UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Modul pro sazbu dokumentů do PDF

Martin Baudys

Bakalářská práce
2015

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2014/2015**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Martin Baudys**
Osobní číslo:	**I10008**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Modul pro sazbu dokumentů do PDF**
Zadávající katedra:	**Katedra informačních technologií**

Zásady pro vypracování:

V teoretické části bude provedena rešerše stávajících nástrojů pro sazbu dokumentů. Tyto nástroje budou porovnány. Dále budou popsány výhody použití XML transformací.
V praktické části bude vytvořena programová knihovna, která na základě vstupního XML dokumentu vytvoří PDF dokument odpovídající šabloně. Hlavní cíl práce bude zaměřen na správnou implementaci již existujících postupů pro převod do PDF. Součástí bude také jednoduchý nástroj pro vytváření těchto šablon.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

NAGEL, Christian. C# 2005: programujeme profesionálně. Vyd. 1. Překlad Jakub Mikulaštík, Petr Dokoupil. Brno: Computer Press, 2006, 1398 s. ISBN 80-251-1181-4.
MAREŠ, Amadeo. 1001 tipů a triků pro C# 2010. Brno: Computer Press, 2011, 416 s. ISBN 978-80-251-3250-0.
msdn.microsoft.com/en-us/library

Vedoucí bakalářské práce: **Ing. Jiří Zechmeister**
Katedra informačních technologií

Datum zadání bakalářské práce: **20. prosince 2014**
Termín odevzdání bakalářské práce: **11. května 2015**

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

Ing. Lukáš Čegan, Ph.D.
vedoucí katedry

V Pardubicích dne 31. března 2015

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 7. 5. 2015 Martin Baudys

Poděkování

Chtěl bych tímto poděkovat panu Ing. Jiřímu Zechmeisterovi za vedení této bakalářské práce a rodině za psychickou a finanční podporu při studiu.

Anotace

Práce se zabývá sazbou dokumentů a převodem dokumentu XML do velice populárního a široce používaného formátu PDF. Obsahem teoretické části je rešerše již existujících nástrojů, pro sazbu dokumentů. Dále jsou popsány základní informace o technologiích, které se hojně využívají pro XML transformace a jejich použití. V této části je také popsán princip transformací a jejich výhody. V praktické části je vysvětlena struktura knihovny dll, naprogramovaná pro samotnou transformaci, popis procesoru FO a v neposlední řadě je vysvětleno použití nástroje pro vytváření šablon XSLT.

Klíčová slova

sazba dokumentů, transformace, XML, PDF, šablona, .NET

Title

Module for converting documents to PDF.

Annotation

The Bachelor Dissertation’s main objective is desktop publishing and transforming XML documents into widely used PDF format. In theoretic part you can find the research of already existing tools for desktop publishing. The next part is about basic technological information, which are used for XML transformation and their usage. In this part you can also find the principle of transformation and it’s advantage. In the practical part there is explained structure of dll library, description of FO processor and at last but not least there is explanation of tools for creating XSLT templates.

Keywords

desktop publishing, transformation, XML, PDF, template, .NET

Obsah

Seznam zkratek

XML	eXtensible Markup Language
PDF	Portable Document Format
XSLT	eXtensible Stylesheet Language Transformations
CSS	Cascading Style Sheets
HTML	HyperText Transfer Protokol
.NET	[dotnet]
DTP	desktop publishing
ISO	International Organization for Standardization
PSD	Photoshop Document
CMYK	Cyan Magenta Yellow blacK
WYSIWYG	What you see is what you get
W3C	World Wide Web Consortium
DOM	Document Object Model
SAX	Simple API for XML
XPath	XML Path Language
PDA	Personal Digital Assistant

Seznam obrázků

Seznam tabulek

Úvod

V informačních technologiích stále častěji narážíme na strukturované informace v podobě dokumentů XML. Tyto dokumenty mohou obsahovat data téměř libovolného typu, a proto vývojáři stále častěji tento formát upřednostňují pro přenos a výměnu dat ve svých aplikacích. Kromě nepřeberného množství výhod mají dokumenty XML také jednu zásadní nevýhodu – jejich obsah je poměrně nečitelný. Lidstvo stále s velkou oblibou využívá tištěného výstupu. Není tedy divu, že existuje mnoho programů, které dokáží dokumenty XML formátovat do čitelné podoby pro tisk. Existují ale také technologie, které umožňují převod XML dat do jiných formátů.

Cílem této bakalářské práce je vytvořit programovou knihovnu, která na základě vstupního XML dokumentu vytvoří PDF dokument odpovídající šabloně. Velice obecně řečeno, je touto šablonou myšlen soubor a souhrn pravidel, podle kterých jsme schopni sestavit nejen dokumenty PDF, ale například HTML stránku, nebo XML dokument s úplně odlišnou strukturou.

Praktickou část tvoří samotná knihovna (dll) a jednoduchý nástroj, který tuto knihovnu využívá pro XML transformace a pomocí něhož máme možnost vytvářet výše zmíněné šablony, aniž bychom měli jakoukoli představu o tom, co je jejich obsahem. V tomto nástroji by mělo být uživateli umožněno definovat styl sazby jednotlivých prvků umístěných do dokumentu. Mezi atributy definovaných stylů bude patřit například odsazení textového bloku od okrajů stránky, název fontu písma, zarovnání písma, jeho barevné podání, velikost, nebo třeba dekorace písma.

Programová knihovna i modul pro sazbu dokumentů jsou vytvořeny na platformě .NET ve vývojovém prostředí Visual Studio 2013. Celý zdrojový kód aplikace je napsán v jazyce C#, proto jakákoliv ukázka zdrojového kódu ponese syntaxi tohoto moderního programovacího jazyka.

Než začneme řešit podrobněji, na jakém principu funguje převod z XML do PDF a jaké využívá technologie, bude vhodné nejprve vysvětlit, co přesně si představujeme pod pojmem sazba dokumentů. Začněme tedy krátkou teorií a představme si několik již existujících nástrojů, které se touto problematikou zabývají.

1 Sazba dokumentů

1.1 O sazbě

Sazba dokumentů spočívá ve vkládání textu a případně grafických prvků, jako jsou obrázky, na virtuální stránku, kterou interpretuje naše obrazovka. Jedná se tedy především o tvorbu tištěných dokumentů pomocí počítače. Anglicky se této metodě říká Desktop Publishing (zkráceně DTP). DTP nástroje dovolují nastavovat rozvržení stránky a pracovat s typografickou úpravou textu a ilustrací dokumentu.

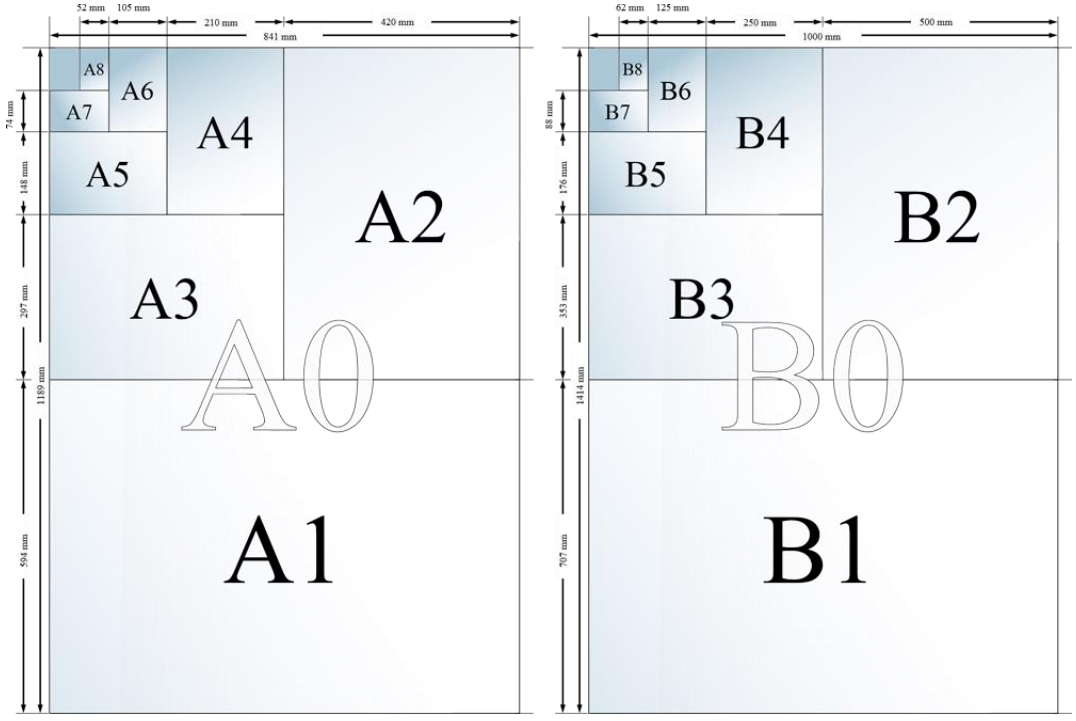
Tato technologie umožňuje vytvářet širokou škálu materiálů, jako jsou tištěné reklamní materiály, letáky, upoutávky, časopisy, knihy a spoustu dalších. Osoba, která pracuje se software pro sazbu dokumentů a vytváří tyto materiály, je tzv. DTP operátor.

1.2 Pravidla sazby

Stejně jako je morálně nepřístupné publikovat texty s pravopisnými chybami, je stejně nepřístupné publikovat texty s vážnými typografickými chybami. Proto existují jistá pravidla pro sazbu dokumentů. O těchto pravidlech by se dalo napsat mnoho, nicméně tomuto tématu se v této práci věnovat nebudeme.

1.3 Formáty tiskovin

Protože se technologie DTP soustředí na přípravu dokumentů pro tisk, nabízejí nám DTP nástroje při vytváření nového dokumentu standardní formáty papírů. Existuje několik řad těchto formátů. Mezi nejznámější patří řady A, B a obálkový formát C.[1]



Obrázek 1- Standardní formáty papírů[1]

Následující tabulka zobrazuje šířku a výšku všech řad papírů ISO A a B včetně obálkové řady C. Rozměry jsou uvedeny v milimetrech.[1]

Tabulka 1- Rozměry standardních formátů papíru

Formát řady A	
A0	841×1189
A1	594×841
A2	420×594
A3	297×420
A4	210×297
A5	148×210
A6	105×148
A7	74×105
A8	52×74
A9	37×52
A10	26×37

Formát řady B	
B0	1000×1414
B1	707×1000
B2	500×707
B3	353×500
B4	250×353
B5	176×250
B6	125×176
B7	88×125
B8	62×88
B9	44×62
B10	31×44

Formát řady C	
C0	917×1297
C1	648×917
C2	458×648
C3	324×458
C4	229×324
C5	162×229
C6	114×162
C7	81×114
C8	57×81
C9	40×57
C10	28×40

1.4 Nástroje pro sazbu dokumentů

Mezi nejvíce rozšířené DTP programy patří například Adobe InDesign, Adobe Photoshop, Corel Draw, Microsoft Publisher a spoustu dalších komerčních i svobodných programů.

Do souboru programů pro sazbu dokumentů můžeme zařadit i textové procesory, jako například Microsoft Word, nebo jeho volně šiřitelnou podobu od vývojářů kancelářského balíčku LibreOffice, kde jej najdeme pod názvem Writer. U textových procesorů je kladen důraz na snadné a intuitivní ovládání.

1.4.1 Adobe InDesign

1.4.1.1 Společnost Adobe

Společnost Adobe je již několik desetiletí vůdčí společností na trhu grafických programů nejen pro tisk. Vyvíjí nejnovější generace softwaru pro úpravy fotografií, videa, tvorbu technických výkresů, sazbu jednostranných i mnohostranných dokumentů atd.

Tato společnost byla založena roku 1982 Johnem Warnockem a Charlesem Geschkem krátce poté, co opustili Xerox Parc za účelem vývoje postscriptovacího jazyka. Firma je založená na oblast počítačové grafiky, publikování a předtiskové přípravy a je známá především jako autor standardů PostScript a PDF. Mezi nejpopulárnější software od této společnosti patří Adobe Acrobat, Adobe Photoshop, Adobe Illustrator, Adobe InDesign a další.[2]

1.4.1.2 Popis programu

Jedná se o profesionální komerční nástroj, který umožňuje navrhovat a připravovat rozvržení stránek pro tisk, digitální distribuci pomocí integrovaných nástrojů a řízení typografie. Aplikace umí propojit obsah z více dokumentů a díky tomu se pak provedené změny projeví ve všech. Pomocí typografických nástrojů lze vytvářet knihy pro elektronické čtečky, protože Adobe InDesign dokáže výsledný dokument exportovat do PDF či EPUB (electronic publication).

Adobe InDesign jako jeden z mála programů tohoto typu dokáže vytvářet interaktivní publikace pro tablety, včetně vkládání zvuku, nebo videa do dokumentu. Samozřejmostí je podpora jak vektorové, tak rastrové grafiky. Díky vektorové grafice se nesetkáváme s problémy při změně velikostí objektů.

