

**Vizuális\_programozás\_L-K-GINFBAL-VIZUPROG-1**

**L\_LA01\_Vizprog\_MAUI – Calculator applikáció**

**Schiller Viktor – GWOQXX**

**Második Évfolyam – 2. Szemeszter**

**NJE GAMF Műszaki és Informatikai Kar**

Tartalom

[1. MAUI interfész 2](#_Toc131358993)

[1.1. Egyszerű és nagyszerű multiplatform mobilalkalmazás készítése 2](#_Toc131358994)

[2. Elrendezés kialakítása 3](#_Toc131358995)

[2.2 MainPage.xaml (kijelző ablak, általános elrendezés, stílus, színek) 5](#_Toc131358996)

[2.3 Gombok elrendezése 7](#_Toc131358997)

[3. A programkód 8](#_Toc131358998)

[3.1 SimpleCalculator osztály 11](#_Toc131358999)

[3.2 MauiProgram.cs leírás 12](#_Toc131359000)

[3.3 Az ’App.xaml.cs’ kódrészlet 12](#_Toc131359001)

[4. Összefoglalás 13](#_Toc131359002)

[FELHASZNÁLT IRODALOM & KÉPEK 15](#_Toc131359003)

# 1. MAUI interfész

## **1.1. Egyszerű és nagyszerű multiplatform mobilalkalmazás készítése**

A MAUI (Multi-platform App UI) egy új, nyílt forráskódú programozási keretrendszer, amely az .NET 6 és a C# programnyelv felhasználásával lehetővé teszi a fejlesztők számára, hogy többplatformos mobilalkalmazásokat hozzanak létre. A MAUI keretrendszer az Xamarin.Forms új verziója, és az Android, iOS, macOS, Windows, valamint az Ubuntu és a Debian Linux platformokat is támogatja.

A MAUI célja, hogy egyszerűbbé tegye a mobilalkalmazások fejlesztését, és egységesítse az alkalmazások megjelenését és viselkedését különböző platformokon. A fejlesztők lehetőséget kapnak arra, hogy egyetlen kódalap felhasználásával hozzanak létre olyan alkalmazásokat, amelyek futtathatóak Androidon, iOS-en és más platformokon is.

A MAUI keretrendszer kódja és az azt használó alkalmazások az .NET 6 keretrendszer alapúak, amelynek köszönhetően nagyobb teljesítményt és optimalizációt kínálnak, valamint lehetővé teszik az újabb .NET funkciók, például a C# 10 nyelvi funkcióinak használatát is.

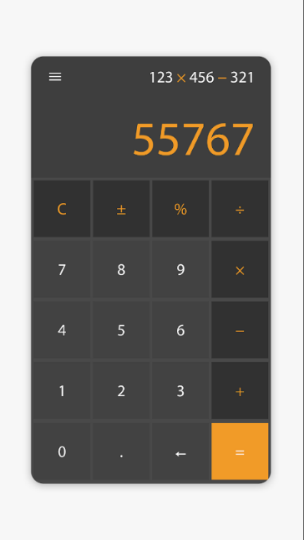
Az általam készített kód, C# és MAUI nyelven írt számológép program, amely az alábbi funkciókat tartalmazza: (+, -, \*, /).



*1. ábra. .NET Maui platform*

# 2. Elrendezés kialakítása

Legelső lépésben kerestem egy minta számológép alkalmazás kinézetet az internet képkeresőjében, ami sablonként fog szolgálni az elemek, gombok, kijelző, a komplett GUI elrendezését illetően, amolyan beépített Windows-osat, letisztult design-al, használható felülettel. (Pont, mint a MIT App Inventor beadandómban😊) Íme:



*2. ábra. A számológép-minta*

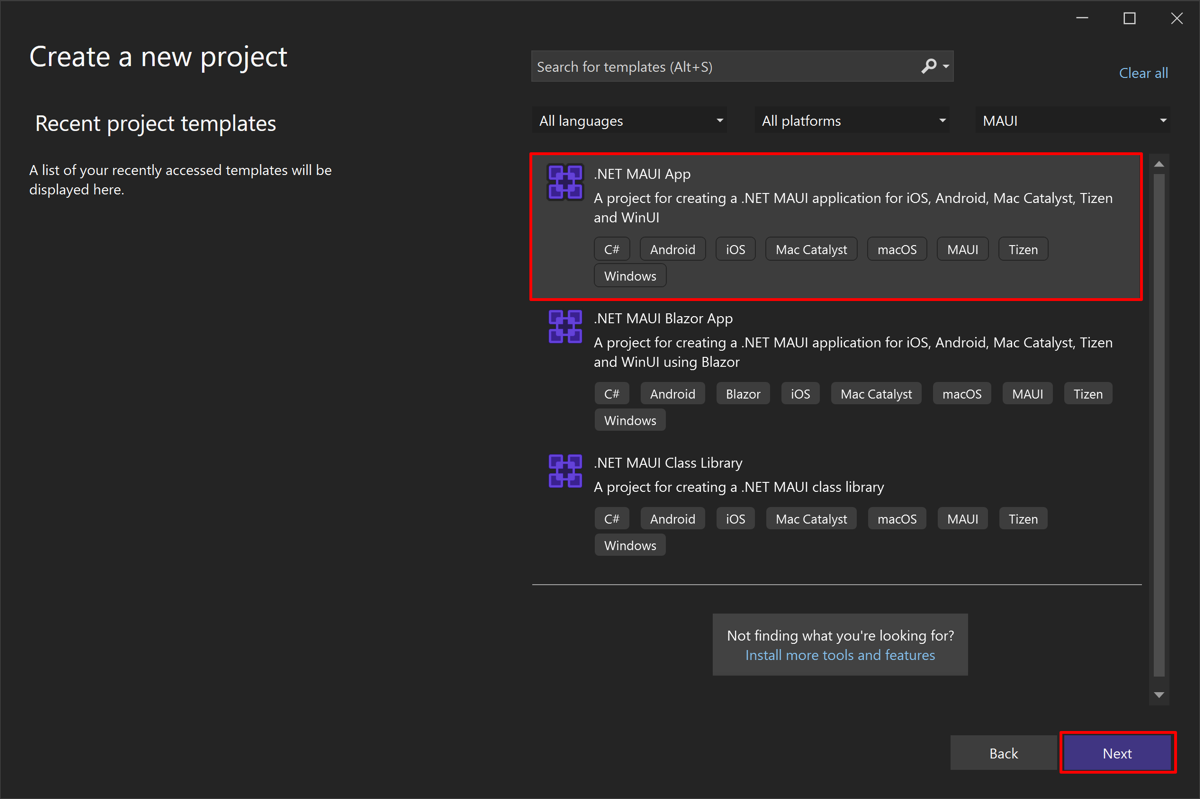
Elindítottam a Visual Studio 2022-t, majd a "Create a new project" lehetőséget a kezdőlapról.

