doTech
SOFTWARE DEVELOPMENT FORCE

Documentación para hacer un apk en .Net MAUI

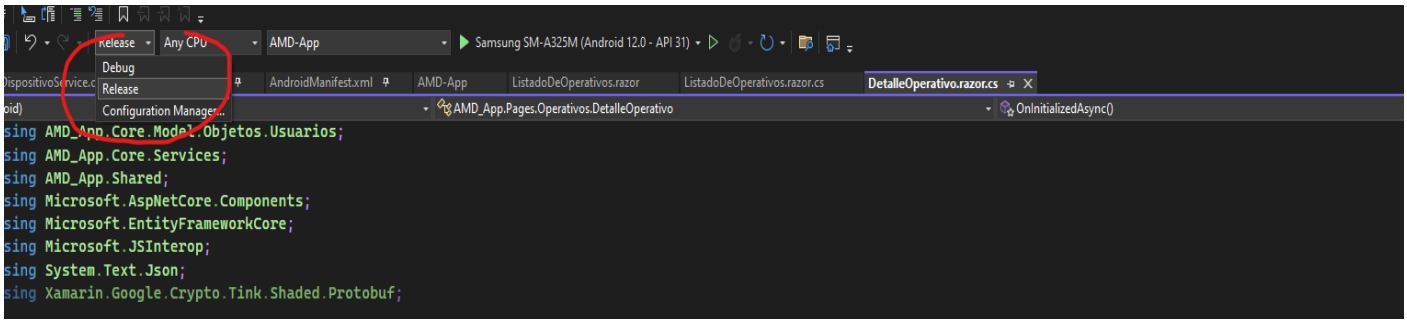
AMD Blazor Hybrid

`

Esta doc. fue escrita el 19/02/2024, con la versión 17.9.0 de Visual Studio y con .Net 8.0

Pasos a seguir para hacer un apk en .Net MAUI:

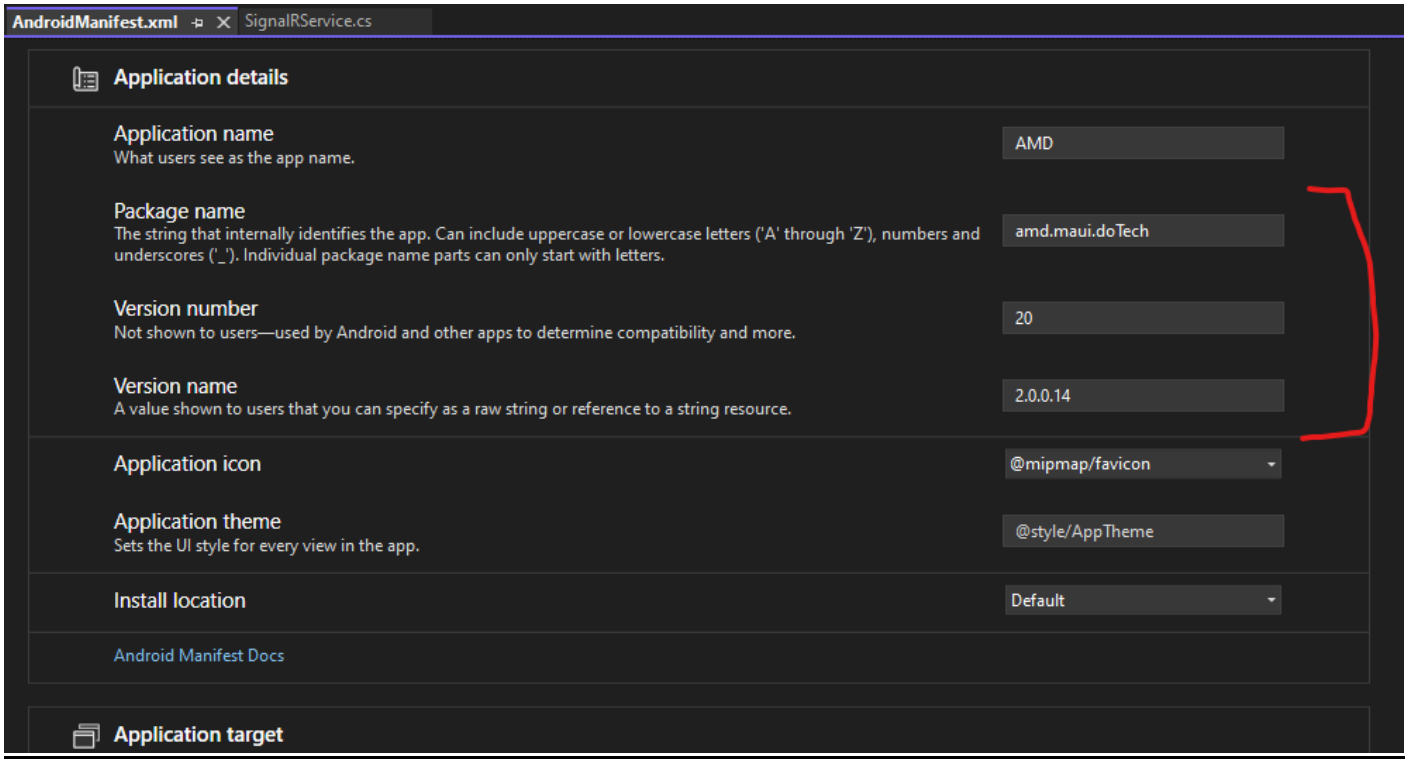
1) Poner el proyecto en modo "Release".



