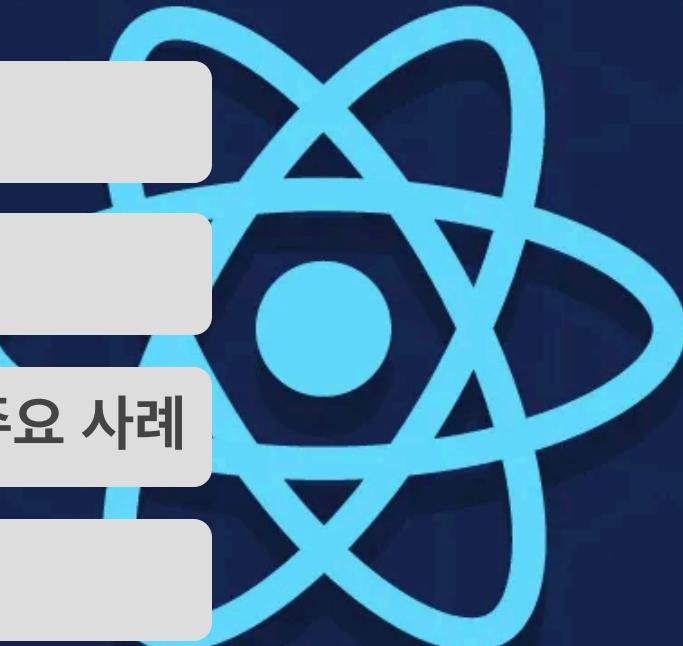
React Native를 통한 웹 vs. 앱 개발 이해

권수정
프롬인사이트 주식회사

2025. 11. 15

List

- 1 웹 vs. 앱
- 2 앱 개발이란?
- 3 앱개발(React Native) 주요 사례
- 4 실습
- 5 React Native 환경구성
- 6 Q&A



React Native



" WEB vs. APP "

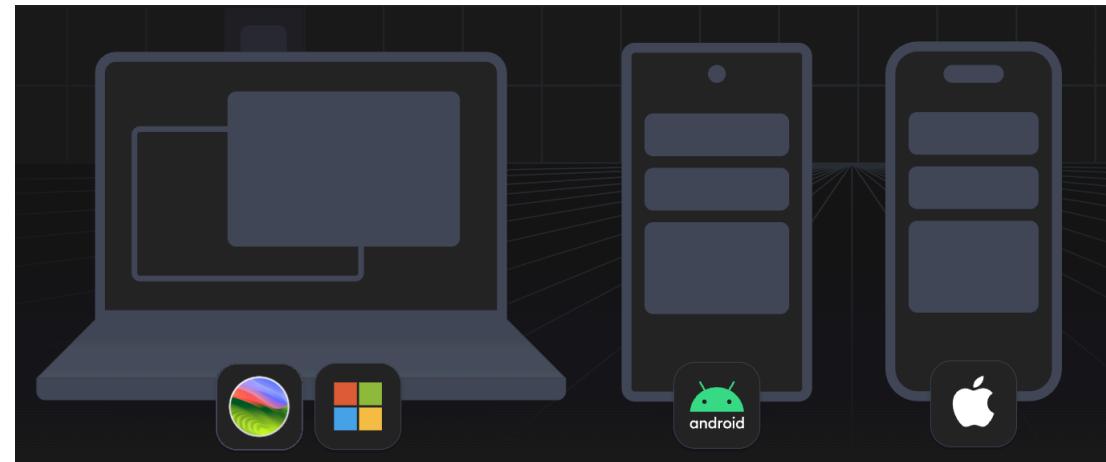
구분	웹	앱
플랫폼	브라우저에서 실행	네이티브 플랫폼에서 실행
성능	반응형 디자인, 제한적 성능	네이티브 성능(카메라, GPS 등) 최대한 활용 가능
업데이트	즉각 배포 가능	스토어 통해 업데이트
접근성	모든 기기에서 접근 가능	플랫폼 별로 설치

"웹 개발 vs. 앱 개발"

