PŘÍRODOVĚDECKÁ FAKULTA UNIVERZITY PALACKÉHO
KATEDRA INFORMATIKY

BAKALÁŘSKÁ PRÁCE

Kreslení obrázků pro výuku matematiky

2013 Jiří Spáčil

Anotace

Práce popisuje vytvořenou aplikaci na kreslení matematických obrázků pro výuku matematiky. Zabývá se způsobem jejího efektivního využívání a přínosem pro uživatele. Představuje způsob implementace v C# a různé možnosti vykreslování grafiky pod platformou Windows. Popisuje způsob převodu obrázků do různých vektorových formátů. V poslední řadě určuje směr dalšího rozvoje této aplikace.

Děkuji svému vedoucímu práce Mgr. Janu Outratovi, Ph.D. za podnětné rady a vstřícnost.

Obsah

Seznam obrázků

1. Představení projektu

Vytvořil jsem interaktivní aplikaci umožnující jednoduší kreslení matematických obrázků, které by se používaly k výuce matematiky. Nakreslené obrázky se pomocí vestavěných funkcí vzájemně kombinují a tak vytvářejí složitější celky. Ty se pak dají přímo vkládat do učebnicových textů pomocí formátu SVG.

1.1. Rozbor zadání

Aplikace měla obsahovat šablony obrázků a ty pak vzájemně modifikovat. Nemělo by se jednat o automatizovanou tvorbu geometrických konstrukcí. Aplikace by ovšem měla zjednodušit a urychlit vlastní kreslení, co by byl i můj hlavní cíl. Další cíl bylo vytvořit uživatelsky přívětivé a jednoduché rozhraní. Z pohledu uživatele je právě uživatelské prostředí jedním z hlavních měřítek pro kladné hodnocení celého projektu.

Celá aplikace na tvorbu matematických obrázků by se dala rozvinout opravdu mnoha způsoby, člověka napadají hned celé desítky funkcí jak jednotlivé matematické obrázky definovat nebo možnosti jejich vzájemné modifikace. Vytvořil jsem základní prostředí pro libovolné aplikování funkcí pro tvorbu takových obrázků. Celá aplikace je navržena tak, aby se další funkce a komponenty daly do tohoto softwaru libovolně dodávat. Tím se vytvoří opravdu užitečný nástroj pro tvorbu a výuku matematiky a geometrie.

1.2. Inspirace

Inspiraci pro tvorbu aplikace jsem čerpal s grafických editorů jako například Corel Draw nebo Inkscape. Tyto grafické editory využívají na malování vektorovou grafiku, která se při tvorbě matematických obrázků snadno nabízí.

Částečně jsem se inspiroval i složitějšími aplikacemi jako CAD systémy (computer-aided drawing - počítačem podporované kreslení). Jednoduše lze vysvětlit pojem CAD i tak, že se jedná o používání grafického programu místo rýsovacího prkna. Samotné rýsování není vlastně nic jiného než tvorba matematických obrázků.

Pro příklad uvedu editor Autocad od firmy Autodesk. Od této firmy je to jeden s jejich nejstarších CAD editorů, ale stále mezi odbornou veřejností populární. Autocad používá pro rýsování jen dvojrozměrnou grafiku a tak simuluje rýsovací prkno. Jejich editor dopočítává rozměry čar, velikost úhlů, automaticky definuje některé pravidelné n-úhelníky atd. Jedná se pouze o generování strojírenských konstrukcí, ale způsob jejich tvorby mě hodně ovlivnil.

V dnešní době CAD editory od Autodesku plně simulují předměty reálného světa. Používají trojrozměrné rozhraní a počítají i s fyzickými vlastnostmi materiálů, ze kterých jsou objekty v jejich editorech definované.

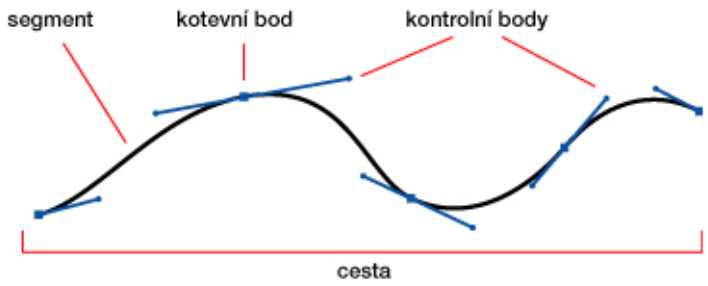
2. Řešení problému

V následujícím textu budu postupně rozebírat témata, která jsem uplatnil při tvorbě aplikace. Tyto témata mi pomohli vyřešit problém s vývojem aplikace a její vlastní implementací.

2.1. Vektorová grafika

Na vykreslování matematických obrázků v aplikaci jsem použil vektorovou grafiku.

Vektorová grafika popisuje obrázek pomocí vektorů. Obrázek je složen z křivek, které se spojují takzvanými kotevními body. Tyto křivky mohou mít barevnou výplň formou jednolité plochy nebo barevného přechodu (gradientu). Jednotlivé křivky mohou na sebe jakkoliv navazovat a jejich vzájemné spojení se definuje cestou od prvního kotevního bodu (startovacího) až k poslednímu. Tímto způsobem lze popsat i tu nejsložitější křivku jakou jsme schopni nakreslit.



Obrázek 1. Znázornění definice cesty vektorové grafiky

2.1.1. Výhody vektorové grafiky

Hlavní výhodou vektorové grafiky je možnost v podstatě libovolného zvětšování již vytvořeného obrázku a to bez sebemenší ztráty na kvalitě.

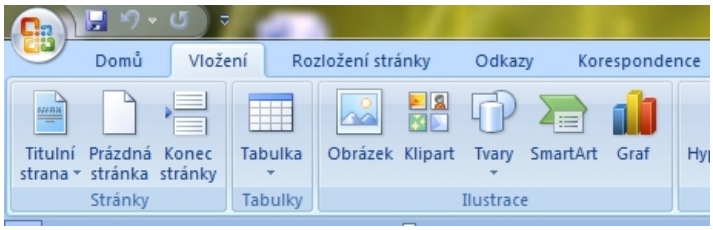
Zatímco u rastrových obrázků máme po zvětšení jen hromadu různobarevných čtverečků, vektorový obrázek se přepočítá a přizpůsobí dané velikosti. Ukládání vektorového obrázku na datový nosič nám zabere mnohem méně místa než bitmapový obrázek. Jak jsem zmínil v textu výše, ukládáme jen polohu kotevních bodů sloučených do takzvané cesty a typ křivek, který je předem definován jako například: linka, kruh, půlkruh, bézierova křivka atd. Formáty pracující s vektorovou grafikou jsou SVG (Scalable Vector Graphics) a PDF (Portable Document Format).

Vektorová grafika má i své nevýhody. Její použití je složitější, nakreslit nějaký obrázek vyžaduje zkušenost a jisté kombinační schopnosti. Pokud složitost grafického obrázku překročí určitou mez je použití vektorové grafiky i poměrně náročné na procesor, a její vykreslování bude pomalé.

2.2. Uživatelské rozhraní

2.2.1. Ribbon bar

Jako hlavní prvek uživatelského rozhraní jsem použil Ribbon Bar (pás karet). Ten má podobu řady přepínacích karet, přičemž každá karta v sobě sdružuje skupinu ikon vzájemně logicky souvisejících funkcí. Tyto karty se zpravidla rozprostírají horizontálně v horní části okna příslušné aplikace. Ribbon je Microsoftem patentovaný ovládací prvek uživatelského rozhraní pro jeho operační systémy Windows. Ribbon se objevuje v novějších verzích programů firmy Microsoft, zejména v kancelářském balíku Microsoft Office.



Obrázek 2. Ribbon bar z Microsoft Office

Ribbon byl poprvé uveden v Office 2007 a po jeho příznivém přijetí byl přidán i do dalších aplikací od Microsoftu – ve Windows 7 jsou tak Ribbonem vybaveny už i WordPad, Malování či Windows Live Movie Maker. V operačním systému Windows 8 se tento prvek nepřestává používat a je jím vybaven i Průzkumník Windows.

Používání Ribbonu v aplikacích mi připadá praktické a jeho ovládání intuitivní. Ovládání celé aplikace a logické rozmístění funkcí se tím stává lehce zapamatovatelné.

Použití Ribbonu pro tuto aplikaci je přínos hned z několika důvodů. Jednak pro libovolné rozšiřování celého projektu, stačí ho jen doplnit další kartou obsahující jiné funkce. Nebo ho může uživatel používat sám jako takový „šuplík“, kam si odloží vytvořené obrázky a pak je následně použít při kreslení složitějších celků.

2.2.2. Implementace Ribbonu

Originální verze pro vývojáře se platí. Existuje mnoho společností, které tento ovládací prvek nabízejí jako komponent přímo do Visual Studia, cena je zhruba 1200,- $ (např. Telerik). Jsou i volné verze, nejsou tolik graficky propracované a nemají všechny funkce originálního ribbonu, např. uživatelskou nápovědu, nebo možnost vložit Riboon přímo do záhlaví hlavního okna aplikace.

Ribbon, který jsem použil je jednou s volných verzí, bylo ho nutno částečně upravit a některé funkce dopsat.

2.2.3. Metafora rýsovací plochy

Jako další hlavní prvek uživatelského rozhraní jsem použil zobrazení papíru A4, který nám poslouží jako metafora pro kreslení na skutečný papír a dává uživateli vizuální přehled o skutečných rozměrech nakresleného obrázku. Rýsovací plocha aplikace má nekonečné okraje a umožňuje uživateli kreslit obrázky libovolné velikosti. Možnost zoomování papíru a různé režimy zobrazování je dnes v aplikacích tohoto typu standardní záležitost.

2.3. Vlastní implementace

K implementaci aplikace jsem použil C#. Je to jazyk vyvinutý společností Microsoft. C# je založen na jazycích C++ a Java. Je to moderní jazyk, jeho tvůrci kladly důraz na efektivní využívání času programátora, kde se nemusí starat o deklaraci paměti, hlídání hranic polí (kolekcí) a používá se zde automatický garbage collector. Všechno je zde udělané tak, aby to bylo přehledné, jednoduché, rychlé.

