

# **React Native 앱 개발의 모든 것**

**4가지 프로젝트로 마스터 하는 웹뷰부터 앱 개발까지**



# 강의의 목적과 전체 구성 개요

강사 소개

# 김대훈

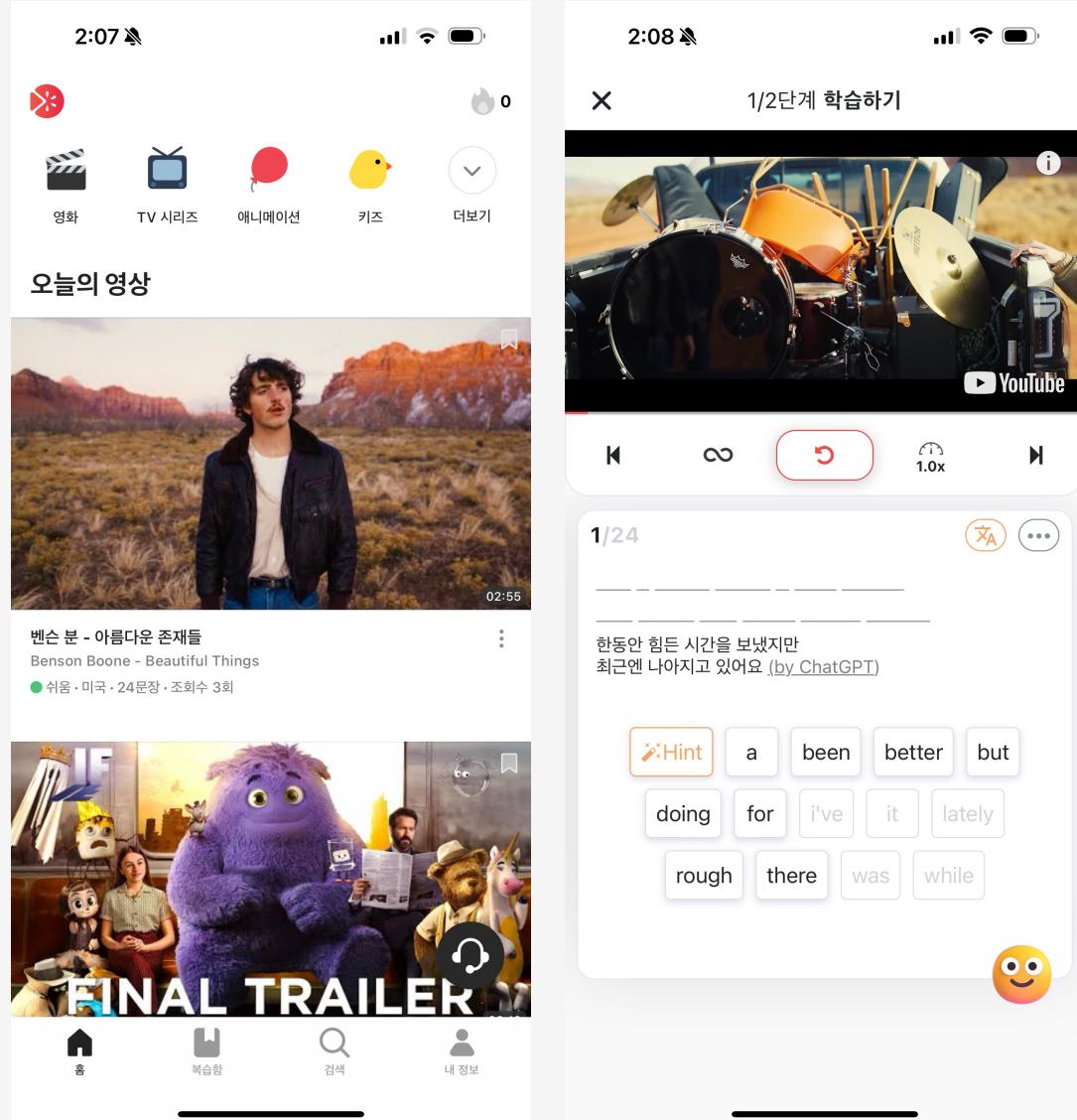
하얀마인드 CTO & Co-founder (2017~)

글로벌 200만+ 다운로드 영어 학습 앱 레드카위 개발 및 운영



# RedKiwi

- React Native를 이용해서 개발 및 운영  
(2017 ~)
- WebView를 이용한 비디오 플레이어 개발
- 글로벌 200만+ 다운로드



## 강의 목표

01

웹뷰의 정의와 작동 원리를 이해할 수 있다

02

React Native 웹뷰를 활용하여 하이브리드 앱을 개발할 수 있다

03

하이브리드 앱을 마켓에 출시하고 업데이트 할 수 있다

Part 1

## React Native를 이용한 기초 웹뷰 개발

- 웹뷰 기초 이론

- React Native 프로젝트 셋업 (React Native CLI / Expo)

- 기초 웹뷰 개발 (React Native CLI / Expo)

Part 2

## 웹뷰 띄우기 학습

- 네이버 앱 클론 코딩 프로젝트

- 네이버 앱 클론 코딩 (React Native CLI / Expo)

