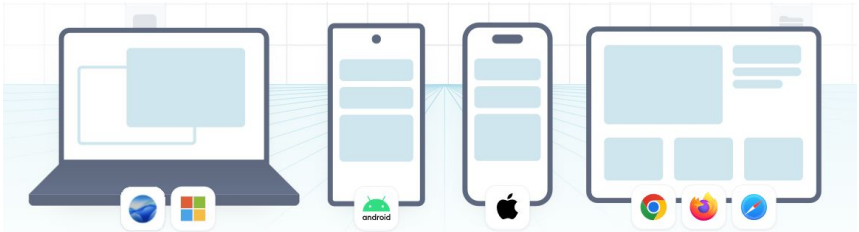
Розробка мобільних додатків

Лекція 1



Контакти

ПІБ: Нерода Сергій Іванович

Електронна пошта: **kipz_nsi@ztu.edu.ua**

Github: **kipznsi**

Gitlab: **@kipz_nsi**

Telegram: **@serhii_neroda**

Група в телеграм: **https://t.me/+NxBGyRxV-c4xMTEy**

Курс: **https://learn.ztu.edu.ua/course/view.php?id=7871**

Оцінювання

60 балів - 8 лр (включно з відвідуванням)

40 балів - модульні контролі

Бонусні бали - 10 (Тези доповідей)

Оцінювання та дедлайни

Всі 8 робіт повинні бути завантажені у **репозиторій**.

Запізнення надає штраф (-1 бал за 2 тижні).

Списування - 0 балів.

Вимоги до оформлення репозиторію з лабораторними роботами

Назва - **MobileLabsRN2026**

Репозиторій має містити **папки для кожної лабораторної роботи**:

`lab1/`

`lab2/`

`...`

`lab8/`

У кожній папці обов’язково:

`README.md` - скріншоти виконання, інструкція запуску.

Головний **README.md**

У корені репозиторію має бути файл **`README.md`** з:

```
 # Лабораторні роботи з дисципліни “Розробка мобільних додатків”


## Студент: Іваненко Іван (ІПЗ-21-1)


### Список робіт

1. [Lab 1 – Вступ. Створення проєкту](lab1/)

...

8. [Lab 8 – Фінальний проєкт](lab8/)
```

Git & коміти

Використовувати **окремі коміти** для логічних змін.

Формат повідомлень у стилі **Conventional Commits (Політика Комітів)**:

```
feat: add login form
```

```
fix: validation bug fixed
```

```
docs: update README
```

Для кожної лабораторної створювати **гілку**:

```
lab1
```

```
...
```

```
lab6
```

і мержити її в `main` після виконання.

Вступ

Мобільний застосунок або додаток – програмне забезпечення, призначене для роботи на смартфонах, планшетах та інших мобільних пристроях.

Розповсюдження застосунків здійснюється через централізовані платформи дистрибуції (магазини застосунків), які забезпечують контроль безпеки, оновлення та монетизацію:

- **Google Play** для **Android**;
- **App Store** для **iOS**;
- **Microsoft Store** для **Windows, Windows Mobile** та **Xbox One**.

Розробка програмного забезпечення для мобільних пристроїв потребує врахування їх обмежень та можливостей.

Мобільні пристрої працюють на акумуляторі та мають менш потужні процесори, ніж персональні комп'ютери, а також мають **більше функцій**, таких як визначення розташування та камери.

Розробникам також доводиться враховувати **широкий спектр розмірів дисплею, різні технічні характеристики та конфігурації обладнання** через сильну конкуренцію мобільного програмного забезпечення та зміни в кожній платформі.

Мобільні додатки спочатку **тестуються** в середовищі розробки, **використовуючи емулятори**, а потім перевіряються на реальних пристроях.

Емулятори забезпечують недорогий спосіб тестування програм на мобільних телефонах, до яких розробники можуть не мати фізичного доступу.

Види мобільних застосунків

1️⃣ Нативні застосунки

Розробляються спеціально для певної платформи (**Android або iOS**).
Використовують мови програмування цієї платформи:

- **Java/Kotlin** – для Android
- **Swift/Objective-C** – для iOS
  ✅ Працюють швидше та мають доступ до всіх функцій пристрою (камера, GPS, датчики).
  ❌ Потрібно писати окремий код для кожної платформи.

Види мобільних застосунків

2️⃣ Кросплатформені застосунки

Один код працює на обох платформах (**Android та iOS**).
Використовують фреймворки:

- **React Native (JavaScript/TypeScript)**
- **Flutter (Dart)**
- **Xamarin (.NET/C#)**
  ✅ Швидше розробляються, але можуть бути менш продуктивними, ніж нативні.

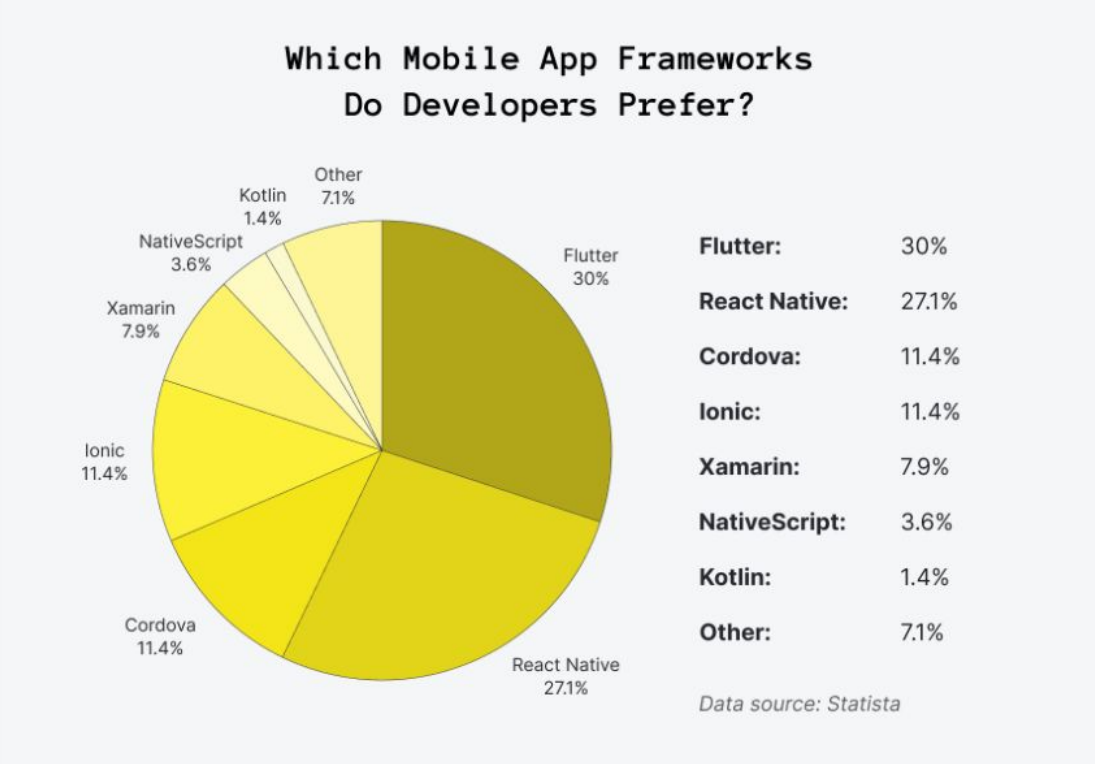
Види мобільних застосунків

3️⃣ Гібридні застосунки

Поєднують веб-технології (**HTML, CSS, JavaScript**) та контейнер для запуску на мобільних пристроях.
Використовують фреймворки:

- **Ionic**
- **Apache Cordova**
  ✅ Легше розробляти, але можуть мають проблеми з продуктивністю.

Статистика вибору фреймворків



Історія **React Native**

2012-2013: Початок експериментів у Facebook

У **2012 році Facebook** зіткнувся з проблемою:

- Додаток Facebook на iOS був **повільним** через використання **WebView** (вбудований браузер).
- Код треба було **переписувати окремо** для iOS та Android.

Інженер **Джордан Волке (Jordan Walke)** запропонував використовувати **JavaScript для керування нативними компонентами**.

- **Суть:** Створення механізму (Bridge), який дозволяє JavaScript-потоку асинхронно взаємодіяти з нативними потоками iOS/Android.

Це стало початком **React Native**.

Історія **React Native**

2014: Внутрішній проєкт Facebook

У 2014 році Facebook **розширив експеримент**:

- Створив команду, яка працювала над **мобільним фреймворком** на основі React.
- Використали ідею **"Learn Once, Write Anywhere"** (навчися один раз – пиши для всіх платформ).
- Фреймворк отримав назву **React Native**.

Уперше технологію показали на **Facebook F8 Conference** у 2014 році.

Історія **React Native**

2015: Офіційний реліз React Native

Березень 2015 – Facebook **відкрив вихідний код React Native**.

Фреймворк підтримував тільки **iOS** (Android додали пізніше).

Реакція спільноти була **дуже позитивною**:

- **Мобільні додатки** можна було писати **на React + JavaScript**.
- Код працював на **iOS без необхідності писати на Swift або Objective-C**.

Історія **React Native**

2016: Додавання підтримки Android

Facebook додав підтримку **Android**, що зробило React Native **повноцінним кросплатформеним фреймворком**.

React Native почали використовувати великі компанії:

- **Facebook Ads Manager**
- **Airbnb**
- **Instagram**
- **Walmart**

Історія **React Native**

2017-2018: Бум популярності та проблеми

React Native **став стандартом** для мобільної розробки на JavaScript.

Вийшло багато **бібліотек та плагінів** для розширення можливостей фреймворку.

Але почали з’являтися проблеми:

- **Продуктивність на Android була гіршою**, ніж у нативних додатках.
- **Велика залежність від JavaScript Bridge** (передача даних між JavaScript і нативним кодом).

Історія **React Native**

2019-2020: Рефакторинг React Native

Facebook розпочав **повний рефакторинг React Native**, щоб покращити продуктивність.

Головні зміни:

- **Fabric:** Новий рушій рендерингу, який працює синхронно. Це дозволяє уникнути "стрибків" інтерфейсу при складних анімаціях.
- **TurboModules:** Нова система взаємодії з нативним кодом. Модулі завантажуються лише тоді, коли вони потрібні (Lazy Loading), що значно прискорює запуск застосунку.
- **Hermes Engine:** Високопродуктивний рушій, розроблений Meta спеціально для Android (а згодом і iOS). Зменшує розмір APK-файлу та споживання пам'яті.
- **Codegen:** Автоматична генерація коду для перевірки типів між JS та нативним середовищем, що зменшує кількість помилок.

Історія **React Native**

2021-2026: Етап зрілості

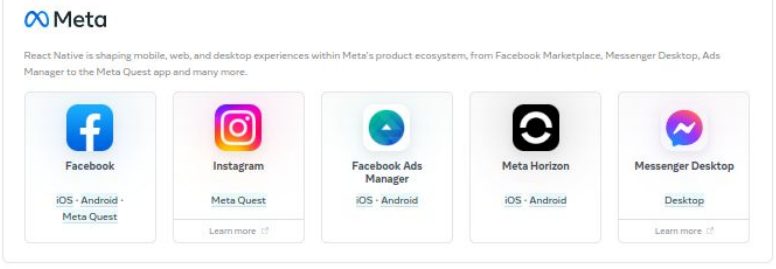
2021-2022: Повна стабільність Hermes. Hermes стає рушієм за замовчуванням для всіх платформ. Покращення підтримки TypeScript.

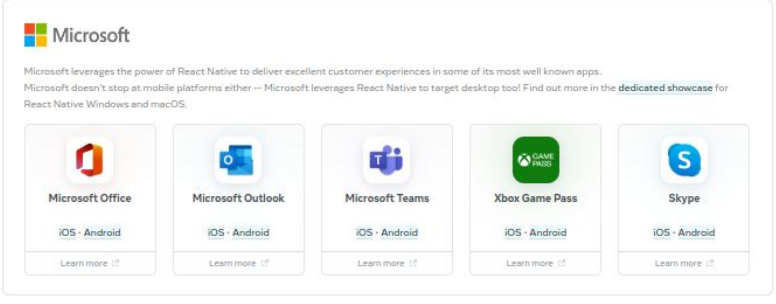
2023-2024: Поширення New Architecture. Більшість популярних бібліотек перейшли на нову архітектуру. Розвиток **Expo** - екосистеми, яка дозволяє розробляти складні застосунки без необхідності запускати Xcode чи Android Studio.

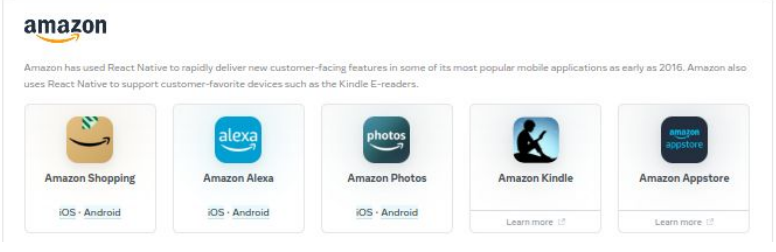
2025-2026: Інтеграція AI та розширення екосистеми.

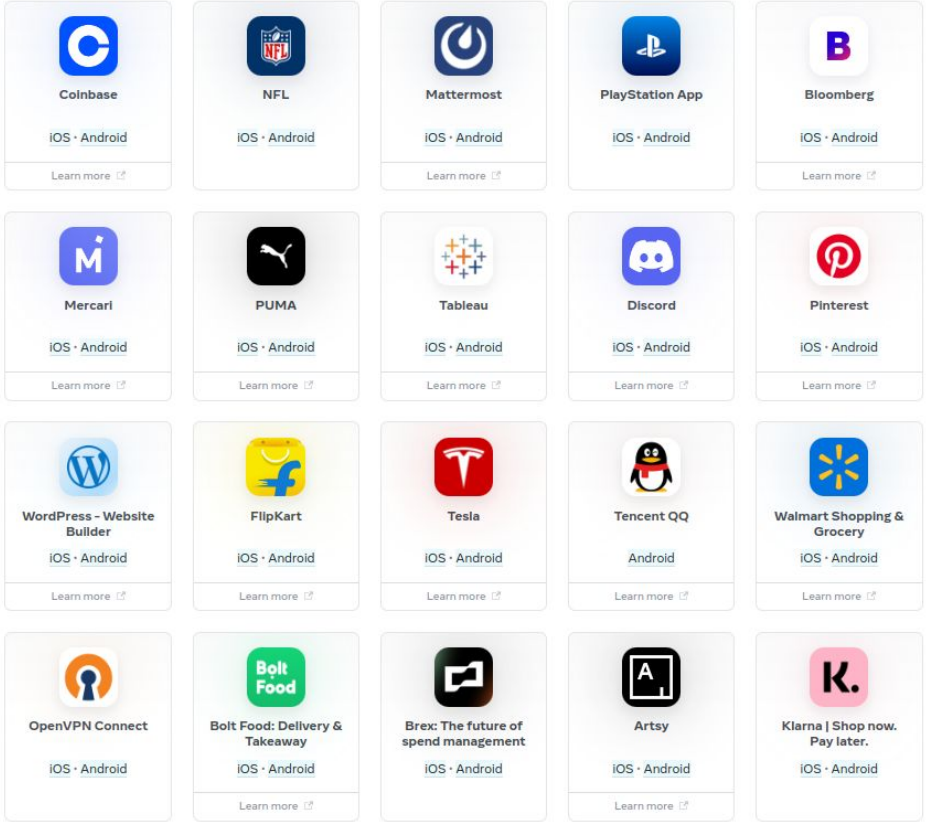
- **AI-Driven Development:** Інтеграція з інструментами генерації коду, що дозволяє створювати інтерфейси за текстовим описом.
- **React Native Everywhere:** Вихід за межі смартфонів. Активна розробка для **Smart TV, Apple Vision Pro (visionOS)** та десктопних систем (Windows/macOS).

Хто використовує









Графік релізів **React Native**

https://github.com/facebook/react-native/releases

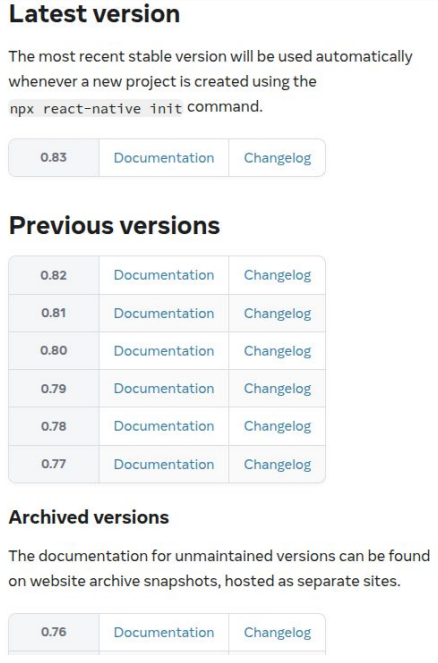
https://reactnative.dev/blog#releases

https://reactnative.dev/versions

Нові версії виходять приблизно кожні **4-6 місяців**.