A projekt varázslóban kiválasztottam a ".NET MAUI App" sablont, majd megadtam a projekt nevét és a projekt helyét. (a szükséges plugin-ok és keretrendszer letöltése után…)

Az "App Frameworks" ablakban kiválasztottam a .NET MAUI-t, majd a "Create" gombra a projekt létrehozásához.

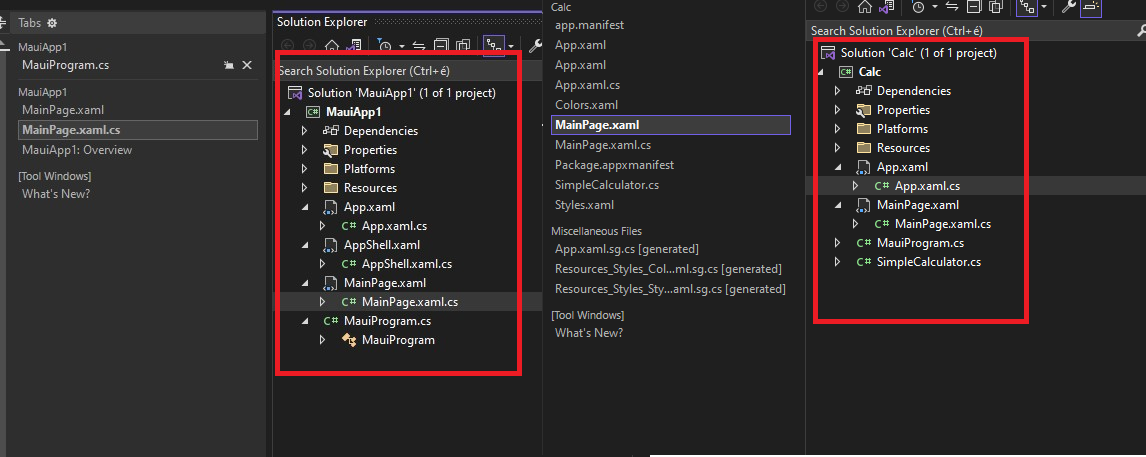
Ezután a Visual Studio 2022 létrehozta az új .MAUI projektet, és megnyitotta a projekt fájljait a Solution Explorerben.

A fájlok között találhatók az alkalmazás általános beállításait tartalmazó fájlok, az alkalmazás logikáját tartalmazó fájlok, valamint az alkalmazás felhasználói felületének XAML fájljai.



*3. ábra. Új projekt indítása*

A Calc alkalmazáshoz létrejött Maui projekt fájljai közül nem mindegyikre volt szükségem. Ilyen volt például az „AppShell.xaml” és „AppShell.xaml.cs” fájlok, mivel egyszerű alkalmazásról lévén szó, nem igényelt további ’navbar’ és menüfunkciókat. Létre lett hozva viszont egy ’SimpleCalculator.cs’ statikus osztály a jobb átláthatóság és egyszerűség kedvéért, ami csak a számításokért felel.

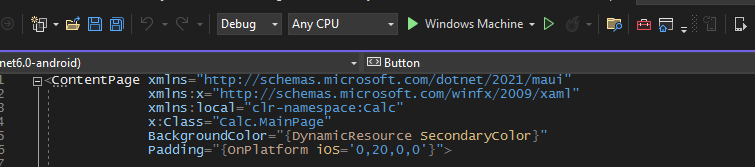


*4. ábra. Solution Explorer tartalma előtte & utána.*

## **2.2 MainPage.xaml (kijelző ablak, általános elrendezés, stílus, színek)**

A ContentPage elem a MAUI platform alapvető UI elemei közé tartozik, amely egy olyan oldalt definiál, amely tartalmazhat más UI elemeket. Az első néhány sorban az XML névtereket határozzák meg. Az xmlns attribútum az XML névterek (XML namespaces) definiálására szolgál, hogy a dokumentum elemeit az azonosítójuk és az azokat tartalmazó névterük alapján lehessen hivatkozni. Az első xmlns attribútum az "http://schemas.microsoft.com/dotnet/2021/maui" névtérre vonatkozik, amely a MAUI platform saját XAML elemeket definiál. A második xmlns attribútum az "http://schemas.microsoft.com/winfx/2009/xaml" névtérre utal, amely a Windows Presentation Foundation (WPF) névtere. A harmadik xmlns attribútum az alkalmazás saját névtérét határozza meg.