C# má i negativní stránky, je mu vytýkán všeobecný pomalý chod některých algoritmů a hlavně komerční stránka celé platformy[1].

Co se týče rychlosti běhu některých algoritmů se ve skutečnosti program psaný v C# nemůže měřit s programem v C++. Problém je zde přímo při překládání kódu. C# se kompiluje do bytecodu (oficiální označení CIL- Common Intermediate Language). Ten se nespouští přímo na procesoru, ale při spuštění je znovu kompilován z bytecodu do nativního kódu a až ten se spustí. Proto je zde menší zpoždění vůči nativnímu kódu přímo. Je zde ještě několik omezení. Jazyk samotný nepodporuje mnohonásobnou dědičnost, těžko se v C# kopírují objekty atd.

Microsoft každou novou verzí Visual Studia nabízí určité vylepšení C#. Tyto nové verze postupně odstraňují některá omezeni tohoto programovacího jazyka. Nyní již je k dispozici verze C# 5.0.

Hlavní důvod proč jsem si vybral C# je právě jeho jednoduché používání a maximální optimalizace času při tvorbě takto náročné aplikace. Důležitým faktorem je dostupnost různých publikací pro tuto platformu a podpora vývojáře na nejrůznějších webových stránkách nabízející konzultace.

2.3.1. Grafika v C#

Grafické rozhraní pro Windows zajišťuje GDI (graphical de vice interface). A novější GDI+, která obsahuje rozsáhlé grafické knihovny. S jejích pomocí jsme schopni vykreslovat vektorovou grafiku. Toto rozhraní používá na zobrazování grafiky okno, v terminologii .NET hovoříme o tzv. formuláři. Tento výraz označuje obdélníkový objekt, který zabírá oblast na obrazovce pro účely činnosti aplikace. Do tohoto formuláře můžeme jednoduše vkládat i spoustu předdefinovaných ovládacích prvků systému Windows. Tyto ovládací prvky většinou při tvorbě jednodušších aplikací zcela vývojářům postačí, jejich jednoduché vkládání a manipulace s nimi, poskytují vývojářům možnost opravdu rychlé tvorby aplikací.

[1]Všeobecně pomalý chod algoritmů se dá řešit efektivním psaním kódu. Například pokud chceme použit třídící algoritmus Quicksort použijeme jeho vnitřní implementaci přímo v překladači, tím se rychlost algoritmu mnohonásobně zvýší. Nakonec nejmodernější hry pro X-Box jsou právě psány v C#.

Bohužel, ve skutečnosti je vykreslování grafiky ve Windows Forms velice pomalé. Z důvodů překreslování celého formuláře metodou OnPaint(), kterou vývojář nemůže volat přímo, ale jen přes metodu Invalidat(), která posléze zajistí vlastní překreslení celého formuláře. Tato metoda a její vlákno, ve kterém se volá, má jen uživatelskou prioritu.

Operace kreslení patří v aplikacích GDI+ téměř vždy mezi nejnáročnější z hlediska zatížení procesoru. Když kreslení probíhá současně s jinými operacemi, bude tyto další operace zdržovat. Kdybyste v tomto příkladu přímo zavolali metodu pro kreslení z metody LoadFile(), pak by metoda LoadFi1e() nemohla vrátit řízení, dokud by nebyla kreslicí operace dokončena. V tomto intervalu by aplikace nemohla reagovat na jiné události. Jestliže však zavoláte metodu Invalidate(), jednoduše zajistíte, že systém Windows vyvolá událost OnPaint() těsně před návratem z metody LoadFi1e().

Toto funkční omezení a zatěžování procesoru je nepřípustné pro interaktivní aplikace, proto jsem zvolil grafiku napsanou ve WPF.

2.3.2. WPF

WPF (Windows Presentation Foundation) je jedno z předních rozšíření rozhraní .NET Framework 3.0. Jedná se o novou knihovnu, která slouží k vytváření uživatelského rozhraní malých klientských aplikací. Je založeno na rozhraní DirectX, a při vykreslování nezatěžuje procesor. Zatímco ovládací prvky Windows Forms jsou nativními ovládacími prvky systému Windows, které užívají popisovače okna založené na pixelech. WPF aplikace popisovače okna nepoužívá, pracuje místo pixelu s primitivy. Použité ovládací prvky mají umístění relativně k velikosti okna aplikace. Snadno se s ním mění velikost uživatelského rozhraní a je postavena na tzv. webovém stylu. WPF má i přímou podporu animací a vektorové grafiky jak 2D i základy 3D.

Psaní vlastního kódu je zde prováděno jazykem XAML (Extensible Application Markup Language), obdoba HTML. XAML je značkovací jazyk, využívaný k popisu grafického rozhraní v aplikacích společnosti Microsoft. Použité grafické elementy ve WPF mají hierarchickou stavbu, každý je součástí většího celku, nebo v sobě nese další dílčí elementy. Tímto jednoduchým způsobem si zde můžeme vytvářet vlastní ovládací prvky aplikace, jako například Ribbon.

Přímo Visual Studio obsahuje jednoduchý XAML editor. Bohužel tvorba grafického rozhraní je zde celkem pomalá. Tento základní XAML editor obsahuje jen hrstku předdefinovaných ovládacích prvků. Pro efektivní tvorbu něčeho

propracovanějšího je zde potřeba použít některé komerční knihovny s XAML elementy (např. Telerik). Některé softwarové firmy používají i vlastní vytvořené frameworky, které za vás generují XAML kód. Takový softwar potom tuto zdlouhavou práci udělá za vás. Pro opravdu profesionální práci pro tvorbu grafického rozhraní Microsoft nabízí Blend XAML editor.

Pro vývoj grafiky jsem využil jen základní XAML editor, který nám Microsoft nabízí. I pro drobné nedostatky byl pro mou práci vhodný. Mnohonásobně jeho kvality převyšují grafiku ve GDI+, nejenom v rychlosti vykreslování, ale i v možnosti definování nejrůznějších událostí pro jakýkoliv grafický prvek zobrazený v okně. Události, které prvky vyvolávají se zde dají směrovat a určovat jim prioritu jejich obsluhy. Využití Blend editoru by bylo zase příliš komplikované, využíval bych pouze jen dvojrozměrnou grafiku. Proto volba XAML pod WPF byla rozumným řešením.

```
<Window x:Class="Matematické_obrázky_11.WinFunkcePrunik"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Průnik" Height="246" Width="300" WindowStyle="ToolWindow">
    <Grid>
        <Grid Height="23" HorizontalAlignment="Left" Margin="235,10,0,0" Name="grid1" VerticalAlignment="Top" Width="23" />
        <Grid Height="23" HorizontalAlignment="Left" Margin="93,10,0,0" Name="grid2" VerticalAlignment="Top" Width="23" />
        <Label Content="Element 1:" Height="29" HorizontalAlignment="Left" Margin="12,10,0,0" Name="label1" VerticalAlignment
        <Label Content="Element 2:" Height="29" HorizontalAlignment="Left" Margin="146,10,0,0" Name="label2" VerticalAlignme
        <Border BorderBrush="Silver" BorderThickness="1" Height="1" Margin="10,48,20,0" Name="border2" VerticalAlignment="To
        <Button Content="Storno" Height="23" HorizontalAlignment="Right" Margin="0,0,11,9" Name="button1" VerticalAlignment="
        <Button Content="Ok" Height="23" HorizontalAlignment="Left" Margin="111,0,0,9" Name="button2" VerticalAlignment="Bot
        <Border BorderBrush="Silver" BorderThickness="1" HorizontalAlignment="Left" Margin="13,0,0,56" Name="border1" Width="
```

Obrázek 3. Ukázka XAML kódu

3. Jak pracovat s aplikací

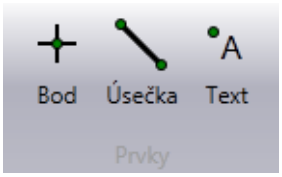
Hlavní cíl mojí aplikace je jednoduší kreslení matematických obrázků. Tento úkol jsem vyřešil tak, aby si uživatel kombinoval jednotlivé předdefinované obrázky do větších celků. Nebo mu aplikace umožní nadefinovat vlastní rozsáhlé kolekce vlastních základních obrázků. Dále v textu se budu zabývat jednoduchým postupem jak používat co možná nejefektivněji aplikaci pro tento účel.

Hlavní princip je práce s Ribbonem a postupné vkládání vytvořených obrázkových kolekcí do tohoto ovládacího prvku. Hlavní prvky pro kreslení jsou tři (čára, bod a text). Každý s těchto prvků umožní uživateli změnu jeho vlastností, jako styl čáry, barvu, velikost atd.

Všechny tyto prvky mají jeden nebo více takzvaných kotevních bodů, které se uživateli zobrazí při najetí myši na příslušný grafický objekt. Tyto kotevní body jsou zobrazovány jako zelená kolečka a jednoduchým uchopením pomocí myši, může uživatel měnit jejich polohu na rýsovací ploše. Kotevní body mohou vytvářet vazby. Tak definují vzájemný vztah mezi dvěma kotevními body. Tento vzájemný vztah definuje uživatel. Vazba kotevního bodu se projeví při změně polohy jednoho z nich. Jeden kotevní bod se pohne a druhý, který je ve vztahu s ním se pohne také, pohyb toho druhého může být libovolným směrem. Směr nám určí příslušná funkce definovaná při vytvoření této vazby. Např.: stejný pohyb obou bodů, nebo inverzní pohyb obou bodů atd.