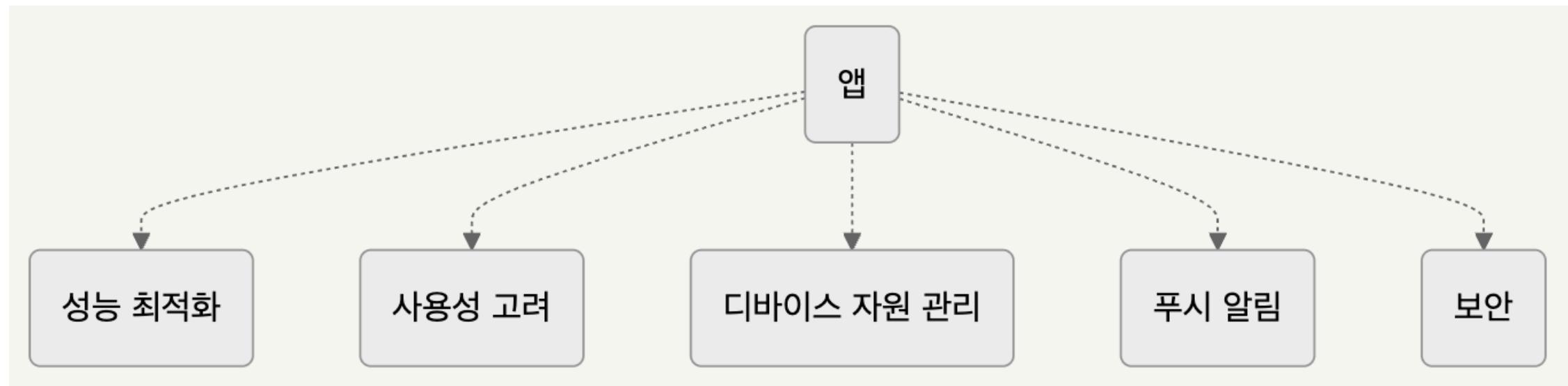
항목	웹 개발	앱 개발
UI	HTML, CSS, JavaScript 사용	네이티브 UI 컴포넌트 + 웹 기술
백엔드 로직	API 및 데이터베이스 공유 가능	동일 로직, 다른 플랫폼 적용
테스트	단위 및 통합 테스트 적용 가능	양쪽에 적용 가능한 테스트 프레임워크

앱 개발 언어 & 플랫폼

- 웹 앱
 - 모든 기기에서 다양한 웹 플랫폼 페이지를 제공하도록 디자인된 애플리케이션
- 네이티브 앱
 - 특정 플랫폼이나 기기를 위해 특별히 디자인된 애플리케이션
- 하이브리드 앱
 - 네이티브 및 웹 애플리케이션 조합
 - 단일 코드 베이스에서 모든 플랫폼에 대해 개발 가능



앱 개발(React Native) 주요 사례



1. 성능 최적화: Meta

웹 기반 vs 네이티브 앱 전환 이유

- 스크롤링 성능 문제
- 네트워크 연결 속도에 따른 렌더링 지연

앱 개발 시 고려사항

- 메모리 관리 최적화
- CPU, GPU 사용 최적화
- 네트워크 호출의 비동기 처리
- 캐싱 전략 수립

Quest while keeping a tremendously high bar for quality. The teams first had to decide how to build them: reusing the existing Android apps, writing a new native Android app, or using React Native to build from scratch. We wanted to offer a hero experience that looked and felt at home on Meta Quest, taking advantage of the additional input types, gestures, and larger visual surface area. Instead of simply porting our mobile social apps, we chose React Native as it enabled our teams to iterate and build quickly with robust animation capabilities, great performance, and a shared platform that powers most of the 2D Meta Quest system apps.

Engineering at Meta: <https://engineering.fb.com/2024/10/02/android/react-at-meta-connect-2024/>