Pro pohodlnou a rychlou profesionální práci s textem je k dispozici rozsáhlá možnost podrobného stylování textu, jako například jeho řez, velikost fontu nebo řádkový proklad. Při využití moderních písem je výhodou podpora technologie OpenType[1].

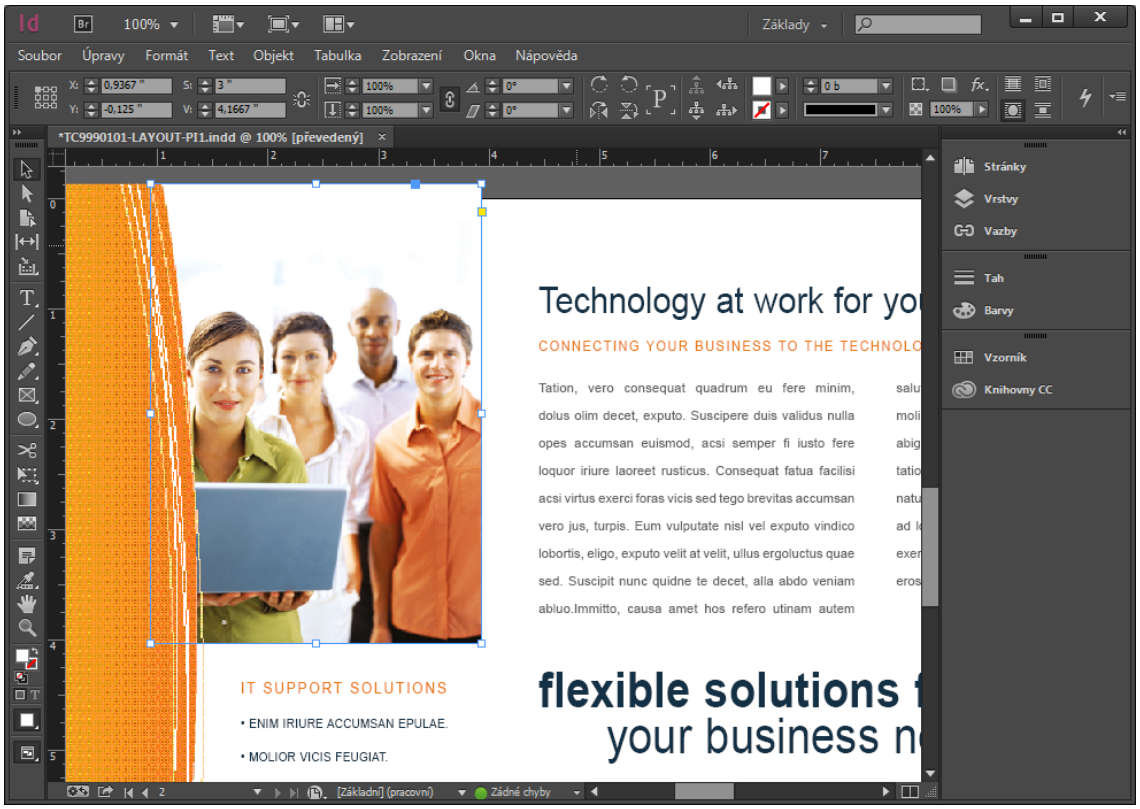
Společnost Adobe je tvůrcem formátu PDF a proto InDesign dokáže pracovat i s tímto formátem. Na vstupu InDesign bez problémů přijme korektní PDF a umožní je vložit do stránky. To velice ulehčí práci například při výrobě tiskovin, obsahujících externě připravenou inzerci. PDF, které takto chceme vložit do připravovaného dokumentu, musí

[1] OpenType je standard pro popis vektorových počítačových písem, vyvinutý společností Microsoft jako nástupce standardu TrueType.

nést všechny důležité informace o obsahu. Těmito informacemi jsou například přímé barvy a použité texty. Pokud by tyto informace nebyli k dispozici, mohli bychom do vytvářeného dokumentu zanést chyby.

PDF formát lze samozřejmě použít i pro výstup z aplikace a není tedy problém výsledek pohodlně vytisknout na velkoformátové tiskárně.

Adobe InDesign podporuje i formát PSD. Rastrová grafika, která je navíc uchovávána ve vrstvách tedy nepředstavuje žádný problém. Provázanost produktů společnosti Adobe je velkou výhodou v případě, že pracujeme například v kolektivu. Grafik tak může vypracovat obsah, který následně uloží jako PSD, se kterým nemá DTP operátor při sazbě žádné problémy.



Obrázek 2- Adobe InDesign, návrh dokumentu

Adobe InDesign je určen zejména pro pokročilejší uživatele a proto jeho ovládání není nikterak intuitivní, nicméně vzhledem k jeho popularitě se na internetu pohybuje dostatečné množství návodů a především šablon. Profesionální šablony je možné zakoupit, nebo lze stáhnout volně šiřitelné. Pro běžného uživatele pak odpadá starost základního nastavení rozvržení dokumentu, formátování textu a postačí, když do šablony vloží požadovaný obsah.

Co běžného uživatele ovšem nepotěší, je cena tohoto produktu. V současnosti se cena pro jednotlivce pohybuje okolo 290 €/rok. Společnost Adobe nabízí zkušební verze aplikací. Po nainstalování této verze, bude aplikace plně funkční po dobu 30 dní. Pokud budeme chtít produkt používat i nadále, je potřeba zakoupit licenci.[3][4]

1.4.1.3 Adobe InDesign, zhodnocení aplikace

1.4.1.3.1 Výhody

- Podpora PDF formátu pro vstup i výstup,
- podpora PSD souborů,
- možnost importovat data z XML,
- možnost importovat šablonu XSLT k dokumentu XML,
- možnost tvorby velkých projektů, například knihy apod.,
- tvorba projektů s multimediálním obsahem.

1.4.1.3.2 Nevýhody

- Vysoká cena,
- složité ovládání pro začátečníky.

1.4.2 Corel Draw Graphics Suite X7

1.4.2.1 Popis programu

Stejně jako Adobe InDesign, je aplikace Corel Draw grafický editor, navržen pro práci s vektorovými objekty, ale dokáže pracovat i s rastrovou grafikou. Za vývojem stojí firma Corel Corporation. Mnozí uživatelé stavějí produkty Adobe a Corel Corporation proti sobě. Je těžké porovnávat dva profesionální nástroje, které nabízejí téměř stejné možnosti. Opravdové výhody a nevýhody poznají skuteční profesionálové, kteří s grafikou pracují dnes a denně.

I tento nástroj umožňuje sazbu dokumentů, to znamená tvorbu letáků, vizitek, webových prezentací a dalších reklamních materiálů. Pro méně zkušené uživatele jsou opět k dispozici šablony, které jsou implementovány již v základu tohoto programu. Jedná se však o jednoduché projekty. Větší a složitější šablony je nutno zakoupit.

Oproti Adobe InDesignu je prostředí na první pohled přívětivější a svůj vzhled nechává postaven na světlých barvách.

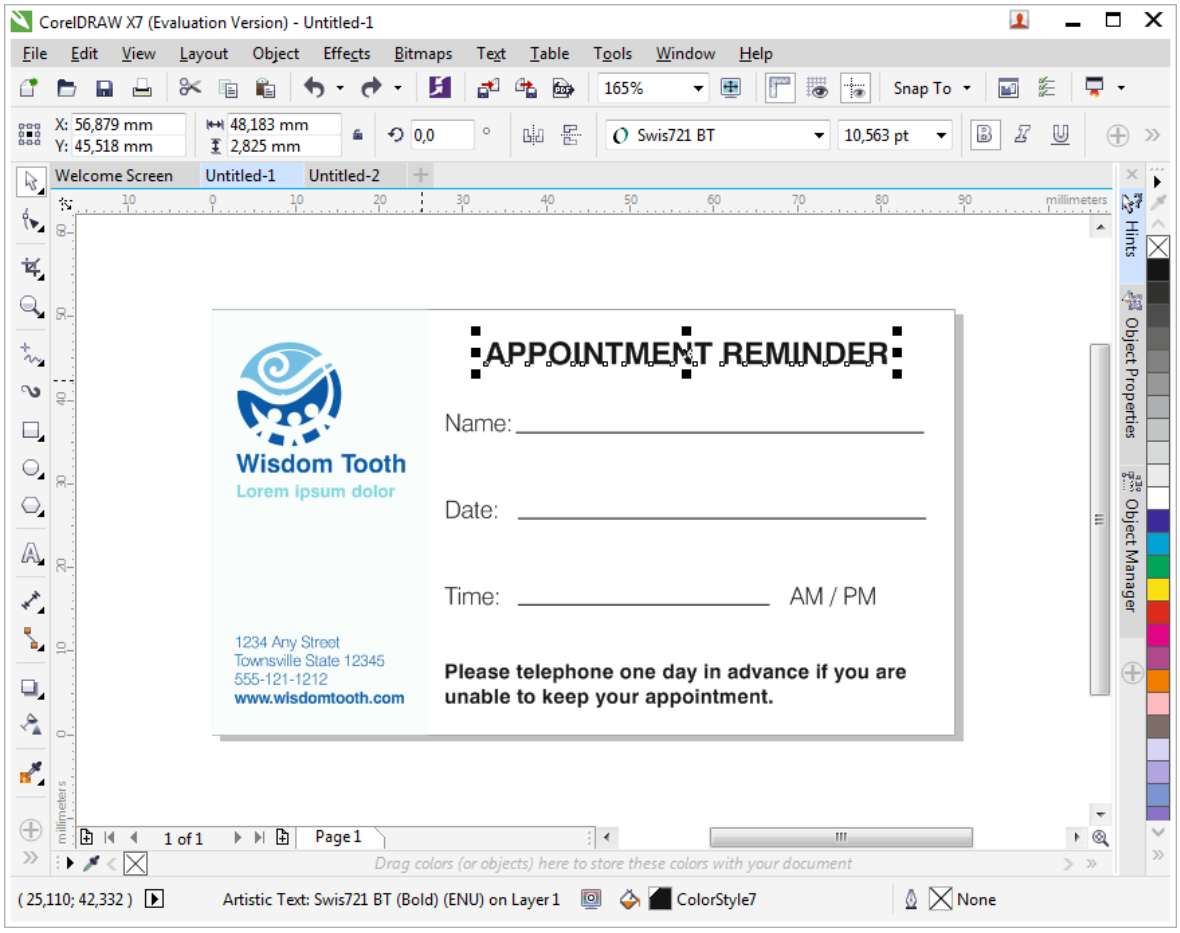
Corel Draw umožňuje import a export velkého množství typů souborů a připojení nejrůznějších periferních zařízení, jako jsou Trackbally, grafické tablety apod.

Balíček Corel Draw Graphics Suite X7 neobsahuje pouze program Corel Draw, ale nabízí i následující nástroje:

- CorelDRAW – aplikace pro intuitivní vektorové ilustrace a stránkový zlom,
- Corel PHOTO-PAINT – bitmapový grafický editor,
- Corel PowerTRACE – modul pro převod rastrů na vektorovou grafiku,
- Corel CONNECT – program pro vyhledávání obsahu,
- Corel CAPTURE – program pro zachytávání obrazovky,
- Corel Website Creator – aplikace pro návrh webových stránek,

- PhotoZoom Pro – modul plug-in pro zvětšení digitálních obrázků,
- ConceptShare – nástroj pro spolupráci online.

Celý tento balíček nástrojů je možné po dobu 30 dnů používat bez jakýchkoliv omezení. Jelikož se jedná o shareware licenci, je po uplynutí třicetidenní zkušební lhůty nutné zakoupit licenci a produkt zaregistrovat. Corel Corporation nabízí také omezené verze pro domácí a studentské účely. Tyto licence slouží pouze pro nekomerční použití.[5]



Obrázek 3- Corel Draw, návrh dokumentu

1.4.2.2 Corel Draw, zhodnocení aplikace

1.4.2.2.1 Výhody

- Přijatelná cena, licenci pro domácí a studijní účely je možné pořídit zhruba za 2000 KČ,
- poměrně jednoduché ovládání,
- možnost importovat PSD soubory,
- multiplatformní s výjimkou Linuxových distribucí.

