Ekonomická fakulta
Faculty of Economics

Jihočeská univerzita v Českých Budějovicích
University of South Bohemia in České Budějovice

Jihočeská univerzita v Českých Budějovicích

Ekonomická fakulta

Katedra aplikované matematiky a informatiky

Bakalářská práce

Podpora vývoje .NET aplikací na operačním systému GNU/Linux

Vypracoval: Josef Sochacký

Vedoucí práce: Mgr. Radim Remeš, Ph.D.

České Budějovice 2022

JIHOČESKÁ UNIVERZITA V ČESKÝCH BUDĚJOVICÍCH

Ekonomická fakulta

Akademický rok: 2020/2021

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Josef SOCHACKÝ**
Osobní číslo:	**E19040**
Studijní program:	**B6209 Systémové inženýrství a informatika**
Studijní obor:	**Ekonomická informatika**
Téma práce:	**Podpora vývoje .NET aplikací na operačním systému GNU/Linux**
Zadávající katedra:	**Katedra aplikované matematiky a informatiky**

Zásady pro vypracování

Cílem práce je vypracovat přehled softwarových prostředků podporující vývoj .NET aplikací v prostředí linuxových operačních systémů. Porovnejte jejich funkce a vlastnosti, uveďte využití těchto softwarových nástrojů. Vhodnou metodou porovnejte jednotlivá řešení, využijte ukázkové aplikace.

Metodický postup:

1. Studium odborné literatury.
2. Teoretický popis konkrétních dostupných produktů.
3. Porovnání a analýza dostupných produktů, zhodnocení jejich použitelnosti v reálném prostředí.
4. Závěry a doporučení.

Rozsah pracovní zprávy:	**40 – 50 stran**
Rozsah grafických prací:	**dle potřeby**
Forma zpracování bakalářské práce:	**tištěná**

Seznam doporučené literatury:

1. Albahari, J. (2021). *C# 9.0 in a Nutshell.* Sebastopol, CA (USA): O'Reilly.
2. Arif, H., & Qureshi, H. (2020). *Adopting .NET 5.* Birmingham, UK: Packt.
3. Metzgar, D. (2018). *.NET Core in Action.* Shelter Island, NY (USA): Manning.
4. Price, M. J. (2020). *C# 9 and .NET 5 – Modern Cross-Platform Development.* Birmingham, UK: Packt.
5. Seroter, R. (2018). *Modernizing .NET Applications.* Sebastopol, CA (USA): O'Reilly.

Vedoucí bakalářské práce: **Mgr. Radim Remeš**
Katedra aplikované matematiky a informatiky

Datum zadání bakalářské práce: 25. ledna 2021
Termín odevzdání bakalářské práce: 15. dubna 2022

JIHOČESKÁ UNIVERZITA
V ČESKÝCH BUDĚJOVICÍCH
EKONOMICKÁ FAKULTA
Studentská 13 (26)
370 05 České Budějovice

doc. Dr. Ing. Dagmar Škodová Parmová
děkanka

doc. RNDr. Tomáš Mrkvička, Ph.D.
vedoucí katedry

Prohlášení

Prohlašuji, že svou bakalářskou práci „Podpora vývoje .NET aplikací na operačním systému GNU/Linux“ jsem vypracoval/a samostatně pouze s použitím pramenů a literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to – v nezkrácené podobě/v úpravě vzniklé vypuštěním vyznačených částí archivovaných Ekonomickou fakultou – elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. Zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

Datum

Podpis studenta

Poděkování

Rád bych poděkoval vedoucímu bakalářské práce Mgr. Radimu Remešovi, Ph.D. za odborné a metodické vedení, podnětné a cenné rady, připomínky, vstřícnost a velkou trpělivost při zpracovávání mé bakalářské práce.

Obsah práce

1 Úvod

Microsoft je jednou z nejznámějších a největších technologických společností a již dlouhou dobu se v oblasti softwarových řešení pohybuje mezi nejlepšími. Tato společnost si také může připsat autorství jednoho z nejrozšířenějších operačních systémů, a to operačního systému Microsoft Windows. Mimo jiné je tato společnost také autorem .NET platformy. .NET platforma je softwarová technologie určená pro vývoj a běh různých druhů aplikací, od aplikací pro stolní počítače a mobilní telefony až po aplikace webové. Tato technologie je na operačním systému Microsoft Windows velmi dobře podporována a také hojně rozšířena. Ale jaká je situace na jiných operačních systémech?

Cílem této bakalářské práce je prozkoumat úroveň podpory této technologie na operačním systému GNU/Linux a také vypracovat přehled a srovnání softwarových prostředků, které tuto technologii a tento operační systém podporují. Reálná použitelnost této podpory bude vyzkoušena při vývoji ukázkové aplikace.

V úvodu práce je popsána samotná .NET platforma, její vlastnosti a architektura. Následuje popis operačního systému GNU/Linux. Poté je prostor věnován různým implementacím .NET platformy, jejich historii a specifickým vlastnostem. Zmíněn je zde také problém softwarových patentů. Následují informace o rozšířenosti .NET platformy na operačním systému GNU/Linux včetně přehledu několika aplikací postavených na této platformě. Poté následuje přehled vývojářských nástrojů, jejich vlastností a funkcí včetně jejich vzájemného porovnání. Dále je prostor věnován jednomu z největších problémů .NET platformy na operačním systému GNU/Linux, a to problému vývoje grafických uživatelských rozhraní. V závěru práce je zdokumentována příprava vývojového prostředí, a to včetně popisu vývoje ukázkové aplikace.

1 .NET platforma a její vlastnosti

Platforma .NET je softwarový spravovaný framework vytvořený společností Microsoft. Cílem tohoto frameworku je usnadnění vývoje různých druhů aplikací, a to od aplikací desktopových a mobilních přes webové stránky a služby až po použití v oblasti IoT (Internet of Things). Na této platformě lze vyvíjet pomocí velkého množství programovacích jazyků, přičemž mezi nejznámější patří C#, F# a Visual Basic. Lze také použít velkého množství editorů zdrojového kódu a stejně tak lze využít rozsáhlého ekosystému rozšiřujících knihoven. .NET platforma není dostupná pouze pomocí jedné implementace, ale těchto vzájemně kompatibilních (dochází k používání stejného API (Application Programming Interface)) implementací existuje vícero. (.„.NET", 2022; .„.NET Framework", 2022; „What Is .NET?", 2022; Seroter, 2018)

Tato platforma má také své specifické vlastnosti. Mezi tyto vlastnosti patří jazyková nezávislost, přenositelnost, výkonnost, bezpečnost, správa paměti, správa knihoven a interoperabilita. Tyto vlastnosti jsou podrobněji pospány dále.

Jazyková nezávislost

Díky specifikaci CTS (Common Type System), která definuje všechny datové typy a způsoby interakce v rámci komponenty CLR (Common Language Runtime) mohou být funkce .NET platformy používány, a dokonce i vytvářeny v mnoha programovacích jazycích. Mezi nejpopulárnější programovací jazyky, které lze použít patří C#, F#, C++/CLI a Visual Basic. Mezi méně známé programovací jazyky patří například Cobra, IronPython a PowerShell. („List of CLI Languages", 2021; .„.NET Framework", 2022)

Přenositelnost

I přes původní úzké zaměření .NET platformy pouze na operační systém Microsoft Windows byla tato platforma navrhována s ohledem na její přenositelnost. Významným krokem k přenositelnosti byla standardizace některých komponent, a to zejména specifikace CLI (Common Language Infrastructure), CTS, programovacího jazyka CIL (Common Intermediate Language) a programovacího jazyka C++/CLI). Tyto standardy byly uznány organizacemi ISO (International Organization for Standardization) a Ecma. Tímto krokem bylo umožněno třetím stranám vyvíjet alternativní, ale zároveň kompatibilní

implementace .NET platformy, a to i pro jiné počítačové architektury a operační systémy, než pro jaké byla tato platforma původně vytvořena. (.„.NET Framework", 2022)

Výkonnost

Pro zajištění co nejvyšší výkonnosti a odezvy aplikací postavených na .NET platformě je využíváno mezipaměti Native Image Cache. Tato mezipamět je společná pro celý operační systém a je v ní uložen již jednou zkompilovaný strojový kód jednotlivých aplikací. Díky této mezipaměti není nutné vytvářet strojový kód opakovaně a tím pádem je opakované spouštění aplikace rychlejší. (.„.NET Framework", 2022)

Pro zrychlení prvního spuštění aplikace je na některých implementacích .NET platformy možné využít nástroje pro ruční AOT (Ahead Of Time) kompilaci. (.„.NET Framework", 2022)

Bezpečnost

Bezpečnost datových typů je zajištěna pomocí standardu CTS a komponenty CLR. V rámci .NET platformy je prováděna kontrola zabraňující špatnému přetypovávání datových typů, kontrola nevhodného volání metod a kontrola velikosti dostupné paměti při přístupu k objektům. (.„.NET Framework", 2022)

Dalším prvkem bezpečnosti je bezpečnostní mechanismus CAS (Code Access Security), který umožňuje omezit oprávnění spuštěného spravovaného kódu. Hlavním účelem tohoto mechanismu je ochrana proti zranitelnostem kódu z externích zdrojů. (.„.NET Framework", 2022)

Správa paměti

Komponenta CLR se mimo jiné také zabývá automatickou správou paměti tedy jejím přidělováním a následným uvolňováním. Paměť je instancím objektů a datovým typům přidělována ze spravované haldy. O uvolňování paměti se stará komponenta garbage collector. Tato komponenta je pravidelně (po dosažení určitého množství obsazené paměti nebo při dostatečné paměťové zátěži) spouštěna na jiném vlákně, než na kterém běží vlastní aplikace. Uvolnění paměti se provádí označením všech nepoužitelných objektů a poté uvolněním jejich přidělené paměti. (.„.NET Framework", 2022)

Nepoužitelným objektem se rozumí takový objekt, na který neexistuje žádný odkaz (přímo nebo přes objektový graf), je nepřístupný nebo nemůže být použit. Takový objekt je

poté označen jako odpad a tím pádem je určen ke sběru pomocí komponenty garbage collector. Pokud neplatí žádná z těchto podmínek je objekt považován za používaný a přidělená paměť mu bude ponechána. („.NET Framework", 2022)

Správa knihoven

Pro správu knihoven v rámci .NET platformy je určen správce balíčků nazvaný NuGet, který je kompatibilní se všemi implementacemi .NET platformy. Jeho hlavním úkolem je umožnit správu a snadné použití rozšiřujících knihoven třetích stran. Tyto balíčky je možné získávat z centrálního repozitáře nebo případně pomocí dalších dodatečných repozitářů.

Interoperabilita

Z důvodu občasné potřeby počítačových systémů využívat funkce aplikací z různých běhových prostředí je tato funkcionalita také podporována. Přístup ke komponentám COM (Component Object Model) je na .NET platformě a operačním systému Microsoft Windows umožněn pomocí speciálního jmenného prostoru *System.Runtime.InteropServices* a *System.EnterpriseServices*. („.NET Framework", 2022)

Přístup k funkcím strojového kódu je pak umožněn pomocí funkce P/Invoke (Platform Invocation Services). Přístup opačným směrem (tedy přístup ze strojového kódu do kódu .NET platformy) je také podporován. („.NET Framework", 2022)

2 .NET platforma a její architektura

V této kapitole jsou popsány jednotlivé části architektury .NET platformy. Mezi hlavní části této architektury patří standard CLI, komponenta CLR a standardní knihovny.

2.1 CLI (Common Language Infrastructure)

CLI je specifikací (standard ISO/IEC 23271 a ECMA 335), která definuje jazykově neutrální platformu pro vývoj a běh aplikací. Díky této specifikace mohou být funkce .NET platformy použity z velkého množství programovacích jazyků. Tato specifikace je rozdělena na čtyři menší části. („Common Language Runtime", 2021; „Overview Of Common Language Infrastructure", 2022)

2.1.1 CTS (Common Type System)

Tato část specifikace definuje datové typy a operace, které jsou podporovány v rámci .NET platformy. Díky této specifikaci je umožněna vzájemná interakce objektů, a to i v rámci různých programovacích jazyků. („Common Language Runtime", 2021; „Overview Of Common Language Infrastructure", 2022)

2.1.2 Metadata

Tato část specifikace definuje jazykově nezávislé informace o struktuře aplikace, a to zejména o jejích třídách a členech těchto tříd. Díky této specifikaci je umožněno správné a bezpečné volání metod. („Common Language Runtime", 2021; „Overview Of Common Language Infrastructure", 2022)

2.1.3 CLS (Common Language Specification)

Tato část specifikace je podmnožinou specifikace CTS a definuje základní pravidla pro spolupráci rozdílných programovacích jazyků. („Common Language Runtime", 2021; „Overview Of Common Language Infrastructure", 2022)

2.1.4 VES (Virtual Execution System)

Úkolem specifikace VES je načíst a spustit aplikace kompatibilní se standardem CLI a také kombinovat samostatně generované části kódu za běhu programu (pro tento účel je využíváno metadat). Součástí je tedy kompilace CIL bajtkódu do strojového kódu. („Common Language Runtime", 2021; „Overview Of Common Language Infrastructure", 2022)

2.2 CLR (Common Language Runtime)

Komponenta CLR je jádrem .NET platformy a slouží jako prováděcí motor běžícího kódu. Svojí podstatou se jedná se o virtuální stroj, ve kterém je poskytováno běhové prostředí a uvnitř kterého jsou spouštěny a řízeny všechny .NET aplikace, a to bez ohledu na to v jakém programovacím jazyce jsou tyto aplikace napsané. Tato komponenta také poskytuje mnoho dalších služeb, mezi které patří například správa paměti včetně garbage collection, obsluha výjimek, kontrola datových typů, zabezpečení a správa vláken. („Introduction to .NET Framework", 2018; .„.NET Framework", 2022)

Kód, který je spouštěn v této komponentě není strojový kód. Aplikace postavené na .NET platformě jsou nejprve kompilovány do CIL bajtkódu. Tomuto bajtkódu se také říká spravovaný kód. (.„.NET Framework", 2022)

Kompilace do strojového kódu je prováděna pomocí komponenty VES a JIT (Just in Time) kompilátoru. Do strojového kódu jsou kompilovány jen ty části aplikace, které jsou aktuálně potřeba, a to až během běhu samotné aplikace. (.„.NET Framework", 2022)

2.3 Standardní knihovny

Součástí .NET platformy je také několik standardních knihoven. Rozsah těchto knihoven je srovnatelný s knihovnami v programovacím jazyce Java a mnohem širší, než u některých jiných programovacích jazyků (například C++). Mezi poskytované funkce patří například manipulace se soubory, XML dokumenty a databázemi, vykreslování GUI (Graphical user interface), kryptografické funkce a funkce pro síťovou komunikaci. Knihovny jsou hierarchicky uspořádány prostřednictvím jmenných prostorů. (.„.NET Framework", 2022; „Standard Libraries (CLI)", 2021)

