**Walkthrough: CI/CD for .NET MAUI Android App Deployment to Google Play Store**

1. Prerequisites:

* A .NET MAUI Android app hosted on GitHub.
* A Google Play Developer account.
* A signing key for your Android app.

2. Setting up Secrets:

In your GitHub repository, go to "Settings" > "Secrets" and add the following secrets:

* **GOOGLE\_PLAY\_JSON\_KEY**: JSON key for your Google Play Developer account.
* **KEYSTORE\_BASE64**: Base64 encoded keystore file.
* **KEYSTORE\_PASSWORD**: Password for the keystore.
* **KEY\_ALIAS**: Alias of the key inside the keystore.
* **KEY\_PASSWORD**: Password for the key.

3. GitHub Actions Workflow:

Create a new file **.github/workflows/android.yml** with the following content:

|  |
| --- |
| name: Android Deployment  on:  push:  branches:  - main  jobs:  build:  runs-on: ubuntu-latest  steps:  - name: Checkout Repository  uses: actions/checkout@v2  - name: Setup .NET SDK  uses: actions/setup-dotnet@v1  with:  dotnet-version: '6.0.x'  - name: Install .NET MAUI Workloads  run: dotnet workload install maui  - name: Build and Publish APK  run: dotnet build -t:SignAndroidPackage  - name: Set up JDK  uses: actions/setup-java@v2  with:  distribution: 'adopt'  - name: Sign APK  run: |  jarsigner -keystore ${{ secrets.KEYSTORE\_BASE64 }} -storepass ${{ secrets.KEYSTORE\_PASSWORD }} -keypass ${{ secrets.KEY\_PASSWORD }} -signedjar ./bin/Release/net6.0-android/bin/YourApp.Android-Signed.apk ./bin/Release/net6.0-android/bin/YourApp.Android.apk ${{ secrets.KEY\_ALIAS }}  - name: Upload Release to GitHub Releases  uses: softprops/action-gh-release@v1  with:  files: ./bin/Release/net6.0-android/bin/YourApp.Android-Signed.apk  deploy:  runs-on: ubuntu-latest  needs: build  steps:  - name: Download Signed APK  uses: actions/download-artifact@v2  with:  name: android-apk  path: .  - name: Deploy to Google Play  uses: wzieba/FakeGooglePlay@v1  with:  serviceAccountKey: ${{ secrets.GOOGLE\_PLAY\_JSON\_KEY }}  packageName: 'com.yourcompany.yourapp'  versionCode: ${{ github.run\_number }}  track: 'production'  userFraction: '1.0' |

4. Explanation:

* The workflow is triggered on every push to the **main** branch.
* It has two jobs: **build** and **deploy**.
* The **build** job builds and signs the Android APK using the specified keystore.
* The **deploy** job downloads the signed APK and deploys it to the Google Play Store using the **FakeGooglePlay** action.

5. Additional Considerations:

* Ensure that your .NET MAUI project file is configured for Android deployment.
* Make sure your keystore file is Base64 encoded and added as a secret in GitHub.
* Adapt package names, file paths, and other configurations according to your project.

6. Conclusion:

With this setup, each push to the main branch triggers a build and deployment process to the Google Play Store, ensuring a streamlined CI/CD pipeline for your .NET MAUI Android app.

Remember to test the workflow thoroughly before deploying to production.

Reference links

<https://learn.microsoft.com/en-gb/dotnet/maui/android/deployment/?view=net-maui-8.0>

<https://learn.microsoft.com/en-gb/dotnet/maui/android/deployment/publish-cli?view=net-maui-8.0>

<https://developer.android.com/studio/publish/upload-bundle>