**C# Příručka pro učitele**

Navigace

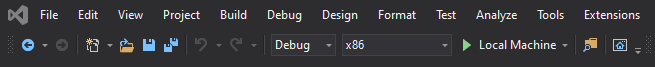
* Tip pro učitele – červeně zvýrazněný
* Odkaz - podtrhnutý

1. **Lekce – Základy C# a Universal Windows Platform**

V první lekci se podíváme na C# jako takový, vysvětlíme si jeho syntax a podíváme se na cykly. Dále se podíváme na prostředí, v němž budeme programovat a na jeho funkce.

**Agenda**:

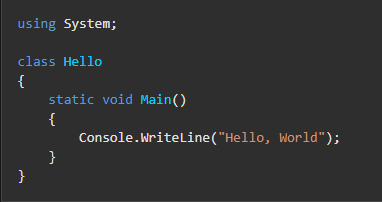
1. Syntax C#
   1. Cykly
2. Manifest
3. Oprávnění aplikace
4. Ikony aplikace
5. MainPage.xaml
6. MainPage.cs
7. Grid system
8. Komponenty
   1. Umístění komponentů
   2. Konfugurace komponentů
9. Spuštění aplikace



1. **Syntax C#**

Programovací jazyk C# je vysokoúrovňový jazyk, jehož syntaxe se moc neliší od syntaxe Javy. Základní stavbu programu si vysvětlíme na příkladu jednoduchého programu „Hello world“ ( viz. obrázek 1 ). V hlavičce programu se vždy za pomoci direktivy using odkazuje na obory názvů nebo používané knihovny. V tomto případě odkazujeme na System. Dále máme třídu class a její název Hello, protože C# je objektově orientovaný jazyk. Na novém řádku je pak složená závorka, jež nám určuje, co vše bude ve třídě obsaženo. Následuje metoda Main() s modifikátorem static a typem void. Uvnitř této metody je pak samotný kód. Na konci řádku kódu by vždy měl být středník. Metod může být ve třídě nespočet a mohou být různě modifikované.

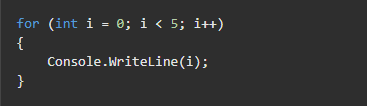
Tip: \*Nyní se můžete zeptat žáků, jaké druhy metod znají. ( odpovědi: Veřejné / Neveřejné; Void ( ta nic nevrací ) / S návratnou hodnotou ( return ) / Konstruktor ( slouží k vytvoření objektu ).\*



1

* 1. **Cykly**

Argument příkazu se vždy píše do kulatých závorek ( viz. obrázek 2 ). V C# máme několik cyklů, do; while(); for() a foreach(); .