Podmnožinou standardních knihoven je BCL (Base Class Library). Tato knihovna obsahuje základní funkce a datové typy a zároveň slouží jako základní API pro komponentu CLR. Tato knihovna musí být dostupná ve všech implementacích .NET platformy. („„.NET Framework", 2022; „Standard Libraries (CLI)", 2021; Seroter, 2018)

3 Operační systém GNU/Linux

Linux nebo také GNU/Linux je rodinou open-source Unix-like operačních systémů postavených na Linuxovém jádře. Linuxové jádro, jehož autorem je Linus Torvalds je dostupné pod open-source GPL-2 licencí a bylo poprvé vydáno 17. září roku 1991. Linux je také zároveň jedním z nejznámějších a nejvíce viditelných příkladů úspěchu open-source komunity. („Linux", 2022)

Linux byl původně vyvinut pro osobní počítače postavené na počítačové architektuře x86, ale později byl naportován na velké množství dalších počítačových architektur (více než jakýkoli jiný operační systém) a to včetně počítačových architektur vestavěných systémů, kde je operační systém obvykle součástí samotného firmwaru. Na těchto systémech je Linux obvykle modifikován podle specifických požadavků využití daného vestavěného systému. Mezi nejobvyklejší vestavěné zařízení, které využívají Linux patří například routery, technologie chytrých domů, televize, automobily, digitální videokamery a chytré hodinky. („Linux", 2022)

Linux je také nejrozšířenějším univerzální operačním systémem, a to zejména díky operačnímu sytému Android, jehož součástí je také Linuxové jádro. V oblasti serverů je Linux také dominantním operačním systémem (přes 96,4 % z 1 000 000 nejnavštěvovanějších webů je hostováno právě na tomto operačním systému). V kategorii sálových počítačů dosahuje Linux dokonce 100% podílu (mezi 500 nejvýkonnějšími sálovými počítači od listopadu roku 2017). O něco horší je situace v kategorii desktopových počítačů, kde dosahuje pouze 2,3% podílu. Jistým protikladem v této kategorii jsou Chromebooky s operačním systémem ChromeOS (který je nestandardní distribucí Linuxu), které dosahují téměř 20% podílu v oblasti notebooků v ceně pod 300 $ a dominují tak vzdělávací trh v USA. („Linux", 2022)

3.1 Linuxová distribuce

Linuxová distribuce je operační systém skládající se z kolekce aplikací, Linuxového jádra a ve většině případů správce balíčků. Standardní Linuxová distribuce obvykle skládá z Linuxového jádra, nástrojů, knihoven, dokumentace a dalších aplikací organizace GNU,

okenního systému (nejčastěji okenní systém X nebo Wayland), správce oken a desktopového prostředí. („Linux Distribution", 2022)

Využití ekosystému organizace GNU ovšem není nezbytné a je možné využít různých alternativních implementací základních systémových nástrojů. Mezi nejznámější a nejrozšířenější alternativní implementace základních systémových nástrojů patří aplikace BusyBox a také alternativní standardní knihovna programovacího jazyka C musl (zde dochází ke značnému ovlivnění kompatibility aplikací). Tato možnost volby vede ke vzniku jisté kontroverznosti názvu GNU/Linux.

Většina aplikací, které jsou součástí Linuxových distribucí je dostupná pod open-source licencí, a to jak ve formě spustitelných souborů, tak ve formě zdrojového kódu. Naproti tomu většina distribucí obsahuje také binární bloby (ty jsou potřebné pro některá hardwarová zařízení), které jsou ve většině případů dostupné pouze pod proprietární licencí ve formě binárních souborů. Jen malé množství aplikací je vyvíjeno samotnými Linuxovými distribucemi, většina aplikací je vyvíjena v rámci široké komunity. Aplikace, které jsou součástí distribucí bývají obvykle upraveny pro specifické potřeby dané distribuce a poté jsou správci balíčků zabaleny do jednotlivých balíčků. Tyto balíčky jsou poté distribuovány z centrálních repozitářů dostupných na internetu. („Linux Distribution", 2022)

Existuje okolo tisíce Linuxových distribucí. Tyto Linuxové distribuce se často zaměřují na určité kategorie výpočetní techniky, a to například na desktopy, servery, mobilní telefony nebo minimální prostředí určená pro vestavěné systémy. Případně mohou být specializovány pouze na určité funkce například telefonní ústředny, firewally nebo emailové servery. Mezi nejznámější komerčně podporované distribuce patří Fedora, openSUSE a Ubuntu. Mezi nejznámější plně komunitní distribuce patří například Debian, Slackware, Gentoo a Arch Linux. („Linux Distribution", 2022)

4 Implementace .NET platformy

Díky standardizaci velké části .NET platformy a její poměrně velké popularitě netrvalo dlouho a došlo ke vzniku většího množství alternativních implementací, a to velmi často pro jiné počítačové architektury a operační systémy než pro jaké, byla dostupná původní implementace. Některé tyto implementace se staly v určitých kruzích velmi oblíbenými, a to i přes existující obavy z možnosti vzniku patentových sporů, neexistujících standardů pro některé části .NET platformy (znesnadnění vývoje a udržování kompatibility) a existenci některých charakteristik specifických pro operační systém Microsoft Windows.

Mezi nejvýznamnější implementace patří původní implementace .NET platformy nazvaná .NET Framework a její nástupce multiplatformní implementace nazvaná .NET (Core). Obě tyto implementace pochází od společnosti Microsoft. Na Linuxu a macOS se stala velmi populární implementací multiplatformní a open-source implementace Mono. Právě tyto implementace a také některé další jsou podrobněji popsány v této kapitole.

4.1 .NET Framework

.NET Framework je původní a proprietární implementace .NET platformy. Jediným podporovaným operačním systémem je Microsoft Windows a mezi podporované počítačové architektury patří IA-32, x86-64, a ARM. Tato implementace byla převládající implementací standardu CLI, a to až do jejího nahrazení implementací .NET (Core). Tato implementace je vyvíjena (v současné době dochází již pouze k opravám bezpečnostních chyb a nedochází k přidávání nových funkcí) společností Microsoft. Hlavním cílem této implementace je usnadnění vývoje aplikací pro platformu Microsoft Windows. (Metzgar, 2018; „„.NET Framework", 2022)

Historie

Vývoj implementace .NET Framework započal na konci 90. let 19. století pod původním názvem NGWS (Next Generation Windows Services). K vydání první předběžné verze došlo na počátku roku 2000 a k vydání první stabilní verze došlo 14. února roku 2001. Ačkoliv byla tato implementace zpočátku vyvíjena jako proprietární software (později došlo k úpravě vývojového modelu) probíhala uvnitř společnosti Microsoft ve spolupráci se společností Intel snaha o standardizaci některých částí této platformy, a to včetně programovacího jazyka C#. K této standardizaci došlo v prosinci roku 2001 a to u organizace

Ecma a později v dubnu roku 2003 došlo ke standardizaci u organizace ISO. V dubnu roku 2019 byla vydána verze .NET Framework 4.8, která se stala poslední verzí této implementace. Tato verze je nadále podporována pouze v rámci bezpečnostních a stabilitních záplat. Žádné další změny nejsou pro tuto implementaci plánovány. (Metzgar, 2018; .„.NET Framework", 2022; Seroter, 2018)

Specifické vlastnosti implementace

.NET Framework byl původní implementací CLI standardu a je složen ze dvou hlavních komponent: rozsáhlé standardní knihovny FCL a komponenty CLR. Kromě základní knihovny BCL obsahuje standardní knihovna FCL také další významné knihovny, mezi které patří například webový framework ASP.NET pro tvorbu webových aplikací, sady knihoven pro tvorbu grafického rozhraní Windows Forms a WPF (Windows Presentation Foundation) a také další rozšiřující knihovny například LINQ (Language Integrated Query), WCF (Windows Communication Foundation), Workflow Foundation a ADO.NET. (.„.NET Framework", 2022)

4.2 .NET (Core)

.NET (Core) je open-source implementace .NET platformy napsána v programovacích jazycích C++ a C#, která je dostupná pod MIT licencí. Mezi podporované operační systémy patří Microsoft Windows, Linux a macOS a mezi podporované počítačové architektury patří IA-32, x86-64 a ARM. Tato implementace je multiplatformním nástupcem původního .NET Frameworku a je také vyvíjena převážně společností Microsoft s přispěním komunity pod záštitou nadace .NET Foundation. (.„.NET", 2022)

Miguel de Icaza – význačný inženýr společnosti Microsoft popsal .NET Core jako: „*Přepracovanou verzi .NET platformy, která je postavená na zjednodušených verzích standardních knihoven.*“ („Microsoft Open Sources .NET and Mono - Miguel de Icaza", 2014)

Immo Landwerth – programový manažer společnosti Microsoft uvedl že: „*.NET Core bude základem pro všechny budoucí .NET platformy.*“ (Immo Landwerth, 2014)

Historie

Vývoj implementace .NET (Core) byl oznámen společností Microsoft 12. listopadu roku 2014. Tato implementace vznikla jako výsledek snahy společnosti Microsoft rozšířit .NET platformu o multiplatformní podporu (zejména pro operační systémy Linux a macOS) a sjednotit roztříštěnou .NET platformu. Dalším cílem bylo zapojit do vývoje této platformy další společnosti a také komunitu. Mezi vývojové milníky implementace .NET (Core) patří:

.NET Core 1 (vydáno 27. června roku 2016): Vydání první stabilní verze

.NET Core 2 (vydáno 14. srpna roku 2017): Přidání podpory pro musl, alternativní standardní knihovnu programovacího jazyka C (používána na populární kontejnerové distribuci Alpine Linux)

.NET Core 3 (vydáno 23. září roku 2019): Přidána podpora pro vývoj grafických aplikací na operačním systému Microsoft Windows (sady knihoven Windows Forms, WPF a UWP (Universal Windows Platform)), výrazné zvýšení výkonu v rámci standardní knihovny

.NET 5 (vydáno 10. listopadu roku 2020): Odstranění přídomku „Core“ a přeskočení verze 4 z důvodu konfliktu verzí s implementací .NET Framework, .NET oficiálně nahrazuje původní implementaci .NET Framework, přidána nativní podpora pro počítačovou architekturu ARM (dříve emulováno z počítačové architektury x86)

.NET 6 (vydáno 8. listopadu roku 2020): Aktuální verze v době psaní této práce

(.„.NET", 2022; .„.NET and .NET Core Official Support Policy", 2022)

Specifické vlastnosti implementace

Implementace .NET (Core) je kompatibilní se standardem CLI a skládá ze dvou hlavních komponent: CoreCLR a CoreFX. CoreCLR a CoreFX jsou srovnatelné s komponentou CLR a standardní knihovnou FCL původní implementace. CoreCLR implementuje specifikaci VES a je kompletním virtuálním strojem a běhovým prostředím. Součástí CoreCLR je také JIT kompilátor nazvaný RyuJIT. Konzolová implementace standardních knihoven CoreFX sdílí část API s původní implementací .NET Framework, ale také přidává některá vlastní API. Součástí této implementace je také alternativa nástroje Ngen (Native Image Generator) z původní implementace. (.„.NET", 2022)

Mezi podporované multiplatformní projekty této implementace patří: webové aplikace (framework ASP.NET), konzolové aplikace, GUI aplikace ale pouze na operačním systému Microsoft Windows a rozšiřující knihovny. („.NET", 2022)

Součástí této implementace je také konzolový nástroj určený k základní správě .NET (Core) platformy a aplikací na ní postavených. („.NET", 2022)

4.3 Mono

Mono je open-source implementace .NET platformy rozšířená o vlastní přidané funkce, která je dostupná pod MIT licencí. Tato implementace je napsána v programovacích jazycích C, C# a XML. Mezi podporované operační systémy patří Microsoft Windows, Linux, macOS, IBM AIX a IBM i a mezi podporované počítačové architektury IA-32, x64, IA-64, ARM, MIPS, PowerPC, SPARC, S390. Tato implementace byla původně vyvíjena společností Ximian (později dceřiná společnost společnosti Novell) a v současné době je vyvíjena společností Xamarin (dceřiná společnost společnosti Microsoft) pod záštitou nadace .NET Foundation. Hlavním cílem této implementace je zpřístupnění .NET platformy na alternativních operačních systémech. Tato implementace je také základem pro projekt Xamarin. Je tedy základem pro vývoj aplikací určených pro mobilní operační systémy Android a iOS při použití stejné kódové základny. („Mono (Software)", 2021)

Historie

Po oznámení vzniku implementace .NET Framework a standardizace specifikace CLI začal Miguel de Icaza ze společnosti Ximian zkoumat přínosy této platformy. Poté co dospěl k pozitivním závěrům začal zvažovat možnosti vývoje Linuxové implementace této platformy. Poté co usoudil, že jeho tým nebude schopný tento vývoj zvládnout oznámil 19. července roku 2001 vznik otevřeného projektu Mono. První stabilní verze této implementace byla vydána po třech letech vývoje a to 30. června roku 2004. V této době také došlo ke změně zaměření projektu, a to do od frameworku určeného pro vývoj desktopových aplikací k frameworku pro vývoj aplikací všech druhů určených pro všechny druhy systémů a se široku podporou počítačových architektur a operačních systémů. („Mono (Software)", 2021)

Po zakoupení společnosti Ximian společností Novell (v roce 2003) došlo k největšímu rozšíření a rozvoji této implementace. Naopak po zakoupení společnosti Novell

společností Attachmate (v dubnu roku 2011) a následném hromadném propouštění byla budoucnost této implementace velmi nejistá. Dne 16. května roku 2011 oznámil Miguel de Icaza (tedy původní autor) obnovení vývoje pod nově založenou společností Xamarin složenou z členů původního týmu. („Mono (Software)", 2021)

Od verze Mono 4.0 dochází ke sdílení částí zdrojového kódu s implementací .NET (Core). Dne 18. března roku 2016 byla společnost Xamarin zakoupena společností Microsoft a 31. března roku 2016 došlo k uvolnění všech komponent tohoto projektu pod open-source MIT licencí. (Metzgar, 2018; „Mono (Software)", 2021)

Specifické vlastnosti implementace

Tento projekt vznikl jako open-source implementace .NET Frameworku pro alternativní operační systémy a alternativní počítačové architektury. Cílem tohoto projektu tedy bylo zachování kompatibility s původní implementací a rozšíření jejích možností a také usnadnění možností portování aplikací určených pro operační sytém Microsoft Windows. („Mono (Software)", 2021)

