VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ANOTAČNÍ DOPLNĚK PRO INTERNET EXPLORER

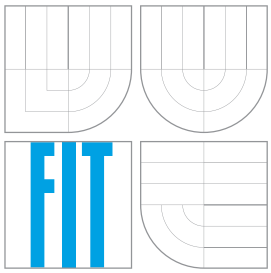
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

PAVEL PĚNKAVA

BRNO 2012

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA

ANOTAČNÍ DOPLNĚK PRO INTERNET EXPLORER

ANNOTATION ADDON FOR INTERNET EXPLORER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

PAVEL PĚNKAVA

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. JAROSLAV DYTRYCH

BRNO 2012

Vysoké učení technické v Brně - Fakulta informačních technologií

Ústav počítačové grafiky a multimédií Akademický rok 2011/2012

Zadání bakalářské práce

Řešitel: **Pěnkava Pavel**
Obor: Informační technologie
Téma: **Anotační doplněk pro Internet Explorer**
Annotation Addon for Internet Explorer
Kategorie: Web

Pokyny:

1. Seznamte se s možnostmi tvorby doplňků ve webovém prohlížeči Internet Explorer.
2. Seznamte se technologiemi potřebnými pro tvorbu doplňků do webového prohlížeče.
3. Navrhněte doplněk pro Internet Explorer, který umožní anotovat text webové stránky a zasílat fragmenty anotovaného textu s anotacemi na server.
4. Implementujte navržené řešení.
5. Zhodnoťte dosažené výsledky a navrhněte další možná vylepšení do budoucna.

Literatura:

- podle doporučení vedoucího

Při obhajobě semestrální části projektu je požadováno:

- Bez požadavků

Podrobné závazné pokyny pro vypracování bakalářské práce naleznete na adrese http://www.fit.vutbr.cz/info/szz/

Technická zpráva bakalářské práce musí obsahovat formulaci cíle, charakteristiku současného stavu, teoretická a odborná východiska řešených problémů a specifikaci etap (20 až 30% celkového rozsahu technické zprávy).

Student odevzdá v jednom výtisku technickou zprávu a v elektronické podobě zdrojový text technické zprávy, úplnou programovou dokumentaci a zdrojové texty programů. Informace v elektronické podobě budou uloženy na standardním nepřepisovatelném paměťovém médiu (CD-R, DVD-R, apod.), které bude vloženo do písemné zprávy tak, aby nemohlo dojít k jeho ztrátě při běžné manipulaci.

Vedoucí: **Dytrych Jaroslav, Ing.**, UPGM FIT VUT
Datum zadání: 1. listopadu 2011
Datum odevzdání: 16. května 2012

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
Fakulta informačních technologií
Ústav počítačové grafiky a multimédií
612 66 Brno, Božetěchova 2

doc. Dr. Ing. Jan Černocký
vedoucí ústavu

Abstrakt

Tato práce se zabývá tvorbou rozšíření prohlížeče Internet Explorer pro vytváření anotací. Je v ní popsán návrh rozšíření a následná implementace řešení. Dále obsahuje vyhodnocení ostatních použitelných technologií pro tvorbu rozšíření spolu s jejich výhodami a nevýhodami. Doplněk je určen pro spolupráci s anotačním serverem projektu 4A Framework (Annotations Anywhere, Annotations Anytime). Formát komunikace mezi nimi tedy vychází ze specifikací projektu.

Abstract

This thesis deals with the creation of the Internet Explorer browser extension for annotation creation. It describes the extension concept and the consequent implementation of solution. It also includes the evaluation of other applicable technologies for extension creation, along with their advantages and disadvantages. Addon is designed to cooperate with the annotation server of 4A Framework (Annotations Anywhere, Annotations Anytime) project. The communication format between them is therefore based on the project specifications.

Klíčová slova

anotace, doplněk, rozšíření, Internet Explorer, web, prohlížeč

Keywords

annotation, addon, extension, Internet Explorer, web, browser

Citace

Pavel Pěnkava: Anotační doplněk pro Internet Explorer, bakalářská práce, Brno, FIT VUT v Brně, 2012

Anotační doplněk pro Internet Explorer

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Jaroslava Dytrycha. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

.....
Pavel Pěnkava
14. května 2012

Poděkování

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Jaroslavu Dytrychovi za užitečné rady, připomínky a vstřícný přístup při vedení práce.

Obsah

Kapitola 1

Úvod

Cílem této práce je analyzovat možnosti pro vytváření rozšíření webového prohlížeče Internet Explorer a pomocí získaných informací navrhnout a implementovat doplněk, který bude umožňovat tvorbu nových anotací a zajišťovat jejich ukládání na anotační server projektu 4A Framework.

V kapitole 2 jsou přiblížena existující vývojová prostředí, frameworky a programovací jazyky, pomocí kterých je možné vyvíjet rozšíření pro prohlížeč Internet Explorer.

Kapitola 3 definuje požadavky na výsledný anotační doplněk z pohledu funkcí a uživatelského rozhraní. Dále vysvětluje základní pojmy, jako je anotace, a představuje projekt 4A Framework spolu s jeho komunikačním protokolem a novým formátem anotace.

Pro fungování doplňku jako celku jsou nezbytné různé technologie, protokoly, jazyky a prostředí. Základní informace o nich získáte v kapitole 4.

Vše o návrhu doplňku obsahuje kapitola 5. Nejprve je v ní znázorněno rozmístění prvků uživatelského rozhraní jednotlivých panelů a oken. Poté popisuje ukládání uživatelského nastavení. Následně jsou detailně vysvětlena validační pravidla pro formulářové prvky. Dále přibližuje situaci kolem vícejazyčného prostředí. Nakonec se zabývá komunikací mezi doplňkem a serverem a v úplném závěru analyzuje některé potenciální problémy v budoucí implementaci.