🚀 Чи варто оновлюватися одразу?

- Якщо **проєкт стабільний**, варто чекати **1-2 місяці** після нового релізу, поки виправлять можливі баги.
- Якщо потрібні **нові функції** – краще оновлюватися швидше.

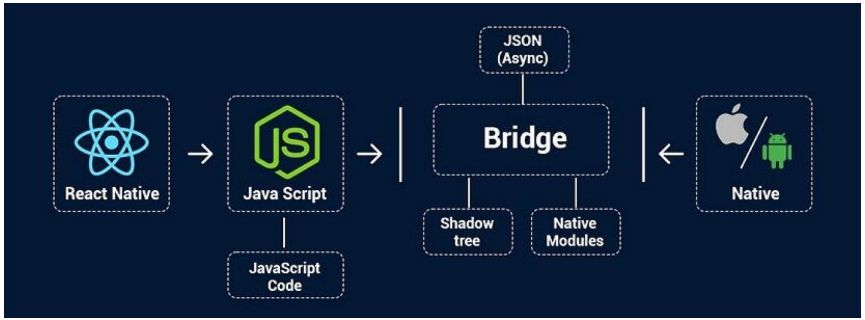


Архітектура **React Native**

Класична архітектура (The Bridge)

До версії 0.68 архітектура базувалася на асинхронному шлюзі — **Bridge**.

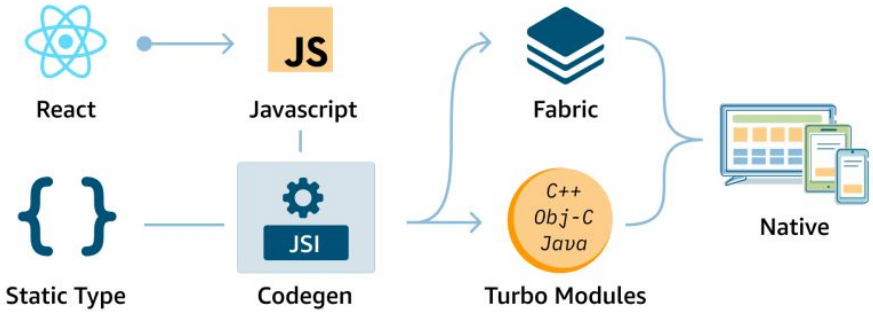
- **Принцип роботи:** JavaScript-потік та Native-потік є ізольованими. Обмін даними відбувається шляхом **серіалізації** об'єктів у формат JSON.
- **Транспорт:** JSON-повідомлення передаються через чергу. Нативний бік десеріалізує ці дані та виконує відповідні системні виклики.
- **Недоліки:**
  - **Високі накладні витрати:** Постійна конвертація даних у JSON споживає ресурси CPU.
  - **Асинхронність:** Неможливість синхронного виконання операцій призводить до проблем із "важкими" UI-елементами (наприклад, складні анімації або обробка високочастотних жестів).
  - **Обмеження пропускної здатності:** При інтенсивному потоці даних Bridge стає "вузьким місцем".



New Architecture (JSI & Fabric)

Сучасна архітектура повністю відмовляється від посередництва Bridge на користь прямої взаємодії.

- **JSI (JavaScript Interface):** Це інтерфейс на базі C++, який дозволяє JavaScript-рушію утримувати прямі посилання на нативні об'єкти .
- **Принцип роботи:** JavaScript може викликати нативні методи синхронно, без необхідності серіалізації даних. Мови фактично співіснують у спільному адресному просторі.
- **Ключові компоненти:**
    - **Fabric:** Новий рендерер, що забезпечує синхронне оновлення UI та пріоритезацію потоків (наприклад, вищий пріоритет для введення даних користувачем).
    - **TurboModules:** Система нативних модулів з підтримкою "лінивого" завантаження, що зменшує споживання оперативної пам'яті та пришвидшує старт додатку.



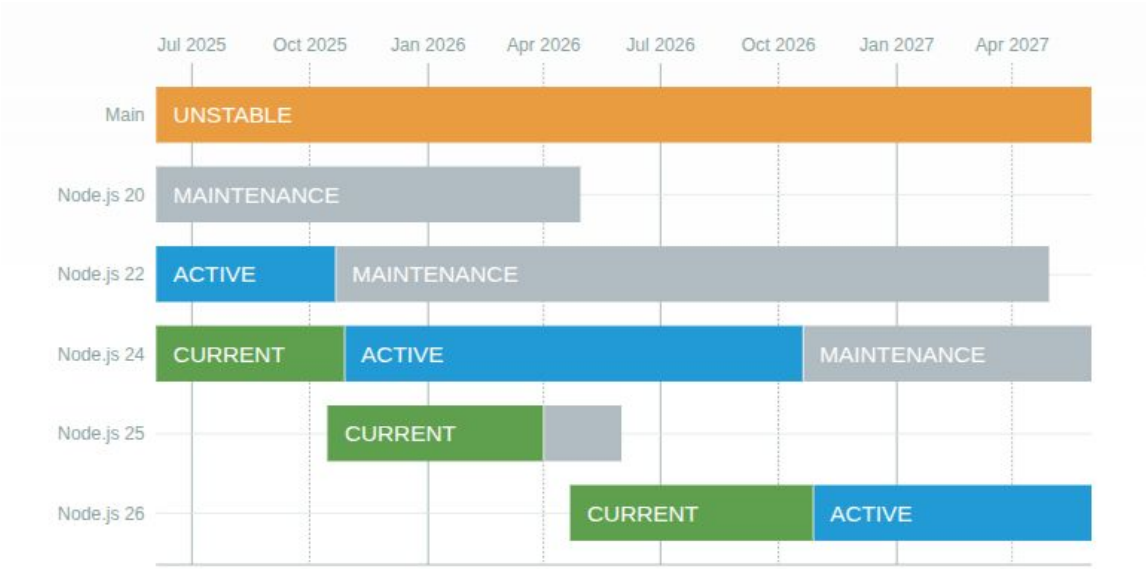
Початок роботи з **React Native**

Для початку роботи з **React Native** потрібно встановити таке програмне забезпечення:

1. **Node.js**: React Native використовує Node.js для виконання JavaScript-коду на сервері та на робочій станції розробника. Можна завантажити та встановити Node.js з офіційного веб-сайту Node.js.
2. **Package manager**: React Native використовує **npm** (Node Package Manager) або **Yarn** для управління залежностями та встановлення необхідних бібліотек. Можна вибрати будь-який з них, але зазвичай рекомендується використовувати **npm**.
3. **SDK для Android або Xcode для iOS**: Для розробки мобільних додатків під **Android** потрібно встановити **Android Studio** та **Android SDK**. Для розробки додатків під **iOS** вам потрібно встановити **Xcode** та **iOS SDK**.
4. **React Native CLI або Expo CLI**: Потрібно вибрати та встановити інтерфейс командного рядка (**CLI**), який дозволяє створювати нові проєкти та виконувати інші операції.
5. **Редактор коду**: Для розробки потрібен редактор коду для написання **JavaScript-коду** та **React-компонентів**. Можна використовувати будь-який текстовий редактор або **IDE**, наприклад **WebStorm, Visual Studio Code, Sublime Text або Atom**.

Node.JS

https://nodejs.org/

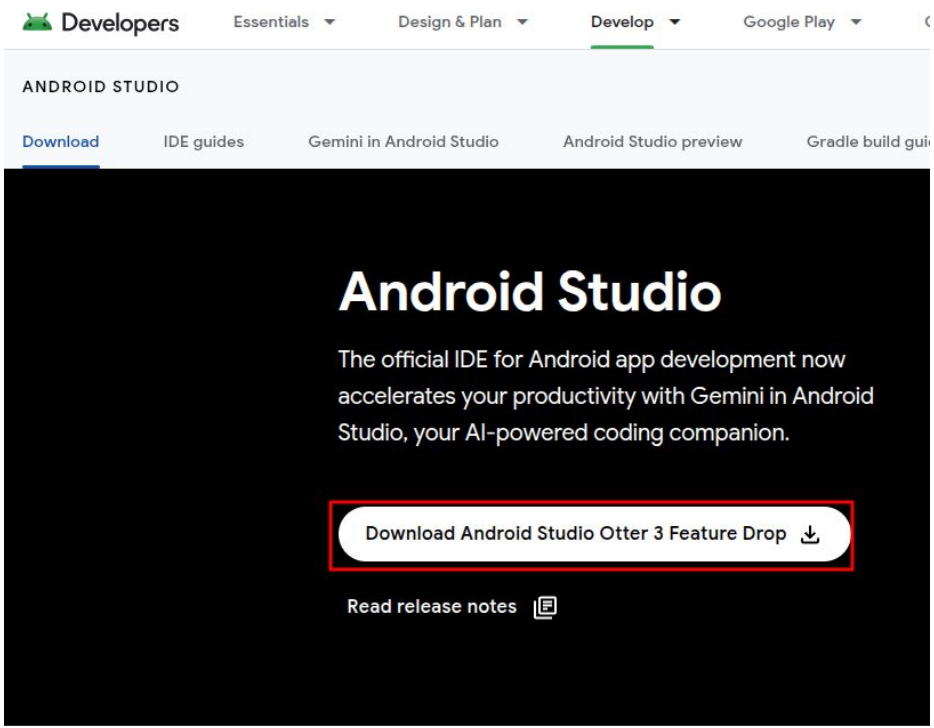


Android Studio

1. Завантажити з офіційного сайту: developer.android.com/studio
2. Встановити необхідні SDK та емулятори.

Android Studio

https://developer.android.com/studio



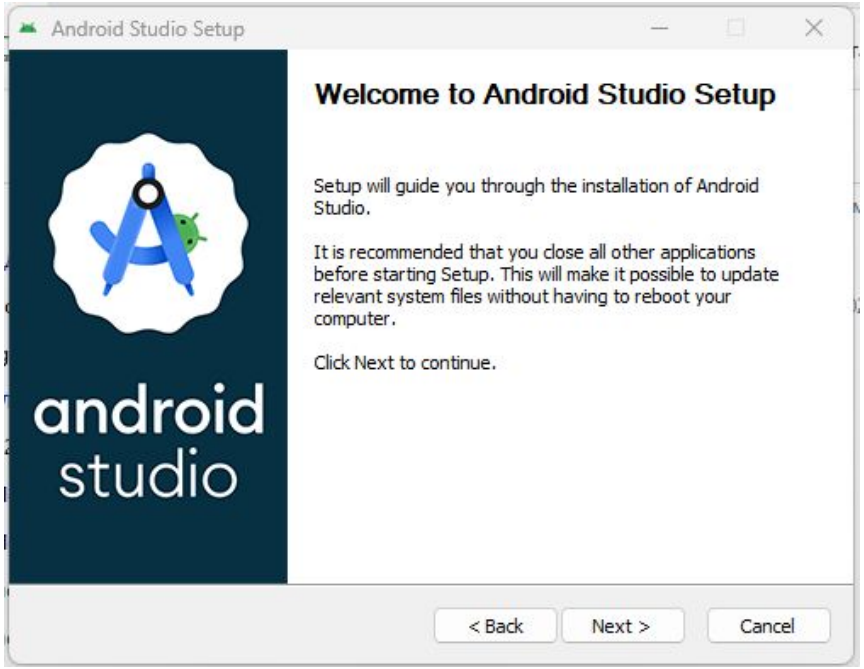
Releases

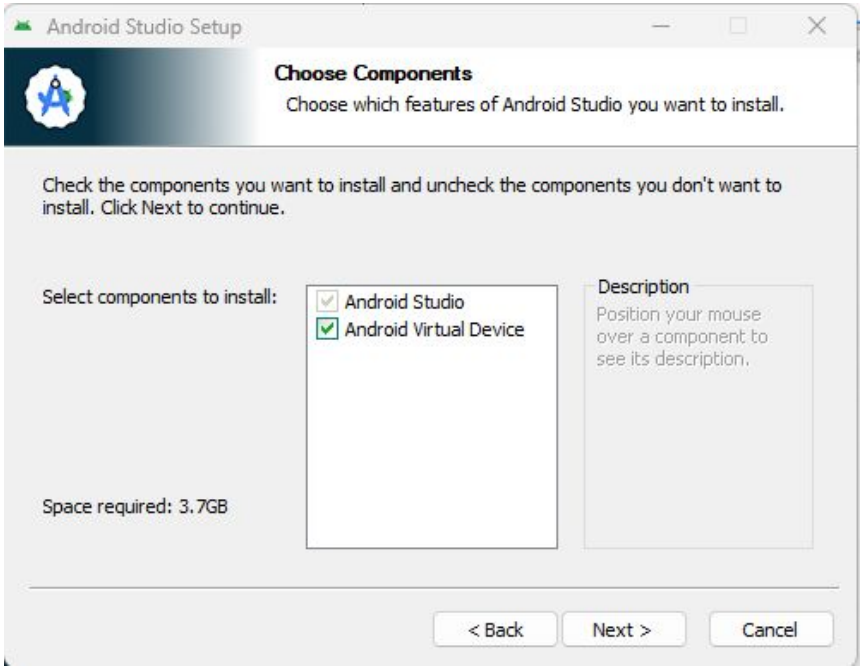
https://developer.android.com/studio/releases

Google використовує **алфавітний порядок назв тварин** для позначення великих оновлень своєї IDE. Ця традиція допомагає розробникам орієнтуватися в хронології версій

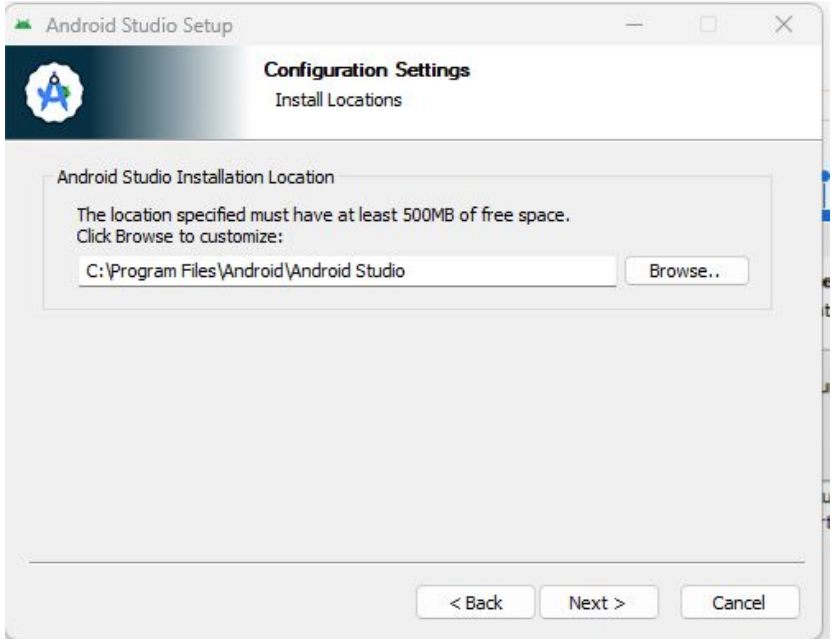
Android Studio version
Otter 3 Feature Drop | 2025.2.3
Otter 2 Feature Drop | 2025.2.2
Otter | 2025.2.1
Narwhal 4 Feature Drop | 2025.1.4
Narwhal 3 Feature Drop | 2025.1.3
Narwhal Feature Drop | 2025.1.2
Narwhal | 2025.1.1
Meerkat Feature Drop | 2024.3.2
Meerkat | 2024.3.1