1.4.2.2.2 Nevýhody

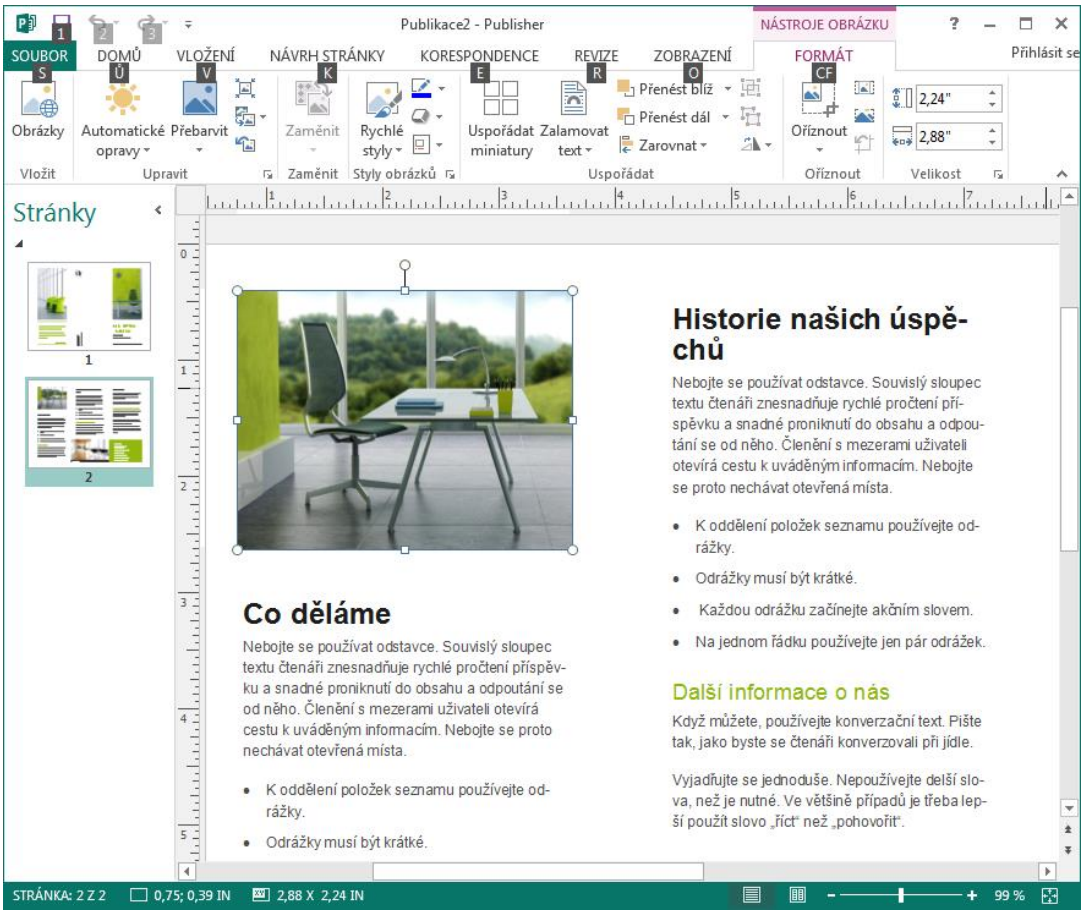
- Neumí importovat data z XML
- Nemá možnost pracovat s multimediálním obsahem

1.4.3 Microsoft Publisher

1.4.3.1 Popis programu

Microsoft Publisher 2013 je nástroj na tvorbu publikací v prostředí balíku MS Office, se kterým Publisher sdílí celou řadu funkcí včetně korektury pravopisu, formátování textů či manipulace s grafickými objekty. Na rozdíl od MS Wordu je Publisher koncipovaný přímo na sestavování různých druhů publikací, bookletů, brožur, zpravodajů a marketingových materiálů, které lze v layout editoru Publisheru rychle skládat prostřednictvím mnoha desítek offline i online dostupných šablon.

Některé šablony Publisheru jsou již součástí instalace příslušné edice MS Office, řada dalších šablon je pak k dispozici online stejně jako tisíce klipartů a dalších speciálních grafických prvků. Můžeme tak pohodlně a rychle vytvářet například kalendáře, štítky, vizitky, diplomy, nabídkové katalogy, oznámení, obálky, reklamy, životopisy, papírové skládanky či obchodní formuláře.



Obrázek 4- Misrosoft Publisher, návrh dokumentu

Stejně jako profesionální nástroje, jako je Adobe InDesign, nebo Corel Draw, disponuje Publisher WYSIWYG[2] prostředím editoru. WYSIWYG editor layoutu stránek je v Publisheru prakticky stejný jako v aplikaci MS Wordu, má však řadu dalších vlastností navíc zaměřených především na práci s objekty a jejich umísťování na stránce. K dispozici máme pokročilé nástroje na práci a úpravy obrázků, které lze libovolně posunovat, zvětšovat, ořezávat a nahrazovat zástupnými symboly.[6]

Microsoft Publisher sice není freewarový software, ale je cenově mnohem dostupnější, než profesionální DTP programy. Návrh publikovaných dokumentů je výrazně jednodušší a tedy i rychlejší. Přesto s ním dovedeme vytvářet plnohodnotné, graficky přívětivé materiály pro kvalitní tisk či online publikaci.

1.4.3.2 Microsoft Publisher, zhodnocení aplikace

1.4.3.2.1 Výhody

- Obsažen v balíku MS Office,
- jednoduché ovládání,
- nízká pořizovací cena.

1.4.3.2.2 Nevýhody

- V základu neumí importovat PDF do dokumentu,
- určen pouze pro platformu Microsoft Windows.

1.4.4 Scribus 1.4.5

Scribus je další program určený pro sazbu dokumentů. Šířen je zdarma pod licencí GNU GPL (General Public Licence). Původně byl tento program vytvořen pro operační systém Linux a později byly vytvořeny jeho verze také pro MS Widnows, Mac OS a další.

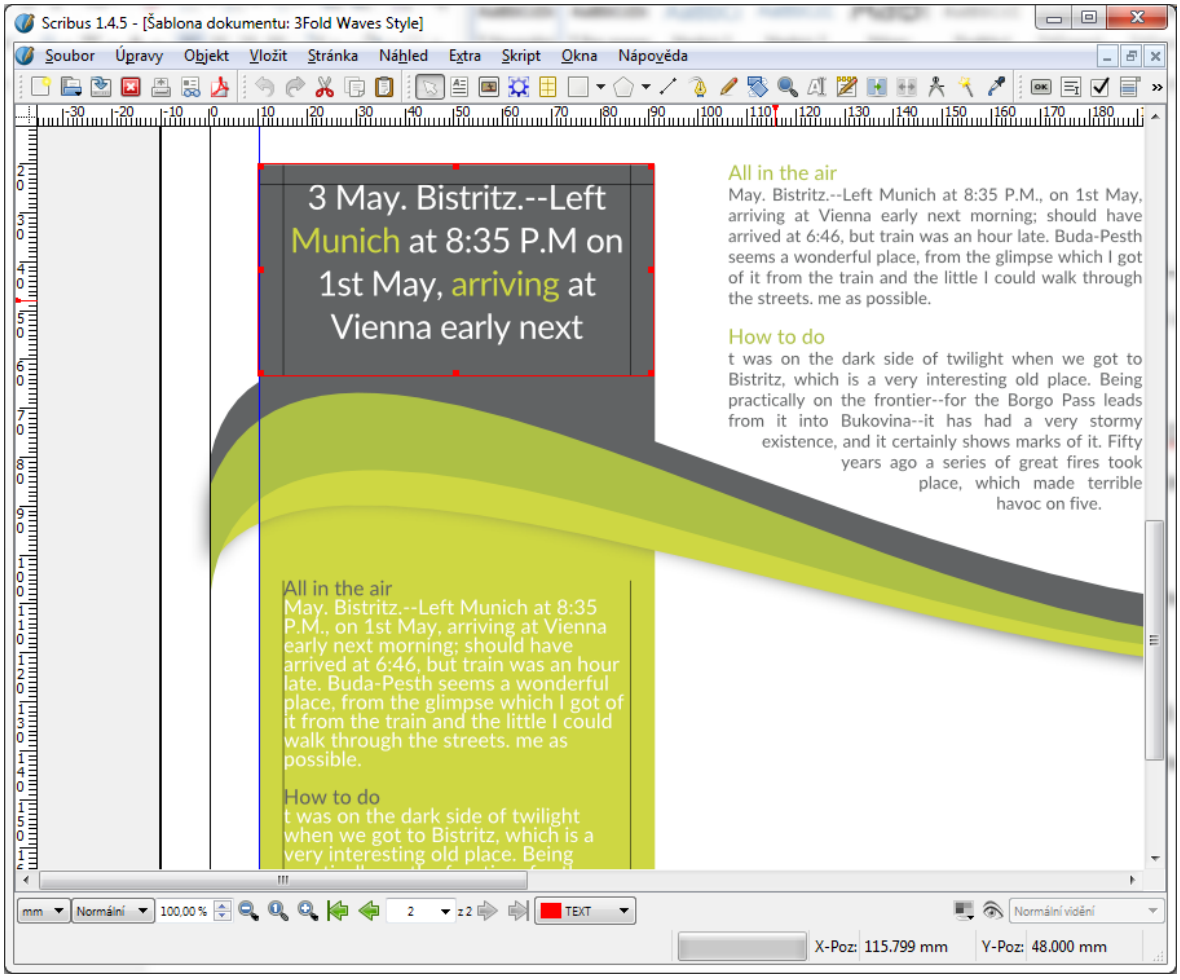
Nabízí podporu většiny textových i grafických formátů, včetně SVG, TrueType a OpenType fontů, CMYK a ICC správu barev, export do PDF, skripty například pro automatickou tvorbu kalendářů a mnoho dalšího.

Na oficiálních stránkách programu je k dispozici rozsáhlá dokumentace a najdeme tam i spoustu výukového materiálu.

Pokud bychom Scribus porovnávali s profesionálními nástroji, je na první pohled patrné, že se jedná o lacinější verzi DTP programu. Vzhled a rozložení aplikace uživatele příliš nezaujme. I přesto se jedná o velmi kvalitní nástroj, ve kterém máme možnost vytvářet působivé dokumenty.

I Scribus nabízí v základu připravené šablony, které ušetří čas při návrhu stránky.[7]

[2] WYSIWYG je zkratka anglického spojení „What you see is what you get“, v překladu doslova „Co vidíš, to dostaneš“. Vysvětleno dále.



Obrázek 5- Sribus, návrh dokumentu

1.4.4.1 Scribus 1.4.5, zhodnocení aplikace

1.4.4.1.1 Výhody

- Zdarma stažitelný,
- kompletně česká lokalizace,
- podpora scriptů Python.

1.4.4.1.2 Nevýhody

- Absence panelů při otevření více souborů

1.5 Zhodnocení nástrojů pro sazbu dokumentů

Všechny zmíněně nástroje pracují v tzv. WYSIWYG módu, což je zkratka, která označuje způsob editace dokumentu v počítači, při kterém je verze zobrazená na obrazovce vzhledově totožná s výslednou podobou.

Na poli sázecích programů je aplikace Adobe InDesign jistě na prvních místech. Jedná se však o profesionální nástroj, který není snadno dostupný všem uživatelům. Proto jej

využívají především profesionální studia a grafici. Corel Draw je již cenově přijatelnější a také velice schopný. Hodí se především pro domácí účely, podobně jako Microsoft Publisher. Pro začátečníky, kteří mají chuť naučit se technologii DTP je velice vhodný nástroj Scribus, který je zcela zdarma.

Všechny tyto aplikace zvládají sazbu na vysoké úrovni a neustále se vyvíjí, takže přicházejí nová vylepšení, která by měla usnadňovat práci a nabídnout nové možnosti při sazbě dokumentů. Existují další alternativy výše zmíněných nástrojů, toto byly pouze ty nejznámější a nejpoužívanější.

2 XML transformace

2.1 XML

EXtensible Markup Language, je jednoduchý a velmi flexibilní značkovací jazyk, který má velice široké uplatnění. Málo kdo se s tímto typem souborů ještě nesetkal, mnohé ale možná překvapí, že historie vývoje XML sahá až do šedesátých let minulého století. První stavební kamínek položila firma IBM. Ta v tehdejších časech usilovala o vytvoření formátu, který by usnadnil ukládání velkého množství právních dokumentů. Výsledkem byl jakýsi obecný značkovací jazyk, který se několik let dále rozšiřoval a v roce 1986 se z něj stal jazyk SQML (Standard Generalized Markup Language). Největším problémem SQML však byla přílišná složitost, jednalo se ale o dobrý základ. Na tomto základu postupně vznikal jazyk HTML pro tvorbu webových stránek. Nicméně se časem ukázalo, že web potřebuje novou technologii, která bude mnohem flexibilnější než HTML a tak v roce 1998 vzniká jazyk XML, který přináší možnost tvorby dokumentů s vlastními tagy. Dnes je XML standardizován a specifikován konsorciem W3C.[8]

XML hraje stále významnější roli při výměně nejrůznějších údajů na webu, ale využívá se například i pro výměnu dat mezi aplikacemi, pro specifikaci konfiguračních souborů aplikací apod. Jazyk XML se rozšířil i do oblasti strojírenství, kde slouží k exportu informací z CNC strojů apod. S nástupem digitálních technologií ve zvukové technice, jako jsou digitální mixážní pulty, našel tento jazyk také své uplatnění.