- 웹뷰 브라우징

- 앱 네비게이션

Part 3

## 웹뷰와 RN 앱 통신을 학습

- 유튜브 SDK 활용 학습 앱 프로젝트

- 유튜브 영상 구간 반복 학습기 (React Native CLI / Expo)

- 웹뷰와 앱의 통신

Part 4

## 앱 리소스 사용 학습

- 챗GPT API 활용 클로바ST 녹음요약 앱 프로젝트

- 내용 요약해주는 AI 녹음기 (React Native CLI / Expo)

- 웹뷰로 앱 리소스 사용하기

- AI API 이용하기

Part 5

## 웹뷰 앱 배포부터 업데이트까지 모든 것

- 웹뷰로 만든 앱 출시하기

- 웹뷰로 만든 앱 업데이트 하기 (웹/앱)

- 푸시 알림 전송하기

# 학습 준비물

## 사전 지식

- React
- JavaScript
- (TypeScript)

## 실습 환경

- Mac OS
- (윈도우는 프로젝트 셋업 가이드 제공)

# 웹뷰의 등장 배경 및 개념

## 학습 목표

01

네이티브 앱, 웹 앱, 하이브리드 앱을 이해하고 차이를 알 수 있다

02

WebView의 역할에 대해 이해한다

## 앱의 종류

01

네이티브 앱

02

웹 앱

03

하이브리드 앱

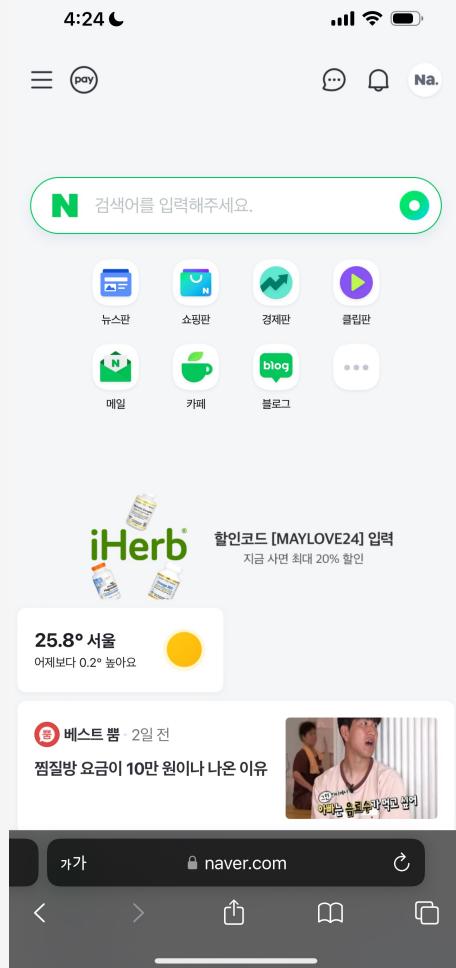
# 네이티브 앱

- 네이티브 앱은 특정 플랫폼(예: iOS, Android)을 위해 최적화되어 개발된 애플리케이션
- 플랫폼의 네이티브 프로그래밍 언어(Swift, Objective-C, Kotlin, Java 등)로 작성
- 장점
  - **성능**: 최적화된 코드로 인해 빠른 성능을 제공
  - **기능 접근성**: 카메라, GPS 등 기기의 하드웨어 기능에 완벽하게 접근 가능
- 단점
  - **개발 비용**: 각 플랫폼마다 별도의 앱을 개발해야 하므로 비용과 시간이 많이 소요됨
  - **유지 관리**: 여러 플랫폼 버전을 유지 관리 해야 함
  - **앱 스토어 심사**: 각 플랫폼의 앱 스토어 정책에 따라 앱 심사 과정을 거쳐야 하며, 이는 시간이 소요되고 때로는 승인이 거부될 수도 있음



