ZÁPADOČESKÁ UNIVERZITA V PLZNI
FAKULTA ELEKTROTECHNICKÁ

KATEDRA ELEKTROMECHANIKY A VÝKONOVÉ ELEKTRONIKY

BAKALÁŘSKÁ PRÁCE

Grafické nadstavby použitelné pro .NET Framework

Jan Pokorný **2013**

Uveďte kompatibilní platformy pro jednotlivé nadstavby.

Navrhněte ukázkové příklady typu jednoduchý dialog, vykreslení bitmapy, vykreslení grafu, apod.

Ověřte příklady na pokud možno největším počtu platforem.

Uveďte možnost využití alternativních vývojových prostředí.

V příkladech preferujte jazyk C#

Abstrakt

Tato bakalářská práce prozkoumává dostupné grafické nadstavby pro jazyk C# a porovnává jejich využitelnost z hlediska uživatelské a programátorské přívětivosti. Dále zjišťuje podporu jednotlivých knihoven i na multiplatformních frameworcích, jako je Mono Project a DotGNU a uvádí pro ně i alternativní vývojová prostředí.

Klíčová slova

grafika, nadstavby, knihovny, platformy, .NET, Framework, Mono, DotGNU, C#, programování, srovnání

Abstract

This work explores the available graphics extensions for C# and compares their applicability in terms of programming and user-friendliness. Further the work finds support each library to multiplatform frameworks, such as the Mono Project and DotGNU and provides an alternative development environments.

Key words

graphic, libraries, platforms, .NET, Framework, Mono, DotGNU, C#, programming, comparison

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně, s použitím odborné literatury a pramenů uvedených v seznamu, který je součástí této práce.

Dále prohlašuji, že veškerý software, použitý při řešení této bakalářské práce, je legální.

.....
podpis

V Plzni dne 29.5.2013 Jan Pokorný

Poděkování

Tímto bych rád poděkoval vedoucímu bakalářské práce Ing. Petrovi Weissarovi, Ph.D., za cenné profesionální rady, připomínky a za metodické vedení práce a mojí mamce PaedDr. Lence Halamové za korekturu českého jazyku.

OBSAH

1. ÚVOD

1.1. CÍL PRÁCE

Cílem práce je vyhledat použitelné grafické knihovny pro vývoj graficko-uživatelských aplikacích, vytvořit v nich krátké ukázkové programy a následně ověřit jejich funkčnost na rozličných platformách. Dále si klade za cíl tyto knihovny zhodnotit z hlediska jejich programovací složitosti a výsledného graficko-uživatelského rozhraní.

1.2. ČÁSTI PRÁCE

Hlavní část práce je rozdělena do několika kapitol. První kapitola pojednává všeobecně o frameworku a jeho využití, včetně popisu .NET, Mono a DotGNU Frameworku. Druhá je zaměřena na vývojová prostředí pro C#, na jejich popis a na zhodnocení každého z nich. Třetí popisuje jednotlivé grafické knihovny a programy v nich vytvořené. Čtvrtá pak tyto programy ověřuje na čtyřech platformách (Windows 7, Kubuntu, Fedora a Raspbian) a popisuje postup pro jejich zprovoznění. Poslední kapitola pak všechno shrnuje do uceleného vyhodnocení.

2. OBECNĚ O FRAMEWORKU

2.1. CO JE TO FRAMEWORK

Framework je velmi rozsáhlý pojem, a proto se podíváme na některé jeho definice. Česká wikipedie říká, že: Framework je softwarová struktura, která slouží jako podpora při programování a vývoji a organizaci jiných softwarových projektů. Může obsahovat podpůrné programy, knihovny API[1], podporu pro návrhové vzory nebo doporučené postupy při vývoji. Cílem frameworku je převzetí typických problémů dané oblasti, čímž se usnadní vývoj tak, aby se návrháři a vývojáři mohli soustředit pouze na svá zadání. [1]

Vývojová platforma vyvinutá odborníky je základ aplikace, který může vývojář použít k vývoji vlastního programu. Programátor se už nemusí starat vždy znovu o stejné úkoly, jako jsou např. spuštění aplikace, systém oken a práce s nimi, budování GUI[2], spouštění akcí, ukládání stavu aplikace, procesy na pozadí, aktualizace aplikace. To vše už za vás zařídil a odladil někdo jiný dlouhým vývojem. Při vytváření aplikace se autor může soustředit jen na své problémy. Využije při tom platformu a přidá jen své moduly, akce, menu, zobrazení, atd. [2]

Jedná se tedy o jakousi nadstavbu, která zajišťuje správné provedení a volání nízkoúrovňových funkcí, o které se programátor nemusí starat. Programátor využívá rozhraní frameworku a soustřeďuje se na vývoj aplikace, což šetří mnoho práce a zefektivňuje vývoj. Framework může i usnadnit přechod mezi platformami, a dokonce i mezi jiným typem hardwaru. Stačí, aby byl vyvinut i pro onu platformu, respektive hardware a program poběží i tam.

2.2. ARCHITEKTURA

Framework se skládá z tzv. frozen spots a hot spots. Frozen spots definují celkovou architekturu softwarové struktury, její základní komponenty a vztahy mezi nimi. Tyto části se nemění při žádném použití frameworku. Naproti tomu hot spots jsou komponenty, které spolu s kódem programátora vytvářejí zcela specifickou funkcionalitu, a proto jsou skoro pokaždé jiné. V objektově orientovaném prostředí je framework tvořen abstraktními a

[1] API (Application Programming Interface) je označení pro rozhraní aplikovaného programování.
[2] GUI (Graphical User Interface) je graficko uživatelské rozhraní.

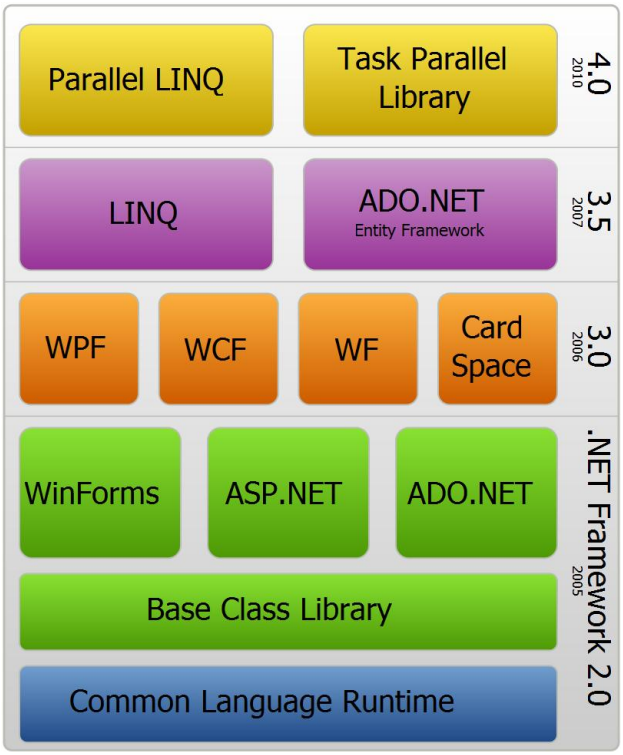
klasickými (neabstraktními) třídami. Hot spots pak mohou být reprezentovány abstraktními třídami a vlastní kód se přidá implementací abstraktních metod. [1]

2.3. POUŽITÉ FRAMEWORKY

2.3.1. MICROSOFT .NET FRAMEWORK

Microsoft .NET Framework je základní vývojový prostředek pro operační systém Windows, snažící se konkurovat Javě. Jeho součástí je velké množství knihoven, kompletní běhové prostředí pro spouštění a provoz programů a objektově orientované nástroje pro vývoj moderních, uživatelsky příjemných aplikací.

Další odvozené frameworky jsou Microsoft .NET Compact Framework - platforma určená pro kapesní počítače a mobilní telefony s operačním systémem Windows Mobile; a Microsoft .NET Micro Framework - platforma určená pro embedded zařízení, s ještě menší výpočetní kapacitou a většími omezeními, než kapesní počítače [3].



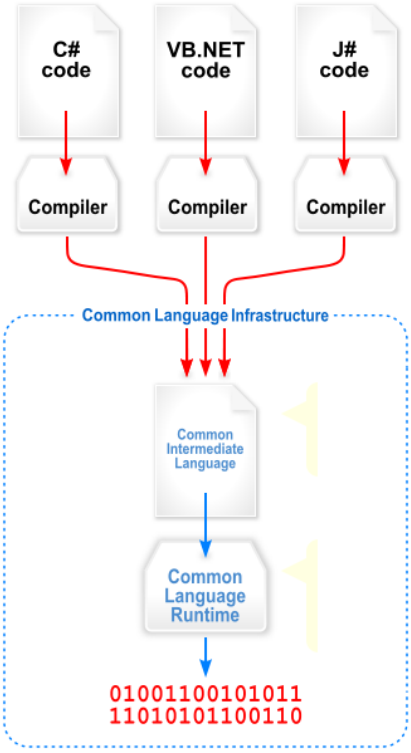
Obr. 1: Architektura .NET Frameworku

Jak ukazuje Obr. 1, celá architektura frameworku stojí na Common Language Runtime (CLR), který tvoří nejdůležitější část frameworku. Tuto část si rozebereme podrobněji v následujícím odstavci.

2.3.1.1. Common Language Infrastructure (CLI)

CLI je obecná infrastruktura frameworku. Microsoft ji implementoval v CLR a CIL[3]. CIL obsahuje definici jazyka, tzv. bytecodu, do kterého kompilátor některého vyššího jazyka (nejčastěji C#, Visual Basic .NET, Delphi nebo J#) zkompiluje kód. Tento kód je následně univerzální, a proto jej můžeme libovolně přenést i na jiné platformy. Tam je tento kód zpracován CLR, která podle dané platformy a hardwaru realtimově[4] překládá kód do nativních instrukcí procesoru pomocí metody JIT[5]. Jedná se o tzv. Virtual Machine neboli virtuální stroj, někdy též označovaný jako Virtual Execution System. Celý tento proces zaznamenává přehledně následující obrázek (Obr. 2).

Dalšími součástmi tohoto systému jsou reakce na výjimky a runtime chyby[6], které právě ono prostředí zachytí, a aplikace tzv. nespadne, pouze se násilně ukončí a framework podá chybové hlášení. Následuje automatická správa paměti, tzv. Garbage Collector, která má za úkol mazat již nevyužívané alokace paměti a následně hledat místo pro nově potřebné. Za nevyužitou alokaci se pokládá taková paměť, kam již neukazuje žádná reference, či ukazatel. (čerpáno z [4], [5])



Obr. 2: Zpracování kódu .NET Frameworkem

[3] CIL je zkratka pro Common Intermediate Language.
[4] Realtime – (tzv. v reálném čase) jedná se o typ zpracování, který probíhá okamžitě.
[5] JIT (Just-in-time) je speciální metoda překladu využívající různé techniky pro urychlení běhu.
[6] Runtime chyba je chyba, která vznikla za běhu aplikace, nikoli např. při jejím překladu.

2.3.1.2. Knihovny

Součástí frameworku je i velké množství knihoven. Ty základní jsou souhrnně označovány jako BCL[7], které obsahují matematické funkce, vstupně-výstupní streamy, podporu pro kolekce, práci se sítěmi apod. Rovněž tvoří základ pro další knihovny, které jsou součástí frameworku od verze 3.0 (viz. Obr. 1), zejména:

- WinForms pro tvorbu grafických API,
- ASP.NET pro vývoj webových aplikací,
- ADO.NET představující množinu tříd nabízejících služby pro přístup k datům a k tvorbě databázových aplikací [6],
- Windows Communications Foundation (WCF), technologie pro vývoj webových služeb a komunikační infrastruktury aplikací,
- Windows Workflow Foundation (WF), technologie pro definování heterogenních sekvenčních procesů,
- Windows Presentation Foundation (WPF), technologie pro vytváření vizuálně působivého grafického uživatelského rozhraní pro aplikace,
- Windows CardSpace, implementace standardu Information Cards a
- LINQ – Language Integrated Query, objektový přístup k datům v databázi, XML a objektech, které implementují rozhraní IEnumerablez. [3]

2.3.1.3. Shrnutí

Microsoft .NET Framework tvoří spolu s Visual Studiem perfektní prostředek pro rychlý vývoj grafických, webových i serverových aplikací a knihoven. Má rozsáhlou podporu dostupnou online, a to i v českém jazyce (viz [7]). Jeho nevýhodou je omezená dostupnost pouze na platformu Windows.

Obr. 3: .NET Framework

[7] BCL je zkratka pro Base Class Library.

2.3.2. MONO PROJECT

Mono je softwarová platforma, navržená tak, aby poskytovala vývojářům možnost vyvíjet i multiplatformní aplikace. Jedná se o open-source implementaci Microsoft .NET Frameworku založenou na standardu ECMA[8].

Mezi podporované operační systémy patří Linux, Microsoft Windows, Mac OS X, BSD a Sun Solaris, Nintendo Wii, Sony PlayStation 3 a Apple iPhone. [8]

Podporované architektury v kombinaci s operačním systémem ukazuje následující tabulka (Tab. 1).

Tab. 1: Podporované architektury Mono projektu

Supported Architectures	Runtime	Operating system
s390, s390x (32 and 64 bits)	JIT	Linux
SPARC (32)	JIT	Solaris, Linux
PowerPC	JIT	Linux, Mac OSX, Wii, PlayStation 3
x86	JIT	Linux, FreeBSD, OpenBSD, NetBSD, Microsoft Windows, Solaris, OS X, Android
x86-64: AMD64 and EM64T (64 bit)	JIT	Linux, FreeBSD, OpenBSD, Solaris, OS X
IA64 Itanium2 (64 bit)	JIT	Linux
ARM: little and big endian	JIT	Linux (both old and new ABI), iPhone, Android
Alpha	JIT	**not maintained**. Linux
MIPS	JIT	Linux
HPPA	JIT	**not maintained** Linux

Mono se snaží dosáhnout plné podpory funkce .NET Framework 4.0, ale kvůli složitosti neimplementuje Windows Presentation Foundation (WPF), Entity Framework (ADO.NET) a Windows Workflow Foundation (WF). Omezeně je jím pak podporován jen Windows Communication Foundation (WCF). Ovšem některé tyto chybějící součásti jsou vyvíjeny samostatně v experimentálním Mono projektu nazývaném Olive. Open-source implementace Silverlightu se nazývá Moonlight, který je součástí Mono od verze 1.9. Vývoj Moonlightu byl však ukončen, a to poslední verzí 2.0, která je kompatibilní se Silverlight 2.0 a některými funkcemi z 3.0. [9]

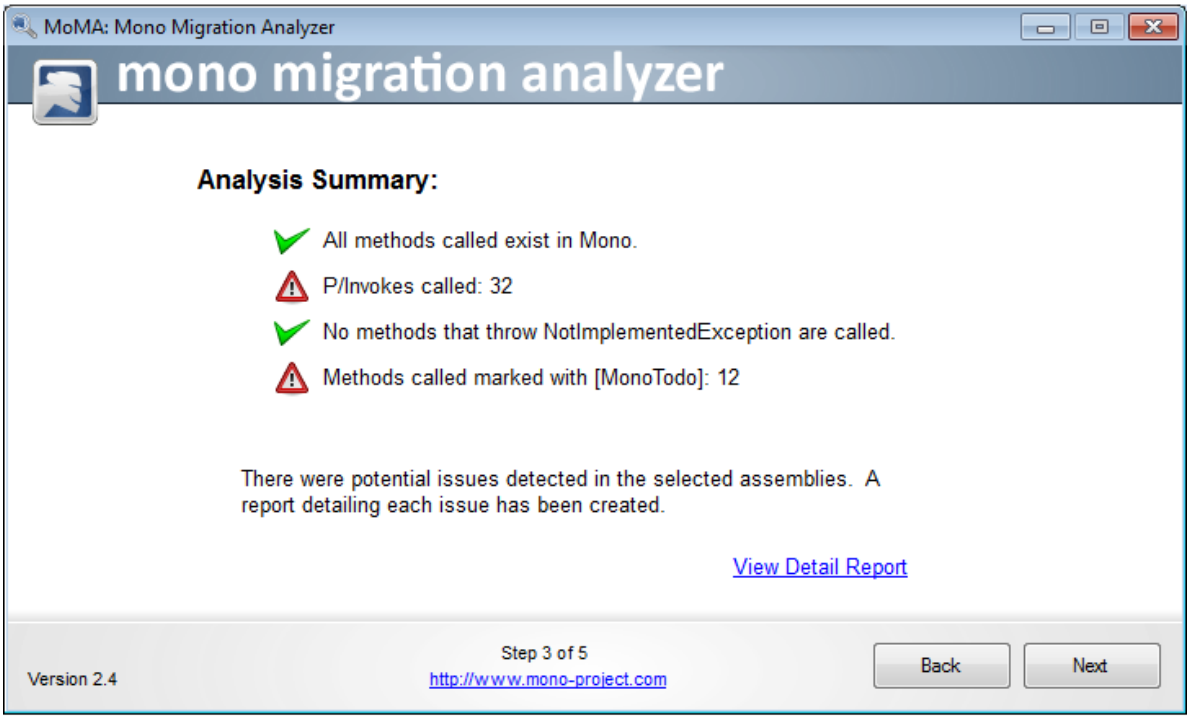
[8] ECMA je společnost, která normalizovala standard pro C# a CLI, jako Ecma-334 a Ecma-335.