2.1.1 Syntaxe jazyka XML

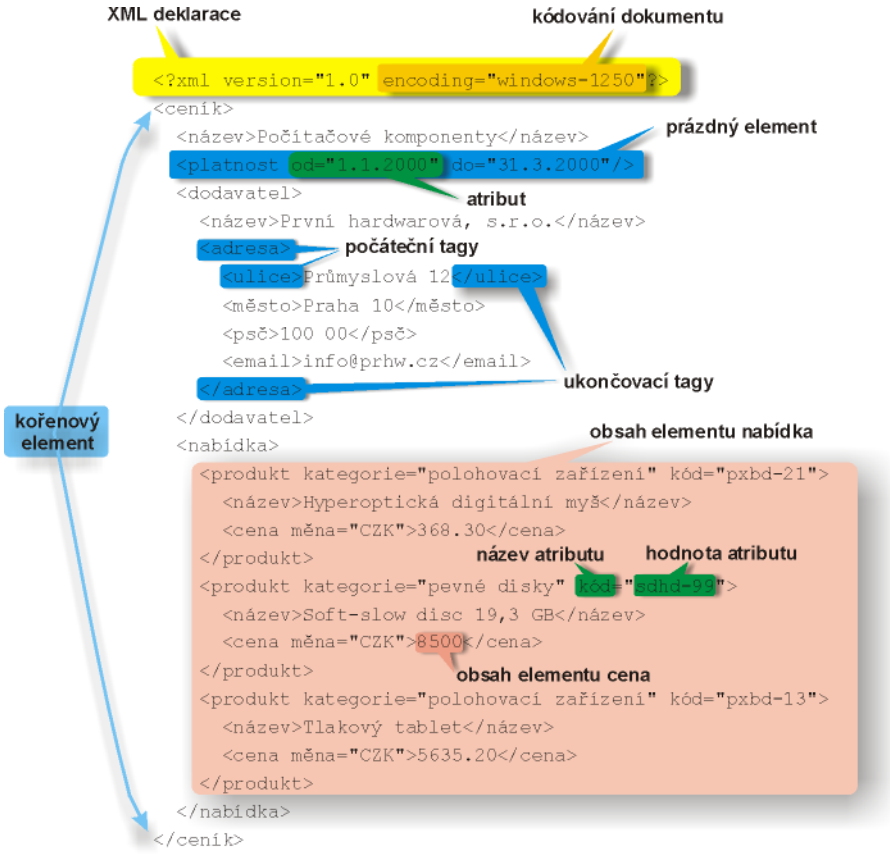
```
<?xml version="1.0" encoding="utf-8"?>
   <faktura>
      <cislo-faktury>20151901</cislo-faktury>
         <dodavatel>
            <jmeno>Petr</jmeno>
            <prijmeni>Novák</prijmeni>
            <adresa>Ulice 111, Mesto 123 45</adresa>
            <ico>12345678</ico>
            <mail>petr.novak@mail.xy</mail>
         </dodavatel>
         ...
   </faktura>
```

Ukázka XML dokumentu 1

Dokument XML je textový soubor a proto je možné jej napsat v jakémkoliv textovém editoru. Jednotlivé části dokumentu se označují pomocí značek. V terminologii jazyka XML se jednotlivým označeným částem dokumentu říká elementy. Elementy do sebe mohou být navzájem vnořené a tím dle potřeby zachycovat strukturu informací uložených v dokumentu. Elementy jsou základním stavebním kamenem každého dokumentu. U každého počátečního tagu můžeme použít ještě atributy. Atributy se používají k upřesnění významu elementu, nebo k přidání dalších důležitých informací.

Aby byl dokument správně strukturovaný, musí být celý obsah XML „obalen“ do kořenového elementu. Bez tohoto elementu není dokument validní a aplikace, které se snažíme předat takto nevalidní XML dokument, nebude pravděpodobně chtít s naším souborem pracovat.

Programy, které kontrolují správnost dokumentů XML, se nazývají parsery.[9]



Obrázek 6- Struktura dokumentu XML[9]

2.1.2 XML parsery

Parser je program, který kontroluje syntaktickou správnost XML dokumentu. Existují parsery, které zároveň kontrolují, zda dokument odpovídá schématu DTD, pokud je k dispozici. Dnešní internetové prohlížeče disponují také vlastním parserem, pochopitelně v případě, že podporují XML.

Parsery v podobě knihoven s oblibou využívají programátoři při načítání XML dokumentu do svých aplikací. Tuto práci velice zjednodušují, protože není nutné psát sáhodlouhý kód pro ošetření chyb. Aby byl život vývojářů co nejjednodušší, existují standardizovaná API rozhraní pro práci s XML dokumenty.

2.1.2.1 DOM

Rozhraní DOM (Document Object Model) reprezentuje XML dokument ve stromové struktuře. Každý element představuje jeden uzel stromu. Toto rozhraní obsahuje potřebné funkce k procházení dokumentu, k modifikacím jednotlivých uzlů a funkce pro mazání a přidávání nových uzlů. V dokumentu se navíc můžeme pohybovat dle vlastních potřeb a není nutné jej procházet celý. Rozhraní DOM najde svoje uplatnění tam, kde je potřeba provádět složitější operace s dokumentem XML.

2.1.2.2 SAX

Rozhraní SAX (Simple API for XML) umožňuje sériový přístup k XML. Je založeno na řízení pomocí událostí. Programátor musí nadefinovat funkce, které se volají v okamžiku, kdy parser narazí na začátek elementu, jeho obsah, konec elementu atp. Funkci jsou pak předány všechny potřebné parametry elementu, který se zpracovává.

Oproti rozhraní DOM, SAX nepotřebuje nahrávat celý dokument do paměti, to znamená, že pokud nepotřebujeme využívat funkčnost DOMu, je vhodné použít rozhraní SAX. Naše aplikace tak bude pravděpodobně pracovat rychleji. Nevýhodou tohoto rozhraní je, že je nutné dokument procházet od začátku do konce.[10]

2.1.3 Jmenné prostory

Bývá dnes dobrým zvykem přiřadit každému nově vytvořenému značkovacímu jazyku jeho vlastní jmenný prostor, aby jej šlo dobře identifikovat. Jmenný prostor je určen URI identifikátorem, který by měl být v celosvětovém měřítku jedinečný. Například všechny instrukce XSLT procesoru patří do společného jmenného prostoru http://www.w3.org/1999/XSL/Transform, nebo všechny XHTML elementy patří do prostoru http://www.w3.org/1999/xhtml.

Vždy deklarujeme jmennému prostoru nějaký prefix, který potom uvedeme před samotný název elementu nebo atributu. Každý název má totiž prefix, dokonce, i když není vidět. V takovém případě totiž značka používá prefix, který má hodnotu prázdného řetězce.[11]

Tabulka 2- Definice některých jmenných prostorů

Prefix	Definice
xsl	<xsl:stylesheet xmlns:xsl=”http://www.w3.org/1999/XSL/Transform”>
fo	<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">

2.1.4 Prezentace XML souboru

Značky dokumentu XML vyznačují pouze význam jeho jednotlivých částí. Když se vrátíme k naší ukázce, vidíme, že dokument obsahuje značky, které říkají toto je číslo faktury, toto je jméno dodavatele a tohle zase jeho emailová adresa. Jsou to prostá data, která v sobě nenesou informace o tom, jak se mají zobrazit na obrazovce počítače, nebo vytisknout na naší tiskárně. Abychom dostali XML dokument do čitelné podoby, musíme zobrazovacímu zařízení sdělit, jak má data interpretovat. K tomu slouží jazyk pro definování stylů XML, tzv. XSLT.[12]

2.2 XSLT

Základní myšlenkou, na které staví jazyk XML, je důsledné oddělení obsahu dokumentu od jeho vzhledu. Proto bylo zapotřebí vytvořit jazyk, který by vhodně definoval vzhledu jednotlivých elementů. Pro stylování webových prezentací je možné použít stylový jazyk CSS, nicméně tento jazyk na XML nestačí. Proto byl vyvinut jazyk XSL, který od svého staršího bratříčka CSS podědil spoustu vlastností. XSL tedy umožňovalo definovat styl pro zarovnání elementů, velikost a styl písma, barvy apod.

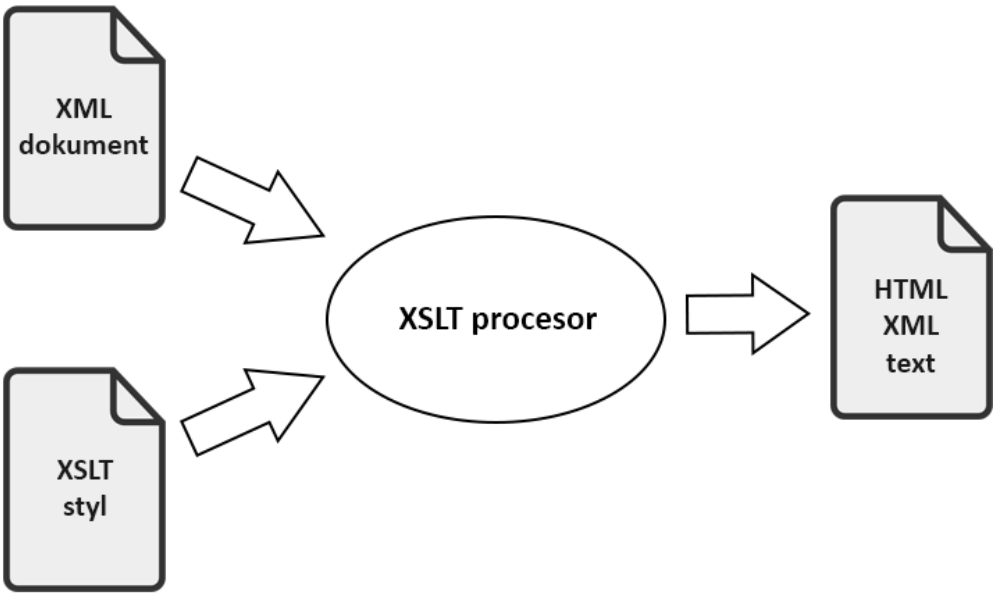
Během vývoje XSL se postupně ukázalo, že by mělo sloužit ke dvěma poměrně odlišným věcem. Tou první věcí je transformace XML dokumentů a druhou je definice vzhledu těchto transformací. Proto se dnes tato technologie jmenuje XSL Transform, zkráceně XSLT (EXtensible Stylesheet Language Transform).[12]

2.2.1 Transformace

XSLT nám dovoluje vytvářet styly, které definují, jak se má dokument XML převést do jiného formátu, například do HTML, do XML s odlišnou strukturou, nebo zkrátka do obyčejného textového souboru. Zejména možnost transformovat XML do formátu HTML je dnes s oblibou využívaná, jelikož většina prohlížečů si zatím se samotnými XML soubory neumí poradit.

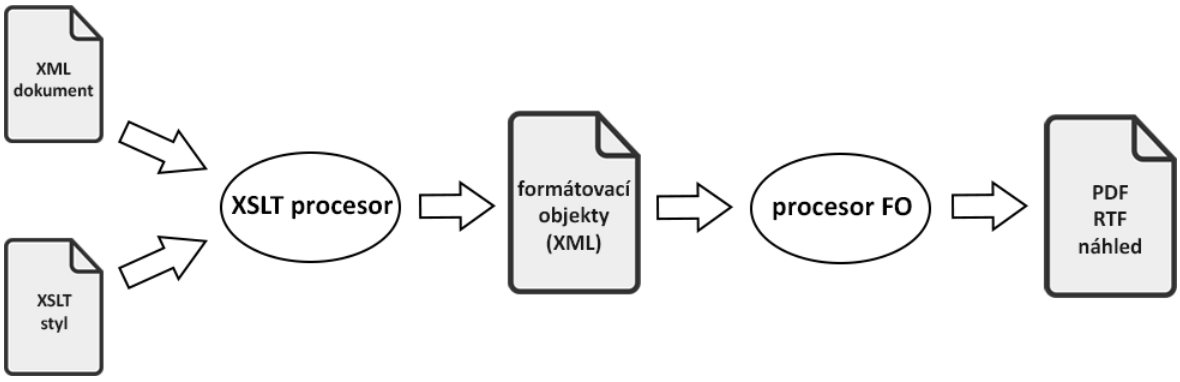
K popisu vzhledu takovýchto transformací slouží tzv. formátovací objekty (XSL FO). Tyto formátovací objekty nabízejí vše, co bychom mohli použít v dokumentu kaskádových stylů CSS, navíc je tu ale možnost definovat layout stránky. Můžeme tak například určit sazbu do více sloupců, definovat oblast hlavičky, patičky atd.