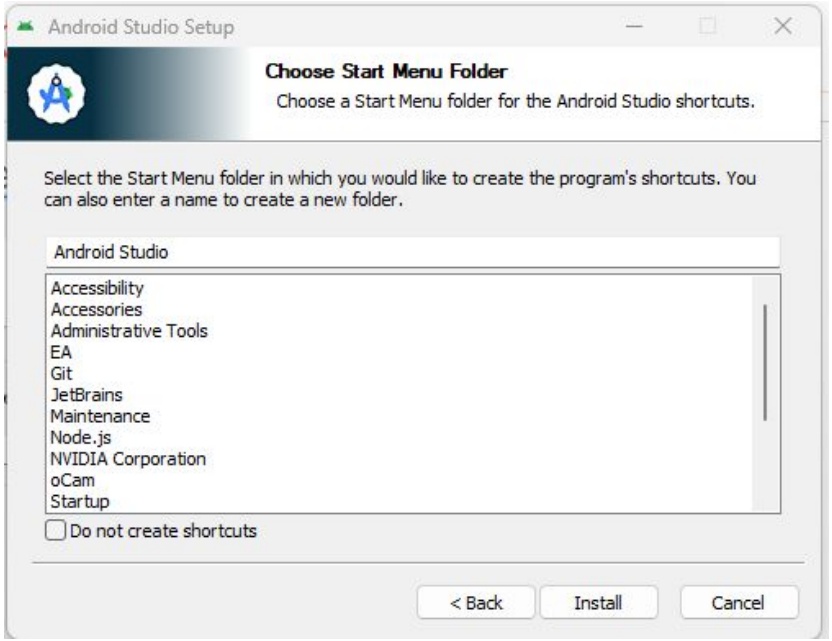
Android Studio Встановлення



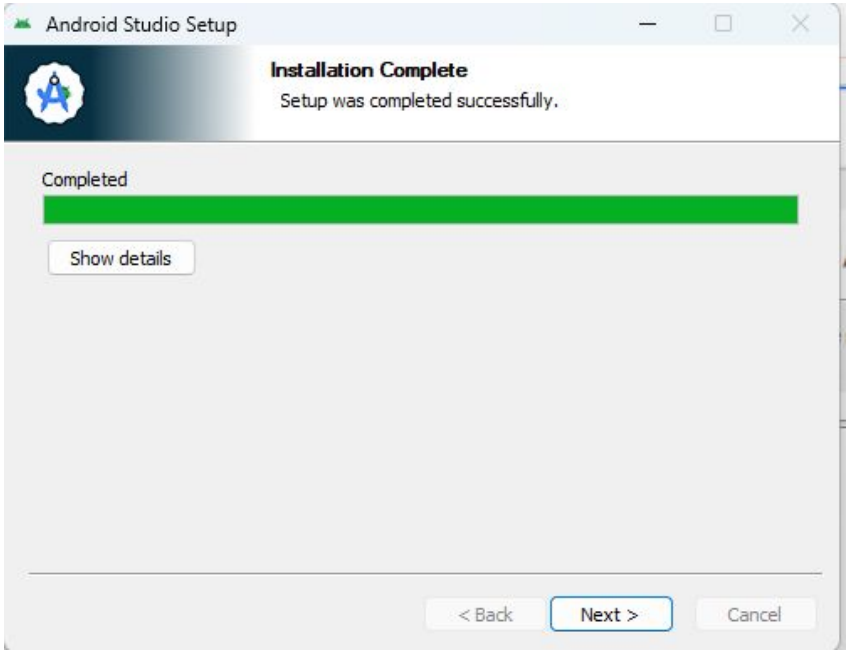


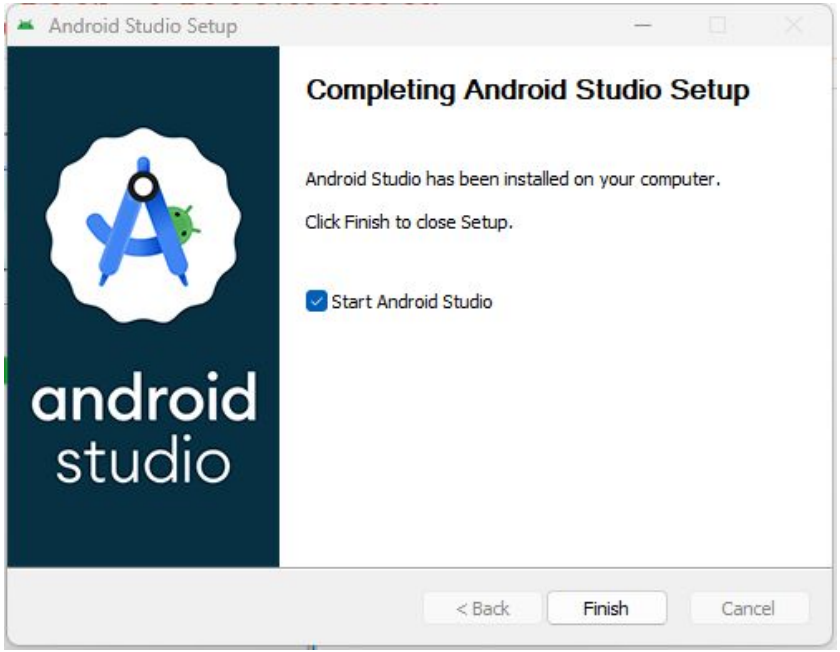
Android Studio Встановлення



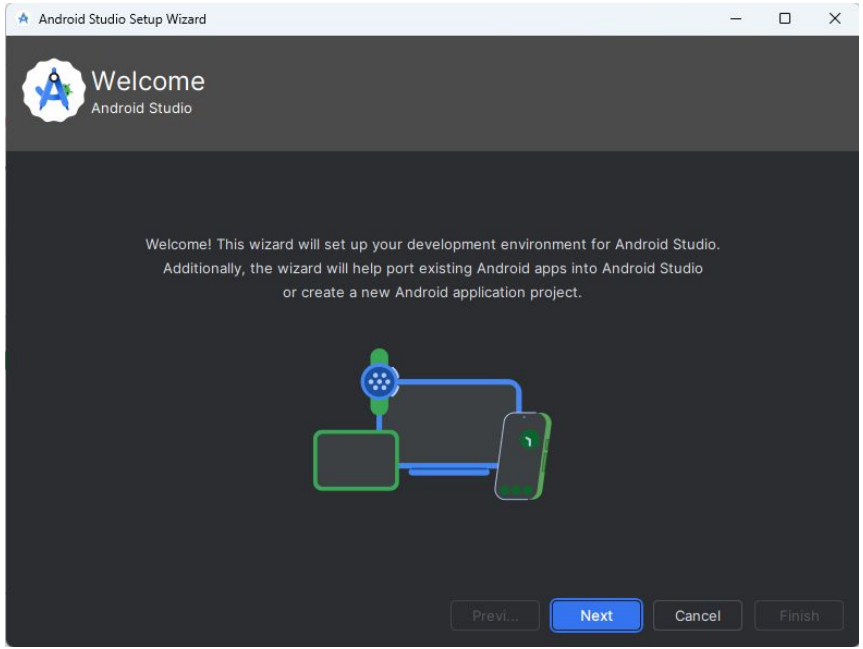


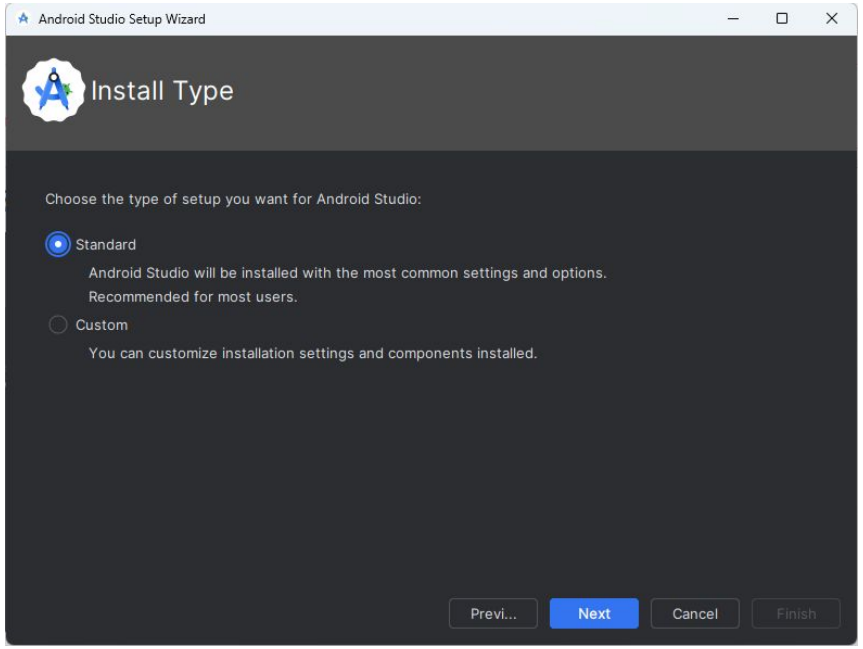
Android Studio Встановлення



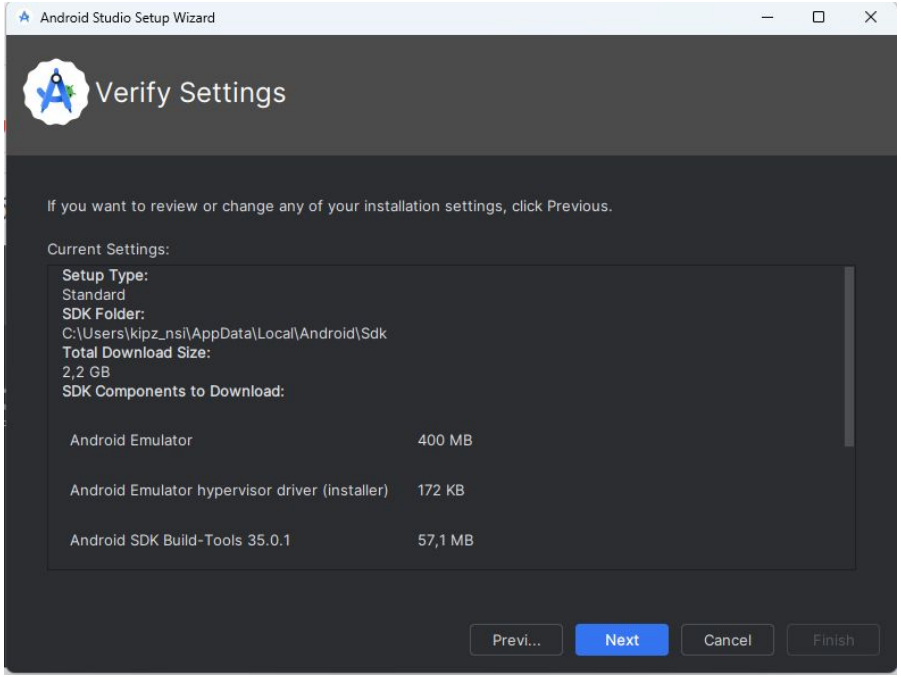


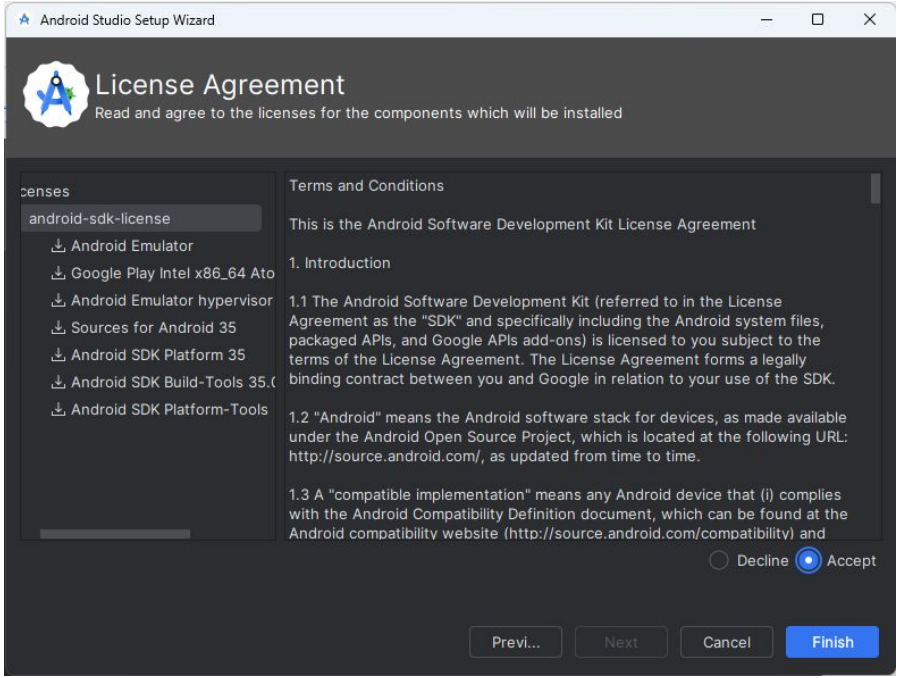
Android Studio Встановлення

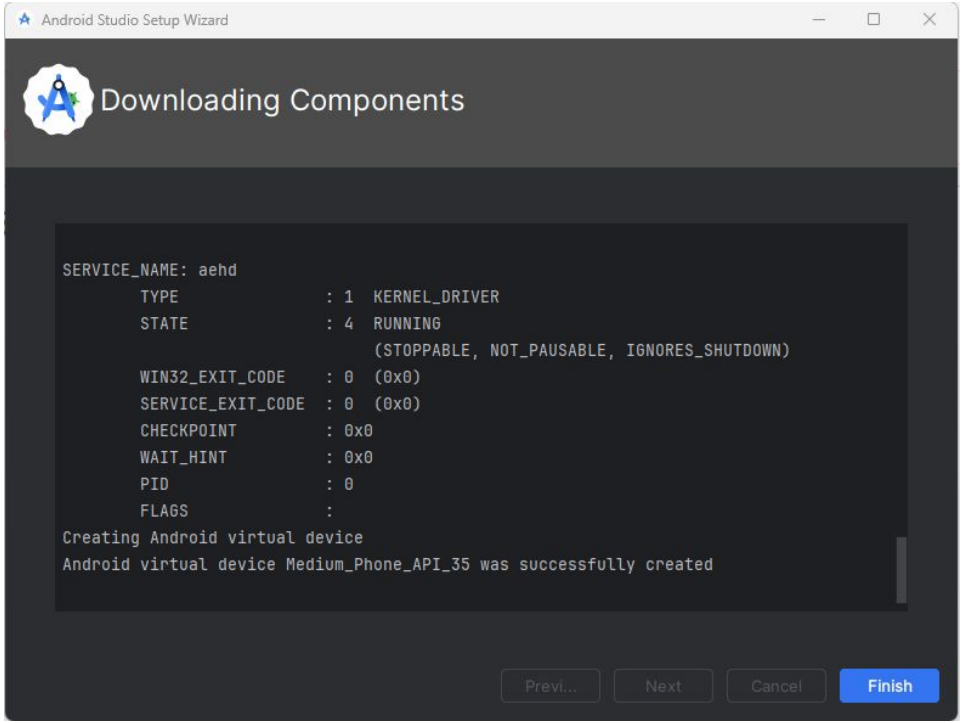


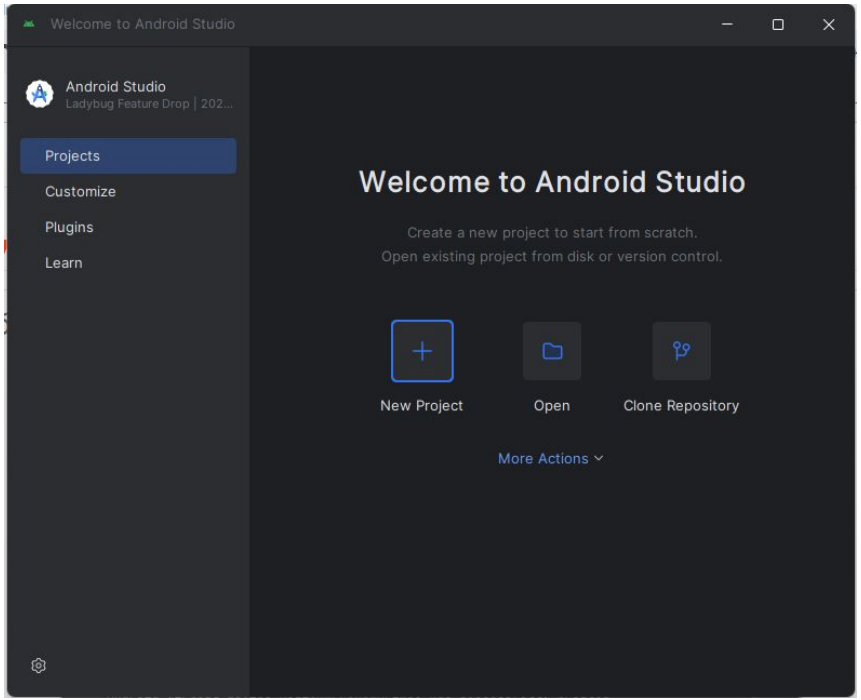


Android Studio Встановлення






Android Studio Setup Wizard
Downloading Components
SERVICE_NAME: aehd
TYPE : 1 KERNEL_DRIVER
STATE : 4 RUNNING
(STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
WIN32_EXIT_CODE : 0 (0x0)
SERVICE_EXIT_CODE : 0 (0x0)
CHECKPOINT : 0x0
WAIT_HINT : 0x0
PID : 0
FLAGS :
Creating Android virtual device
Android virtual device Medium_Phone_API_35 was successfully created
Previ...
Next
Cancel
Finish