2. 사용성 고려: Google Maps

앱 개발 시 고려사항

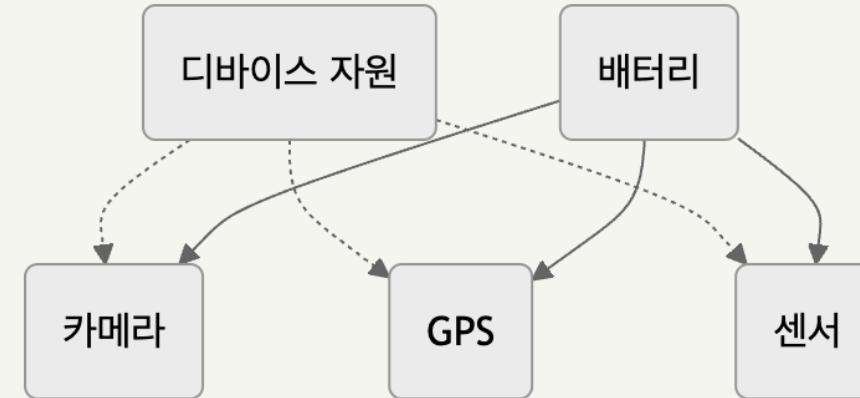
- 사용자 위치 기반
- 데이터 시각화
- 경로, 네비게이션
- 오프라인 지도 저장

구분	웹 개발	앱 개발
데이터 로딩	실시간	오프라인 저장 지원
네트워크 의존성	높음 (오프라인 모드 지원)	낮음
저장소 활용	제한적	로컬 저장소 적극 활용
데이터 동기화	실시간	백그라운드 동기화 지원