Kromě základních komponent obsahuje tato implementace také rozšiřující komponenty pro lepší integraci s Linuxovými desktopy (zejména pro desktopové prostředí GNOME) a komponenty Microsoft kompatibility. V rámci komponent pro integraci s Linuxovými desktopy vznikly různé sady knihoven pro různé účely. Mezi nejdůležitější z nich patří: sada knihoven Gtk# pro tvorbu GUI pomocí sady knihoven GTK, sada knihoven Gecko# pro práci s vykreslovacím jádrem Gecko webového prohlížeče Mozilla Firefox, sada knihoven pro integraci s POSIX standardem, sada knihoven pro správu databází, sady knihoven pro zabezpečení a sady knihoven pro práci s XML dokumenty. V rámci komponent Microsoft kompatibility byly vyvíjeny různé sady knihoven pro usnadnění portování aplikací původně napsaných pro operační systém Microsoft Windows. Mezi tyto komponenty patří alternativní implementace sad knihoven ADO.NET, ASP.NET a Windows Forms. („Mono (Software)", 2021)

Podporované funkce této implementace jsou v současné době na úrovni .NET Frameworku verze 4.7 s omezeními v podobě chybějící podpory WPF a Windows Workflow Foundation a také pouze částečné podpory pro WCF a ASP.NET. („Mono (Software)", 2021)

Xamarin.iOS a Xamarin.Android jsou implementace .NET platformy založené na projektu Mono, které jsou určené pro operační systémy Android a iOS. Původně byly tyto implementace dostupné pouze pod proprietární licencí, ale po koupi společnosti Xamarin společností Microsoft došlo k jejich přelicencování a následnému uvolnění pod open-source MIT licencí. Specifické pro tyto implementace je, že zdrojový kód aplikací je kompilován přímo do strojového kódu cílové počítačové architektury. („Mono (Software)", 2021; „Xamarin App Development with Visual Studio | Visual Studio", 2022)

4.4 .NET Micro Framework

Tato implementace .NET platformy byla určena pro zařízení s extrémně nízkými systémovými prostředky tedy zejména pro vestavěné systémy. Tato implementace obsahovala zmenšenou verzi CLR s podporou pouze pro programovací jazyk C#. Ladění aplikací bylo možné provádět pomocí softwarového emulátoru či přímo na fyzickém hardwaru, oboje při použití IDE (Integrated development environment) Microsoft Visual Studio. Součástí byla vlastní sada knihoven pro tvorbu GUI, která byla odvozena ze sady knihoven WPF. Součástí této implementace byly také další sady knihoven cílené na vývoj aplikací určených pro běh na vestavěných systémech. Tento projekt již není v aktivním vývoji. („.NET Micro Framework", 2022)

4.5 Portable.NET (DotGNU)

Cílem tohoto projektu bylo vytvoření open-source implementace standardu CLI, části standardní knihovny FCL a kompilátoru pro programovací jazyk C#. Podporováno bylo velké množství procesorových architektur a operačních systémů. Poslední stabilní verze byla vydána v roce 2009 a v prosinci roku 2012 byl tento projekt rozpuštěn. („DotGNU Portable.NET", 2022)

4.6 CrossNet

Tato implementace implementovala standard CLI a některé části standardní knihovny FCL, a to pod open-source MIT licencí. Specifické pro tuto implementaci je, že nevyužívala kompilace do CIL bajtkódu, ale zdrojový kód byl přímo kompilován do strojového kódu cílové počítačové architektury. Účelem této specifické vlastnosti bylo zpřístupnění .NET platformy i na počítačových architekturách pro které neexistoval JIT kompilátor. Tento

projekt již není v aktivním vývoji a jeho poslední verze byla vydána 14. října roku 2007. („CrossNet - Web archive", 2022)

4.7 Microsoft Silverlight

Microsoft Silverlight byla implementace .NET platformy určená pro běh v rámci webových prohlížečů (prohlížeč Internet Explorer) a to pomocí externích modulů. Cílem této implementace bylo umožnění vývoje a běhu webových aplikací se zaměřením na podporu multimédií, tvorbu grafiky a tvorbu animací. Aplikace bylo možné vyvíjet pomocí programovacích jazyků a vývojářských nástrojů podporujících standard CLI. Podpora pro tento produkt byla ukončena 12. října roku 2021. („Microsoft Silverlight", 2022)

4.8 Moonlight

Moonlight byla alternativní open-source implementace aplikačního frameworku Microsoft Silverlight vyvinutá v rámci projektu Mono. Tato implementace byla poprvé vydána 20. ledna 2009 společně s verzí Mono 1.9. Tato implementace podporovala Silverlight API ve verzích 1.0, 2.0 a 3.0. Vývoj byl zastaven 29. května roku 2012 a to z důvodu nedostatečného zájmu ze strany komunity a omezenému rozšíření této technologie. („Moonlight (Runtime)", 2021)

4.9 Problém softwarových patentů

I přes standardizaci většiny komponent .NET platformy a tím pádem přijetí podmínek organizací ISO a ECMA o možnosti využívání všech nutných patentů souvisejících s těmito standardy panovali zejména v open-source komunitě obavy ze vzniku patentových sporů. Hlavním důvodem byla chybějící standardizace některých komponent potřebných pro vznik úplných alternativních implementací. Mezi tyto nestandardizované části patřily například sada knihoven Windows Forms, sada knihoven ADO.NET a webový framework ASP.NET. (.„.NET Framework", 2022; „Will open source get snagged in .Net? - ZDNet Asia News", 2011)

Ke zmírnění těchto obav došlo 3. října roku 2007 kdy společnost Microsoft oznámila, uvolnění zdrojových kódů standardních knihoven implementace .NET Framework verze 3.5 pod open-source Ms-RSL licencí. K vlastnímu uvolnění těchto zdrojových kódu došlo 16. ledna roku 2008. Uvolněny byly zdrojové kódy pro standardní knihovnu BCL, webový

framework ASP.NET, sadu knihoven ADO.NET, sadu knihoven Windows Forms, sadu knihoven WPF, a sadu knihoven pro práci s XML. Dále byl dán příslib uvolnění dalších sad knihoven LINQ, WCF a Windows Workflow Foundation. Další velmi významné zmírnění těchto obav proběhlo v listopadu roku 2014 aktualizací patentových ujednání. Součástí tohoto kroku bylo zmírnění podmínek vztahujících se ke standardizovaným částem .NET platformy (nezávislost na verzi standardu při dodržení kompatibility se standardem CLI) a ke zpřístupnění všech patentů pro ostatní komponenty .NET platformy (pro ty zdokumentované na MSDN (Microsoft Developer Network). Tento krok plně zlegalizoval většinu alternativních implementací (do té doby nejasný stav) a umožnil alternativním implementacím implementovat moderní vlastnosti .NET platformy. Další postup nastal 31. března roku 2016 kdy došlo ke kompletnímu přelicencování všech komponent projektu Mono (včetně proprietárních) a to pod open-source MIT licenci. Dalším krokem bylo uvolnění a přelicencování zdrojových kódů sad knihoven Windows Forms, WPF a WinUI (Windows UI Library) pod open-source MIT licenci. („Mono (Software)", 2021; .„.NET Framework", 2022)

Díky těmto krokům je tedy v současné době možné vytvořit plně open-source implementaci .NET platformy, a to bez obav z patentových sporů.

5 .NET platforma na operačním systému GNU/Linux

Ačkoliv byla .NET platforma mezi vývojáři poměrně populární a velké množství aplikací bylo postaveno právě na této platformě, původní implementace s názvem .NET Framework (uvedena 13. února roku 2002) byla dostupná pouze pro operační systém Microsoft Windows a na jiné operační systémy se nikdy nativně nedostala (byla dostupná pouze při využití vrstvy kompatibility Wine na Linuxu a macOS).

První možnost pro nativní využití technologií .NET platformy na Linuxu se tak objevila až se vznikem projektu DotGNU, a to v lednu roku 2001. Tato implementace ovšem nikdy nedosáhla příliš velké popularity ani rozšíření a byla tak velmi rychle zastíněna a nahrazena jiným projektem s podobným cílem, a to projektem Mono (založen 30. června roku 2004). Tento projekt se posléze stal dominantní implementací .NET platformy pro Linux a macOS.

Dominantní postavení tohoto projektu ale netrvalo věčně a bylo narušeno příchodem nové multiplatformní a open-source implementace od původního autora .NET platformy tedy společnosti Microsoft. Tato nová implementace byla pojmenována .NET (Core) a byla představena 27. června roku 2016.

I přesto, že jsou obě implementace (.NET (Core) a Mono) stále v aktivním vývoji, a dokonce mezi nimi dochází ke sdílení částí zdrojového kódu (díky kompatibilním licencím), množství aplikací využívajících implementaci Mono se neustále snižuje a množství aplikací využívajících implementaci .NET (Core) se neustále zvyšuje. Výjimkou jsou některé aplikace využívající rozšířené funkce .NET Frameworku a aplikace, které byly a jsou vyvíjeny primárně pro operační systém Microsoft Windows a až později byly naportovány na jiné operační systémy. Pro tyto aplikace poskytuje implementace Mono úplnější kompatibilitu s původním .NET Frameworkem a také poskytuje některé rozšiřující funkce, které nejsou v implementaci .NET (Core) dostupné.

V současné době je tedy možné tvrdit, že implementace .NET (Core) nahrazuje a vytlačuje implementaci Mono. Tímto také dochází ke sjednocování roztříštěné .NET platformy pod jednu společnou implementaci.

5.1 Rozšíření

Na Linuxu nebyla .NET platforma nikdy příliš oblíbená a ani nedosáhla příliš velkého rozšíření. Jednou z překážek pro větší rozšíření této platformy v rámci Linuxové komunity byla obava ohledně softwarových patentů, kterými byla .NET platforma zatížena a tím pádem legálnost alternativních implementací. Další překážkou byla a stále je značná nedůvěra Linuxové komunity vůči společnosti Microsoft, a to zejména kvůli jejímu historickému anti-kompetitivnímu chování.

K největšímu rozšíření .NET platformy na Linuxových operačních systémech došlo mezi lety 2003 a 2010 kdy do oblasti Linuxových serverů a desktopu vstoupila společnost Novell. Společnost Novell v této době také zakoupila společnost Ximian. Společnost Ximian se zabývala vývojem desktopových aplikací pro Linux a také stála za projektem Mono a byla velkým podporovatelem projektu GNOME (desktopové prostředí). Díky této akvizici začala být implementace Mono využívána v některých projektech společnosti Novell a tato společnost také financovala vývoj některých aplikací postavených na této implementaci. V této době se také implementace Mono stala součástí výchozí instalace některých významných Linuxových distribucí (například Linuxové distribuce Ubuntu a openSUSE). Zároveň také nezůstalo pouze u samotné implementace. V rámci projektu GNOME bylo také několik Mono aplikací přidáno do seznamu doporučovaných aplikací (například výchozím přehrávačem hudby se stala aplikace Banshee media player).

Tento úspěch ovšem neměl dlouhého trvání a byl rychle zvrácen po prodeji společnosti Novell společnosti Attachmate. Po této akvizici omezila společnost Novell své investice v oblasti Linuxového desktopu a tím pádem došlo k poklesu vývojové aktivity v projektech Mono a GNOME. Následným efektem bylo, že vývoj některých desktopových Linuxových aplikací (často postavených na implementaci Mono) byl zastaven. Netrvalo dlouho a implementace Mono a aplikace postavené na této implementaci přestaly být součástí výchozích instalací Linuxových distribucí (v Linuxové distribuci Ubuntu bylo Mono součástí výchozí instalace mezi verzemi 10.04 a 12.04, tedy mezi lety 2010 a 2012). Rozšířenost .NET platformy na Linuxu v této době tedy značně poklesla.

Ale situace se pomalu začíná obracet k lepšímu. V současné době se .NET platforma v Linuxu začíná opět rozšiřovat, a to zejména díky vzniku nové a sjednocující implementace

.NET (Core), vyřešení problému softwarových patentů a díky webovému frameworku ASP.NET. Vývoj desktopových aplikací postavených na .NET platformě je však stále velmi neobvyklý, a to také z důvodu absence oficiální sady knihoven pro tvorbu GUI.

5.2 Linuxové aplikace postavené na .NET platformě

V této podkapitole jsou představeny některé významnější Linuxové aplikace, které byly nebo jsou postavené na platformě .NET.

Aplikace v aktivním vývoji

Pinta: Jedná se o multiplatformní open-source bitmapový editor obrázků podobný aplikaci Paint.NET dostupný pod MIT licencí. Poslední verze této aplikace byla vydána 13. ledna roku 2022. Tato aplikace využívá platformu .NET a sadu knihoven Gtk#. („Pinta (Software)", 2022)

KeePass: Jedná se o multiplatformní open-source správce hesel, který je dostupný pod GPL-2 licencí. Poslední verze této aplikace byla vydána 9. ledna roku 2022. Tato aplikace využívá platformu Mono. („KeePass", 2022)

SparkleShare: Jedná se o multiplatformní cloudové úložiště a aplikaci pro synchronizování souborů. Obě tyto části jsou dostupné pod GPL-3 licencí. Jako backend úložiště je využíván verzovací systém Git. Poslední verze této aplikace byla vydána 28. listopadu roku 2020. („SparkleShare", 2021)

Gbrainy: Jedná se o multiplatformní hru pro cvičení paměti dostupnou pod GPL-2 licencí. Poslední verze této aplikace byla vydána 6. ledna roku 2022. Tato aplikace využívá platformu Mono a sadu knihoven Gtk#. („Apps/gbrainy - GNOME Wiki!", 2022)

Herní engine Unity: Platforma Mono je součástí herního enginu Unity. Jejím účelem je zajištění podpory pro skriptování pomocí programovacího jazyka C#. („Mono (Software)", 2021)

Terraria: Jedná se o multiplatformní počítačovou RPG (Role-playing game) hru napsanou v programovacím jazyce C# jejíž Linuxový port využívá implementace Mono.

Aplikace bez aktivního vývoje

Banshee media player: Jednalo se o multiplatformní open-source multimediální přehrávač dostupný pod MIT licencí. Tato aplikace využívala platformu Mono a knihovnu Gtk# a multimediální platformu GStreamer pro kódování a dekódování různých multimediálních formátů. Po určitou dobu sloužila tato aplikace jako výchozí přehrávač hudby v Linuxových distribucích Ubuntu, Linux Mint a openSUSE. Poslední verze této aplikace byla vydána 18. února roku 2014 a její repozitář je archivován. („Banshee (Media Player)", 2022)

GNOME Do: Jednalo se o Linuxový open-source spouštěč aplikací dostupný pod GPL-3 licencí. Poslední verze této aplikace byla vydána 12. listopadu 2014 a v jejím repozitáři neprobíhá žádná aktivita. („GNOME Do", 2022)

Beagle: Jednalo se o Linuxový open-source vyhledávač informací dostupný pod mixem X11/MIT a Apache licencí. Poslední verze této aplikace byla vydána 26. ledna 2009 a její repozitář je archivován. Tato aplikace využívala platformu Mono a knihovnu Gtk#. („Beagle (Software)", 2022)

Docky: Jednalo se o Linuxový open-source aplikační dock dostupný pod GPL-3 licencí. Poslední verze této aplikace byla vydána 2. září 2015 a v jejím repozitáři neprobíhá žádná aktivita. („Docky in Launchpad", 2022)

F-Spot: Jednalo se o Linuxového open-source správce obrázků dostupného pod MIT licencí. Poslední verze této aplikace byla vydána 19. prosinec 2010 a v jejím repozitáři neprobíhá žádná aktivita. („F-Spot", 2022)

6 Vývojářské nástroje

Vývojářům aplikací postavených na .NET platformě je k dispozici mnoho vývojářských nástrojů, které jim umožňují vývoj právě na této platformě. Pravděpodobně nejznámějšími a nejrozšířenějšími jsou nástroje Visual Studio, Visual Studio pro Mac a Visual Studio Code od společnosti Microsoft. Tyto nástroje mají jedno společné, a to že jsou vyvíjeny společností Microsoft tedy původním autorem .NET platformy.