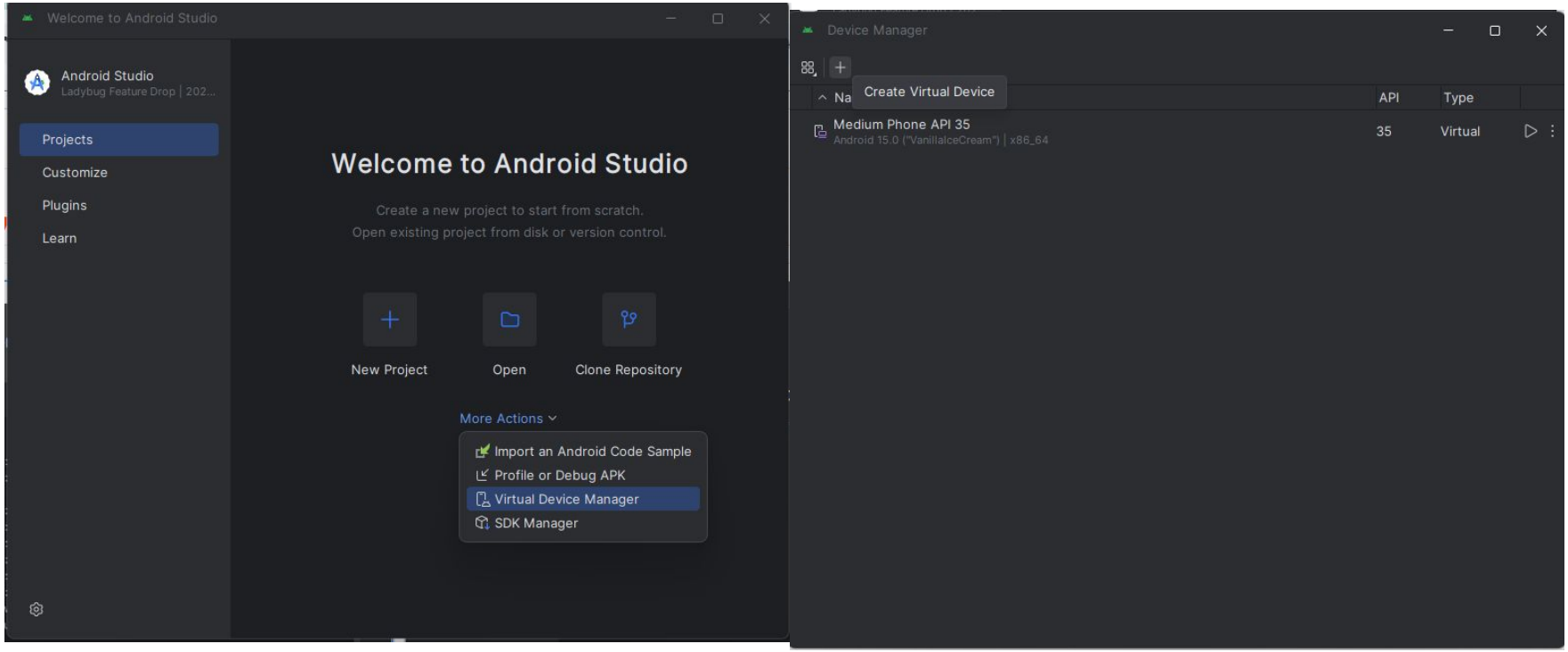

Welcome to Android Studio
Android Studio
Ladybug Feature Drop | 202...
Projects
Customize
Plugins
Learn
Welcome to Android Studio
Create a new project to start from scratch.
Open existing project from disk or version control.
New Project
Open
Clone Repository
More Actions

Тестування застосунку

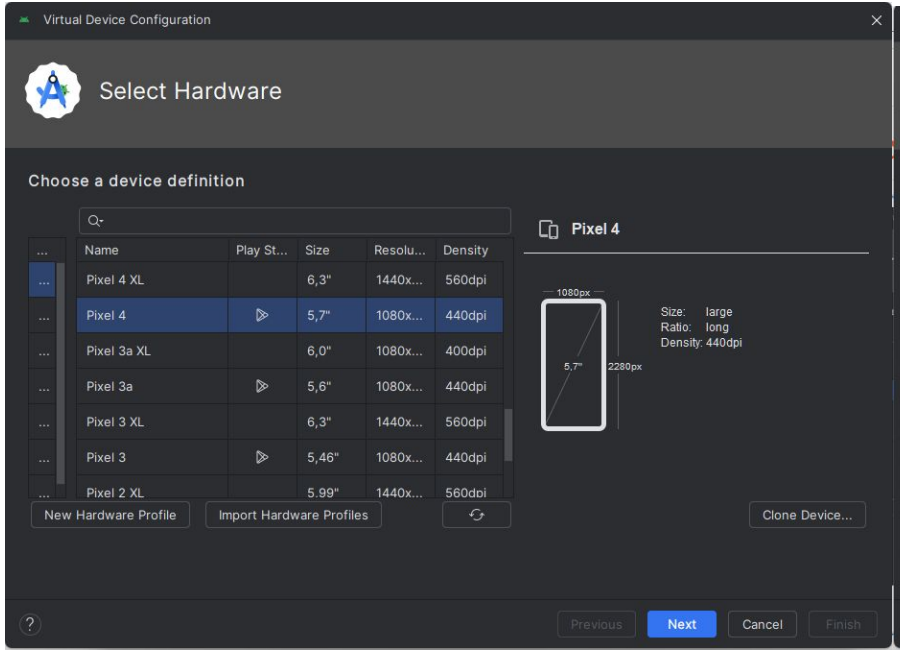
Після встановлення **Android Studio** та необхідних **SDK**, є два основні сценарії для тестування застосунку:

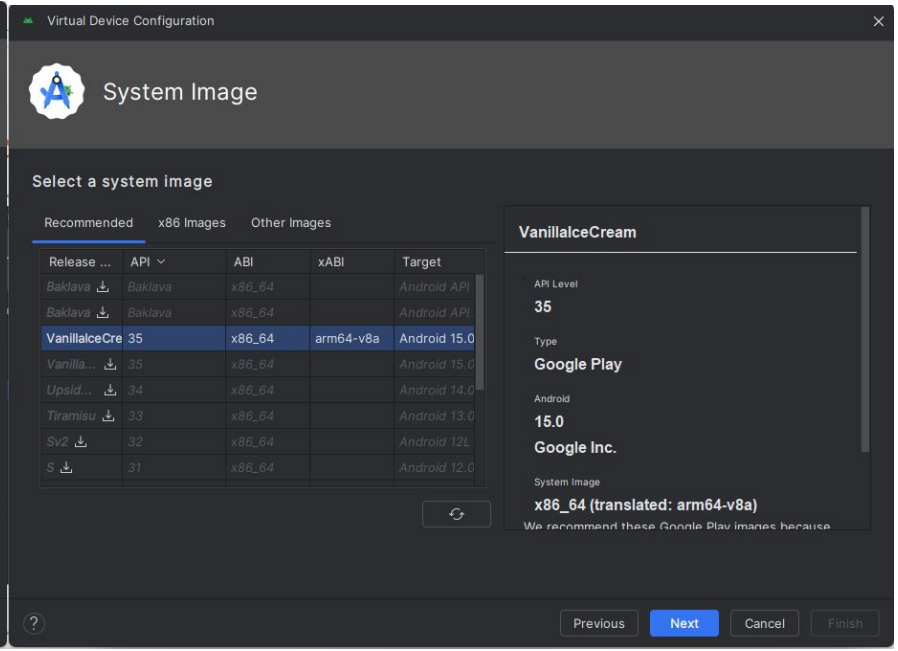
1. **Android Emulator (Віртуальний пристрій):**
    - *Плюси:* Не потребує наявності реального смартфона; дозволяє тестувати різні версії Android та розміри екранів.
    - *Мінуси:* Споживає значні ресурси ОЗП (RAM) та процесора ПК.
2. **Фізичний пристрій (Real Device):**
    - *Плюси:* Максимально точне відображення продуктивності; не навантажує комп'ютер; дозволяє тестувати специфічні функції (камеру, акселерометр).
    - *Мінуси:* Потребує попереднього налаштування системи.

Android Emulator

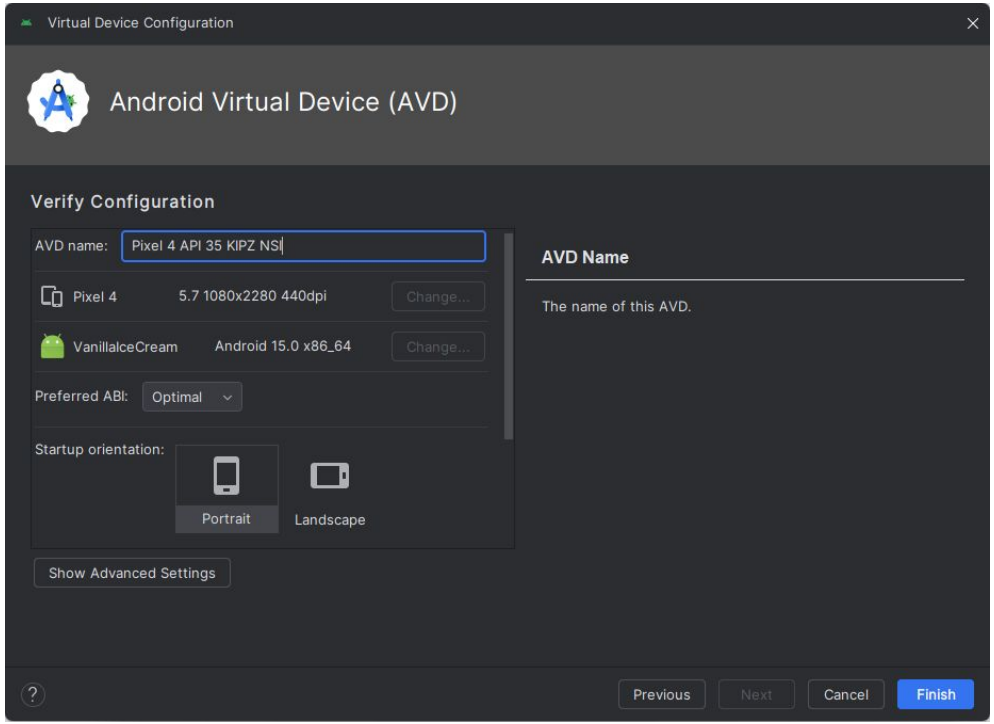


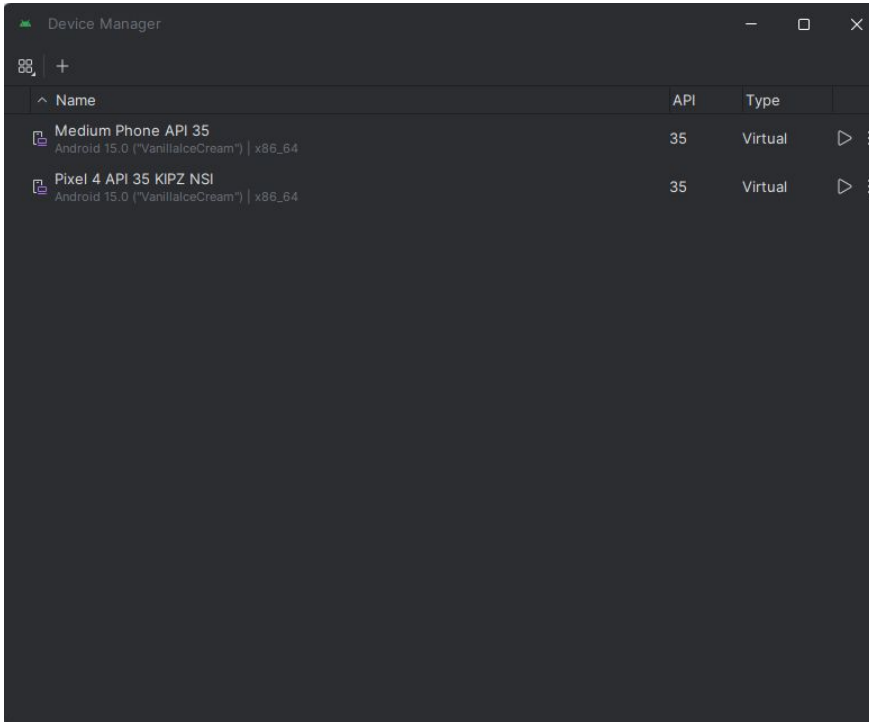
Android Emulator



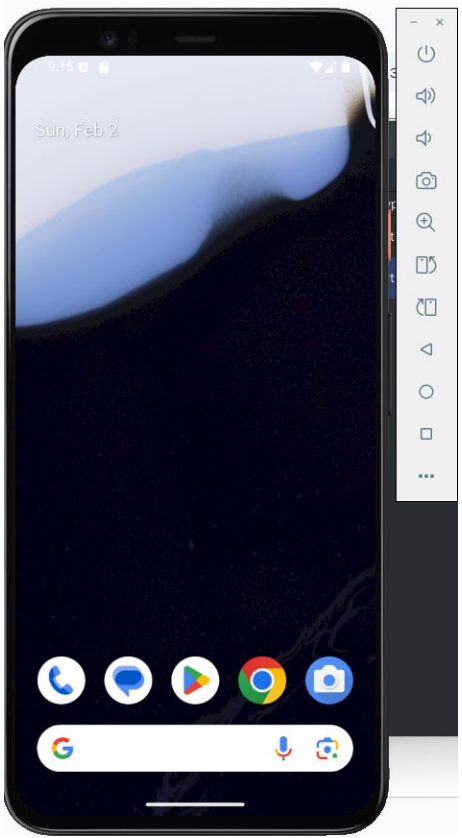


Android Emulator



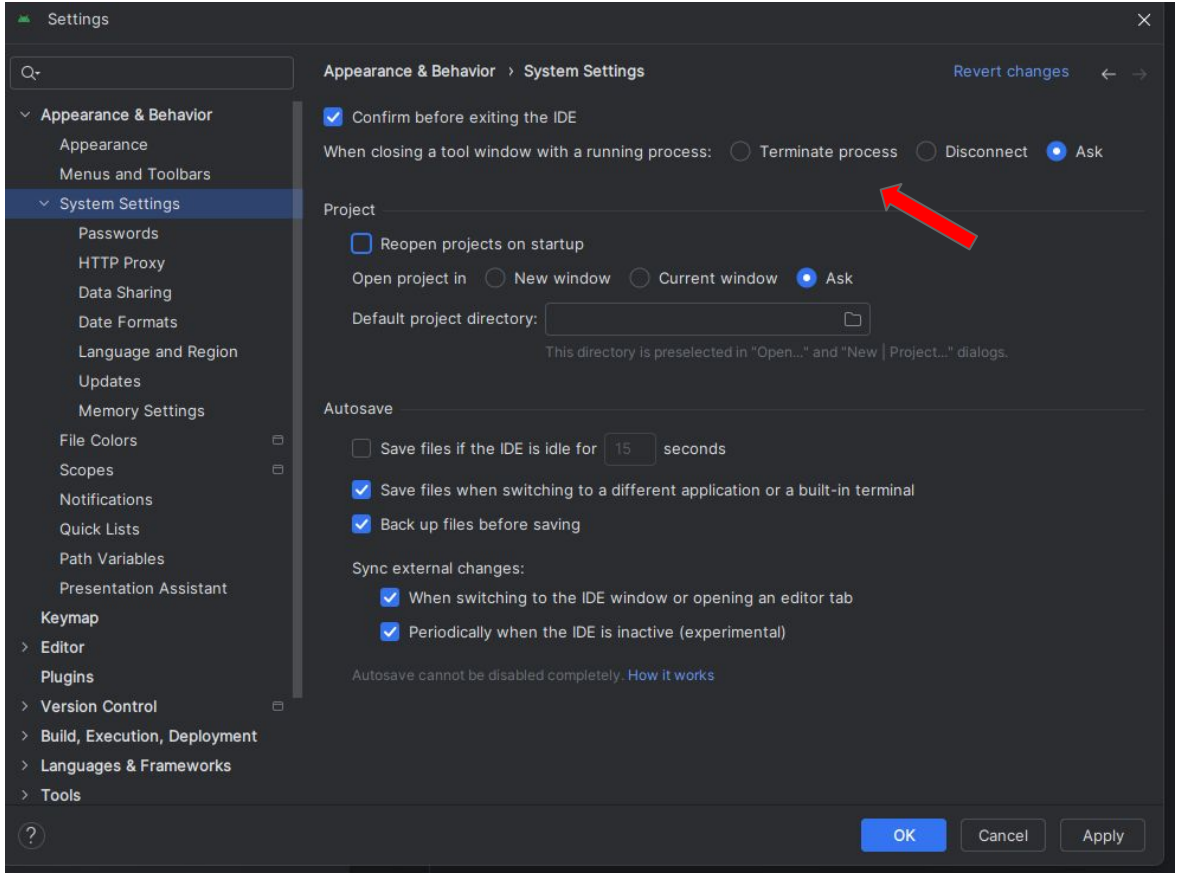


Android Emulator




```
kipz_nsi@Best MINGW64 ~/Desktop/react native
$ adb devices
List of devices attached
emulator-5554   device
```

Android Emulator



Можливі проблеми

Віртуалізація

Емулятор Android або iOS – це **віртуальний пристрій, що працює всередині комп’ютера**. Щоб запускати та виконувати код, емулятору потрібен доступ до ресурсів **процесора, оперативної пам’яті та графіки**.