- Grafický objekt čára - je tvořena dvěma kotevními body. Jejich posun nám mění délku a směr této čáry. Můžeme měnit vlastnosti pera, které čáru vykresluje, měnit jeho šířku, barvu a styl.
- Grafický objekt bod - je graficky znázorněn křížkem pevné velikosti. Má jeden kotevní bod, který je vždy uprostřed tohoto objektu. Změna vlastností pera je obdobná jako u čáry.
- Grafický objekt text - je tvořen jedním kotevním bodem a vlastním textovým řetězcem. Kotevní bod a textový řetězec můžou mít mezi sebou různou vzdálenost. Můžete pohybovat jen textovým řetězcem nebo kotevním bodem. Pohyb kotevního bodu ovšem hýbe i textovým řetězcem. Vlastnosti tohoto grafického prvku se nastavují trochu jinak. Uživatel musí dvakrát kliknout na textový řetězec a zobrazí se okno nazvané „Textový editor“. V něm můžeme upravovat vlastní text, volit fond, velikost písma a jeho styl.

Tyto tři základní grafické objekty jsou jakési základní stavební kameny celého kreslení, jejich vzájemná kombinace a přidělování různých vlastností uživateli umožňují vytvářet složitější celky, které si uloží pojmenované do Ribbonu.



Obrázek 4. Základní grafické objekty

S jejich pomocí a dalšími kombinacemi kreslí větší a náročnější obrázky.

Aplikace disponuje nástroji nejen pro různé změny zobrazování na rýsovacím plátně, ale i složitějšími funkcemi, které se aplikují na jednotlivé grafické objekty nebo jejich skupiny.

Návod jak efektivně kreslit matematické obrázky pomocí této aplikace si můžeme přečíst v části 5. Před tím se podrobně seznamte s uživatelskou příručkou, ve které je popis všech kombinačních funkcí.

4. Uživatelská příručka

4.1. Instalace

Aplikace je určena pro Windows XP, Vista, 7, 8.

1. Spusťte instalační soubor Matematické obrázky. Zobrazí se instalační průvodce.
2. Instalační průvodce sám zkontroluje, jestli počítač má instalované prostředí .NET Framework 4. Pokud ne, nabídne přes internet doinstalování.
3. Pokud je všechno v pořádku, pokračujte v instalaci tlačítkem Další.
4. Vyberte instalační složku, kam chcete aplikaci nainstalovat. Je automaticky nastavena složka ProgramFiles. Potvrďte tlačítkem Další.
5. Ještě jednou potvrďte instalaci tlačítkem Další.
6. Instalace probíhá, chvíli počkejte, až instalační program dokončí práci.
7. Stisknete tlačítko Zavřít.

Po úspěšné instalaci se na pracovní ploše objeví zástupce spustitelného souboru Matematické obrázky, který můžete ihned spustit.

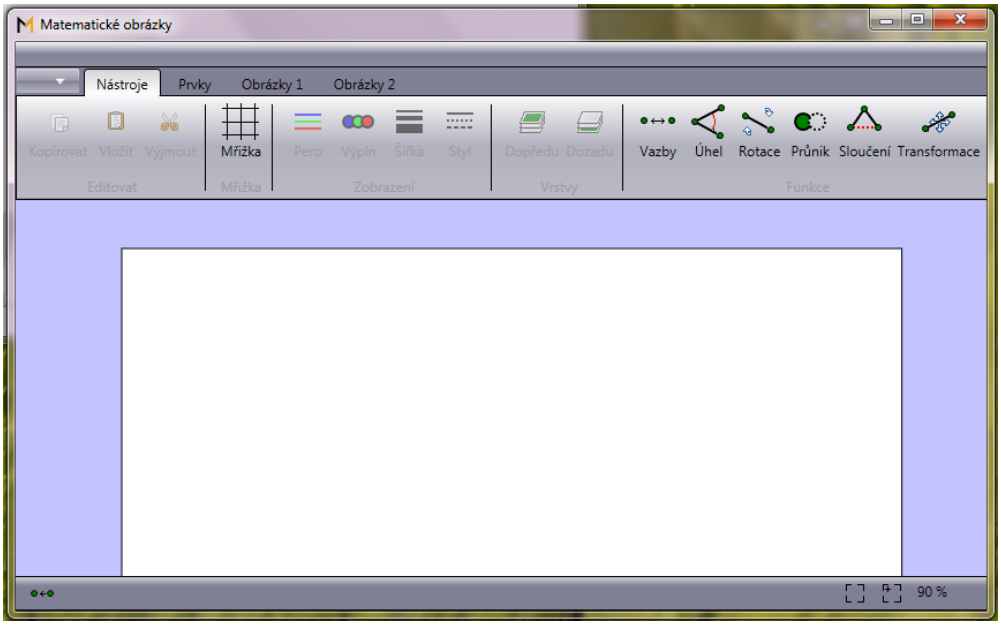
4.2. Začínáme

4.2.1. Popis hlavního okna

Hlavní okno aplikace je rozděleno na tři základní části. Ribbon (pás karet) mám sdružuje všechny ovládací funkce do logických celku. V aplikaci je umístěn horní části okna. V prostřední části se nachází vlastní rýsovací papír, na který budeme kreslit. V dolní části hlavního okna je status bar, to je pásek zobrazující různé informace. Jsou na něm umístěny i některé ikony umožnující navigaci po rýsovací ploše.

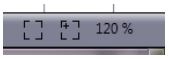
4.2.2. Navigace uživatele po plátně

Papír pro kreslení je umístěn na ploše, která má nekonečné okraje, takže je možno kreslit i mimo tento papír do jakékoliv velikosti. Posouvání celé viditelné plochy, kterou uživatel právě chce vidět (tzv. kamery), se provádí myší pomocí prostředního tlačítka (kolečka). Na jakémkoli místě můžeme toto tlačítko (kolečko) zmáčknout a držet. To znamená, jako by jsme chytily papír a pomocí posunu myši s ním posouvaly do všech stran.



Obrázek 5. Hlavní okno

Kolečko má i funkci zoomu. Pokud budeme kolečkem otáčet, papír se bude přibližovat nebo oddalovat. Střed zoomování nám určuje ukazatel myši. Hodnotu aktuálního přiblížení v procentech si můžeme přečíst vpravo dole.

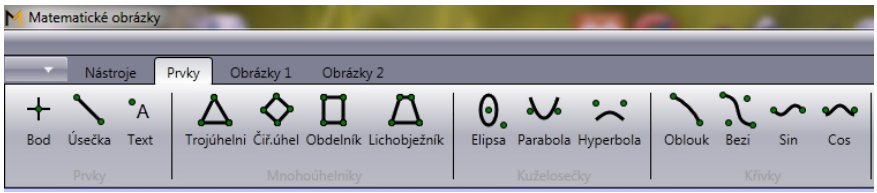


Obrázek 6. Ikony pro zobrazování plátna

Vedle tohoto ukazatele jsou dvě ikony. První nám rýsovací papír automaticky vyrovná na začátek okna do jeho levého horního rohu a zoom nastaví na 100%. Druhá nám rýsovací plátno (začínající pozice) umístí do polohy přesně na střed okna a přizpůsobí i zoom hodnotu. Tyto tlačítka nám zaručí, aby se uživatel v nekonečných okrajích při kreslení neztratil.

4.2.3. Popis ribbonu

Ribbon (pás karet) je tvořen čtyřmi hlavními záložkami: Nástroje, Prvky, Obrázky 1, Obrázky 2. Po kliknutí na vybranou záložku se nám její funkce zobrazí.



Obrázek 7. Ribbon aplikace Matematické obrázky

- Záložka nástroje - sdružuje funkce pro editaci obrázků, a funkce na jejich vzájemné slučování.
- Záložka prvky - zde najdeme předdefinované obrázky základních prvků, mnohoúhelníků, kuželoseček i některých křivek (Sin, Cos).
- Záložka Obrázky - tyto záložky fungují jako jakási databáze, kterou uživatel dynamicky definuje. Měly by sloužit pro vkládání vlastních jednoduchých obrázků. Tyto obrázky by se měly postupně používat jako základní kameny pro propracovanější celky, které si zde můžeme opět odkládat.

4.2.4. Vkládání grafických objektů

V příslušných záložkách Ribbonu si vyberte obrázek a klikněte na něj. Ten se vloží na rýsovací plochu a my s ním dále můžeme pracovat.

4.2.5. Vybírání grafických objektů nebo jejich skupin

Každý grafický objekt, který nakreslíme se dá označit a dále s ním pracovat. Toto se provádí jednoduše tak, že na něj klikneme a čáry tohoto prvku se zabarví do zelena. Můžeme vybrat i skupinu objektů, na kterou je možné aplikovat funkce. Výběr se provádí najetím ukazatele myší mimo grafické objekty a držením levého tlačítka, tažením myši po plátně se nám zobrazí modré pole. Když prvek bude pokryt tímto modrým polem, tak se jeho čáry zbarví do modra. To znamená, že je vybrán do skupiny a my na něj později můžeme aplikovat vybrané funkce. Modrá pole můžeme libovolně rozšiřovat tažením jakékoli jeho hrany, nebo popřípadě celé pole přesouvat pomocí levého tlačítka myši.

4.2.6. Základní manipulace s grafickými objekty

Grafické objekty, které se zde vykreslují jsou tvořený čárami. Tyto čáry mohou mít různé vlastnosti, šířku, barvu, styl pera. Čáry jsou zakončeny takzvanými kotevními body. Zobrazují se po najetí myši na objekt. Při označení vybraného objektu se jeho kotevní body zobrazí a jsou viditelné i když je tento objekt překrytý jiným. Při posunu těchto bodů, měníme délku a směr čáry vedoucí mezi těmito body. Pro přesunu celého objektu v nezměněném stavu musíme uchopit objekt za čáru, tím posuneme i jeho kotevní body.

4.3. Záložka Nástroje

4.3.1. Kopírovat, Vložit, Smazat

Tyto základní funkce můžeme využívat tak, jak jsme zvyklí i z jiných aplikací. Stačí označit objekt nebo skupinu objektu a vybrat požadovanou funkci. Fungují i standardní klávesové zkratky.