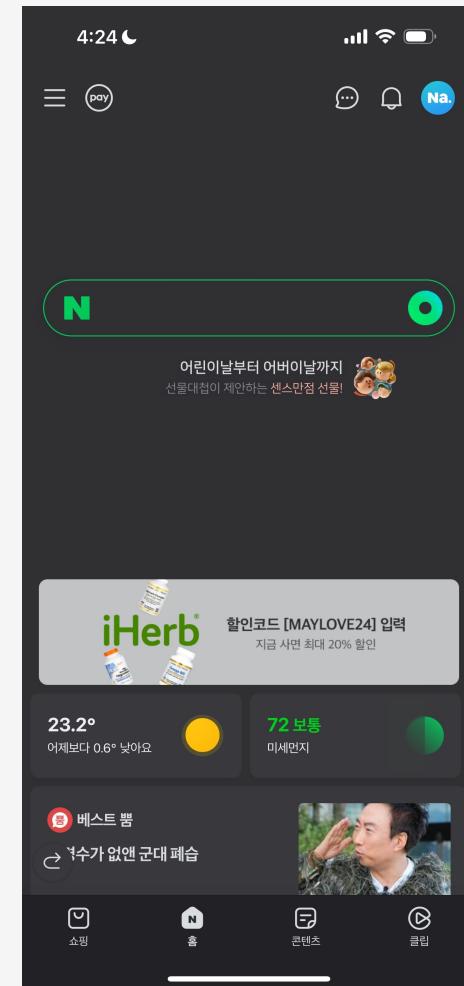
# 웹 앱

- 웹 기술(예: HTML, CSS, JavaScript)을 사용하여 개발
- 별도의 앱 다운로드 없이 모바일 브라우저를 통해 접근
- 장점
  - **플랫폼 독립성**: 어떤 운영 체제에서도 실행할 수 있음
  - **개발 및 유지 관리**: 하나의 코드베이스로 여러 플랫폼에서 운영될 수 있어 개발 및 유지 관리가 용이
  - **앱 스토어 심사 불필요**: 앱 스토어를 통하지 않고 웹 브라우저를 통해 직접 접근할 수 있어, 복잡한 앱 스토어 심사 과정이 필요 없음
- 단점
  - **성능 제한**: 네이티브 앱보다 느릴 수 있으며, 기기의 모든 기능을 사용할 수 없음
  - **접근성 제한**: 웹 앱에 접근하려면 URL을 입력해야 하므로, 사전에 다운로드 받아 바로 실행할 수 있는 네이티브 앱에 비해 접근성이 떨어짐



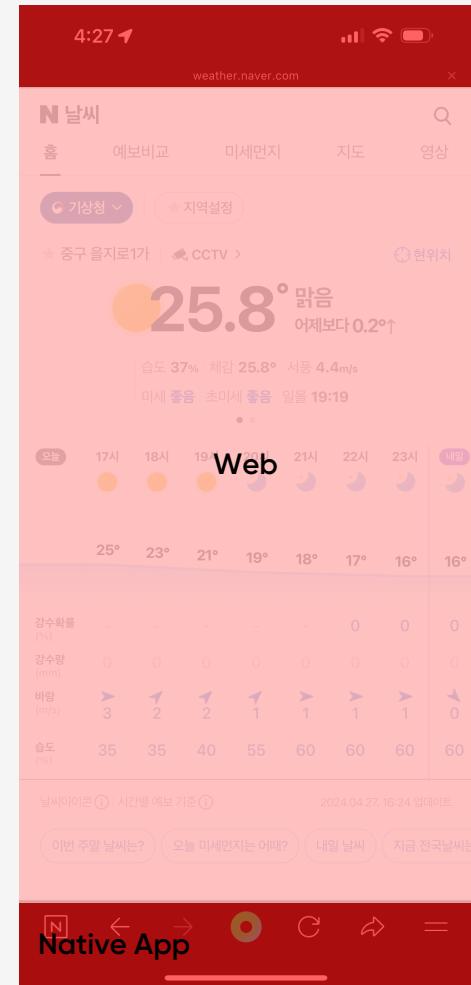
# 하이브리드 앱

- 네이티브 앱과 웹 앱의 특성을 결합한 앱
- 웹 기술로 개발하여, 네이티브 앱의 웹뷰를 통해 실행
- 장점
  - **개발 효율성**: 하나의 코드베이스로 여러 플랫폼에서 작동되며, 개발 시간과 비용을 절약할 수 있음. 웹 기술을 활용하므로 웹 개발자도 쉽게 접근 가능
  - **기능 통합**: 네이티브 앱의 기능적 이점을 모두 활용할 수 있음. (예: 카메라, GPS 등)
  - **신속한 업데이트**: 앱 스토어를 통하지 않고도 앱의 일부 웹 콘텐츠를 직접 업데이트할 수 있어, 사용자에게 신속한 변경과 개선을 제공할 수 있음
- 단점
  - **성능 제한**: 웹뷰를 통한 웹 콘텐츠 렌더링은 순수 네이티브 앱에 비해 성능이 떨어질 수 있음. (예: 복잡한 애니케이션 등)



## 웹뷰

- 웹뷰는 네이티브 애플리케이션 내에서 웹 페이지를 표시하기 위해 사용되는 컴포넌트
- 웹뷰는 기본적으로 내장된 브라우저 엔진(예: Android의 Chromium, iOS의 WebKit)을 사용하여 웹 콘텐츠를 로드하고 표시
- 웹뷰는 앱 내에서 웹 콘텐츠를 렌더링하지만, 단순한 콘텐츠 표시 기능을 넘어서 앱의 네이티브 기능과 웹 콘텐츠 간의 데이터 교환 및 상호 작용이 가능