- **Без віртуалізації**: емулятор працює **повільно**, тому що не має прямого доступу до ресурсів процесора.
- **З увімкненою віртуалізацією**: комп’ютер виділяє окремі потоки процесора, і емулятор працює майже так само швидко, як реальний пристрій.

Якщо віртуалізація вимкнена, Android-емулятор працює повільно або взагалі не запускається!

Віртуалізація

У BIOS/UEFI розділ:

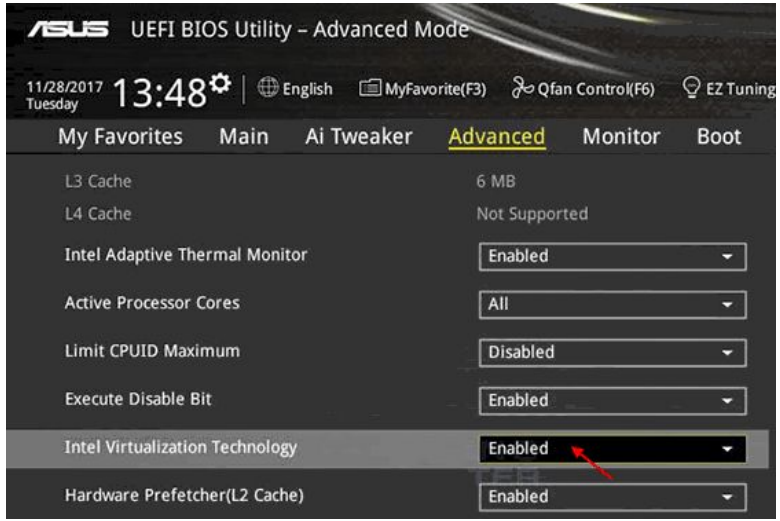
"Advanced", **"CPU Configuration"** або **"Processor"**.

Параметри:

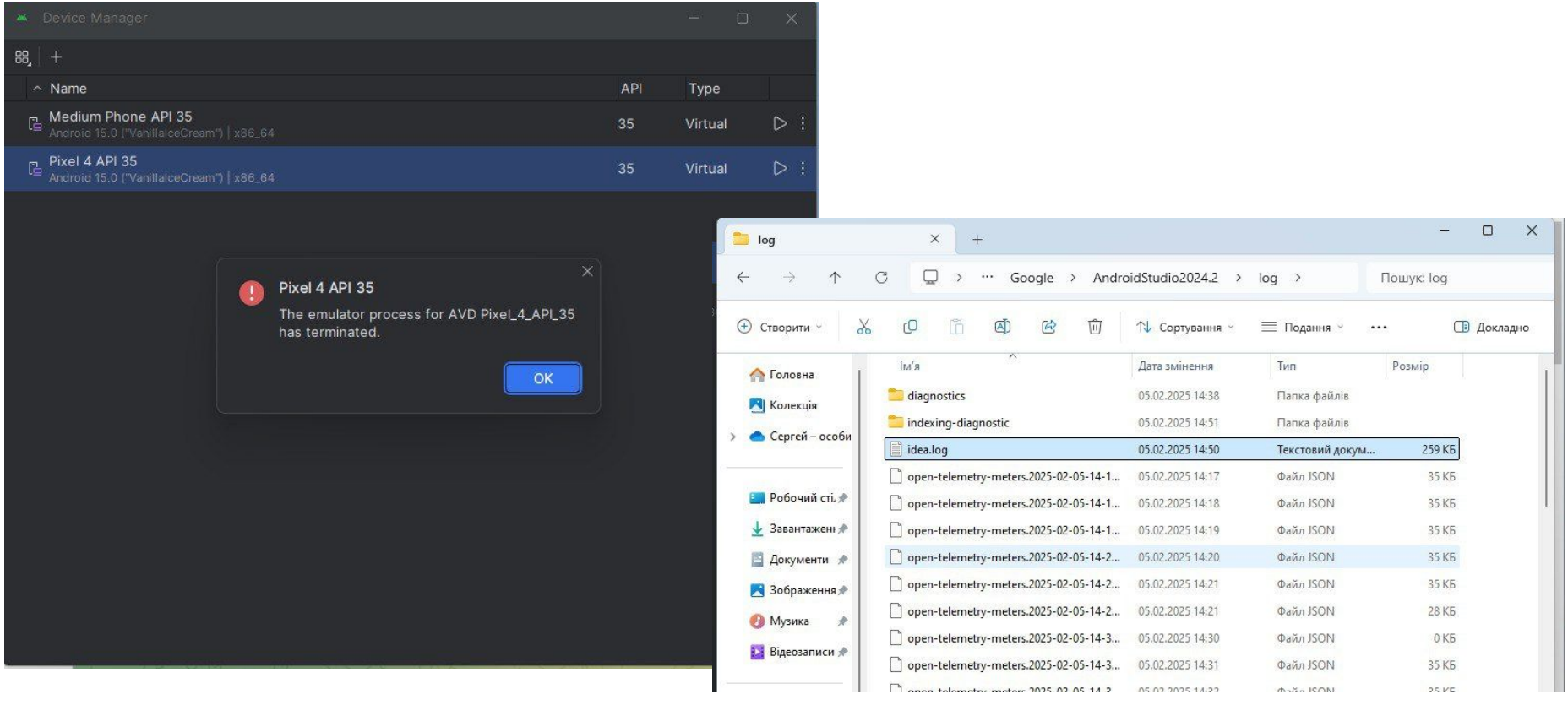
- `Intel VT-x` (для Intel)
- `AMD-V` (для AMD)

Увімкнути (Enable) опцію.

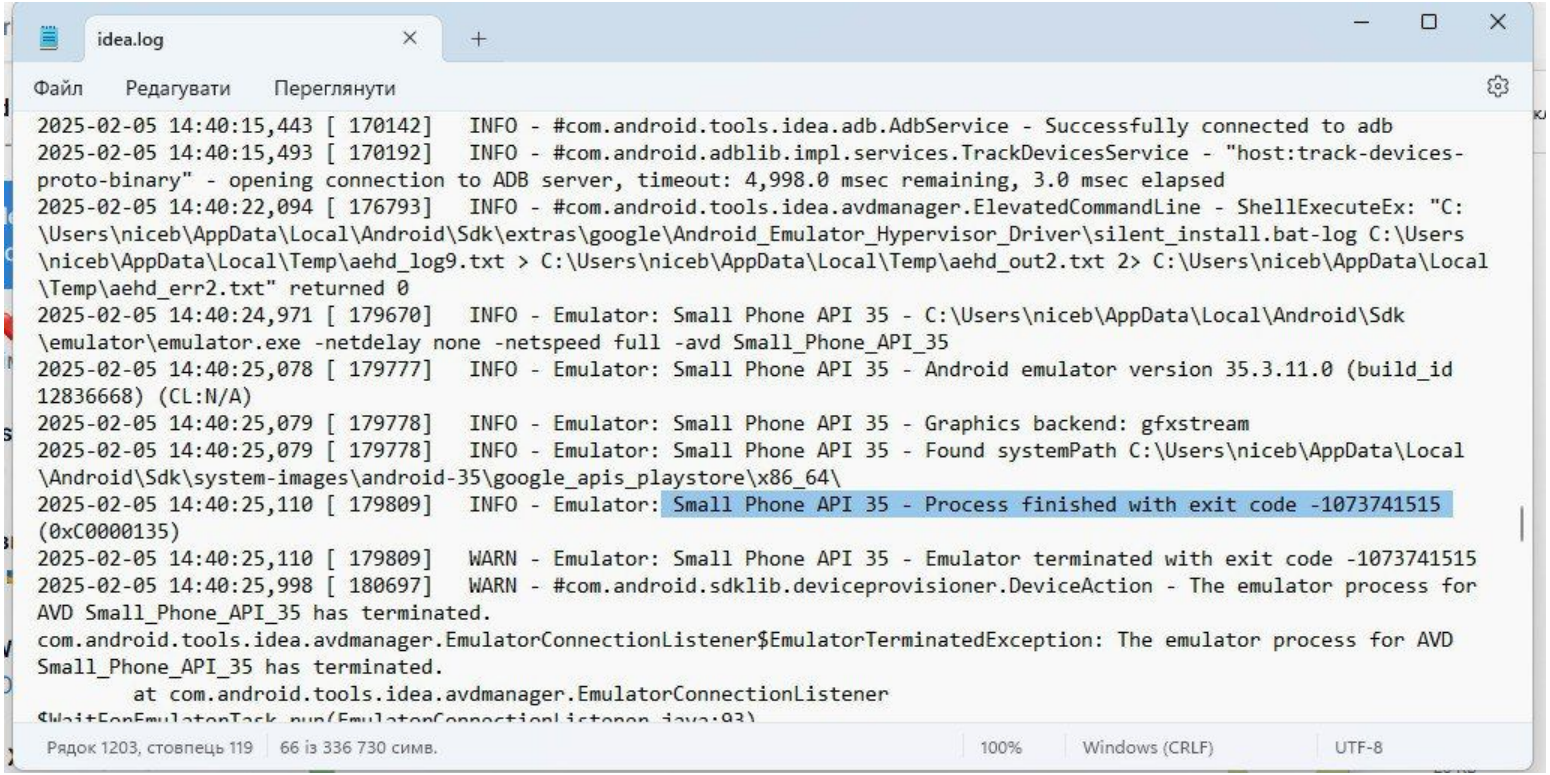
Зберегти зміни (`F10` → `Save & Exit`).



Перегляд логів



Перегляд логів

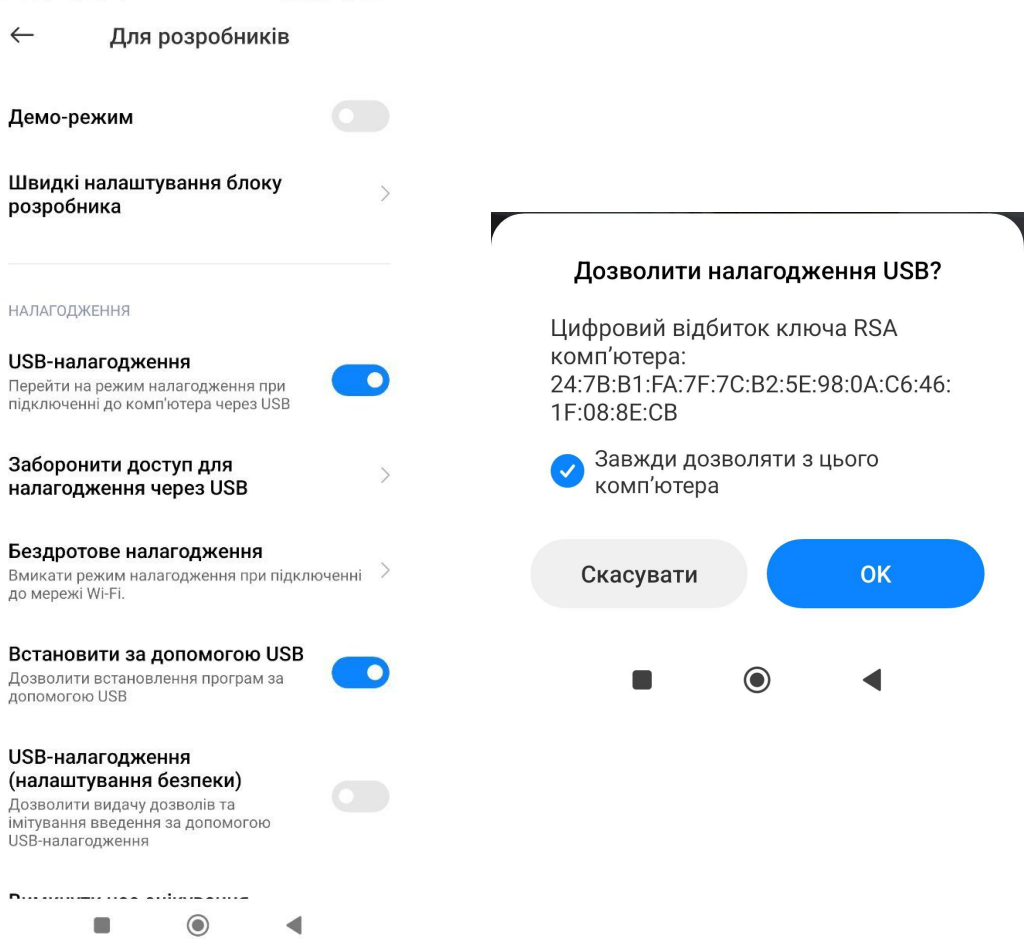


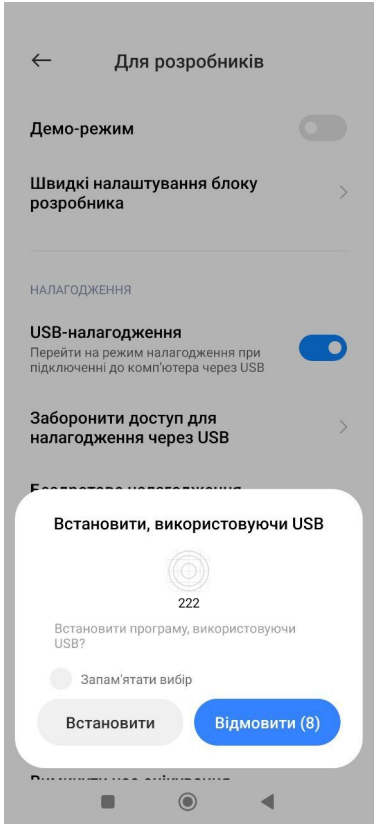
Фізичний пристрій Android

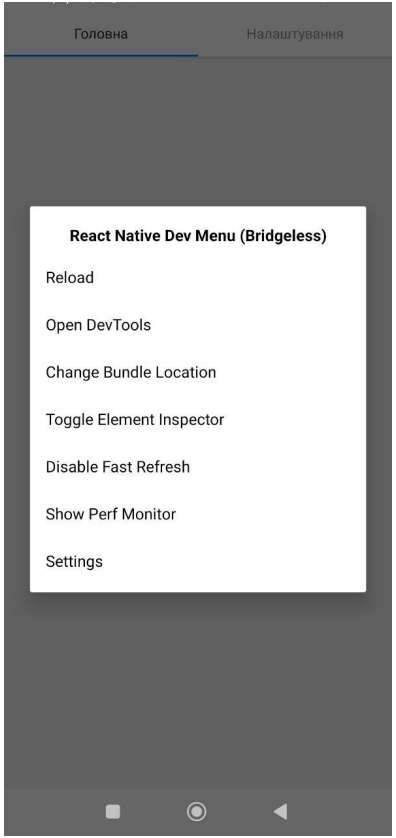
Щоб працювати з React Native на фізичному Android-пристрої, необхідно увімкнути режим розробника та підключити пристрій до комп'ютера.

1. Відкрийте **Налаштування** (`Settings`).
2. Перейдіть у **Про телефон** (`About phone`).
3. Натисніть **7 разів** на **"Номер збірки"** (`Build number`), доки не з’явиться повідомлення **"Ви стали розробником!"**.
4. Поверніться в **Налаштування** та перейдіть до пункту **Для розробників** (`Developer options`).
5. Увімкніть **"Налагодження через USB"** (`USB Debugging`).