- Klávesy Ctrl + C pro kopírování příslušného obrázku do paměťové schránky.
- Klávesy Ctrl + X pro mazání příslušného obrázku a zároveň jeho obraz zkopíruje také do paměťové schránky.
- Klávesy Ctrl + V pro vkládání objektu z této paměťové schránky.

Pomocí těchto funkcí můžeme velice urychlit a maximálně optimalizovat rychlost kreslení obrázků.

4.3.2. Mřížka

Kliknutí na toto tlačítko nám zobrazí mřížku po celé rýsovací ploše. Mřížka slouží pro přesnou navigaci na ploše a umožňuje jednoduší směrovou orientaci nakreslených objektů.

4.3.3. Pero, Výplň, Šířka, Styl

Tyto funkce využíváme ke změně vlastností jednotlivých grafických objektů. Nastaví se barva výplně nebo barva čar. Nastaví se šířka nebo styl nakreslené čáry (plná, čerchovaná atd.).

4.3.4. Dopředu, Dozadu

Funkce umožní vybranému grafickému objektu nebo skupině jejich zobrazení v popředí nebo v pozadí oproti ostatním objektům.

4.3.5. Vazby, Úhel, Rotace, Průnik, Sloučení, Transformace

Tyto funkce tvoří hlavní kombinační a transformační metody, které aplikace využívá. Proto jim budeme věnovat při vysvětlování více času. Na funkce jsou definované klávesové zkratky.

- Klávesa V pro vazby.
- Klávesa U pro úhel.
- Klávesa R pro rotaci.
- Klávesa P pro průnik.
- Klávesa S pro sloučení.
- Klávesa T pro Transformace.

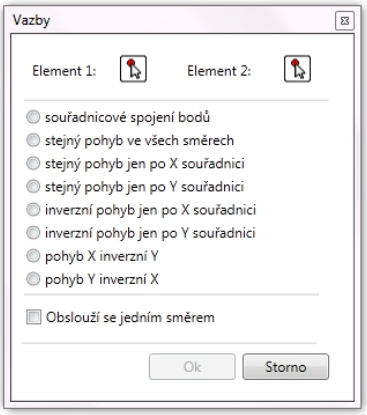
Po kliknutí na libovolnou funkci se nám zobrazí tzv. Nástrojové okno. To je okno přes které danou funkci nastavíme. Jejich grafické uspořádání je zase rozděleno do několika části, tyto části jsou odděleny čarou.

V horní části nástrojového okna jsou ikony, které nám umožňují vybrat grafické objekty. Na tyto objekty (nazveme je Elementy) budeme danou funkci využívat. Ikony nám pomocí obrázků napovídají, které elementy může daná funkce využívat např. kotevní bod nebo čára.

K čemu funkce jsou, si nejlépe vysvětlíme na jednoduchých příkladech uvedené pod stručným popisem každé funkce.

4.3.6. Funkce Vazby

To je funkce, kterou vytváříme nějaký vztah mezi dvěma kotevními body. Tento vztah nám slouží při pohybu kotevních bodů po rýsovací ploše.



Obrázek 8. Okno funkce vazby

Příklad:

1. Vložte dvě čáry na rýsovací plochu.
2. Vyvolejte funkci Vazby.
3. Klikněte do nástrojového okna funkce na ikonu Element 1, ten chcete definovat jako první. Ikona se aktivuje (oranžová). Potom vyberte libovolný kotevní bod na rýsovací ploše na který chcete funkci uplatnit. V našem případě třeba na kotevní bod čáry vlevo. Pokud ho funkce příjme, tak zmodrá a ikona Elementu 1 se změní na zelenou fajfku.

4. Klikněte do nástrojového okna funkce na ikonu Element 2. Ikona se aktivuje (oranžová). Potom vyberte libovolný kotevní bod na rýsovací ploše, na který chcete funkci uplatnit. Třeba na kotevní bod čáry vpravo. Pokud ho funkce příjme, tak zmodrá a ikona Elementu 1 se změní na zelenou fajfku.

5. V prostřední části nástrojového okna vyberte druh vazby, který chcete uplatnit pro tyto dva definované elementy, např. stejný pohyb ve všech směrech.

6. Potvrďte tlačítkem OK.

Definovali jsme si první vazbu a teď si ukážeme jestli funguje. Zkuste pohnout jednou čarou. A vidíte, že se nám hýbe i kotevní bod čáry druhé. Pokud chcete tuto vazbu zrušit, musíte přes ikonu Definované vazby, která je umístěná vlevo dole na status baru. Ale o tom později v části 4.6.

Funkce Vazba jde přidělit jakémukoliv kotevnímu bodu na jakékoli čáře, textu nebo kružnici. Zkrátka na jakýkoliv zelený bod co uvidíte na rýsovací ploše. Vazby jsou různé, přidělujeme jim nejenom možnost stejného pohybu, ale i inverzního pohybu, nebo umístění bodů na stejné souřadnice.

Různé druhy obsluhy vazeb:

- Souřadnicové spojení bodů - znamená, že Element 1 se přesune na souřadnice Elementu 2.

- Stejný pohyb ve všech směrech - při pohybu Elementu 1 se nám hýbe i Element 2 o stejnou vzdálenosti v jakémkoli směru.

- Stejný pohyb jen po X souřadnici - pokud se hýbe Element 1 ve vodorovném směru, tak se pohybuje i Element 2 o stejnou vzdálenost jen ve vodorovném směru.

- Stejný pohyb jen po Y souřadnici - pokud se hýbe Element 1 ve svislém směru, tak se pohybuje i Element 2 o stejnou vzdálenost jen ve svislém směru.

- Inverzní pohyb jen po X souřadnici. - pokud se hýbe Element 1 ve vodorovném směru, tak se pohybuje i Element 2 v opačném směru o stejnou vzdálenost jen vodorovně.

- Inverzní pohyb jen po Y souřadnici - pokud se hýbe Element 1 ve svislém směru, tak se pohybuje i Element 2 v opačném směru o stejnou vzdálenost jen svisle.

- Pohyb X inverzní Y - pokud se hýbe Element 1 ve vodorovném směru, tak se pohybuje i Element 2 ve svislém směru o stejnou vzdálenost.

- Pohyb Y inverzní X - pokud se hýbe Element 1 ve svislém směru, tak se pohybuje i Element 2 ve vodorovném směru o stejnou vzdálenost.
- Vazbu jenom jedním směrem - znamená, že při pohybu Element 1 se vazba obslouží, ale pří pohybu Elementu 2 už ne.

Tímto způsobem pomocí jednoduchých čar si můžeme nadefinovat složitější obrázky, které při přesouvání po rýsovacím plátně neztrácejí tvar nebo mezi sebou drží nějaký logický vztah. Jako takový jednoduchý příklad nám poslouží obrázek obdélníku. Ten je složen ze čtyř čar, které jsou mezi sebou propojené příslušnou vazbou.

4.3.7. Funkce Úhel

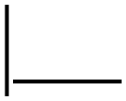
Funkce úhel je funkce, která definuje vztah mezi dvěma čárami. V tomto případě je to vztah velikosti úhlu, které čáry mezi sebou svírají.

Příklad:

1. Vložte dvě čáry na rýsovací plochu.
2. Vyvolejte funkci Úhel.
3. Definujte dva elementy. V tomto případě čáry, které budou mezi sebou úhel svítat. Jejich definování je obdobné jako v předchozím příkladu.
4. V prostřední části nástrojového okna vyberte jaký úhel. Buď hodnotu zadejte číslem, nebo vyberte z možností Kolmost, Rovnoběžnost.
5. Potvrďte tlačítkem OK.

Obě čáry drží námi zadaný úhel. Tato funkce je spíše pomocná. Je zde jedno uživatelské omezení. Nedají se nastavit na jednu čáru dva úhly i když budou ve vztahu s jinými prvky.

Obrázek 9. Úhel čar, začátek



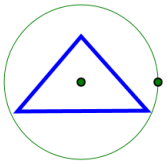
Obrázek 10. Úhel čar, výsledek

4.3.8. Funkce Rotace

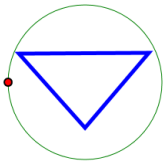
Funkce rotace je funkce, která zprostředkovává rotaci objektu, nebo celého obrázku.

Příklad:

1. Vložte některý grafický objekt na rýsovací plochu.
2. Vyvolejte funkci Rotace.
3. Musíte definovat, co má rotovat. Klikněte do nástrojového okna funkce na ikonu Elementu 1. Ikona se aktivuje (oranžová). Potom klikněte na graficky objekt na rýsovací ploše, kterým chcete rotovat. Pokud chcete rotovat skupinu nebo celým obrázkem, označte tyto objekty do modrého obdélníku a potom na tento obdélník kliknout. Tím tyto objekty uvnitř něj předáte funkci. Pokud ho funkce příjme ikona Elementu 1 se změní na zelenou fajfku a na rýsovacím plátně se nám ukáže terč. Jeho střed nám určuje střed rotace a druhé zelené kolečko zase náklon rotace. Oběma kolečky se dá libovolně pohybovat.
4. V prostřední části nástrojového okna vyberte jaký způsob rotace chcete uplatnit. Rotujte pomocí zeleného kolečka na terči, nebo hodnotu rotace zadejte číslem.
5. Po potvrzení klikněte na OK.



Obrázek 11. Rotace, začátek



Obrázek 12. Rotace, průběh

4.3.9. Funkce Transformace

Tato funkce nám zajistí změnu velikost nebo přesun vybraného objektu nebo celého obrázku.

Příklad:

1. Vložte některý grafický objekt na rýsovací plochu.

2. Vyvolejte funkci Transformace.

3. Musíte definovat, co se má transformovat. Uplatníte stejný postup jako v předchozím příkladu. Pokud váš výběr funkce Transformace přijala, objeví se vám na rýsovacím plátně zelený obdélník. Ten můžete libovolně přesouvat a s ním i objekty, které jste definovali jako Element 1. Pokud pohybujeme jeho bodem vpravo dole, tak měníme poměr velikosti těchto objektů, včetně šířky čar, kterými jsou nakresleny.