Kapitola 6 postupně popisuje všechny problémy, na které jsem narazil během implementace doplňku, spolu s jejich řešením. Zahrnuje mimo jiné řešení chyb ve frameworku SpicIE, nedostatečné prostředky rozhraní Internet Exploreru pro práci s dokumentem a komplikace spojené s využitím některých nástrojů Microsoft Visual Studia.

Velmi důležitou součástí vývoje doplňku je jeho testování. Tomu se věnuje kapitola 7. V jejích podkapitolách jsou přiblíženy jednotlivé metody testování doplňku spolu s jejich výsledky a s opravami, které na jejich základě bylo nutné provést.

Poslední kapitola 8 shrnuje postup práce na vývoji doplňku a přináší seznam možných budoucích rozšíření a vylepšení.

Kapitola 2

Analýza možností tvorby rozšíření

Rozšíření prohlížeče umožňují vytvářet panely aplikace (Bar), panely nástrojů (Toolbar), pomocníky prohlížeče (Browser Helper Object, zkráceně BHO), upravovat menu, přidávat tlačítka a přistupovat k objektům a datům prohlížeče. Pro Internet Explorer mohou být doplňky psány několika programovacími jazyky v různých vývojových prostředích, nicméně výsledným výstupem je vždy DLL knihovna. Tato knihovna musí obsahovat minimálně objekt pomocníka prohlížeče, který je vstupním bodem celého rozšíření. Teprve prostřednictvím BHO se totiž při spuštění prohlížeče vytváří instance barů, toolbarů, atd.

2.1 Add-in Express for Internet Explorer and Microsoft .net

Nejjednodušším nástrojem pro vývoj rozšíření je produkt Add-in Express for Internet Explorer and Microsoft .net [2]. K jeho využití potřebujete vývojové prostředí Microsoft Visual Studio 2005/2008/2010 (mimo Express edicí), protože Add-in Express rozšiřuje jeho funkčnost a nelze jej tedy použít samostatně. Po instalaci máte možnost vytvářet rozšíření pro Internet Explorer 6, 7, 8 a 9, a to od verze 7 i pro 64-bitové edice. Můžete si také zvolit jeden z podporovaných programovacích jazyků, kterým budete psát vlastní zdrojový kód – Visual Basic .NET, Visual C# nebo Visual C++. K dispozici je i průvodce vytvořením instalátoru vytvářeného rozšíření. Tvorba doplňků je tedy velmi jednoduchá, je zaručena kompatibilita se všemi aktuálně používanými verzemi Internet Exploreru a projekt je dále vyvíjen. Bohužel je vše vykoupeno poměrně vysokou cenou, kterou musíte za tento produkt zaplatit. Pořizovací cena se pohybuje od $179 za standardní verzi až po $494 za prémiovou. Pokud se tedy nechcete tvorbě doplňků věnovat dlouhodobě, není na tvorbu jednoho doplňku příliš vhodný.

2.2 SpicIE framework

Projekt SpicIE [22], podobně jako Add-in Express, umožňuje tvorbu doplňku v prostředí Microsoft Visual Studio 2008 (všechny edice). Funguje i pod Microsoft Visual Studio 2010, ale v něm již nejsou k dispozici šablonové soubory pro vytváření nových projektů. Tvorba rozšíření je podle autora testovaná pro Internet Explorer 7 a 8, ale funguje i pro verze 6, 9 a 10, bohužel v žádné verzi pro 64-bitové edice. Doplňky potřebují ke svému běhu Microsoft .NET Framework v2.0, v3.0 nebo v3.5, popřípadě neoficiálně v4. Vlastní kód je pak možné na základě šablon psát v jazyce Visual C# nebo Visual Basic .NET. Je možné vytvořit instalátor vlastního rozšíření, ale pouze manuálně bez průvodce. Tvorba doplňku s ním

není tak jednoduchá jako s využitím Add-in Express, nepodporuje 64-bitové edice a dále se nevyvíjí, ale obrovskou výhodou je, že framework je kompletně zdarma a pro případné úpravy a modifikace je k dispozici i jeho zdrojový kód. Pro napsání jediného doplňku je tedy vhodnou volbou.

2.3 Programování mimo prostředí podporující tvorbu rozšíření pro Internet Explorer

V již zmíněných jazycích Visual C#, Visual Basic .NET a Visual C++ je možné napsat vlastní rozšíření i bez podpory vývojového prostředí. Teoreticky tak jde kód celého rozšíření napsat v textovém editoru, zkompilovat jej, v registrech přidat příslušné klíče pro registraci DLL knihovny a doplněk bude fungovat správně. V praxi by to ale znamenalo napsání velkého množství kódu pro samotnou registraci doplňku a základní funkcionalitu, které už ale psalo mnoho lidí před Vámi, a stejně tak nemožnost využívat návrhář zobrazení v Microsoft Visual Studiu. Díky návrháři zobrazení totiž ušetříte spoustu času s umisťováním prvků grafického rozhraní, protože je můžete jednoduše rozmístit pomocí myši a vidíte jejich rozložení. Navíc by bylo nutné důkladně prostudovat mnohdy neaktuální a ne moc přehlednou dokumentaci na stránkách MSDN (*Microsoft Developer Network*) [5]. Pro tvorbu doplňku je tato cesta samozřejmě možná, ale časově nejnáročnější a nejvíce komplikovaná.

2.4 Předpoklady pro vývoj doplňku

Na základě vlastností jednotlivých alternativ tvorby rozšíření, popsaných výše, jsem se rozhodl pro využití frameworku SpicIE kvůli jeho ceně a nabízeným možnostem v kombinaci s využitím programovacího jazyka C# pro jeho syntaxi příbuznou jazyku C a dále pro platformu Microsoft .NET Framework v4 Client Profile, protože je nejaktuálnější, bohatě dostačuje potřebám doplňku, a navíc je k dispozici prostřednictvím systému Microsoft Windows Update všem uživatelům od operačního systému Windows XP (Service Pack 3).