Následující obrázek znázorňuje princip transformace dokumentu XML do jiného formátu na základě XSLT stylu.[12]



Obrázek 7- Ukázka jednoduché transformace

V případě, že naše XSLT šablona obsahuje formátovací objekty, je výsledkem transformace další XML dokument s těmito formátovacími objekty. Tento dokument pak umí interpretovat procesor FO. Procesor FO může z našeho XML dokumentu dále vytvořit PDF dokument, dokument typu RTF, nebo výsledek zobrazí na obrazovce. Následující obrázek zobrazuje princip transformace s využitím formátovacích objektů.[12]



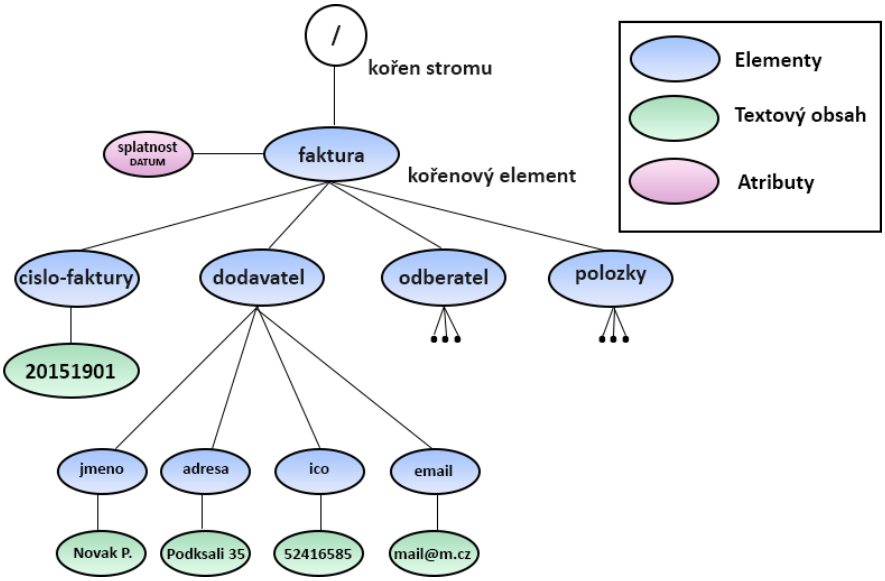
Obrázek 8- Ukázka komplexnější transformace

2.3 XPath

Aby bylo vůbec možné vytvářet šablony, které definují transformace a jejich styl a musí existovat jazyk, pomocí kterého budeme schopni vybírat jednotlivé uzly dokumentu XML, na které se budou transformace aplikovat.

Jazyk XPath (XML Path Language) je samostatný standard W3C, který se používá v několika dalších jazycích včetně XSLT. V XPathu lze zapisovat jednoduché výrazy, které

vybírají části XML dokumentu. XML dokument je přitom chápán jako stromová struktura, kde jsou jednotlivé elementy, atributy a text chápány jako uzly.



Obrázek 9- Stromová struktura dokumentu XML pro XPath výrazy

Výrazy v XPath jsou podobné zápisu cest ve struktuře adresářů. Kdybychom například chtěli vybrat jméno dodavatele, použijeme XPath výraz /faktura/dodavatel/jmeno. Lomítko přitom odděluje jednu úroveň ve stromu. Znamená to, že položka musí být dítětem faktury. Pokud nám na hloubce vnoření nezáleží, použijeme dvě lomítka bezprostředně za sebou – hledají se pak všichni potomci v libovolné úrovni stromu. Následující tabulka představuje některé možnosti zápisu výrazů a jejich vysvětlení.

Tabulka 3- Výrazy XPath

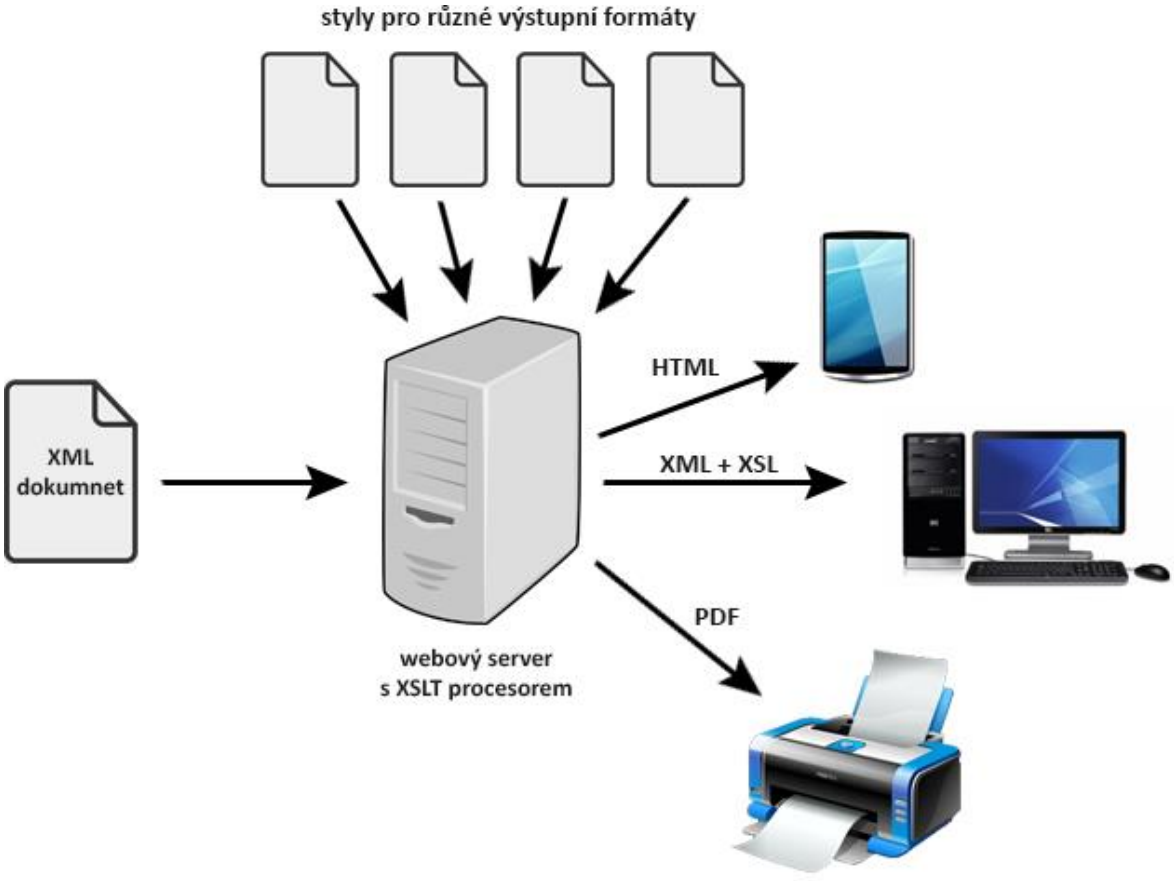
XPath výraz	výsledek
`/bookstore/book[1]`	Vybere první element *book*, který je potomkem elementu *bookstore*
`/bookstore/book[last()]`	Vybere poslední element *book*, který je potomkem elementu *bookstore*
`/bookstore/book[last()-1]`	Vybere předposlední element *book*, který je potomkem elementu *bookstore*
`/bookstore/book[position()<3]`	Vybere první dva elementy *book*, které jsou potomky elementu *bookstore*
`//title[@lang]`	Vybere všechny elementy *titile*, které mají atribut *lang*
`//title[@lang='en']`	Vybere všechny elementy *titile*, které mají atribut *lang* s hodnotou en
`/bookstore/book[price>35.00]`	Vybere všechny elementy *book*, které jsou pomtky elementu *bookstore* a které obsahují element *price* s hodnotou větší než *35.00*
`/bookstore/book[price>35.00]/title`	Vybere všechny elementy title, které jsou pomtky elementu book z elementu bookstore a které obsahují element price s hodnotou větší než 35.00

2.4 Výhody XML transformací

Oddělení obsahu XML dokumentu od jeho vzhledu nám může přijít poněkud nepřirozené. Přináší to však některé výhody. Jednou z výhod je například formátování jednoho dokumentu pomocí více různých stylů. Pokud bychom měli XML dokument, který obsahuje data ceníku naší firmy, můžeme pomocí jednoho stylu vytvořit naformátovaný ceník pro tištěnou podobu a pomocí jiného stylu můžeme stejná data naformátovat pro webové rozhraní.

Výhodou je také jednoduchá aktualizace obsahu. Jelikož se data nacházejí pouze na jednom místě v našem XML dokumentu, postačí, když tento dokument zaměníme za jiný, nebo změníme jeho současná data za jiná a po transformaci se k uživateli dostává čerstvý dokument.

Oblastí, kde se tento přístup začíná prosazovat nejrychleji, je Web. V současné době je zcela běžné, že k jednotlivým informacím potřebujeme přistupovat i z jiných zařízení než jsou PC. Těmito zařízeními můžou být mobilní telefony, tablety, PDA apod. Každé zařízení má jiné schopnosti a možnosti. Webový server může rozpoznat klienta a před přenesením XML dokumentu ho transformovat pomocí stylu, který mu nejvíce vyhovuje.[13]



Obrázek 10- Transformace XML v praxi

Transformace se dají také využít v případě, že chceme například z XML vygenerovat dávku SQL příkazu pro přidání zaměstnanců do databáze. Lze totiž nadefinovat styl, který převede naše XML do prostého textového formátu.

3 Praktická část

V této kapitole bude popsána praktická část práce, která zahrnuje knihovnu poskytující metody pro převod z XML do PDF, procesor FO.NET, který se stará o sestavení dokumentů PDF a nástroj pro tvorbu šablon XSLT.

3.1 Knihovna XmlPdfTransformace

Je programová knihovna, která nabízí metody pro převod dokumentu XML do PDF, nebo do souboru formátovacích objektů XSL-FO. Obě tyto operace se provádí na základě šablony XSLT a v obou případech se využívá transformace XSL, kterou platforma .NET Framework podporuje díky třídám umístěných ve jmenném prostoru System.Xml.Xsl.

Pouze v případě, že je XML dokument převáděn přímo do PDF, je navíc pro tento úkon využito procesoru FO.NET. Ten má na starosti samotné sestavení dokumentu PDF.

Knihovna je určena pro .NET Framework verze 3.5 a vyšší.

Tabulka 4- Metody knihovny XmlPdfTransformace

Metody	
Název	**Popis**
PrevestDoXslFo(XDocument, XDocument)	Na základě vstupního XML dokumentu a odpovídající šabloně XSLT vrátí objekt XDocument, jehož obsahem jsou formátovací objekty odpovídající vstupnímu XML dokumnetu.
PrevestDoXslFo(IXPathNavigable, IXPathNavigable)	Na základě vstupního XML dokumentu a odpovídající šabloně XSLT vrátí objekt XDocument, jehož obsahem jsou formátovací objekty odpovídající vstupnímu XML dokumnetu.
TransformaceDoPdf(Xdocument, String)	Vytvoří PDF dokument ze vstupního souboru, který obsahuje formátovací objekty XSL-FO. Soubor následně uloží na požadované místo na disku.
TransformaceDoPdf(IXPathNavigable, IXPathNavigable, String)	Vytvoří PDF dokument na základě šablony stylů pro vstupní dokument XML a uloží jej na požadované místo na disku.

3.1.1 Převod do XSL-FO

Soubor formátovacích objektů z XML a šablony získáme voláním přetížené metody *PrevestDoXslFo*. Tento formát se může hodit v případě, že budeme mít možnost použít k sestavení výsledného dokumentu vlastní FO procesor. Některé procesory FO zvládají z formátovacích objektů vytvářet dokumenty i jiných formátů, než je PDF. Přetížená metoda PrevestDoXslFo má tedy následující dva tvary:

PrevestDoXslFo(XDocument, XDocument) – tento formát zápisu počítá se vstupním zdrojovým dokumentem dat XML a odpovídající šablonou k těmto datům, která popisujíce transformaci a její styly. V obou případech se jedná o objekty typu XDocument. Samotná metoda také vrací objekt typu XDocument a jedná se tedy o dokument FO.

PrevestDoXslFo(IXPathNavigable, IXPathNavigable) – pro větší pohodlnost je v knihově implementována tato metoda, která pracuje s rozhraním typu IXPathNavigable. XML dokument a šablonu je pak potřeba metodě předávat parametry tohoto typu. Vrací opět sestavený XML dokument FO v podobně XDocument.

3.1.2 Převod do PDF

Chceme-li transformovat dokument XML na základě šablony přímo do dokumentu PDF, můžeme k tomu použít metodu *TransformaceDoPdf*. V případě, že pracujeme přímo se souborem, jehož obsahem jsou formátovací objekty, je vytvoření PDF dokumentu ještě jednodušší. Přetížená metoda TransformaceDoPdf má následující tvary:

TransformaceDoPdf(XDocument, String) – v prvním parametru metoda přijímá XML soubor, který obsahuje formátovací objekty pro stavbu PDF dokumentu. Druhý parametr přebírá cestu pro uložení tohoto dokumentu na disk.

TransformaceDoPdf(IXPathNavigable, IXPathNavigable, String) **–** pomocí této metody je transformace XML do PDF velice jednoduchá. První parametr IXPathNavigable představuje XML data, druhý parametr šablonu XSLT a třetí parametr představuje opět cestu pro uložení výsledného PDF na disk.

3.1.3 Výjimka ArgumentException

Všechny 4 metody obsažené v knihovně XmlPdfTransformace kontrolují správnost vstupních parametrů. Jestliže metodě předáme data, která nejsou validní pro výkon transformace, je v tomto okamžiku vyvolána výjimka ArgumentException.

3.2 Procesor FO.NET

Knihovna XmlPdfTransformace sama o sobě není jádrem, které realizuje převod dokumentů z XML do PDF. Jak již bylo znázorněno na obrázku Obrázek 8- Ukázka komplexnější transformace, PDF dokument sestavuje FO procesor. Knihovna XmlPdfTransformace využívá procesor FO.NET. Jedná se o volně stažitelný[3] software šiřitelný pod licencí Apache License 2.0.