3. 디바이스 자원 관리: messenger

주요 자원 관리 항목

- 카메라 접근 최적화
- GPS 사용 관리
- 센서 데이터 처리
- 배터리 소모 최적화



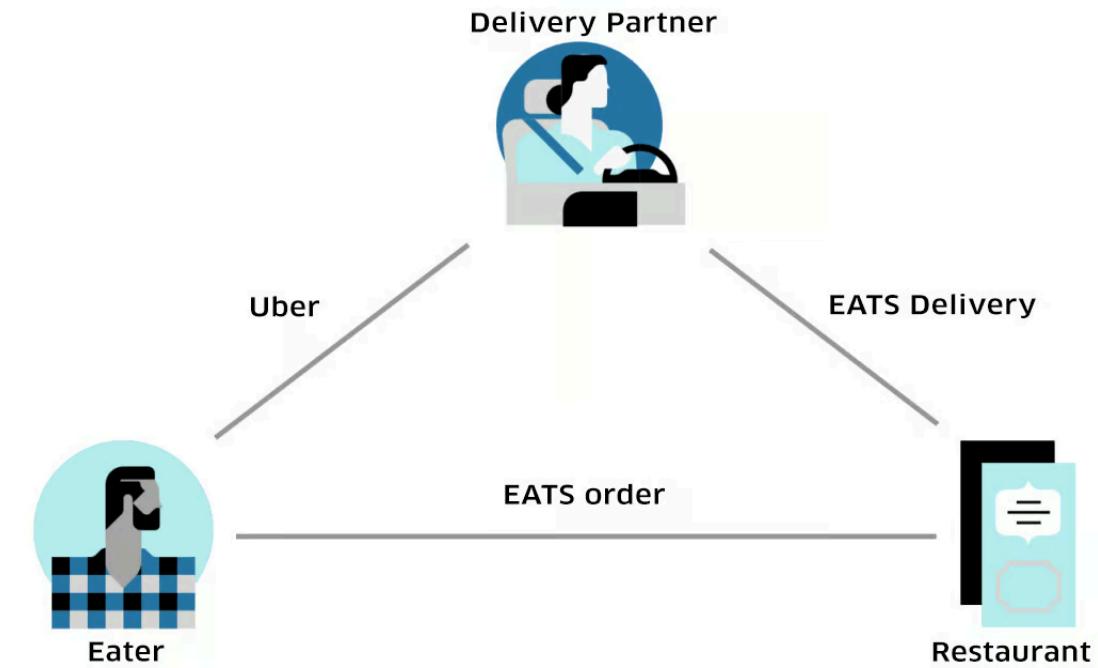
최적화 전략

- 필요 시에만 자원 활성화
- 백그라운드 작업 최소화
- 배터리 사용량 모니터링

4. 푸시 알림: Uber eats

알림 시스템 비교

특징	웹 알림	앱 푸시 알림
도달률	낮음	높음
사용자 응답률	중간	높음
커스터마이징	제한적	다양한 옵션
오프라인 지원	불가능	가능



source: <https://www.uber.com/en-KR/blog/ubereats-react-native/>

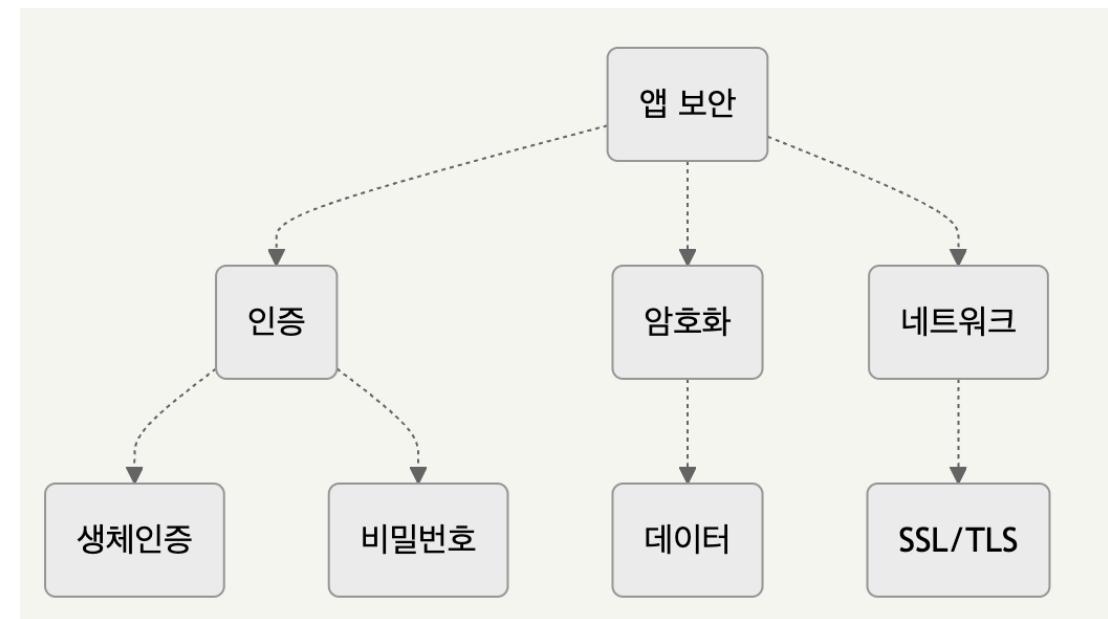
5. 보안: Dave (Digital Bank)

앱 보안 핵심 요소

- 생체 인증 통합
- 데이터 암호화
- 안전한 네트워크 통신
- OS 레벨 보안 기능

보안 강화 전략

- 민감 정보 암호화
- 강력한 사용자 인증
- 완벽해결 X: 보안 취약점 정기 점검



<https://reactnative.dev/docs/security>

웹뷰 전환 사례: Instagram

첫 테스트: 간단한 푸시 알림 설정

- Android 메소드 수 관리
(약 3500개 메소드 추가)

개발 속도 향상

- 코드 공유를 통한 빠른 배포
(iOS/Android 간 85-99% 코드 재사용)
- Live/Hot Reloading으로 빠른 반복 개발
- 사용자 경험: 웹뷰 대비 향상된 사용자 경험
- 성능: 웹뷰 대비 시작 시간 개선

기능	코드 공유	특징
Post Promote	99%	웹뷰 → RN 전환
SMS Captcha	97%	보안 체크포인트
Comment Moderation	85%	커뮤니티 보호
Push Notifications	92%	첫 테스트 케이스

source: <https://instagram-engineering.com/react-native-at-instagram-dd828a9a90c7>

앱 개발을 위한 다양한 언어

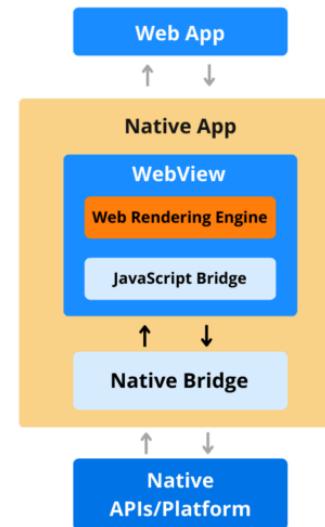
- **Swift or Objective-C (iOS)**
 - Apple에서 개발한 공식 언어
 - iOS 앱 개발 언어
 - **Kotlin or Java (Android)**
 - Android 앱 개발에 주로 사용
 - Java보다 간결한 문법
 - **Flutter**
 - Google 개발 Dart 기반
(성능, UI 커스터마이징 강점)
 - 하나의 코드로 웹, 네이티브 앱 개발 가능
 - **React Native**
 - Meta 개발 React 기반
- 다양한 **크로스플랫폼** 개발 언어 : 웹 기술 렌더링 방식
 - Cordova: apache
 - Ionic: 웹 기술(HTML, CSS, JS)로 고성능 앱 개발
 - Titanium: JavaScript 기반 네이티브 앱
 - NativeScript: Angular, Vue.js 등으로 네이티브 앱 개발
 - Xamarin: .NET, C#으로 iOS, Android, Windows 앱 개발

React Native ?