Vzhledem k tomu, že v programovacím jazyce C# jsem ještě nikdy dříve nepsal, musel jsem se s ním nejdříve seznámit. Zjistil jsem ale, že jazyk C# je podobný objektově orientovaným jazykům C++ a Java, což bylo výhodné, protože tyto jazyky jsem již znal. Dále jsem musel prostudovat i Microsoft Visual Studio, jelikož ani v něm jsem zatím nepracoval. Posledním přípravným krokem pak bylo prostudování frameworku SpicIE, díky kterému jsem konečně mohl začít vytvářet přímo rozšíření pro Internet Explorer.

Kapitola 3

Analýza požadavků na doplněk

Požadavky na doplněk vycházely z projektu 4A Framework [10], popsaného v podkapitole 3.3, s jehož serverem bude doplněk komunikovat. Výsledné rozšíření by se také mělo stát jeho součástí, stejně jako doplňky pro webové prohlížeče Mozilla Firefox a Opera.

3.1 Pojem anotace

Anotace je stručným rozšířením, dodatkem, vysvětlivkou nebo také poznámkou k původní informaci, která mívá ve většině případů textovou formu. Dalším významem je samotný proces tvorby anotace, nicméně dále budu pod pojmem anotace uvažovat pouze prvně zmíněný význam ve tvaru podstatného jména. Existuje více druhů anotací, mezi něž patří i strukturované anotace, které jsou podstatné právě pro vývoj doplňku. Strukturovaná anotace projektu 4A Framework zahrnuje, podobně jako ostatní druhy anotací, vlastní textový obsah, ale zároveň ji rozšiřuje o další informace. Anotace tak mají vždy jeden typ ze stromové struktury typů, která může být libovolně rozšiřována. Na základě výběru typu má pak každá anotace také množinu atributů, které mohou obsahovat nejrůznější další informace doplňujícího charakteru.

Na základě typu a vyplněných relevantních atributů jednotlivých anotací dochází ke kategorizaci anotovaného obsahu, který je pak snadno zpracovatelný, tříditelný a lze v něm snadněji a přehledněji vyhledávat.

3.2 Požadavky na funkce

Vzhledem k tomu, že funkce serveru jsou velmi rozsáhlé, je pravděpodobné, že vytvořený doplněk pro Internet Explorer zatím nebude umět vše, co server nabízí, protože by značně překročil rozsah bakalářské práce. Výsledné rozšíření by tedy mělo umožňovat minimálně přihlašování a odhlašování uživatelů registrovaných na serveru, načítání a ukládání uživatelského nastavení (adresa serveru, port, uživatelské jméno) a zejména vytváření a ukládání anotace webových stránek. Pro jednoznačné určení pozic anotovaných fragmentů textu v načteném těle dokumentu je požadován jazyk XPath, který je sice určen pro XML dokumenty, ale lze jej použít i pro HTML. U každé anotace také musí být uložen její typ. Stromová struktura typů je uložena na serveru, ze kterého ji doplněk získává. Případně pokud uživateli žádný z existujících typů nevyhovuje, může přidat nový. Pro každý typ anotace jsou na serveru uloženy také jeho odpovídající atributy. Atribut má vlastní výchozí typ, který

může pro konkrétní anotaci uživatel v opodstatněných případech změnit, popřípadě rovnou nastavit nový výchozí typ atributu na serveru. Mezi základní typy atributů patří:

- textový řetězec (String),
- celé číslo (Integer),
- desetinné číslo (Decimal),
- datum (Date),
- čas (Time),
- datum a čas (DateTime),
- URI identifikátor (URI),
- pravdivostní hodnota (Boolean),
- zeměpisná poloha (GeoPoint).

Jednotlivé typy mají své specifické požadavky, které musí atributy daného typu splňovat, jinak nebude jejich hodnota akceptována, jak bude popsáno v kapitole 5.4. Stejně, jako může uživatel měnit výchozí typ atributu, může přidávat i nové atributy a odebírat existující. To lze provést opět pro konkrétní anotaci, případně také uložit změny pro daný typ anotace přímo na server, aby je mohli využít i ostatní uživatelé. Před začátkem práce s dokumentem (obvykle ihned po načtení stránky prohlížečem) se musí provádět synchronizace, která zajišťuje, že na serveru bude uložena aktuální kopie anotovaného dokumentu.

3.3 4A Framework

Projekt 4A Framework (*Annotations Anywhere, Annotations Anytime*) umožňuje uživatelům mimo jiné anotovat dokumenty a sdílet tyto anotace mezi sebou. Nejdůležitějšími částmi projektu jsou nový XML formát anotace, komunikační protokol pro komunikaci serveru s klienty a návrh uživatelského rozhraní klientů. K dispozici je také anotační server, který zpracovává a ukládá všechny anotace. O vytváření anotací by se potom měli starat klienti serveru, kterými jsou vyvíjené anotační doplňky pro JavaScriptové editory a webové prohlížeče Mozilla Firefox, Opera a Internet Explorer.

3.4 Komunikační protokol 4A Frameworku

Pro tvorbu doplňku bude zásadní protokol, potřebný pro komunikaci se serverem. Od specifikace protokolu (aktuální verze je k dispozici na stránkách projektu [11]) se také odvíjí vývoj serveru a klientů. Všechna komunikace probíhá přes protokol HTTP ve formátu XML dokumentu, jehož kořenovým elementem je element `<messages>`, který obsahuje všechny zprávy přenášené v obou směrech mezi serverem a klientem. Doplněk bude z kompletní specifikace protokolu využívat jen následující podmnožinu, protože vzhledem k požadavkům nebude mít některé pokročilé funkce.