[Configure on-device developer options | Android Studio]


Для розробників
Демо-режим
Швидкі налаштування блоку розробника
НАЛАГОДЖЕННЯ
USB-налагодження
Перейти на режим налагодження при підключенні до комп'ютера через USB
Заборонити доступ для налагодження через USB
Бездротове налагодження
Вмикати режим налагодження при підключенні до мережі Wi-Fi.
Встановити за допомогою USB
Дозволити встановлення програм за допомогою USB
USB-налагодження (налаштування безпеки)
Дозволити видачу дозволів та імітування введення за допомогою USB-налагодження
Дозволити налагодження USB?
Цифровий відбиток ключа RSA комп'ютера:
24:7B:B1:FA:7F:7C:B2:5E:98:0A:C6:46:1F:08:8E:CB
Завжди дозволяти з цього комп'ютера
Скасувати
OK


Для розробників
Демо-режим
Швидкі налаштування блоку розробника
НАЛАГОДЖЕННЯ
USB-налагодження
Перейти на режим налагодження при підключенні до комп'ютера через USB
Заборонити доступ для налагодження через USB
Встановити, використовуючи USB
222
Встановити програму, використовуючи USB?
Запам'ятати вибір
Встановити
Відмовити (8)


Головна
Налаштування
React Native Dev Menu (Bridgeless)
Reload
Open DevTools
Change Bundle Location
Toggle Element Inspector
Disable Fast Refresh
Show Perf Monitor
Settings

React Native CLI , Expo CLI

React Native CLI

2015 – Епоха "Ручної збірки": Facebook представив **React Native**, але не було зручного інструменту для швидкого створення проєктів.

2016 – з'явилася перша версія `react-native-cli`, яка дозволяла створювати нові проєкти через команду `react-native init`.

2019 – Facebook оновив CLI та інтегрував його у ядро React Native. Тепер команда `react-native init` більше не використовується, а замість неї використовують **`npx react-native`**.

2020-2023 – React Native CLI продовжує розвиватися, додаючи нові можливості та покращуючи підтримку iOS/Android.

Expo

Expo – це платформа для **спрощеної розробки мобільних додатків на React Native** без необхідності редагувати нативний код (Swift/Kotlin).

Ключові події в історії Expo:

- **2015** – Facebook відкриває вихідний код **React Native**, що дозволяє писати мобільні додатки на JavaScript.
- **2016** – розробники **Чарлі Чівер, Джеймс Айд** та команда створюють **Exponent**, щоб зробити React Native простішим у використанні.
- **2017** – **Exponent перейменовується в Expo** і стає **відкритим проєктом**.
- **2018-2020** – Expo стає **стандартним інструментом для розробки мобільних додатків**, додається підтримка Expo SDK.
- **2021+** – Expo запускає **EAS (Expo Application Services)**, що дозволяє деплоїти додатки.

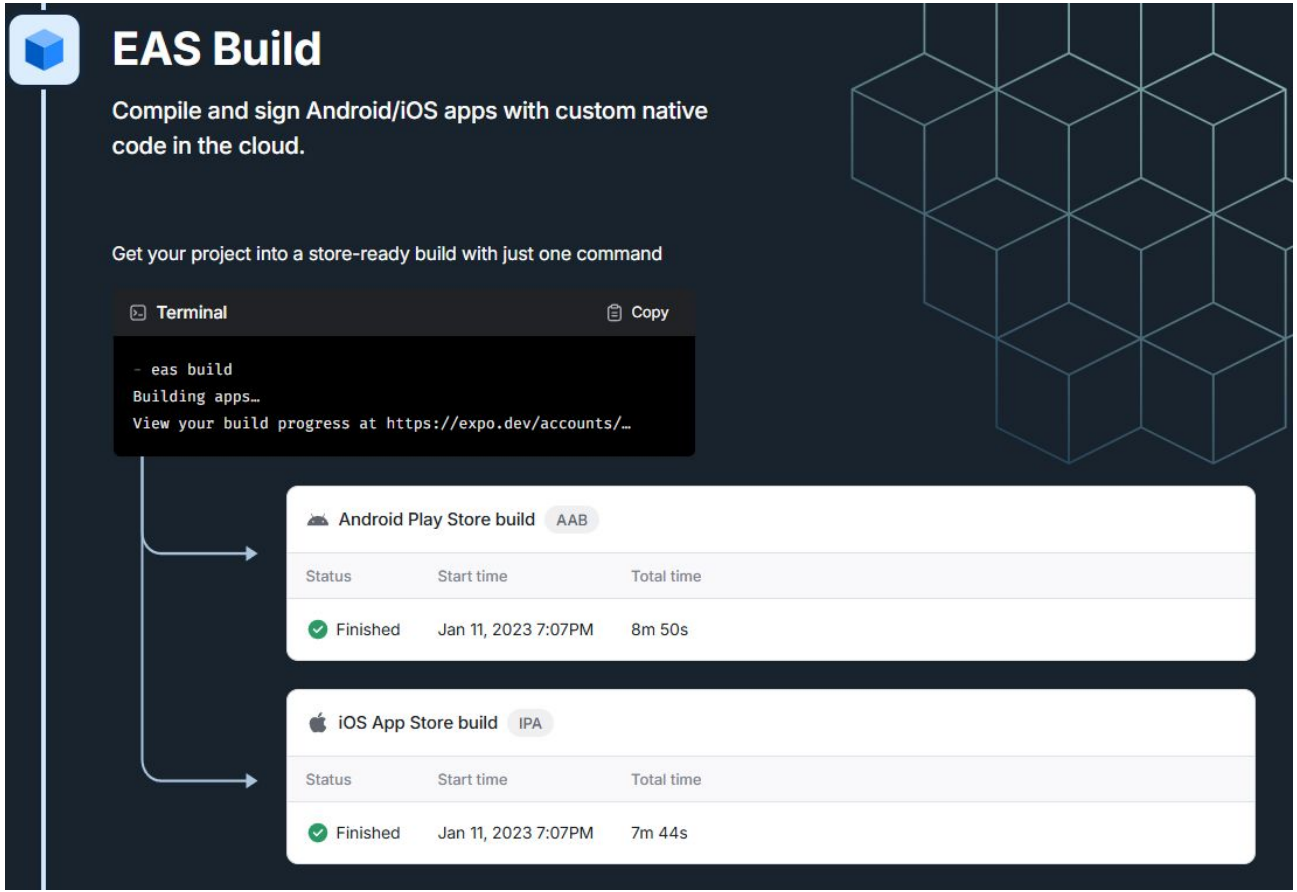
Переваги Expo

Не потрібно Xcode/Android Studio для запуску додатків.

Expo Go дозволяє тестувати код миттєво на телефоні.

Вбудовані API для камери, геолокації, push-нотифікацій.

Швидкий деплой через EAS без складних налаштувань.


EAS Build
Compile and sign Android/iOS apps with custom native code in the cloud.
Get your project into a store-ready build with just one command
Terminal
Copy
- eas build
Building apps…
View your build progress at https://expo.dev/accounts/…
Android Play Store build
AAB
Status
Start time
Total time
Finished
Jan 11, 2023 7:07PM
8m 50s
iOS App Store build
IPA
Status
Start time
Total time
Finished
Jan 11, 2023 7:07PM
7m 44s

Free plan

The best way to start using EAS, no credit card required.

Free

Get Started

- ✓ 30 mobile app builds per month
- ✓ Submit to the app stores
- ✓ Send updates to 1,000 MAUs

On-demand plan

Popular

Pay as you go and scale flexibly.

Based on usage

View usage pricing

Select Plan

- ✓ Kick off builds and submissions faster with the high priority builds queue
- ✓ 2-hour build timeout

Production plan

For teams with growing user bases.

$99/month

+ additional usage

Select Plan

- ✓ $99 of build credit
- ✓ Send updates to 50,000 MAUs
- ✓ 2 build concurrencies

Enterprise plan

Advanced features for large apps and established teams.

$999/month

+ additional usage

Select Plan

- ✓ $999 of build credit
- ✓ Send updates to 1,000,000 MAUs
- ✓ 5 build concurrencies

Need more? Contact sales

Expo vs React Native CLI

Обидва підходи активно використовуються у **React Native**-спільноті, але їх популярність залежить від **типу проєкту**.

- **Expo** популярніший серед **початківців, стартапів та швидкої розробки**.
- **React Native CLI (Bare Workflow)** більше використовується **великою індустрією, корпораціями та складними проєктами**.

За статистикою npm (станом на початок 2025):

- `expo` має **близько 2.7 млн завантажень на тиждень**
- `react-native-cli` має **приблизно 2.1 млн завантажень на тиждень**

expo

54.0.32 • Public • Published 8 days ago

Readme | Code Beta | 21 Dependencies | 1741 Dependents | 827 Versions

expo

The `expo` package is a single package you can install in any React Native app to begin using Expo modules. API Reference.

- includes core infrastructure for Expo modules: `expo-modules-core` and `expo-modules-autolinking`.
- bundles a minimal set of Expo modules that are required by nearly every app, such as `expo-asset`.
- provides `@expo/cli`, a small CLI that provides a clean interface around both bundlers (such as Metro and Webpack) and native build tools (Xcode, Simulator.app, Android Studio, ADB, etc.), can generate native projects with `npx expo prebuild`, and aligns compatible package versions with `npx expo install`.
- exposes a JavaScript module that configures an app at runtime as needed to use `expo-font` and to function in Expo Go (optional, only if applicable).

See CONTRIBUTING for instructions on working on this package.

Keywords

expo

Install

```
> npm i expo
```

Repository

github.com/expo/expo

Homepage

github.com/expo/expo/tree/main/packages/expo

Weekly Downloads

2 700 942

Version	License
54.0.32	MIT

Unpacked Size	Total Files
870 kB	376

@react-native-community/cli

20.1.1 • Public • Published a day ago

Readme | Code Beta | 15 Dependencies | 112 Dependents | 330 Versions

React Native CLI

Command line tools to interact with React Native projects.

This package contains source code for `@react-native-community/cli`, the actual CLI that comes bundled with React Native. You don't need to install it separately in your project.

See the list of available commands.

Keywords

none

Install

```
> npm i @react-native-community/cli
```

Repository

github.com/react-native-community/cli

Homepage

github.com/react-native-community/cli/tree/main/packages/cli

Weekly Downloads

2 181 718

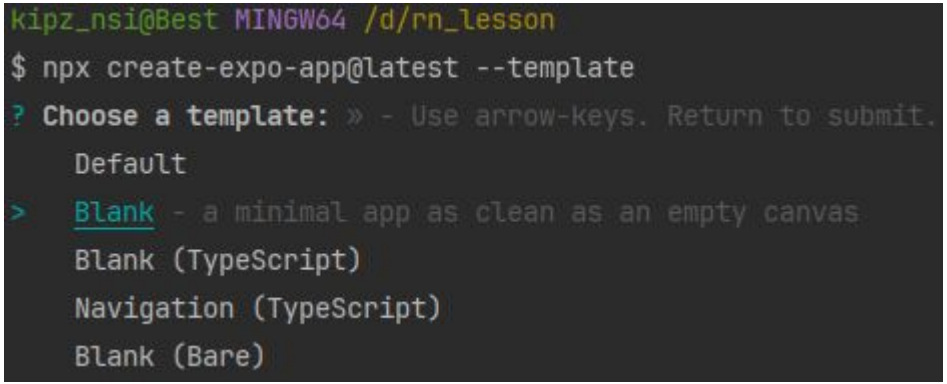
Version

20.1.1

License

MIT

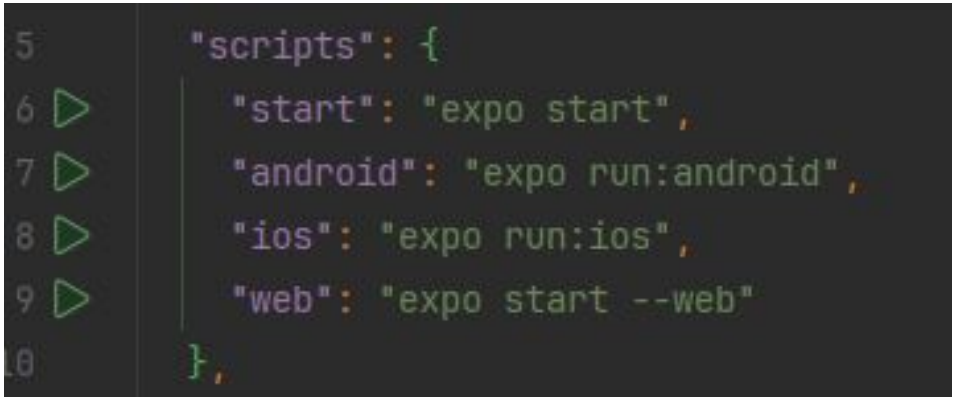
Створення нового додатку



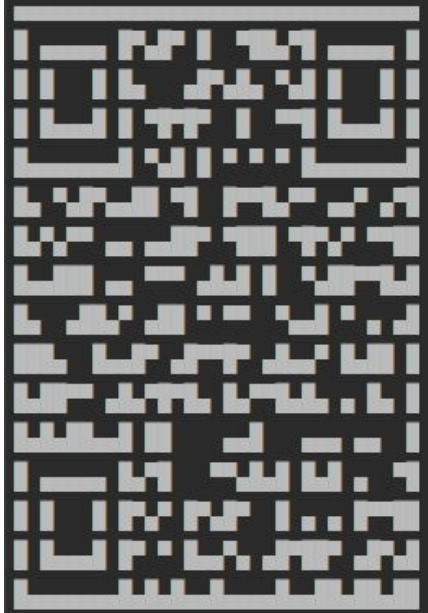
npx — це утиліта, яка входить до складу **Node.js (починаючи з версії 5.2.0 npm)** і використовується для **виконання npm-пакетів без їх глобальної установки**.

npx create-expo-app@latest --template

```
kipz_nsi@Best MINGW64 /d/rn_lesson
$ npx create-expo-app@latest --help

  Info
    Creates a new Expo project

  Usage
    $ npx create-expo-app <path> [options]

  Options
    -y, --yes             Use the default options for creating a project
        --no-install      Skip installing npm packages or CocoaPods
    -t, --template [pkg]  NPM template to use: default, blank, blank-typescript, tabs, bare-minimum. Default: default
    -e, --example [name]  Example name from https://github.com/expo/examples.
    -v, --version         Version number
    -h, --help            Usage info

    To choose a template pass in the --template arg:

    $ npx create-expo-app --template

    To choose an Expo example pass in the --example arg:

    $ npx create-expo-app --example
    $ npx create-expo-app --example with-router
```


"scripts": {
"start": "expo start",
"android": "expo run:android",
"ios": "expo run:ios",
"web": "expo start --web"
},

```
kipz_nsi@Best MINGW64 /d/rn_lesson/lesson (master)
$ npx expo start --tunnel
Starting project at D:\rn_lesson\lesson
Starting Metro Bundler
Tunnel connected.
Tunnel ready.
```



```
› Metro waiting on exp://tirqzkc-serhiineroda-8081.exp.direct
› Scan the QR code above with Expo Go (Android) or the Camera app (iOS)
```

Metro Bundler у React Native

Metro Bundler – це **швидкий JavaScript-бандлер**, який використовується в **React Native** для **обробки, трансформації та доставки коду** в мобільний додаток.
Metro Bundler **збирає весь JavaScript-код у єдиний файл**, який виконується на пристрої (емуляторі або реальному телефоні).

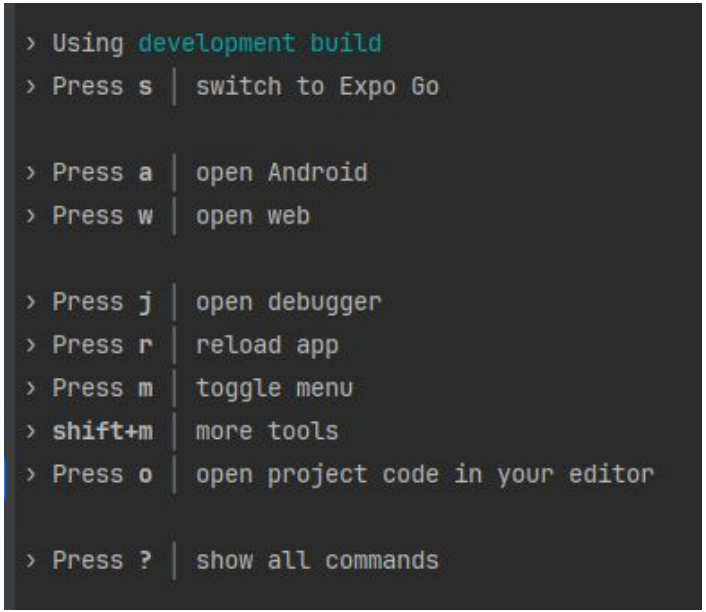
Як працює Metro Bundler?