Ovšem většina těchto nástrojů není dostupná pro Linux. Mezi tyto nedostupné nástroje patří Visual Studio a Visual Studio pro Mac. Tyto dva nástroje sice podporují multiplatformní vývoj, ale lze je provozovat pouze na jejich příslušných operačních systémech. Těmi jsou Microsoft Windows v případě nástroje Visual Studio a macOS v případě nástroje Visual Studio pro Mac. Situace je však odlišná u nástroje Visual Studio Code. Tento nástroj podporuje multiplatformní vývoj, přičemž sám je také multiplatformní aplikací a lze jej tedy provozovat na Linuxu.

Tento nástroj ovšem není jediným použitelným nástrojem pro vývoj aplikací postavených na .NET platformě, který lze provozovat na Linuxu. V této podkapitole jsou popsány a porovnány další vývojářské nástroje, které mohou být k tomuto účelu použity. Nástroje zde popsané jsou buďto doporučované přímo společností Microsoft nebo jsou autorem této práce považovány za významné a jsou vyvíjeny některou z významných linuxových organizací.

6.1 .NET CLI

.NET CLI je multiplatformním nástrojem pro správu prostředí implementace .NET (Core). Jeho hlavním účelem je vytváření, obnovování, sestavování, spouštění a publikování aplikací postavených na této implementaci, a to prostřednictvím konzole. Tento nástroj také umožňuje správu NuGet balíčků tedy rozšiřujících knihoven. Podporovanými operačními systémy jsou Microsoft Windows, Linux a macOS. Tento nástroj je na všech podporovaných operačních systémech distribuován společně s .NET (Core) SDK. (MSDN, 2022)

6.2 OmniSharp

OmniSharp je sadou nástrojů, knihoven a rozšíření navržených pro integraci do již existujících editorů zdrojového kódu. Tato sada je dostupná pod open-source MIT licencí. Tento projekt byl založen v roce 2012 a je vyvíjen převážně komunitou jejíž součástí je několik zaměstnanců společnosti Microsoft a také za podpory nadace .NET Foundation. Hlavním účelem této sady nástrojů je vylepšení podpory a možností multiplatformního vývoje aplikací postavených na libovolné implementaci .NET platformy (se zaměřením na implementaci .NET (Core)) a programovacím jazyce C#, a to v již existujících editorech zdrojového kódu na všech operačních systémech. Rozšíření pro integraci s editory zdrojového kódu jsou dostupná pro Visual Studio Code, Vim, Emacs, Atom, Sublime Text a Brackets. Podporované funkce této sady nástrojů se u jednotlivých editorů zdrojového kódu liší, ale obvykle je podporováno inteligentní dokončovaní kódu, správa odkazů a automatické formátování zdrojového kódu. („OmniSharp", 2022; „OmniSharp - .NET and IntelliSense on any platform with your editor of choice", 2022)

6.3 Visual Studio Code

Visual Studio Code je multiplatformní editor zdrojového kódu postavený na frameworku Electron. Tento editor je dostupný pod open-source MIT licencí a zároveň také pod proprietární licencí, a to v případě oficiálních sestavení distribuovaných společností Microsoft. Tento nástroj je vyvíjen převážně společností Microsoft s přispěním komunity. Mezi podporované operační systémy patří Microsoft Windows, Linux a macOS a mezi podporované počítačové architektury IA-32, x86-64 a ARM. Hlavním účelem tohoto editoru je umožnění multiplatformního vývoje aplikací. Podle dotazníku Stack Overflow 2021 Developer Survey, kterého se zúčastnilo téměř 82 000 respondentů dosáhl tento nástroj téměř 70% podílu a stal se tak nejpopulárnějším vývojářským nástrojem v tomto dotazníku. („Stack Overflow Developer Survey 2021", 2021; „Visual Studio Code", 2022)

Funkce a vlastnosti

Pro plnou podporu .NET platformy a programovacího jazyka C# je nutné nainstalovat oficiální rozšíření *C#*, které je vyvíjené společností Microsoft. Toto rozšíření je založené na sadě nástrojů OmniSharp a je dostupné v centrálním repozitáři VS Code Marketplace. Dále zmíněné funkce a vlastnosti obsahují také funkcionalitu přidanou pomocí rozšíření *C#*.

Podporované jazyky: Mezi podporované jazyky patří většina běžných jazyků, a to zejména C++, C#, CSS, Dart, Dockerfile, F#, Go, HTML, Java, JavaScript, JSON, Julia, Less, Markdown, PHP, PowerShell, Python, R, SCSS, T-SQL a TypeScript.

Podporované implementace .NET platformy: Mezi plně podporované implementace patří .NET (Core) a Mono. Podpora pro .NET Framework je pouze částečná.

Podporované typy projektů .NET platformy: Mezi podporované typy projektů patří všechny projekty implementace .NET (Core), projekty MSBuild a C# skripty (CSX). Podpora pro projekty implementace .NET Framework je pouze částečná, nejsou podporovány například projekty ASP.NET MVC a projekty herního enginu Unity.

Ladění: Ladění je podporováno pro platformy Node.js, .NET (Core) a v omezené míře pro .NET Framework (pouze 64bitové aplikace při využití ladicích symbolů ve formátu Portable PDBs). Podporu pro ladění implementace Mono lze přidat pomocí rozšíření *Mono debugging*.

Analýza zdrojového kódu: Podporována je analýza zdrojového kódu během jeho psaní. Při nalezení chyby nebo nesprávnosti dojde k jejímu zvýraznění a také k navržení oprav nebo doporučení.

Asistence při psaní zdrojového kódu: Podporováno je zvýraznění syntaxe, párování závorek a dočasné skrývání kódu neboli code folding. K dispozici jsou také uživatelsky konfigurovatelné úryvky neboli configurable snippets, automatická refaktorizace a formátování zdrojového kódu včetně inteligentního doplňování částí kódu.

Verzování: Verzování zdrojového kódu je podporováno formou integrace s verzovacími nástroji Git, Apache Subversion a Perforce. Tato integrace umožňuje zobrazovat změny v právě otevřeném projektu, vytvářet nové repozitáře a zadávat požadavky push a pull.

Rozšiřitelnost: Podporované funkce a vlastnosti tohoto nástroje lze rozšířit pomocí dalších rozšíření. Tyto rozšíření jsou distribuovány pomocí centrálního repozitáře nazvaného VS Code Marketplace. Možnosti rozšíření jsou velmi široké je možné například přidat nové překlady uživatelského rozhraní, přidat podporu pro další programovací jazyky, přidávat nové grafické motivy (vzhled nástroje), instalovat ladící programy, instalovat programy pro

statickou analýzu, přidávat nové lintery pomocí protokolu LSP (Language Server Protocol) a přidávat podporu pro vzdálená datová úložiště.

Další vlastnosti: Tento nástroj nevyužívá projektového systému, místo toho umožňuje svým uživatelům otevřít jeden nebo více adresářů, které lze ukládat do pracovních prostorů pro jejich opětovné použití.

(„Visual Studio Code", 2022)

Instalace a konfigurace

Tento nástroj je oficiálně podporován na Linuxových distribucích Debian, Ubuntu, Red Hat, Fedora a SUSE (openSUSE).

Pro instalaci na distribucích Debian a Ubuntu je poskytován balíček ve formátu deb. Pro instalaci na distribucích Red Hat, Fedora a SUSE (openSUSE) je poskytován balíček ve formátu rpm. Tyto balíčky je nutné stáhnout a poté nainstalovat pomocí balíčkovacích systému jednotlivých distribucí. Při instalaci těchto balíčků je také přidán oficiální repozitář, s jehož pomocí bude aktuálnost tohoto nástroje udržována pomocí balíčkovacího systému. Je také možné využít oficiálně podporovaného balíčku ve formátu snap a komunitně udržovaného balíčku ve formátu flatpak.

Praktická použitelnost

Tento nástroj podporuje .NET platformu na dobré úrovni a podle názoru autora může být použit pro vývoj aplikací v reálném prostředí.

6.4 JetBrains Rider

JetBrains Rider je multiplatformní IDE postavené na platformě IntelliJ. Toto vývojové prostředí je dostupné pod proprietární licencí a k jeho používání je nutný JetBrains účet a vlastnictví platné licence. Tento nástroj je vyvíjen společností JetBrains. Mezi podporované operační systémy patří Microsoft Windows, Linux a macOS a podporovanou počítačovou architekturou je x86-64. Hlavním účelem tohoto IDE je umožnění multiplatformního vývoje aplikací postavených na .NET platformě ať už webových stránek, služeb nebo aplikací. Podle dotazníku Stack Overflow 2021 Developer Survey, kterého se zúčastnilo téměř 82 000 respondentů dosáhl tento nástroj téměř 4% podílu. („Rider", 2022; „Stack Overflow Developer Survey 2021", 2021)

Funkce a vlastnosti

Tato aplikace je přímo určena pro vývoj aplikací postavených na .NET platformě. Instalace pluginu pro vylepšení podpory tedy není nutná.

Podporované jazyky: Mezi podporované jazyky patří C#, F#, VB.NET, ASP.NET (ASPX a Razor view engine), XAML, XML, JavaScript, TypeScript, JSON, HTML, CSS, SCSS, LESS, a SQL.

Podporované implementace .NET platformy: Mezi podporované implementace patří .NET (Core), Mono a .NET Framework.

Podporované typy projektů .NET platformy: Mezi podporované typy projektů patří všechny projekty implementace .NET (Core), projekty MSBuild (včetně .NET desktopových aplikací), C# skripty (CSX), projekty herního enginu Unity, projekty platformy Xamarin a webové projekty APS.NET.

Ladění: Ladění je podporováno pro implementace .NET (Core), Mono a .NET Framework s výjimkou UWP aplikací. Součástí je také nástroj pro trasování zásobníku.

Analýza zdrojového kódu: Podporována je analýza zdrojového kódu během jeho psaní, a to v celém projektu tedy i v neotevřených souborech. Při nalezení chyby nebo nesprávnosti dojde k jejímu zvýraznění a také k navržení oprav nebo doporučení. Tyto návrhy je možné aplikovat jednotlivě nebo hromadně.

Asistence při psaní zdrojového kódu: Podporováno je zvýraznění syntaxe, párování závorek, automatický import jmenných prostorů a také dokončování import directives. K dispozici jsou také rychlé informační nápovědy, ikony pro navigaci v rámci dědičnosti neboli gutter icons, automatická refaktorizace a formátování zdrojového kódu včetně inteligentního doplňování částí kódu.

Verzování: Verzování zdrojového kódu je podporováno formou integrace s verzovacími nástroji Git, Apache Subversion, Mercurial, Perforce a TFS (s možností podpory dalších pomocí pluginů). Tato integrace umožňuje zobrazovat místní a vzdálené změny (a ukládat je pro pozdější použití), zobrazovat změny mezi soubory neboli diff a zadávat požadavky push a commit. Součástí je také vizuální nástroj pro řešení konfliktů.

Rozšiřitelnost: Podporované funkce a vlastnosti tohoto nástroje lze rozšířit pomocí dalších pluginů, které jsou kompatibilní s platformami ReSharper a IntelliJ. Kromě pluginů, které jsou součástí instalace jsou k dispozici například pluginy pro podporu dokumentů ve formátu Markdown, .gitignore, a skriptů programovacího jazyka Python.

Další vlastnosti: Tento nástroj podporuje spuštění a debuggování unit testů založených na formátu NUnit, xUnit.net nebo MSTest. Tyto testy lze prohlížet, různými způsoby seskupovat, rozdělovat a zobrazovat jejich výstupy. Další funkcí je práce s SQL databázemi. K databázím se lze připojit, upravovat jejich schémata, upravovat data jednotlivých tabulek, spouštět dotazy a analyzovat schémata pomocí UML diagramů. NuGet balíčky je možné spravovat pomocí integrovaného grafického nástroje. Dostupné je také rozšíření pro dekompilaci.

(„Rider", 2022)

Instalace a konfigurace

Tento nástroj je oficiálně podporován na všech desktopových Linuxových distribucích. Pro instalaci je poskytován archiv ve formátu tar.gz a nástroj JetBrains Toolbox.

Pro instalaci pomocí archivu ve formátu tar.gz je nutné tento archiv stáhnout a rozbalit do vybrané složky (doporučována složka /opt). Při instalaci tímto způsobem nejsou vytvořeny ikony pro spuštění a nástroj nebude automaticky aktualizován.

Pro instalaci pomocí nástroje JetBrains Toolbox je nutné na webových stránkách společnosti JetBrains zvolit možnost instalace pomocí tohoto nástroje. Poté je nutné stáhnout archiv ve formátu tar.gz a rozbalit jej do vybrané složky (doporučována složka /opt) a spustit tento nástroj pomocí příkazu jetbrains-toolbox. Případně je možné využít poskytovaného skriptu pro zautomatizování tohoto procesu. Po spuštění tohoto nástroje je nutné se přihlásit, a to pomocí účtu společnosti JetBrains, vyhledat aplikaci Rider a stisknout tlačítko pro instalaci. Při instalaci tímto způsobem je možné vytvořit ikony pro spuštění a také nastavit automatické aktualizace, oboje prostřednictvím nástroje JetBrains Toolbox.

Praktická použitelnost

Tento nástroj podporuje .NET platformu na velmi dobré úrovni, a dokonce je z velké části srovnatelný s nástrojem Visual Studio, který není pro Linux dostupný. Podle názoru autora může být použit (a je také používán) pro vývoj aplikací v reálném prostředí.

6.5 GNOME Builder

GNOME Builder je IDE dostupné pod open-source GPL-3 licencí. Tento nástroj je vyvíjen převážně organizací GNOME s přispěním komunity. Jediným podporovaným operačním systémem je Linux a mezi podporované počítačové architektury patří x86-64 a ARM. Hlavním účelem tohoto vývojového prostředí je vývoj aplikací založených na platformě GNOME tedy podpora programovacích jazyků C a Rust a sady knihoven GTK. V dotazníku Stack Overflow 2021 Developer Survey, kterého se zúčastnilo téměř 82 000 respondentů nebyl tento nástroj obsažen. („Apps/Builder - GNOME Wiki!", 2022; „GNOME Builder", 2022; „Stack Overflow Developer Survey 2021", 2021)

Funkce a vlastnosti

Pro vylepšení podpory programovacího jazyka C# bylo vyvíjeno komunitní rozšíření *builder-csharp*. Toto rozšíření již není v aktivním vývoji a na nejnovější verzi (v době psaní práce) tohoto nástroje jej již nelze nainstalovat.