2) Abrir el AndroidManifest.xml ubicado en ./Platforms/Android y actualizar los siguientes parámetros:

- **Version number** (importantísimo para que el apk no se crashee despues al instalarlo por errores de compatibilidad)
- **Version name**

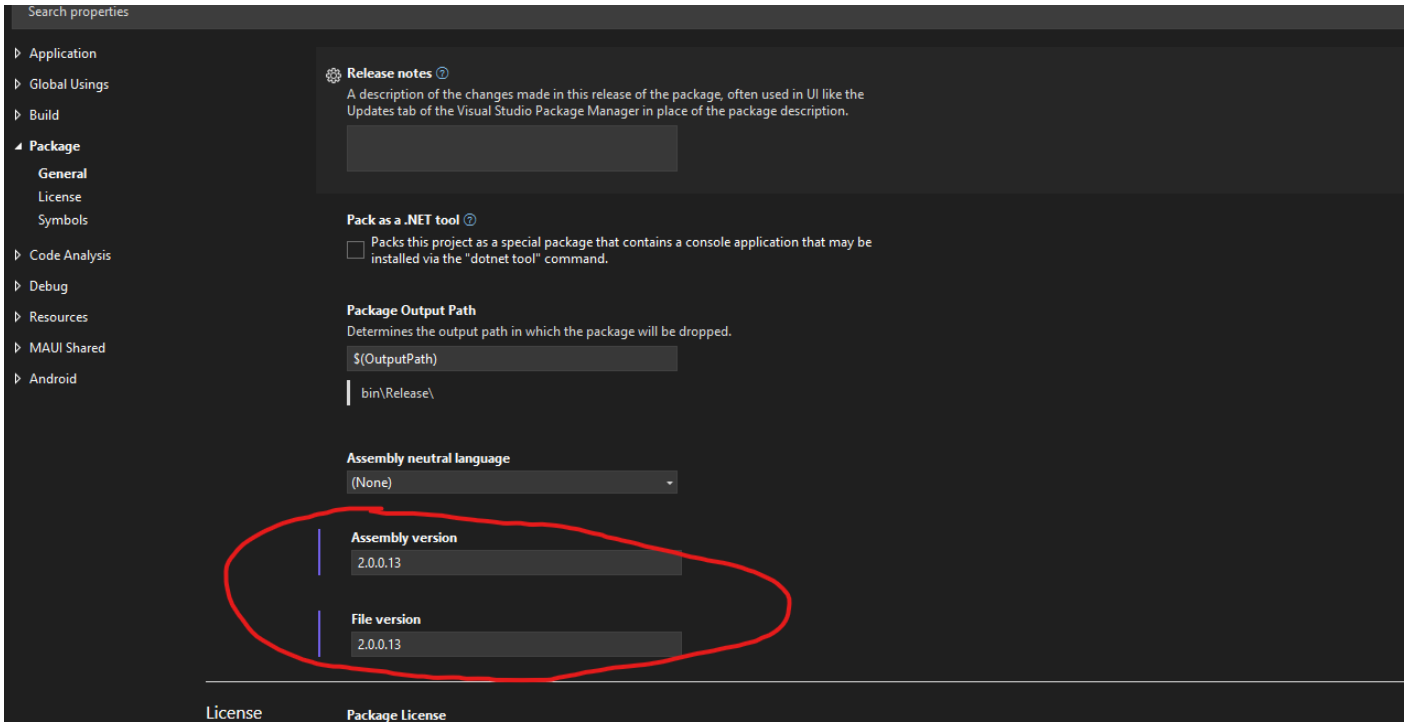
(Además, corroborar que el Package Name esté correcto. Debe decir: amd.maui.doTech, mientras que el Application name debe decir: AMD).