2.3.2.1. Mono Migration Analyzer

Jedná se o nástroj pomáhající identifikovat problémy, které by mohly nastat při přenášení .NET aplikací na Mono. Konkrétně pomáhá určit specifická volání jednotlivých platforem (označujících se jako P / Invoke) a také oblasti, které dosud nejsou podporovány. [10]

Chyby, na které nástroj upozorňuje:

- Missing Methods – Jedná se o chybějící metody, které stále nejsou v Mono projektu implementovány. Takovéto programy nejdou spustit vůbec.
- P/Invokes – Toto značí přímá volání nativního kódu, která Mono umí spustit jen v případě, že je tento kód pro platformu dostupný (např. v knihovnách).
- NotImplementedException – Jedná se speciální typ výjimky, která je vyvolána při chybějící metodě, což může i nemusí způsobit pád aplikace.
- MonoTodo – Tímto jsou označovány metody, které buďto mají prázdné tělo (metoda existuje, ale nevykoná žádnou funkci) nebo jsou nekompletní a neodladěné (takové metody mohou i nemusejí způsobit pád aplikace). [11]



Obr. 4: Ukázka programu MoMA

2.3.2.2. Součásti Mono projektu

Mezi základní součásti Mono projektu patří:

- C# Kompilátor – Kompletní kompilátor pro jazyk C# od verze 1.0 až po 4.0.
- Mono Runtime – Jedná se o implementaci Common Language Infrastructure, který poskytuje JIT kompilátor, knihovny, garbage collector, systém vláken a výjimek.
- Base Class Library (BCL) – Komplexní sada knihoven kompatibilních s .NET Frameworkem.
- Mono Class Library – Rozšířená sada knihoven, které jsou nad rámec BCL, jako například třídy Gtk+, zip soubory, LDAP[9], OpenGL[10], atd. Ty jsou užitečné zejména pro vývoj aplikací pod Linuxem. [8]

Obr. 5: Mono projekt

2.3.2.3. Shrnutí

Mono projekt je velice dobrá alternativa k .NET Frameworku, což ho rozhodně staví na jedno z prvních míst pro vývoj multiplatformních aplikací. Jeho nevýhodou je ovšem opoždění vývoje za konkurenčním Microsoftem a neimplementování některých částí .NET Frameworku.

2.3.3. DotGNU

DotGNU je součásti projektu GNU, jehož cílem je poskytnout svobodnou softwarovou náhradu za Microsoft .NET Framework. Dále se snaží dosáhnout lepší podpory na jiných než Windows platformách (GNU/Linux (na PC, Sparc, iPAQ, Sharp Zaurus, PlayStation 2,

[9] LDAP (Lightweight Directory Access Protocol) je definovaný protokol pro práci s daty na serveru.
[10] OpenGL je multiplatformní API pro tvorbu grafiky.

Xbox,...), *BSD, Cygwin/Mingw32, Mac OS X, Solaris, AIX) a na více typech procesorů (x86, ppc, arm, parisc, s390, ia64, alpha, mips, sparc). Základem je sada tříd, která se snaží být stoprocentně kompatibilní s Common Language Infrastructure. [12]

Hlavní části projektu DotGNU jsou:

- Portable.NET – Implementace standardu CLI dle normy Ecma-335. Zahrnuje software pro kompilaci a spouštění jazyků Visual Basic.NET, C# a C, které využívají BCL, XML a Windows Forms.
- phpGroupWare – Uživatelská webová souprava, která poskytuje informace o webových aplikacích.
- DGEE[11] - Prostředí pro webovou správu.
- libJIT – Knihovna pro vývoj pokročilých aplikací, které využívají Virtual Machine a Just-in-time kompilaci. [13] (ověřeno zde [12])

2.3.3.1. Shrnutí

Jedná se o další multiplatformní alternativu k Microsoft .NET Frameworku, ale protože v prosinci roku 2012 byl vývoj zastaven (viz [12]), nebudeme tento framework dále rozebírat.

Obr. 6: DotGNU

[11] DGEE je zkratka pro DotGNU Execution Environment.

3. VÝVOJOVÉ PROSTŘEDÍ

Vývojové prostředí je program, který slouží pro snadný vývoj aplikace. Jedná se ve své podstatě o specializovaný editor textu, umožňující nejrůznější úpravy a editace, jako třeba refactoring – kompletní přejmenování názvů metod, funkcí, proměnných atd., syntax-coloring – neboli obarvování klíčových slov a dalších speciálních typů textu použitých při programování, což podstatně zpřehledňuje celý program. Samozřejmostí je ukládání, načítání a vytváření projektů se všemi dostupnými nastaveními. Co už samozřejmostí není, je kompilátor a linker, který se většinou instaluje a vkládá zvlášť. Takovéto programy se většinou označují jako IDE[12].

3.1. MICROSOFT VISUAL STUDIO

3.1.1. VÝVOJ

Poprvé bylo vydáno v roce 1997 jako Microsoft Visual Studio 97, rok nato pak Microsoft Visual Studio 6.0. Obě tato prostředí ještě neměla podporu .NET a obsahovala jen základní jazyky, jako C/C++, Visual Basic, Visual J++ a nástroje pro web a databáze. Zlom přišel s verzí Microsoft Visual Studio .NET (2002), kde Microsoft umožnil vývoj pod .NET Frameworkem v jazycích Visual Basic.NET a C#. Dalších několik verzí Visual Studia (2005, 2008 a 2010) jen přidával a zlepšoval funkci jednotlivých nástrojů a jazyků tak, aby mohl využít všechny funkcionality nových verzí frameworku. Verze 2010 představila ještě i nový multiparadigmatický[13] jazyk F#. Aktuální verze je Visual Studio 2012, které budeme v této práci využívat. (čerpáno z [14] a z [15])

3.1.2. SOUČÁSTI

3.1.2.1. Editor kódu

Samozřejmostí je editor kódu, který umožňuje napovídání a automatické dokončování pomocí IntelliSense. Obsahuje i další funkce pro snadnou navigaci a hledání v kódu s podporou regulárních výrazů, dále číslování řádek, skrývání částí kódu, podporuje vícepoložkovou schránku a seznam úkolů (tzv. ToDo), které se zobrazí ve speciálním okně. [15]

[12] IDE (Integrated Development Environment) je integrované vývojové prostředí.
[13] Multiparadigmatický jazyk je jazyk, který podporuje více než jeden typ návrhového vzoru programu.

3.1.2.2. Debugger

Visual Studio obsahuje debugger, který pracuje jak se spravovaným kódem, tak se strojovým kódem. Debugger Visual Studia může také vytvořit obsahy paměti a později je nahrát pro debugging. Jsou také podporovány vícejádrové programy. [15]

3.1.2.3. Designer

- WinForms Designer

WinForms designer je používán pro GUI aplikace za použití WinForms. Obsahuje paletu ovládacích prvků (včetně tlačítek, progress barů, popisek a jiných prvků), které mohou být uchopeny a umístěny na povrch formuláře. Rozložení je možné ovládat ukládáním prvků do kontejnerů nebo uzamykáním na stranu formuláře.

- WPF Designer

WPF designer, zvaný Cider, byl představen ve Visual Studiu 2008. Stejně jako WinForms designer podporuje drag-and-drop. Vytváří se s ním uživatelské rozhraní pro Windows Presentation Foundation. Podporuje všechny funkce WPF včetně propojení dat a automatickou správu rozložení. Generuje XAML[14] kód pro UI.

- Web designer

Visual Studio také obsahuje editor webových stránek a designer, který je umožňuje vytvářet uchopováním a pokládáním prvků. Je používán pro vývoj aplikací ASP.NET a podporuje HTML, CSS a JavaScript. Používá code-behind model pro spojení s kódem ASP.NET. Zobrazovací engine je sdílen s Microsoft Expression Webem.

- Designer tříd

Designer tříd se používá pro vytváření a úpravu tříd (včetně jejich členů a přístupu) použitím modelu UML. Designer tříd může vygenerovat kódy C# a VB.NET pro třídy a metody. Také může vygenerovat diagramy z ručně psaných tříd.

- Designer dat

Designer dat může být použit pro grafickou úpravu databázových schémat, včetně psaných tabulek, primárních a cizích klíčů a omezení. Také může být použit pro design dotazů v grafickém zobrazení.

[14] XAML je značkovací jazyk podobný HTML.

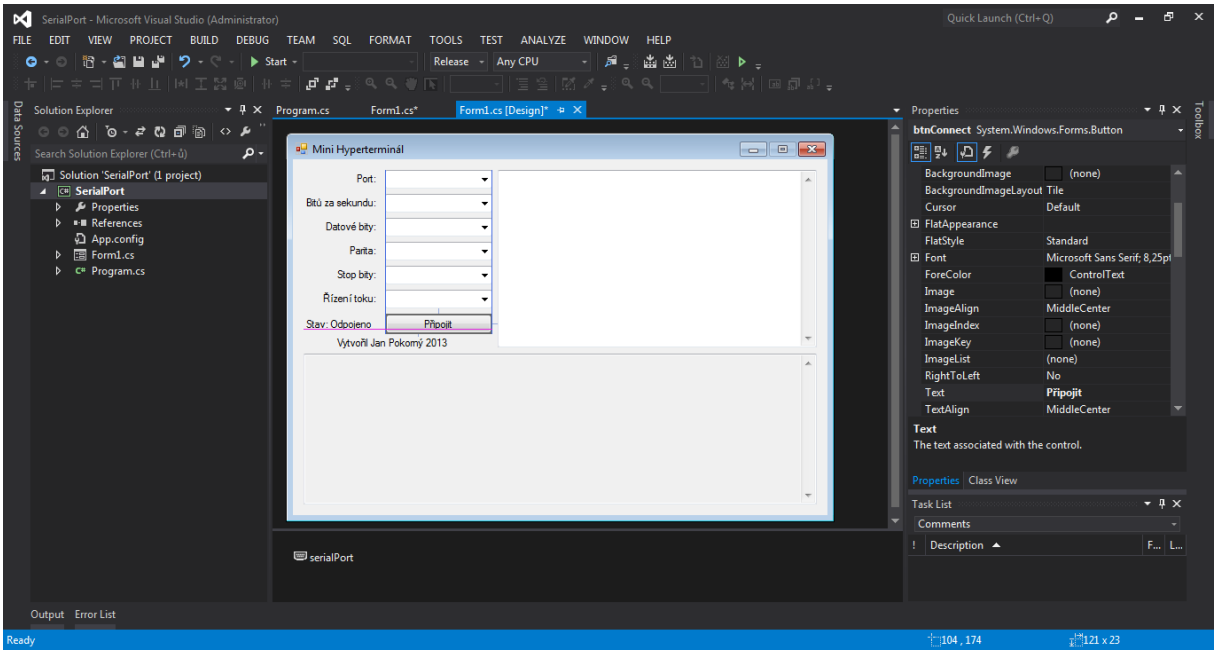
- Designer mapování

Od Visual Studia 2008 je designer mapování používán funkcí LINQ to SQL k zobrazení mapování mezi databázovými schématy a třídami, které zabalují data. [15]

3.1.3. SHRNUTÍ

Visual Studio je velmi komplexní vývojové prostředí podporující vývoj od mobilních přes desktopové až po webové aplikace. Je jedním z nejprofesionálnějších prostředí určených ke správě rozsáhlých programů a projektů celkově. Obsahuje sadu funkcí pro ulehčení vývoje, jako je třeba ladicí systém, grafický návrhář s automatickým zarovnáním nebo třeba prohlížeč závislostí a návazností jednotlivých metod nebo i celých tříd. Lze si nechat i graficky vygenerovat tyto závislosti v podobě obrázku pro snadnější orientaci v programu.

Jeho nevýhodou je jeho komerční vývoj, a tedy i cena, která se pohybuje řádově od 15‘000 Kč do 150‘000 Kč. Pro studenty, jejichž škola spolupracuje se společností Microsoft, je verze Professional zdarma.



Obr. 7: Microsoft Visual Studio 2012

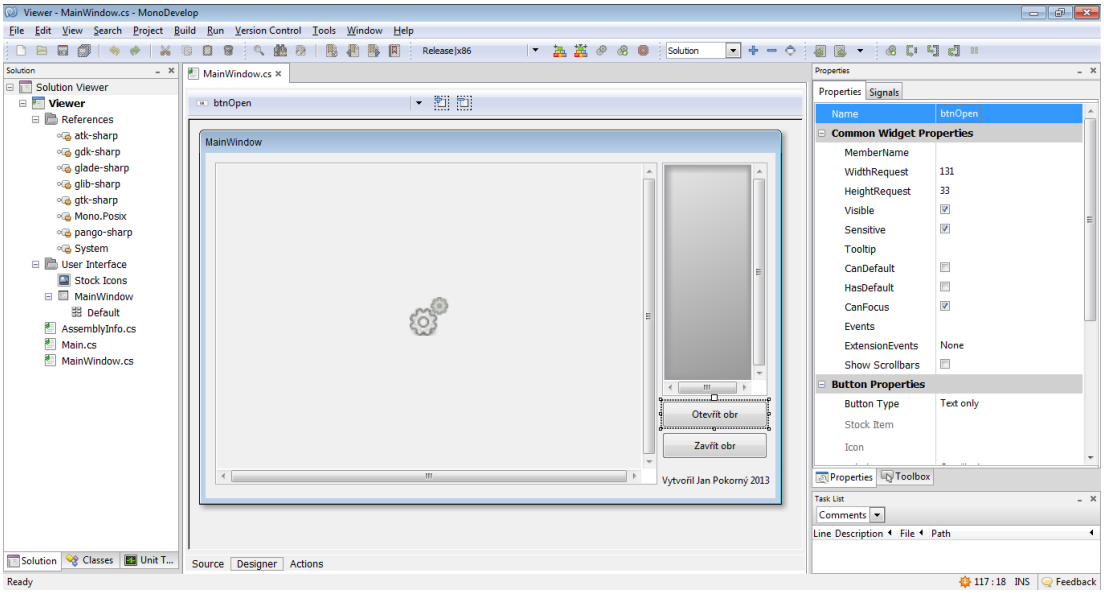
3.2. MONODEVELOP

3.2.1. SOUČÁSTI

- Pokročilé editování textu - podpora kompletování textu pro C# 4, návrhové vzory i skrývání bloků kódu
- Konfigurovatelná plocha - možnost vlastního uspořádání vzhledu, definic vlastních klávesových zkratek a připojení externích nástrojů
- Podpora více jazyků - C#, Visual Basic.Net, C/C++, Vala
- Integrovaný Debugger - pro debugování Mono a nativních aplikací
- GTK# designer – pro snadné vyvíjení GTK# aplikací
- Další nástroje - integrace makefile souborů, unit testing[15] a podpora lokalizace [16]

3.2.2. SHRNUTÍ

MonoDevelop je multiplatformní IDE primárně určené k navrhování aplikací v .NET jazycích, zejména C# a ASP.NET. Umožňuje vývojářům rychle psát desktopové i webové aplikace pro Linux, Windows i Max OSX. Je určený pro tvorbu graficko-uživatelských aplikací, zejména pro knihovnu GTK#. Rovněž obsahuje grafický návrhář, ovšem s podstatně menšími možnostmi než u Visual Studia. Ani jeho okno s Properties (vlastnostmi) neobsahuje veškerá nastavení komponent, a tak spoustu jich je nutno dopsat ručně v kódu. Výhodou ale je, že je zdarma pod licencí LGPL[16].



Obr. 8: MonoDevelop

[15] Unit testing je funkce pro ověřování správné funkčnosti programu.
[16] LGPL je licence svobodného softwaru, publikovaná Free Software Foundation.

3.3. SHARPDEVELOP

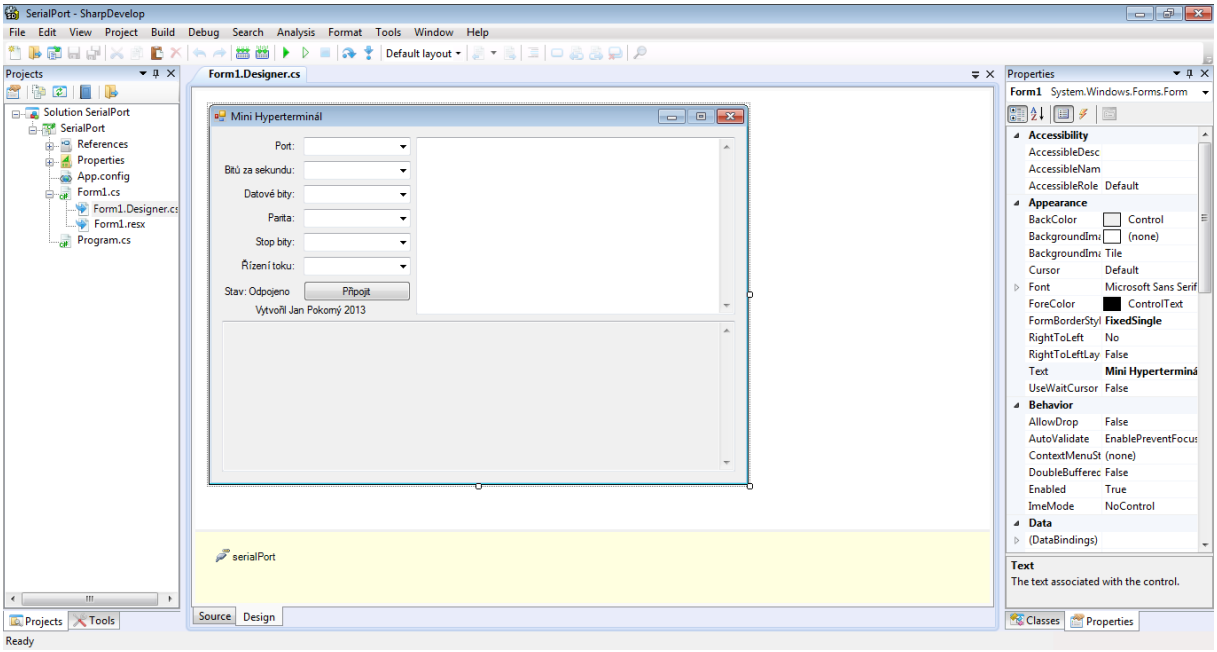
SharpDevelop, někdy označován jen jako #Develop, je Open-source IDE pro vývoj .NET aplikací až do verze 4.0.