Podporované jazyky: Mezi podporované jazyky patří C, C++, Python, Rust, CSS, HTML, JS, JSON, Python, Ruby, SCSS, shell skripty a XML.

Podporované implementace .NET platformy: Částečně podporovaná je pouze implementace Mono.

Podporované typy projektů .NET platformy: Mezi podporované typy projektů patří pouze vlastní typ projektu prázdné a grafické aplikace (využití sady knihoven GTK a GTKSharp) při použití sestavovacího systému Meson.

Ladění: Ladění je podporováno pouze pro implementaci Mono.

Analýza zdrojového kódu: Pro programovací jazyk C# není analýza zdrojového kódu podporována.

Asistence při psaní zdrojového kódu: Pro programovací jazyk C# není asistence při psaní zdrojového kódu podporována.

Verzování: Verzování zdrojového kódu je podporováno formou integrace s verzovacím nástrojem Git. Tato integrace umožňuje zobrazovat změny v právě otevřeném projektu a zadávat požadavky push a pull.

Rozšiřitelnost: Podporované funkce a vlastnosti tohoto nástroje lze rozšířit pomocí dalších pluginů. Tyto pluginy mohou být napsány v programovacích jazycích C, Python 3 a Vala.

Instalace a konfigurace

Tento nástroj je oficiálně podporován na všech desktopových Linuxových distribucích. Tento nástroj se obvykle nachází v oficiálních repozitářích jednotlivých distribucí a lze jej tedy nainstalovat pomocí správce balíčků dané distribuce.

Je také možné využít oficiálně podporovaného balíčku ve formátu flatpak.

Praktická použitelnost

Tento nástroj podporuje .NET platformu velmi omezeně a podle názoru autora by mohl být pro vývoj aplikací v reálném prostředí použit jen velmi obtížně.

6.6 KDevelop

KDevelop je multiplatformní IDE dostupné pod open-source GPL-2 licencí. Tento nástroj je vyvíjen převážně organizací KDE s přispěním komunity. Mezi podporované operační systémy patří Microsoft Windows, Linux a macOS a mezi podporované počítačové architektury x86-64 a ARM. Hlavním účelem tohoto vývojového prostředí je umožnění vývoje aplikací založených na platformě KDE tedy primárně podpora programovacího jazyka C++ a sady knihoven Qt. V dotazníku Stack Overflow 2021 Developer Survey, kterého se zúčastnilo téměř 82 000 respondentů nebyl tento nástroj obsažen. („KDevelop", 2022)

Funkce a vlastnosti

Pro vylepšení podpory programovacího jazyka C# a platformy .NET není dostupné žádné rozšíření.

Podporované jazyky: Mezi podporované jazyky patří C, C++, Python, PHP, Java, Fortran, Ruby, Ada, Pascal, SQL a shell skripty.

Podporované implementace .NET platformy: Nejsou podporovány žádné implementace .NET platformy.

Podporované typy projektů .NET platformy: Nejsou podporovány žádné typy .NET projektů.

Ladění: Ladění .NET platformy není podporováno.

Analýza zdrojového kódu: Pro programovací jazyk C# není analýza zdrojového kódu podporována.

Asistence při psaní zdrojového kódu: Pro programovací jazyk C# je dostupná pouze základní asistence, tedy zvýrazňování kódu a párování závorek.

Verzování: Verzování zdrojového kódu je podporováno formou integrace s verzovacími nástroji CVS, Apache Subversion, Perforce, ClearCase, Git, Mercurial a Bazaar. Tato integrace umožňuje zobrazovat změny v právě otevřeném projektu a zadávat požadavky push a pull.

Rozšiřitelnost: Podporované funkce a vlastnosti tohoto nástroje lze rozšířit pomocí dalších pluginů.

Instalace a konfigurace

Tento nástroj je oficiálně podporován na všech desktopových Linuxových distribucích. Tento nástroj se obvykle nachází v oficiálních repozitářích jednotlivých distribucí a lze jej tedy nainstalovat pomocí správce balíčků dané distribuce.

Je také možné využít oficiálně podporovaného balíčku ve formátu snap a také oficiálně podporovaného balíčku ve formátu flatpak.

Praktická použitelnost

Tento nástroj nepodporuje .NET platformu. Pro vývoj aplikací v reálném prostředí tedy nemůže být použit.

6.7 MonoDevelop (historické)

MonoDevelop případně Xamarin Studio bylo multiplatformní IDE dostupné pod kombinací open-source LGPL a MIT X11 licencí a zároveň také pod proprietární licencí, a to v případě sestavení Xamarin Studio distribuovaných společností Xamarin. Tento nástroj byl vyvíjen převážně projektem Mono a společností Xamarin s přispěním komunity. Mezi podporované operační systémy patřily Microsoft Windows, Linux a macOS a mezi podporované počítačové architektury x86-64 a ARM. Hlavním účelem tohoto vývojového prostředí bylo umožnění multiplatformního vývoje softwaru postaveného na platformě .NET Framework a Mono. Součástí tohoto nástroje byl také vizuální designer GUI pro sadu knihoven GTK# nazývaný Stetic. V dotazníku Stack Overflow 2021 Developer Survey,

kterého se zúčastnilo téměř 82 000 respondentů nebyl tento nástroj obsažen. Poslední verze tohoto nástroje byla vydána 21. září 2018 a tento nástroj již není v aktivním vývoji. Přesto je stále tento nástroj součástí Linuxové verze herního enginu Unity a na jeho základě je postaven nástroj Visual Studio pro Mac. („MonoDevelop", 2022)

6.8 Přehled vlastností a jejich srovnání

Tabulka 1: Srovnání vybraných vlastností vývojářských nástrojů část 1/2

Funkce a vlastnosti	Vývojářské nástroje			
	Visual Studio Code	JetBrains Rider	GNOME Builder	KDevelop
Licence	MIT, proprietární	proprietární	GPL-3	GPL-2
Podporované operační systémy	Microsoft Windows, Linux, macOS	Microsoft Windows, Linux, macOS	Linux	Microsoft Windows, Linux, macOS
Rozšířenost	70 %	4 %	Bez dat	Bez dat
Podporované jazyky	C++, C#, CSS, Dart, Dockerfile, F#, Go, HTML, Java, JavaScript, JSON, Julia, Less, Markdown, PHP, PowerShell, Python, R, SCSS, T-SQL, TypeScript	C#, F#, VB.NET, ASP.NET, XAML, XML, JavaScript, TypeScript, JSON, HTML, CSS, SCSS, LESS, SQL	C, C++, Python, Rust, CSS, HTML, JS, JSON, Python, Ruby, SCSS, shell, XML	C, C++, Python, PHP, Java, Fortran, Ruby, Ada, Pascal, SQL, shell
Podporované implementace	.NET (Core), Mono, .NET Framework (neúplné)	Všechny implementace	Mono	Bez podpory

Zdroj: Autor

Tabulka 2: Srovnání vybraných vlastností vývojářských nástrojů část 2/2

Funkce a vlastnosti	Vývojářské nástroje			
	Visual Studio Code	JetBrains Rider	GNOME Builder	KDevelop
Podpora pro ladění	.NET (Core), Mono, .NET Framework (neúplné)	Všechny implementace	Mono	Bez podpory
Analýza kódu	Standardní podpora	Standardní podpora	Bez podpory (pro C#)	Bez podpory (pro C#)
Asistence při psaní kódu	Standardní podpora	Standardní podpora	Bez podpory (pro C#)	Pouze základní podpora
Podpora verzovacích systémů	Git, Apache Subversion, Perforce	Git, Apache Subversion, Mercurial, Perforce, TFS	Git	CVS, Apache Subversion, Perforce, ClearCase, Git, Mercurial, Bazaar
Podpora rozšíření	Ano (centrální repozitář)	Ano (centrální repozitář)	Ano	Ano

Zdroj: Autor

6.9 Závěr a doporučení

Ačkoliv není na Linuxových operačních systémech dostupná oficiální sada knihoven pro tvorbu GUI komunita tuto mezeru poměrně úspěšně vyplnila. Ze zde zmíněných sad knihoven se autorovi této práce zdají jako nejperspektivnější sady knihoven MAUI a Avalonia.

V případě sady knihoven MAUI je to z důvodu, že tato sada knihoven je vyvíjena společností Microsoft a v budoucnu by se mohla stát oficiální GUI sadou knihoven platformy .NET (Core). Jejím problémem je ovšem nejistá podpora Linuxových operačních systémů.

V případě sady knihoven Avalonia je to z důvodu rostoucí podpory od velkých společností v odvětví, zejména společnosti JetBrains, která pracuje na vizuálním editoru uživatelských rozhraní pro tuto sadu knihoven, a to pro integraci do IDE JetBrains Rider.

7 Tvorba grafických uživatelských rozhraní

Jednou z překážek pro adopci .NET platformy (zejména implementace .NET (Core)) na Linuxu je absence oficiální sady knihoven pro tvorbu GUI.

Pro implementaci .NET (Core) nebyla zpočátku dostupná žádná oficiální sada knihoven pro tvorbu GUI, a to na žádném z podporovaných operačních systémů (ani Microsoft Windows). Od .NET (Core) verze 3.0 se tato situace změnila a na operačním systému Microsoft Windows jsou pro tvorbu GUI dostupné sady knihoven Windows Forms a WPF. Další vývoj nastal v .NET (Core) verze 6, kdy se součástí této implementace stala sada knihoven pro tvorbu GUI nazvaná MAUI. Tato sada knihoven je prozatím ve fázi předběžného přístupu a jejím cílem je umožnit multiplatformní vývoj GUI. Problémem této sady knihoven je ovšem nejasná podpora pro Linux. Prozatím je sada knihoven MAUI na Linuxu podporována pouze prostřednictvím komunity a budoucnost této podpory je tak značně nejistá.

Dalším problémem v oblasti tvorby GUI je absence nativních sad knihoven. Většina desktopových Linuxových aplikací používá sadu knihoven GTK nebo Qt a aplikace, které využívají odlišné sady knihoven vypadají velmi často nepatřičně a mají jiné charakteristiky chování.

Z tohoto důvodu vzniklo v rámci projektu Mono několik pokusů o využití právě těchto sad knihoven. Ovšem ani širší komunita nezahálela a v jejím rámci vzniklo větší množství sad knihoven pro tvorbu GUI. Některé z těchto sad knihoven jsou popsány právě v této podkapitole.

7.1 MAUI

MAUI je multiplatformní sada knihoven ve fázi předběžného přístupu určená pro tvorbu GUI. Tato sada knihoven je napsána v programovacím jazyce C# a je dostupná pod open-source MIT licencí. Vyvíjena je společností Microsoft s přispěním komunity. Podpora je dostupná prostřednictvím komunity. Mezi podporované operační systémy patří Microsoft Windows, macOS, Android a iOS. Pro Linux je podpora zajišťována komunitou. Mezi podporované implementace .NET platformy patří .NET (Core), .NET Framework a Mono. Hlavním cílem této sady knihoven je tvorba multiplatformních aplikací pro desktopové

a mobilní operační systémy při použití stejné kódové základny. Sada knihoven MAUI je součástí .NET (Core) verze 6. („FAQs · Dotnet/Maui Wiki", 2022; ..*NET Multi-platform App UI (.NET MAUI)*, 2020/2022)

Vlastnosti a funkce
Tvorba uživatelského rozhraní pomocí značkovacích jazyků: Podporováno pomocí značkovacího jazyka XAML.

Nástroje pro vizuální návrh uživatelského rozhraní: Pro vizuální návrh uživatelského rozhraní nejsou pro Linux dostupné žádné nástroje.

Vývojářské nástroje s rozšířenou podporou: Vývojářské nástroje s rozšířenou podporou nejsou pro Linux dostupné.

(„FAQs · Dotnet/Maui Wiki", 2022; ..*NET Multi-platform App UI (.NET MAUI)*, 2020/2022)

7.2 GtkSharp

GtkSharp je multiplatformní sada knihoven určená pro tvorbu GUI. Tato sada knihoven je napsána v programovacím jazyce C# a je dostupná pod open-source GPL-2 licencí. Vyvíjena je open-source komunitou s možností komunitní podpory. Podporovanými operačními systémy jsou Microsoft Windows, Linux a macOS. Mezi podporované implementace .NET platformy patří .NET (Core), .NET Framework a Mono. Hlavním cílem této sady knihoven je tvorba nativních GTK aplikací, které mají stejný vzhled a charakteristiky chování na všech desktopových operačních systémech. („GTK", 2022; *GtkSharp*, 2017/2022)

Sada knihoven GtkSharp je dostupná prostřednictvím správce balíčků Nuget. Samotnou sadu knihoven lze získat pomocí balíčku *GtkSharp*, zatímco šablony aplikací lze získat pomocí balíčku *GtkSharp.Template.CSharp.*

Vlastnosti a funkce
Tvorba uživatelského rozhraní pomocí značkovacích jazyků: Podporováno pomocí značkovacího jazyka XML.

Nástroje pro vizuální návrh uživatelského rozhraní: Pro vizuální návrh uživatelského rozhraní jsou dostupné nástroje Glade, Cambalache a Stetic (součást IDE MonoDevelop).

Vývojářské nástroje s rozšířenou podporou: Mezi vývojářské nástroje s rozšířenou podporu patří MonoDevelop a GNOME Builder.

(„GTK", 2022; *GtkSharp*, 2017/2022)

7.3 Qml.Net

Qml.Net je multiplatformní sada knihoven určená pro tvorbu GUI. Tato sada knihoven je napsána v programovacím jazyce C# a je dostupná pod open-source MIT licencí. Vyvíjena je open-source komunitou s možností komunitní podpory. Podporovanými operačními systémy jsou Microsoft Windows, Linux a macOS. Mezi podporované implementace .NET platformy patří .NET (Core), .NET Framework a Mono. Hlavním cílem této sady knihoven je umožnění integrace mezi platformou .NET a platformou Qt. Platforma Qt má stejný vzhled a charakteristiky chování na všech podporovaných operačních systémech. (*Qml.Net - First look*, 2017/2022; „Qt | Design & Development Framework for Cross-Platform Applications", 2022)

Principem této sady knihoven je zpřístupňování požadovaných .NET objektů pro hostitelský QML engine. S těmito registrovanými objekty neboli portály lze poté zacházet tak jako by byly nativními komponentami jazyka QML. Je tedy možné zacházet s nimi jako s objekty v programovacím jazyce JavaScript a umožnit jejich vzájemnou komunikaci. (*Qml.Net - First look*, 2017/2022; „Qt | Design & Development Framework for Cross-Platform Applications", 2022)

Sada knihoven Qml.Net je dostupná prostřednictvím správce balíčků Nuget. Samotnou sadu knihoven lze získat pomocí balíčku *Qml.*Net a balíčku specifického pro cílový operační systém *Qml.Net.WindowsBinaries* nebo *Qml.Net.LinuxBinaries* nebo *Qml.Net.OSXBinaries.*

Vlastnosti a funkce

Tvorba uživatelského rozhraní pomocí značkovacích jazyků: Podporováno pomocí značkovacího jazyka QML.