3) Abrir las propiedades del proyecto en el Visual Studio, y actualizar los siguientes parámetros:

- En la sección “Package/General”:

- **Assembly version** (debe coincidir con el Version name especificado en el AndroidManifest.xml).
- **File version** (debe coincidir con el Version name especificado en el AndroidManifest.xml).



- En la sección “MAUI Shared/General”:

- **Application Display Version** (debe coincidir con el Version name especificado en el AndroidManifest.xml).
- **Application Version** (debe coincidir con el Version number especificado en el AndroidManifest.xml).

(Además, corroborar que el Application ID esté correcto. Debe coincidir con el Package name del AndroidManifest.xml).

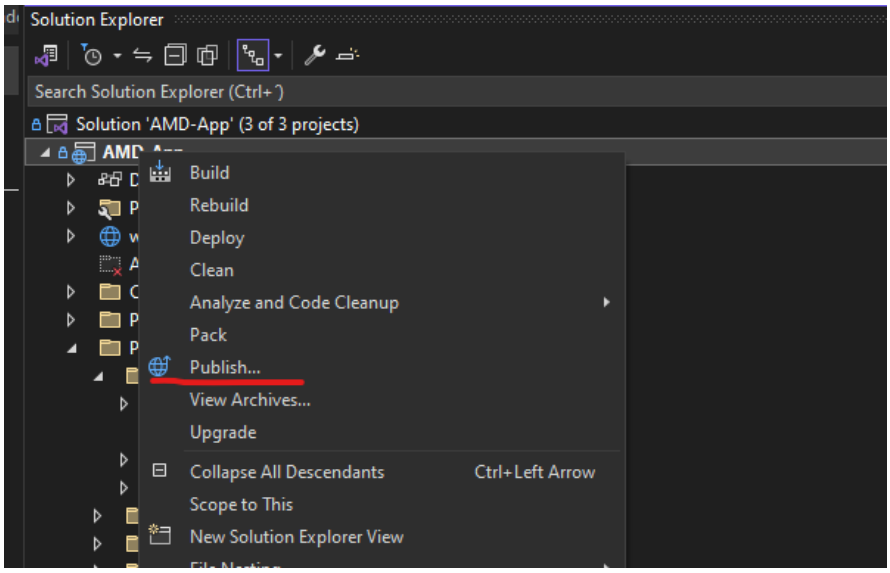
4) Setear en el appsettings.json las url que correspondan al ambiente en el que se quiere deployar (QA, DEMO, PROD, etc).

5) Si el deploy es para un ambiente no productivo; es decir, para QA o DEMO, etc. en el HelperService.cs a la propiedad AppVersion concatenarle al final, la primera letra del ambiente. Así:

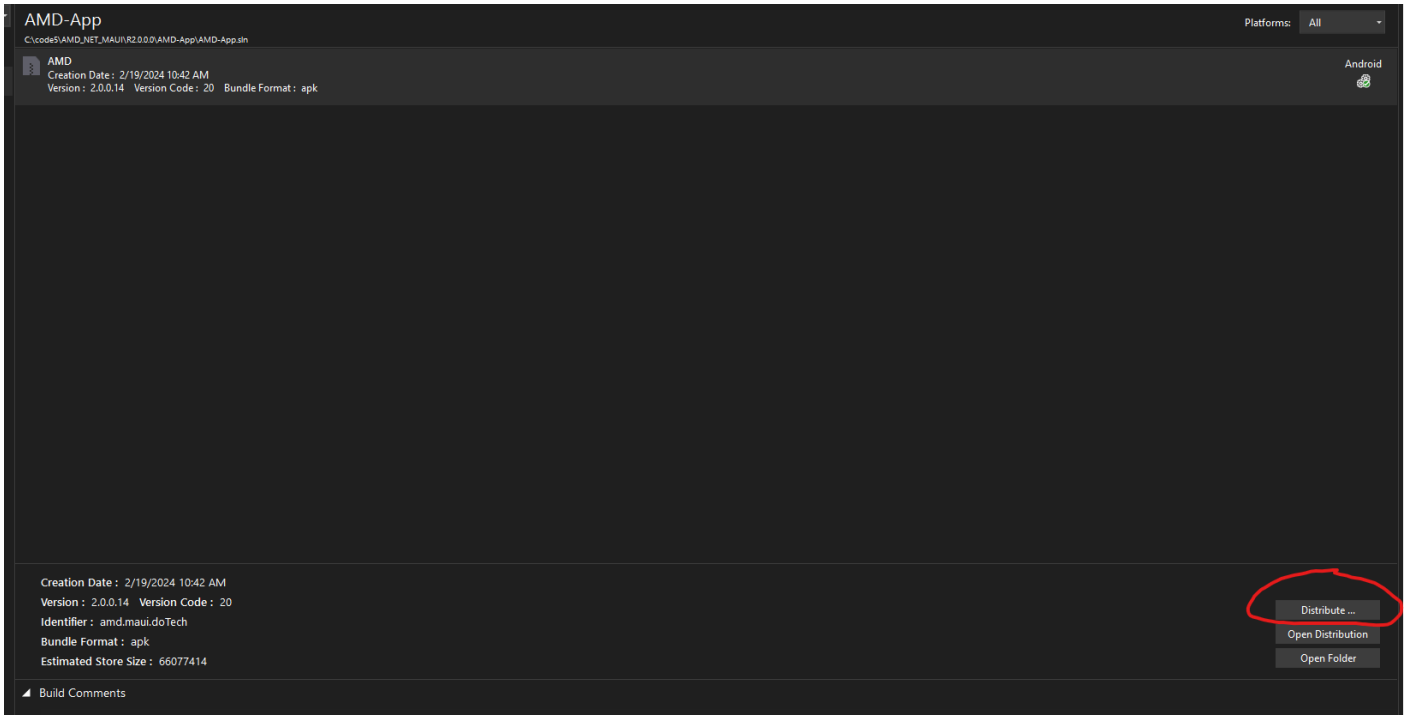
```
35 references | Nicolas Bulfon, 1 hour ago | 3 authors, 12 changes | 2 incoming changes
public class HelperService : BaseService
{
    #region Propiedades
    NetworkAccess accessType = Connectivity.Current.NetworkAccess;
    2 references | Gabriel Bulfon | 1 author, 1 change | 1 incoming change
    public bool NetworkStatus { get; set; }
    public EventHandler<ErrorGlobal>? OnError;
    2 references | Gabriel Bulfon | 1 author, 1 change
    public ParametricasService _parametricasService { get; set; }
    /// <summary>
    /// Versión de la app.
    /// Para las versiones de qa, demo, (salvo prod), concatenar acá la Q o la D.
    /// </summary>
    5 references | 3 authors, 6 changes | 2 incoming changes
    public string AppVersion
    { get
        {
            return  Microsoft.Maui.ApplicationModel.VersionTracking.CurrentVersion + "Q"  ;
        }
    }
```

(Asegurarse también, que el código del dispositivo no esté hardcodeado, en el DispositivoService.cs).