Mezi podporované jazyky patří C#, VB.NET, Boo, IronPython, IronRuby a F#. Dále jsou podporovány grafické nadstavby včetně designeru pro Windows Presentation Foundation (WPF), Windows Forms a ASP.NET MVC. SharpDevelop též umožňuje vývoj C/C++ aplikací bez použití .NET Frameworku. [17]

3.3.1. SHRNUTÍ

SharpDevelop je velmi dobrá náhrada komerčního Visual Studia. Obsahuje mnohé jeho funkce a utility pro snadnou práci s návrhem i laděním aplikace a na rozdíl od Visual Studia je zdarma.

Nevýhodou je dostupnost pouze pro Windows.



Obr. 9: SharpDevelop

4. GRAFICKÉ KNIHOVNY A PROGRAMY

4.1. GRAFICKÉ KNIHOVNY

4.1.1. WINFORMS

Celá tato grafická knihovna je již implementována v .NET Frameworku, proto není potřeba ji doinstalovávat zvlášť. Nabízí celou řadu komponent, které jsou již navrženy tak, aby co nejvíce splňovaly požadavky na vývoj.

Všechny WinForms aplikace byly programovány v prostředí Visual Studia 2012.

4.1.2. GTK#

Jedná se o multiplatformní knihovnu využívající Cario knihovnu od verze 2.0. Cario poskytuje základní API pro tvorbu grafických aplikací fungujících na OpenGL. GTK je využíváno jako standardní grafické prostředí GNOME, které nahradilo starší knihovnu Motif.

K programování GTK aplikací je využit MonoDevelop.

4.1.3. QT4DOTNET

Qt4DotNet (neboli Qt pro .NET, stažitelné z odkazu [18]) využívá známou grafickou knihovnu Qt. Problémem je, že nikoli přímo, ale prostřednictvím QtJambi, který je vytvořen pro Javu. Qt4DotNet je tedy jakousi nadstavbou nad QtJambi a teprve ta nad nativním Qt. Toto je velká nevýhoda, protože ke spuštění jsou potřebné nejen nativní Qt, ale i JRE[17] a knihovna QtJambi, což velmi komplikuje práci, zejména na systémech Linux.

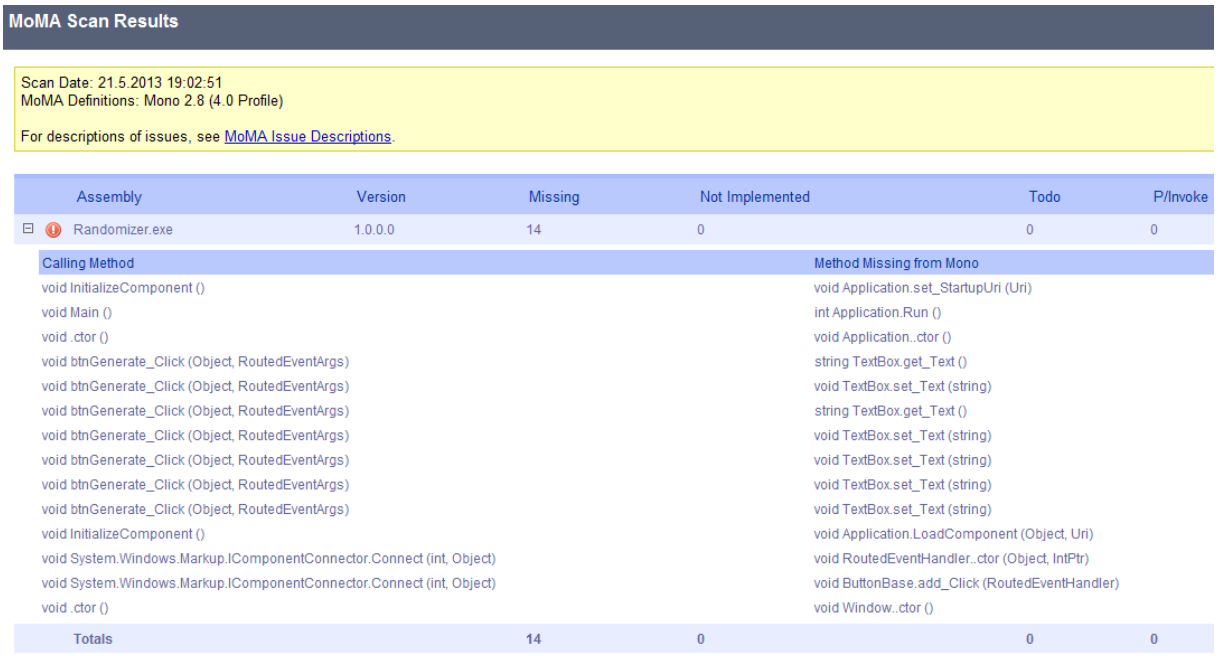
Qt aplikace byly tvořeny ve Visual Studiu 2012.

4.1.4. WPF

Knihovna Windows Presentation Foundation je grafická knihovna podporující vektorovou grafiku. Rovněž je celá implementována v .NET Frameworku od verze 3.0.

Jako jediný program zde byl vytvořen Randomizer, na kterém byla skutečně ověřena nepřenositelnost WPF aplikací pod Linux (Mono). Jak ukazuje následující obrázek s výpisem chyb z programu Mono Migration Analyzer, Mono dosud neimplementuje důležité části potřebné pro spuštění.

[17] JRE (Java Runtime Environment) je prostředí pro spouštění Java aplikací.



MoMA Scan Results

Scan Date: 21.5.2013 19:02:51
MoMA Definitions: Mono 2.8 (4.0 Profile)

For descriptions of issues, see MoMA Issue Descriptions.

Assembly	Version	Missing	Not Implemented	Todo	P/Invoke
Randomizer.exe	1.0.0.0	14	0	0	0

Calling Method	Method Missing from Mono
void InitializeComponent ()	void Application.set_StartupUri (Uri)
void Main ()	int Application.Run ()
void .ctor ()	void Application..ctor ()
void btnGenerate_Click (Object, RoutedEventArgs)	string TextBox.get_Text ()
void btnGenerate_Click (Object, RoutedEventArgs)	void TextBox.set_Text (string)
void btnGenerate_Click (Object, RoutedEventArgs)	string TextBox.get_Text ()
void btnGenerate_Click (Object, RoutedEventArgs)	void TextBox.set_Text (string)
void btnGenerate_Click (Object, RoutedEventArgs)	void TextBox.set_Text (string)
void btnGenerate_Click (Object, RoutedEventArgs)	void TextBox.set_Text (string)
void btnGenerate_Click (Object, RoutedEventArgs)	void TextBox.set_Text (string)
void InitializeComponent ()	void Application.LoadComponent (Object, Uri)
void System.Windows.Markup.IComponentConnector.Connect (int, Object)	void RoutedEventHandler..ctor (Object, IntPtr)
void System.Windows.Markup.IComponentConnector.Connect (int, Object)	void ButtonBase.add_Click (RoutedEventHandler)
void .ctor ()	void Window..ctor ()

Totals	14	0	0	0

Obr. 10: Výpis chyb z MoMA pro WPF aplikaci Randomizer

4.1.5. GWEN.NET

Tato grafická knihovna, vytvořená Garry Newmanem, je kompletním přepsáním C++ verze knihovny GWEN, primárně určené pro vývoj her. Součástí knihovny jsou i GUI komponenty pro tvorbu neherních aplikací.

Pro tuto knihovnu neexistují žádné návody ani tutoriály, proto ani nebyly vytvořeny žádné programy. Ani samostatným testováním stylem „pokus-omyl“ se nepodařilo docílit spuštění funkční aplikace.

Knihovna je dostupná na internetu [19].

4.2. PROGRAMY

Následující ukázky kódu jsou buďto klíčové části, nebo části výrazně odlišné mezi knihovnami. Pro úplné pochopení je potřeba prostudovat celý kód.

4.2.1. RANDOMIZER

Prvním jednoduchým programem je generátor náhodných čísel. Ukazuje nám možnosti zadávání čísel v textovém formátu a jejich parsování. Pomocí matematického jádra frameworku následně generuje náhodné číslo v zadaném rozsahu.

Implementace tohoto programu proběhla bez potíží, protože všechny potřebné funkce i prostředky byly u všech knihoven podobné. Jedině u Qt je potřeba ručně přidávat všechny akce prostřednictvím tohoto kódu:

```
this.btnGenerate.OnMousePressEvent += new Action<QPushButton,
                com.trolltech.qt.gui.QMouseEvent>(btnGenerate_Click);
```

4.2.2. GRAPHS

Druhý program je založen na „zakreslování“ jednotlivých bodů, které jsou spojovány do lomené čáry nebo zaoblené křivky. Program obsahuje: výběr typu křivky pomocí komponenty RadioButton, vypnutí zobrazení bodů pomocí CheckButton, ruční zadávání bodů přes dvě textová pole (TextBox), označená X a Y, a možnost celý graf uložit jako obrázek.

Úkolem tohoto programu je zjistit hlavně způsob vykreslování grafických primitiv, vhodných např. pro grafy a podporu obrázkových formátů pro ukládání.

Zde byla implementace obtížnější, protože každá z knihoven řeší vykreslování unikátním způsobem.

- WinForms

Ukládání plochy jako obrázek je vytvořeno takto:

```
Bitmap bmp = new Bitmap(this.pctGraph.Width, this.pctGraph.Height);
this.pctGraph.DrawToBitmap(bmp, new Rectangle(new Point(0, 0),
                this.pctGraph.Size));
bmp.Save(this.saveFile.FileName);
```

Pro vykreslování je zde event Paint, který vygeneruje metodu s argumentem PaintEventArgs e. Třída pro zpracování grafiky se následně získá takto:

```
Graphics g = e.Graphics;
```

Pro vykreslování primitiv jsou hotovy základní metody:

```
g.DrawRectangle(pBlack, new Rectangle(p, new Size(2, 2)));
g.DrawLines(pBlack, this.points.ToArray());
g.DrawCurve(pBlack, this.points.ToArray());
```

- GTK

K povolení událostí myši slouží tento kód:

```
this.pctGraph.AddEvents((int)Gdk.EventMask.ButtonMotionMask);//povolení
událostí
this.pctGraph.AddEvents((int)Gdk.EventMask.ButtonReleaseMask);
```

Knihovna má pro vykreslování událost ExposeEvent. Získání přístup k tomuto vykreslování se provede takto:

```
Gdk.GC gc = new Gdk.GC(this.pctGraph.GdkWindow);
//následuje průběžné ukládání do bufferu
this.pixbuf.GetFromDrawable(this.pctGraph.GdkWindow,
                this.pctGraph.GdkWindow.Colormap, 0, 0, 0, 0, w, h);
```

Pro vykreslení primitiv jsou hotovy tyto základní metody:

```
this.pctGraph.GdkWindow.DrawRectangle(gc, true, r);
this.pctGraph.GdkWindow.DrawLines(gc, this.points.ToArray());
```

Ukládání plochy jako obrázek je vytvořeno takto:

```
this.pixbuf.Save(fName, fSuf);
```

- Qt4DotNet

Ukládání plochy jako obrázek je vytvořeno takto:

```
QPixmap pixmapPrinter = new QPixmap(size);
this.pctGraph.render(pixmapPrinter);
QPainter painter = new QPainter(pixmapPrinter);
painter.drawPixmap(printingRect, pixmapPrinter);
bool isOk = pixmapPrinter.save(fileName);
```

U knihovny Qt se událost pro vykreslení jmenuje OnPaintEvent. Přístup k vykreslování provádí QPainter takto:

```
QPainter painter = new QPainter(this.pctGraph);
```

Kreslení primitiv se provádí takto:

```
painter.drawRect(new QRect(p, new QSize(2, 2)));
painter.drawLine(p1, p2); //p1 a p2 jsou body
```

4.2.3. VIEWER

Viewer neboli prohlížeč je třetí z programů, který umožňuje načítání obrázků a jejich zobrazení. Obsahuje komponentu ListBox pro snadné přepínání již načtených obrázků, které lze i z listu smazat prostřednictvím tlačítka Smazat obr.

Program je zaměřen na ověření schopnosti načítat rozličné obrázkové formáty a zobrazovat je. Dále ukazuje využití a implementaci ListBoxu.

Implementace prohlížeče:

- WinForms

Filtr pro OpenDialog se zadává takto:

```
this.openFile.Filter = "Bitmap (*.bmp)|*.bmp|Image (*.jpg)|*.jpg|GIF
                (*.gif)|*.gif|PNG (*.png)|*.png|All (*.*)|*.*";
```

O celé načtení se zde stará třída Bitmap:

```
Bitmap bmp = new Bitmap(this.openFile.FileName);
```

Vykreslení je také věcí jednoduchou:

```
this.pctMain.Image = bmp;
```

- GTK

Základní chybou je, že GTK nemá přímo ListBox jako WinFroms, ale jen TreeView, pomocí kterého se onen ListBox dá napsat:

```
public class ListBox : TreeView
{
        private ListStore ls;
        public ListBox(string title)
        {
                Gtk.TreeViewColumn column = new Gtk.TreeViewColumn();
                column.Title = title;
                Gtk.CellRendererText cell = new Gtk.CellRendererText();
                //přidání buňky do sloupce
                column.PackStart(cell, false);
                column.AddAttribute(cell, "text", 0);
                this.AppendColumn(column);
                this.ls = new ListStore(typeof(string));
                this.Model = ls;
        }
}
```

O celé načtení se zde stará třída Pixbuf:

```
Gdk.Pixbuf pixbuf = new Gdk.Pixbuf(fName);
```

Vykreslení je také věcí jednoduchou:

```
this.pctMain.Pixbuf = pixbuf;
```

- Qt4DotNet

Filtr pro QFileDialog se zadává takto:

```
string fileName = QFileDialog.getOpenFileName(this, "Otevřít", "", new
                QFileDialog.Filter("Bitmap (*.bmp);;Image (.jpg);;GIF
                (*.gif);;PNG (*.png);;All (*.*)"));
```

O celé načtení se zde stará třída QPixmap:

```
QPixmap pixmapPrinter = new QPixmap();
pixmapPrinter.load(fileName);
```

Vykreslení je rovněž jednoduché:

```
this.pctMain.setPixmap(pixmapPrinter);
```

4.2.4. SERIAL PORT

Posledním z testovacích programů je Mini hyperterminál, prozkoumávající možnosti sériového portu. Má kompletní nastavení pro standardní funkce sériového portu a možnost připojit se na něj. V horním textovém poli pak lze zadávat znaky, které se okamžitě přenášejí přes COM. Naopak dole je textové pole pro příjem znaků.

Tento program byl zvolen proto, že na vývojových, případně embedded zařízeních je komunikace skrz sériovou linku velmi využívaná a potřebná.

Nejdůležitější části implementace hyperterminálu:

- WinForms

Žádost o přepnutí do grafického vlákna:

```
this.txtCnsRec.Invoke((MethodInvoker)delegate { this.txtCnsRec.Text =
                this.txtRecText; });
```

- GTK

Žádost o přepnutí do grafického vlákna:

```
Gtk.Application.Invoke(delegate { this.txtCnsRec.Buffer.Text = this.rcvText;
                });
```

- Qt4DotNet

V knihovně Qt se nepodařilo dohledat metodu Invoke ani jinou alternativu. Aplikace tedy nepřijímá znaky.

5. TESTOVÁNÍ

5.1. WINDOWS 7 - .NET FRAMEWORK

5.1.1. POSTUP PRO SPUŠTĚNÍ

Pro spuštění .NET aplikací je potřeba nainstalovat příslušný .NET Framework, který je dostupný na oficiálních stránkách Microsoftu. Jeho instalace je snadná a v českém jazyce. Všechny programy byly testovány na verzi 4.5.

5.1.2. OVĚŘENÍ PROGRAMŮ

5.1.2.1. WinForms

Jak již bylo zmíněno, WinForms jsou již součástí Frameworku, tedy s jejich spuštěním nebyl žádný problém.