Nástroje pro vizuální návrh uživatelského rozhraní: Pro vizuální návrh uživatelského rozhraní jsou dostupné nástroje Qt Design Studio a Qt Creator IDE.

Vývojářské nástroje s rozšířenou podporou: Mezi vývojářské nástroje s rozšířenou podporu patří Qt Creator IDE.

(*Qml.Net - First look*, 2017/2022; „Qt | Design & Development Framework for Cross-Platform Applications", 2022)

7.4 Xwt

Xwt je multiplatformní sada knihoven určená pro tvorbu GUI. Tato sada knihoven je napsána v programovacím jazyce C# a je dostupná pod open-source MIT licencí. Vyvíjena je open-source komunitou s možností komunitní podpory. Podporovanými operačními systémy jsou Microsoft Windows, Linux a macOS. Mezi podporované implementace patří Mono. Hlavním cílem této sady knihoven je tvorba aplikací, které využívají nativní vzhled a charakteristiky chování na všech desktopových operačních systémech. (*Xwt - Introduction*, 2011/2022)

Principem vykreslování této sady knihoven je využívání nativních sad knihoven cílového operačního systému. Pro Microsoft Windows jsou dostupné nativní sady knihoven WPF a GTK (pomocí knihovny Gtk#). Pro Linuxové operační systémy je dostupná nativní sada knihoven GTK (pomocí Gtk#). Pro macOS jsou dostupné nativní sady knihoven Cocoa (pomocí Xamarin.Mac) a GTK (pomocí Gtk#). (*Xwt - Introduction*, 2011/2022)

Sada knihoven Xwt je dostupná prostřednictvím správce balíčků Nuget. Samotnou sadu knihoven lze získat pomocí balíčku *Xwt* a balíčku specifického pro nativní sadu knihoven cílového operačního systému a operační systém samotný *Xwt.WPF* nebo *Xwt.Gtk.Windows* nebo *Xwt.Gtk* nebo *Xwt.Gtk3* nebo *Xwt.XamMac* nebo *Xwt.Gtk.Mac*.

Vlastnosti a funkce

Tvorba uživatelského rozhraní pomocí značkovacích jazyků: Bez podpory pro značkovací jazyky.

Nástroje pro vizuální návrh uživatelského rozhraní: Pro vizuální návrh uživatelského rozhraní nejsou dostupné žádné nástroje.

Vývojářské nástroje s rozšířenou podporou: Nejsou dostupné žádné vývojářské nástroje s rozšířenou podporou.

(*Xwt - Introduction*, 2011/2022)

7.5 Avalonia

Avalonia je multiplatformní sada knihoven určená pro tvorbu GUI. Tato sada knihoven je napsána v programovacím jazyce C# a je dostupná pod open-source MIT licencí. Vyvíjena je open-source komunitou s možností komunitní a placené podpory. Podporovanými operačními systémy jsou Microsoft Windows, Linux a macOS. Mezi podporované implementace .NET platformy patří .NET (Core), .NET Framework a Mono. Hlavním cílem této sady knihoven je tvorba aplikací, které mají stejný vzhled a charakteristiky chování na všech desktopových operačních systémech. Tato sada knihoven je podporována společnostmi JetBrains, Unity, Schneider Electric a Mailbird. (*AvaloniaUI/Avalonia*, 2013/2022; „Avalonia - Welcome", 2022)

Sada knihoven Avalonia je dostupná prostřednictvím správce balíčků Nuget. Samotnou sadu knihoven lze získat pomocí balíčků *Avalonia* a *Avalonia.Desktop*.

Vlastnosti a funkce

Tvorba uživatelského rozhraní pomocí značkovacích jazyků: Podporováno pomocí značkovacího jazyka XAML.

Nástroje pro vizuální návrh uživatelského rozhraní: Pro vizuální návrh uživatelského rozhraní je dostupný nástroj DevTools.

Vývojářské nástroje s rozšířenou podporou: Mezi vývojářské nástroje s rozšířenou podporu patří JetBrains Rider a multiplatformní IDE AvalonStudio.

(*AvaloniaUI/Avalonia*, 2013/2022; „Avalonia - Welcome", 2022)

7.6 Eto

Eto je multiplatformní sada knihoven určená pro tvorbu GUI. Tato sada knihoven je napsána v programovacím jazyce C# a je dostupná pod open-source BSD-3 licencí. Vyvíjena je open-source komunitou s možností komunitní podpory. Podporovanými operačními systémy jsou Microsoft Windows, Linux a macOS. Mezi podporované implementace .NET platformy patří .NET (Core), .NET Framework a Mono. Hlavním cílem této sady knihoven je tvorba aplikací, které využívají nativní vzhled a charakteristiky chování na všech desktopových operačních systémech. (*Eto.Forms*, 2011/2022; „Eto.Forms Visual Studio Addin - Visual Studio Marketplace", 2022; „Quick Start · Picoe/Eto Wiki", 2022)

Principem vykreslování této sady knihoven je využívání nativních sad knihoven cílového operačního systému. Pro Microsoft Windows jsou dostupné nativní sady knihoven WPF a Windows Forms. Pro Linux je dostupná nativní sada knihoven GTK (pomocí Gtk#). Pro macOS jsou dostupné nativní sady knihoven Cocoa (pomocí Xamarin.Mac) a GTK (pomocí MonoMac). (*Eto.Forms*, 2011/2022; „Eto.Forms Visual Studio Addin - Visual Studio Marketplace", 2022; „Quick Start · Picoe/Eto Wiki", 2022)

Sada knihoven Eto je dostupná prostřednictvím správce balíčků Nuget. Samotnou sadu knihoven lze získat pomocí balíčku *Eto* a balíčku specifického pro nativní sadu knihoven cílového operačního systému a operační systém samotný *Eto.WinForms* nebo *Eto.Wpf* nebo *Eto.Gtk* nebo *Eto.Mac* nebo *Eto.XamMac*.

Vlastnosti a funkce

Tvorba uživatelského rozhraní pomocí značkovacích jazyků: Podporováno pomocí značkovacích jazyků XAML a JSON.

Nástroje pro vizuální návrh uživatelského rozhraní: Pro vizuální návrh uživatelského rozhraní nejsou pro Linux dostupné žádné nástroje.

Vývojářské nástroje s rozšířenou podporou: Vývojářské nástroje s rozšířenou podporou nejsou pro Linux dostupné.

(*Eto.Forms*, 2011/2022; „Eto.Forms Visual Studio Addin - Visual Studio Marketplace", 2022; „Quick Start · Picoe/Eto Wiki", 2022)

7.7 Uno

Uno je multiplatformní sada knihoven určená pro tvorbu GUI. Tato sada knihoven je napsána v programovacím jazyce C# a je dostupná pod open-source Apache 2.0 licencí. Vyvíjena je open-source komunitou s možností komunitní a placené podpory. Podporovanými operačními systémy jsou Microsoft Windows, Linux, macOS, Android a iOS. Mezi podporované implementace .NET platformy patří .NET (Core), .NET Framework a Mono. Hlavním cílem této sady knihoven je tvorba multiplatformních aplikací pro desktopové a mobilní operační systémy za použití stejné kódové základny. („Uno - Get Started", 2022; *unoplatform/uno*, 2018/2022; „Uno.UI 4.1.9", 2022)

Sada knihoven Uno je dostupná prostřednictvím správce balíčků Nuget. Šablony aplikací lze získat pomocí balíčku *Uno.ProjectTemplates.Dotnet.*

Vlastnosti a funkce
Tvorba uživatelského rozhraní pomocí značkovacích jazyků: Podporováno pomocí značkovacího jazyka XAML.

Nástroje pro vizuální návrh uživatelského rozhraní: Pro vizuální návrh uživatelského rozhraní nejsou pro Linux dostupné žádné nástroje.

Vývojářské nástroje s rozšířenou podporou: Mezi vývojářské nástroje s rozšířenou podporu patří Visual Studio Code a JetBrains Rider. („Uno - Get Started", 2022; *unoplatform/uno*, 2018/2022; „Uno.UI 4.1.9", 2022)

7.8 Windows Forms (historické)

Windows Forms je původně proprietární od 4. prosince roku 2018 open-source sada knihoven určená pro tvorbu GUI. Tato sada knihoven je napsána v programovacím jazyce C# a je dostupná pod open-source MIT licencí. Vyvíjena je společností Microsoft. Podporovaným operačním systémem je Microsoft Windows. Mezi podporované implementace .NET platformy patří .NET (Core), .NET Framework a Mono. Hlavním cílem této sady knihoven je tvorba desktopových aplikací určených pro operační systém Microsoft Windows. („Windows Forms", 2021; *Windows Forms*, 2018/2022; „WinForms | Mono", 2022)

V rámci projektu Mono byla vyvinuta neúplná multiplatformní implementace této sady knihoven s omezením 32bitové aplikace. Tato implementace vznikla jako snaha o zjednodušení portování aplikací určených pro Microsoft Windows na Linux. Tato implementace již není v aktivním vývoji. („Windows Forms", 2021; *Windows Forms,* 2018/2022; „WinForms | Mono", 2022)

7.9 Přehled vlastností a jejich srovnání

Tabulka 3: Srovnání vybraných vlastností vybraných sad knihoven pro tvorbu GUI

Funkce a vlastnosti	Sady knihoven pro tvorbu GUI				
	MAUI	GtkSharp	Qml.Net	Avalonia	Eto
Licence	MIT	GPL-2	MIT	MIT	BSD-3
Druh podpory	Komunitní (vyvíjeno společností Microsoft)	Komunitní	Komunitní	Komunitní a placená	Komunitní a placená
Podporované operační systémy	Windows, macOS, Android, IOS, Linux (komunitně)	Windows, Linux, macOS	Windows, Linux, macOS	Windows, Linux, macOS	Windows, Linux, macOS
Podporované značkovací jazyky	XAML	XML	QML	XAML	XAML, JSON
Nástroje pro vizuální návrh uživatelského rozhraní	Nedostupné (pro Linux)	Glade, Cambalache, Stetic (součást IDE)	Qt Design Studio, Qt Creator IDE	DevTools (integrováno)	Nedostupné (pro Linux)
Vývojářské nástroje s rozšířenou podporou	Nedostupné (pro Linux)	MonoDevelop, GNOME Builder	Qt Creator IDE	JetBrains Rider, AvalonStudio	Nedostupné (pro Linux)

Zdroj: Autor

7.10 Závěr a doporučení

Ačkoliv není na Linuxových operačních systémech dostupné jedno z nejrozšířenějších a nejlépe podporovaných IDE Visual Studio stále je dostupné nemalé množství nástrojů, které mohou tento nástroj nahradit.

Jako nejplnohodnotnější náhrada tohoto nástroje se jeví nástroj JetBrains Rider, který je, co se podporovaných funkcí a vlastností týče na velmi vysoké úrovni a některými vývojáři je dokonce preferován, a to i na operačním systému Microsoft Windows. Jeho nevýhodou je nutnost nákupu licence.

Další alternativou může také být editor zdrojového Visual Studio Code. Nabídka jeho funkcí je velmi široká a jde o jeden z nejpoužívanějších vývojářských nástrojů. Jeho další výhodou je také to, že je dostupný zcela zdarma. Jeho nevýhodou je omezená integrace s jednotlivými platformami. V případě .NET platformy jde například o to, že není možné celý vývoj aplikace obsloužit přímo z tohoto nástroje, ale je nutné využít pomocného konzolového nástroje (například pro vytváření nových projektů a přidávání Nuget balíčků).

8 Konfigurace vývojového prostředí

Cílem této kapitoly je zdokumentovat potřebné kroky pro vytvoření vývojového prostředí v Linuxových operačních systémech. Toto vývojové prostředí bude určeno pro vývoj aplikací a webových služeb postavených na platformě .NET a programovacím jazyce C#.

8.1 Volba komponent vývojového prostředí

Pro účely této podkapitoly byla zvolena Linuxová distribuce Ubuntu, a to ve verzi 20.04. Tato distribuce byla zvolena z důvodu své dlouhodobé podpory, velké rozšířenosti a statutu oficiálně podporované Linuxové distribuce pro používání produktů společnosti Microsoft.

Jako implementace .NET platformy byla zvolena implementace .NET (Core) ve verzi 6. Tato implementace byla zvolena z důvodu své dlouhodobé podpory, velikosti vývojářské základny, aktivity vývoje a také proto že dle společnosti Microsoft (a i některých jiných včetně autora) je právě tato implementace budoucností .NET platformy.

Jako editor zdrojového kódu bylo zvolen nástroj Visual Studio Code. Tento editor byl zvolen z důvodu své volné dostupnosti, velké rozšířenosti, dobré podpory pro vývoj aplikací .NET platformy a programovacího jazyka C# a také proto, že je oficiálně doporučován společností Microsoft jako nástroj vhodný pro vývoj aplikací postavených právě na této platformě.

8.2 Konfigurace implementace .NET (Core)

Na Linuxové distribuci Ubuntu a stejně tak na velkém množství ostatních Linuxových distribucí není aktuální verze implementace .NET (Core) dostupná v hlavních repozitářích, a proto je instalace o něco složitější. Pro Ubuntu jsou aktuální verze implementace .NET (Core) oficiálně distribuovány pomocí rozšiřujícího repozitáře spravovaného společností Microsoft.

Postup instalace

Všechny zde zmíněné příkazy je nutné zadávat v konzoli, samotné příkazy a názvy jsou zvýrazněny kurzívou. Původ těchto instrukcí je z webových stránek https://dotnet.microsoft.com.

Stažení balíčku s přídavným repozitářem a s GPG šifrovacími klíči pro ověřování autentičnosti: *wget https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-prod.deb -O packages-microsoft-prod.deb*

Instalace staženého balíčku: *sudo dpkg -i packages-microsoft-prod.deb*

Odstranění staženého balíčku: *rm packages-microsoft-prod.deb*

Instalace balíčku pro přidání podpory HTTPS protokolu do balíčkovacího systému APT: *sudo apt-get install -y apt-transport-https*

Aktualizace indexu dostupných balíčků: *sudo apt-get update*

Instalace SDK pro platformu .NET (Core): *sudo apt-get install -y dotnet-sdk-6.0*

Kontrolu instalace lze provést pomocí příkazů *dotnet --list-sdks* a *dotnet --list-runtimes*

8.3 Konfigurace nástroje Visual Studio Code

Na Linuxové distribuci Ubuntu a stejně tak na většině ostatních není editor zdrojového kódu Visual Studio Code dostupný v hlavních repozitářích, a proto je instalace o něco složitější. Pro Ubuntu je editor Visual Studio Code oficiálně distribuován formou deb a snap balíčku, které jsou dostupné na stránkách projektu Visual Studio Code.