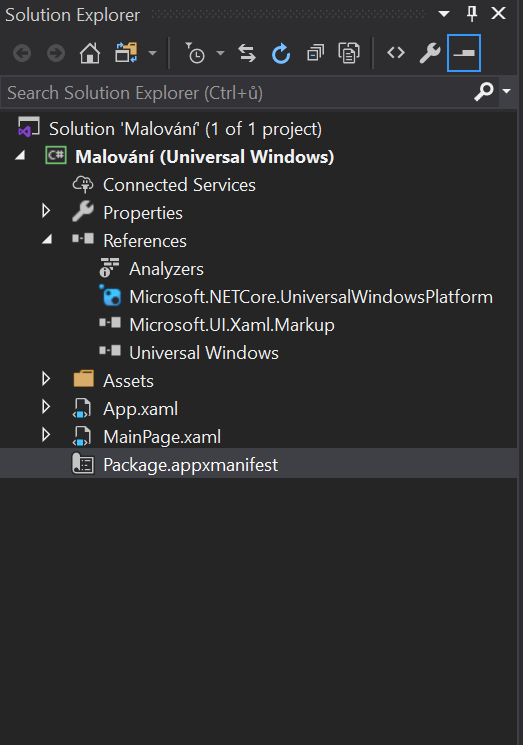


2

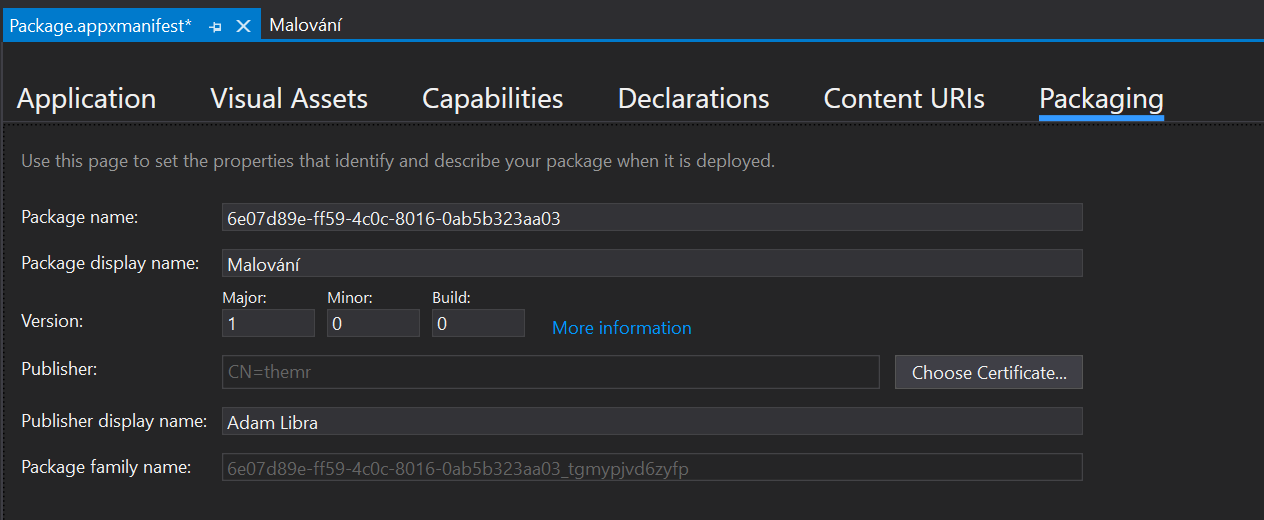
1. **Manifest**

Nyní si otevřeme Visual Studio, vytvoříme nový projekt - **Blank App ( Universal Windiws )**. UWP neboli Universal Windows Platform, je univerzální platforma pro vývoj aplikací psaných v C#, C++, Visual Basic nebo JavaScriptu. Tato platforma je podporována napříč zařízeními, od malých zařízení a IoT ( internet of things ), přes telefony, počítače, Xbox, až po virtuální realitu. Uživatel musí autorizovat přístup různým druhům zařízení, díky čemuž je UWP bezpečná. Díky univerzalitě této platformy není problém s přizpůsobením UI ( user interface ) na různých druzích a velikostech zařízení. UWP aplikace jsou běžně podporovány na Microsoft Store, což může značne rozšířit dosah.

Abychom se dostali do nastavení aplikace, musíme jít do „Package.appxmanifest“ ( viz. obrázek 3 ).

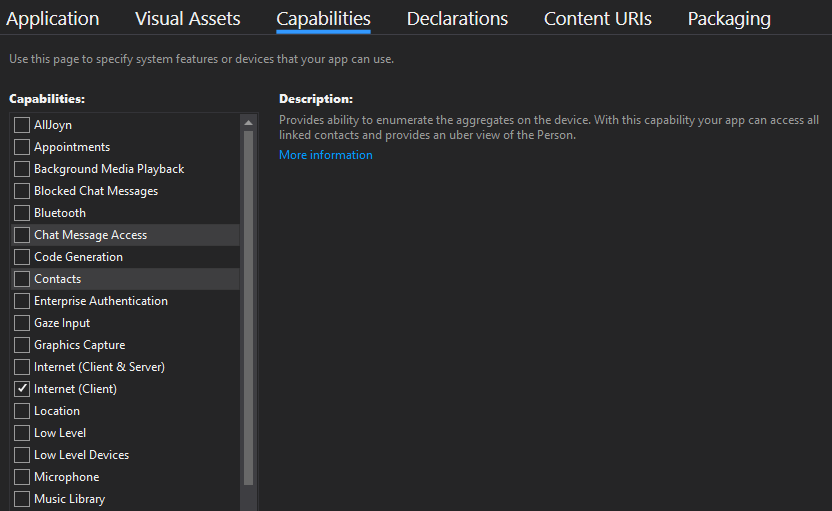


3

Zde se nejdříve podíváme na záložku „**Packaging**“, kde si můžeme nastavit jméno a verzi aplikace, přidat certifikát vývojáře a nastavit podpis autora aplikace ( viz. obrázek 4 ).