- **Randomizer** – funkční

Zvládá parsovat pouze desetinnou čárku.

- **Graphs** – funkční

Umí v pořádku ukládat formáty BMP, JPG, GIF a PNG. S vykreslováním přímých i zaoblených čar není žádný problém.

- **Viewer** – funkční

Prohlížeč dokáže zobrazovat formáty BMP, JPG, GIF a PNG.

- **SerialPort** – funkční

Program na zkoušení sériového portu (Mini hyperterminál) dokáže správně odesílat i přijímat znaky přes sériovou linku. Rovněž nastavení funguje správně.

5.1.2.2. GTK#

- **Randomizer** – funkční

Zvládá parsovat pouze desetinnou čárku.

- **Graphs** – funkční (částečně)

Fungují obě verze tohoto programu, jak s bílou, tak neutrální plochou. Zvládá tedy následující kód, zajišťující bílou kreslicí plochu:

```
this.pctGraph.GdkWindow.Background = new Color(255, 255, 255);
```

Umí v pořádku ukládat formáty BMP a PNG. Nemá vlastní funkci pro vykreslování zaoblených čar.

- **Viewer** – funkční

Prohlížeč dokáže zobrazovat formáty BMP, JPG, GIF a PNG. Nezvládá načítat názvy obsahující diakritiku.

- **SerialPort** – funkční

Program na zkoušení sériového portu (Mini hyperterminál) dokáže správně odesílat i přijímat znaky přes sériovou linku. Rovněž nastavení funguje správně.

5.1.2.3. Qt4DotNet

- **Randomizer** – funkční

Zvládá parsovat pouze desetinnou čárku.

- **Graphs** – funkční (částečně)

Při spouštění se objevuje chyba Qt: Could not initialize OLE (error 80010106), na běh aplikace ovšem nemá žádný viditelný vliv.

Umí v pořádku ukládat jen formát PNG. Nemá vlastní funkci pro vykreslování zaoblených čar.

- **Viewer** – funkční (částečně)

Při spouštění se objevuje chyba Qt: Could not initialize OLE (error 80010106), na běh aplikace nemá žádný viditelný vliv.

Prohlížeč dokáže zobrazovat formáty BMP, GIF a PNG, některé PNG ale nenačte.

- **SerialPort** – funkční (částečně)

Při spuštění se objevuje chyba Qt: Could not initialize OLE (error 80010106), na běh aplikace nemá žádný viditelný vliv.

Program na zkoušení sériového portu (Mini hyperterminál) dokáže správně odesílat znaky, ale při příjmu je problém s přijímacím vláknem. Standardně se tento problém dá řešit tzv. invokováním metody, ale ani po důkladném prozkoumání funkcí QT se invoke nepodařilo najít. Ostatní nastavení funguje již správně.

5.1.2.4. WPF

- Randomizer – funkční

Zvládá parsovat pouze desetinnou čárku.

5.2. WINDOWS 7 – MONO

5.2.1. POSTUP PRO SPUŠTĚNÍ

Instalace Mona není nijak složitá a probíhá jako u každé jiné typické aplikace pro Windows. Mono projekt lze stáhnout na jeho oficiálních stránkách. Spolu s Monem se nainstaluje i vlastní příkazový řádek, nutný pro spouštění Mono aplikací.

Pro ověření správnosti nainstalování stačí v Mono Command Prompt napsat:

mono --version

Pokud vše proběhlo v pořádku, mělo by se ukázat:

Mono JIT compiler version *verze*

Aplikaci lze následně spustit jen v tomtéž příkazovém řádku, a to příkazem:

mono */cesta/název_programu.exe*

Pokud se vaše aplikace nespustí, vypíše se příslušná chyba v příkazovém řádku. V opačném případě můžeme přesměrovat výpis chyby do textového souboru pomocí příkazu:

mono */cesta/název_programu.exe* 2> *logfile.log*

Je důležité dávat pozor na to, že Mono pro Windows netoleruje v názvech mezery ani diakritiku.

5.2.2. OVĚŘENÍ PROGRAMŮ

5.2.2.1. WinForms

- **Randomizer** – funkční

Zvládá parsovat pouze desetinnou tečku. Desetinná čárka je vynechána, tedy např. číslo 1,5 je chápáno jako 15.

- **Graphs** – funkční

Umí v pořádku ukládat formáty BMP, JPG, GIF a PNG. S vykreslováním přímých i zaoblených čar není žádný problém.

- **Viewer** – funkční

Prohlížeč dokáže zobrazovat formáty BMP, JPG, GIF a PNG. V názvech nevadí mezery ani diakritika.

- **SerialPort** – funkční

Program na zkoušení sériového portu (Mini hyperterminál) dokáže správně odesílat i přijímat znaky přes sériovou linku. Rovněž nastavení funguje správně.

5.2.2.2. GTK#

- **Randomizer** – funkční

Zvládá parsovat pouze desetinnou tečku. Desetinná čárka je vynechána. Tedy např. číslo 1,5 je chápáno jako 15.

- **Graphs** – funkční

Umí v pořádku ukládat formáty BMP a PNG. Nemá vlastní funkci pro vykreslování oblených čar.

- **Viewer** – funkční

Prohlížeč dokáže zobrazovat formáty BMP, JPG, GIF a PNG. Nezvládá načítat názvy obsahující diakritiku.

- **SerialPort** – funkční

Program na zkoušení sériového portu (Mini hyperterminál) dokáže správně odesílat i přijímat znaky přes sériovou linku. Rovněž nastavení funguje správně.

5.2.2.3. Qt4DotNet

- **Randomizer** – funkční

Zvládá parsovat pouze desetinnou čárku.

- **Graphs** – funkční (částečně)

Při spouštění se objevuje chyba Qt: Could not initialize OLE (error 80010106), na běh aplikace nemá žádný viditelný vliv.

Umí v pořádku ukládat jen formát PNG. Nemá vlastní funkci pro vykreslování zaoblených čar.

- **Viewer** – funkční (částečně)

Při spouštění se objevuje chyba Qt: Could not initialize OLE (error 80010106), na běh aplikace nemá žádný viditelný vliv.

Prohlížeč dokáže zobrazovat formáty BMP, GIF a PNG. Některé PNG a BMP ale nenačte.

- **SerialPort** – funkční (částečně)

Při spouštění se objevuje chyba Qt: Could not initialize OLE (error 80010106), na běh aplikace nemá žádný viditelný vliv.

Program na zkoušení sériového portu (Mini hyperterminál) dokáže správně odesílat znaky, ale při příjmu je problém s přijímacím vláknem. Standardně se tento problém dá řešit tzv. invokováním metody, ale ani po důkladném prozkoumání funkcí QT se invoke nepodařilo najít. Ostatní nastavení funguje již správně.

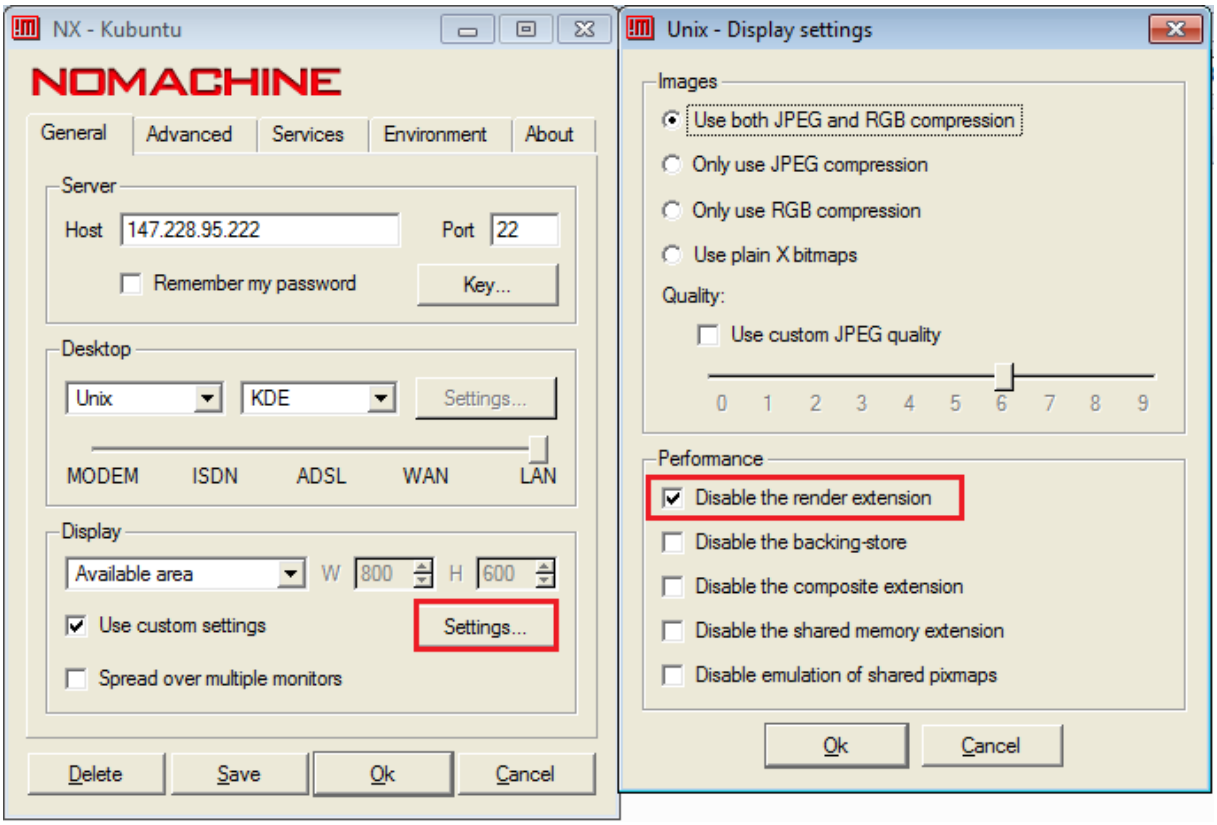
5.2.2.4. WPF

- **Randomizer** – nefunkční

Jak již bylo řečeno, WPF nejsou na Linuxu podporovány, což potvrdil i pokus s tímto programem.

5.3. KUBUNTU – MONO

Kubuntu bylo testováno přes program NoMachine (NxServer a NxClient), umožňující vzdálený přístup. Zde se vyskytla zcela zvláštní chyba, kvůli které nebyly v prostředí vidět žádné texty (popisky, tlačítka atd.) Chybu se podařilo vyřešit zákazem renderovacího rozšíření (Disable the render extension), viz následující Obr. 11. Prostředí Kubuntu bylo KDE.



Obr. 11: Nastavení pro Kubuntu

5.3.1. POSTUP PRO SPUŠTĚNÍ

Pro spuštění .NET aplikací je potřeba nainstalovat Mono. Nejsnadnější je zde spustit Terminál (konzoli) a napsat příkaz pro stažení a nainstalování balíku mono-complete:

sudo apt-get install mono-complete

V případě, že instalace nenalezne online cesty k souborům, je nutno provést update systému přes:

sudo apt-get update

Ověření správnosti nainstalování Mona se provede příkazem:

mono --version

Pokud vše proběhlo v pořádku, mělo by se ukázat:

Mono JIT compiler version *verze*

Jakmile je Mono úspěšně nainstalováno, lze program spustit pouhým poklepáním, případně přes terminál:

mono */cesta/název_programu.exe*

Pokud se vaše aplikace nespustí, vypíše se příslušná chyba v příkazovém řádku. V opačném případě můžeme přesměrovat výpis chyby do textového souboru pomocí příkazu:

mono */cesta/název_programu.exe* 2> *logfile.log*

5.3.2. OVĚŘENÍ PROGRAMŮ

5.3.2.1. WinForms

- **Randomizer –** funkční

Zvládá parsovat pouze desetinnou tečku. Desetinná čárka je vynechána, tedy např. číslo 1,5 je chápáno jako 15.

- **Graphs** – funkční

Umí ukládat formáty BMP, JPG, GIF a PNG, ale pro nativní prohlížeč funguje jen PNG. S diakritikou v názvu ani s vykreslováním přímých i zaoblených čar není žádný problém.

- **Viewer** – funkční

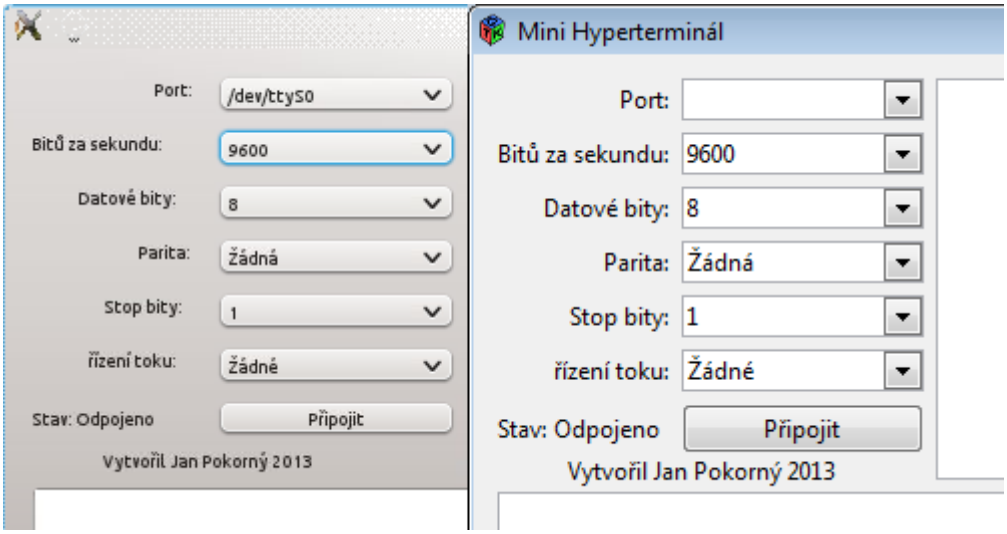
Prohlížeč dokáže zobrazovat formáty BMP, JPG, GIF a PNG. Fungují i soubory s diakritikou a mezerami v názvu.

- **SerialPort** – funkční

Jelikož program běží vzdáleně na serveru, nelze ověřit funkčnost odesílání a přijímání znaků. Porty COM, zde na Linuxu označované jako /dev/ttyS, program nalezne v pořádku.

5.3.2.2. GTK#

Problém všech GTK aplikací pod Kubuntu je rozhození pozic popisků. Jak ukazuje následující obrázek, vlevo je aplikace spuštěna v Kubuntu, vpravo pod systémem Windows.



Obr. 12: Chybné zobrazení – Kubuntu

- **Randomizer** – funkční

Zvládá parsovat pouze desetinnou tečku. Desetinná čárka je vynechána. Tedy např. číslo 1,5 je chápáno jako 15.

- **Graphs** – funkční (částečně)

Největším problémem je zde ignorování událostí myši. Po stisknutí jakéhokoli tlačítka se událost nevyvolá, a proto nelze na plochu kreslit. Ruční zadávání funguje dobře. Ukládat je možné pouze formáty PNG. Nemá vlastní funkci pro vykreslování zaoblených čar. Dále zde nefunguje příkaz pro nastavení bíle plochy. Plocha se zobrazí černě, a proto je v programu příkaz zakomentován. To způsobí, že plocha barevně splyne s pozadím.

Nefunkční kód:

```
this.pctGraph.GdkWindow.Background = new Color(255, 255, 255);
```

- **Viewer** – funkční

Prohlížeč dokáže zobrazovat formáty BMP, JPG, GIF a PNG. Fungují i soubory s diakritikou a mezerami v názvu.

- **SerialPort** – funkční

Protože program běží vzdáleně na serveru, nelze ověřit funkčnost odesílání a přijímání znaků. Porty COM, zde na Linuxu označované jako /dev/ttyS, program nalezne v pořádku.