4. Po potvrzení klikněte na OK.

Obrázek 13. Transformace, začátek

Obrázek 14. Transformace, průběh

4.3.10. Funkce Sloučení

Funkce nám umožní sloučit několik čar do jednoho n-úhelníku, který můžeme vybarvovat nebo dále využívat ve funkci průnik.

Příklad:

1. Vložte dvě čáry na rýsovací plochu.

2. Jednu z nich posuňte tak, aby se čáry dotýkali přesně jen jedním kotevním bodem. Pokud si jejich polohou nejste jistí, na tyto kotevní body uplatněte funkci vazba - souřadnicové spojení bodu.

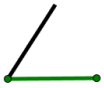
3. Vyvolejte funkci Sloučení.

4. Jako Element 1 zaklikněte libovolnou čáru, zabarví se modře. K této čáře můžete postupným zaklikáváním dalších a dalších čar přidávat jednotlivé segmenty(části) n-úhelníku, který chceme vytvořit. V našem případě druhou nakreslenou čáru. Pokud se obě čáry dotýkají v kotevním bode, tak se i druhá čára zabarví do modra.

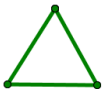
5. V další části nástrojového okna zaklikněte „spoj počáteční a konečný bod smyčky“.

6. Potvrďte tlačítkem OK.

Výsledkem této funkce bude trojúhelník. Ten se zobrazí jako jeden objekt se třemi kotevními body. Tento trojúhelník můžete vybarvovat nebo dále poskytnout funkci průnik.



Obrázek 15. Slučování čar, začátek



Obrázek 16. Slučování čar, vysledek

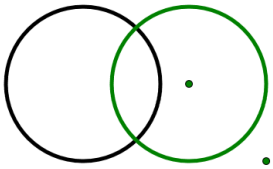
4.3.11. Funkce Průnik

Funkce nám vytváří možnost mezi sebou kombinovat dva objekty podle různého nastavení. Typy tohoto nastavení jsou: Průnik, Sjednocení, Rozdíl, Xor. Můžete podle ní vytvářet např. Vénnovy diagramy. Do funkce nám mohou vstupovat pouze uzavřené grafické objekty např, kruh nebo sloučený n-úhelník.

Příklad:

1. Vložte dva kruhy na rýsovací plochu.
2. Jeden z nich posuňte tak aby, se částečně překrývaly.
3. Vyvolejte funkci Průnik.
4. Jako její Elementy definujte oba tyto kruhy.
5. V prostřední části nástrojového okna vyberte jaký typ funkce průnik se má aplikovat. V našem případě zaklikněte průnik.
6. Potvrďte tlačítkem OK.

Výsledek můžete vidět na obrázku 18. I v tomto objektu platí nadále všechny kotevní body, které byly doposud definované a i jejich vazby.



Obrázek 17. Průnik kruhů, začátek

Obrázek 18. Průnik kruhů, vysledek

4.4. Záložky Prvky

V této záložce jsou předdefinované prvky pro jednodušší kreslení. Základní prvky bod, úsečka, text slouží jako základní kameny všech kreslených obrázků. Další prvky jako mnohoúhelníky, kuželosečky a některé křivky jsou zde předdefinované jako jejich kombinace.

4.4.1. Prvky

V této části záložky jsou umístěný základní grafický objekty Bod, Text, Úsečka. Jejich význam vysvětluje část 3.

4.4.2. Mnohoúhelníky

V této části Ribbonu jsou předdefinované některé mnohoúhelníky. Trojúhelník, obdélník, kosodélník, lichoběžník. Tyto prvky byly vytvořeny jen z úseček navzájem propojených vazbami. Pro jejich snadnější manipulaci a možnosti je vyplňovat barvou, jsou tyto úsečky sloučeny do jedné uzavřené křivky (pomocí funkce sloučit).

4.4.3. Kuželosečky

Elipsa, parabola, hyperbola. Tyto křivky jsou tvořeny jednou čarou. Možnost změny tvaru křivky se ovládá pomocí dvou kotevních bodů. Vliv na tuto křivku má jejich vzájemná poloha. V této verzi aplikace zatím nejdou kuželosečky slučovat (funkcí sloučení) do jedné spojené křivky.

4.4.4. Křivky

Zde jsou zastoupeny křivky jako oblouk, bézierova křivka, křivky průběhu funkce sinus nebo cosinus.

- Sin, Cos - ovládání a modifikace se provádí stejně jako u kuželoseček. I tyto křivky zatím nejdou vzájemně slučovat.
- Bézierova křivka - je pojmenovaná po francouzském matematiku Bézierovi. Tato aplikace využívá její tzv. jednoduchou kubickou křivku. Definuje se čtyřmi kotevními body. Křivka začíná v prvním kotevním bodě a končí v posledním. Další dva body jsou zde jako řídící, lze si je představit jako magnety, které k sobě křivku přitahují. Tímto jednoduchým způsobem můžete nakreslit libovolný tvar. Na tyto křivky můžete uplatnit funkci sloučit a vzájemně je spojovat do jakéhokoliv obrazce.
- Oblouk - je tvořen dvěma kotevními body. Tuto křivku také můžete libovolně slučovat. Používá se ke grafickému znázornění úhlů.

4.5. Záložka Obrázky 1 a Obrázky 2

Tyto záložky fungují jako databáze definovaných obrázků. První dvě ikony v této záložce umožňují libovolné přidávání nebo odebírání nakreslených obrázků do příslušné záložky. Přidávání se provádí označením objektu nebo skupiny (modrý obdélník) a kliknutím na ikonu přidat. Vyskočí vám okno s dotazem na zadání jména. Každý takto definovaný objekt, který chcete uchovat v Ribbonu musí mít jméno. Odebírání se provádí přes ikonu Odeber. Po kliknutí na toto tlačítko se zobrazí okno se jmény jednotlivých obrázků. Vyberte, který chcete z Ribbonu odstranit a potvrďte výběr tlačítkem OK. Všechny změny, které provedete v Ribbonu jsou trvalé, po správném vypnutí aplikace se tyto změny uloží.

Obrázek 19. Tlačítka pro přidání a odebírání prvků do Ribbonu

4.6. Jak zrušit definované vazby

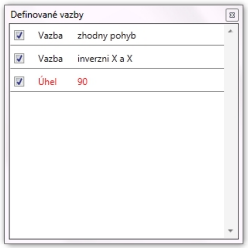
K tomuto účelu je ikona vlevo dole. Pokud na ni kliknete, zobrazí se okno nazvané Definované vazby. Jestliže máte vytvořenou jakoukoliv vazbu nebo úhel (dále jen vztah), ten se zde zobrazí i s příslušným popisem. U každé položky, která definuje jeden vztah je ikona zaškrtávacího políčka. Pokud je zaškrtnuto, vztah je aktivní a obsluhuje se. Pokud zde symbol fajfky chybí, tak je tento vztah nefunkční.

Aplikace sama detekuje i kolizní vztahy. Nemůžeme zadat vazbu, která je v rozporu s jinou. Tato kolize se projeví, až v případě pohybu nějakého kotevního bodu. Pokud by ke kolizi došlo, vlastní pohyb se neprovede a kotevní bod změní barvu na černou. V okně Definované vazby se kolizní vztah zobrazí červeně. Ten se musí odškrtnutím vypnout a tím znemožnit, aby ke kolizi při pohybu nedocházelo. Okno „Definované vazby“ můžete mít stále zobrazeno a zároveň kreslit. Jakékoliv změny v definici vazeb nebo úhlů se zde dynamicky zobrazují.

Jako příklad kolizního vztahu si uvedeme dva transformační body, pokud na ně definujete dvě logicky rozdílné vazby, například stejný pohyb a zároveň inverzní pohyb. Tak se při pokusu o jejich pohyb, detekuje kolize a příslušné vztahy se neobslouží.

Upozornění: Uživatelsky definované vztahy se v tomto okně zobrazují, ale pokud obrázek uložíte buď do Ribbonu nebo na hard disk a znovu otevřete. Tyto

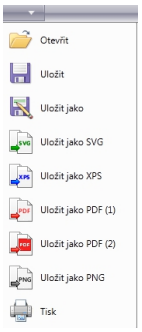
uložené vztahy budou sice aktivní, ale nemůžete je zpětně vypnout (nezobrazí se v okně Definované vazby). Důvod je zřejmý, aplikace slouží k postupnému definování obrázků od jednoduchých ke složitějším. Pokud by se zde tyto vztahy zpětně zobrazovali, okno by bylo zavalené spoustou vztahů a jejich modifikace by byla nepřehledná.



Obrázek 20. Okno Definované vazby

4.7. Ukládání

Vpravo nahoře je šedá ikona pro menu. Přes ni se dá nakreslený obrázek uložit nebo exportovat do jiných formátu, které používají jiné aplikace.



Obrázek 21. Znázorněné menu

- Formát XPS - XML Paper Specification, je nový formát dokumentů založený na XML podobný formátu PDF od Adobe Systems. Používá ho Windows od verze Vista.

- Formát SVG - Scalable Vector Graphics, který popisuje dvojrozměrnou vektorovou grafiku pomocí XML. Hodně se používá na Internetu. Aplikace pracující s SVG CorelDraw, Inscape.
- Formát PDF(1) - Portable Document Format, souborový formát vyvinutý firmou Adobe pro ukládání dokumentů nezávisle na softwaru i hardwaru.
- Formát PDF(2) - do formátu PDF se obrázek uloží ve formě bitmapy.
- Format PNG - bitmapový format, podporující průhlednost.

4.8. Otevírání

Otvírání uloženého obrázku se provádí také jen přes ikonu menu. Otvírat jdou soubory vytvořené jen v této aplikaci, aby se zachovala funkčnost všech vazeb a uhlů.

4.9. Tisk

Tisk nakresleného obrázku se také provádí jen přes ikonu menu. Aplikace vám nabídne standardní dialog pro tiskárnu. Tiskne se oblast, kterou právě vidíme na monitoru. Proto pro správné zarovnání tisku na zobrazovaný papír klikněte na ikonu (zarovnat) na status baru.