Az x:Class attribútum határozza meg a C# kódhoz tartozó osztály nevét, amely a XAML-ben definiált oldalt vezérli, jelen esetben ez a ’Calc.MainPage’.



*5. ábra. A fejléc*

Az oldal egyetlen gyermek eleme egy **Grid**, amelynek beállításai határozzák meg a számológép megjelenését. A Grid elem a felhasználói felület egy négyzetes rácshálózatát határozza meg. A Grid.RowDefinitions és Grid.ColumnDefinitions elemek segítségével definiálhatók a sorok és oszlopok száma és mérete. Az itt definiált rácshálózatban helyezkednek el az alkalmazás gombjai és a kijelzője.

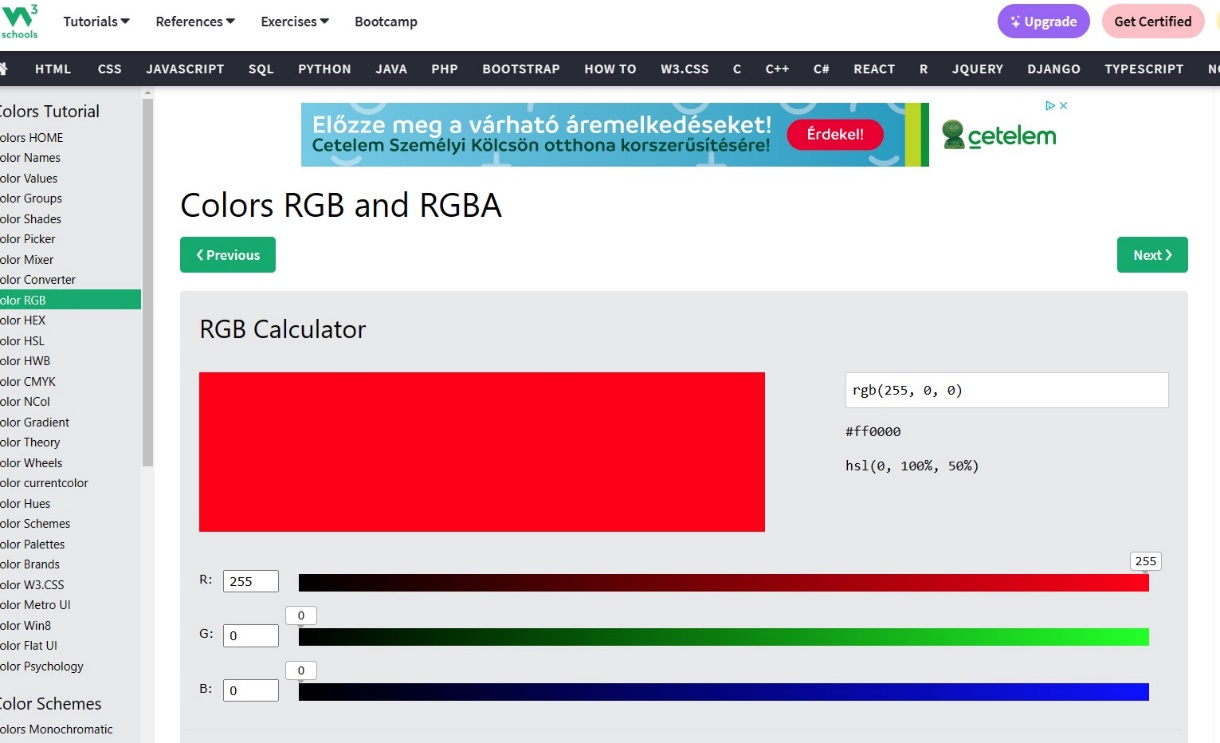
A Grid elem attribútumai a következők:

Padding: Meghatározza a Grid elem belsejének térközét. Az értékek a bal és jobb oldali, valamint a felső és alsó térközt adhatók meg. Az adott példában a bal oldali térköz 5 pixel, a jobb oldali pedig 0, a felső és alsó pedig 0.

RowSpacing: A sorok közötti térközt határozza meg pixelben.

ColumnSpacing: Az oszlopok közötti térközt határozza meg pixelben.

BackgroundColor: A Grid hátterének színét határozza meg hexadecimális kóddal. A kódok és színek pontos összeegyeztetéséhez segítségemre volt a [www.w3schools.com](http://www.w3schools.com) oldal.



*6. ábra. Színkódok a W3Schools oldalán.*

A Grid elemnek van egy RowDefinitions és egy ColumnDefinitions része is. Ezek a részek határozzák meg a Grid-en belüli sorok és oszlopok számát, illetve méretüket.

A RowDefinitions részben hat sor van, amelyek magassága a Height attribútummal határozható meg. Az \* karakter egy olyan értéket jelöl, amely a rendelkezésre álló helyet tölti ki, vagyis a hat sor közül az első két sor magassága az elérhető hely kétszerese (a kijelző miatt), míg a többi sor csak az elérhető hely maradék részét használja ki.