5.3.2.3. Qt4DotNet

Přes všechnu snahu se nepodařilo Qt pod Kubuntu zprovoznit. Problém nastává nejspíš někde v nastavení cest k nativním knihovnám nebo přímo v oněch knihovnách. Program je vždy ukončen s touto chybou:

```
Unhandled Exception: System.TypeInitializationException: An exception was thrown by the type
initializer for com.trolltech.qt.gui.QApplication ---> System.TypeInitializationException: An exception
was thrown by the type initializer for com.trolltech.qt.core.QCoreApplication --->
System.TypeInitializationException: An exception was thrown by the type initializer for
com.trolltech.qt.core.QObject ---> System.TypeInitializationException: An exception was thrown by
the type initializer for com.trolltech.qt.QtJambiObject ---> System.TypeInitializationException: An
exception was thrown by the type initializer for com.trolltech.qt.QtJambi_LibraryInitializer --->
java.lang.RuntimeException: Loading library failed, progress so far:
    No 'qtjambi-deployment.xml' found in classpath, loading libraries via 'java.library.path'
    Loading library: 'libQtCore.so.4'...
     - using 'java.library.path'
```

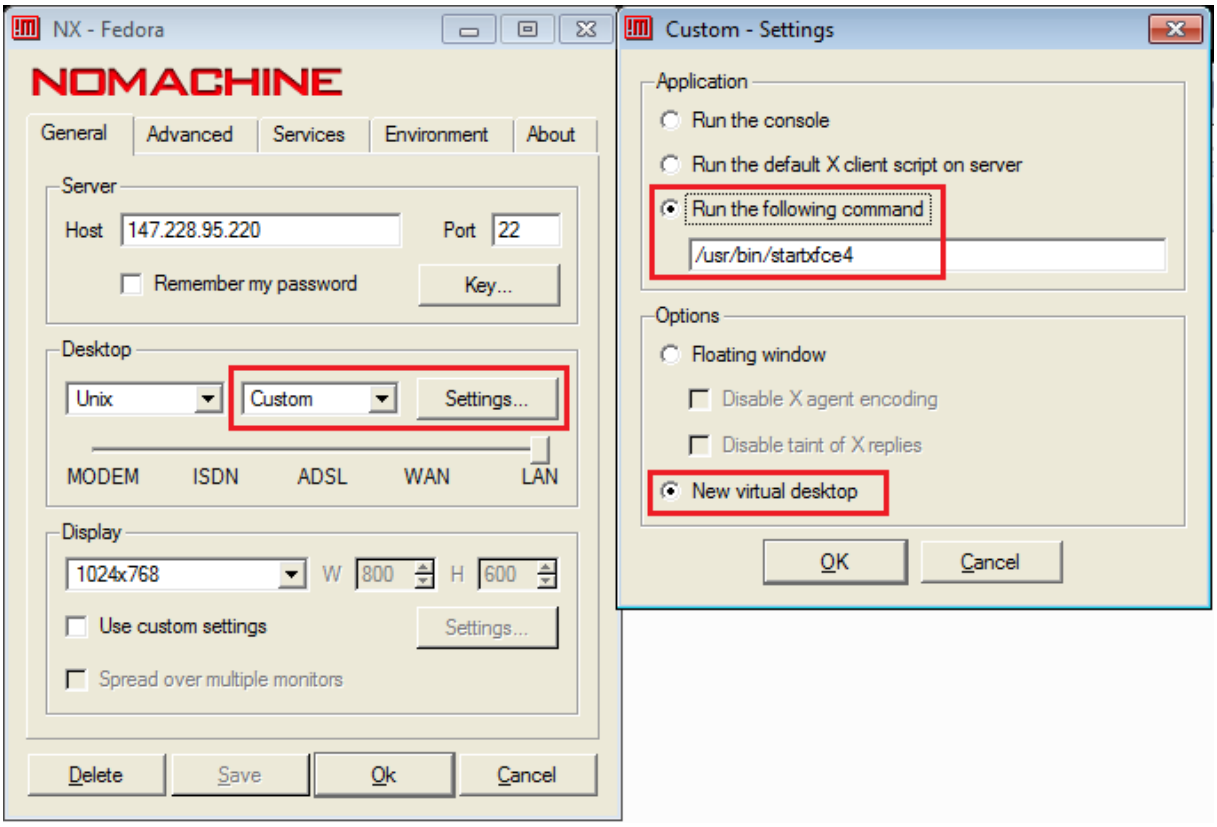
Tuto chybu neobjasnily ani různé návody na internetových fórech či stránkách. Nepomohlo ani ruční nastavení cesty ke knihovnám pomocí příkazu:

export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/*cesta*/lib

- **Randomizer** – nefunkční
- **Graphs** – nefunkční
- **Viewer** – nefunkční
- **SerialPort** – nefunkční

5.4. FEDORA – MONO

K Fedoře se využíval rovněž program NoMachine. Implicitně bylo voleno pro Fedoru prostředí GNOME, ovšem podařilo se zprovoznit prostředí o něco lepší a rychlejší. Postup je zobrazen červeně na následujícím obrázku (Obr. 13). Příkaz je: /usr/bin/startxfce4



Obr. 13: Nastavení pro Fedoru

5.4.1. POSTUP PRO SPUŠTĚNÍ

Pro spuštění .NET aplikací je potřeba nainstalovat Mono. Nejsnadnější je i zde spustit Terminál (konzoli) a napsat příkaz pro stažení a nainstalování balíku mono:

sudo yum -y install monodevelop* mono-*

V případě, že instalace nenalezne online cesty k souborům, je nutno provést update systému přes:

sudo yum update

Ověření správnosti nainstalování Mona se provede příkazem:

mono --version

Pokud vše proběhlo v pořádku, mělo by se ukázat:

Mono JIT compiler version *verze*

Jakmile je Mono úspěšně nainstalováno, lze program spustit pouhým poklepáním, případně přes terminál:

mono */cesta/název_programu.exe*

Pokud se vaše aplikace nespustí, vypíše se příslušná chyba v příkazovém řádku. V opačném případě můžeme přesměrovat výpis chyby do textového souboru pomocí příkazu:

mono ***/cesta/název_programu.exe*** **2>** ***logfile.log***

5.4.2. OVĚŘENÍ PROGRAMŮ

5.4.2.1. WinForms

- **Randomizer –** funkční

Zvládá parsovat pouze desetinnou tečku. Desetinná čárka je vynechána, tedy např. číslo 1,5 je chápáno jako 15.

- **Graphs** – funkční

Umí ukládat formáty BMP, JPG, GIF a PNG, ale pro nativní prohlížeč funguje jen PNG. S diakritikou v názvu ani s vykreslováním přímých i zaoblených čar není žádný problém.

- **Viewer** – funkční

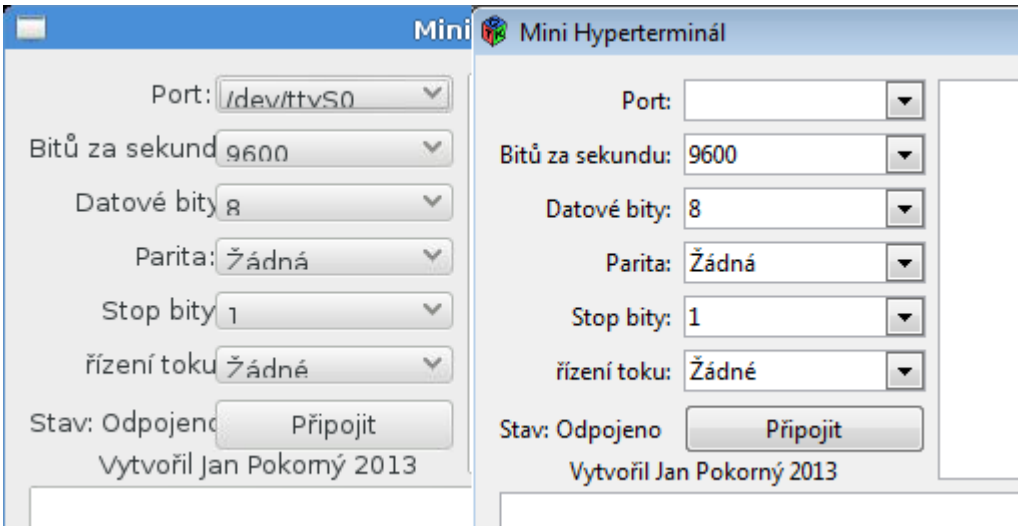
Prohlížeč dokáže zobrazovat formáty BMP, JPG, GIF a PNG. Fungují i soubory s diakritikou a mezerami v názvu.

- **SerialPort** – funkční

Protože program běží vzdáleně na serveru, nelze ověřit funkčnost odesílání a přijímání znaků. Porty COM, zde na Linuxu označované jako /dev/ttyS, program nalezne v pořádku.

5.4.2.2. GTK#

Problém všech aplikací GTK pod Fedorou je rozhození textu a fontů. Jak ukazuje následující obrázek, vlevo vidíme aplikaci na Fedoře, vpravo na systému Windows.



Obr. 14: Chybné zobrazení – Fedora

- **Randomizer –** funkční

Zvládá parsovat pouze desetinnou tečku. Desetinná čárka je vynechána. Tedy např. číslo 1,5 je chápáno jako 15.

- **Graphs** – funkční (částečně)

Největším problémem je zde ignorování událostí myši. Po stisknutí jakéhokoli tlačítka se událost nevyvolá, a proto nelze na plochu kreslit. Ruční zadávání funguje dobře. Ukládání je možné pouze formáty PNG. Nemá vlastní funkci pro vykreslování zaoblených čar. Dále zde nefunguje příkaz pro nastavení bíle plochy. Plocha se zobrazí černě, a proto je v programu příkaz zakomentován. To způsobí, že plocha barevně splyne s pozadím.

Nefunkční kód:

```
this.pctGraph.GdkWindow.Background = new Color(255, 255, 255);
```

- **Viewer** – funkční

Prohlížeč dokáže zobrazovat formáty BMP, JPG, GIF a PNG. Fungují i soubory s diakritikou a mezerami v názvu.

- **SerialPort** – funkční

Protože program běží vzdáleně na serveru, nelze ověřit funkčnost odesílání a přijímání znaků. Porty COM, zde na Linuxu označované jako /dev/ttyS, program nalezne v pořádku.

5.4.2.3. Qt4DotNet

Přes všechnu snahu se nepodařilo Qt ani pod Fedorou zprovoznit. Problém nastává nejspíš někde v nastavení cest k nativním knihovnám nebo přímo v oněch knihovnách. Program je vždy ukončen s touto chybou:

```
Unhandled Exception: System.TypeInitializationException: An exception was thrown by the type
initializer for com.trolltech.qt.gui.QApplication ---> System.TypeInitializationException: An exception
was thrown by the type initializer for com.trolltech.qt.core.QCoreApplication --->
System.TypeInitializationException: An exception was thrown by the type initializer for
com.trolltech.qt.core.QObject ---> System.TypeInitializationException: An exception was thrown by
the type initializer for com.trolltech.qt.QtJambiObject ---> System.TypeInitializationException: An
exception was thrown by the type initializer for com.trolltech.qt.QtJambi_LibraryInitializer --->
java.lang.RuntimeException: Loading library failed, progress so far:
    No 'qtjambi-deployment.xml' found in classpath, loading libraries via 'java.library.path'
    Loading library: 'libQtCore.so.4'...
     - using 'java.library.path'
```

Tuto chybu neobjasnily ani různé návody na internetových fórech či stránkách. Nepomohlo ani ruční nastavení cesty ke knihovnám pomocí příkazu:

export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/*cesta*/lib

- **Randomizer** – nefunkční
- **Graphs** – nefunkční
- **Viewer** – nefunkční
- **SerialPort** – nefunkční

5.5. RASPBIAN (HARD FLOAT) – MONO

5.5.1. POSTUP PRO SPUŠTĚNÍ

Pro spuštění .NET aplikací je potřeba nainstalovat Mono. Nejsnadnější je zde spustit Terminál (konzoli) a napsat příkaz pro stažení a nainstalování balíku mono-complete:

sudo apt-get install mono-complete

V případě, že instalace nenalezne online cesty k souborům, je nutno provést update systému přes příkaz:

sudo apt-get update

Ověření správnosti nainstalování Mona se provede příkazem:

mono --version

Pokud vše proběhlo v pořádku, mělo by se ukázat:

Mono JIT compiler version *verze*

Jakmile je Mono úspěšně nainstalováno, lze program spustit pouhým poklepáním, případně přes terminál:

mono /*cesta/název_programu.exe*

Pokud se vaše aplikace nespustí, vypíše se příslušná chyba v příkazovém řádku. V opačném případě můžeme přesměrovat výpis chyby do textového souboru pomocí příkazu:

mono /*cesta/název_programu.exe* 2> *logfile.log*

5.5.2. OVĚŘENÍ PROGRAMŮ

5.5.2.1. WinForms

Žádná WinForms aplikace nefunguje. Aplikace vyhazují tuto chybu:

```
System.Drawing.Font.CreateFont
(string,single,System.Drawing.FontStyle,System.Drawing.GraphicsUnit,byte,bool) <0x00147>
  at System.Drawing.Font..ctor
(string,single,System.Drawing.FontStyle,System.Drawing.GraphicsUnit,byte,bool) <0x0007f>
  at System.Drawing.Font..ctor (string,single,string) <0x00057>
  at (wrapper remoting-invoke-with-check) System.Drawing.Font..ctor (string,single,string) <0xffffffff>
```

Podle RaspberryPi fora ([20]) je tato chyba způsobena tzv. HardFloat Raspbianem, se kterým si Mono neumí korektně poradit. Problém byl vyřešen nainstalováním SoftFloat Raspbianu, viz dále.

- **Randomizer** – nefunkční
- **Graphs** – nefunkční
- **Viewer** – nefunkční
- **SerialPort** – nefunkční

5.5.2.2. GTK#

- **Randomizer –** funkční (částečně)

Vůbec neumí parsovat desetinná čísla.

- **Graphs** – funkční (částečně)

Největším problémem je zde ignorování událostí myši. Po stisknutí jakéhokoli tlačítka se událost nevyvolá, a proto nelze na plochu kreslit. Ruční zadávání funguje dobře.

Ukládat je možné pouze formáty PNG. Nemá vlastní funkci pro vykreslování zaoblených čar. Dále zde nefunguje příkaz pro nastavení bíle plochy. Plocha se zobrazí černě, a proto je v programu příkaz zakomentován. To způsobí, že plocha barevně splyne s pozadím.

Nefunkční kód:

```
this.pctGraph.GdkWindow.Background = new Color(255, 255, 255);
```

- **Viewer** – funkční

Prohlížeč dokáže zobrazovat formáty BMP, JPG, GIF a PNG. Fungují i soubory s diakritikou a mezerami v názvu.

- **SerialPort** – funkční (částečně)

Program nepřijímá znaky. Nastavení i odesílání je v pořádku.

5.5.2.3. Qt4DotNet

Přes všechnu snahu se nepodařilo Qt pod Raspbianem zprovoznit. Problém nastává nejspíš někde v nastavení cest k nativním knihovnám nebo přímo v oněch knihovnách. Program je vždy ukončen s touto chybou:

```
Unhandled Exception: System.TypeInitializationException: An exception was thrown by the type
initializer for com.trolltech.qt.gui.QApplication ---> System.TypeInitializationException: An exception
was thrown by the type initializer for com.trolltech.qt.core.QCoreApplication --->
System.TypeInitializationException: An exception was thrown by the type initializer for
com.trolltech.qt.core.QObject ---> System.TypeInitializationException: An exception was thrown by
the type initializer for com.trolltech.qt.QtJambiObject ---> System.TypeInitializationException: An
exception was thrown by the type initializer for com.trolltech.qt.QtJambi_LibraryInitializer --->
java.lang.RuntimeException: Loading library failed, progress so far:
    No 'qtjambi-deployment.xml' found in classpath, loading libraries via 'java.library.path'
    Loading library: 'libQtCore.so.4'...
     - using 'java.library.path'
```

Tuto chybu neobjasnily ani různé návody na internetových fórech či stránkách. Nepomohlo ani ruční nastavení cesty ke knihovnám pomocí příkazu:

export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/*cesta*/lib

- **Randomizer** – nefunkční
- **Graphs** – nefunkční
- **Viewer** – nefunkční
- **SerialPort** – nefunkční

5.6. RASPBIAN (SOFT FLOAT) – MONO

5.6.1. POSTUP PRO SPUŠTĚNÍ

Pro spuštění .NET aplikací je potřeba nainstalovat Mono. Nejsnadnější je zde spustit Terminál (konzoli) a napsat příkaz pro stažení a nainstalování balíku mono-complete:

sudo apt-get install mono-complete

V případě, že instalace nenalezne online cesty k souborům, je nutno provést update systému přes:

sudo apt-get update

Ověření správnosti nainstalování Mona se provede příkazem:

mono --version

Pokud vše proběhlo v pořádku, mělo by se ukázat:

Mono JIT compiler version *verze*

Jakmile je Mono úspěšně nainstalováno, lze program spustit pouhým poklepáním, případně přes terminál:

mono */cesta/název_programu.exe*

Pokud se vaše aplikace nespustí, vypíše se příslušná chyba v příkazovém řádku. V opačném případě můžeme přesměrovat výpis chyby do textového souboru pomocí příkazu:

mono */cesta/název_programu.exe* 2> *logfile.log*

5.6.2. OVĚŘENÍ PROGRAMŮ

5.6.2.1. WinForms

- **Randomizer –** funkční

Zvládá parsovat pouze desetinnou tečku. Desetinná čárka je vynechána, tedy např. číslo 1,5 je chápáno jako 15.

- **Graphs** – funkční

Umí ukládat formáty BMP, JPG, GIF a PNG, ale pro nativní prohlížeč funguje jen JPG a PNG. S diakritikou v názvu ani s vykreslováním přímých i zaoblených čar není žádný problém.

- **Viewer** – funkční

Prohlížeč dokáže zobrazovat formáty BMP, JPG, GIF a PNG. Fungují i soubory s diakritikou a mezerami v názvu.

- **SerialPort** – funkční (částečně)

Program nepřijímá znaky. Porty COM, zde na Linuxu označované jako /dev/tty, program nalezne v pořádku. Ostatní nastavení i odesílání je v pořádku.

5.6.2.2. GTK#

- **Randomizer –** funkční

Zvládá parsovat pouze desetinnou tečku. Desetinná čárka je vynechána. Tedy např. číslo 1,5 je chápáno jako 15.