FO.NET byl napsán pro Microsoft .NET Framework a je vhodný pro použití v jakémkoli jazyce pracující na platformě .NET, jako například C#, nebo VB.NET. Mimo sazbu dokumentů do PDF podporuje FO.NET také nastavení jejich základních vlastností, jako je například autor dokumentu, heslo, umožňuje nastavit atribut, který zakazuje tisk, nebo kopírování textu atp.

3.2.1 Apache License 2.0

Vznikla pod komunitou The Apache Software Foundation. Tato licence nás opravňuje k volnému stažení, modifikaci a dokonce i dalšímu šíření aplikace. V případě modifikace, je povinností autora změněné řádky kódu okomentovat. Je zakázáno měnit, nebo mazat původní komentáře. Pokud se rozhodneme software dále šířit, tak jedině pod stejnou licencí, jako je Apache License 2.0. Tak znění podmínky, kterým vyjadřujete souhlas již v případě stažení programu.

3.2.2 Použití procesoru FO.NET

Chceme-li využít knihovnu implementující jádro pro převod XML do PDF, musíme ji zahrnout do projektu a připojit jí k našemu programu. Poté se nám zpřístupní její metody. Následující řádky kódu demonstrují příklad vytvoření instance procesoru a převod jednoduchého souboru s formátovacími objekty (hello.fo) do PDF.[14]

[3] https://fonet.codeplex.com/

3.2.2.1 Soubor hello.fo

```
<?xml version="1.0" encoding="utf-8"?>
<fo:root xmlns:fo="http://www.w3.org/1999/XSL/Format">

  <fo:layout-master-set>
    <fo:simple-page-master master-name="simple"
                  page-height="29.7cm"
                  page-width="21cm"
                  margin-top="1cm"
                  margin-bottom="2cm"
                  margin-left="2.5cm"
                  margin-right="2.5cm">
      <fo:region-body margin-top="3cm"/>
      <fo:region-before extent="3cm"/>
      <fo:region-after extent="1.5cm"/>
    </fo:simple-page-master>
  </fo:layout-master-set>

  <fo:page-sequence master-reference="simple">
    <fo:flow flow-name="xsl-region-body">
      <fo:block font-size="18pt" color="black" text-align="center">
        Hello, World!
      </fo:block>
    </fo:flow>
  </fo:page-sequence>

</fo:root>
```

3.2.2.2 Kód pro realizaci převodu dokumentu do PDF

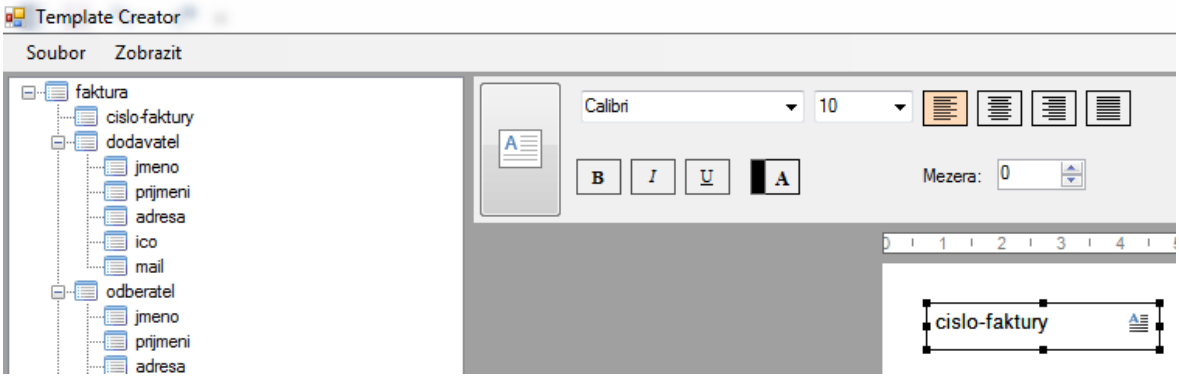
```
using System.IO;
using Fonet;

namespace FonetExample {
    class HelloWorld {
        static void Main(string[] args) {
            FonetDriver driver = FonetDriver.Make();
            driver.Render("hello.fo", "hello.pdf");
        }
    }
}
```

3.3 Nástroj TemplateCreator

TemplateCreator je aplikace určená pro tvorbu šablony XSLT pro daný dokument XML. Aby bylo možné začít šablonou vytvářet, je potřeba programu XML soubor předat. Jestliže je soubor validní, uloží jej program do paměti a zobrazí uživateli jeho stromovou strukturu v komponentě TreeView. Ta se nachází na levé straně aplikace.

Většina moderních programů, určených pro sazbu dokumentů, disponuje jakýmsi plátnem, do kterého je možné sázet jednotlivé prvky. V tomto nástroji představuje grafické plátno komponenta, kterou ve Visual Studiu najdeme pod názvem PictureBox. Tato komponenta dává uživateli jasný přehled o tom, jak bude dokument vypadat, kde budou prvky rozmístěny a navíc umožňuje jejich rychlou a pohodlnou editaci. Dalo by se říci, že se jedná o velice zjednodušenou verzi WYSIWYG editoru.

Z komponenty TreeView je možné metodou Drag and Drop přetahovat uzly dokumentu XML na grafické plátno a vytvořit tak vazbu mezi jednotlivými elementy dokumentu XML a jeho šablonou. Před tímto samotným vložením, je nejdříve zapotřebí vyznačit na plátně formou obdélníků, na jaké pozici se budou data z XML nacházet.



Obrázek 11- Komponenty nástroje TemplateCreator

3.3.1 Jednoduchý text a seznamy

Jestliže máme připravené kontejnery pro data z XML souboru v podobě obdélníků, můžeme elementy vkládat do dokumentu. Dokument XML je ve většině případů rozmanitý a obsahuje několik úrovní. Vkládáme-li do kontejneru element, který nemá žádné potomky, je vytvořeno jednoduché textové pole, obsahující blok textu.

Budou-li obsahem XML souboru například data faktury, najdeme v něm jistě element, který představuje dodavatele zboží. Do tohoto elementu budou s největší pravděpodobností vnořeny další elementy, nesoucí data jako je jméno a příjmení dodavatele, jeho adresa, kontakt atd. V případě, že se uživatel rozhodne vložit na grafické plátno element, který vlastní nějaké potomky, bude vytvořen seznam těchto potomků. Ty se do výsledného dokumentu vykreslí tak, jak bychom od seznamu očekávali.

3.3.2 Tabulky

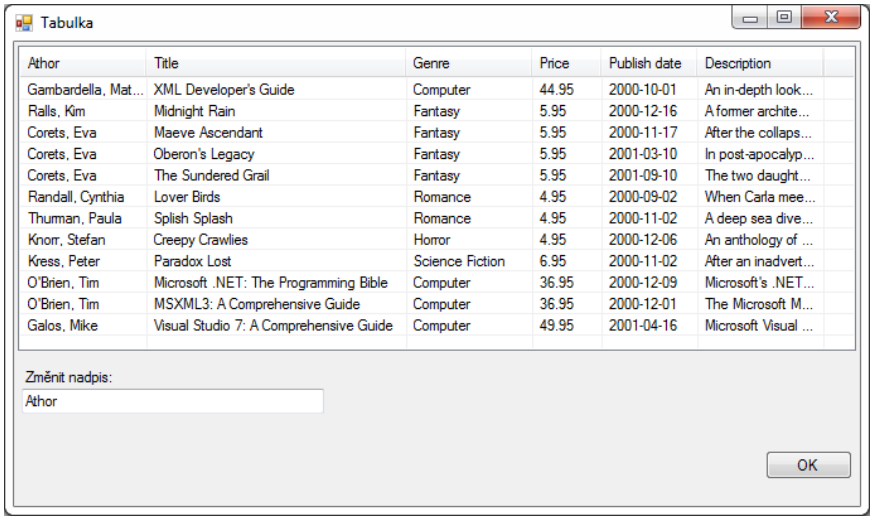
Velmi často může XML soubor obsahovat data, která by se dala reprezentovat pomocí tabulky. Představíme-li si například vyexportovaná data z databáze knihovny, kde se opakují stejné záznamy, bude tabulka jistě na místě.

3.3.2.1 Příklad, soubor books.xml

```
<?xml version="1.0"?>
<catalog>
 <books>
   <book id="bk101">
      <author>Gambardella, Matthew</author>
      <title>XML Developer's Guide</title>
      <genre>Computer</genre>
      <price>44.95</price>
      <publish_date>2000-10-01</publish_date>
      <description>An in-depth look at creating applications with XML.</description>
   </book>
   <book id="bk102">
      <author>Ralls, Kim</author>
      <title>Midnight Rain</title>
      <genre>Fantasy</genre>
      <price>5.95</price>
      <publish_date>2000-12-16</publish_date>
      <description>A former architect battles corporate zombies,
      an evil sorceress, and her own childhood to become queen
      of the world.</description>
   </book>
 </books>
</catalog>
```

Ukázka XML dokumentu 2

Aplikace automaticky rozpozná, že se jedná o tabulku v případě, je-li do kontejneru vložen element, který obaluje opakující se elementy. V souboru books.xml tabulku představuje element <books>. Při vkládání tabulkových dat do dokumentu, se zobrazí dialogové okno, ve kterém má uživatel možnost přizpůsobit tabulku svému obrazu. K dispozici je možnost rozšiřování/zužování sloupců a změna nadpisů sloupců.



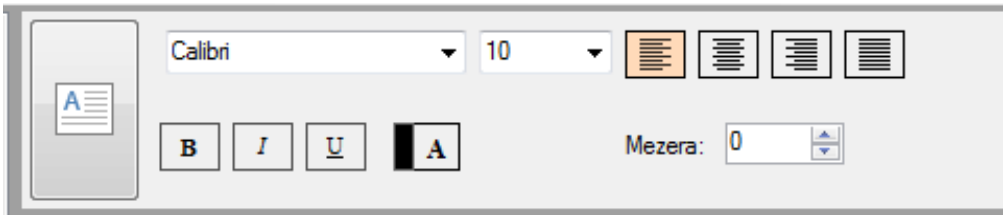
Obrázek 12- Dialog tabulky

Tabulka 5- Legenda typů kontejnerů

	Jednoduché textové pole
	Seznam
	Tabulka

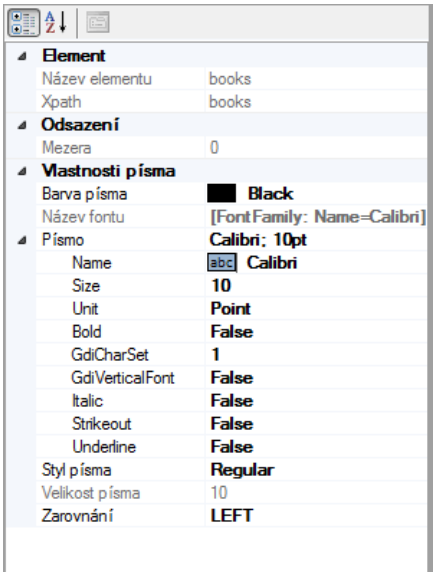
3.3.3 Formátování textu

Aplikace umožňuje formátovat text uvnitř kontejneru dvěma způsoby. Prvním a zároveň přirozenějším způsobem je formátování přes lištu nástrojů, která se nachází přímo nad grafickým plátnem. Lišta nabízí možnost volby použitého fontu písma, nastavení jeho velikosti, zarovnání, dekorace, barva a možnost nastavení mezery mezi bloky.



Obrázek 13- Lišta nástrojů k formátování textu

Některé vlastnosti písma a jeho formátování je možné nastavovat ještě druhým způsobem, který nabízí komponenta PropertyGrig. Tu najdeme v pravé části aplikace a její obsah se aktualizuje dle vybraného kontejneru. PropertyGrid informuje uživatele také o tom, jaký element z dokumentu XML byl kontejneru přidělen a jaký je výraz XPath pro tento prvek.

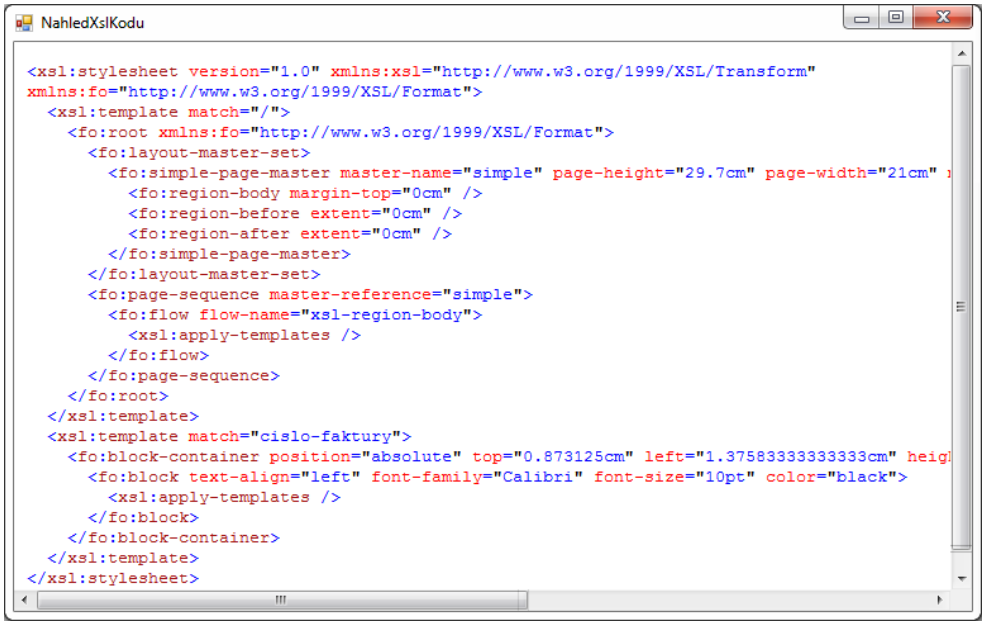


Obrázek 14- Tabulka vlastností formátu písma

3.3.4 Možnosti ukládání

Aplikace nabízí uložení šablony ve formátu XSLT, již přetransformovaný dokument XML v podobě XSL-FO, nebo je možné uložit dokument přímo do PDF.