- Správa sezení – kompletní podpora od připojení k serveru, přes přihlášení uživatele až po jeho odhlášení.

- Synchronizace dokumentu – při načtení stránky se automaticky odešle kopie aktuálního dokumentu na server a doplněk získá URI anotované kopie.
- Přenos typů anotace – doplněk bude načítat typy ze serveru a ukládat nové (prozatím nebude umožňovat vícenásobnou dědičnost, komentáře, zakázání modifikace atributů a uživatelské skupiny); dále bude možné přidávat, měnit a mazat jejich atributy (podporovanými typy atributů budou základní typy z podkapitoly 3.2).
- Přenos anotací – přenos anotací bude pouze jednosměrný (z doplňku na server); získávání anotací ze serveru, jejich vizualizace a práce s nimi nebude implementována, ale je s ní počítáno jako s budoucím rozšířením.
- Chyby a varování – doplněk bude zobrazovat a reagovat pouze na relevantní chyby a varování, které souvisí s výše uvedenými body komunikace.
- Potvrzení bez odeslání dat – tuto zprávu bude doplněk očekávat jako potvrzení odhlášení uživatele nebo potvrzení uložení změn atributů.

Kapitola 4

Technologie užité v rozšíření

V rámci doplňku jsou, kromě již zmíněného frameworku SpicIE a programovacího jazyka Visual C#, využity různé další technologie, které jsou nutné pro správné fungování doplňku jako celku. V následujících podkapitolách proto budou všechny představeny a popsáno jejich využití.

4.1 C#/Visual C#

C# je objektově orientovaný programovací jazyk vytvořený společností Microsoft, zveřejněný v roce 2001 a později standardizovaný pod specifikací ECMA-334 (aktuální je čtvrtá edice [6]) a odpovídající ISO/IEC 23270:2006 [21]. Přejímá syntaxi z jazyka C a vychází z programovacích jazyků C++ a Java. Poslední stabilní verzí je aktuálně 4, stejně jako Microsoft .NET Frameworku, který se od počátku vyvíjí zároveň. Kód jazyka C# se totiž překládá a následně spouští právě pod .NET Frameworkem, jak bude popsáno v následující podkapitole 4.2. Visual C# je pak pokročilá implementace společnosti Microsoft ve vývojovém prostředí Microsoft Visual Studio, která převažuje nad ostatními současnými implementacemi jazyka C#. Více informací najdete v [30].

4.2 Microsoft .NET Framework

Platforma Microsoft .NET Framework [1] umožňuje překládat zdrojové kódy různých programovacích jazyků, jako je Visual Basic .NET, C#, C++, F# a jiné, do mezijazyka Common Intermediate Language (CIL) a poté je beze změny programového kódu spouštět na rozmanitých architekturách prostřednictvím Common Language Runtime (CLR) [28]. Mimo jiné tak umožňuje i spolupráci kódu, který byl psán v odlišných zdrojových programovacích jazycích. Platforma .NET Framework je k dispozici pro osobní počítače, ale také pro přenosná zařízení (mobilní telefony, kapesní počítače), vestavěná zařízení a další. Kromě překladače a běhového prostředí také obsahuje velké množství podpůrných knihoven, které značně usnadňují vývoj aplikací. Aktuální verze je .NET Framework 4, která je kompaktnější než předchozí verze [16], ale připravuje se už i verze 4.5. Další informace můžete získat v [32].

4.3 SpicIE framework

Framework SpicIE obaluje složité standardní rozhraní COM (*Component Object Model*) [7] pro rozšíření Internet Exploreru a umožňuje tak zaměřit se na vlastní kód a ne na registraci DLL knihovny, zápis do registrů, atd. Tvůrcem frameworku je uživatel vystupující na MSDN fórech pod přezdívkou JohnSpicIE. Projekt se sice dále nevyvíjí a kromě diskuzí na fórech je úplně bez podpory, ale zdrojové kódy jsou k dispozici každému, takže jej může převzít kdokoliv jiný. Během vývoje jsem také objevil, že zveřejněná verze obsahuje několik zásadních chyb, které jsem byl nucen opravit, aby správně fungovaly veškeré důležité funkce rozšíření, jak bude zmíněno v implementaci 6.9. Více informací o frameworku najdete na domovské stránce [22].

4.4 XML

Obecný značkovací jazyk XML (*Extensible Markup Language*) definuje pravidla kódování dokumentů a je navržený tak, aby byl přehledný pro člověka, ale zároveň zpracovatelný počítačem. Byl vytvořen konsorciem W3C (*World Wide Web Consortium*)[1] a standardizován ve verzi 1.0 (aktuální je pátá edice [13]) a 1.1 (aktuální je druhá edice [14]). XML dokumenty obsahují textová data s podporou znaků pro všechny světové jazyky Unicode [33]. V současnosti je to velmi rozšířený jazyk pro vytváření dokumentů, ale i pro přenos dat. Bližší pohled na jazyk XML najdete v [24].

4.5 XHTML

Jazyk webových stránek XHTML (*eXtensible HyperText Markup Language*), vycházející z původního jazyka HTML (*HyperText Markup Language*), je založen na jazyce XML. Oproti HTML tak má mnohem přísnější pravidla [18], například značky (tagy) musí být ukončené. Základ v jazyce XML také umožňuje zpracování dokumentů stejnými prostředky jako zpracování obecného XML dokumentu. Stejně jako XML byl i jazyk XHTML vytvořen a standardizován konsorciem W3C (aktuální je verze 1.1 druhá edice [35] a pracuje se na verzi 5).