- **Graphs** – funkční (částečně)

Největším problémem je zde ignorování událostí myši. Po stisknutí jakéhokoli tlačítka se událost nevyvolá, a proto nelze na plochu kreslit. Ruční zadávání funguje dobře. Ukládání je možné pouze formáty PNG. Nemá vlastní funkci pro vykreslování zaoblených čar. Dále zde nefunguje příkaz pro nastavení bíle plochy. Plocha se zobrazí černě, a proto je v programu příkaz zakomentován. To způsobí, že plocha barevně splyne s pozadím.

Nefunkční kód:

```
this.pctGraph.GdkWindow.Background = new Color(255, 255, 255);
```

- **Viewer** – funkční

Prohlížeč dokáže zobrazovat formáty BMP, JPG, GIF a PNG. Fungují i soubory s diakritikou a mezerami v názvu.

- **SerialPort** – funkční (částečně)

Program nepřijímá znaky. Nastavení i odesílání je v pořádku.

5.6.2.3. Qt4DotNet

Přes všechnu snahu se nepodařilo Qt pod Raspbianem zprovoznit. Problém nastává nejspíš někde v nastavení cest k nativním knihovnám nebo přímo v oněch knihovnách. Program je vždy ukončen s touto chybou:

```
Unhandled Exception: System.TypeInitializationException: An exception was thrown by the type
initializer for com.trolltech.qt.gui.QApplication ---> System.TypeInitializationException: An exception
was thrown by the type initializer for com.trolltech.qt.core.QCoreApplication --->
System.TypeInitializationException: An exception was thrown by the type initializer for
com.trolltech.qt.core.QObject ---> System.TypeInitializationException: An exception was thrown by
the type initializer for com.trolltech.qt.QtJambiObject ---> System.TypeInitializationException: An
exception was thrown by the type initializer for com.trolltech.qt.QtJambi_LibraryInitializer --->
java.lang.RuntimeException: Loading library failed, progress so far:
    No 'qtjambi-deployment.xml' found in classpath, loading libraries via 'java.library.path'
    Loading library: 'libQtCore.so.4'...
     - using 'java.library.path'
```

Tuto chybu neobjasnily ani různé návody na internetových fórech či stránkách. Nepomohlo ani ruční nastavení cesty ke knihovnám pomocí příkazu:

export LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:/*cesta*/lib

- **Randomizer** – nefunkční
- **Graphs** – nefunkční
- **Viewer** – nefunkční
- **SerialPort** – nefunkční

6. ZÁVĚR

Mono projekt se ukázal být kvalitní alternativou .NET Frameworku. Jeho instalace na systémech Windows i Linux je jednoduchá a nevyžaduje žádné zvláštní závislosti. Spolu s Mono projektem je k dispozici i IDE MonoDevelop, které je vhodné pro vývoj aplikací pod Linuxem.

Pro vývoj aplikací pod systémem Windows jednoznačně vede Microsoft Visual Studio spolu s grafickou knihovnou WinForms. Ze všech testů vyplynulo, že tato knihovna spolehlivě funguje i pod všemi testovanými Linuxovými operačními systémy kromě HardFloat Raspbianu, kde však nefunkčnost byla způsobena chybou Mono projektu.

Druhou vhodnou alternativou je knihovna GTK#, která sice fungovala na všech systémech včetně HardFloat Raspbianu, ale s podstatnými chybami.

Naopak známá knihovna Qt se neosvědčila vůbec. Její instalace byla velmi složitá a s mnoha závislostmi (QtJambi, nativní Qt, Java), které se i přes všechnu snahu nepodařilo zprovoznit a správně nastavit.

Knihovna WPF stále není implementována, tudíž její využití pro systémy Linux nepřipadá v úvahu vůbec.

Ani v další knihovně GWEN.NET se nepodařilo žádnou aplikaci zprovoznit. Pokusné zkoušení však naznačovalo, že se nejedná o knihovnu založenou na hotovém API pro vývoj GUI, ale o nízkoúrovňovou knihovnu, kde je nutné napsat svůj vlastní renderer okenního GUI.

Následující tabulka (Tab. 2) ukazuje funkčnost / nefunkčnost jednotlivých knihoven na testovaných operačních systémech.

Tab. 2: Zhodnocení

	WinForms	***GTK#***	***Qt4DotNet***	***WPF***
Win 7 - .NET Framework	funguje	funguje	funguje	funguje
Win 7 - Mono	funguje	funguje	funguje	nefunguje
Kubuntu	funguje	funguje	nefunguje	nefunguje
Fedora	funguje	funguje	nefunguje	nefunguje
Raspbian (HardFloat)	nefunguje	funguje	nefunguje	nefunguje
Raspbian (SoftFloat)	funguje	funguje	nefunguje	nefunguje

7. CITOVANÁ LITERATURA

[1] „Framework," Wikipedie, 16. Březen 2013. [Online]. Available: http://cs.wikipedia.org/wiki/Framework. [Přístup získán 26. Březen 2013].

[2] „Vývojová platforma," Wikipedie, 31. Březen 2012. [Online]. Available: http://cs.wikipedia.org/wiki/Vývojová_platforma. [Přístup získán 26. Březen 2013].

[3] „.NET Framework," Wikipedie, 8. Březen 2013. [Online]. Available: http://cs.wikipedia.org/wiki/.NET. [Přístup získán 4. Duben 2013].

[4] ECMA, „Standard ECMA-335," Červen 2012. [Online]. Available: http://www.ecma-international.org/publications/standards/Ecma-335.htm. [Přístup získán 25. Duben 2013].

[5] Microsoft, „Architektura .NET," 2001. [Online]. Available: http://msdn.microsoft.com/cs-cz/dd727769.aspx. [Přístup získán 25. Duben 2013].

[6] M. Běhálek, „ADO.NET," Katedra informatiky | FEI | VŠB-TU Ostrava, 2007. [Online]. Available: http://www.cs.vsb.cz/behalek/vyuka/pcsharp/text/ch07s01.html. [Přístup získán 4. Duben 2013].

[7] Microsoft, „MSDN," Microsoft, 2013. [Online]. Available: http://msdn.microsoft.com/cs-cz/. [Přístup získán 4. Duben 2013].

[8] „Mono project," Xamarin, [Online]. Available: http://mono-project.com/What_is_Mono. [Přístup získán 26. Březen 2013].

[9] „Mono," Wikipedia, 10. Březen 2013. [Online]. Available: http://en.wikipedia.org/wiki/Mono_(software). [Přístup získán 4. Duben 2013].

[10] „MoMA," Mono, [Online]. Available: http://www.mono-project.com/MoMA. [Přístup získán 4. Duben 2013].

[11] „Popis," Mono, [Online]. Available: http://mono-project.com/MoMA_-_Issue_Descriptions. [Přístup získán 4. Duben 2013].

[12] „DotGNU,“ GNU, Prosinec 2012. [Online]. Available: http://www.dotgnu.org/. [Přístup získán 5. Duben 2013].

[13] „DotGNU,“ Wikipedia, 26. Únor 2013. [Online]. Available: http://en.wikipedia.org/wiki/DotGNU. [Přístup získán 5. Duben 2013].

[14] „Microsoft Visual Studio,“ Wikipedia, 14. Květen 2013. [Online]. Available: http://en.wikipedia.org/wiki/Microsoft_Visual_Studio. [Přístup získán 15. Květen 2013].

[15] „Microsoft Visual Studio,“ Wikipedie, 14. Duben 2013. [Online]. Available: http://cs.wikipedia.org/wiki/Microsoft_Visual_Studio. [Přístup získán 15. Květen 2013].

[16] X. a. M. komunita, „MonoDevelop,“ [Online]. Available: http://monodevelop.com/. [Přístup získán 16. Květen 2013].

[17] „#develop Features,“ ic#code, 2012. [Online]. Available: http://www.icsharpcode.net/opensource/sd/features.aspx. [Přístup získán 20. Květen 2013].

[18] „qt4dotnet,“ Google Project, 14. Července 2010. [Online]. Available: http://code.google.com/p/qt4dotnet/. [Přístup získán 12. Února 2013].

[19] G. Newman, „gwen-dotnet,“ Google Project, [Online]. Available: http://code.google.com/p/gwen-dotnet/. [Přístup získán 12. Únor 2013].

[20] „Raspbian vs Wheezy (beta) Mono support,“ RaspberryPi, 18. Červen 2012. [Online]. Available: http://www.raspberrypi.org/phpBB3/viewtopic.php?f=66&t=11634. [Přístup získán 16. Květen 2013].

OBRÁZKY

TABULKY

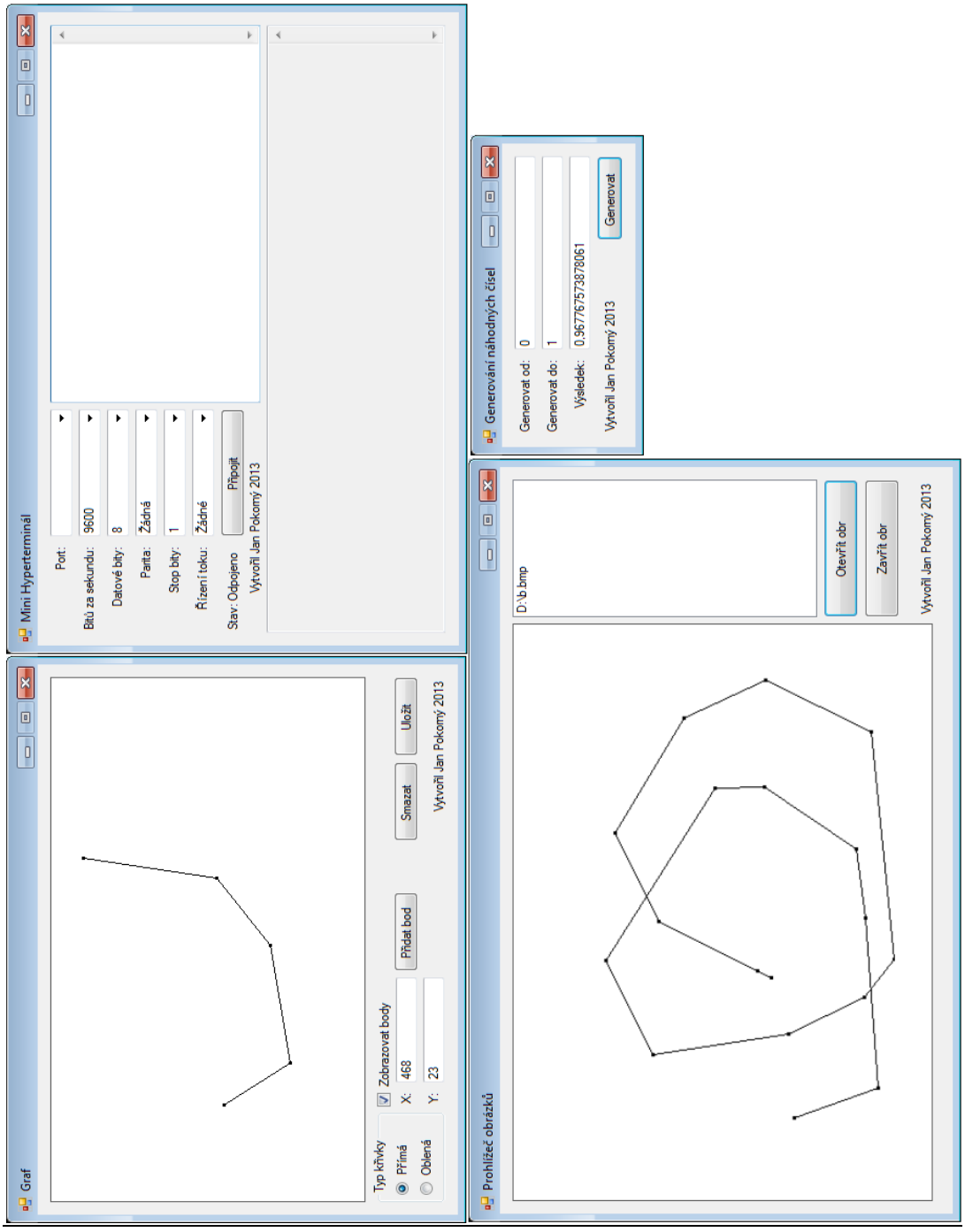
PŘÍLOHY

Přílohy na CD-ROM:

- **Elektronická verze práce**
- **Programy v jednotlivých knihovnách**
- **Obrázky pro srovnání grafického vzhledu**