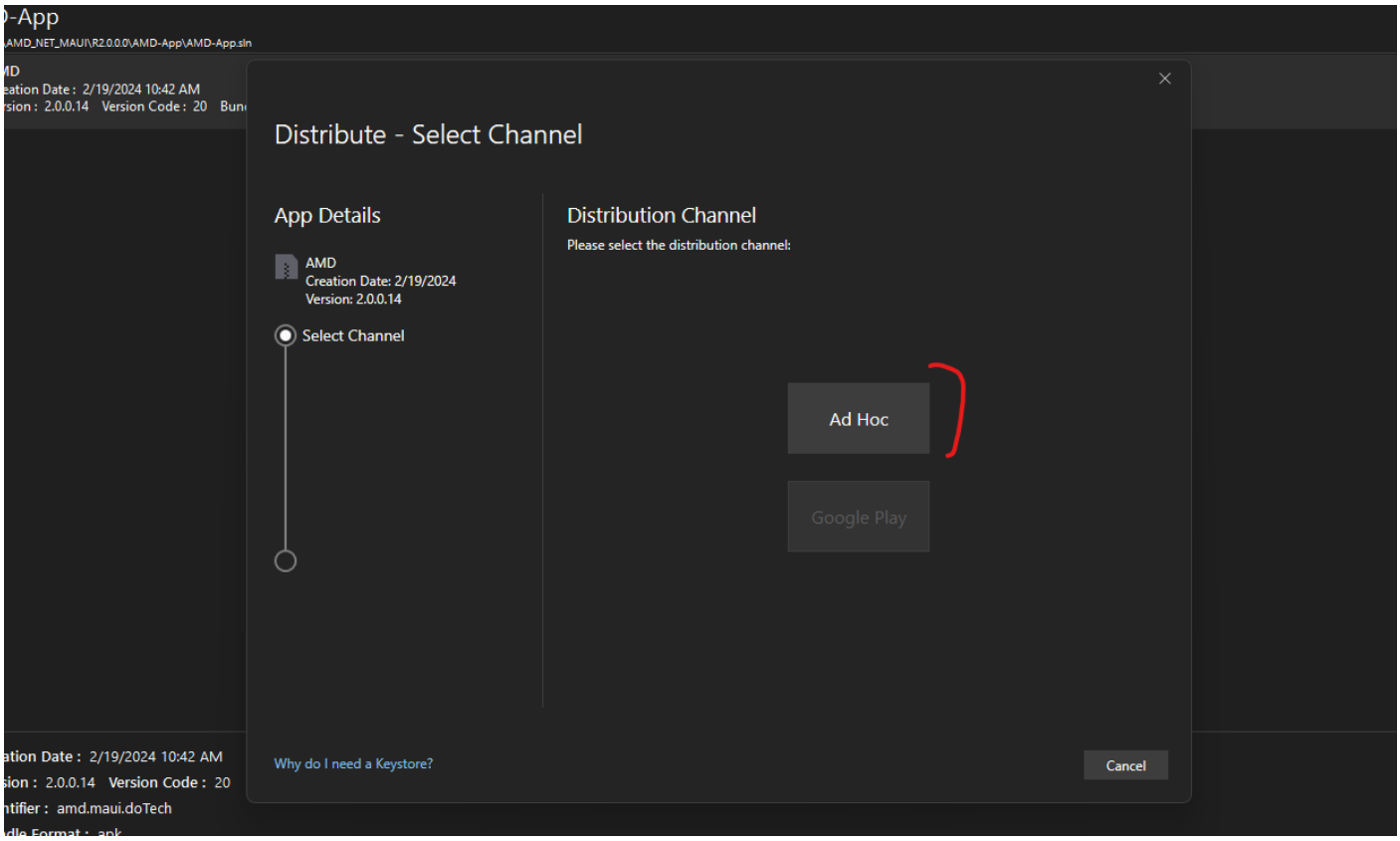
6) Sobre el proyecto, hacer click en Publish para realizar el deploy