## 학습 요약



네이티브 앱: 플랫폼에 최적화된 고성능 앱, 하지만 높은 개발 리소스와 앱 스토어 심사가 필요



웹 앱: 플랫폼 독립적이고 개발 효율성, 성능 제한과 불편한 접근성



하이브리드 앱: 웹 앱의 개발 효율성과 네이티브 앱의 기능적 이점을 결합, 성능에는 일정한 제한



웹뷰: 앱 내에서 웹 콘텐츠를 표시하고, 네이티브 앱의 기능과 상호작용 가능

**React Native CLI / Expo를 이용한 기초 웹뷰 개발**

**학습 목표 및 구성**

# 학습 구성



## 학습 목표

01

React Native의 기본 개념을 이해할 수 있다

02

React Native CLI와 TypeScript를 활용하여 React Native 프로젝트를 초기화할 수 있다

03

Expo와 TypeScript를 활용하여 React Native 프로젝트를 초기화할 수 있다

04

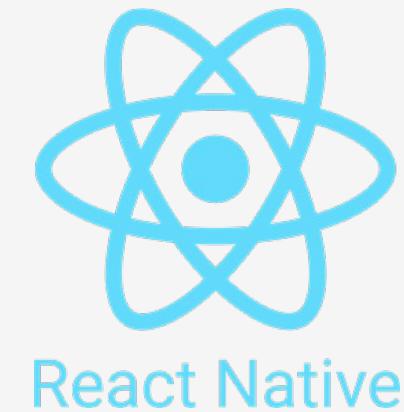
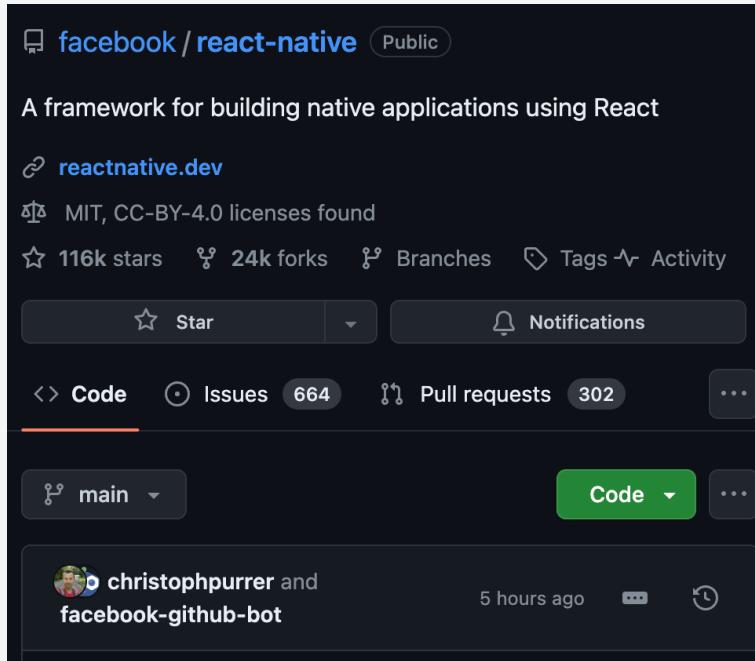
웹뷰 라이브러리를 설치하고 웹 콘텐츠를 표시할 수 있다

**React Native CLI / Expo를 이용한 기초 웹뷰 개발**

# **React Native란 무엇인가?**

# React Native

- React Native는 페이스북이 만든 오픈 소스 모바일 애플리케이션 프레임워크
- React와 JavaScript를 이용해서 네이티브 앱 개발



# 왜 React Native가 인기가 많을까요?

## 높은 개발 생산성

### 기존 네이티브 앱 개발

iOS – Objective-C, Swift

Android – Java, Kotlin

플랫폼에 맞는 언어로 각각 개발이 필요함

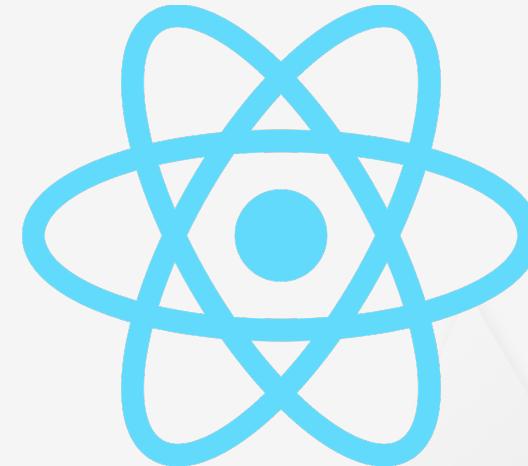
### React Native

iOS, Android – JavaScript

하나의 언어로 동시에 개발이 가능함

## 활용이 높은 언어

- 전체 웹 사이트의 약 98%가 JavaScript를 클라이언트 프로그래밍 언어로 사용<sup>[1]</sup>
- JavaScript는 클라이언트 뿐만 아니라, Node.js로 서버 프로그래밍도 가능하게 함
- 웹 기술로 높은 인기를 보이는 React<sup>[2]</sup>
- 많은 개발자가 쉽게 접근 할 수 있고, 팀내에 같은 개발 언어를 공유할 수 있음



[1] <https://www.simplilearn.com/best-programming-languages-start-learning-today-article>