A ColumnDefinitions részben négy oszlop van, amelyek szélessége a Width attribútummal határozható meg. Az \* karakter itt is az elérhető helyet tölti ki, vagyis a négy oszlop egyenlő szélességű lesz. Az összes gombnak van egy Text attribútuma, amely a gombra ’írja’ a számot vagy az operátort.

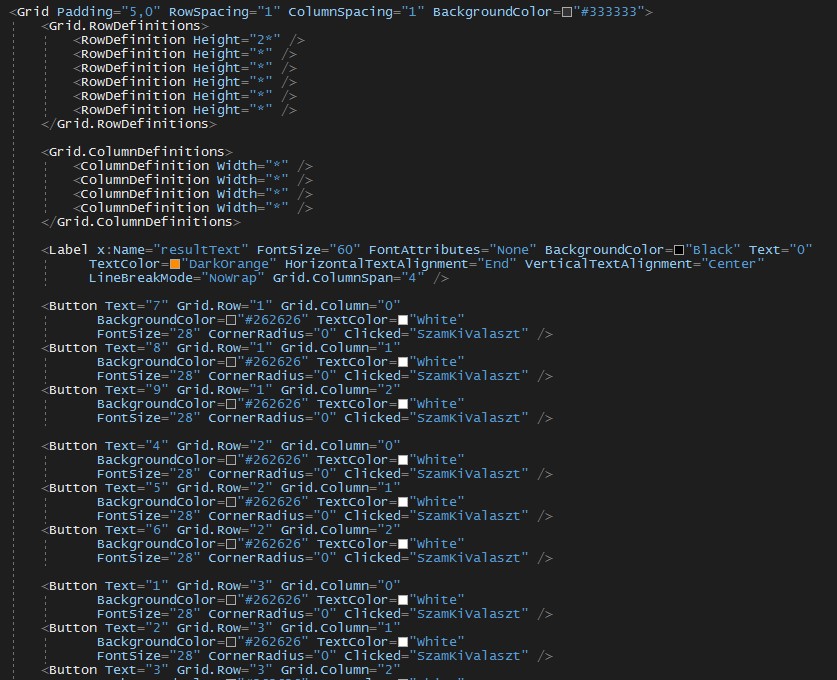
A Label elem definiálja a számológép kijelzőjét, ahol az eredmények jelennek meg. A Grid.ColumnSpan attribútummal megadja (4), hogy a kijelző a teljes rács szélességét elfoglalja.

## **2.3 Gombok elrendezése**

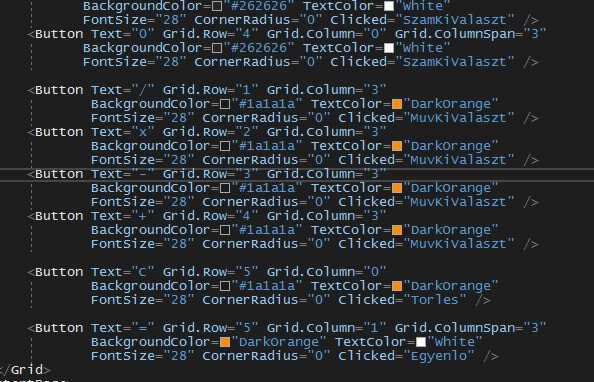
A gombokat Button elemek reprezentálják. Az Text attribútum határozza meg a gomb szövegét, a Grid.Row és Grid.Column attribútumok pedig a gomb helyzetét a rácsban. A BackgroundColor attribútum határozza meg a gomb hátterét, a TextColor attribútum pedig a szövegszínét.

A BackgroundColor attribútum meghatározza a gomb hátterét, a TextColor attribútum pedig a betűszínét. A Clicked attribútummal meghatározzuk a kattintás eseményét, amely az adott gombhoz tartozó metódust hívja meg a kódban.

A számokat és az alapvető matematikai műveleteket kezelő eseményeket (Click események) a SzamKiValaszt és MuvKiValaszt metódusok kezelik. Az Egyenlo metódus az egyenlőségjelre kattintás eseményére reagál, és kiszámítja az eredményt. A Torles metódus a C gombra kattintás eseményére reagál, és törli az összes számot és műveletet a kijelzőről.



*7.a) ábra. A Grid.*



*7.b) ábra. A Grid (folyt.)*

# 3. A programkód

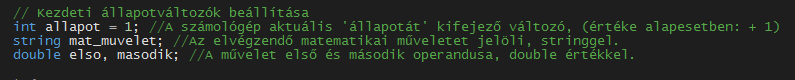
Az alábbiakban részletesebben dokumentálom a számológép kódját:



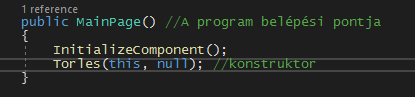
A ’Calc’ névtérben található a számológép alkalmazás.



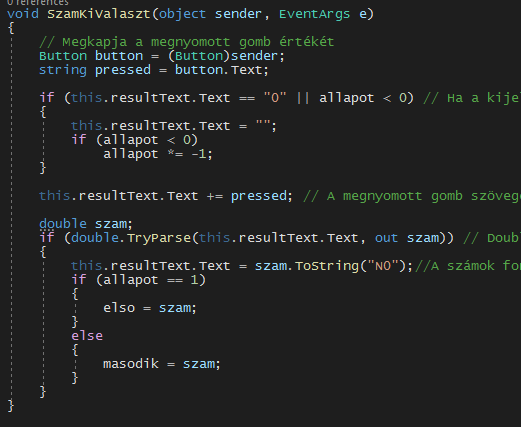
Ez az osztály a felhasználói felületet kezeli.



