MAUI 프로젝트를 배포하는 방법은 대상 플랫폼에 따라 다릅니다. 각 플랫폼(Android, iOS, Windows)에 맞는 앱 패키지를 빌드하고 생성한 다음, 앱 스토어에 제출하거나 직접 배포합니다.

**1. 배포 전 필수 확인 사항**

* **릴리스(Release) 모드**: 배포를 위해 빌드 모드를 \*\*Debug\*\*에서 \*\*Release\*\*로 변경하세요. 릴리스 모드는 앱 성능을 최적화하고 패키지 크기를 줄입니다.
* **앱 정보**: .csproj 파일이나 프로젝트 속성에서 앱 이름, 버전, ID 등을 설정하세요.
* **앱 서명**: 앱을 스토어에 제출하거나 배포하려면 디지털 서명이 필요합니다. 플랫폼별로 서명 방법을 준비해야 합니다.
  + **Android**: 키스토어(keystore) 파일을 생성하고 .csproj 파일에 등록해야 합니다.
  + **iOS**: Apple 개발자 계정으로 서명 인증서와 프로비저닝 프로파일을 발급받아야 합니다.

**2. Visual Studio를 이용한 배포**

Visual Studio는 각 플랫폼에 맞는 앱 패키지를 쉽게 생성할 수 있는 기능을 제공합니다.

**안드로이드 (Android)**

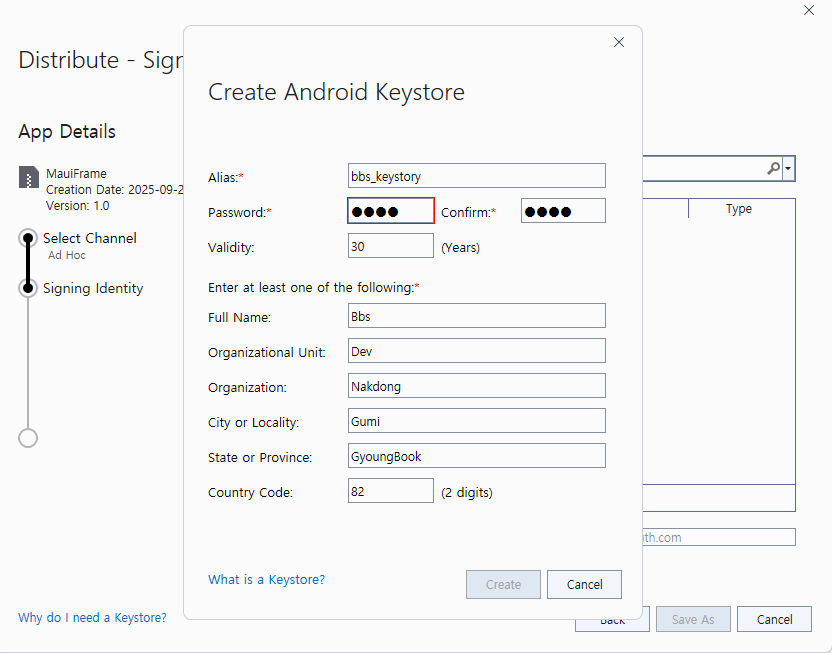
1. 솔루션 탐색기에서 MAUI 프로젝트를 마우스 오른쪽 버튼으로 클릭하고 \*\*게시...\*\*를 선택합니다.
2. 대상 플랫폼으로 \*\*안드로이드\*\*를 선택합니다.
3. 배포 채널을 선택합니다.
   * **임시**: 직접 배포할 .apk 파일 또는 .aab 파일을 생성합니다.
   * **Google Play**: 구글 플레이스토어에 제출할 .aab 파일을 생성합니다.
4. 키스토어를 생성하거나 기존 키스토어를 가져와 앱에 서명합니다.
5. 게시가 완료되면 프로젝트의 bin\Release 폴더에 최종 패키지 파일이 생성됩니다.