[2] <https://survey.stackoverflow.co/2023/>

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

const App = () => (
  <View>
    <Text>Hello React Native!</Text>
  </View>
);

export default App;
```

## React Native 코드 예시

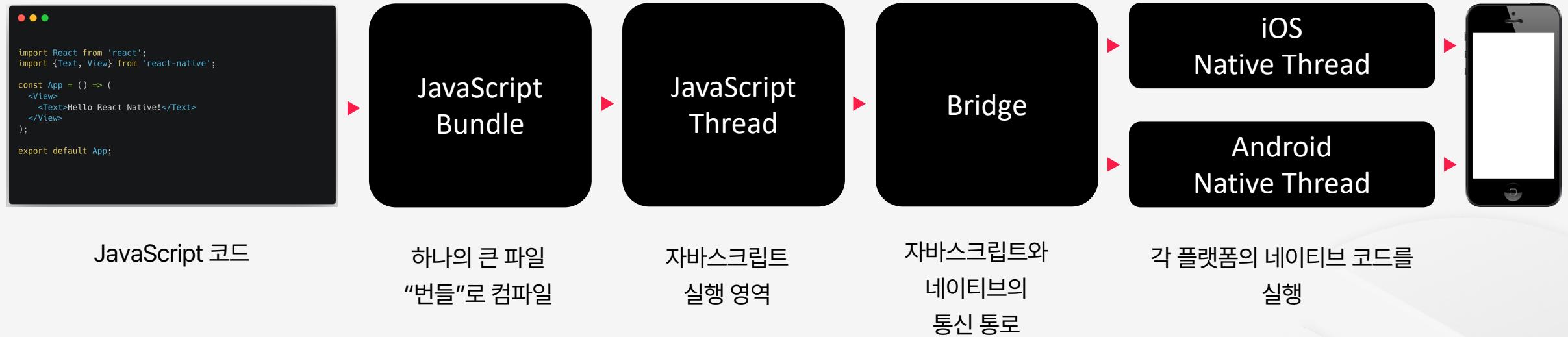
React를 사용해봤다면 익숙한 코드!

## 네이티브 앱 개발

- React Native 애플리케이션은 네이티브 코드와 직접적으로 상호작용
- 빠른 성능 (순수 네이티브 앱 보다는 느릴 수 있음)
- 각 플랫폼 네이티브 UI 컴포넌트를 이용하여 제공 가능



같은 컴포넌트를 사용했지만 플랫폼에 따라 다른 UI가 나타남



## React Native 동작 방식

Bridge 때문에 구조적으로 네이티브 앱 보다 성능 저하가 존재!

# 하이브리드 앱을 위한 React Native

# 하이브리드 앱을 위한 React Native

## 기존 네이티브 앱 개발

- 프론트엔드 개발자가 각 플랫폼 언어 학습 필요
- 웹뷰를 위한 iOS 앱, Android 앱 별도 제작 및 관리 필요

## React Native

- React를 기반으로, 적은 학습으로 프론트엔드 개발자가 앱 개발 가능
- 안드로이드 앱, iOS 앱 모두 제작 및 관리 가능

**React Native CLI를 이용한 기초 웹뷰 개발**

**앱 개발 환경 셋업**

**(React Native CLI)**

## 학습 목표

01

React Native CLI가 무엇인지 이해한다

02

React Native CLI / TypeScript를 이용해서 React Native 프로젝트를 초기화 할 수 있다

03

프로젝트를 빌드하여 iOS 앱을 실행 할 수 있다

04

프로젝트를 빌드하여 Android 앱을 실행 할 수 있다

# React Native CLI

- React Native 프로젝트를 생성, 관리, 빌드하고 디버깅하는 데 사용되는 표준 도구
- 장점
  - 자유도가 높아서 React Native를 100% 사용 가능
  - **네이티브 코드 접근**: 복잡한 네이티브 기능이나 타사 네이티브 모듈을 쉽게 통합 가능
  - **유연성**: 다양한 라이브러리 사용 가능
  - **커스터마이징**: 앱의 빌드 설정, 종속성 등을 자세히 조정 가능
- 단점
  - **초기 세팅 시간**: 프로젝트를 시작하는 데 시간이 더 오래 걸림 (Xcode, Android Studio 등 설치)
  - **설정 복잡성**: 모든 라이브러리를 직접 설치하고 설정해야 함
- 참고 사이트
  - <https://reactnative.dev/docs/0.73/getting-started>
  - <https://reactnative.dev/docs/0.73/environment-setup>

# Visual Studio Code (VSCode)

- Microsoft에서 개발한 경량화되면서도 강력한 소스 코드 편집기
- **다양한 언어 지원:** JavaScript, Python, Java, C# 등과 같은 다수의 프로그래밍 언어를 지원
- **확장성:** 풍부한 확장 프로그램 사용 가능 (ESLint, Prettier, Copilot...)
- <https://code.visualstudio.com/>

# macOS - iOS

- iOS 개발을 위해서 맥 필요
- Node
  - React Native의 JavaScript 런타임 환경
  - Homebrew를 이용해서 설치
    - Homebrew: macOS 패키지 매니저
- Watchman
  - Facebook에서 개발한 파일 시스템 모니터링 도구. React Native에서 파일이 변경되면 자동으로 변경 사항 반영
- Xcode
  - Apple에서 개발하고 배포하는 통합 개발 환경
  - Command Line Tools: 개발에 필요한 유ти리티 제공. React Native 애플리케이션을 iOS 시뮬레이터에서 실행하거나, iOS 기기에 빌드하기 위해 필요한 명령어 제공
    - `xcode-select --install`
  - Simulator 설치
  - CocoaPods: macOS와 iOS 프로젝트를 위한 의존성 관리자로, Swift 및 Objective-C의 라이브러리 관리
  - Ruby: Cocoapods 설치를 위해 필요 (<https://github.com/rbenv/rbenv>)
  - Bundler: Ruby 패키지 관리자 설치 (`gem install bundler`)

# macOS - Android

- Node
  - React Native 개발에서 서버 측 스크립트 실행 및 다양한 도구와 라이브러리의 관리를 위한 JavaScript 런타임 환경