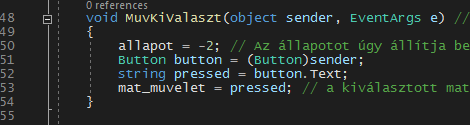
Az ’allapot’ változó az aktuális számológép állapotát jelöli. Alapértelmezett értéke 1, ami azt jelenti, hogy az első operandusra várunk. Ha -1, akkor új műveletre van szükségünk, ha -2, akkor egy művelet kiválasztásra került, és az utolsó operandusra várunk. A ’mat\_muvelet’ változó tárolja a kiválasztott matematikai műveletet. Az ’elso’ és ’masodik’ változók pedig tárolják az aktuális művelet operandusait.



Az osztály konstruktora, amely inicializálja a számológépet az alapértelmezett állapotba, törli a számológép állapotát és 0-ra állítja a kijelzőt.

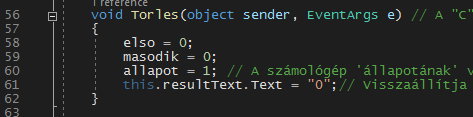


Ez a függvény kezeli a numerikus billentyűk gombnyomási eseményeit. Megkapja a megnyomott gomb értékét, majd hozzáadja azt a kijelző tartalmához. Ha az állapot negatív, vagy a kijelző tartalma 0, akkor törli a kijelzőt és az állapotot pozitívra állítja. Ha a megnyomott gomb értéke egy szám, akkor az ’elso’ vagy a ’masodik’ változóba menti az értéket, attól függően, hogy milyen állapotban van a számológép. Ha a kijelzőn már szerepel egy szám, akkor a gombnyomás hatására a kijelzőn megjelenő számjegyek összefűződnek, így hosszabb számot lehet megadni.

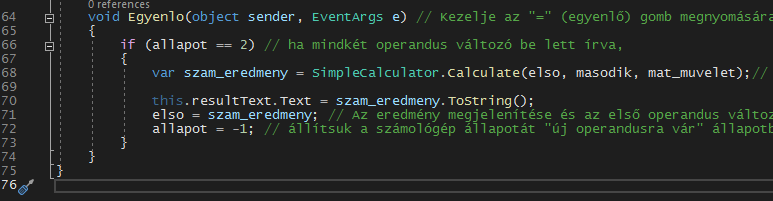


Ez a függvény ’void MuvKiValaszt()’ a matematikai műveletekkel kapcsolatos gombnyomásokat kezeli, amikor a felhasználó valamelyik műveleti gombot nyomja meg a számológépen.

Az első lépésben a függvény az állapot változót -2 értékre állítja, ami jelzi, hogy matematikai műveletet választottak. A függvény megkapja a megnyomott gomb értékét, és elmenti a kiválasztott műveletet a mat\_muvelet változóba.



A Torles függvény a "C" gomb eseménykezelője, ami a kijelzőt visszaállítja 0-ra, és a számológép állapotát újraindítja. Ennek érdekében az ’elso’ és ’masodik’ változók értékét nullára állítja, az ’allapot’ változó értékét pedig 1-re állítja. A ’resultText’ szövegdoboz tartalmát is visszaállítja 0-ra.

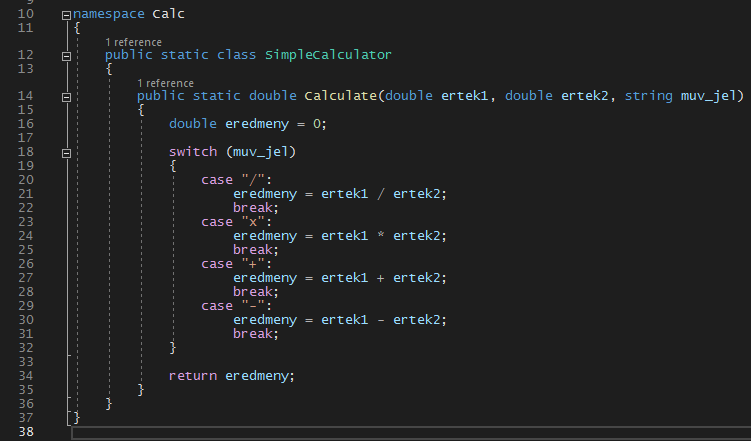


Az ’Egyenlo’ függvény az "=" gomb eseménykezelője, ami akkor hívódik meg, amikor az "=" gombra kattintanak. Az eseménykezelő ellenőrzi, hogy az összes operandus változó be lett-e már írva (allapot == 2). Ha igen, akkor meghívja a SimpleCalculator.Calculate() függvényt az első és a második operandus változókkal, valamint a mat\_muvelet értékével. A SimpleCalculator.Calculate() függvény kiszámítja a művelet eredményét, amit a szam\_eredmeny változóban tárolunk. Az eredményt átkonvertáljuk szöveggé, majd beállítjuk a resultText szövegdoboz tartalmát az eredmény szöveges reprezentációjára. Az ’elso’ változó értékét beállítjuk az eredményre, hogy a számológép további számításokra legyen felkészülve. Végül az ’allapot’ változó értékét -1-re állítjuk, hogy a számológép új operandusra várjon.