4.10. Undo, redo

Funkce undo (zpět) nám umožní jakýkoliv krok, který provedete v aplikaci, ho vrátit zpět. Funkce redo (znovu) nám umožní kroky, které jsme vrátili zpět opět provést.

5. Návod na efektivní kreslení matematických obrázků

Jak jsem v textu již několikrát zmínil, aplikace by se měla správně používat tak, že si nejprve vytvoříme jednoduší obrázky. Uložíme je do Ribbonu a pomocí nich vytváříme složitější. Dále v textu si popíšeme krok po kroku, jak správně postupovat při tvorbě takových obrázků. Všechny níže zmíněné obrázky můžete pro ukázku najít v záložce Obrázky 1.

Po přečtení části 5.1. se vám možná způsob kreslení bude zdát těžkopádný a složitý. Nadefinování prvních obrázků vám poslouží jako základní kameny. Rychlost kreslení složitějších a složitějších struktur výrazně urychlí a potom bude její používání jednoduché a rychlé.

5.1. Základní obrázky

5.1.1. Bod s textem

Zkusíte si první jednoduchý prvek, který bude poměrně využívaný pro tvorbu matematických obrázků.

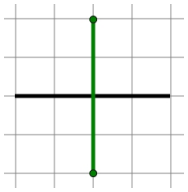
1. Vložte prvek bod.
2. Vložte prvek Text.
3. Definujte na jejich kotevní body vazbu - souřadnicové spojení bodů.
4. Po potvrzení vazby, označte oba objekty pomocí modrého pole.
5. Vložte je do Ribbonu pomocí tlačítka přidat.
6. Pojmenujte si ho.

Tímto způsobem jste si nadefinovali vlastní kombinovaný obrázek a ten již můžete využívat k dalším kreslení.

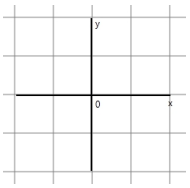
5.1.2. Osový kříž

Tento prvek bude trochu složitější, ale při jeho definici si nejlépe ukážeme význam vazeb.

1. Vložte dvě čáry.
2. Znázorněte si mřížku, pomocí příslušného tlačítka v Ribbonu.



Obrázek 22. Rozmístění čar



Obrázek 23. Osový kříž

3. Ručně nastavte obě čáry tak, aby byly k sobě kolmé a křížily se ve svém středu. Jejich přesnou polohu vám pomůže nastavit znázornění mřížky jak ilustruje obrázek 22..

4. Vyvolejte funkci vazba.

5. Definujte do této funkce oba kotevní body svislé čáry a přidělte jí obsluhu typu „stejný pohyb jen po X souřadnici“.

6. Potvrďte tlačítkem OK.

7. Vyvolejte funkci vazba.

8. Definujte do této funkce oba kotevní body vodorovné čáry a přidělte jí obsluhu typu „stejný pohyb jen po Y souřadnici“.

9. Potvrďte tlačítkem OK. Takto definované vazby vám zaručí stálou kolmost obou čar.

10. Vložte objekt „Bod s textem“, který jste uložili do Ribonu a umístěte ho ručně na střed překřížených čar.

11. Definujeme si další vazbu, jako její elementy přidělíme tentokrát kotevní bod právě vloženého objektu „Bod s textem“ a jeden s kotevních bodů svislé čáry. Jako typ obsluhy zadejte „stejný pohyb jen po X souřadnici“.

12. Definujte vazbu mezi kotevním bodem vodorovné čáry a kotevního bodu objektu „Bod s textem“ a obsluhu zadejte „stejný pohyb jen po X souřadnici“.

13. Vložte dva nové objekty Text.

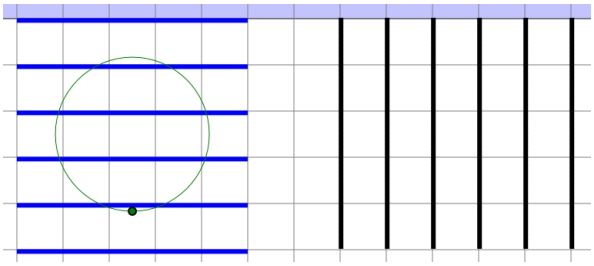
14. Jednomu z nich přidělíte vazbu k jednomu kotevnímu bodu svislé čáry, obsluhu nastavte „souřadnicové spojení bodů“.

15. Druhému z nich přidělíte vazbu k jednomu kotevnímu bodu vodorovné čáry, obsluhu nastavte „souřadnicové spojení bodů“.

16. Změníme nápisy textových objektů na označen X, Y a uprostřed kříže 0.
17. Označte modrým obdélníkem nakreslený kříž a celému tomuto objektu nastavte tloušťku čar na hodnotu 2.
18. Označte vše modrým obdélníkem a vložte ho do Ribbonu.
19. Pojmenujte si ho.

5.1.3. Mřížka ke grafům

1. Vložte čáru.
2. Znázorněte si mřížku, pomocí příslušného tlačítka v Ribbonu.
3. Čáru ručně nastavte do kolmé pozice.
4. Zkopírujte ji a její kopii posuňte o kus dál.
5. Tento krok opakujte pětkrát.
6. Všechny tyto čáry označte a použijte kopírovat.
7. Vyvolejte funkci transformace.
8. Znovu označte nakreslené čáry a ty vložte jako element do funkce transformace.
9. Znázorněný zelený obdélník uchopte myší uprostřed a přetáhněte o kus dál.
10. Potvrďte tlačítkem OK.
11. Použijte funkci vložit (ctrl+v). Ze schránky se vloží okopírované čáry.
12. Znovu označte jednu polovinu znázorněných čar.
13. Vyvolejte funkci rotace.
14. Vložte do této funkce označené čáry a otočte je o 90° jako na obrázku 24.
15. Pomocí funkce transformace přesuňte levé čáry na pozice pravých. Vznikne mřížka.
16. Po vycentrování čar potvrďte tlačítkem OK.
17. Označte všechny čáry a nastavte jim šířku na hodnotu 1 a styl na čerchované čáry.
18. Označte vše modrým obdélníkem a vložte mřížku do Ribbonu.
19. Pojmenujte si ji.



Obrázek 24. Postup při tvorbě mřížky

5.1.4. Znázornění úhlu

Tento prvek, budeme potřebovat jen na lepší grafické znázornění kreslených úhlů.

1. Vložte oblouk.
2. Vložte čáru.
3. Pokud není některý kotevní bod čáry na některém kotevním bodu oblouku, tak je musíte ručně přesunout, nebo definovat vazbu na tyto kotevní body - souřadnicové spojení bodů.
4. Vyvolejte funkci sloučit.
5. Jako element označte oblouk.
6. Jako další navazující element označte čáru.
7. Zatrhněte políčko spoj počáteční a koncový bod smyčky.
8. Potvrďte tlačítkem OK.
9. Výslednému prvku změníme šířku pera na 2.
10. Změníme barvu pera na tmavě zelenou.
11. Změníme výplň na světle zelenou.
12. Vložte prvek do Ribonu.
13. Pojmenujte ho.

Obrázek 25. Znázornění úhlu

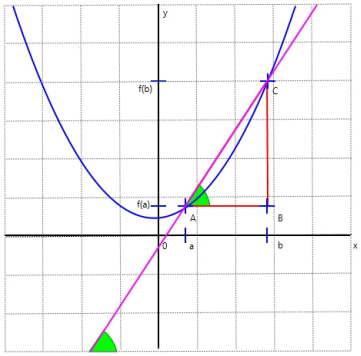
5.2. Složitější matematické obrázky

Zde je stručný popis tvorby výsledných matematických obrázků. Všechny níže uvedené jsou pro příklad uloženy v záložce Obrázky 2.

5.2.1. Sečna funkce se zvýrazněným trojúhelníkem

1. Vložte vámi nadefinovaný prvek „Mřížka ke grafu“.
2. Vložte vámi nadefinovaný prvek „Osa kříže“.
3. Kříži upravte velikost a polohu.
4. Vložte parabolu.
5. Změňte jejich šířku a barvu pera.
6. Vložte trojúhelník.
7. Změňte mu šířku a barvu pera.
8. Nastavte pomocí funkce úhel na dvou ramenech trojúhelníku 90°.
9. Vložte tři vámi nadefinované prvky „Bod s textem“.
10. Upravte jim text na A,B,C a pomocí funkce vazby jim přidělte polohu na vrcholy trojúhelníku.
11. Upravte pozici trojúhelníku podle obrázku 26.
12. Vložte čáru.
13. Změňte její šířku a barvu pera.
14. Upravte její pozici tak, aby procházela body A, C.
15. Označte čáru a dejte ji do pozadí tlačítkem „Dozadu“.
16. Vložte dva vámi definované prvky „Znázornění úhlu“.
17. Nastavte jim polohu podle obrázku 26. a dejte je také do pozadí.
18. Vložte čtyři vámi nadefinované prvky „Bod s textem“.

19. Upravte jejich pozici podle obrázku tak, aby každý z nich ležel na jedné z os osového kříže.
20. Změňte jim text na a, b, f(a), f(b).
21. Můžete jim definovat příslušnou vazbu s příslušným bodem.
22. Opatřete obrázek příslušnými popisky.

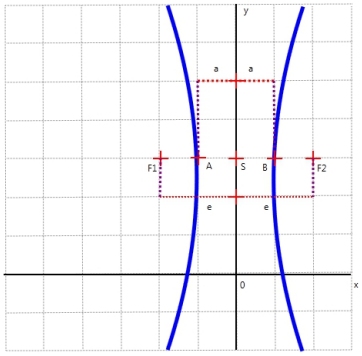


Obrázek 26. Sečna funkce s výrazněným trojúhelníkem

5.2.2. Hyperbola s popisky

1. Vložte vámi nadefinovaný prvek „Mřížka ke grafu“.
2. Vložte vámi nadefinovaný prvek „Osa kříže“.
3. Kříži upravte velikost a polohu.
4. Vložte dvě hyperboly.
5. Otočte je proti sobě a jejich kotevní body umístěte do stejné roviny.
6. Definujte vazbu „stejný pohyb jen po Y souřadnici“ na proti sobě stojícími kotevní body.
7. Nastavte hyperboly podle obrázku 27.
8. Vlože čáru.
9. Vložte dva objekty „Text“.