  - Homebrew를 이용해서 설치
    - Homebrew: macOS 패키지 매니저
- Watchman
  - Facebook에서 개발한 파일 시스템 모니터링 도구. React Native에서 파일이 변경되면 자동으로 변경 사항 반영
- Java Development Kit
  - Java 애플리케이션을 개발하고 실행하는 데 필요한 컴파일러, Java 런타임 환경(JRE), 라이브러리 및 다양한 유틸리티 도구들을 포함
  - JAVA\_HOME=/Library/Java/JavaVirtualMachines/zulu-17.jdk/Contents/Home  
PATH=\$PATH:\$JAVA\_HOME/bin  
export JAVA\_HOME  
export PATH
- Android Studio
  - Google이 제공하는 공식 통합 개발 환경(IDE)
  - Android SDK 설치 및 관리
  - 내장된 Android 에뮬레이터를 사용하여 가상의 Android 디바이스에서 애플리케이션을 테스트하고 디버깅할 수 있음

# 프로젝트 초기화

- **Yarn 설치**
  - Yarn: JavaScript 패키지 관리자. React Native에서 라이브러리 설치시 사용
  - `npm install --global yarn`
- **프로젝트 초기화**
  - `npx react-native@X.XX.X init AwesomeProject --version X.XX.X`
- **매트로 서버 실행하기**
  - React Native 앱에 필요한 모든 자바스크립트 코드와 에셋을 실시간으로 번들링하고, 변경 사항을 빠르게 앱에 적용
  - `yarn start`
- **빌드**
  - `yarn ios`
  - `yarn android`

# Windows - Android

- Node
  - React Native 개발에서 서버 측 스크립트 실행 및 다양한 도구와 라이브러리의 관리를 위한 JavaScript 런타임 환경
  - Chocolatey를 이용해서 설치
    - Chocolatey: 윈도우 패키지 매니저
    - choco install -y nodejs --version=18.20.2
- Java Development Kit
  - Java 애플리케이션을 개발하고 실행하는 데 필요한 컴파일러, Java 런타임 환경(JRE), 라이브러리 및 다양한 유ти리티 도구들을 포함
  - choco install -y microsoft-openjdk17
- Android Studio
  - Google이 제공하는 공식 통합 개발 환경(IDE)
  - 내장된 Android 에뮬레이터를 사용하여 가상의 Android 디바이스에서 애플리케이션을 테스트하고 디버깅할 수 있음

# 프로젝트 초기화

- **Yarn 설치**
  - Yarn: JavaScript 패키지 관리자. React Native에서 라이브러리 설치시 사용
  - npm install --global yarn
  - Unauthrization 에러: Set-ExecutionPolicy Unrestricted
- **프로젝트 초기화**
  - npx react-native@X.XX.X init AwesomeProject --version X.XX.X
- **매트로 서버 실행하기**
  - React Native 앱에 필요한 모든 자바스크립트 코드와 에셋을 실시간으로 번들링하고, 변경 사항을 빠르게 앱에 적용
  - yarn start
- **빌드**
  - yarn android

**React Native CLI를 이용한 기초 웹뷰 개발**

**웹뷰로 웹 사이트 로드하기**

**(React Native CLI)**

## 학습 목표

01

React Native 프로젝트에 WebView 라이브러리를 설치 할 수 있다

02

WebView 컴포넌트를 이용하여 웹 사이트를 앱으로 보여줄 수 있다

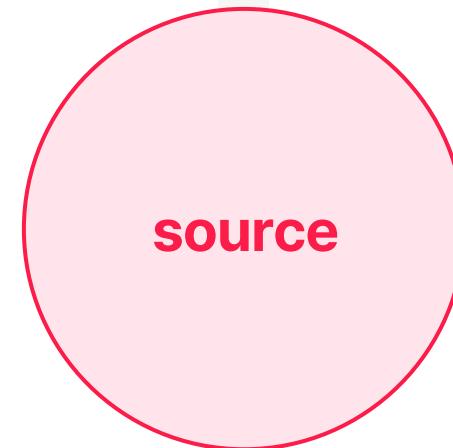
## react-native-webview

- React Native 애플리케이션에서 웹 콘텐츠를 표시하기 위한 컴포넌트인 WebView를 제공하는 라이브러리
- <https://github.com/react-native-webview/react-native-webview>

## 웹 사이트 소스 형태

### URI

인터넷에서 자원을 식별하는데  
사용되는 문자열  
(예: 웹 페이지, 파일)  
" <https://fastcampus.co.kr/>"



### Static HTML

고정된 HTML 코드  
빠른 로딩  
`<h1>Hello WebView!</h1>`

**React Native CLI를 이용한 기초 웹뷰 개발  
Expo의 소개 및 장단점**

## 학습 목표

01

Expo가 무엇인지 이해한다

02

React Native CLI와 Expo의 차이점을 이해할 수 있다