4

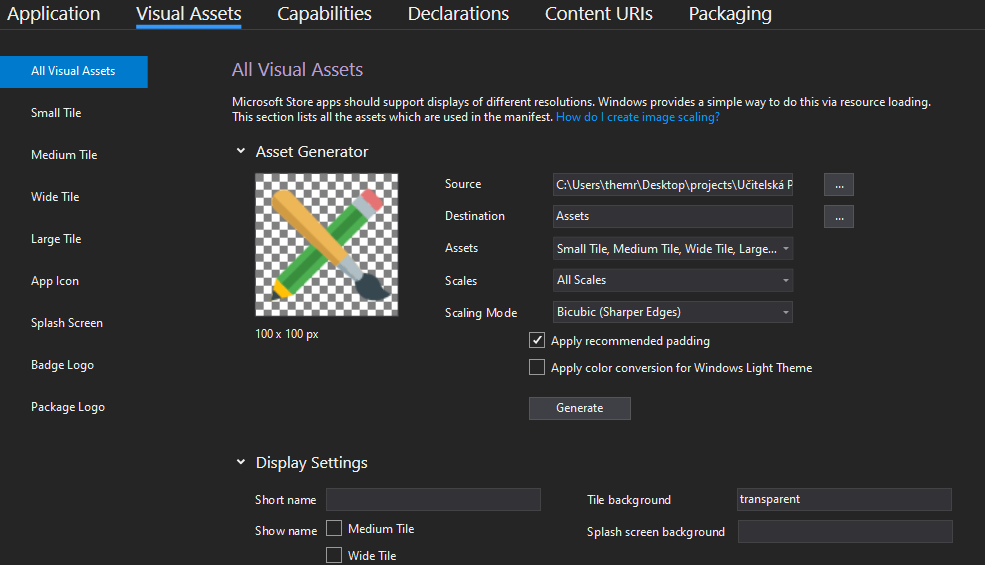
1. **Oprávnění aplikace**

Jako další se podíváme na položku **Capabilities**. Zde nenalzeneme nic jiného, nežli oprávnění aplikace. Můžeme tu aplikaci povolovat přístup k aplikacím zařízení, jako například ke Kontaktům, Kalendáři nebo Hudbě. Také zde můžeme povolit přístup k funkcím zařízení, jako třeba bluetooth, síti, úložišti, webkameře nebo mikrofonu.

5

1. **Ikony aplikace**

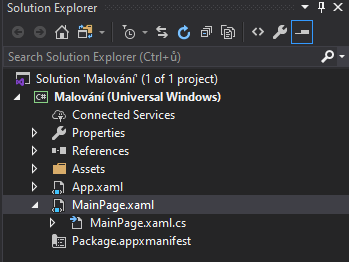
Po nastavení základní funkčnosti, se konečně můžeme podívat na design. Základní designové prvky nastavujeme v položce **Visual Assets**. Nejdříve se podíváme na ikonu aplikace. V záložce **All Visual Assets** si zadáme zdroj obrázku ( source ) a zbytek nastavení můžeme nechat jako výchozí. Doporučuji vybírat ikonu takovou, která bude mít transparentní pozadí. Mnou použitou ikonu naleznete ve složce „**Použité materiály**“. Dále už jen klikneme na **Generate** a můžeme si všimnout, že Visual Studio za nás udělalo veškerou práci s generováním responzivní ikonky.