Postup instalace

Všechny zde zmíněné příkazy je nutné zadávat v konzoli, samotné příkazy a názvy jsou zvýrazněny kurzívou. Původ těchto instrukcí je z webových stránek https://code.visualstudio.com.

Nejprve je nutné navštívit webové stránky https://code.visualstudio.com a zde je nutné kliknout na tlačítko pro stažení. Po kliknutí dojde k přesměrování na další webovou stránku se seznamem instalátorů a balíčků pro různé operační systémy. Zde je nutné kliknout na tlačítko .deb a nechat zvolený balíček stáhnout. Poté je nutné stažený balíček nainstalovat.

Přechod do složky pro stažené soubory: *cd Stažené*

Instalace staženého balíčku: *sudo apt install ./code*.deb*

Odstranění staženého balíčku: *rm code*.deb*

Po provedení těchto kroků by mělo dojít k vytvoření zástupce pro spuštění tohoto nástroje. Tento nástroj je také možné spustit pomocí příkazu *code*.

Postup konfigurace

Po spuštění nástroje Visual Studio Code je uživatel vyzván k instalaci místní jazykové sady. Pro její instalaci postačuje stisknout tlačítko nainstalovat a poté nástroj restartovat.

Pro vylepšení podpory .NET platformy a programovacího jazyka C# je vhodné nainstalovat rozšíření *C#*. Toto rozšíření lze nainstalovat kliknutím na tlačítko *Rozšíření* v levém postranním panelu. Po kliknutí se otevře postranní panel s polem pro vyhledávání. Do tohoto pole je nutné zadat výraz *C#*, po vyhledání je nutné kliknout na rozšíření s tímto názvem a dále kliknout tlačítko instalovat. Jakmile bude instalace dokončena (instalace může chvíli trvat) je nutné nástroj Visual Studio Code restartovat. Poté je tento nástroj připraven k použití.

9 Popis vývoje ukázkové aplikace

V této kapitole je popsán vývoj ukázkové desktopové aplikace s GUI pro operační systémy Linux a Microsoft Windows. Pro vytvoření uživatelské rozhraní bude použita sada knihoven GtkSharp. Rozložení uživatelského rozhraní bude navrženo pomocí nástroje Glade určeného pro návrh GUI. Funkcí aplikace bude automatická detekce operačního systému a zobrazení detekovaného operačního systému formou obrázku a textu.

9.1 Vytvoření nového projektu a řešení

Vytvoření nového projektu a řešení nelze provést přímo pomocí nástroje Visual Studio Code, ale je nutné využít konzole a nástroje .NET CLI (konzoly lze otevřít přímo v nástroji Visual Studio Code). Pro vytvoření nového řešení a projektu je nutné zadat následující příkazy.

Vytvoření nové složky OSDetection: *mkdir OSDetection*

Přesun do složky OSDetection: *cd OSDetection*

Vytvoření nového řešení: *dotnet new sln*

Vytvoření nového .NET (Core) projektu s názvem OSDetection a cílovou verzí .NET (Core) 6: *dotnet new console -o OSDetection -f net6.0*

Přidání projektu OSDetection do řešení: *dotnet sln add OSDetection*

Po provedení těchto kroků je možné vytvořené řešení otevřít v nástroji Visual Studio Code, a to pomocí volby otevřít složku. Po otevření složky řešení je uživatel vyzván k vygenerování ladících symbolů, zde je vhodné toto vygenerování povolit (lze dodatečně vygenerovat pomocí palety příkazů).

Dalším krokem je instalace sady knihoven GtkSharp (dostupná prostřednictvím správce balíčků NuGet). Tuto instalaci je opět nutné provést pomocí příkazového řádku a nástroje .NET CLI.

Instalace sady knihoven GtkSharp do projektu OSDetection: *dotnet add OSDetection package GtkSharp*

Po provedení těchto kroků je již projekt připraven k samotnému psaní kódu.

9.2 Psaní zdrojového kódu

Aplikace se skládá z jednoho řešení (soubor OSDetection.sln) a jednoho projektu (složka OSDetection). Samotná aplikace je rozdělena do čtyř souborů OSDetection.csproj, MainWindow.ui, Program.cs, MainWindow.cs a OperatingSystem.cs.

9.2.1 Soubor OSDetection.sln

Tento soubor řešení obsahuje informace o uspořádání projektů, které jsou součástí tohoto řešení. Tento soubor není pro vývoj samotné aplikace příliš důležitý a ve většině případů je generován automatizovaně prostřednictvím vývojářských nástrojů.

9.2.2 Soubor OSDetection.csproj

Tento soubor projektu obsahuje informace o všech potřebných souborech používaných v rámci aplikace. Součástí tohoto souboru jsou také odkazy na použité balíčky (knihovny a jejich sestavy). Tento soubor je vygenerován automatizovaně a následně manuálně upravován manuálně. Sestavovací systém MSBuild (Microsoft Build Engine) tento soubor využívá ke správnému sestavení projektu. Obsah tohoto souboru je psán pomocí značkovacího jazyka XML a MSBuild XML schématu.

Ze skupiny `<PropertyGroup>` je důležitý zejména tag `<TargetFramework>` Pomocí tohoto tagu lze změnit verzi cílového frameworku. Například pro změnu cílového frameworku z .NET (Core) 6 na .NET (Core) 5 by zde bylo nutné zapsat net5.0 místo net6.0.

```
<PropertyGroup>
    <OutputType>Exe</OutputType>
    <TargetFramework>net6.0</TargetFramework>
    <ImplicitUsings>enable</ImplicitUsings>
    <Nullable>enable</Nullable>
</PropertyGroup>
```

Nyní se již dostáváme k samotným položkám, jednotlivé položky jsou od sebe odděleny pomocí tagu `<ItemGroup>`.

Tag `<PackageReference>` obsahuje informace o potřebných balíčcích (rozšiřujících knihovnách). V tomto projektu je obsažen balíček GtkSharp, který je určen pro tvorbu GUI.

```
<ItemGroup>
    <PackageReference Include="GtkSharp" Version="3.24.24.34" />
</ItemGroup>
```

Tag `<EmbeddedResource>` definuje prostředky, které se při sestavování mají stát součástí generované sestavy. V tomto případě se jedná o XML soubor, ve kterém je definováno rozložení uživatelského rozhraní.

```
<ItemGroup>
    <None Remove="**\*.ui" />
    <EmbeddedResource Include="**\*.ui">
        <LogicalName>%(Filename)%(Extension)</LogicalName>
    </EmbeddedResource>
</ItemGroup>
```

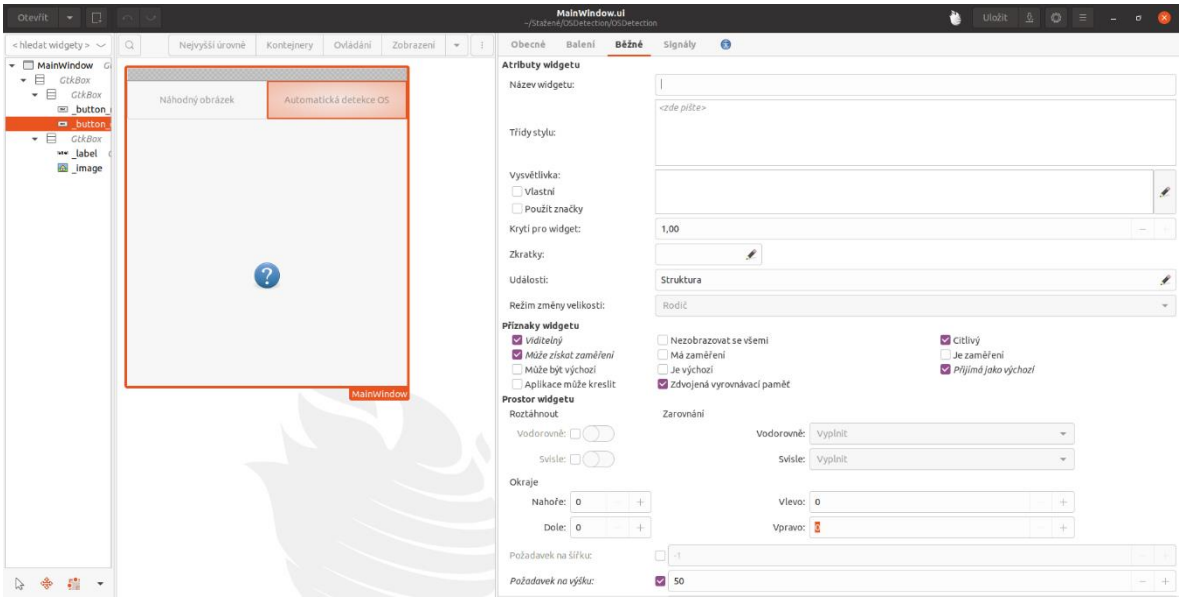
Tag `<Content>` definuje soubory, které se při sestavovaní mají překopírovat do výstupní složky společně s vytvořenou sestavou. V tomto případě se jedná o všechny obrázky, které jsou využívány v rámci detekce operačního systému. Je zde obsaženo více variant s podmínkami pro operační systémy s různými oddělovači.

```
<ItemGroup>
  <Content Include="res/img/windows.png"
Condition="$([MSBuild]::IsOSPlatform('Linux')) or
$([MSBuild]::IsOSPlatform('macOS'))">
        <CopyToOutputDirectory>Always</CopyToOutputDirectory>
    </Content>
    <Content Include="res\img\windows.png"
Condition="$([MSBuild]::IsOSPlatform('Windows'))">
        <CopyToOutputDirectory>Always</CopyToOutputDirectory>
    </Content>
</ItemGroup>
```

9.2.3 Soubor MainWindow.ui

V tomto souboru je definováno rozložení uživatelského rozhraní aplikace, a to pomocí značkovacího jazyka XML. Tento soubor byl vygenerován s pomocí WYSIWYG (What You See Is What You Get) nástroje Glade (Obrázek 1), který je určen pro návrh GUI při použití sady knihoven GTK.

Obrázek 1: Hlavní obrazovka nástroje Glade s náhledem uživatelského rozhraní vyvíjené aplikace



Zdroj: Autor

Jednotlivé objekty tedy prvky uživatelského rozhraní jsou zde zapsány pomocí tagu `<object>` a jejich vlastnosti pomocí tagu `<property>`. Pokud je nějaký objekt potomkem jiného objektu je tak uvnitř rodičovského tagu a sám je uveden pomocí tagu `<child>`. Tag `<packing>` slouží k nastavení vlastností vyplňování prostoru v rámci rodičovského objektu.

```
<interface>
    <object class="GtkBox">
        <property name="visible">True</property>
        <property name="can_focus">False</property>
        <property name="orientation">vertical</property>
        <child>
            <object class="GtkButton" id="_button_random">
                <property name="label" translatable="yes">Náhodný
obrázek</property>
                <property name="height_request">60</property>
                <property name="visible">True</property>
                <property name="can_focus">True</property>
                <property name="receives_default">True</property>
            </object>
            <packing>
                <property name="expand">True</property>
                <property name="fill">True</property>
                <property name="position">0</property>
            </packing>
        </child>
    </object>
</interface>
```

9.2.4 Soubor Program.cs

V tomto souboru je obsažena pouze metoda `Main`, která je automaticky volána po spuštění aplikace. V rámci této metody dochází k přípravě základních informací o aplikaci, a nakonec k samotnému spuštění aplikace včetně zobrazení hlavního (a jediného) okna.

```
public static void Main(string[] args) {
    Application.Init();

    var app = new Application("os.detection", GLib.ApplicationFlags.None);
    app.Register(GLib.Cancellable.Current);

    var win = new MainWindow();
    app.AddWindow(win);

    win.Show();
    Application.Run();
    }
```

9.2.5 Soubor OperatingSystem.cs

V tomto souboru je obsaženy pomocné metody pro některé funkce aplikace.

Metoda IsWindows slouží k detekci operačního systému, pokud je tato metoda skutečně zavolána na dotazovaném operačním systému pak je vrácena hodnota pravda. Pokud tomu tak není je vrácena hodnota nepravda.

```
public static bool IsWindows() {
    return RuntimeInformation.IsOSPlatform(OSPlatform.Windows);
    }
```

Metoda SetWindows slouží k nastavení hodnot pro detekovaný operační systém. Mezi nastavované hodnoty patří label (textový řetězec) a image (plocha pro obrázek) s logem daného operačního systému. Formát cesty k obrázku je formátován tímto způsobem z důvodu multiplatformovosti aplikace (rozdílnost oddělovačů na různých operačních systémech).

```
public static void SetWindows(Gtk.Image image, Gtk.Label label) {
    image.Pixbuf = new
Gdk.Pixbuf(String.Format(".{0}res{0}img{0}windows.png",
System.IO.Path.DirectorySeparatorChar));
    label.Text = "Automatická detekce: Program je spuštěn na OS Microsoft
Windows!";
     }
```

Metoda SetRandom je obdobou předchozí metody pouze nenastavuje hodnoty na detekovaný operační systém, ale volí náhodný obrázek a do textového řetězce zapisuje kolikrát bylo tlačítko stisknuto.

```
public static void SetRandom(Gtk.Image image, Gtk.Label label) {
    counter++;
    int rnd = new Random().Next(0,3);

    image.Pixbuf = new Gdk.Pixbuf(String.Format(".{0}res{0}img{0}{1}.png",
System.IO.Path.DirectorySeparatorChar, osName[rnd]));
    label.Text = String.Format("Od spuštění aplikace jste {0}krát provedli
náhodný výběr!", counter);
    }
```

9.2.6 Soubor MainWindow.cs

V tomto souboru je obsažena logika hlavního okna aplikace.