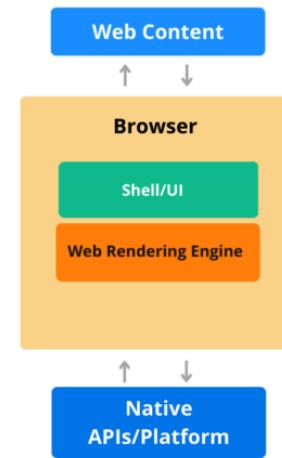
"Learn Once, Write Anywhere"

- 오픈소스 "하이브리드" 프레임워크
- React와 유사한 방식으로 컴포넌트를 사용한 UI 구축
- 네이티브 기능(카메라, GPS 등) 활용 가능
- 웹과 유사한 컴포넌트 기반 구조 (React와 유사)
- 플랫폼 독립적
- 주요 차이점은 웹뷰 대신 네이티브 뷰를 사용
- iOS와 Android 간 코드 재사용으로 개발 효율성 높음

Native App with a WebView

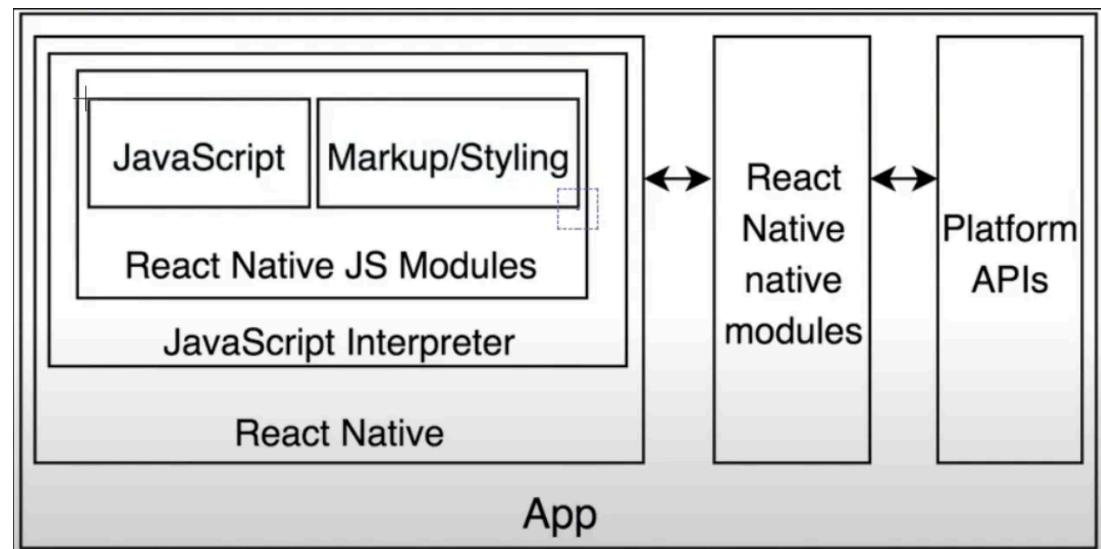


Web App



React Native 공식문서: <https://reactnative.dev/>
cite image: <https://www.scalablepath.com/react-native/react-native-vs-native>

QnA. 윈도우 혹은 리눅스에서 iOS app 개발 !?