Příloha A: Windows 7 – .NET Framework

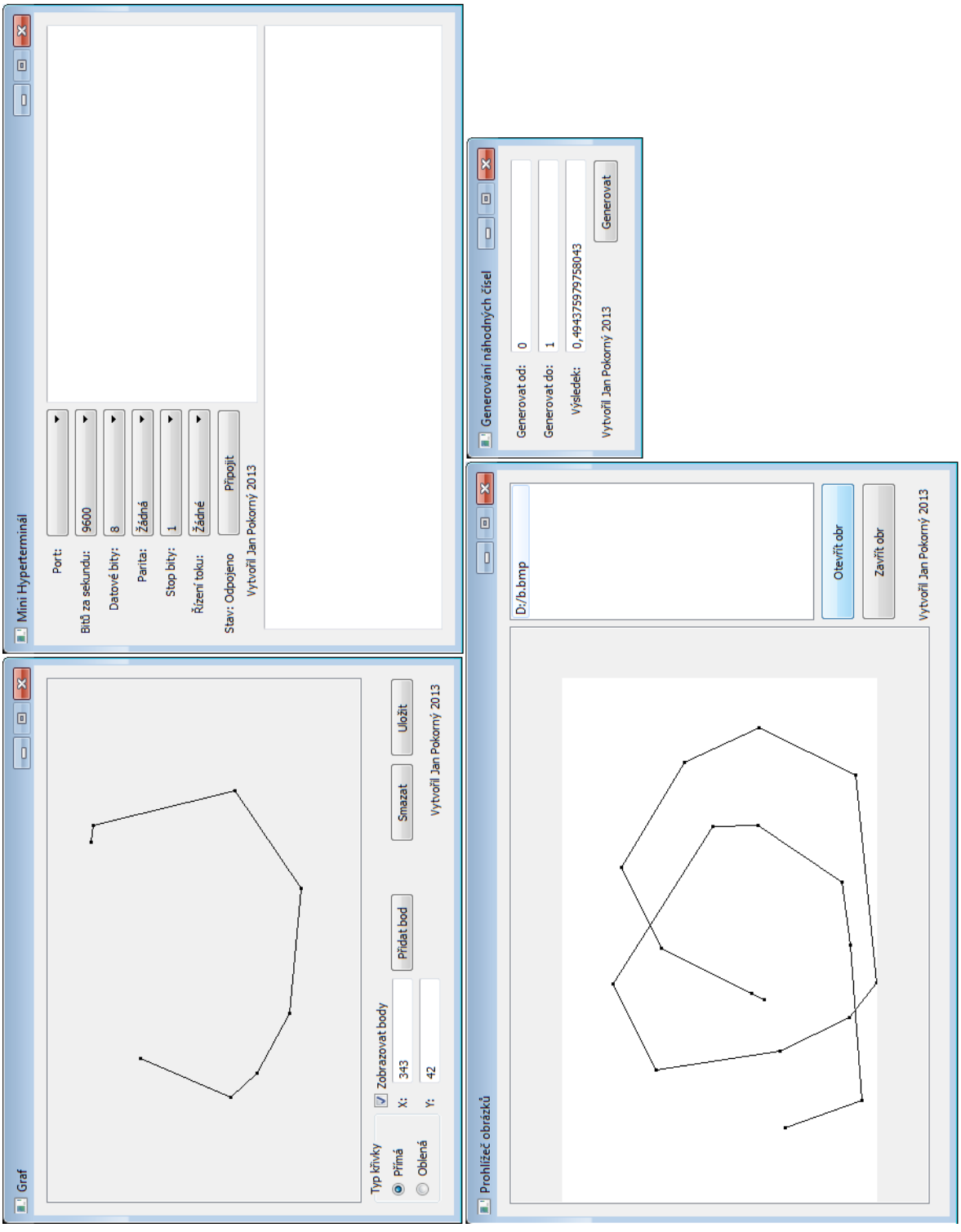
WinForms



GTK#

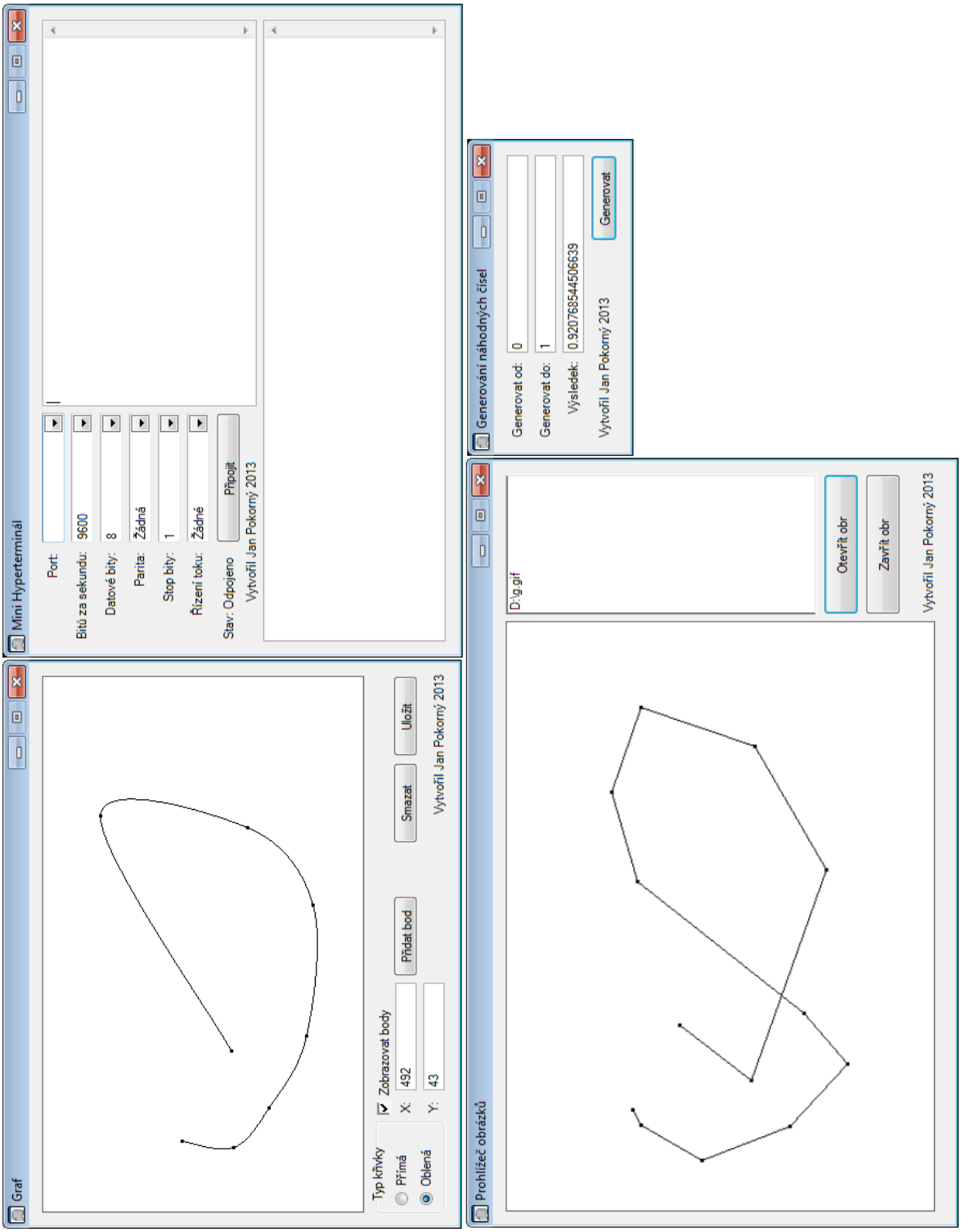

Mini Hyperterminál
Port:
Bitů za sekundu: 9600
Datové bity: 8
Parita: Žádná
Stop bity: 1
řízení toku: Žádné
Připojit
Stav: Odpojeno
Vytvořil Jan Pokorný 2013
Generátor náhodných čísel
Generovat od: 0
Generovat do: 1
Výsledek: 0,10477630938625 7
Generovat
Vytvořil Jan Pokorný 2013
MainWindow
Typ přímky
Přímá
Oblená
X: 327
Y: 89
Přidat bod
Zobrazovat body
Smazat
Vytvořil Jan Pokorný 2013
Uložit
MainWindow
Soubory
D:\p.png
Otevřít obr
Zavřít obr
Vytvořil Jan Pokorný 2013

Qt4DotNet

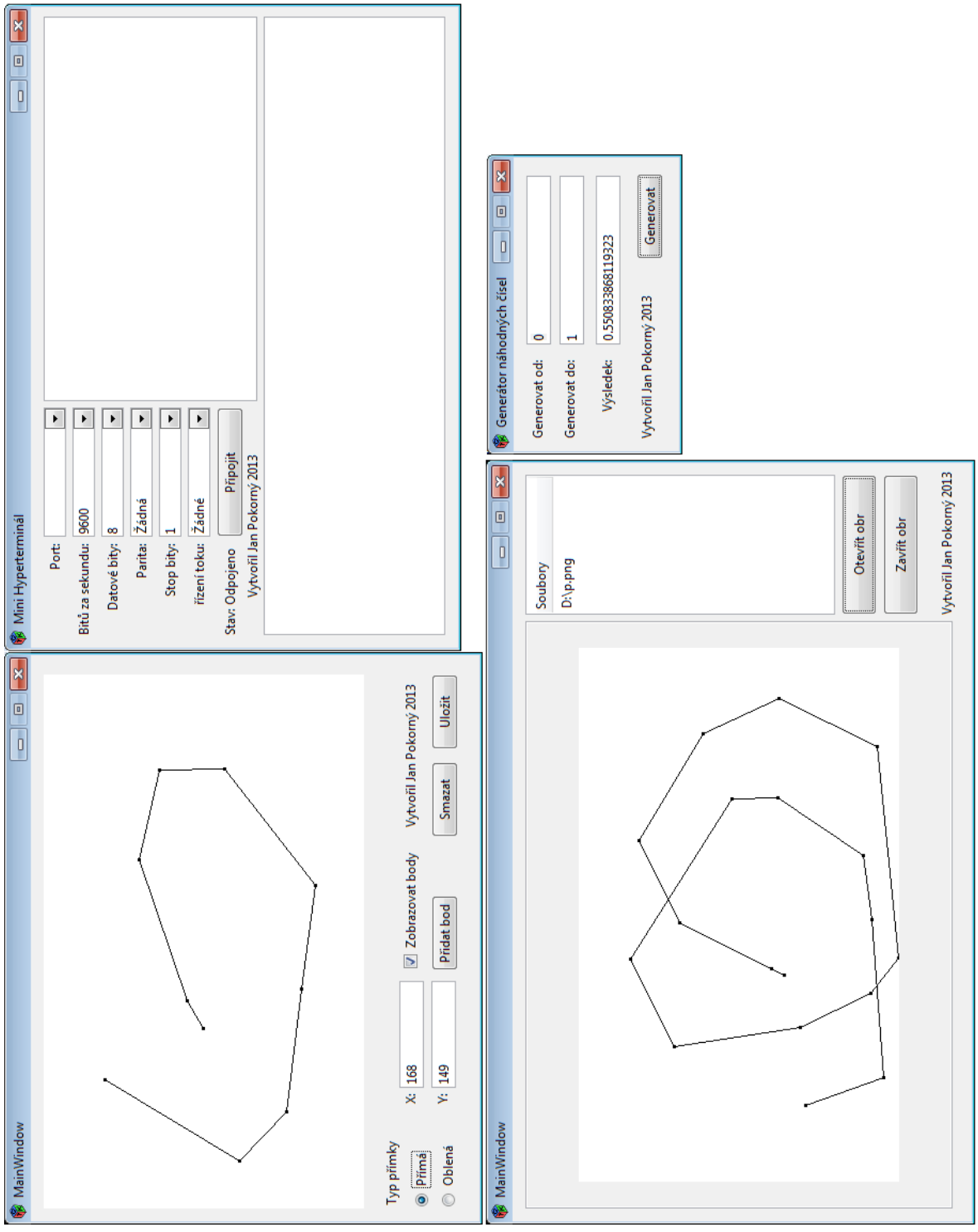

Mini Hyperterminal
Port:
Bitů za sekundu:
9600
Datové bity:
8
Parita:
Žádná
Stop bity:
1
Řízení toku:
Žádné
Připojit
Stav: Odpojeno
Vytvořil Jan Pokorný 2013
Generování náhodných čísel
Generovat od:
0
Generovat do:
1
Výsledek:
0,4943759797580 43
Generovat
Vytvořil Jan Pokorný 2013
Graf
Typ křivky
Přímá
Oblená
Zobrazovat body
X:
343
Y:
42
Přidat bod
Smazat
Uložit
Vytvořil Jan Pokorný 2013
Prohlížeč obrázků
D:/b.bmp
Otevřít obr
Zavřít obr
Vytvořil Jan Pokorný 2013

Příloha B: Windows 7 – Mono

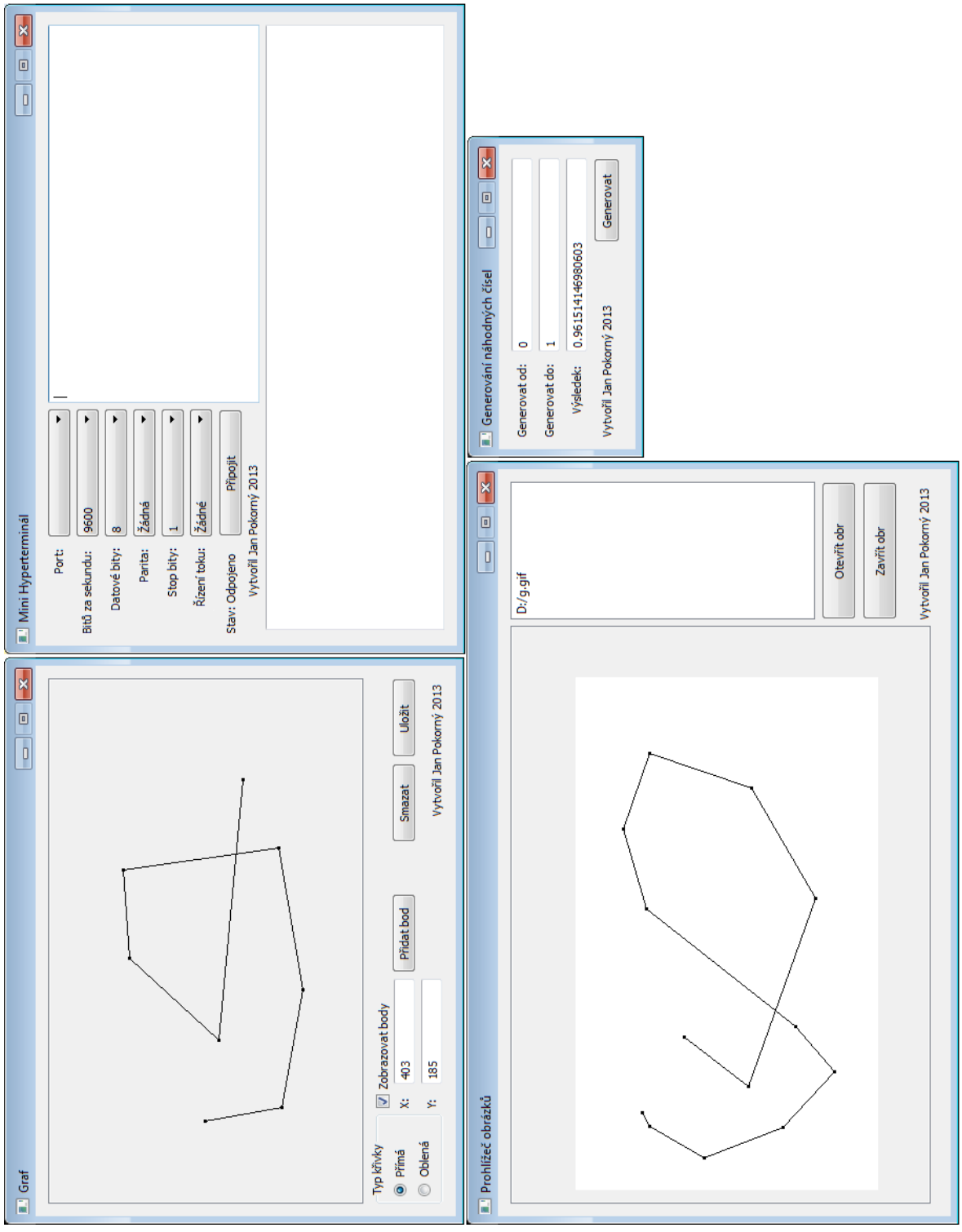
WinForms



GTK#

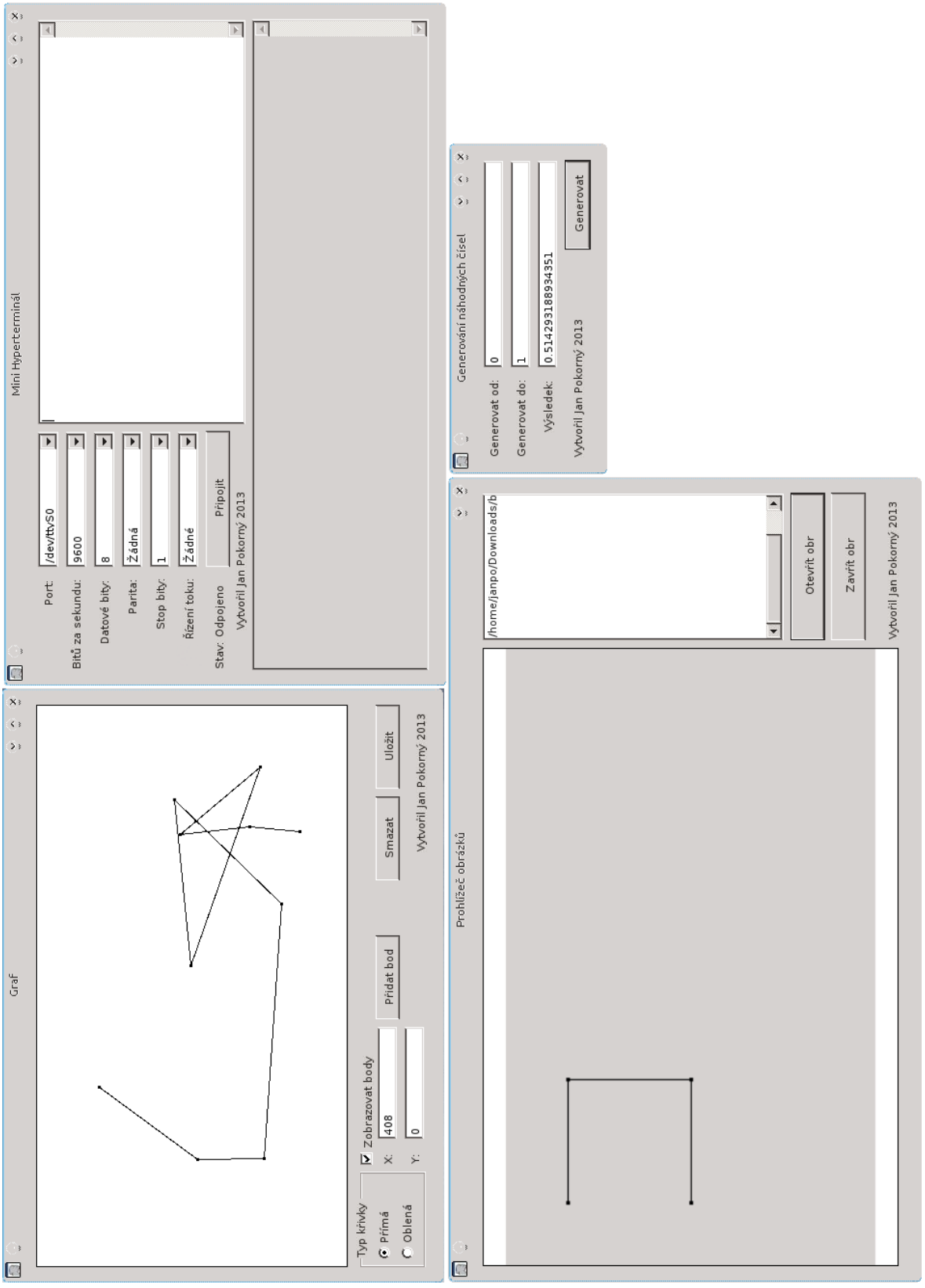

Mini Hyperterminál
Port:
Bitů za sekundu: 9600
Datové bity: 8
Parita: Žádná
Stop bity: 1
řízení toku: Žádné
Připojit
Stav: Odpojeno
Vytvořil Jan Pokorný 2013
MainWindow
Typ přímky
Přímá
Oblená
X: 168
Y: 149
Zobrazovat body
Přidat bod
Smazat
Uložit
Vytvořil Jan Pokorný 2013
Generátor náhodných čísel
Generovat od: 0
Generovat do: 1
Výsledek: 0.550833868119323
Vytvořil Jan Pokorný 2013
Generovat
MainWindow
Soubory
D:\p.png
Otevřít obr
Zavřít obr
Vytvořil Jan Pokorný 2013