6

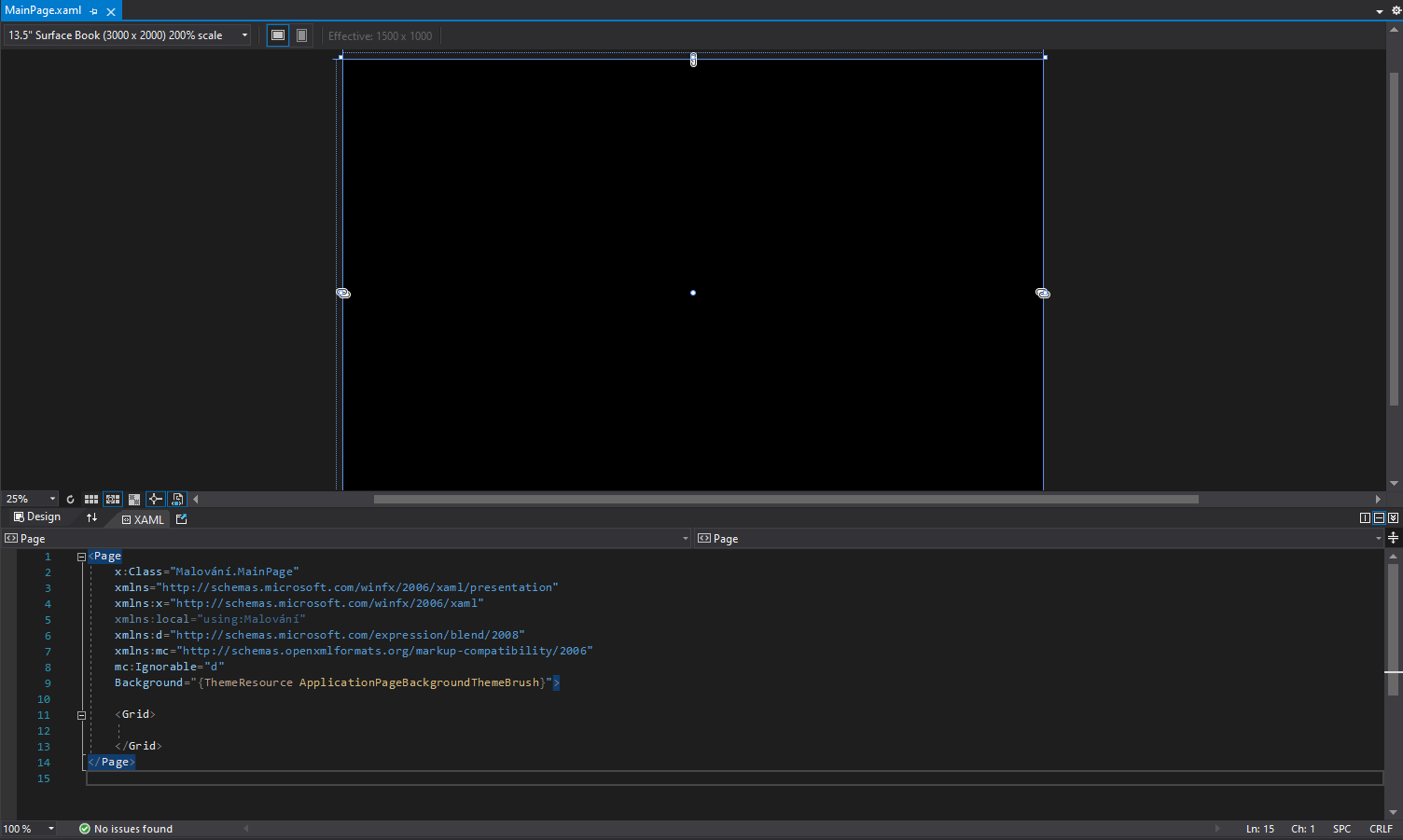
1. **MainPage.xaml**

Nyní se konečně můžeme podívat na samotný kód. Otevřeme si MainPage.xaml ( viz. obrázek 7 ).



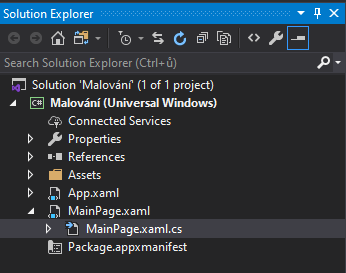
7

Nyní se nám VS rozdělilo na dvě části, horní, kde můžeme vidět živý náhled grafiky a měnit mezi různými náhledy podle responzivnosti ( levý horní roh, viz obrázek 8 ) a nalezneme zde i užitečné možnosti zobrazení v podobě sítě a možností uchycování elementů na ni ( v levo uprstřed, viz obrázek 8 ). V dolní části VS, se nacháyí samotný XAML kód. XAMLu se budeme věnovat až za chvíli.



8

1. **MainPage.cs**

Xaml nám definije jen grafickou stránku aplikace, co tedy definuje funkční? Tuto roli nám plní práve C#.Jeho zdrojový kód nalezneme v souboru **MainPage.cs** ( viz obrázek 9 ).

9

Tip: \* Než se vrhneme do samotné tvorby aplikace, procivičíme si, co jsme se doposud dozvěděli.

**Otázka č. 1**: Co je to UWP? Co tato zkratka znamená? Jaké má výhody oprotí klasickému programování bez jakékoliv jiné platformy?