4.6 DOM

DOM (*Document Object Model*) je multiplatformní a jazykově nezávislé rozhraní pro práci s XML, XHTML nebo HTML dokumenty. Každý dokument je reprezentován objektovým stromem, s nímž aplikace pracují. Mohou tak měnit jeho obsah, strukturu i styl. První implementace, `DOM Level 0` a `Intermediate DOM`, vznikaly od roku 1996 pro dnes již historické verze prohlížečů Netscape Navigator a Internet Explorer bez standardizace. Teprve v roce 1998 vytvořilo konsorcium W3C první standardizovaný DOM, označovaný `DOM Level 1`. Nyní je většinou prohlížečů podporován nejnovější `DOM Level 3` standard, který oproti prvnímu a druhému přinesl řadu nových možností a funkcí, jak je popsáno v seznamech změn na [8].

[1] http://www.w3.org/

4.7 JavaScript/JScript

Objektově orientovaný skriptovací jazyk JavaScript, většinou používaný na webových stránkách, byl pod společností Netscape vytvořen Brendanem Eichem[2]. V roce 1997 byl jazyk pod jménem ECMAScript standardizován se specifikací ECMA-262 (aktuální je edice 5.1 [12]). Jeho implementace je využívána většinou současných webových prohlížečů. Internet Explorer sice z ECMAScriptu také vychází, ale má vlastní implementaci pod jménem JScript [23], která není úplně kompatibilní s ostatními implementacemi JavaScriptu. Internet Explorer se automaticky ke všemu JavaScriptu chová, jako by se jednalo o JScript, a může tak docházet k problémům, kdy v ostatních prohlížečích kód funguje, zatímco v Internet Exploreru dochází k chybám. Naštěstí jde o minimum případů a lze tak psát standardní JavaScriptový kód s případnými drobnými úpravami pro zajištění kompatibility s JScriptem. Více o JavaScriptu se dočtete například v [15].

4.8 XPath

Dotazovací jazyk XPath (*XML Path Language*) slouží k jednoznačnému výběru elementů z XML dokumentu na základě jejich pozice ve stromové struktuře. Standard XPath 1.0 byl vytvořen konsorciem W3C v roce 1999 [37] a v současnosti je široce používán. Aktuální verze je XPath 2.0 [36], která přináší řadu novinek, ale v porovnání s první verzí také mění některé základní koncepty. Anotační server 4A předpokládá určení pozice fragmentu dokumentu právě v jazyce XPath, přičemž umožňuje využít pouze některé vlastnosti tohoto jazyka, který je tak pro projekt velmi důležitý. Na ukázku zde uvedu jednoduchý příklad.

XPath výraz `/HTML[1]/BODY[1]/DIV[2]/P[1]/text()[1]` označuje při čtení zprava první textový uzel v prvním elementu `P` ve druhém elementu `DIV` v prvním elementu `BODY` v prvním elementu `HTML`, což ve stromové reprezentaci odpovídá následujícímu:

```
|-> Element (<HTML>)
  |-> Element (<BODY>)
    |-> Element (<DIV>)
      |-> Element (<H1>)
        |-> Textový uzel
    |-> Element (<DIV>)
      |-> Element (<P>)
        |-> Textový uzel              <== vybraný uzel
        |-> Element (<SPAN>)
          |-> Textový uzel
        |-> Textový uzel
```

4.9 HTTP

Aplikační protokol HTTP (*Hypertext Transfer Protocol*) je základním stavebním kamenem dnešního Internetu. Jeho prostřednictvím komunikuje klientská aplikace (obvykle webový prohlížeč) se serverem, na kterém se nacházejí požadované informace, respektive data. Přestože se protokol HTTP používal už delší dobu předtím, konce vývoje standardu se dočkal až ve verzi 1.1 v roce 1999 pod RFC-2616 [19] díky organizaci IETF (*Internet Engineering*

[2] http://brendaneich.com/

Task Force)[3] a W3C konsorciu. Jelikož je protokol bezstavový (komunikace se skládá pouze z dvojice zpráv požadavek – odpověď), musí se o správu sezení (uchování stavu) obvykle postarat serverová aplikace využívající protokol.

4.10 Microsoft Visual Studio

Produkt Microsoft Visual Studio je integrované vývojové prostředí od společnosti Microsoft pro operační systém Windows. Umožňuje tvorbu aplikací od konzolových, přes aplikace s grafickým rozhraním, až po webové stránky. Lze v něm vyvíjet aplikace pro osobní počítače, přenosná zařízení, mobilní telefony a další zařízení. Jeho velkým přínosem je návrhář zobrazení, díky kterému lze prvky uživatelského rozhraní snadno rozmístit tak, jak je potřebujete. Dále obsahuje systém automatického doplňování IntelliSense, jež ocení jak začátečníci (zjistí díky němu například všechny atributy a metody tříd), tak profesionálové, protože nebudou muset psát všechen kód ručně. Kromě zmíněných výhod má uživatel k dispozici mnoho dalších funkcí, o kterých se můžete dočíst například v [30]. Zatím poslední stabilní verzí je Microsoft Visual Studio 2010 [26], která existuje v několika edicích. Jedinou neplacenou je z nich Express, která je k dispozici pro každý programovací jazyk samostatně, zatímco ostatní verze jsou již placené a pro všechny programovací jazyky mají jedno společné prostředí.

4.11 Microsoft Windows Installer

Microsoft Windows Installer [34] je služba operačního systému Windows, která umožňuje instalaci, údržbu a odinstalaci software distribuovaného v balíčcích MSI (soubor s příponou .msi). Instalace balíčku lze spouštět s různými parametry a můžete tak docílit například bezobslužné instalace nebo vynucení restartu počítače po dokončení instalace. Také je možné balíčky integrovat do instalačního média operačního systému Windows, takže bude balíček nainstalován zároveň s operačním systémem. První verze (Microsoft Installer 1.0) byla vydána spolu s produktem Microsoft Office 2000, zatímco pozdější verze již byly součástí modernějších systémů Windows. Například Microsoft Windows 7 má v sobě zabudovaný zatím nejnovější Windows Installer 5.0.