# Expo

- React Native 프로젝트를 더 쉽게 시작하고 개발할 수 있도록 돋는 프레임워크 및 플랫폼
- 장점
  - **쉬운 시작:** React Native 개발 환경을 복잡한 설정 없이 간단히 시작할 수 있음
  - **내장 API와 컴포넌트:** 카메라, 위치, 알림 등 모바일 기기의 하드웨어 기능에 접근할 수 있는 다양한 API를 내장
  - **Expo Go 앱:** 사용중인 디바이스에서 Expo Go라는 모바일 앱을 통해 개발 중인 프로젝트를 실시간으로 테스트 가능
- 단점
  - **기능적 한계:** 모든 네이티브 기능에 접근 할 수 있고, 특정 모듈들은 Expo에서 제공하지 않을 수 있음
  - **최적화 문제:** 네이티브 코드에 접근 하기 어렵기 때문에, 성능 최적화에 한계가 있을 수 있음
    - Expo를 순수 React Native로 전환하여 해결 가능 (Eject)
- 참고 사이트
  - <https://docs.expo.dev/overview/>

# React Native CLI vs Expo 프로젝트 초기화 비교

## React Native CLI

Node, Watchman

Xcode, Cocoapods, Ruby,  
Bundler

Java, Android Studio

-----  
`npx react-native init`

`bundle exec pod install`

`yarn start`

`yarn ios / yarn build`

## Expo

Node, Watchman

Git

Expo Go App

-----  
`npx create-expo-app`

`npx expo`

`npx expo start`

프로젝트 설정

# React Native CLI vs Expo

	React Native CLI	Expo
시작의 용이성	설정이 복잡할 수 있음	쉽고 빠른 프로젝트 시작 가능
네이티브 모듈 사용	모든 네이티브 기능 접근 가능	제한적 Expo SDK에서 지원하는 기능만 사용 가능
코드 작성의 자유도	모든 네이티브 코드 수정 가능	네이티브 코드 접근 제한
필요한 개발 환경	iOS 개발 시 macOS 필수	macOS 없이도 iOS와 Android 개발 가능
성능	높은 성능 직접 최적화 가능	상대적으로 낮은 성능 최적화 제한적
라이브러리 접근	모든 타사 라이브러리 및 도구 사용 가능	Expo 에코시스템 내에서 지원되는 라이브러리 및 도구에 제한됨
커스터마이징 가능성	완전히 커스터마이징 가능	커스터마이징에 제한 Expo 설정을 벗어난 변경사항에는 "eject" 필요

# 언제 무엇을 써야할까?

## React Native CLI

- **고도의 맞춤화 필요:** 애플리케이션에 특정 네이티브 기능이나 세밀한 최적화가 필요한 경우
- **대규모 프로덕션 앱:** 대규모 프로젝트나 성능이 중요한 앱
- **특정 네이티브 라이브러리 사용:** 특정 네이티브 라이브러리나 타사 라이브러리가 필요한 경우
- **완전한 플랫폼 기능 활용:** 모바일 플랫폼의 모든 기능을 활용해야 하는 경우 (예를 들어 특정 API에 접근하거나, 맞춤형 모듈을 개발해야 할 때)

## Expo

- **신속한 프로토타이핑:** 빠르게 아이디어를 구현해야 할 때
- **간단한 앱 개발:** 복잡한 네이티브 기능이 필요 없는 간단한 모바일 애플리케이션을 개발할 때
- **윈도우 사용:** Expo를 사용하면 Windows에서도 iOS 애플리케이션 개발을 진행할 수 있음

**React Native CLI를 이용한 기초 웹뷰 개발**

**앱 개발 환경 셋업 (Expo)**

## 학습 목표

**01** Expo / TypeScript를 이용해서 React Native 프로젝트를 초기화 할 수 있다

**02** Expo를 이용해서 iOS 앱을 실행 할 수 있다.

**03** Expo를 이용해서 Android 앱을 실행 할 수 있다.

# Visual Studio Code (VSCode)

- Microsoft에서 개발한 경량화되면서도 강력한 소스 코드 편집기
- **다양한 언어 지원:** JavaScript, Python, Java, C# 등과 같은 다수의 프로그래밍 언어를 지원
- **확장성:** 풍부한 확장 프로그램 사용 가능 (ESLint, Prettier, Copilot...)
- <https://code.visualstudio.com/>

# macOS

- Node
  - React Native의 JavaScript 런타임 환경
  - Homebrew를 이용해서 설치
    - Homebrew: macOS 패키지 매니저
  - brew install node@18
- Watchman
  - Facebook에서 개발한 파일 시스템 모니터링 도구. React Native에서 파일이 변경되면 자동으로 변경 사항 반영
  - brew install watchman
- Git
  - 소스 코드 관리 시스템