V této části kódu dochází k přípravě objektů, které budou posléze obsahovat odkazy na skutečné objekty (prvky) uživatelského rozhraní.

```
 [UI] private Label _label = null;
[UI] private Button _button_random = null;
[UI] private Button _button_detect = null;
[UI] private Image _image = null;
```

V této části kódu je definován název souboru s návrhem uživatelského rozhraní.

```
public MainWindow() : this(new Builder("MainWindow.ui")) { }
```

V této části kódu dochází k přípravě objektu samotného okna, respektive plochy pro vykreslení tohoto okna.

```
private MainWindow(Builder builder) :
base(builder.GetRawOwnedObject("MainWindow")) {
    builder.Autoconnect(this);
```

V této části kódu dochází k definování obslužných metod pro jednotlivé události ovládacích prvků.

```
DeleteEvent += Window_DeleteEvent;
_button_random.Clicked += ButtonRandomize_Clicked;
_button_detect.Clicked += ButtonDetect_Clicked;
```

V této části kódu je nastaven výchozí textový řetězec a výchozí obrázek, který je zobrazen po spuštění aplikace, a to pomocí volání metody SetUnknown, která je obdobou metody SetWindows. Jako parametry jsou předávány objekty, kterých se má změna týkat.

```
OperatingSystem.SetUnknown(_image, _label);
```

Následující metoda je volána při zavření hlavního okna aplikace a jejím úkolem je ukončit běh samotné aplikace.

```
private void Window_DeleteEvent(object sender, DeleteEventArgs a) {
    Application.Quit();
    }
```

Následující metoda je volána při kliknutí na tlačítko s funkcí náhodného výběru. A to pomocí volání metody SetRandom, která byla popsána výše.

```
private void ButtonRandomize_Clicked(object sender, EventArgs a) {
    OperatingSystem.SetRandom(_image, _label);
    }
```

Následující metoda je volána při kliknutí na tlačítko pro automatickou detekci operačního systému. Úkolem této metody je ověřit na jakém operačním systému je aplikace spuštěna (pomocí metod pro detekci operačního systému která byly popsána výše) a podle výsledku zavolat metody pro nastavení hodnot příslušného systému.

```
private void ButtonDetect_Clicked(object sender, EventArgs a) {
    if (OperatingSystem.IsWindows()) {
        OperatingSystem.SetWindows(_image, _label);
        }

        else if (OperatingSystem.IsLinux()) {
            OperatingSystem.SetLinux(_image, _label);
            }

        else if (OperatingSystem.IsMacOS()) {
            OperatingSystem.SetMacOS(_image, _label);
            }

        else {
            OperatingSystem.SetUnknown(_image, _label);
             }
```

9.3 Výsledná aplikace

Výsledná aplikace byla otestována na operačních systémech Linux a Microsoft Windows.

Obrázek 2: Různé obrazovky výsledné aplikace

Zdroj: Autor

V levém horním obrázku (Obrázek 2) lze vidět výchozí obrazovku aplikace (Linux). V tuto chvíli čeká aplikace na stisk tlačítka. Po stisknutí tlačítka pro automatickou detekci operačního systému se zobrazí obrazovka s logem detekovaného operačního systému a také dojde ke změně zobrazovaného textu. Výsledek této akce lze vidět na obrázcích (Obrázek 2) vpravo nahoře (Linux) a vlevo dole (Windows). Další akcí, kterou je možno provést je výběr náhodného obrázku. Tato akce se provede po stisknutí tlačítka pro náhodný obrázek. Po stisknutí tohoto tlačítka dojde k zobrazení náhodného loga operačního systému a také k zaznamenání počtů stisknutí tohoto tlačítka (počet stisknutí je zobrazen v textovém řetězci). Výsledek této akce lze vidět na obrázku (Obrázek 2) vpravo dole (Linux).

Zdrojový kód této aplikace je přiložen na CD, které je přílohou této práce.

10 Závěr

Cílem této bakalářské práce bylo prozkoumat úroveň podpory .NET platformy na operačním systému GNU/Linux a také vypracovat přehled a srovnání softwarových prostředků, které tuto technologii a tento operační systém podporují. Reálná použitelnost této podpory měla být vyzkoušena při vývoji ukázkové aplikace.

V úvodu práce došlo k představení .NET platformy, tedy technologie pro vývoj a běh aplikací. Došlo zde k sepsání obecných vlastností této platformy, přičemž mezi ty nejdůležitější patří jazyková nezávislost a přenositelnost. Následoval popis architektury, která je složena ze standardizované specifikace, komponenty implementující tuto specifikaci a z rozsáhlých standardních knihoven.

Další část práce se věnovala operačnímu systému GNU/Linux. Tento operační systém zde byl představen a dále bylo popsáno z jakých komponent se skládá a jaký je způsob jeho distribuce.

Poté následovalo zjištění, že .NET platforma není pouze jedním produktem, ale existuje větší množství vzájemně kompatibilních implementací. Proto byly představeny různé implementace této platformy, a to s jejich historií a specifickými vlastnostmi. Došlo zde také ke zmínění problému softwarových patentů, kterým byly alternativní implementace nuceny čelit.

Dále byl popsán historický a současný stav .NET platformy na operačním systému GNU/Linux se zaměřením na rozšíření této platformy. Došlo také na představení několika aplikací postavených na této platformě.

Dalším krokem bylo ověření dostupnosti vývojářských nástrojů s podporou pro .NET platformu na operačním systému GNU/Linux. Součástí této části práce je i přehled těchto vývojářských nástrojů včetně popisu jejich funkcí, vlastností a dostupnosti. Tyto nástroje jsou zde také porovnány.

Dále byl zvýrazněn jeden z velkých problémů .NET platformy na operačním systému GNU/Linux, a to absence oficiálních sad knihoven pro tvorbu grafických uživatelských rozhraní. Tento problém byl ovšem částečně vyřešen s pomocí komunity, a tak je součástí této části přehled dostupných produktů včetně jejich funkcí, vlastností a dostupnosti. Tato řešení byla také vzájemně porovnána.

Následně byla zdokumentována příprava vývojového prostředí. Toto vývojové prostředí bylo složeno z implementace .NET (Core) a editoru zdrojového kódu Visual Studio Code.

Podpora .NET platformy a funkčnost připraveného vývojového prostředí byla ověřena na vývoji jednoduché aplikace. Během tohoto vývoje vyplynuly na povrch některé problémy, a to zejména nedostatek dokumentace k tvorbě uživatelských grafických rozhraní. Ale i přes některé drobné problémy byl vývoj aplikace dokončen a bylo tedy dokázáno, že je možné vyvinout .NET aplikaci na operačním systému GNU/Linux.

I. Summary and keywords

The bachelor thesis explores the support and real-world usability of the .NET developer platform on the GNU/Linux operating system. The introduction describes the .NET platform itself, its features and architecture. Then comes a description of the GNU/Linux operating system. The space is then devoted to various implementations of the.NET platform, their history, and specific features. The problem of software patents is also mentioned here. Then the prevalence of the .NET platform on the GNU/Linux operating system is explored, including an overview of several applications built on this platform. This is followed by an overview of development tools, their features, and functions, including their comparison. The next part is devoted to one of the biggest problems of the .NET platform on the GNU/Linux operating system, namely the problem of developing graphical user interfaces. At the end of the work, the preparation of the development environment is documented, including a description of the sample application development process.

Keywords: support, real-world usability, .NET platform, GNU/Linux, implementations, software patents, application, development tools, graphical user interfaces, development environment, sample application, comparison

II. Seznam literatury

Apps/Builder—GNOME Wiki! (2022). Získáno 9. únor 2022, z https://wiki.gnome.org/Apps/Builder

Apps/gbrainy—GNOME Wiki! (2022). Získáno 11. únor 2022, z https://wiki.gnome.org/Apps/gbrainy

AvaloniaUI/Avalonia [C#]. (2022). AvaloniaUI. Získáno z https://github.com/AvaloniaUI/Avalonia (Original work published 2013)

Avalonia—Welcome. (2022). Získáno 8. březen 2022, z https://docs.avaloniaui.net/

Banshee (media player). (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=Banshee_(media_player)&oldid=1075517593

Beagle (software). (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=Beagle_(software)&oldid=1072958369

Common Language Runtime. (2021). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=Common_Language_Runtime&oldid=1008949276

CrossNet—Web archive. (2022). Získáno 7. únor 2022, z https://web.archive.org/web/20100125035452/http://www.codeplex.com/crossnet

Docky in Launchpad. (2022). Získáno 11. březen 2022, z Launchpad website: https://launchpad.net/docky/+index

DotGNU Portable.NET. (2022). Získáno 7. únor 2022, z https://www.gnu.org/software/dotgnu/pnet.html

Eto.Forms [C#]. (2022). Picoe Software Solutions. Získáno z https://github.com/picoe/Eto (Original work published 2011)

Eto.Forms Visual Studio Addin—Visual Studio Marketplace. (2022). Získáno 8. březen 2022, z https://marketplace.visualstudio.com/items?itemName=CurtisWensley.EtoFormsVisualStudioAddin

FAQs · dotnet/maui Wiki. (2022). Získáno 9. březen 2022, z GitHub website: https://github.com/dotnet/maui

F-Spot. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=F-Spot&oldid=1076073922

GNOME Builder. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=GNOME_Builder&oldid=1070992793

GNOME Do. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=GNOME_Do&oldid=1072958386

GTK. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=GTK&oldid=1080758676

GtkSharp [C#]. (2022). GtkSharp. Získáno z https://github.com/GtkSharp/GtkSharp (Original work published 2017)

Immo Landwerth. (2014). .NET Core is opensource. Získáno 7. únor 2022, z https://devblogs.microsoft.com/dotnet/net-core-is-open-source

Introduction to .NET Framework. (2018, prosinec 11). Získáno 9. březen 2022, z GeeksforGeeks website: https://www.geeksforgeeks.org/introduction-to-net-framework/

KDevelop. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=KDevelop&oldid=1068508303

KeePass. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=KeePass&oldid=1072958400

Linux. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=Linux&oldid=1081453853

Linux distribution. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=Linux_distribution&oldid=1080457605

List of CLI languages. (2021). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=List_of_CLI_languages&oldid=1045456303

Metzgar, D. (2018). *.NET Core in Action* (1st vyd.). Shelter Island: Manning.

Microsoft Open Sources .NET and Mono—Miguel de Icaza. (2014). Získáno 7. únor 2022, z https://tirania.org/blog/archive/2014/Nov-12.html

Microsoft Silverlight. (2022). Získáno 7. únor 2022, z https://www.microsoft.com/silverlight/Default

Mono (software). (2021). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=Mono_(software)&oldid=1061626372

MonoDevelop. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=MonoDevelop&oldid=1073581777

Moonlight (runtime). (2021). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=Moonlight_(runtime)&oldid=1057643789

MSDN. (2022). .NET CLI. Získáno 7. únor 2022, z https://docs.microsoft.com/en-us/dotnet/core/tools/

.NET. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=.NET&oldid=1066156008

.NET and .NET Core official support policy. (2022). Získáno 7. únor 2022, z Microsoft website: https://dotnet.microsoft.com/en-us/platform/support/policy/dotnet-core

.NET Framework. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=.NET_Framework&oldid=1067811060

.NET Micro Framework. (2022). Získáno 7. únor 2022, z https://netmf.github.io/

.NET Multi-platform App UI (.NET MAUI) [C#]. (2022). .NET Platform. Získáno z https://github.com/dotnet/maui (Original work published 2020)

OmniSharp. (2022). Získáno 4. březen 2022, z GitHub website: https://github.com/OmniSharp

OmniSharp—.NET and IntelliSense on any platform with your editor of choice. (2022). Získáno 30. březen 2022, z http://www.omnisharp.net/#about

Overview Of Common Language Infrastructure. (2022). Získáno 9. březen 2022, z https://www.c-sharpcorner.com/blogs/overview-of-common-language-infrastructure

Pinta (software). (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=Pinta_(software)&oldid=1072958415

Qml.Net—First look [C#]. (2022). Qml.Net. Získáno z https://github.com/qmlnet/qmlnet (Original work published 2017)

Qt | Design & Development Framework for Cross-platform Applications. (2022). Získáno 8. březen 2022, z https://www.qt.io/product

Quick Start · picoe/Eto Wiki. (2022). Získáno 1. duben 2022, z GitHub website: https://github.com/picoe/Eto

Rider: The Cross-Platform .NET IDE from JetBrains. (2022). Získáno 2. březen 2022, z JetBrains website: https://www.jetbrains.com/rider/

Seroter, R. (2018). *Modernizing .NET Applications* (First edition). Birmingham: O'Reilly.

SparkleShare. (2021). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=SparkleShare&oldid=1052140209

Stack Overflow Developer Survey 2021. (2021). Získáno 11. březen 2022, z Stack Overflow website: https://insights.stackoverflow.com/survey/2021/?utm_source=social-share&utm_medium=social&utm_campaign=dev-survey-2021

Standard Libraries (CLI). (2021). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=Standard_Libraries_(CLI)&oldid=1059815978

Uno—Get Started. (2022). Získáno 9. březen 2022, z https://platform.uno/docs/articles/get-started.html

Unoplatform/uno [C#]. (2022). Uno Platform. Získáno z https://github.com/unoplatform/uno (Original work published 2018)

Uno.UI 4.1.9. (2022). Získáno 9. březen 2022, z https://www.nuget.org/packages/Uno.UI/

Visual Studio Code. (2022). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=Visual_Studio_Code&oldid=1078856229

What is .NET? An open-source developer platform. (2022). Získáno 9. březen 2022, z Microsoft website: https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet

Will open source get snagged in .Net? - ZDNet Asia News. (2011). Získáno 12. březen 2022, z http://web.archive.org/web/20111105191541/http://www.zdnetasia.com/will-open-source-get-snagged-in-net-30102692.htm

Windows Forms. (2021). In *Wikipedia*. Získáno z https://en.wikipedia.org/w/index.php?title=Windows_Forms&oldid=1044774100

Windows Forms [C#]. (2022). .NET Platform. Získáno z https://github.com/dotnet/winforms (Original work published 2018)

WinForms | Mono. (2022). Získáno 9. březen 2022, z https://www.mono-project.com/docs/gui/winforms/

Xamarin App Development with Visual Studio | Visual Studio. (2022). Získáno 7. únor 2022, z https://visualstudio.microsoft.com/xamarin/

Xwt—Introduction [C#]. (2022). Mono Project. Získáno z https://github.com/mono/xwt (Original work published 2011)

III. Seznam obrázků, tabulek a zkratek

Seznam obrázků

Seznam tabulek

Seznam zkratek

IoT – internet věcí

API – aplikační rozhraní

CTS – část specifikace .NET platformy pro datové typy

CLR – komponenta .NET platformy zodpovědná za spouštění a běh kódu

CLI – standard .NET platformy

CIL – bajtkód .NET platformy

AOT metoda kompilování kódu „předem“

CAS – bezpečnostní technologie .NET platformy pro omezení oprávnění spuštěného kódu

COM – binární aplikační rozhraní

CLS – část specifikace .NET platformy pro programovací jazyky

VES – část specifikace .NET platformy pro zpracování bajtkódu

JIT – metoda kompilování kódu „právě v čas“

BCL – základní sada knihoven .NET platformy

WPF – moderní sada knihoven pro tvorbu grafických rozhraní

LINQ – sada knihoven pro pokročilejší zpracovávání dat

WCF – běhové prostředí určené pro vývoj vzájemně propojených aplikací

MSDN – webová stránka s technickou dokumentací produktů společnosti Microsoft

WinUI – moderní sada knihoven pro tvorbu grafických rozhraní

NGWS – původní název implementace .NET Framework

UWP – moderní sada knihoven pro tvorbu grafických rozhraní

GUI – grafické uživatelské rozhraní

IDE – integrované vývojové prostředí

LSP – protokol pro poskytování specifických vlastností programovacích jazyků

MSBuild – sestavovací systém .NET platformy

WYSIWYG – editace dokumentů při které je podoba editovaného dokumentu totožná s výsledným

IV. Přílohy

Příloha 1: CD se zdrojovým kódem aplikace a plným textem této bakalářské práce