[3] http://www.ietf.org/

Kapitola 5

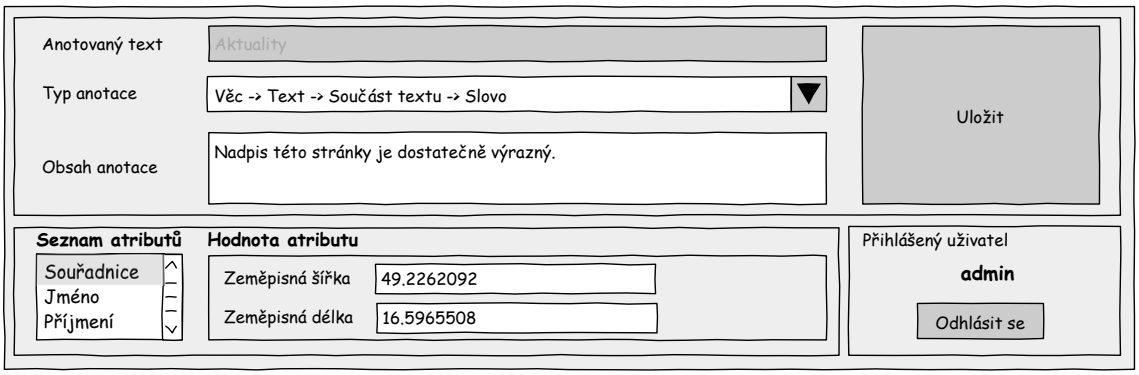
Návrh doplňku

Pro tvorbu doplňku bylo podstatné navrhnout nejdříve uživatelské rozhraní pro komunikaci s uživatelem. To se bude z velké části skládat z formulářových prvků, pro které bylo nutné vytvořit pravidla validace. Také bylo nezbytné připravit překlad doplňku do dalších jazyků a zjistit, které údaje bude doplněk za běhu ukládat v klientském počítači pro použití při dalších spuštěních. Následovalo zvolení systému komunikace doplňku s anotačním serverem. Nakonec bylo vhodné promyslet případné problémy s implementací, aby jim bylo možné předejít.

5.1 Uživatelské rozhraní

Doplněk byl od počátku koncipován jako rozšiřující panel prohlížeče (bar), který bude obsahovat jednotlivé interaktivní prvky grafického rozhraní pro komunikaci s uživatelem. Z požadavků na doplněk, popsaných v kapitole 3, by rozhraní doplňku mělo obsahovat následující prvky:

- přihlašovací panel,
  - pole pro zadání uživatelského jména a hesla,
  - pole pro nastavení parametrů připojení,
  - tlačítko pro přihlášení uživatele,
- anotační panel,
  - informace o přihlášeném uživateli,
  - tlačítko pro odhlášení uživatele,
  - pole pro zobrazení anotovaného textu,
  - výběr typu anotace s funkcí automatického doplňování,
  - seznam atributů dle vybraného typu anotace,
  - pole pro zadávání hodnot jednotlivých atributů,
  - textové pole pro vlastní obsah anotace,
  - tlačítko pro uložení anotace,
- okno pro nastavení parametrů nových/upravovaných atributů,



Obrázek 5.1: Návrh grafického uživatelského rozhraní anotačního panelu

- pole pro zadání jména a výběr typu atributu,
- zaškrtávací políčko pro nastavení, zda bude atribut povinný.

Přihlašovací panel bude kromě vyplnění uživatelského jména a hesla umožňovat měnit parametry připojení k serveru a měl by předcházet zobrazení anotačního panelu. Bude tedy překrývat panel pro vytváření anotace a odkryje jej teprve po přihlášení uživatele. Při odhlášení uživatele bude postup opačný, takže přihlašovací panel opět překryje anotační panel a nepřihlášenému uživateli tak zabrání v provádění nepřípustných úprav. Návrh grafického rozhraní anotačního panelu je pak vidět na obrázku 5.1.

5.2 Prvky pro hodnoty atributů

Ke každému typu anotace může být přiřazeno velké množství atributů. Každý takový atribut má vlastní typ, který určuje, jaký druh hodnoty bude atribut obsahovat. Aby bylo možné upravovat hodnoty všech atributů, bylo by potřeba pro každý z nich vytvořit vlastní sadu formulářových prvků na úpravu hodnoty. To se může zdát jako dobré řešení, protože formulářové pole každého atributu tak v sobě bude přímo obsahovat hodnotu atributu. Tato varianta má ale jednu zásadní nevýhodu, a to takovou, že pokud bude anotace zahrnovat větší množství atributů, bude množství odpovídajících formulářových prvků také vysoké a důsledkem bude velmi negativní vliv na množství využité paměti.

Alternativní variantou, pro kterou jsem se rozhodl, je využití toho, že existuje jen několik základních typů atributů, a že pokud jsou atributy stejného základního typu, musí také splňovat stejná validační pravidla. Díky tomu bude stačit vytvořit sadu formulářových prvků vždy jen pro každý typ atributu a jednotlivé atributy tohoto typu pak budou využívat pouze jednu společnou sadu prvků. Jak ale může být zřejmé, má toto řešení nevýhodu v tom, že hodnota atributu se upravuje pro několik různých atributů ve stejném poli. Abych tento nedostatek obešel, bude nutné mít hodnoty atributů samostatně uložené mimo formulářové prvky a vždy před přechodem na jiný atribut uložit hodnotu aktuálního atributu z pole do úložiště hodnot a načíst z úložiště do pole hodnotu toho atributu, na který přecházím. V porovnání s první variantou bude tedy potřeba postarat se o ukládání a načítání hodnot atributů, ale množství spotřebované paměti pro formulářové prvky bude minimální a vždy konstantní.