## **3.1 SimpleCalculator osztály**

A program egy 'SimpleCalculator' osztályt használ az eredmény kiszámítására. A program így moduláris, és lehetővé teszi a későbbi kódfejlesztéseket és az új funkciók hozzáadását.

A SimpleCalculator osztály egy statikus osztály, amelyben egy statikus Calculate metódus található. A metódus három bemenő paramétert vár: két double típusú számot (ertek1 és ertek2), valamint egy stringet (muv\_jel), amely a végrehajtandó matematikai műveletet jelöli.

**

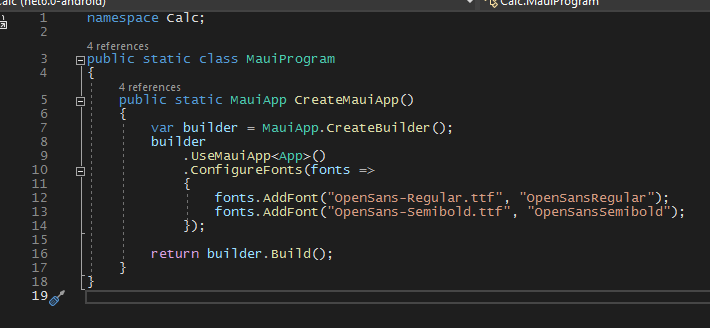
*8. ábra. Az eredmények kiszámolására szolgáló ’SimpleCalculator’ osztály*

A metódus egy switch utasítással eldönti, hogy milyen műveletet kell végrehajtani a bemenő paramétereken, és az eredményt double típusú értékként adja vissza. Az eredményt a hívó oldalnak kell kezelnie, a MainPage.xaml.cs oldalon a 'szam\_eredmeny' változóban kell eltárolni az egyenlőségjel megnyomásakor.

## **3.2 MauiProgram.cs leírás**

Ez a kódrészlet egy statikus osztályt definiál a Calc névtérben, MauiProgram néven. Az osztály egy CreateMauiApp() nevű statikus metódust tartalmaz, ami egy MauiApp példányt ad vissza.

Az UseMauiApp<App>() metódushívás beállítja a felhasználandó fő alkalmazás osztályát App-ra, amelyet a Calc névtérben definiáltam. A ConfigureFonts() metódushívás lehetővé teszi a betűtípus beállítását, és két ttf fájlt (OpenSans-Regular.ttf és OpenSans-Semibold.ttf) ad hozzá a betűtípusok listájához, amelyeket a kódban lehet használni. Végül a Build() metódus elkészíti és visszaadja a MauiApp példányt a beállított konfigurációval.

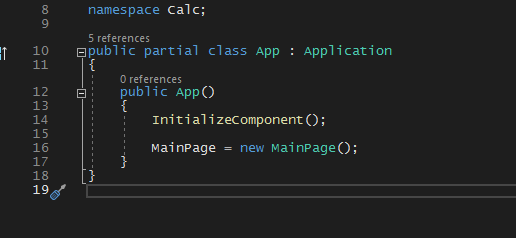


*9. ábra. A ’MauiProgram.cs’ osztály*

## **3.3 Az ’App.xaml.cs’ kódrészlet**

Ez a kódrészlet egy Calc nevű névtérben található App osztályt definiál, amely az Application osztályból származik és részbeni osztálydefiníció. Az Application osztály egy absztrakt osztály, amely lehetővé teszi az alkalmazások számára, hogy inicializálják az alkalmazásukat, beállítsák a betöltendő oldalakat, kezeljék az alkalmazáséletciklust stb. Ez a Xamarin Forms alkalmazás belépési pontjának tekinthető. A kód deklarál egy "App" nevű nyilvános & részleges osztályt, amely kiterjeszti az "Application" osztályt. A konstruktorban meghívásra kerül az InitializeComponent() metódus, ami egy generált metódus, amely inicializálja az oldal XAML fájljában definiált komponenseket és vezérlőelemeket. Ezután a MainPage tulajdonságot a MainPage osztály egy új példányára állítjuk, amely az alkalmazás főoldalaként fog szolgálni.