**Odpovědi na otázku č. 1**: Universal Windows Platform je narozdíl od klasického způsobu programování jednodušší na zorientování v základní funkčnosti kódu, zajišťuje bezpečnost a umožňuje programovat na více zařízení najednou.

**Otázka č. 2**: Kde se nastavuje základní funkčost aplikace? Například podpis nebo verze?

**Odpověď na otázku č. 2**: Veškerá tato nastavení nalezneme v souboru **Package.appxmanifest.** Tyto konkrétní možnosti nalezneme pod záložkou **Packaging**

**Otázka č. 3**: Kde nalezneme oprávnění aplikace a kde si můžeme změnit ikonu?

**Odpověď na otázku č. 3**: V Package.appxmanifest v záložce **Capabilities** nalezneme veškerá oprávnění aplikace. V záložce **Visual Assets** pak nalezneme veškerá nastavení ikon. \*

1. **Grid System**

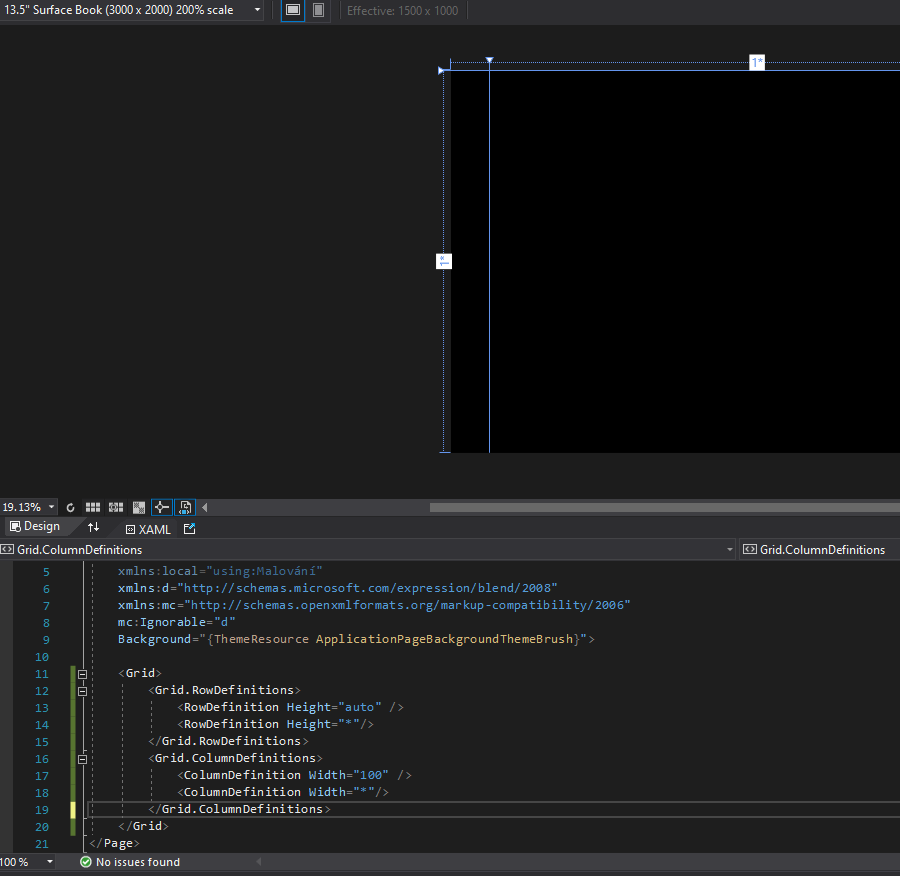
Nyní se překlikneme zpět do **MainPage.xaml** a podíváme se na rozložení aplikace neboli grid system. Grid nám bude definovat:

1. Šířku a výšku
2. Řádky a sloupce

Nejdříve bude potřeba popsat si trochu strukturu XAMLu. XAML se skládá podobně jako XML nebo HTML z párů tagů, otevíracího a zavíracího tagu. Uvnitř tagu můžeme psát různé atributy. U UWP se nám základní schéma XAMLu generuje automaticky, a to včetně páru <Grid></Grid>, do kterého budeme past veškerý XAML kód.