5.3 Uživatelské nastavení

Aby uživatel nemusel pokaždé při přihlašování vyplňovat ty samé údaje, bude mít na přihlašovací obrazovce k dispozici zaškrtávací políčko, pomocí kterého si sám nastaví, zda mají být adresa serveru, číslo portu a uživatelské jméno uloženy i pro příští použití, anebo mají být při odhlášení zapomenuty a nahrazeny dříve uloženými (respektive výchozími). Toto nastavení by se mělo ukládat do konfiguračního souboru a z toho důvodu nebude možné ukládat i uživatelské heslo, které by tak mohlo být snadno zneužitelné, popřípadě i zjistitelné jiným člověkem.

5.4 Validace formulářových polí

Pro většinu kontrolních prvků v rámci celého doplňku budou platit určitá pravidla pro validaci, bez kterých nebudou vyplněné hodnoty platné a budou zamítnuty. Některé panely budou k validaci obsahovat jen minimum prvků, zatímco jiné jich budou obsahovat více a navíc budou mít i závislosti mezi sebou. Podle množství validovaných prvků bude doplněk informovat o chybách různým způsobem, a to buď jen textovou zprávou, nebo i změnou barvy příslušných polí.

5.4.1 Pole přihlašovacího panelu

Přihlašovací panel bude obsahovat jen několik prvků, jejichž validace bude z části zajištěna automaticky díky povaze ovládacích prvků. Žádné z polí nesmí zůstat nevyplněné, což bude jedno ze základních pravidel validace celého panelu. Pole pro zadání adresy serveru by mělo být omezeno pouze na vstupní znaky platné pro IP adresu nebo hostname (textový řetězec, který se překládá na IP adresu). Pole pro zadání portu bude moci obsahovat pouze celé číslo z rozsahu 0-65 535, což by mělo být možné zajistit automaticky pomocí komponenty `NumericUpDown`. Zbylé dvě pole pro jméno a heslo uživatele nepotřebují žádnou validaci, protože uživatelské jméno i heslo může být jakékoliv, nicméně budou omezeny maximální délkou na 255 znaků. Určitým způsobem validace bude také přímo autentifikace uživatele – pokud totiž uživatel zadá špatnou kombinaci přihlašovacího jména a hesla, nebude přihlášení akceptováno. O všech případných porušeních validačních pravidel bude uživatel informován po kliknutí na přihlašovací tlačítko.

5.4.2 Pole anotačního panelu

Panel pro tvorbu anotace bude mít komplexnější systém validace. Základní pravidlo bude ovšem podobné jako u přihlašovacího panelu – některé prvky nebudou moci zůstat nevyplněné, a to pole pro zobrazení anotovaného textu (odpovídá vybraným fragmentům na stránce), pole pro výběr typu anotace, a pokud jsou k vybranému typu přiřazené povinné atributy, musí být zadána i jejich hodnota. Dále pole pro výběr typu bude muset obsahovat textovou hodnotu odpovídající linearizované reprezentaci existujícího typu anotace ze stromu typů na serveru. V případě, že uživatel nebude chtít použít existující typ, ale místo toho vytvořit nový, bude mít možnost použít alfanumerické znaky, podtržítka nebo tečky ve jméně typu a pro oddělení typů v linearizované cestě stromem kombinaci znaků „->“. Obsah anotace bude volitelný a bude pro něj platit pouze omezení dané ovládacím prvkem. Předchozí validační pravidla budou kontrolována až při pokusu o uložení nové anotace, zatímco validace hodnot atributů bude probíhat okamžitě po jejich zadání. Všechna

pole, která nebudou splňovat validační pravidla, budou barevně zvýrazněna a bude k nim přidán vysvětlující text chyby, aby uživatel věděl, kde k ní došlo a také jak má tuto chybu opravit.

Pravidla pro pole základních typů atributů z kapitoly 3.2:

- Textový řetězec – bude mít pouze omezení maximální délky na 255 znaků, nezávisle na obsahu.
- Celé číslo – může obsahovat pouze číslice, znaky plus a mínus a hodnota je omezena velikostí typu `Integer` (-2 147 483 648 - 2 147 483 647).
- Desetinné číslo – bude umožňovat vložení číslic, znaků plus, mínus a čárka/tečka a výsledná hodnota bude muset mít maximálně 5 číslic před desetinnou čárkou (čísla odpovídající hodnotám mezi -100 000 a 100 000).
- Datum – může obsahovat pouze reálná data z kalendáře udávající rok, měsíc a den.
- Čas – hodnoty mohou být pouze reálné časové konstanty udávající hodiny, minuty a sekundy.
- Datum a čas – bude obsahovat jak výše uvedené datum, tak i čas, takže musí zahrnovat rok, měsíc, den, hodiny, minuty i sekundy.
- URI identifikátor – musí obsahovat schéma (například http, https, ftp, atd.), dvojtečku a dále hierarchickou část, která je již víceméně libovolná[1].
- Pravdivostní hodnota – není potřeba validovat, protože bude k dispozici jen přepínač „Ano“ a „Ne“.
- Zeměpisná poloha – zeměpisná šířka a délka se ukládají jako desetinná čísla, takže se mohou opět skládat z číslic, znaků plus, mínus a čárka/tečka a výsledná hodnota bude muset mít maximálně 7 číslic před desetinnou čárkou (čísla odpovídající hodnotám mezi -10 000 000 a 10 000 000). I když skutečné hodnoty tak vysokých čísel zdaleka nedosahují, server i databáze je umožňují uložit, takže doplněk je bude také podporovat. Dále pokud je zeměpisná šířka nebo délka vyplněná, musí být zadána i druhá hodnota.