Qt4DotNet

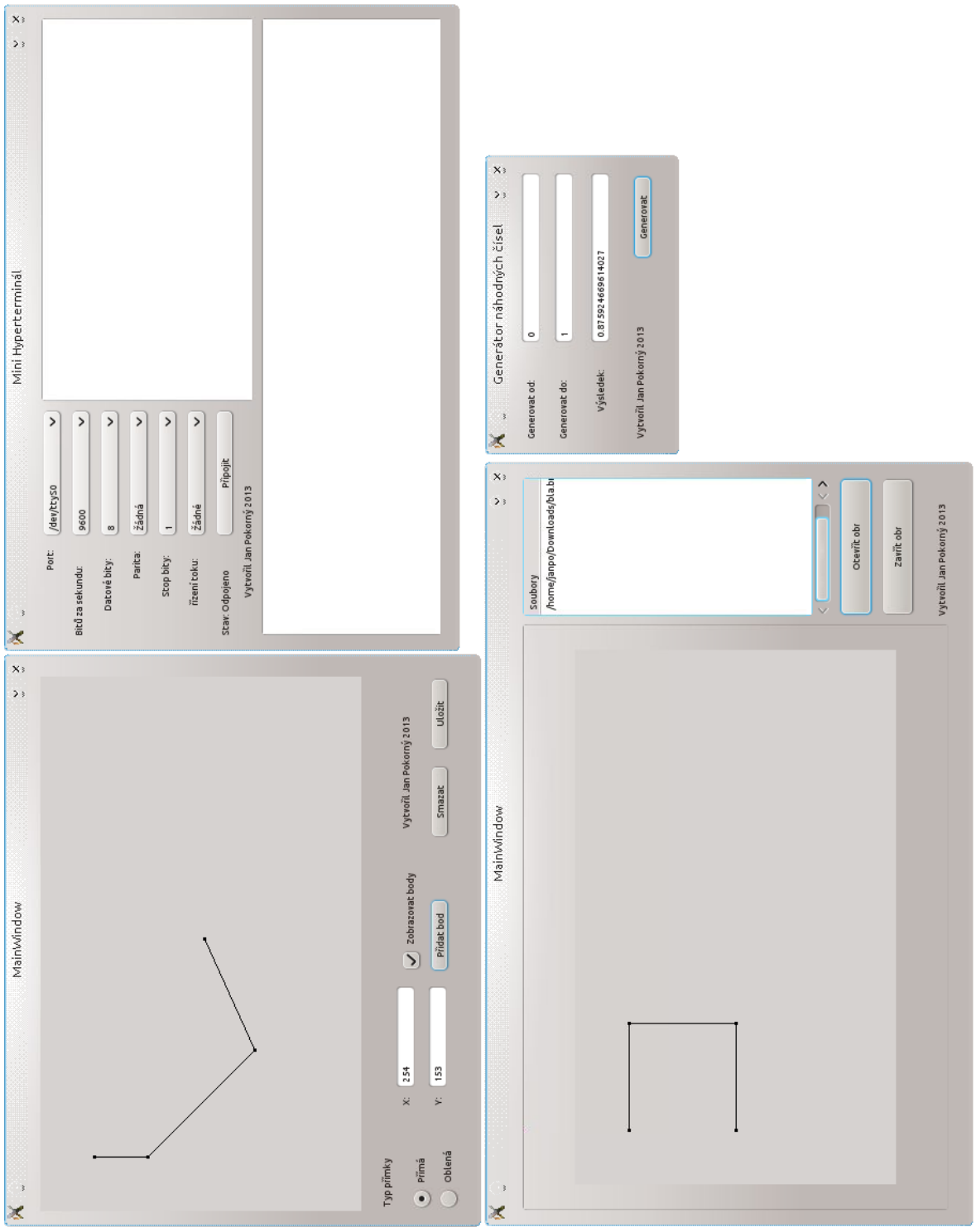

Mini Hyperterminál
Port:
Bitů za sekundu:
9600
Datové bity:
8
Parita:
Žádná
Stop bity:
1
Řízení toku:
Žádné
Připojit
Stav: Odpojeno
Vytvořil Jan Pokorný 2013
Generování náhodných čísel
Generovat od:
0
Generovat do:
1
Výsledek:
0.961514146980603
Generovat
Vytvořil Jan Pokorný 2013
Graf
Typ křivky
Přímá
Oblená
Zobrazovat body
X:
403
Y:
185
Přidat bod
Smazat
Uložit
Vytvořil Jan Pokorný 2013
Prohlížeč obrázků
D:/g.gif
Otevřít obr
Zavřít obr
Vytvořil Jan Pokorný 2013

Příloha C: Kubuntu – Mono

WinForms

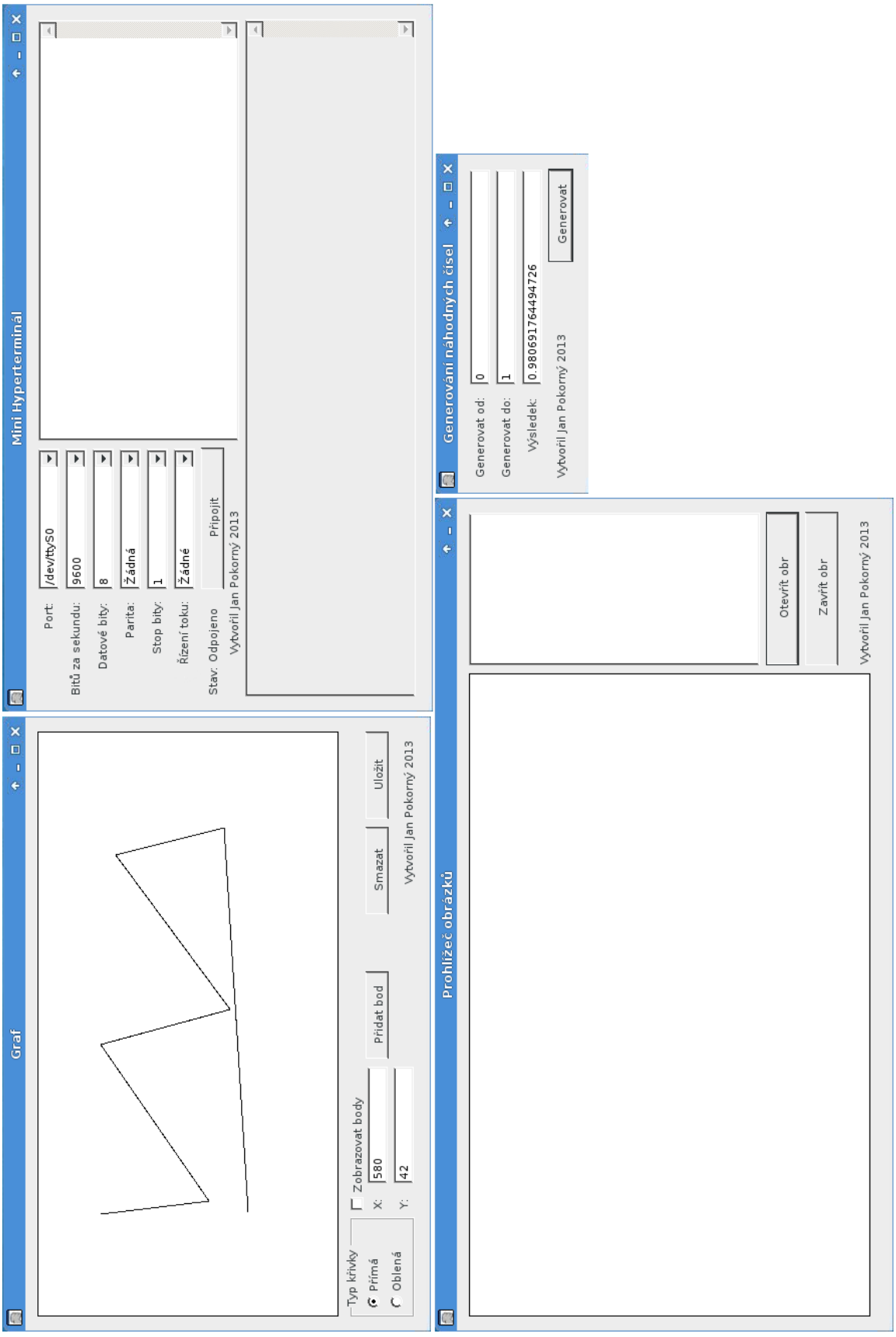


GTK#


Mini Hyperterminál
Port:
/dev/ttyS0
Bitů za sekundu:
9600
Datové bity:
8
Parita:
Žádná
Stop bity:
1
Řízení toku:
Žádné
Připojit
Stav: Odpojeno
Vytvořil Jan Pokorný 2013
Generátor náhodných čísel
Generovat od:
0
Generovat do:
1
Výsledek:
0.875924669614027
Generovat
Vytvořil Jan Pokorný 2013
MainWindow
Soubory
/home/janpo/Downloads/bla.bi
Otevřít obr
Zavřít obr
Vytvořil Jan Pokorný 2013
MainWindow
Typ přímky
Přímá
Oblená
X:
254
Y:
153
Zobrazovat body
Přidat bod
Smazat
Uložit
Vytvořil Jan Pokorný 2013

Příloha D: Fedora – Mono

WinForms

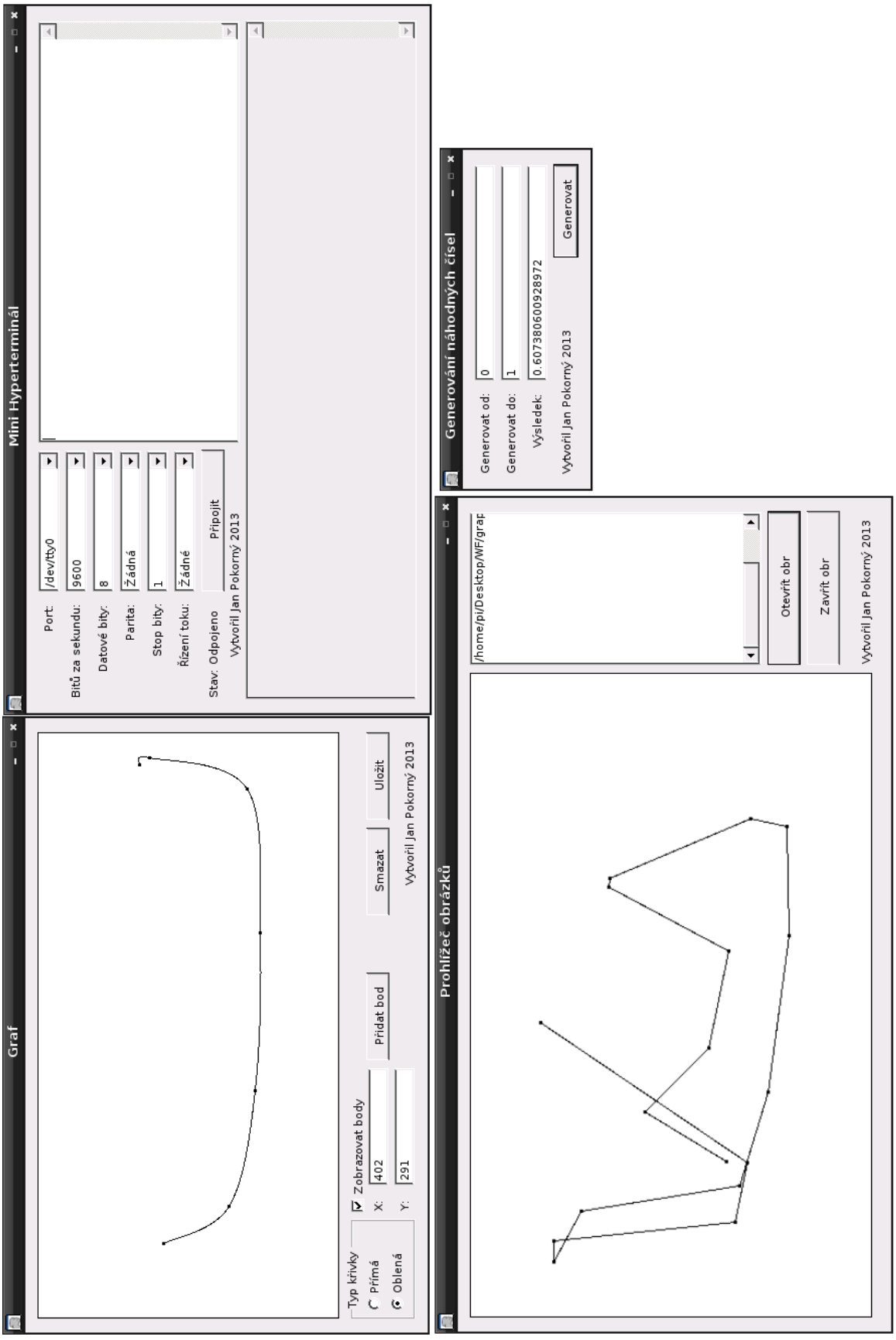


GTK#

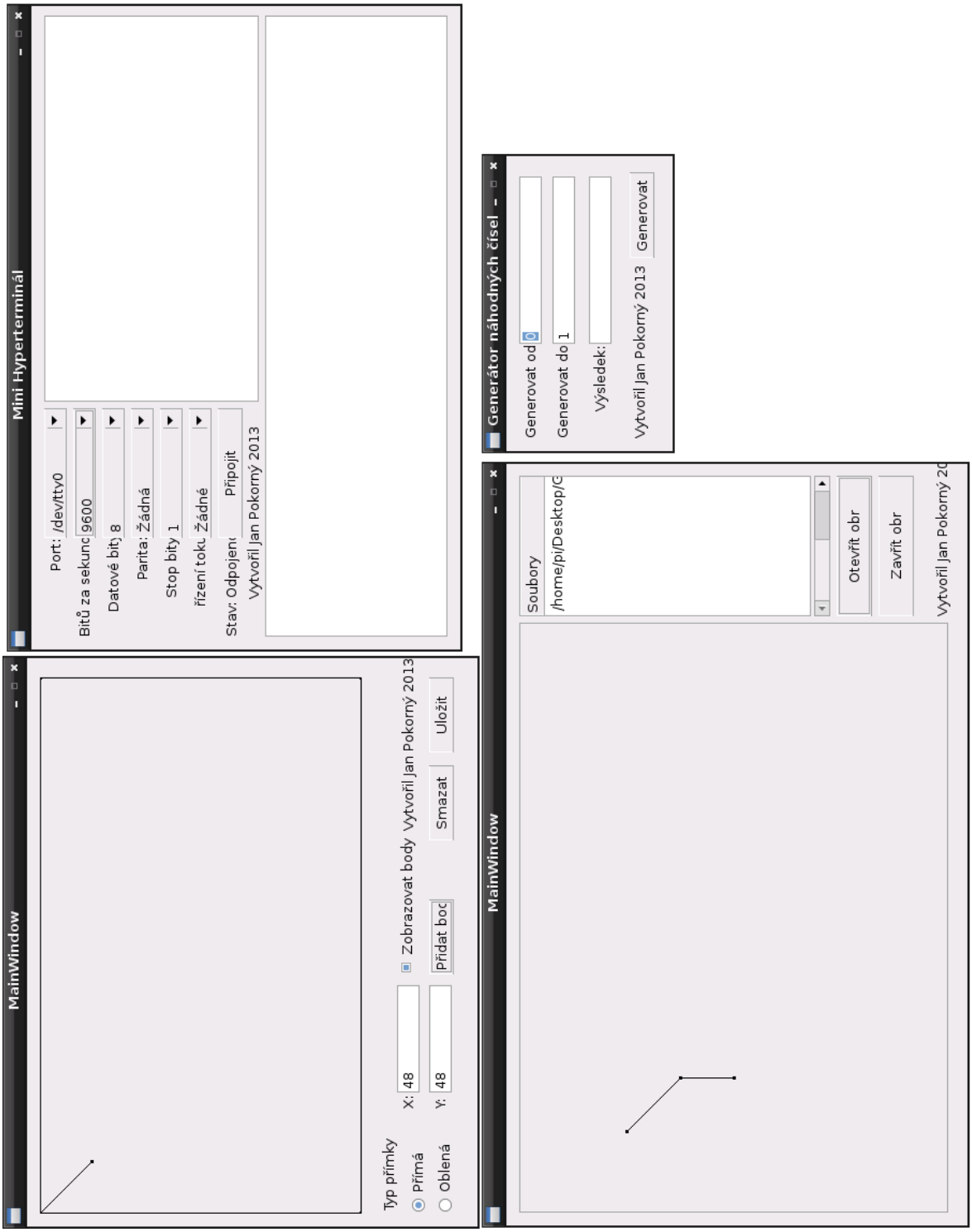

Mini Hyperterminál
Port: /dev/ttyS0
Bitů za sekund 9600
Datové bity 8
Parita: Žádná
Stop bity 1
řízení toku Žádné
Stav: Odpojenc
Připojit
Vytvořil Jan Pokorný 2013
Generátor náhodných čí
Generovat od 0
Generovat do 1
Výsledek: 0.480609773416356
Vytvořil Jan Pokorný 2013
Generovat
MainWindow
Typ přímky
Přímá
Oblená
X: 300
Y: 85
Přidat bo
Zobrazovat body
Vytvořil Jan Pokorný 2013
Smazat
Uložit
MainWindow
Soubory
Otevřít obr
Zavřít obr
Vytvořil Jan Pokorný 20

Příloha E: Raspbian (SoftFloat) – Mono

WinForms



GTK#


Mini Hyperterminál
Port: /dev/tty0
Bitů za sekund 9600
Datové bity 8
Parita: Žádná
Stop bity 1
řízení toku: Žádné
Stav: Odpojen
Připojit
Vytvořil Jan Pokorný 2013
Generátor náhodných čísel
Generovat od 0
Generovat do 1
Výsledek:
Vytvořil Jan Pokorný 2013
Generovat
MainWindow
Typ přímky
Přímá
Oblená
X: 48
Y: 48
Zobrazovat body
Vytvořil Jan Pokorný 2013
Přidat boc
Smazat
Uložit
MainWindow
Soubory
/home/pi/Desktop/G
Otevřít obr
Zavřít obr
Vytvořil Jan Pokorný 20