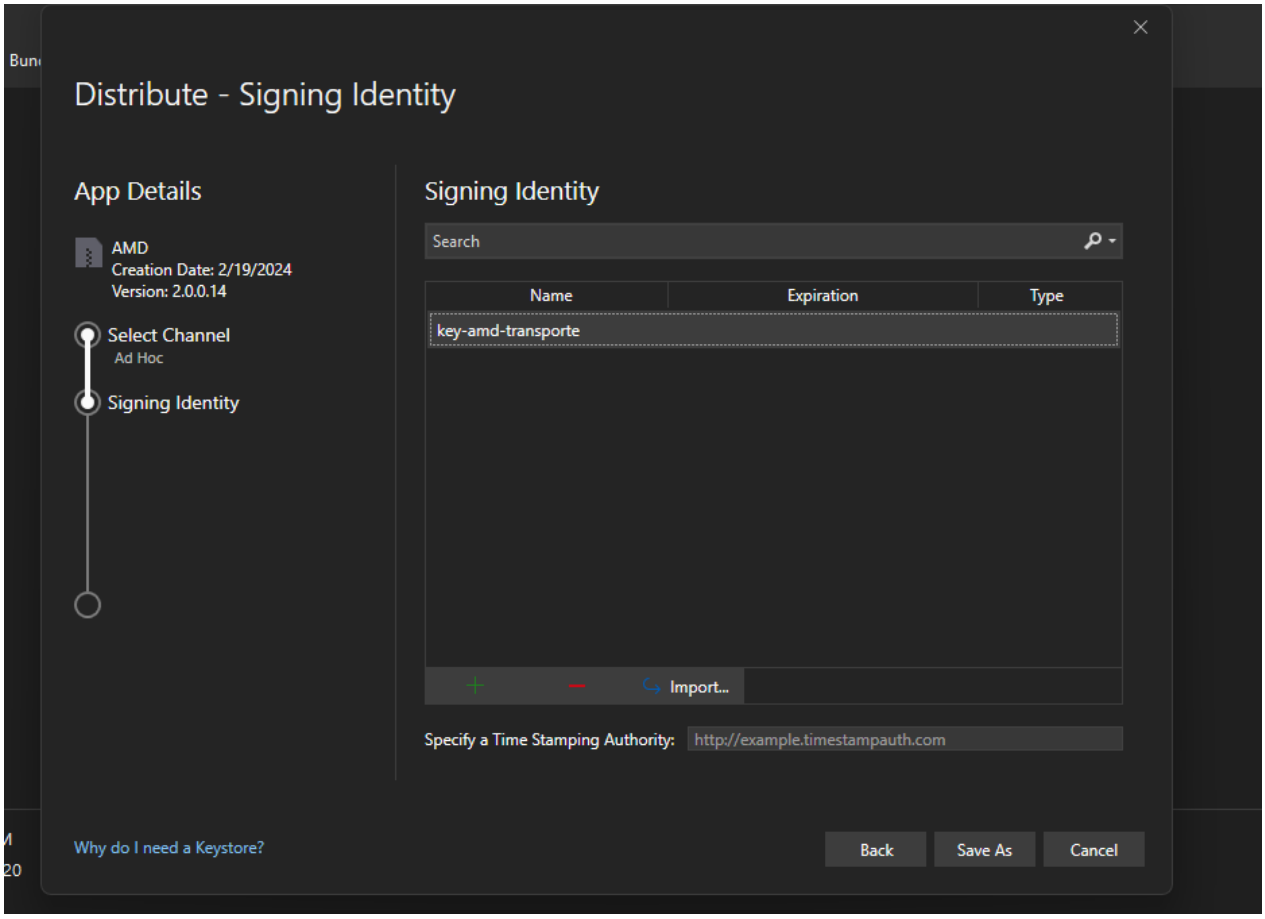
7) Una vez creado el apk, firmarlo, clickeando en la opción Distribute. Una vez ahí, seguir los pasos:



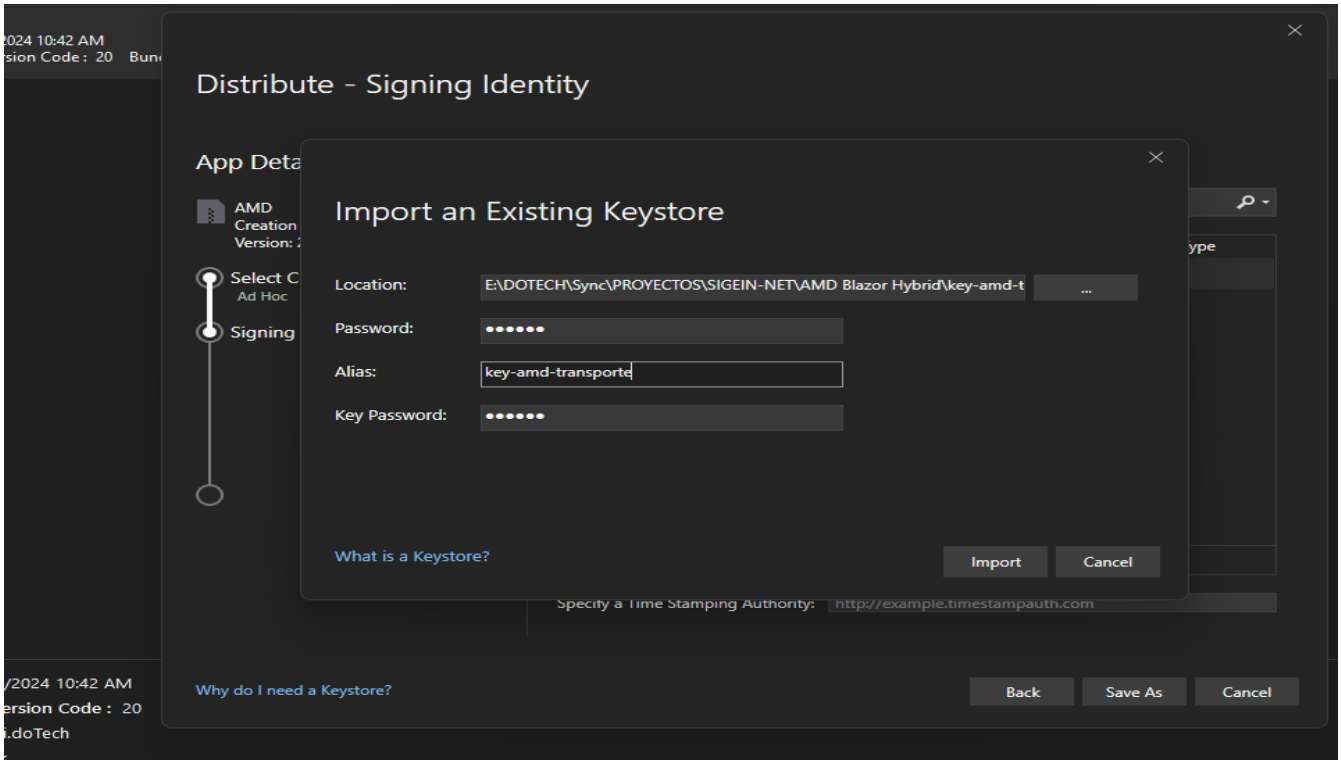
I) Clickear en Ad Hoc



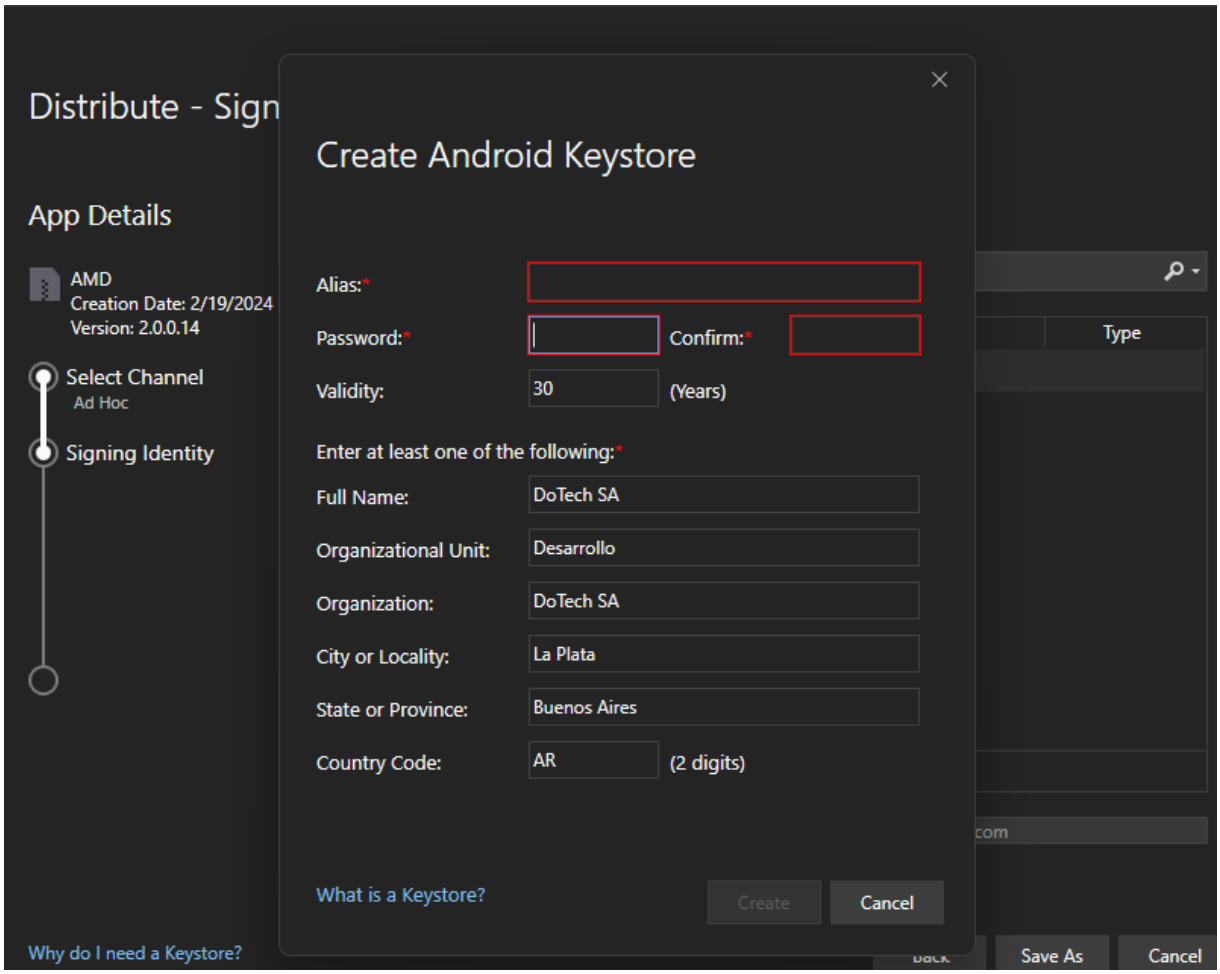
II) Si ya tenemos una key guardada, y queremos firmar el apk con ella, seleccionarla y Clickear Save As:



Si este no es el caso, podemos clickear en Import para importar una key que ya tengamos hecha con anterioridad (importante: la key que importemos debe ser de la extensión .keystore para que funcione óptimamente). Allí seleccionaremos la ubicación de la key, y completaremos los datos que correspondan. En ese caso, nos debería quedar algo así:

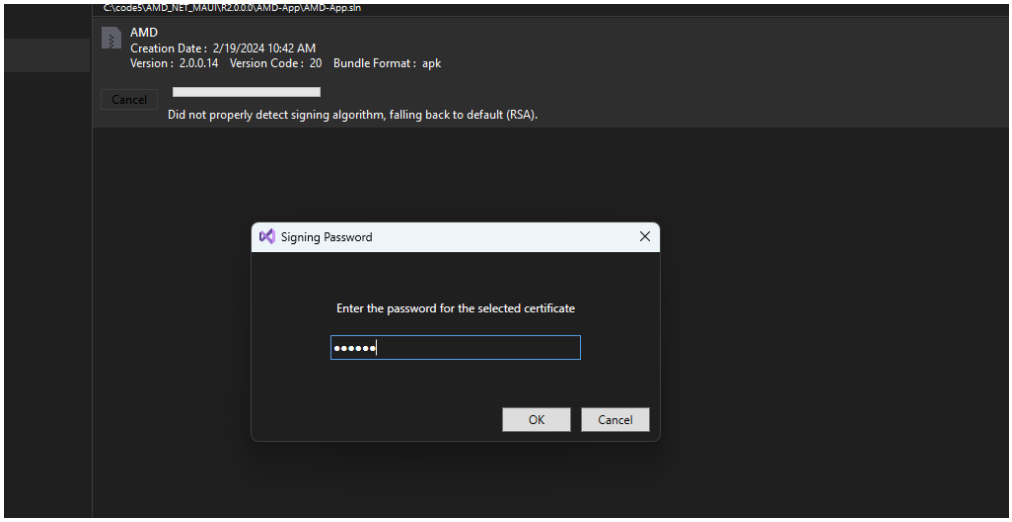


Si por otro motivo, queremos directamente crear una key nueva, clickeamos sobre el signo + y allí completaremos los datos necesarios. Acá dejo una guía de los datos que pueden ir como ejemplo:



Finalmente, si creamos o importamos una key, seleccionamos en Save As, donde indicaremos el directorio donde queremos que el Visual Studio nos deje el apk armado.

En ese momento al seleccionar el directorio donde guardarlo, nos pedirá otra vez que pongamos la password para el certificado:



8) Listo, ya se puede subir el apk al directorio que queramos.