**iOS**

1. MAUI 프로젝트를 마우스 오른쪽 버튼으로 클릭하고 \*\*게시...\*\*를 선택합니다.
2. 대상 플랫폼으로 \*\*iOS\*\*를 선택합니다.
3. 배포 채널을 선택합니다.
   * **임시**: 테스트용으로 배포할 .ipa 파일을 생성합니다.
   * **App Store**: 앱스토어에 제출할 .ipa 파일을 생성합니다.
4. 유효한 Apple 개발자 계정과 서명 인증서가 필요합니다. Visual Studio에서 자동 프로비저닝을 통해 필요한 파일을 생성할 수 있습니다.
5. 게시가 완료되면 .ipa 파일이 생성되며, 이 파일을 App Store Connect 포털에 업로드하여 제출할 수 있습니다.

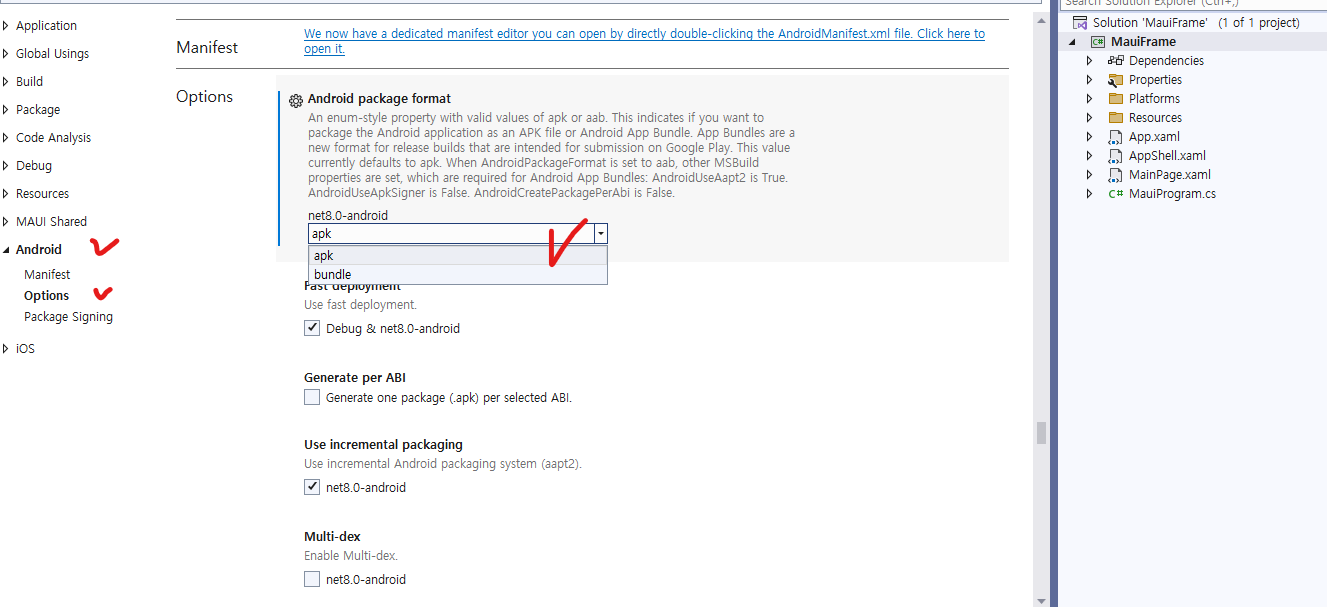
**윈도우 (Windows)**

1. MAUI 프로젝트를 마우스 오른쪽 버튼으로 클릭하고 \*\*게시...\*\*를 선택합니다.
2. 대상 플랫폼으로 \*\*Windows\*\*를 선택합니다.
3. \*\*사이드로딩(Sideloading)\*\*을 선택하여 로컬에서 직접 배포할 .msix 또는 .msixbundle 패키지를 생성합니다.
4. 또는 \*\*Microsoft Store\*\*를 선택하여 마이크로소프트 스토어에 제출할 패키지를 생성합니다.
5. 생성된 패키지를 사용자에게 배포하거나 마이크로소프트 스토어에 제출하여 게시할 수 있습니다.



비번: 123456

기본값 Bundle : aab 파일



네, MAUI 안드로이드 앱은 .apk 파일로 배포가 가능합니다. 하지만 **구글 플레이 스토어에 업로드할 때는 .aab (Android App Bundle) 형식으로 제출하는 것이 필수적**입니다.