XAML má defaultně nastavený grid jako jeden sloupec a jeden řádek. Nejdříve si nadefinujeme řady a sloupce ( viz. obrázek 10 ). Všechno budeme teď psát do <Grid>. Řady definujeme pomocí <Grid.RowDefinitions> </Grid.RowDefinitions>. To tohoto páru vpíšeme pro každou řadu jeden tag <RowDefinition/>. V našem případě to budou právě dva. Každému z nich definujeme výšku pomocí atributu Height=” “. První tag bude mít hodnotu “auto” a druhý “\*”.

Nyní si nadefinujeme sloupce. Opět vytvoříme pár tagů, tentokrát to bude <Grid.ColumnDefinitions> </Grid.ColumnDefinitions>, do kterých budeme past naše sloupce. Sloupce budeme chtít vytvořit dva, u nichž budeme definovat šířku, výška je stejně jako u šířka řad defaultně nastavena na 100% zobrazení. Napíšeme tedy <ColumnDefinition Width=“100”/> , což nám vytvoří první sloupec s šířkou 100px. A druhý, <ColumnDefinition Width=”\*”/>, což vyplní zbytek prostoru.



10

1. **Komponenty** (viz. obrázek 11 )

Teď, když máme grid nastavený, mužeme do něj začít přidávat prvky neboli komponenty. Nejprve budeme přidávat tlačítko, za pomoci tagu <Button> </Button> . Dále přidáme nadpis, za pomoci tagu <TextBlock/>.

* 1. **Umístění komponentů**

Nyní si komponenty umístíme. K tomu slouží několik atributů:

1. Grid.Row=”*číslo*“ – Tento atribut nám prvky umístí do příslušné řady, čísla indexujeme od 0 od shora
2. Grid.Column=”*číslo*” – Tento atribut nám prvky umístí do příslušného sloupce, čísla indexujeme od 0 z leva
3. Grid.RowSpan=”*číslo*“ – Tento atribut vybere řádek příslušné šíře po celé výšce
4. Grid.ColumnSpan=”*číslo*” – Tento atribut vybere sloupec příslušné výšky po celé šíři

Naše tlačítko si umístíme do levého horního rohu. K tomu použijeme atributy Grid.Row=”0” a Grid.Column=”0”.

Náš nadpis chceme umístit úplně nahoru a doprostřed, proto mu přiřadíme atributy Grid.Row=”0” a Grid.ColumnSpan=”2”.

