

Novinky VS 2022, .NET 6 a C#10

MGR. TOMÁŠ HAVETTA

Lektor – Mgr. Tomáš Havetta

- ▶ První program 1979
- ▶ C# od roku 2002, tedy od verze 1.0
- ▶ MCT od roku 2006
- ▶ Desktop App, Distribuované app
- ▶ Code review

Plán

- ▶ Proč přejít na .NET 6
- ▶ C# 10 – novinky
- ▶ .NET 6 – není .NET jako .NET
- ▶ VS 2022 – proč opustit VS 201x

.NET svět v roce 2018

.NET Framework

Verze 4.x

Windows only

All app

.NET Core

Verze 2.x

Win, Linux, Mac

CLI a Web

XAMARIN

Mobile platform

.NET Standard verze 2.x

.NET svět v roce 2019

.NET Framework
Verze 4.8
Windows only
All app

.NET Core
Verze 3.1
Win, Linux, Mac
CLI, Web a Win desktop

XAMARIN
Mobile platform

.NET Standard verze 2.x

.NET od 2020 a jeho budoucnost

.NET Schedule



- .NET 5.0 release in November 2020
- Major releases every year
- LTS for even numbered releases
- Predictable schedule, minor releases as needed

.NET 6

- ▶ Web App – ASP.NET minimal API
- ▶ Desktop App – WF, WPF, UWP, MAUI
- ▶ CLI
- ▶ Vylepšená správa RAM, GC
- ▶ Podpora ARM64 platformy včetně macOS Arm64
- ▶ C# 10, F# 6, VB vylepšení pro WinForms
- ▶ HTTP/3 support
- ▶ Package validation
- ▶ Hot reloading

VS 2022

- ▶ První 64-bitové IDE v historii VS
 - ▶ POZOR – část extenzi z VS 201x nejde použít
- ▶ Jediné VS pro .NET 6
- ▶ Intellicode
- ▶ Hot reload
- ▶ Find in Files
- ▶ Force Run To Cursor

C# 10

- ▶ Novinky v práci s namespace
- ▶ Novinky v lambda výrazech
- ▶ Novinky v strukturách - record struct
- ▶ Interpolating strings
- ▶ CallerArgumentExpression attribute

C# 10 - namespaces

- ▶ Global using
 - ▶ Pokud Vás už nebaví psát na začátek souboru plno using
- ▶ Project file => ImplicitUsings
 - ▶ Přidání dalšího namespace pomocí ItemGroup/Using/@Include
 - ▶ Odebrání namespace z defaultních ItemGroup/Using/@Remove
 - ▶ Při Include lze pomocí atributu Static= " True" vytvořit static using
 - ▶ Při Include lze pomocí atributu Alias= " XY" vytvořit alias pro namespace
- ▶ File Scope Namespace => Šetříme místo

Interpolating strings

- ▶ `$"xxx {promA} yyy {promB} zzz"`
- ▶ Standardně se překládá jako `String.Format("xxx {0} yyy {1} zzz",promA,promB)`
- ▶ Můžou nastat situace, kdy tento standard nevyhovuje
 - ▶ Interpolating string jako parametr metody, který se nakonec celý nevyužije, případně se nepoužije vůbec
 - ▶ Vyskládání řetězce by šlo optimalizovat použitím Span
 - ▶ ...

Interpolating strings - demo

- ▶ [InterpolatedStringHandler]
- ▶ `public ref struct LogStringHandler`
- ▶ `public LogStringHandler(int literalLength, int formattedCount)`
- ▶ `public void/bool AppendLiteral(string s)`
- ▶ `public void/bool AppendFormatted<T>(T data)`
- ▶ `internal string GetFormattedText()`

- ▶ `public void LogMessage(LogLevel level, string msg)`
- ▶ `public void LogMessage(LogLevel level, LogStringHandler handler)`

C# 10 – novinky v lambda výrazech

- ▶ „Natural type“ pro lambda výrazy
 - ▶ Delegát
 - ▶ Expression
 - ▶ Nemá
- ▶ Určení návratového typu
- ▶ Přidání atributů k lambda výrazu, parametrům a návratovému typu
 - ▶ Nemá efekt pro běh, pouze pro analýzu kódu a reflexi

C# 10 – record struct

- ▶ C# 9 zavedl record pro referenční typy
- ▶ C# 10 rozšířil tento mechanismus i pro hodnotové typy
 - ▶ Pro jednotnost přidal možnost zápisu „record class“
- ▶ Automaticky generované věci
 - ▶ Porovnávání obsahem
 - ▶ GetHashCode()
 - ▶ `private bool PrintMembers(StringBuilder builder)`
 - ▶ ToString() používající PrintMembers(...)

C# 10 - CallerArgumentExpression attribute

- ▶ Umožňuje kompilátoru jednoduše získat string odpovídající použitému výrazu
- ▶ Hodí se pro výpisy chyb v testech

.NET 6 - performance

- ▶ <https://devblogs.microsoft.com/dotnet/performance-improvements-in-net-6/>
- ▶ Přepřacovaný runtime
- ▶ JIT a inline
- ▶ Přepřacovaný FileStream
- ▶ A mnoho dalšího

.NET 6 – desktop app

- ▶ Windows Form
- ▶ WPF
- ▶ UWP
- ▶ MAUI

.NET 6 – JSON serializace

- ▶ Runtime reflexe nahrazena generovaným kódem
- ▶ Významné zvýšení rychlosti a zmenšení nároků na RAM

```
[JsonSerializable(typeof(Xxxxxx))]  
internal partial class MyJsonContext : JsonSerializerContext  
{  
}
```

VS 2022 – Hot Reload

- ▶ Hot Reload je dál vylepšován
- ▶ Při debug režimu podporuje všechny platformy (.NET Fnm, .NET x)
- ▶ Plná podpora v projektech .NET 6
- ▶ Omezení v XAMARIN
- ▶ Ne každou změnu kódu Hot Reload zvládne bez restartu aplikace

VS 2022 - IntelliCode

- ▶ Rozšíření schopností IntelliSense
- ▶ Vychází ze zápisu kódu a zvyklostí vývojáře (týmu)
- ▶ Systém ve vývoji

VS 2022 – Multi Repo Support

- ▶ Volba spadá zatím do „Preview Features“
- ▶ Solution může obsahovat mnoho projektů
- ▶ Některé projekty mohou být v jiném GIT úložišti

VS 2022 – Find in Files

- ▶ Hledání ve velkých projektech bývalo často zdrojem problémů
- ▶ VS 2022 má úplně nový mechanismus hledání

Rady do života

- ▶ Přejít do .NET světa neodkládejte
- ▶ VS 2022 je významnou změnou pro vývojáře