**.apk와 .aab의 차이점**

* **.apk (Android Package)**: 단일 파일 안에 앱의 모든 리소스(코드, 이미지, 레이아웃 등)가 포함되어 있습니다. 과거에는 모든 사용자에게 동일한 .apk 파일을 배포했습니다.
* **.aab (Android App Bundle)**: 앱의 모든 코드를 담고 있지만, 특정 기기에 필요한 리소스(예: 특정 CPU 아키텍처나 언어에 맞는 리소스)를 포함하지 않습니다. 대신, 플레이 스토어가 사용자의 기기에 최적화된 .apk 파일을 동적으로 생성하여 제공합니다.

**.apk 배포의 장단점**

**장점**:

* **직접 배포**: 플레이 스토어 없이 사용자에게 직접 파일을 전달하여 설치하게 할 수 있습니다. (사이드로딩)
* **간단한 테스트**: 개발 중인 앱을 동료나 테스터에게 쉽게 전달하여 테스트할 수 있습니다.

**단점**:

* **비효율적**: 단일 apk 파일에 모든 기기 아키텍처(arm64, x86 등)의 코드가 포함되어 있어 파일 크기가 큽니다.
* **플레이 스토어 업로드 불가**: **구글 플레이 개발자 콘솔은 2021년 8월부터 .aab 파일만 업로드를 허용합니다.**

**MAUI에서 .apk 파일 생성 방법**

Visual Studio를 사용하면 .aab와 .apk 파일 모두 생성할 수 있습니다.

1. **릴리스 모드 빌드**: Visual Studio 상단 메뉴에서 Debug를 \*\*Release\*\*로 변경하세요.
2. **게시 마법사 실행**: 솔루션 탐색기에서 안드로이드 프로젝트를 마우스 오른쪽 버튼으로 클릭하고 \*\*게시...\*\*를 선택합니다.
3. **임시(Ad Hoc) 배포 선택**: 마법사에서 **임시(Ad Hoc)** 채널을 선택합니다.
4. **apk 파일 생성 확인**: 마법사를 완료하면 .aab 파일이 생성되지만, **릴리스 빌드** 설정에서 .apk 파일도 함께 생성하도록 구성할 수 있습니다.

**결론**

.apk 파일로의 배포는 테스트나 사내 앱처럼 **플레이 스토어를 통하지 않는 경우에만 사용**해야 합니다. 구글 플레이 스토어에 앱을 제출할 계획이라면 .aab 파일을 사용하는 것이 유일한 방법입니다.

To publish a MAUI project, you need to create an application package for the specific platform you are targeting (Android, iOS, macOS, or Windows). This process involves configuring project settings, building the application, and generating the final package, which can then be distributed to app stores or directly to users.

**1. Prerequisite Checks**

Before you begin, ensure your project is ready for publishing:

* **Configuration:** Switch your build configuration to **Release**. This optimizes your app for performance and reduces the package size.
* **App Information:** Fill in your application's details in the **MauiApp.csproj** file or through the project properties. This includes the app name, version number, and application ID.
* **Signing:** You must digitally sign your application to verify its authenticity.
  + **Android:** Create a keystore and configure it in your csproj file or Visual Studio's Android options.
  + **iOS/macOS:** You need an Apple Developer account to create a signing certificate and provisioning profile.

**2. Publishing from Visual Studio**

Visual Studio provides a streamlined interface to publish your app.

**For Android:**

1. In Solution Explorer, right-click on your MAUI project and select **Publish...**.
2. Choose **Android** as the target platform.
3. Select **Ad Hoc** for direct distribution or **Google Play** to publish to the Google Play Store.
4. Visual Studio will guide you through creating or importing a signing keystore.
5. After the process is complete, a .aab (Android App Bundle) or .apk file will be generated in your project's bin\Release folder. The .aab format is preferred for Google Play.

**For iOS:**

1. Right-click your MAUI project and select **Publish...**.
2. Choose **iOS**.
3. Select **Ad Hoc** or **App Store**.
4. You will need a valid Apple Developer account and a configured signing identity. Visual Studio can help you automatically provision these if you are signed into your Apple account.
5. After publishing, a .ipa file will be generated. You can submit this file to the App Store Connect portal.

**For Windows:**

1. Right-click your MAUI project and select **Publish...**.
2. Choose **Windows**.
3. Select the **Sideloading** option for local distribution or the **Microsoft Store** option.
4. Follow the prompts to create the .msix or .msixbundle package.
5. The generated package can be distributed to users or submitted to the Microsoft Store.