- **1. Обробляє код** → конвертує сучасний JavaScript (ES6, JSX) у формат, який може виконати Android/iOS.
- **2. Оптимізує код** → прибирає зайве, мінімізує файли.
- **3. Створює бандл (`index.bundle`)** → це **єдиний файл** з усім кодом.
- **4. Доставляє код в емулятор/пристрій** → автоматично відправляє оновлення в Expo Go або React Native CLI.

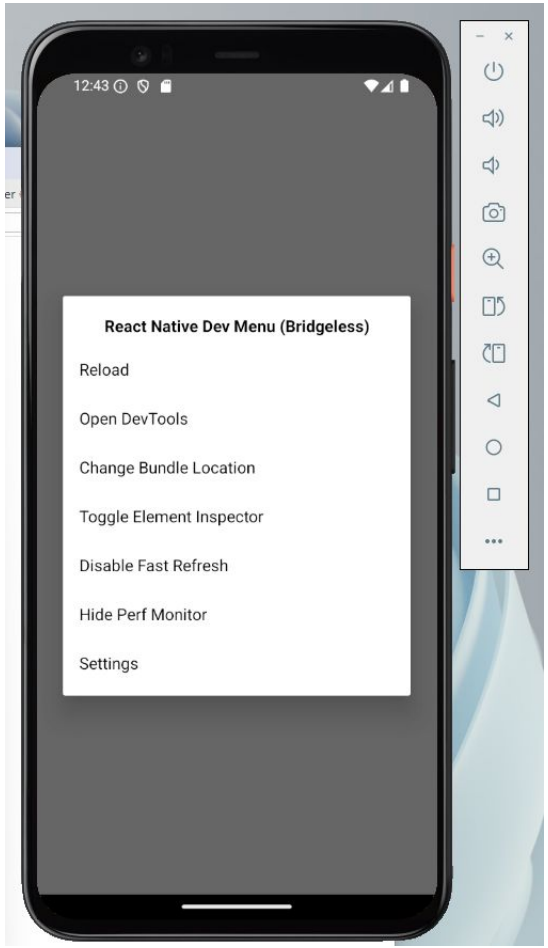
При кожному оновленні файлу Metro відправляє зміни на пристрій без перезапуску додатку.

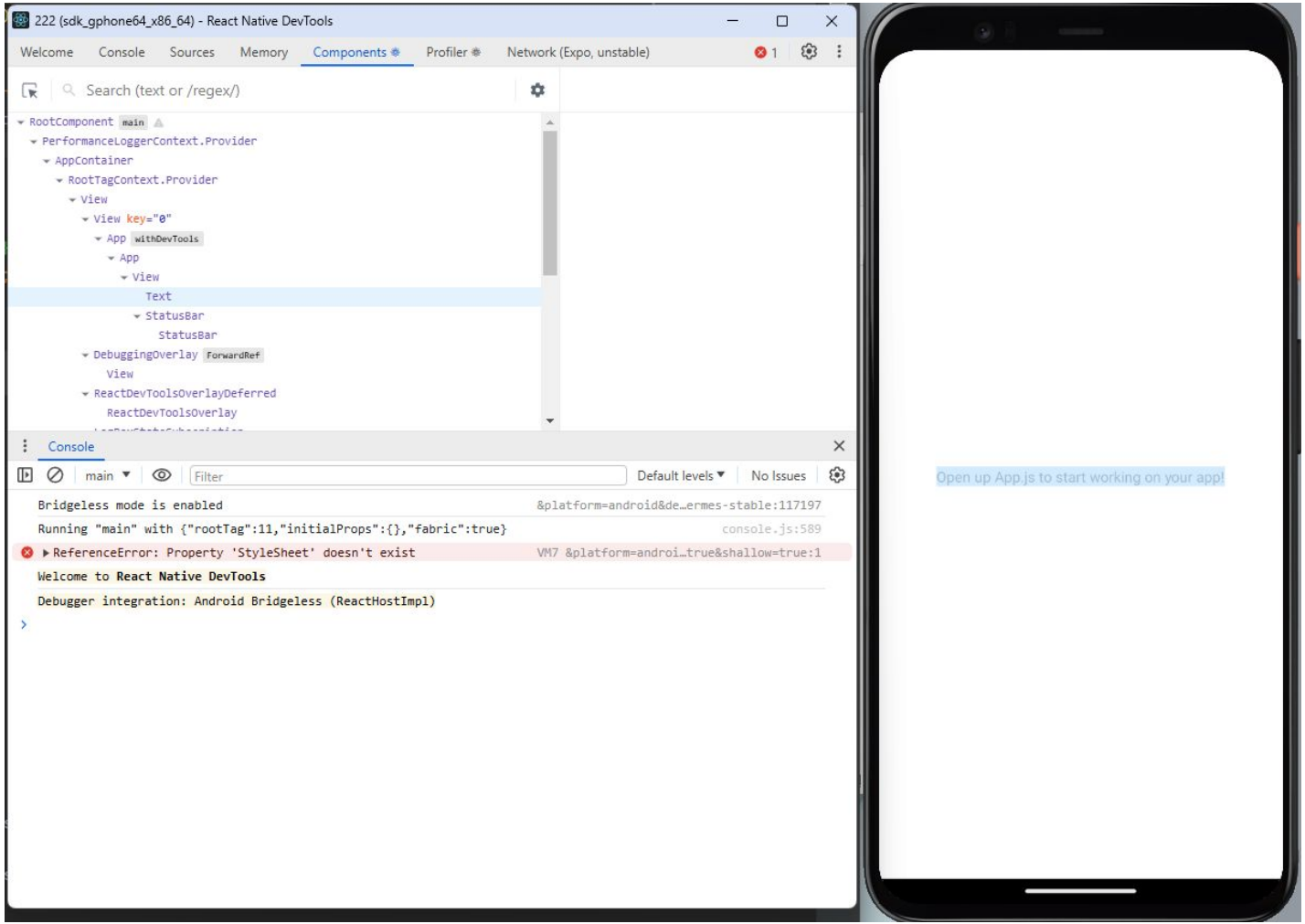
Дебагінг

Клавіша	Команда	Практичне застосування
`a`	**Open Android**	**Запуск:** миттєво відкриває додаток на емуляторі або смартфоні.
`s`	**Switch Mode**	**Гнучкість:** перемикання між стандартним *Expo Go* та кастомним *Dev Build*.
`w`	**Open Web**	**Швидкість:** запуск у браузері для перевірки верстки без навантаження на ПК.
`r`	**Reload**	**Синхронізація:** примусове оновлення коду, якщо Fast Refresh «завис».
`j`	**Debugger**	**Аналіз:** відкриває Chrome DevTools для перегляду `console.log` та помилок.
`m`	**Dev Menu**	**Інспекція:** відкриває меню *на телефоні* (Inspector, Perf Monitor).
`o`	**Open Code**	**Зручність:** шорткат для миттєвого відкриття проєкту у VS Code.

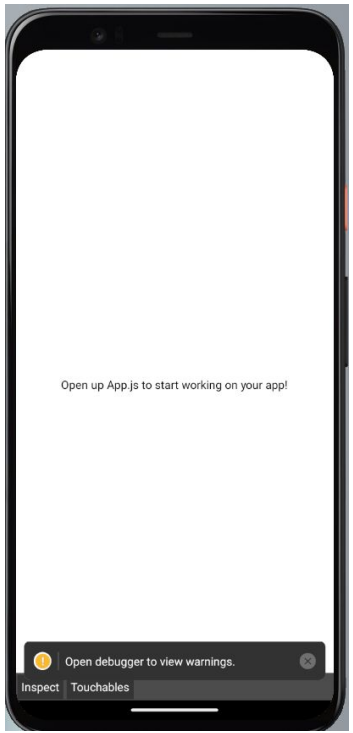


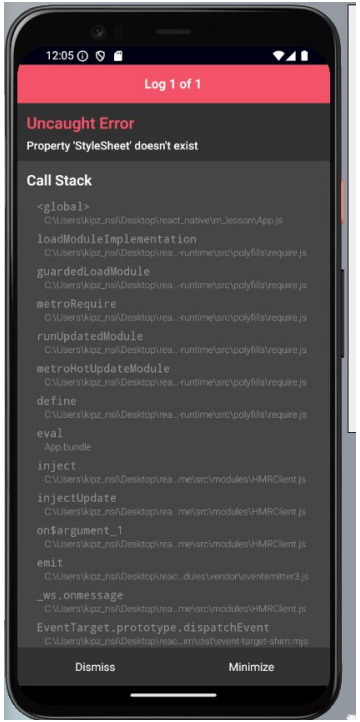
```
› Press m | toggle menu
```

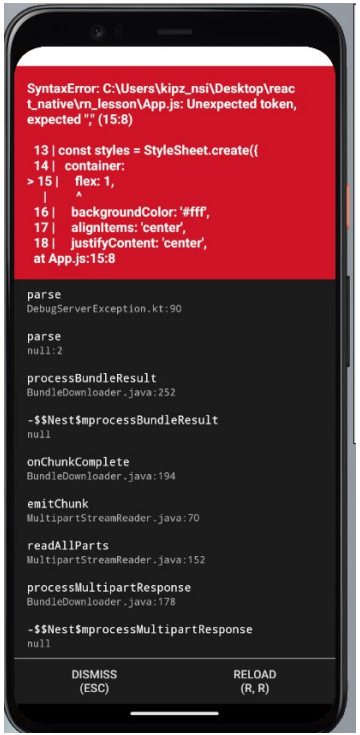



222 (sdk_gphone64_x86_64) - React Native DevTools
Welcome
Console
Sources
Memory
Components
Profiler
Network (Expo, unstable)
Search (text or /regex/)
RootComponent main
PerformanceLoggerContext.Provider
AppContainer
RootTagContext.Provider
View
View key="0"
App withDevTools
App
View
Text
StatusBar
StatusBar
DebuggingOverlay ForwardRef
View
ReactDevToolsOverlayDeferred
ReactDevToolsOverlay
Console
main
Filter
Default levels
No Issues
Bridgeless mode is enabled
&platform=android&de…ermes-stable:117197
Running "main" with {"rootTag":11,"initialProps":{},"fabric":true}
console.js:589
ReferenceError: Property 'StyleSheet' doesn't exist
VM7 &platform=androi…true&shallow=true:1
Welcome to React Native DevTools
Debugger integration: Android Bridgeless (ReactHostImpl)
Open up App.js to start working on your app!

LogBox


Open up App.js to start working on your app!
Open debugger to view warnings.
Inspect
Touchables


12:05
Log 1 of 1
Uncaught Error
Property 'StyleSheet' doesn't exist
Call Stack
<global>
C:\Users\kipz_nsi\Desktop\react_native\rn_lesson\App.js
loadModuleImplementation
C:\Users\kipz_nsi\Desktop\rea...-runtime\src\polyfills\require.js
guardedLoadModule
C:\Users\kipz_nsi\Desktop\rea...-runtime\src\polyfills\require.js
metroRequire
C:\Users\kipz_nsi\Desktop\rea...-runtime\src\polyfills\require.js
runUpdatedModule
C:\Users\kipz_nsi\Desktop\rea...-runtime\src\polyfills\require.js
metroHotUpdateModule
C:\Users\kipz_nsi\Desktop\rea...-runtime\src\polyfills\require.js
define
C:\Users\kipz_nsi\Desktop\rea...-runtime\src\polyfills\require.js
eval
App.bundle
inject
C:\Users\kipz_nsi\Desktop\rea...me\src\vmodules\HMRClient.js
injectUpdate
C:\Users\kipz_nsi\Desktop\rea...me\src\vmodules\HMRClient.js
on$argument_1
C:\Users\kipz_nsi\Desktop\rea...me\src\vmodules\HMRClient.js
emit
C:\Users\kipz_nsi\Desktop\reac...dules\vendor\eventemitter3.js
_ws.onmessage
C:\Users\kipz_nsi\Desktop\rea...me\src\vmodules\HMRClient.js
EventTarget.prototype.dispatchEvent
C:\Users\kipz_nsi\Desktop\reac...im\dist\event-target-shim.mjs
Dismiss
Minimize


SyntaxError: C:\Users\kipz_nsi\Desktop\react_native\rn_lesson\App.js: Unexpected token, expected "," (15:8)
13 | const styles = StyleSheet.create({
14 | container:
> 15 | flex: 1,
| ^
16 | backgroundColor: '#fff',
17 | alignItems: 'center',
18 | justifyContent: 'center',
at App.js:15:8
parse
DebugServerException.kt:90
parse
null:2
processBundleResult
BundleDownloader.java:252
-$$Nest$mprocessBundleResult
null
onChunkComplete
BundleDownloader.java:194
emitChunk
MultipartStreamReader.java:70
readAllParts
MultipartStreamReader.java:152
processMultipartResponse
BundleDownloader.java:178
-$$Nest$mprocessMultipartResponse
null
DISMISS
(ESC)
RELOAD
(R, R)

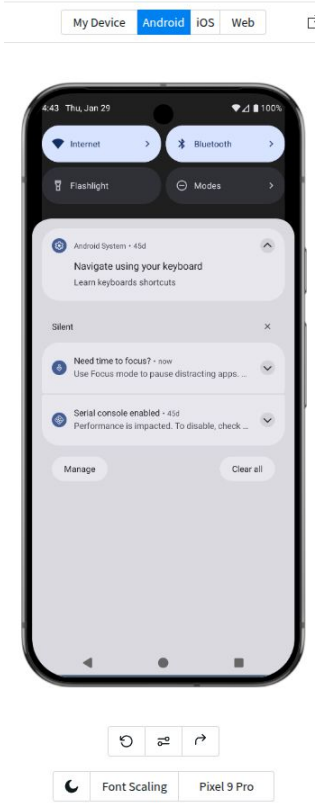
Чистка кешу

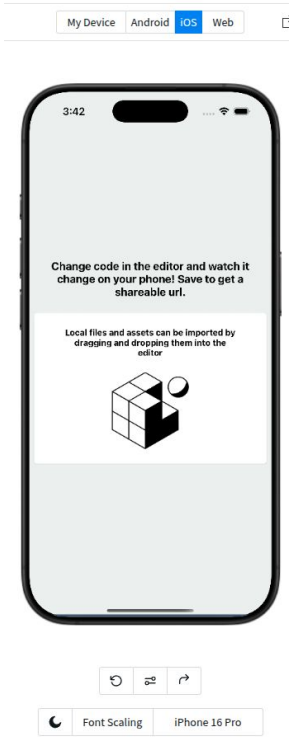
```
npx expo start --clear
```

Expo Snack: Хмарне середовище розробки

Expo Snack

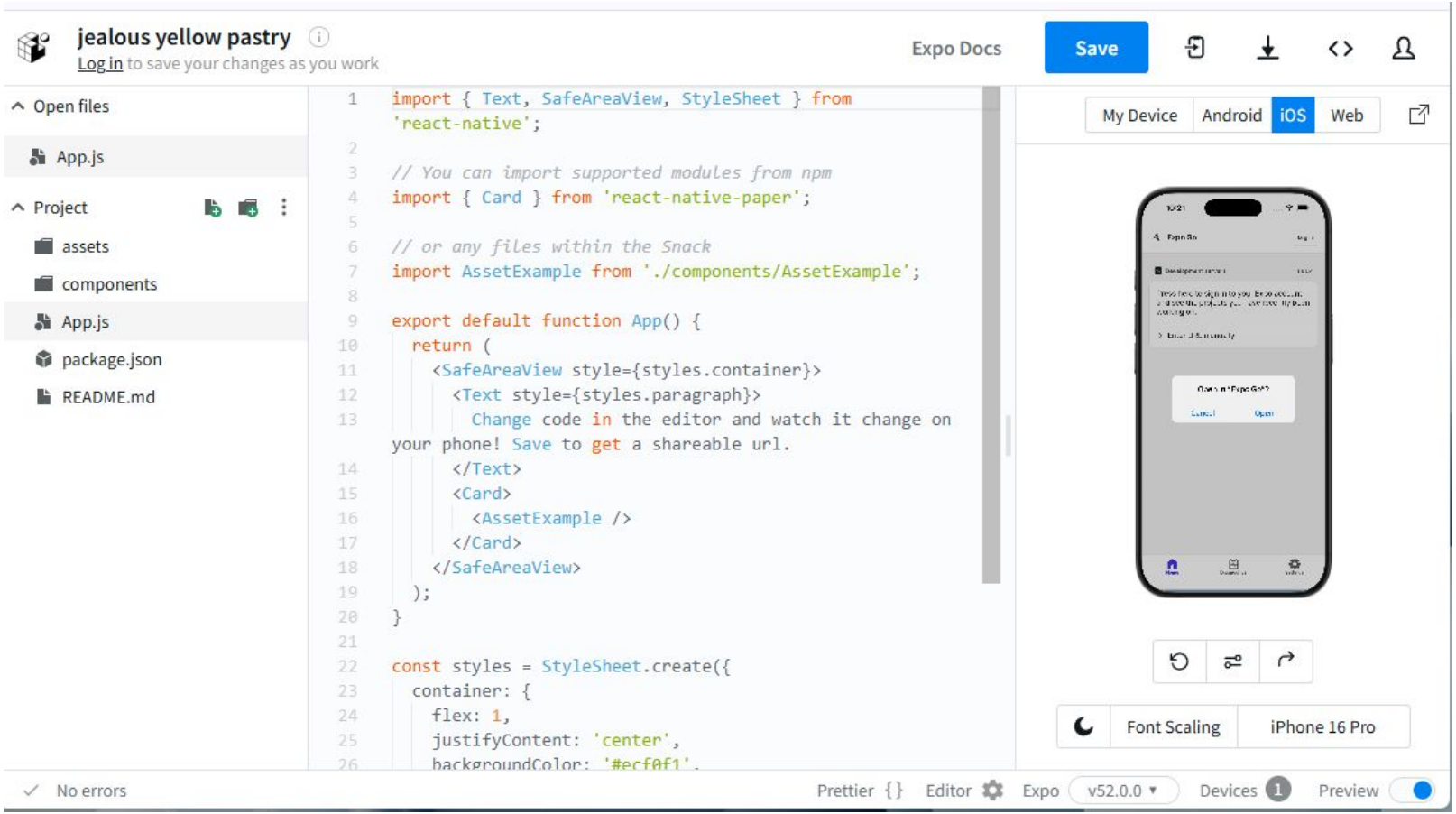
Expo Snack — це інтерактивна веб-платформа (пісочниця), яка дозволяє проектувати, програмувати та запускати мобільні застосунки на React Native безпосередньо у браузері, без локального встановлення SDK.



