Currency Converter App

We are going to build an app that excepts a user’s input value and converts that value from a selected form of currency, such as “Dollars ($)” to another form of currency, such as GBP - Great Britain Pounds (£). This app will be really helpful when traveling abroad to estimate more accurately how much things will cost, to make sure you have enough funds in your bank account to get that treasured souvenir!

As you may know, the currency exchange rate changes constantly based on a number factors such as market value, trades, etc. We are going to make a call to a public REST API to get the latest exchange rates, which are in the form of percentages converted to decimals, such as GBP – “0.7918907687605552”.

1. In Visual Studio, create a new “.Net MAUI App”:

A screenshot of a project

Description automatically generated

1. Target Framework: .NET 8.0 (or whatever is the latest version installed on your device). Click Continue

A screenshot of a computer

Description automatically generated

1. Type in a Project Name, I used “MyCurrencyConverter”. Leave all the Version Control options checked. Click Create.

A screenshot of a computer

Description automatically generated

1. After a minute or two, the packages and SDKs and their dependencies will load completely. Let’s take a look at what we’ve got…

A screenshot of a computer

Description automatically generated

1. I like to run the app right away to make sure it loaded correctly and my physical devices and emulators are all set up. Go ahead and select the “My Mac (MacCatalyst) dropdown at the top of Visual Studio.

A screenshot of a computer

Description automatically generated

1. Select one of the iOS simulators in the list, your choice! Click the Play button to the left of the dropdown. The Play button turns into a square, indicating that the simulator you chose is launching.

A screenshot of a computer

Description automatically generated

1. Give it a minute or two to load, and if all goes well, the basic .NET MAUI template app will launch on the simulator. Test it out if you like, click the button and see what happens! You can click the square up at the time to stop the app.

A screenshot of a phone

Description automatically generated

1. Great! Now let’s make sure our physical device is set up correctly. Plug in your iPhone with a USB -> Lighting cable into your Mac. To make sure you set up your development team and certificates correctly, Go to Visual Studio menu > Preferences > Publishing > Apple Developer Account. You should see an Account in the Accounts list and a Team associated with that account.

A screenshot of a computer

Description automatically generated

1. Click View Details, then Click Download All Profiles. You should see your certificates and their associated provisioning profiles in the display…

A screenshot of a computer

Description automatically generated

1. Now, right click on the Project in the Solution Explorer. Scroll down in the dropdown list and select “Properties”

A screenshot of a computer

Description automatically generated

1. Under iOS, expand arrow on the left, and select Bundle Signing. Make sure the following settings are selected:

* Configuration: Debug (Active)
* Platform: Any CPU
* Signing Identity: Developer (Automatic)
* Provisioning Profile: Automatic

You can leave Custom Entitlements and Additional Arguments blank for now.

A screenshot of a computer

Description automatically generated

1. Now in the device dropdown, select your Device. Make sure your device is plugged in and unlocked, and click the Play button.

A screenshot of a computer

Description automatically generated

You can view the status of the run in the lower left corner.

A screenshot of a phone

Description automatically generated

1. If all goes well, the app will launch on your device. Click the button and try it out!

A screenshot of a phone

Description automatically generated