  - brew install git

# 프로젝트 초기화

- Yarn 설치 (Optional, But recommended)
  - Yarn: JavaScript 패키지 관리자. React Native에서 라이브러리 설치시 사용
  - `npm install --global yarn`
- 프로젝트 초기화
  - `yarn create expo --template expo-template-blank-typescript@50`
- 개발 서버 실행하기
  - `yarn start`

# Windows

- Node

- React Native 개발에서 서버 측 스크립트 실행 및 다양한 도구와 라이브러리의 관리를 위한 JavaScript 런타임 환경
- Chocolatey를 이용해서 설치
  - Chocolatey: 윈도우 패키지 매니저
  - choco install -y nodejs --version=18.20.2

- Git

- 소스 코드 관리 시스템
- <https://git-scm.com/download/win>

# 프로젝트 초기화

- Yarn 설치 (Optional, But recommended)
  - Yarn: JavaScript 패키지 관리자. React Native에서 라이브러리 설치시 사용
  - npm install --global yarn
  - Unauthrization 에러: Set-ExecutionPolicy Unrestricted
- 프로젝트 초기화
  - yarn create expo --template expo-template-blank-typescript@50
- 개발 서버 실행하기
  - yarn start

**React Native CLI를 이용한 기초 웹뷰 개발**

**웹뷰로 웹 사이트 로드하기**

**(Expo)**

## 학습 목표

01

React Native 프로젝트에 WebView 라이브러리를 설치 할 수 있다

02

WebView 컴포넌트를 이용하여 웹 사이트를 앱으로 보여줄 수 있다

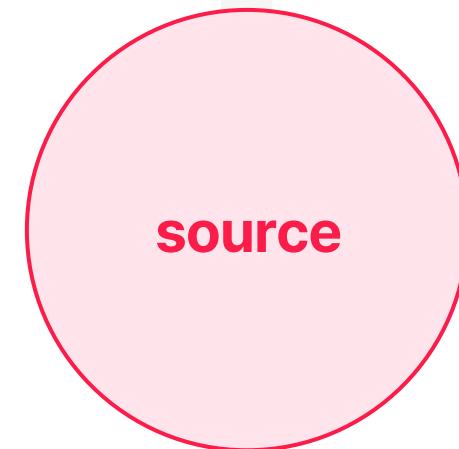
## react-native-webview

- 웹 콘텐츠를 표시하기 위한 컴포넌트인 WebView를 제공하는 라이브러리
- <https://docs.expo.dev/versions/latest/sdk/webview/>

## 웹 사이트 소스 형태

### URI

인터넷에서 자원을 식별하는데  
사용되는 문자열  
(예: 웹 페이지, 파일)  
" <https://fastcampus.co.kr/>"



### Static HTML

고정된 HTML 코드  
빠른 로딩  
`<h1>Hello WebView!</h1>`

**React Native CLI를 이용한 기초 웹뷰 개발**

**학습 리뷰**

# 애플리케이션의 종류

## 네이티브 앱

- 특정 플랫폼을 위해 최적화되어 개발된 애플리케이션
- 빠른 성능 및 하드웨어 기능 접근성
- 높은 개발 및 유지 관리 비용
- 앱 스토어 심사

## 웹 앱

- 웹 기술(HTML, CSS, JS)을 사용하여 개발
- 플랫폼 독립성
- 제한적인 성능 및 기능 접근성
- 앱에 비해 불편한 서비스 접근성

## 하이브리드 앱

- 웹 기술로 개발하여 네이티브 앱의 웹뷰를 통해 실행
- 높은 개발 효율성
- 하드웨어 기능 사용 가능
- 신속한 업데이트
- 순수 네이티브 앱에 비해 낮은 성능

# React Native CLI vs Expo

	React Native CLI	Expo
시작의 용이성	설정이 복잡할 수 있음	쉽고 빠른 프로젝트 시작 가능
네이티브 모듈 사용	모든 네이티브 기능 접근 가능	제한적 Expo SDK에서 지원하는 기능만 사용 가능
코드 작성의 자유도	모든 네이티브 코드 수정 가능	네이티브 코드 접근 제한
필요한 개발 환경	iOS 개발 시 macOS 필수	macOS 없이도 iOS와 Android 개발 가능
성능	높은 성능 직접 최적화 가능	상대적으로 낮은 성능 최적화 제한적
개발 툴 및 라이브러리 접근	모든 타사 라이브러리 및 도구 사용 가능	Expo 에코시스템 내에서 지원되는 라이브러리 및 도구에 제한됨
커스터마이징 가능성	완전히 커스터마이징 가능	커스터마이징에 제한 Expo 설정을 벗어난 변경사항에는 "eject" 필요

## react-native-webview

- 웹 콘텐츠를 표시하기 위한 컴포넌트인 WebView를 제공하는 라이브러리
- 설치 방법
  - yarn add react-native-webview
  - yarn expo install react-native-webview
- 웹 사이트 소스 형태
  - URI
    - 인터넷에서 자원을 식별하는데 사용되는 문자열
    - 예) 웹페이지 (<https://fastcampus.co.kr/>)
  - Static HTML
    - 고정된 HTML 코드
    - 예) <h1>Hello WebView</h1>