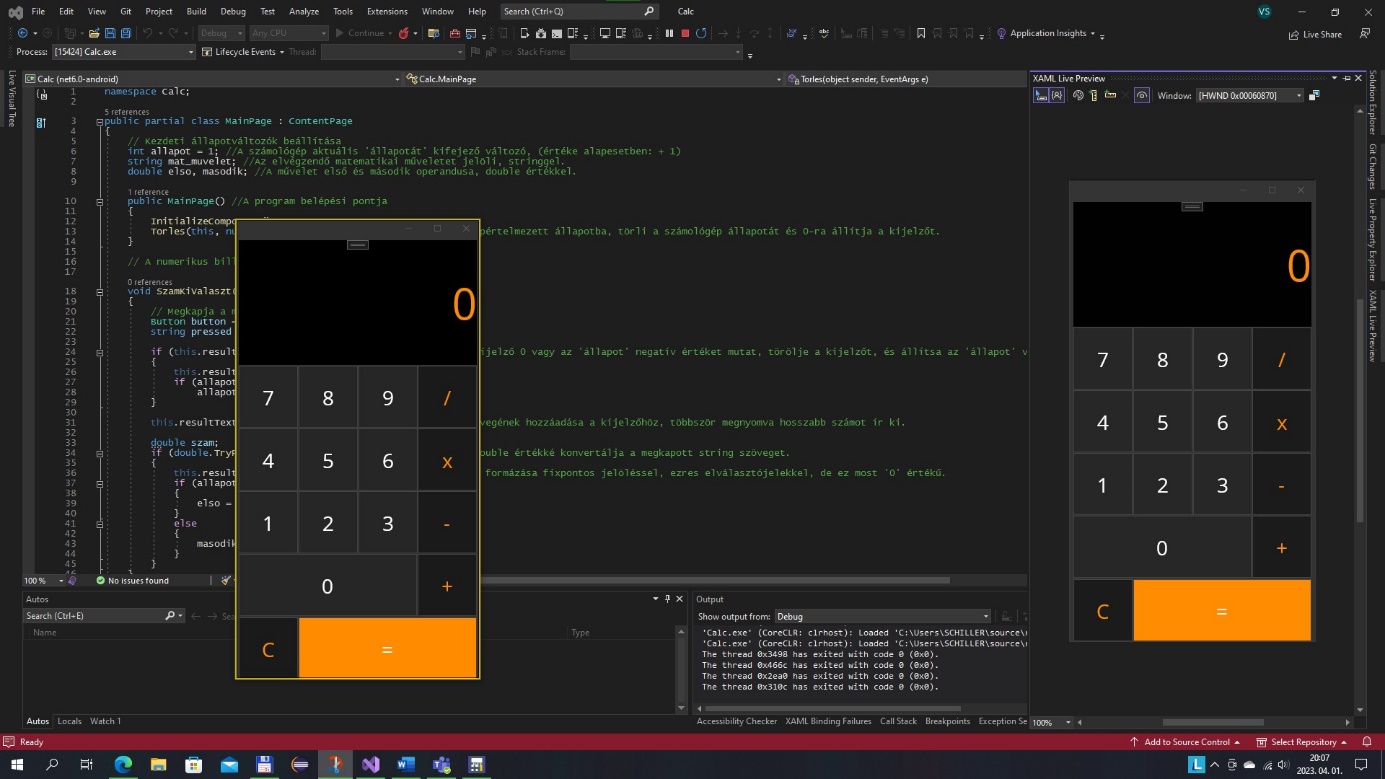
Összességében ez a kód beállítja az alkalmazás alapvető szerkezetét, és biztosítja, hogy a főoldal megfelelően inicializálva legyen, és készen álljon a megjelenítésre.



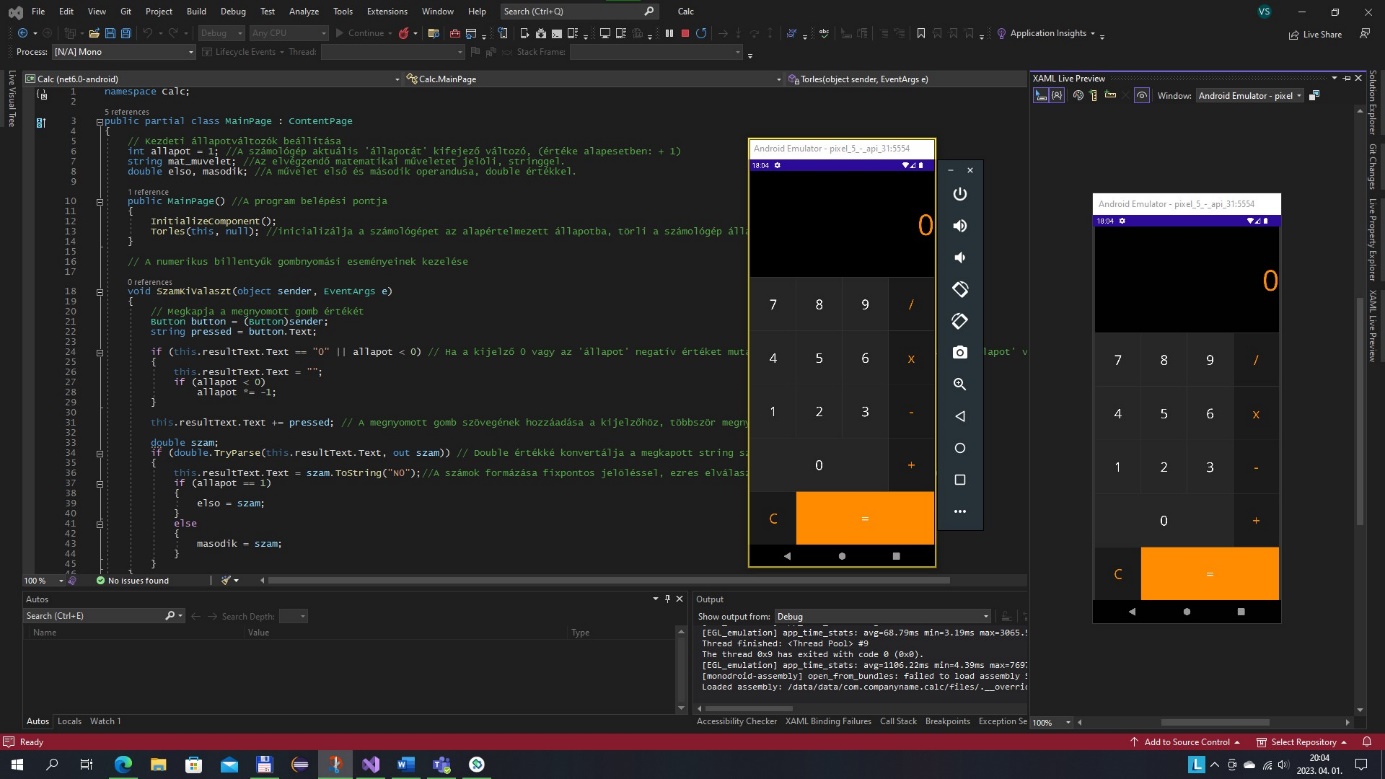
*10. ábra. Az ’App.xaml.cs’ osztály kódja*