- 모바일 웹을 만드는게 아님!!
- 앱을 만드는 것!
→
 앱에 있는 것들을 다 설치해야 함

React Native 앱 개발환경

- iOS
 - Xcode 설치
 - CocoaPods 설치: `sudo gem install cocoapods`
- Android
 - Android Studio 설치
 - JAVA_HOME, ANDROID_HOME 환경 변수 설정

• 필수 도구 설치

```
# 1. Node.js 설치  
node -v  
  
# 2. Watchman 설치  
npm install watchman  
  
# 3. react-native cli 설치  
npm install -g react-native-cli  
...  
# 시뮬레이터 설치...설치... 설치....
```

React Native 테스트 환경 구성하기

- SW 없이 테스트, VSCode로 모바일 테스트 환경
- 테스트 용으로 일단 **node -v, npm** 만 있으면 됨!!!
- 라우팅, 네이티브 모듈의 표준 라이브러리 등 앱을 더 쉽게 개발할 수 있는 개발 도구

- **expo.dev** (*)

```
npm install --global expo-cli  
npx create-expo-app@latest ozapp --template blank
```

- **watchman**

- 폰에서 앱다운로드 안드로이드(Expo), 애플(Expo go)

[(*)]<https://docs.expo.dev/get-started/set-up-your-environment/>

QnA. 어떤 언어를 선택할까?

- <https://memory2.co/blog/cross-platform-app-development-trends-and-frameworks>
- <https://technostacks.com/blog/mobile-app-development-frameworks/>

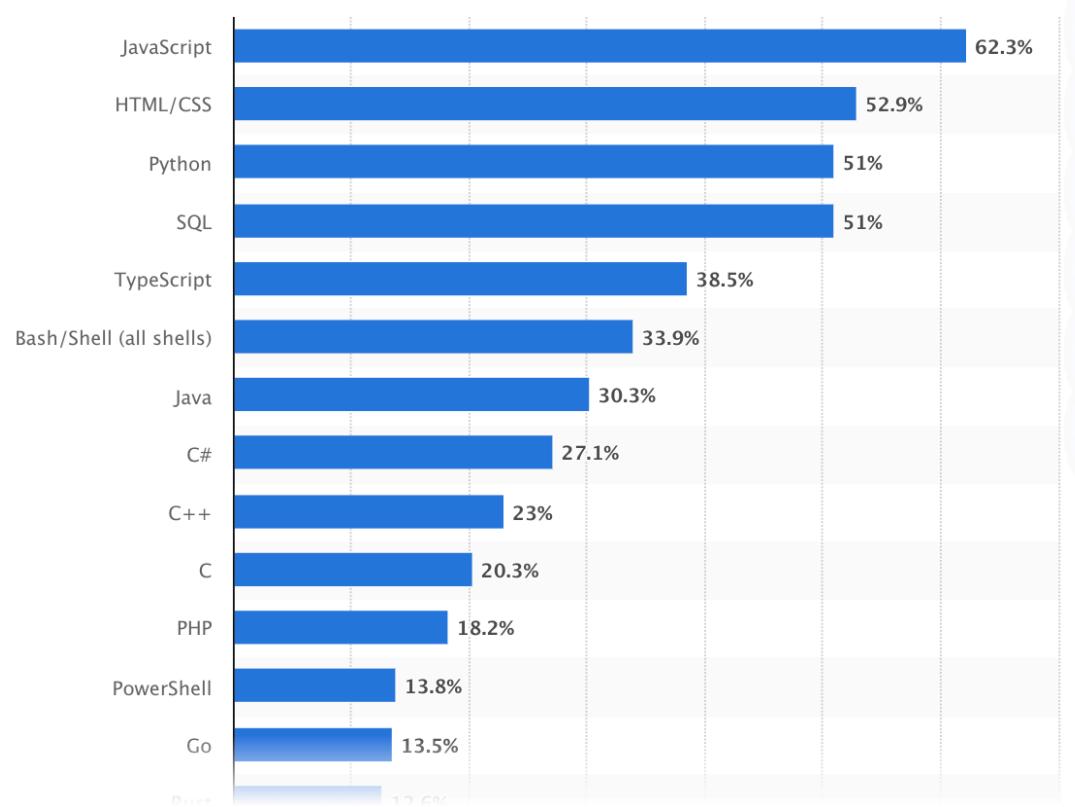


Figure: Most used programming languages among developers worldwide as of 2024

레퍼런스

1. 공식 문서

- [React Native](#)

2. 커뮤니티

- [React Native Community](#)
- Stack Overflow 태그: react-native

3. 인용 사이트

- [Differences Between Web and App Development](#)
- [Cross-Platform App Development Trends & Frameworks 2024](#)

4. 도서