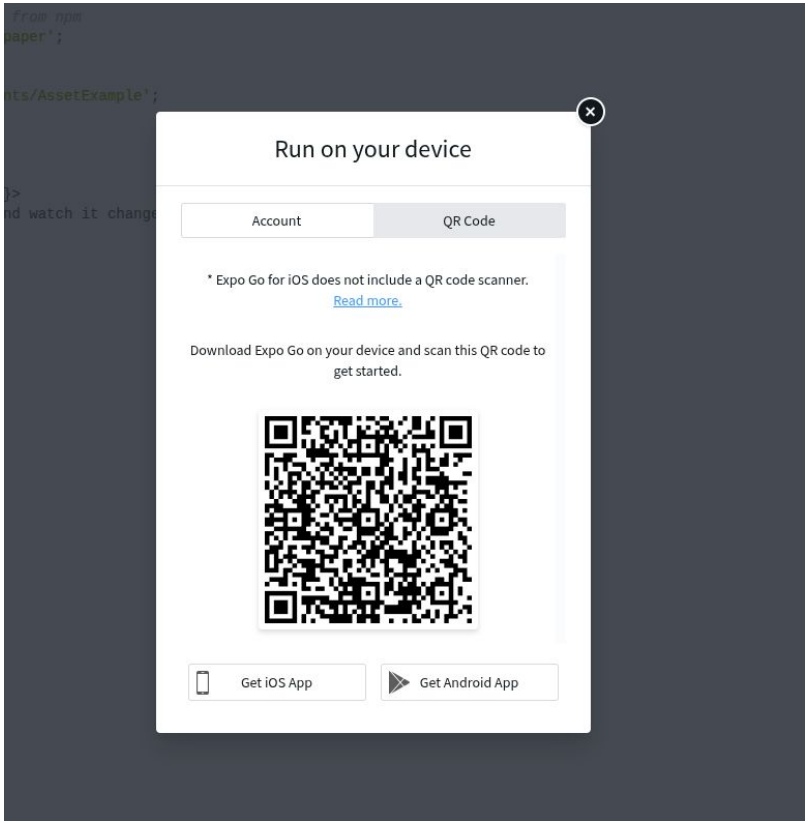


Ключові технологічні можливості:

- **Zero-Config Setup:** Миттєвий старт розробки без налаштування Android Studio, Xcode або Node.js на локальній машині.
- **Live Preview:** Синхронна візуалізація інтерфейсу на трьох платформах одночасно: **Android**, **iOS** та **Web**.
- **Hot Reloading:** Внесені зміни в коді відображаються миттєво
- **Cloud-based Dependencies:** Підтримка бібліотек з екосистеми Expo SDK.


jealous yellow pastry
Log in to save your changes as you work
Expo Docs
Save
Open files
App.js
Project
assets
components
App.js
package.json
README.md
1 import { Text, SafeAreaView, StyleSheet } from 'react-native';
2
3 // You can import supported modules from npm
4 import { Card } from 'react-native-paper';
5
6 // or any files within the Snack
7 import AssetExample from './components/AssetExample';
8
9 export default function App() {
10 return (
11 <SafeAreaView style={styles.container}>
12 <Text style={styles.paragraph}>
13 Change code in the editor and watch it change on your phone! Save to get a shareable url.
14 </Text>
15 <Card>
16 <AssetExample />
17 </Card>
18 </SafeAreaView>
19);
20 }
21
22 const styles = StyleSheet.create({
23 container: {
24 flex: 1,
25 justifyContent: 'center',
26 backgroundColor: '#ecf0f1',
My Device
Android
iOS
Web
Font Scaling
iPhone 16 Pro
No errors
Prettier
Editor
Expo
v52.0.0
Devices
1
Preview

Run on your device



Взаємодія **Expo CLI** та **Android Studio**

Коли ви натискаєте клавішу **"a"** у терміналі під час роботи Expo або обираєте **"Run on Android Device/Emulator"** у Snack, відбувається наступне:

- **Expo CLI** звертається до змінної оточення `ANDROID_HOME`.
- Через цю змінну він знаходить шлях до папки `emulator` та `platform-tools`.
- Система виконує пошук активних віртуальних пристроїв (AVD) та запускає інструмент **adb** (Android Debug Bridge).

Як перевірити готовність?

Перед запуском проекту виконайте команду: `npx expo doctor` Вона перевірить, чи бачить Expo ваш Android SDK та чи готові змінні середовища до роботи

Змінні середовища

`ANDROID_HOME`

Ця змінна вказує шлях до кореневої папки Android SDK.

- **Шлях за замовчуванням (Windows):** `C:\Users\ВАШ_КОРИСТУВАЧ\AppData\Local\Android\Sdk`
- **Як знайти:** Відкрийте Android Studio → Settings → Languages & Frameworks → Android SDK. Там буде вказано «Android SDK Location».

Мало просто вказати на SDK, треба «зареєструвати» конкретні утиліти в системі, щоб команда `adb` або `emulator` працювала в будь-якій папці.

Потрібно додати такі шляхи (відносно вашого SDK):

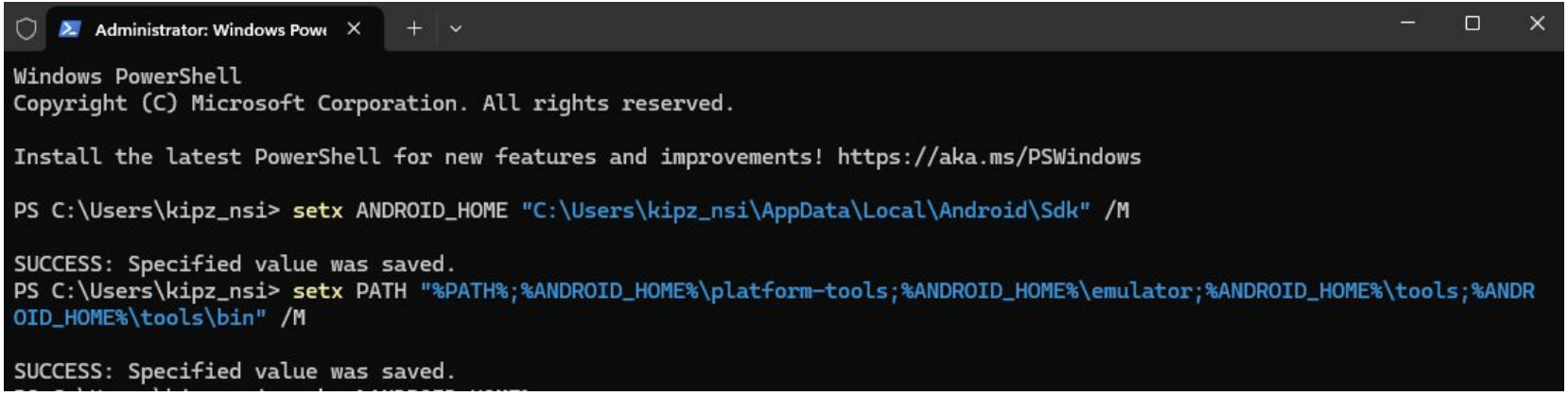
1. `%ANDROID_HOME%\platform-tools` — тут знаходиться **adb** (найважливіша утиліта для зв'язку з телефоном).
2. `%ANDROID_HOME%\emulator` — для запуску віртуальних пристроїв.
3. `%ANDROID_HOME%\tools` та `%ANDROID_HOME%\tools\bin` — додаткові інструменти збірки.

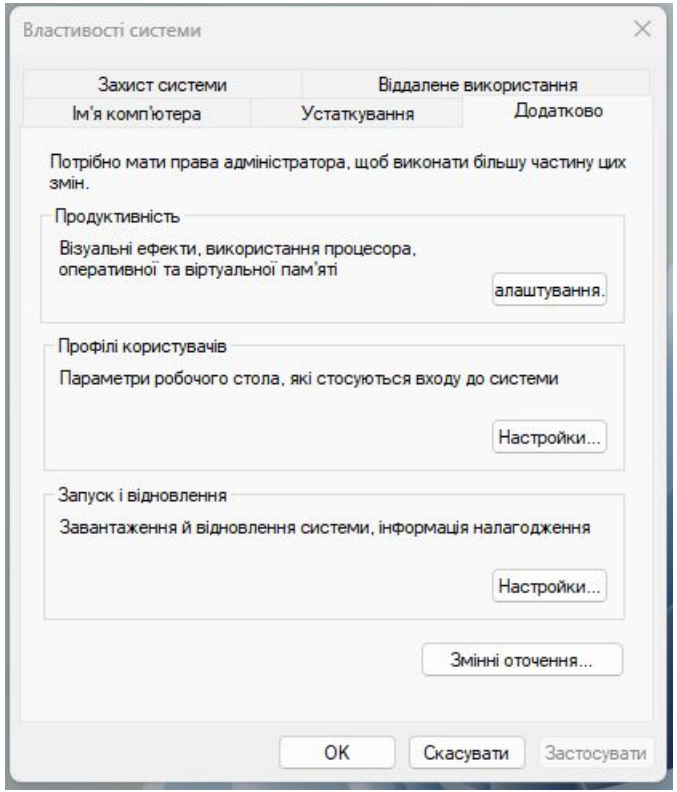
Змінні середовища

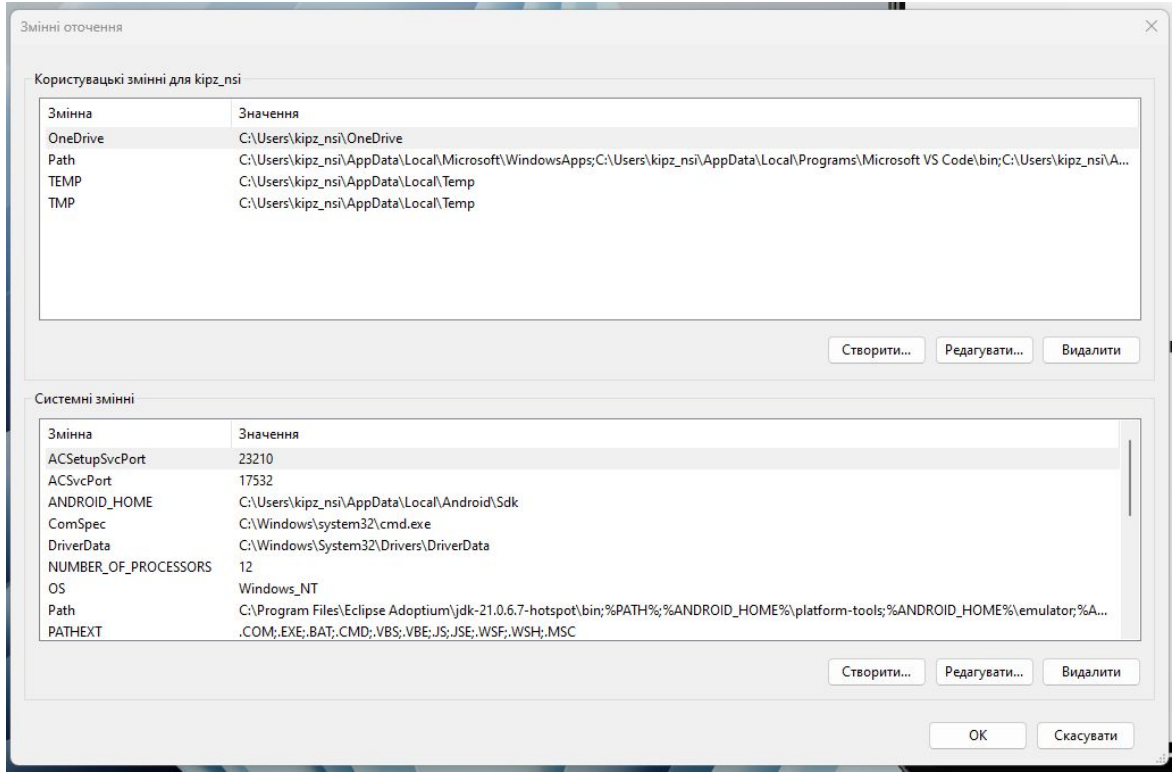
Відкрийте **Командний рядок (cmd)** або **PowerShell** (запустіть від імені адміністратора).

setx ANDROID_HOME "C:\Users\ВАШ_КОРИСТУВАЧ\AppData\Local\Android\Sdk"
setx PATH "%PATH%;%ANDROID_HOME%\platform-tools;%ANDROID_HOME%\emulator;%ANDROID_HOME%\tools;%ANDROID_HOME%\tools\bin"

Перезапустіть **cmd** або комп'ютер, щоб зміни вступили в силу.




Властивості системи
Захист системи
Віддалене використання
Ім'я комп'ютера
Устаткування
Додатково
Потрібно мати права адміністратора, щоб виконати більшу частину цих змін.
Продуктивність
Візуальні ефекти, використання процесора, оперативної та віртуальної пам'яті
алаштування.
Профілі користувачів
Параметри робочого стола, які стосуються входу до системи
Настройки...
Запуск і відновлення
Завантаження й відновлення системи, інформація налагодження
Настройки...
Змінні оточення...
OK
Скасувати
Застосувати


Змінні оточення
Користувацькі змінні для kipz_nsi
Змінна
Значення
OneDrive
C:\Users\kipz_nsi\OneDrive
Path
C:\Users\kipz_nsi\AppData\Local\Microsoft\WindowsApps;C:\Users\kipz_nsi\AppData\Local\Programs\Microsoft VS Code\bin;C:\Users\kipz_nsi\A...
TEMP
C:\Users\kipz_nsi\AppData\Local\Temp
TMP
C:\Users\kipz_nsi\AppData\Local\Temp
Створити...
Редагувати...
Видалити
Системні змінні
Змінна
Значення
ACSetupSvcPort
23210
ACSvcPort
17532
ANDROID_HOME
C:\Users\kipz_nsi\AppData\Local\Android\Sdk
ComSpec
C:\Windows\system32\cmd.exe
DriverData
C:\Windows\System32\Drivers\DriverData
NUMBER_OF_PROCESSORS
12
OS
Windows_NT
Path
C:\Program Files\Eclipse Adoptium\jdk-21.0.6.7-hotspot\bin;%PATH%;%ANDROID_HOME%\platform-tools;%ANDROID_HOME%\emulator;%A...
PATHEXT
.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
Створити...
Редагувати...
Видалити
OK
Скасувати