10. Definujte vazbu stejný pohyb ve všech směrech na kotevní bod tohoto jednoho s textů a některý kotevní bod čáry. Tím si zaručíme stejnou polohu textu při přesunu čáry.
11. Čáře nastavte barvu červenou, šířku 2 a styl čerchovaný.
12. Čáru i s texty zkopírujte a nastavíme jejich polohu podle obrázku 27.
13. Vložíme pět vámi definovaných prvků „Bod s textem“.
14. Upravte jim text a umístěte je podle obrázku.
15. Můžete jim definovat příslušné vazby.
16. Vložte zbytek pomocných čar s červenou nebo fialovou barvou.



Obrázek 27. Hyperbola s popisky

6. Rozbor architektury aplikace a struktury kódu

Vytvořená aplikace je interaktivní editor, kde hodně záleží na uživatelském prostředí a jednoduchosti a přehlednosti ovládání. Pro vývoj kódu jsem zvolil evoluční model. Naprogramoval jsem základní GUI a pomocí různých testů jsem požadavky na ovládání různě upravoval. Při postupném řešení složitějších problémů se celá architektura aplikace změnila. Celé její jádro prošlo také evolučním vývojem. Při konečném ujasnění požadavků se třetí verze mohla konečně naplno rozvíjet a do aplikace přidávat další funkce. Proto kód aplikace má některé části zbytečně složitě strukturované, to je daň za použití evolučního modelu vývoje.

Základní architektura je rozdělena na tři hlavní části. Tyto části řídí tři třídy, jakoby manažeři, kteří jsou v hierarchii tříd na jejím vrcholu a mají na sebe přímě odkazy. Veškerá komunikace mezi těmito třemi stupni probíhá přes ně. Tyto tři základní třídy již mají v sobě jiné objekty nebo jejich celé kolekce. Ty vytváří v aplikaci složitou stromovou strukturu objektů.

Další text stručně popisuje jednotlivé vrstvy kódu aplikace. Pro lepší představu o fungování struktury aplikace jsem tyto vrstvy pojmenoval podle řídících objektů každé z nich.

6.1. Manažer GUI

Přijímá události různých zobrazovaných grafických prvků, přiděluje jim prioritu a stará se o jejich směrování. Stará se o zobrazování grafických objektů, která jsou umístění v kolekcích v části „Manažer Jádro“. V jeho canvasu se skládá výsledný obraz.

6.2. Manažer Jádro

Zde jsou umístěny kolekce grafických objektů, jednotlivé objekty jsou řazeny do stromové struktury, např. objekt obsahuje čáry, čára obsahuje kotevní bod, kotevní bod obsahuje vazbu atd. Stromovou strukturu zde upravuje několik statický tříd, které se starají o správnou hierarchii stromu a kontrolují kolizní stavy při obsluze vazeb. Starají se o řízení stavů, ve kterých se celá aplikace nachází např.: zákaz pohybu grafických objektů, zákaz výběru grafických objektů, povolení zobrazení kotevních bodů atd. Tento manažer obsahuje i několik abstraktních tříd, které umožňují polymorfismus. S jeho pomocí umožňují programátorovy libovolně dopisovat funkce do aplikace, které si zadavatel práce vymyslí např.: rotace, transformace, sloučení atd. Všechny tyto funkce mají stejně napsané jádro, jejich kód

se doplní o obsluhu, která se provede jinak. Proto i tyto funkce se s uživatelského hlediska ovládají podobně. Funkcím se podobně zadávají parametry a definují jejich elementy.

6.3. Manažer Ukládání

Tato třída a její objekty se starají o práci s hard diskem a ukládání obrázků do XML souborů. Řídí transformace do jiných formátů, starají se o uložení grafických objektů pro Ribbon. Kontrolují i správnost načtených dat pomocí kontrolního součtu a přidělování ID čísel pro sestavování stromové hierarchie.

7. Použité algoritmy

Pro tuto aplikaci jsem vytvořil celou řadu algoritmů. V této části bych zmínit jenom ty nejzajímavější.

7.1. Přidělování ID čísel

Veškeré objekty, které aplikace používá jsou řazeny do stromové hierarchie. Při používání funkce vazby dokonce mluvíme již o síti objektů. Celá tato struktura se musí dát libovolně kopírovat, ukládat a zpět vkládat. Jazyk C# nedokáže jednoduše zkopírovat ani jednoduché objekty natož tak složitou strukturu. Proto jsem zde vymyslel jednoduché řešení. Všem objektům, které se vytvoří a později se budou kopírovat, jim aplikace přidělit tzv. identifikační číslo (dále jen ID). Při kopírování a zpětném vkládání se toto číslo využívá k vytvoření nové kopírované struktury.

Nejprve se musí všechny nově vytvořené objekty, které jsme zkopírovali (grafické prvky, vazby, úhly) vložit do speciálních kolekcí. Tyto objekty mají ve svých slotech, místo přímého ukazatele na jejich potomky, ID těchto potomků. Postupně se tyto kolekce prochází a přiřazuje se místo ID čísla příslušný ukazatel na nově vytvořený objekt umístěn fyzicky v těchto kolekcích. Po vytvoření této zkopírované struktury se musí zajistit změna původních ID (uložených) na nová ID (která aplikace nově použije pro případné další kopírování). Celá takto vytvořená struktura se vloží do kolekce nakreslených obrázků a zobrazí se na rýsovacím plátně.

7.2. Kontrola kolize vazeb

Jakýkoli pohyb kotevních bodů je řešen dvoufázově. Začíná se žádostí o vyvolání pohybu. Kotevní body, které o pohyb žádají, si nejprve uloží do vlastního speciálního slotu nové souřadnice, kam by se měli přesunout. Pokud mají na sebe navázány jiné vztahy (funkce vazby nebo funkce úhel) s jinými kotevními body, tak i tyto body vyvolají žádost o pohyb. Pokud žádáme o pohyb kotevní bod, který má již svůj slot s novými souřadnicemi obsazen, provede se kontrola jestli žádáme o přesun na ty samé souřadnice. Výsledek kontroly se pošle do statické třídy, která celý pohyb řídí. Po skončení fáze žádostí statická třída provede fázi druhou. Buď všem kotevním bodům, které o pohyb žádaly vyhoví a body přesune nebo přesun nepovolí a zjistí, který bod měl v souřadnicích kolizi. Ve fázi žádostí je potřeba ještě ošetřit cyklické pohyby, kdy jeden kotevní bod žádá o pohyb druhy a ten druhý žádá zase ten první. Toto hlídá stejná statická třída, která shromažďuje ID čísla vztahů, které již byly obslouženy. Každá žádost o další obsluhu musí tuto kolekci projít a v případě shody ID čísla se tento vztah neobslouží.

7.3. Obsluha funkce uhel

Do této funkce vstupují dvě čáry, každá je tvořena dvěma kotevními body. Pojmenujeme si je k1, k2, k3, k4. Tyto kotevní body jsou společně ve vztahu nazvaném úhel. Ten je definován číslem, které nám vyjadřuje ve skutečnosti velikost úhlu, který svírají dva vektory. Vektor V1 tvořený souřadnicemi bodů k1, k2 a vektor V2 tvořený souřadnicemi bodů k3 a k4. Při pohybu jakéhokoliv z těchto bodů se nám změní i úhel mezi vektory. Aplikace vypočítá odchylku v úhlu a snaží se ji zpětně vyrovnat pohybem jiného bodu. Například pohne-li se bod k1, vypočítá se možný pohyb bodů k2, k3, k4, který tento vztah vrátí zpět na požadovanou hodnotu. Tedy každý pohyb vytvoří tři různá řešení. Uplatňuji zde systém priorit, definují které řešení se má obsloužit jak první. Pokud toto řešení selže (skončí v kolizním stavu) přechází se na řešení druhé atd. Zní to jednoduše, ale pohyb všech kotevních bodů ošetřuje algoritmus kontrola kolize vazeb. Při zamítnutí pohybu se složitě musí v rekurzi vracet a uplatňovat jiné větve výpočtu pro vyrovnání daného úhlu.

7.4. Kontrolní součet

Tento jednoduchý algoritmus je aplikován při ukládání obrázků na pevný disk. Na disk se nakreslené obrázky i jejich vzájemné vztahy ukládají ve formátu XML. XML je značkovací jazyk, který data ukládá do stromové struktury. To je pro účely této aplikace výrazné zjednodušení, bohužel XML je textový formát, který dokáže uživatel jednoduše změnit. Proto na celý řetězec XML je před uložením aplikován algoritmus, který celý tento řetězec rozloží na jednotlivé znaky a přidělí

jim číselnou hodnotu podle ascci. Tyto hodnoty sečte a výsledné číslo uloží do speciálního tagu k souboru XML. Ten pak celý uloží na disk. Při otevírání souboru se algoritmus provede naopak. Pokud kontrolní součty souhlasí, souboj je nezměněný a může se otevřít. Pokud ne, aplikace vypíše chybu a soubor se neotevře.

8. Převod do jiných formátů

8.1. Ukládání do SVG

Jedná se o souborový formát nazvaný Scalable Vector Graphics, neboli SVG, který je založený na značkovacím jazyku XML. Bohužel tento formát Microsoft nepodporuje a je nutno použít nějaké knihovny třetích stran. Sintax SVG je jednoduchá, proto tento textový formát vytvářím v aplikaci přímo bez nutnosti používat jakoukoli knihovnu. Stačí postupně procházet všechny grafické objety, které jsou nakresleny, brát si z nich potřebné hodnoty jako souřadnice kotevních bodů, typ vykresleného objektu (čára, kruh) nebo styl vykreslené čáry (šířka, barva). Pomocí těchto hodnot sestavit textový řetězec do požadovaného tvaru. Sintax řetězce nám přesně defiunje W3C (word widw web consorcium). Stačilo vymyslet vhodný algoritmus, který vytvoří tzv. PATH obsahující data pro vykreslení cesty dané čáry. Toto řešení je poměrně jednoduché a funguje spolehlivě.