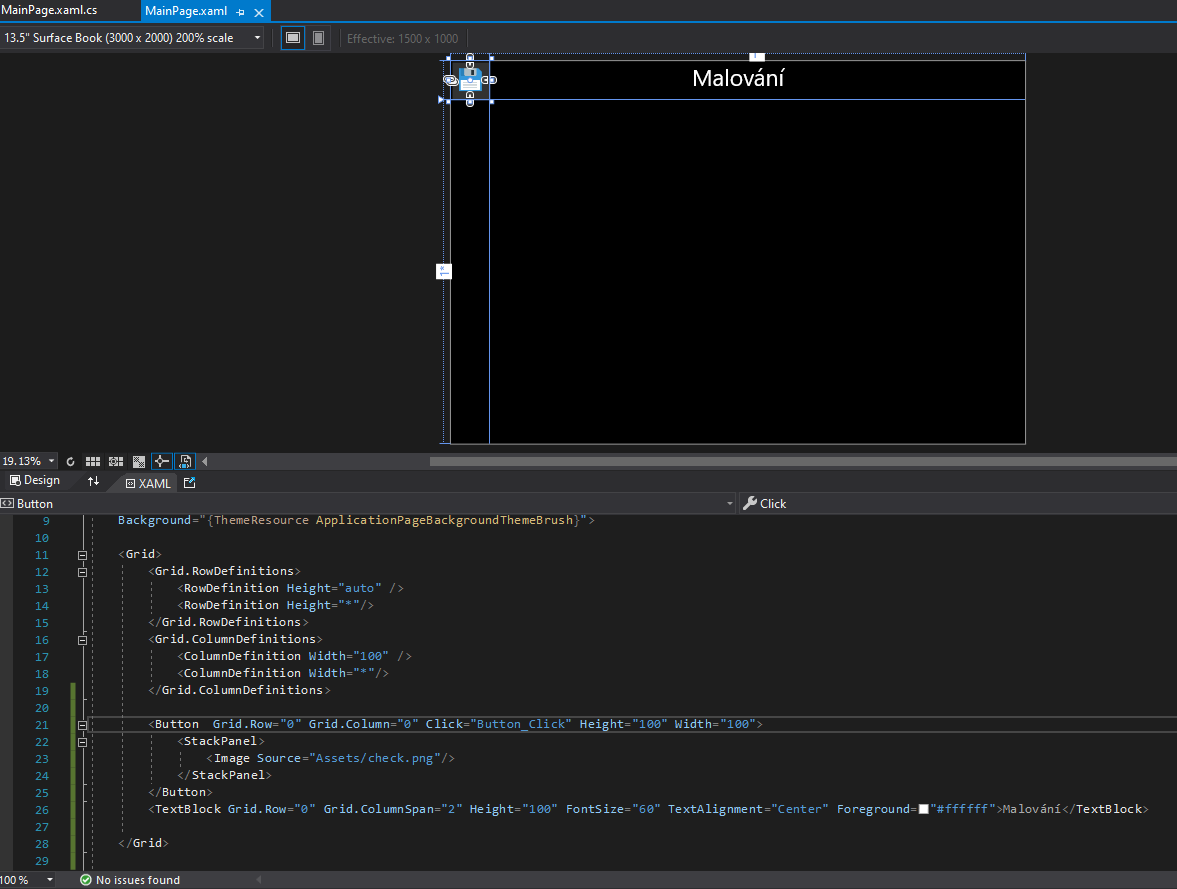
* 1. **Konfigurace komponentů**

Naše elementy je možné si s pomocí XAMLu upravovat nespočtem různých atributů. My si dnes ukážeme jen některé.

1. Height=”*hodnota*” a Width=”*hodnota*” – Nastavují výšku a šířku prvku
2. FontSize=”*hodnota*” – Nastavuje velikost písma
3. TextAlignment=”*hodnota*” – Určuje zarovnání textu
4. Foreground=”*hodnota*” – Barva popředí, třeba textu
5. Background=”*hodnota*” – Barva pozadí
6. Source=”*umístění nebo odkaz na soubor*” – Odkazuje na zdroj souboru

Naše tlačítko budeme chtít mít ve tvaru čtverce, proto mu přidáme hodnoty Height=“100“ a Width=“100“, dále mu přidáme hodnotu Click=“Button\_Click“, tato hodnota nám zajistí vyvolávání události po kliknutí na tlačítko. Zatím nepotřebujeme vědět víc, tohoto budeme využívat až později. Dále mu přiřadíme ikonku, k tomu využijeme párového tagu <StackPanel> </StackPanel>, který nám umožní vkládat další prvky do prvku nadřazeného. Do <StackPanel> vpíšeme tag <Image/> s parametrem Source=“Assets/check.png“, místo Assets/check.png můžeme vepsat umístění a název jakékoliv jiné ikonky.

Našemu nadpisu přidáme hodnoty Height=“100“, FontSize=“60“, TextAlignment=“Center“ a Foreground=“#ffffff“, díky jimž dostaneme bílý nadpis velikosti 60pixelů, zarovnaný na střed. Mezi tag a uzavírací tag napíšeme Malování, tento text se pak zobrazí v aplikaci.



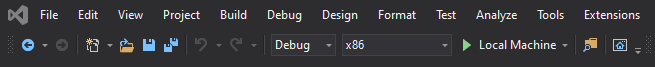
11

1. Spuštění

Nyní si spustíme naši aplikaci. Visual Studio nám poskytuje hned několik možností spuštění aplikace, jednou z nich je spuštění na lokálním zařízení.

Pro spuštění na lokálním zařízení jednoduše klikneme na tlačítko spustit na lokálním zařízení. ( viz. obrázek 12 )

Pro ozkoušení responsivnosti můžeme spustit debugování v emulátoru a spustit třeba na mobilu



Tip: \* Nakonec hodiny si opět položíme několik otázek

**Otázka č. 1**: Jaké tagy používáme na nastavení rozložení?

**Odpověď na otázku č. 1**: <Grid>, <Grid.RowDefinition>, <Grid.ColumnDefinition>

**Otázka č. 2**: Jakým parametrem zarovnáme text na střed?

**Odpověď na otázku č. 2**: TextAlignment=“Center“

**Otázka č. 3**: Co nám slouží ke spuštění debugování pro jiné platformy než je počítač?

**Odpověď na otázku č. 3**: Emulátor \*

Dokumentace:

<https://docs.microsoft.com/cs-cz/dotnet/csharp/>

<https://visualstudio.microsoft.com/cs/vs/msft-android-emulator/>