Ukládáme-li šablonu XSLT, je k dispozici její náhled ať už pro kontrolu jejího obsahu, nebo pouze pro náhled toho, jak transformace dopadla. K možnosti náhledu se uživatel dostane přes hlavní menu: Zobrazit -> Kód šablony XSLT.

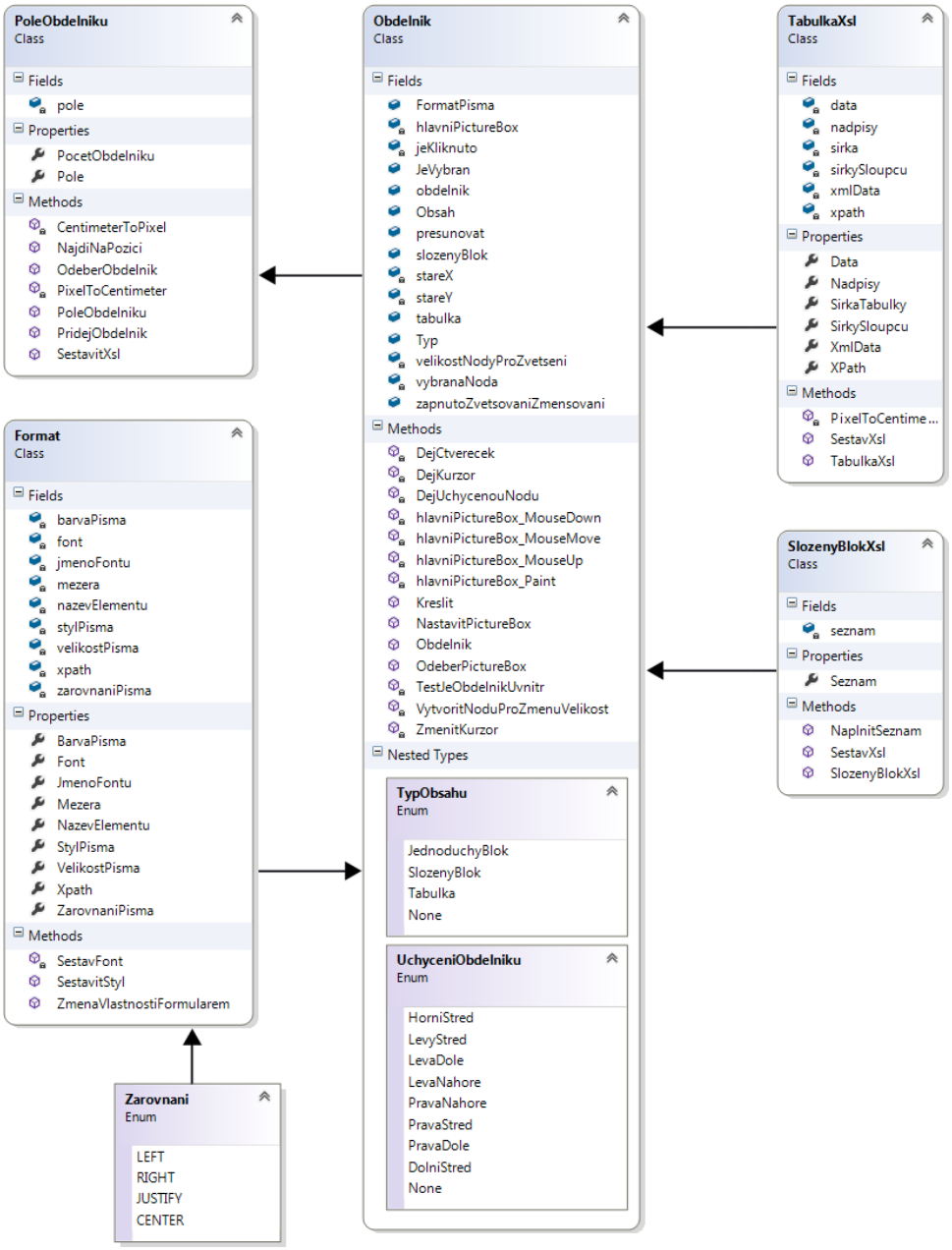


Obrázek 15- Náhled na kód šablony

3.4 Datové struktury

Hlavním stavebním prvkem aplikace je třída Obdelnik. Jedná se o kontejner, pro všechny nastavené vlastnosti prvku umístěného do stránky. Třída je úzce spojena s komponentou grafického plátna, neboť jejím úkolem je také vykreslovat objekty na PictureBox. Kreslení v jazyce C# je poměrně jednoduchou záležitostí, stará se o něj totiž rozhraní GDI+.

Následující obrázek popisuje strukturu aplikace a zobrazuje vazby mezi jednotlivými třídami.



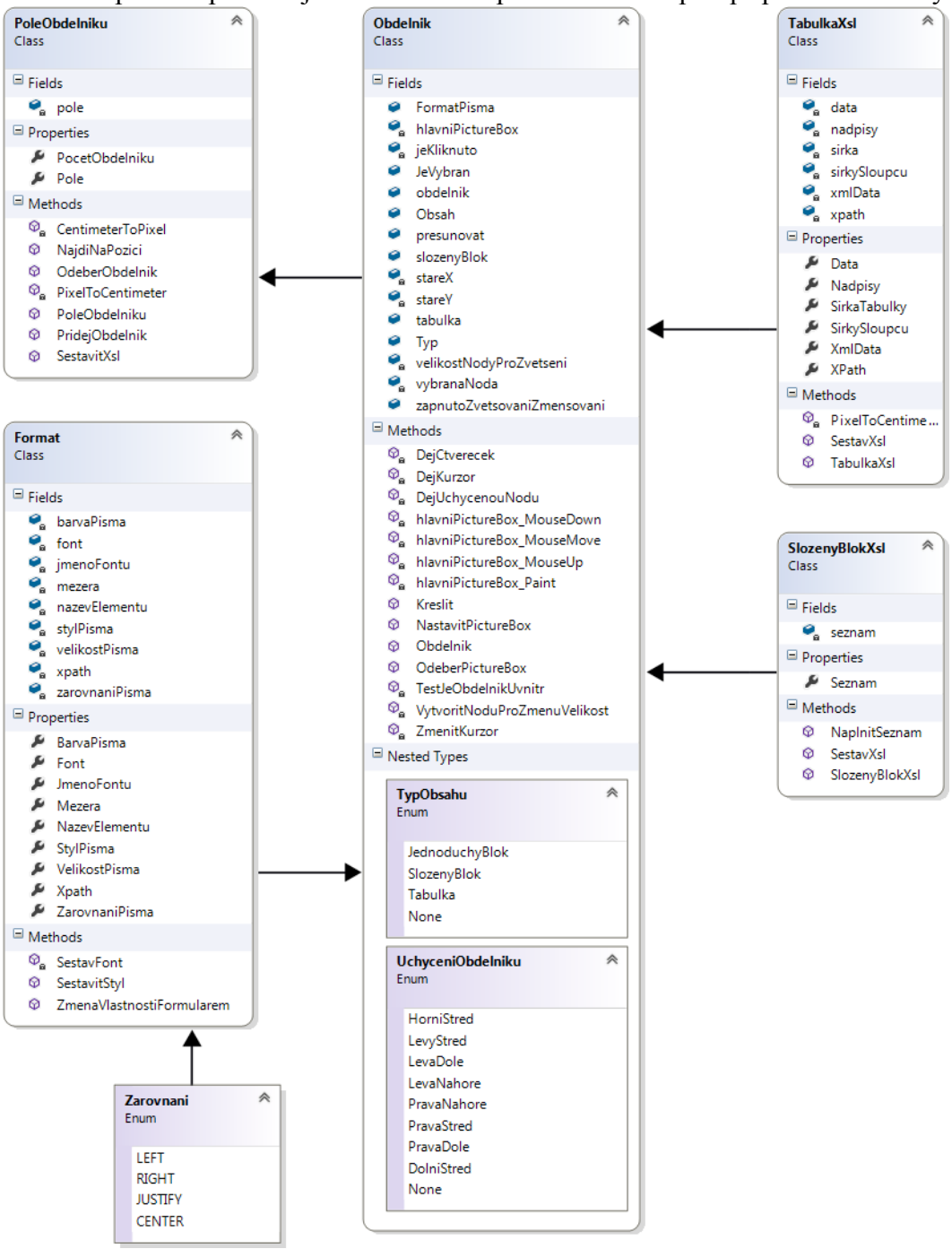
Obrázek 16- Diagram tříd

3.4.1 Rozhraní GDI+ a třída Graphics

K silným stránkám systému Windows patří jeho schopnost abstrahovat od detailů konkrétní zařízení, takže se jimi vývojář nemusí zabývat. Když počítač cokoli vykresluje na obrazovku, odesílá pokyny grafické kartě. Na trhu je nespočetné množství grafických karet a většina z nich poskytuje odlišné sady instrukcí a možnosti. Pokud bychom toto měli brát v úvahu, bylo by programování aplikace, která pracuje s grafikou téměř nemožné, protože by to znamenalo napsat pro každou grafickou kartu jiný kód. Tento problém řeší rozhraní GDI+.

Rozhraní GDI+ poskytuje úroveň abstrakce, která skrývá rozdíly mezi jednotlivými grafickými kartami, a proto stačí volat funkce rozhraní Windows API pro příslušný úkol a rozhraní GDI+ zjistí, jak pomocí konkrétní grafické karty klienta splnit libovolnou operaci, kterou daný úsek kódu požaduje.

Třída Graphics zapouzdřuje GDI+ kreslící plochu a nabízí předpřipravené metody pro



kreslení základní grafiky, jako jsou čáry, oblouky, křivky, elipsy, obdélníky a další.[15]

3.4.2 Class Obdelnik

Třída Obdelnik používá z třídy Graphics metodu DrawRect, pro kreslení obdélníků, které představují kontejner pro data z XML. Text a obrázek uvnitř kontejneru využívají pro vykreslování metody DrawString a DrawImage. Jednotlivé instance této třídy jsou předávány grafickému plátnu, tedy instanci třídy PictureBox. O samotné kreslení se stará metoda Kreslit().

Třída Obdelnik disponuje několika proměnnými:

- `PictureBox hlavniPictureBox` – obsahuje referenci na PictureBox, kterému instance Obdelnik patří,
- `Rectangle obdelnik` – důležitá proměnná pro vykreslování obdélníka na grafické plátno,
- `Format FormatPisma` – velice důležitá proměnná pro uchovávání nastavených parametrů písma,
- `String Obsah` – uchovává výraz XPath v textové podobě,
- `TabulkaXsl tabulka` – proměnná, ve které je uložena struktura tabulky, je-li instance Obdelnik typu Tabulka,
- `SlozenyBlokXsl slozenyBlok` – proměnná, ve které je uložena struktura složeného bloku, je-li instance Obdelnik typu SlozenyBlok,
- další.

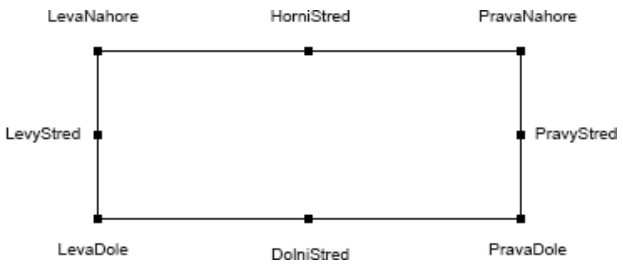
Dále je třída Obdelnik vlastníkem výčtových typů:

3.4.2.1 Enum TypObsahu

Díky výčtovému datovému typu TypObsahu je určeno, zda se jedná o kontejner, obsahující jednoduchý textový blok, složený blok (neboli seznam) a tabulku.

3.4.2.2 Enum UchyceniObdelniku

Ve třídě Obdelnik je k dispozici další výčtový typ UchyceniObdelniku. Aby bylo možné obdélníkům pohodlně nastavovat jejich velikost, kreslí se společně s nimi ještě body pro změnu velikosti. Pomocí tohoto výčtového typu se určuje, který bod byl vybrán a podle toho se přepočítává velikost.