# 4. Összefoglalás

A VS2022 Debug funkciójával ’Windows Machine’-on és ’Pixel 5 Android’ emulátoron is lefutott a projekt, mindkét esetben zökkenőmentesen. A projekt könyvtárába mind a Windows-os .exe fájl, mind az Android-os .apk fájl belekerült. A számológép számítási teljesítménye természetesen rendkívül alap, mindössze a négy fő műveletet tartalmazza, de úgy lett megírva, hogy később tovább bővíthető akár tudományos számológéppé is. Design-ja módosítható, de jelenleg is letisztult, és próbáltam követni a ’minta’ fotón szereplő Kalkulátor kinézetét. Az eredmény magáért beszél:

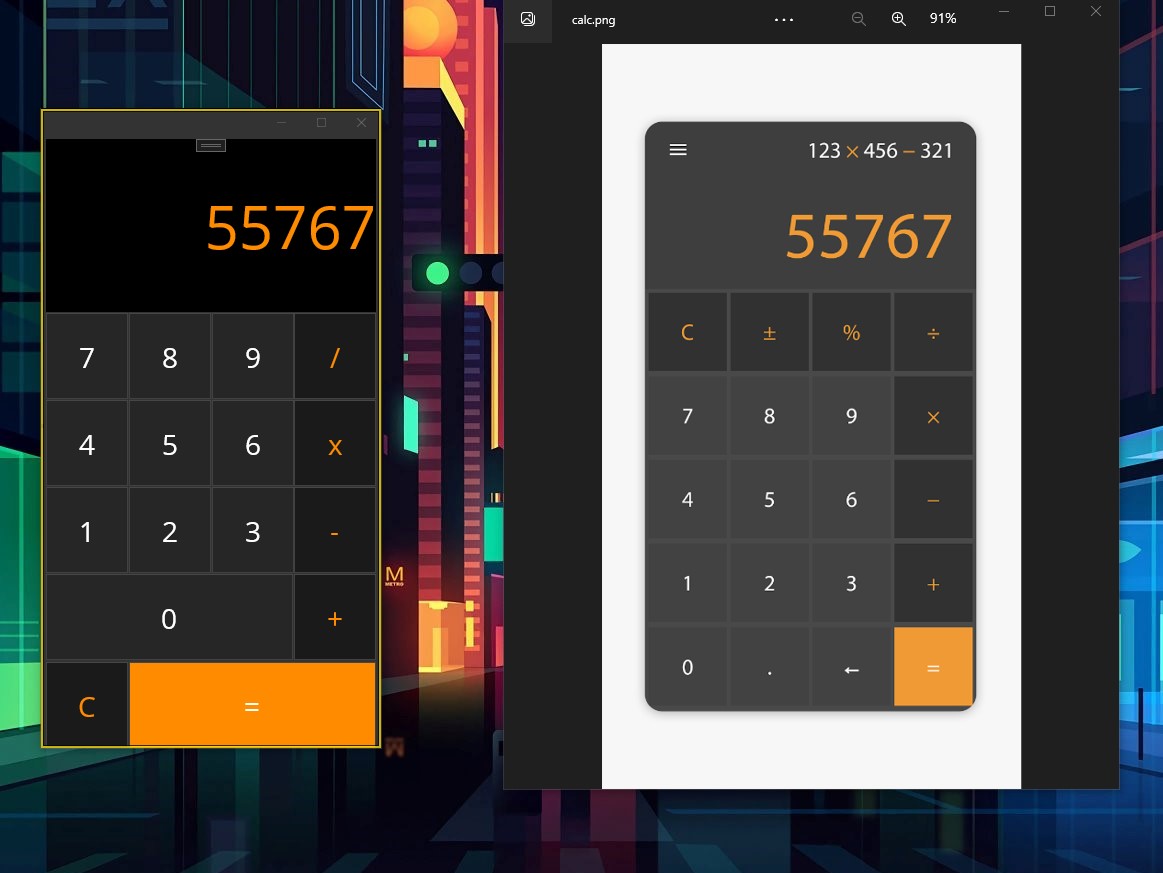


*11. ábra. Debug & Run ’Windows Machine’-on a Live Preview-vel*



*12. ábra. Debug & Run ’Android – Pixel 5’-ön a Live Preview-vel*

Összehasonlításképp:



*13. ábra. Az eredmény*Az űrlap teteje

Az űrlap teteje

FELHASZNÁLT IRODALOM & KÉPEK

* W3Schools [Colors RGB and RGBA (w3schools.com)](https://www.w3schools.com/colors/colors_rgb.asp)
* .NET Multi-platform App UI documentation [.NET Multi-platform App UI documentation - .NET MAUI | Microsoft Learn](https://learn.microsoft.com/en-us/dotnet/maui/?view=net-maui-6.0)
* Főcím kép https://images.app.goo.gl/jofd4MA7zJebe3vt6
* 1. kép https://images.app.goo.gl/695rqCge1Gsu8YZX6