Problém nastává pokud v aplikaci uživatel použije funkci průnik. Uživatel nakreslí Vennův diagram pomocí dvou vzájemně propojených kružnic. Toto propojení kružnic na monitoru zajišťuje grafická knihovna WPF. Aplikace vnitřně vidí stále dvě samostatné kružnice a nastavuje jenom jejich zobrazovací mód pomocí knihovny WPF. Proto i převod do SVG generuje dvě samostatné kružnice. V tomto jediném případě se export do SVG nezobrazí správě.

Řešením tohoto problému by bylo dopočítat souřadnice jejích průsečíků a definovat cestu čáry v SVG přes tyto body. Bohužel v této verzi aplikace, nejsem tohoto kroku schopen. Aplikace je navržena jen na zobrazování obrázků. Pro takový výpočet průsečíků jakýchkoli objektů by bylo zapotřebí předělat celé jádro aplikace. Potom by byla aplikace plnohodnotným vektorovým editorem a to by bylo nad rámec požadovaného zadání.

8.2. Ukládání do PDF

Jedná se souborový formát (Portable Document Format – Přenosný formát dokumentů) vytvořený firmou Adobe. Microsoft tento formát nepodporuje, má vlastní pod názvem XPS. Proto pro převod dokumentu do formátu PDF je

potřeba využít knihoven třetích stran.

K převodu obrázku do formátu PDF zde využívám jednu komerční knihovnu (NiPdf), ta funguje naprosto perfektně. Stačí do ni vložit nakreslený obrázek ve formátu XPS a zbytek práce udělá za vás. Tato knihovna v aplikaci funguje v demo verzi. Proto ve vygenerovaném PDF dokumentu se objeví červený nápis „TRIAL“. Tato knihovna je v aplikaci použita a její funkci si můžete ověřit v menu, položka převod do PDF(1). [2]

Samozřejmě existují knihovny, které jsou k dispozici zdarma např. ItexScharp nedo PDFnet. Bohužel jejich použití je složitější. Převod pomocí knihovny ItexScharp, kterou jsem v aplikaci také použil, neumí vytvoří PDF dokument pomocí transformace z jiného formátu. Ale nové PDF pomůže vývojáři vysázet pomocí příkazů např. writer.LineTo(x,y). Takže zde bylo potřeba vytvořit algoritmus pracující na principu zásobníkového automatu, který na vstupu má řetězec ve formátu PATH WPF, ten přečte a pomocí příkazů této knihovny vysází nový PDF dokument.

ItexScharp funguje dobře, ale bohužel výše zmíněnou uživatelskou funkci průnik dvou kružnic, také nedokáže interpretovat. Nedá se zde definovat ani složitější vzorkování čáry. ItexScharp podporuje jen vzorkování definované nanejvýš třemi čísly, proto některé složitější vzory čerchování zde nejsou možné. Drobné problémy byly i s interpretací souřadnic pro text a fonty a i zobrazování oblouků zde nefungovalo přesně. I přes napsání celkem složitého algoritmu pro rozložení textového řetězce z PATH WPF do funkci vykreslující PDF, jsem nakonec od exportu přes tuto knihovnu ustoupil a využívám jen ukládání pomocí bitmapy. Tato funkce je v aplikaci zastoupena v menu položka převod do PDF(2).

[2] Celý problém s C# a Windows je jeho komerčnost. Vývojáři mají možnost využívat opravdu spoustu užitečných knihoven a nástrojů psaních přímo pro Visual Studio. Vše ovšem za „drobný“ poplatek. Pokud máte dostatek finančních prostředků je psaní programů v C# poměrně snadná záležitost a s takovou podporou by to bylo i několikanásobně rychlejší a efektivnější.

9. Možnosti rozšíření aplikace

Téma matematické obrázky nabízí mnoho možností jak rozšířit a zlepšit tuto aplikaci. Zlepšit se dá jak uživatelské prostředí, tak i rozšíření grafických funkcí a odstranění některých uživatelských omezení. Další funkcí pro vylepšení aplikace je možnost připínat grafické objekty (kotevní body nebo čáry) ke mřížce. Ohledně lepší ovladatelnosti by měly být definované vztahy grafických objektů pro uživatele přístupné v jakémkoli čase. To znamená umožnit uživateli pohodlně a jednoduše tento vztah graficky znázornit a jakkoli ho měnit. Doplnit by se mohly i exporty do jiných formátů.

10. Motivace pro další práci

Od matematických obrázků je jen malý krůček k plnohodnotnému vektorovému editoru a potom i k aplikaci umožnující generování všech geometrických konstrukcí. Aby aplikace fungovala i jako plnohodnotná pomůcka, pro výuku geometrie, je potřeba přidat kótování čar a úhlů. Tím se vytvořil plnohodnotný CAD editor.

Téma CAD editorů mě zajímá. Přál bych si takovou aplikaci naprogramovat. Efektivní práce s grafikou na platformě Windows ovšem vyžaduje používat přímo knihovny DirectX (grafické knihovny firmy Microsoft). Tyto knihovny nám nabízejí mnoho dalších funkcí, jak efektivně a jednoduše řešit grafiku v našem vektorovém editoru např. výpočet průsečíků dvou přímek, úhel dvou přímek, rotace elipsy atd. Jedná se o knihovny, které nejsou objektové a jejich studium mi jistě zabere spousty času.

Po nabytých zkušenostech s touto prací a omezeních na které jsem ve Windows narazil, je třeba se zamyslet na jaké platformě další verzi aplikace implementovat. Teď se spíše přikládám k „Java“ a multiplatformní grafické knihovny OpenGL (Open Graphics Library).

Závěr

Po nabytých zkušenostech s touto prací se rád pustím do jejího dalšího rozvoje. Vytvořená aplikace v této fázi nabízí uživateli celou škálu funkcí a možnosti jak jednoduše kreslit matematické obrázky. Aplikace dává uživateli možnost dynamicky vytvářet jakýkoli matematický obrázek a simuluje základy geometrických konstrukcí, proto je v této oblasti zajisté přínosem a plně vyhovuje zadání. Beru na zřetel možnou změnu platformy pro implementaci.

Conclusions

After gaining experiences with this project I will be glad to start working on developing it further. At this phase, the created application offers a whole range of functions and possibilities for the user to easily draw mathematical images. The application gives the user the ability to dynamically create any mathematical image and it simulates basics of geometrical constructions, therefore it surely is an asset in this area and it fully complies with the assignment. I am aware of the possible change in the platform for implementation.

Reference

[1] Christian Nagel, Bill Evjen, Jay Glynn, Karli Watson, Morgan Skinner *C# Programujeme profesionálně* Computer Pres, 2009.

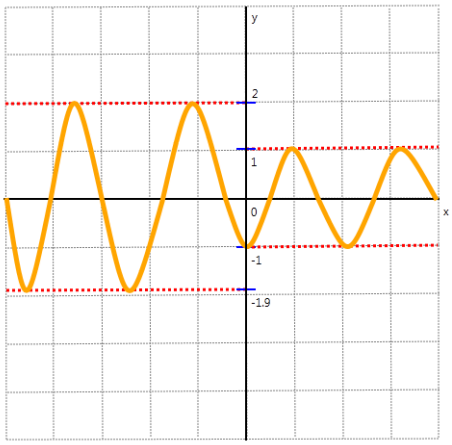
[2] Charles Petzold. *Mistrovství ve WPF* Computer Pres, 2010.

[3] Bruno Lowagie *iText in Action* Manning Publications, 2010.

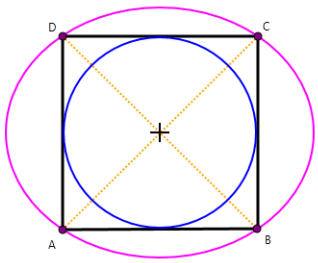
[4] Amandeo Mareš. *1001 tipů a triků pro C#* Computer Pres, 2008.

[5] Wikipedia. *The Free Encyklopedia*

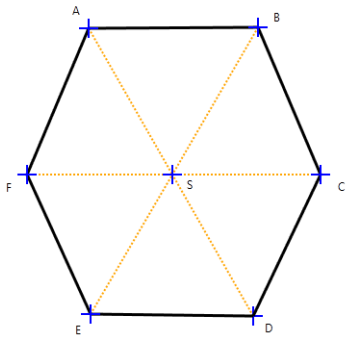
A. Obrázky vytvořené aplikací Matematické obrázky



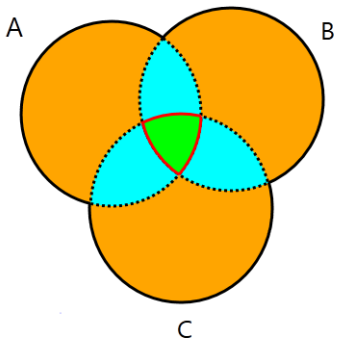
Obrázek 28. Funkce X



Obrázek 29. Kružnice opsaná



Obrázek 30. Šestiúhelník



Obrázek 31. Průnik množin

B. Obsah přiloženého CD

`bin/`
: Instalátor aplikace Matematické obrázky spustitelné přímo z CD.

`doc/`
: Dokumentace práce ve formátu PDF, vytvořená dle závazného stylu KI PřF pro diplomové práce, včetně všech příloh, a všechny soubory nutné pro bezproblémové vygenerování PDF souboru dokumentace (v ZIP archivu), tj. zdrojový text dokumentace, vložené obrázky, apod.

`src/`
: Kompletní zdrojové texty programu Matematické obrázky

`readme.txt`
: Instrukce pro instalaci a spuštění programu Matematické obrázky, včetně požadavků pro jeho provoz.