Obrázek 17- Změna velikosti kontejneru

3.4.3 Class Format

Úkolem třídy Format je držet v paměti informace o nastavených parametrech písma. Počítá se s častou změnou parametrů, proto má každá vlastnost postavenou metodu Set ale i Get, aby bylo možné vlastnosti číst. Mezi tyto vlastnosti patří:

- typ písma,
- velikost písma,
- barva,
- zarovnání písma,
- styl písma (tučné, podtržené, kurziva).

Třída Format z těchto vlastností dokáže sestavit styl ve formě atributů pro XML. Styl se pak jednoduše vloží do šablony XSLT, o to se nicméně stará třída PoleObdelniku.

3.4.3.1 Ukázka sestavení stylu

```
switch (zarovnaniPisma)
{
      case Zarovnani.LEFT:
          styl += "text-align=\"left\" ";
          break;
      case Zarovnani.RIGHT:
          styl += "text-align=\"right\" ";
          break;
      case Zarovnani.JUSTIFY:
          styl += "text-align=\"justify\" ";
          break;
      case Zarovnani.CENTER:
          styl += "text-align=\"center\" ";
          break;
      default:
          styl += "";
          break;
}
```

Instance této třídy je v grafickém uživatelském rozhraní předávána komponentě PropertyGrid, kde má uživatel přehled o nastavených hodnotách a zároveň tato komponenta umožňuje některé atributy měnit.

3.4.4 Class TabulkaXsl

Tato třída představuje datový typ, který uchovává všechna nastavení pro tabulku. Tabulka je mnohem komplexnější, než složený blok, nebo jednoduchý textový blok. Jelikož obsahuje nadpisy sloupců, je potřeba uchovávat jejich data a nastavení jejich šířky. Protože se nepředpokládá, že bude mít tabulka vždy konstantní počet sloupců, je pro uchovávání jejich nadpisů a šířek sloupců použita datová struktura List.

Dialogové okno, které se zobrazí při snaze vložit do dokumentu element představující tabulku, pracuje s touto třídou a právě toto okno umožňuje uživateli tabulku přizpůsobit požadovaným představám. Toto okno také poskytuje náhled dat, která tabulce náleží, a proto je nutné předávat instanci této třídy XML dokument, ze kterého se data do tabulky načtou. Jedná se o zdrojové XML, nad nímž je vytvářena šablona.

Důležité proměnné obsažené ve třídě TabulkaXsl:

- `IList<string> nadpisy` – datová struktura pro uchování a editaci nadpisů tabulky
- `TreeNode data` – uzel stromu TreeNode pro inicializaci nadpisů
- `int sirka` – celková šířka tabulky v pixelech
- `IList<string> sirkySloupcu` – datová struktura pro uchování šířky nadpisů
- `XDocument xmlData` – xml dokument, ze kterého se načítají data do tabulky

3.4.5 Class SlozenyBlok

Třída SlozenyBlok je oproti tabulce velice stručná. Obsahuje pouze list pro uchování názvů elementů, které jsou potomky elementu vkládaného do dokumentu a metodou, která se opět postará o sestavení potřebného stylu pro šablonu XSLT.

3.4.6 Class PoleObdelniku

Tato třída je dalším základním pilířem aplikace. Díky seznamu uchovává všechny instance třídy Obdelnik. Implementuje metody pro přidávání a odebírání obdélníků, dále metodu pro zjištění počtu prvků v seznamu, metodu pro vyhledávání specifického prvku a nejdůležitější metodu pro sestavení výsledné šablony XSLT.

Závěr

Cílem této práce bylo vytvořit knihovnu, která umožní převádět dokumenty XML do čitelného formátu PDF dle šablony popisující transformaci XML a její vzhled. Výsledkem je velice kompaktní knihovna obsahující čtyři statické metody, které programátorovi zpříjemní práci s převodem. V metodách je již vyřešeno ukládání sestavených souborů PDF, nebo XSL-FO na disk počítače, a proto programátorovi tato starost odpadá. Funkčnost knihovny byla otestována v nástroji na tvorbu šablon, pro kterou byla přímo napsána, nicméně najde své uplatnění všude tam, kde se pracuje s daty ve formátu XML. Je potřeba mít na paměti, že samotný soubor XML nestačí a je třeba k němu vždy vytvořit odpovídající šablonu. Například většina účetních programů umí exportovat data jednotlivých faktur právě do dokumentu XML. Pokud budeme mít k takovému souboru příslušnou šablonu, je možné převádět faktury do PDF a rozesílat je v čitelné podobě zákazníkům.

Nástroj pro tvorbu šablon XSLT poskytuje základní funkce, které bychom čekali od programu pro sazbu dokumentů, má však jeden zásadní nedostatek. Ačkoli může být kód šablony velice rozmanitý a dalo by se říci nekonečně dlouhý, je použití šablony omezeno pouze na jednu stránku papírového formátu A4. Většina profesionálních programů, jako je například Adobe InDesign, mají možnost pracovat s grafickými prvky. Umožňují vytvářet různé tvary, které se dají použít jako pozadí pro bloky, nebo nabízejí import obrázků. Funkce tohoto typu tato aplikace také postrádá, proto je vhodné ji použít například pro sazbu pro tiskopisy, předtisknuté formuláře apod. Jazyk pro vytváření šablon má mnohonásobně více možností, než bylo použito v této práci a proto je možné aplikaci dále rozvíjet.

Při vývoji programu TemplateCreator představovalo největší problém sestavení výsledné šablony XSLT, neboť obsahuje dva jmenné prostory. Platforma .NET Framework poskytuje bohatou a silnou sadu nástrojů pro práci s jazykem XML a je možné pomocí těchto nástrojů dokumenty XML vytvářet, editovat a provádět nad jejich strukturou různé dotazy, podobné jazyku SQL. I přes tyto možnosti, které .NET nabízí, je výsledná šablona v programu TemplateCreator poskládána dohromady pouhým spojováním textových řetězců, což je neefektivní a v budoucnu by bylo dobré těchto možností využít.

Jak knihovna, tak nástroj pro tvorbu šablon byly vyvíjeny na platformě .NET Framework a jejich použití se tedy omezuje pouze na produkty společnosti Microsoft. Vzhledem k tomu, že využívají technologie, které byly integrovány do .NET Frameworku verze 3.5, je nutné mít na počítači nainstalovanou tuto verzi nebo vyšší.

Vhledem k tomu, že byl do knihovny zahrnut FO procesor, který podléhá licenci Apache License 2.0, je potřeba mít na paměti, že i šíření a používání této knihovny je možné pouze pod touto licencí a není možné ji využívat pro komerční účely.

Zdroje informací

[1] Formát papíru. *Wikipeda* [online]. [cit. 2015-04-27]. Dostupné z: http://cs.wikipedia.org/wiki/Form%C3%A1t_pap%C3%ADru

[2] Adobe Systems. *Wikipeda* [online]. [cit. 2015-04-27]. Dostupné z: http://cs.wikipedia.org/wiki/Adobe_Systems

[3] Adobe InDesign CC. *Adobe: Creative, marketing, and document management solutions* [online]. 2015 [cit. 2015-04-27]. Dostupné z: http://www.adobe.com/cz/products/indesign.html

[4] TŘEŠŇÁK, Kamil. InDesign v praxi - hlavní rysy programu. *Svět tisku* [online]. 2005 [cit. 2015-04-27]. Dostupné z: http://www.svettisku.cz/buxus/generate_page.php?page_id=1667

[5] CorelDraw. *Wikipeda* [online]. [cit. 2015-04-27]. Dostupné z: http://cs.wikipedia.org/wiki/CorelDraw

[6] Microsoft Publisher 2010: snadná tvorba publikací v prostředí MS Office. *Grafika.cz: vše o počítačové grafice* [online]. 2011 [cit. 2015-04-27]. Dostupné z: http://www.grafika.cz/rubriky/software/microsoft-publisher-2010-snadna-tvorba-publikaci-v-prostredi-ms-office-138269cz

[7] Scribus Wiki. *Scribus Wiki* [online]. 2014 [cit. 2015-04-27]. Dostupné z: http://wiki.scribus.net/canvas/Scribus

[8] KOSEK, Jiří. XML -- staronový formát. *Domovská stránka Jirky Koska -- "VŠE O WWW"* [online]. 1999 [cit. 2015-04-27]. Dostupné z: http://www.kosek.cz/clanky/xml/xml-historie.html

[9] KOSEK, Jiří. Základy jazyka XML. *Domovská stránka Jirky Koska -- "VŠE O WWW"* [online]. 1999 [cit. 2015-04-27]. Dostupné z: http://www.kosek.cz/clanky/swn-xml/syntaxe.html

[10] KOSEK, Jiří. XML & aplikace: Parsery. *Domovská stránka Jirky Koska -- "VŠE O WWW"* [online]. 1999 [cit. 2015-04-27]. Dostupné z: http://www.kosek.cz/clanky/swn-xml/ar02s30.html

[11] BŘÍZA, Petr. Kompletní průvodce XSLT – jmenné prostory. *Interval.cz: Svět Internetu, Technologií a Bezpečnosti* [online]. 2004 [cit. 2015-04-27]. Dostupné z: http://wiki.scribus.net/canvas/Scribus

[12] KOSEK, Jiří. Kapitola 1. Úvod: Základní principy XSLT. *Domovská stránka Jirky Koska -- "VŠE O WWW"* [online]. 1999 [cit. 2015-04-27]. Dostupné z: http://www.kosek.cz/xml/xslt/uvod.html

[13] KOSEK, Jiří. Kapitola 1. Úvod: Použití stylů. *Domovská stránka Jirky Koska -- "VŠE O WWW"* [online]. 1999 [cit. 2015-04-27]. Dostupné z: http://www.kosek.cz/xml/xslt/pouziti.html

[14] FO.NET. *CodePlex: Project Hosting for Open Source Software* [online]. 2004 [cit. 2015-04-27]. Dostupné z: https://fonet.codeplex.com/

[15] NAGEL, Christian. *C# 2008: programujeme profesionálně.* Vyd. 1. Brno: Computer Press, 2009, 2 sv. (1126, 772 s.). Programujeme profesionálně. ISBN 978-80-251-2401-7.

Příloha A – kód XML dokumentu faktura.xml

```
<?xml version="1.0" encoding="utf-8"?>
<faktura>
 <cislo-faktury>20110001</cislo-faktury>
 <dodavatel>
  <jmeno>Petr</jmeno>
  <prijmeni>Novak</prijmeni>
  <adresa>Virtualni adresa 111, Virtualni mesto 123 45</adresa>
  <ico>12345678</ico>
  <mail>petr.novak@mail.xy</mail>
 </dodavatel>
 <odberatel>
  <jmeno>Jaroslav</jmeno>
  <prijmeni>Horky</prijmeni>
  <adresa>Virtualni adresa 321, Virtualni mesto 65 321</adresa>
  <ico>55228844</ico>
  <mail>jaroslav.horky@mail.xy</mail>
 </odberatel>
 <polozky>
  <polozka>
   <nazev>HS FLAMINGO VEGA</nazev>
   <pocet-kusu>1</pocet-kusu>
   <cena-ks>4 799,-</cena-ks>
   <cena-celkem>4 799,-</cena-celkem>
  </polozka>
  <polozka>
   <nazev>Pirelli Winter 240 SottoZero</nazev>
   <pocet-kusu>4</pocet-kusu>
   <cena-ks>3 396,-</cena-ks>
   <cena-celkem>13 584,-</cena-celkem>
  </polozka>
  <polozka>
   <nazev>E-PL3 Kit + 14-42 mm black/black</nazev>
   <pocet-kusu>1</pocet-kusu>
   <cena-ks>5 599,-</cena-ks>
   <cena-celkem>5 599,-</cena-celkem>
  </polozka>
  <polozka>
   <nazev>Baterie Samsung EB494358VU</nazev>
   <pocet-kusu>2</pocet-kusu>
   <cena-ks>149,-</cena-ks>
   <cena-celkem>298,-</cena-celkem>
  </polozka>
  <polozka>
   <nazev>Sony SmartBand SWR10</nazev>
   <pocet-kusu>1</pocet-kusu>
   <cena-ks>1 080,-</cena-ks>
   <cena-celkem>1 080,-</cena-celkem>
  </polozka>
 </polozky>
 <vysledna-cena>25 360,-</vysledna-cena>
</faktura>
```

Příloha B – výsledek sestavení PDF pro faktura.xml

20110001

Petr	Jaroslav
Novak	Horky
Virtualni adresa 111, Virtualni mesto 123 45	Virtualni adresa 321, Virtualni mesto 65 321
12345678	55228844
petr.novak@mail.xy	jaroslav.horky@mail.xy

Nazev zbozi	Pocet kusu	Cena za kus	Cena celkem
HS FLAMINGO VEGA	1	4 799,-	4 799,-
Pirelli Winter 240 SottoZero	4	3 396,-	13 584,-
E-PL3 Kit + 14-42 mm black/black	1	5 599,-	5 599,-
Baterie Samsung EB494358VU	2	149,-	298,-
Sony SmartBand SWR10	1	1 080,-	1 080,-

25 360,-