Výše uvedená pravidla budou uživateli zabraňovat opustit dané pole (v případě zeměpisné polohy skupinu polí), dokud nebude jeho hodnota opravena na korektní hodnotu nebo úplně vymazána.

5.4.3 Pole okna pro nastavení parametrů nových/upravovaných atributů

Podobně jako u přihlašovacího panelu bude i zde jen málo prvků k validaci, ale opět bude platit, že většina prvků musí být vyplněna. Jméno atributu musí být zadáno a mít maximální délku 255 znaků. Obsahovat může pouze alfanumerické znaky, podtržítka, tečky, pomlčky a mezery. Dále musí být zvolený jeden ze základních typů atributů. A nakonec by mělo být zaškrtávací políčko pro určení, zda bude atribut povinný nebo ne, nicméně jej není potřeba validovat, protože bývá jen dvoustavové. V případě úpravy již existujícího atributu nebude možné měnit jeho jméno, nicméně ostatní pravidla budou platit stále.

[1] Kompletní specifikaci najdete na http://tools.ietf.org/html/rfc3986

5.5 Seznam typů anotací

Jak již bylo zmíněno, typy anotací jsou na serveru uloženy ve formě stromu typů. To znamená, že všechny typy, kromě kořenových, mají svůj rodičovský typ a mohou tak vytvářet víceúrovňovou strukturu. Výběr typu anotace uživatelem však bude zajištěn prostřednictvím rozbalovacího seznamu (ComboBox), ve kterém budou jednotlivé typy reprezentovány jako textové položky. Pro zobrazení tedy bude potřeba jejich linearizované jméno. To lze pro určitý typ získat tak, že se pomocí kombinace znaků „->“ spojí jméno typu a linearizované jméno jeho rodiče. Jde tak o rekurzivní vyhodnocení, kde je ukončující podmínkou pro zastavení zanořování nalezení nadřazeného typu, který již nemá rodiče, je tedy kořenový a jeho linearizované jméno odpovídá standardnímu jménu. Ve stejném formátu je také nutné zadávat i jméno nového typu, pokud jej chce uživatel přidat.

5.6 Podpora více jazyků

S uživatelským rozhraním úzce souvisí i jazyk, jakým bude doplněk s uživatelem komunikovat. Díky podpoře v Microsoft Visual Studio je možné vytvořit překladové soubory pro jednotlivé jazyky. Doplněk tedy bude v základu obsahovat výchozí neutrální jazyk angličtinu a pak přídavnou češtinu. Jazyk se automaticky zvolí podle národního prostředí operačního systému, pod kterým bude Internet Explorer s anotačním doplňkem spouštěn. Další jazykové lokalizace bude samozřejmě možné přidávat, a to i dodatečně, když už bude doplněk zkompilovaný. Překladové soubory se totiž kompilují do samostatných DLL knihoven. Zejména pro překladatele pak bude k vytvoření jazykových lokalizací vhodnější použít nástroje z balíku Microsoft Windows SDK for Windows 7 and .NET Framework 4 [27], protože ten je celý zdarma, jeho instalace je mnohem menší a rychlejší než Visual Studia a z hlediska překladu poskytuje stejné funkce v jednodušším a přehlednějším rozhraní.

5.7 Komunikace se serverem

Doplněk je navržen pro spolupráci se serverem projektu 4A Framework. Komunikace mezi nimi bude probíhat metodou POST prostřednictvím protokolu HTTP ve formátu XML na základě komunikačního protokolu daného specifikacemi projektu a popsaného výše v podkapitole 3.4. Počátkem komunikace mezi doplňkem a serverem bude připojení k serveru. Jako odpověď na požadavek doplněk obdrží ID sezení (sessionID), které se pak pro identifikaci použije při veškeré další komunikaci se serverem. Jakmile je doplněk připojený, přihlásí se uživatel a doplněk získá jeho jméno a ID. Ihned po přihlášení (popřípadě po načtení nové stránky, pokud je již uživatel přihlášen) proběhne synchronizace dokumentu. Ta zahrnuje zaslání kopie anotovaného dokumentu na server a získání URI anotované kopie. Následně doplněk ze serveru načte strom typů anotací.

Od synchronizace dokumentu a načtení typů se již bude komunikace odvíjet od činností uživatele. Pokud uživatel vytvoří nový typ, zašle se požadavek ihned na server a strom typů anotací se zaktualizuje tak, že bude obsahovat všechny nově vytvořené typy (kromě typu samotného to budou i všechny do té doby neexistující rodičovské typy). Při změně typu anotace se automaticky načtou ze serveru jeho atributy. V případě, že to bude uživatel požadovat, se změny v atributech daného typu anotace uloží na server. Mezi změny v atributech patří přidání a odebrání atributu, změna typu atributu nebo nastavení či zrušení příznaku, který určuje, zda je atribut vyžadován. K další výměně informací pak dojde při

uložení anotace, kdy se odešlou všechny anotované fragmenty, vybraný typ anotace, vyplněné atributy, obsah anotace, autor a další údaje. Posledním typem komunikace pak bude odhlášení uživatele.

5.8 Sekce CDATA v těle HTML dokumentu

Obvyklým problémem při zanoření jednoho dokumentu se značkami (tagy) do druhého je odlišení koncové značky v těle vkládaného dokumentu od koncové značky v těle dokumentu, do kterého je vkládán. Příkladem toho je element CDATA, který může být obsažen v těle HTML dokumentu. Při přenosu anotace v XML formátu pak vyvstane problém, protože HTML dokument s elementem CDATA by měl být vložen v jiném CDATA elementu, což specifikace zakazuje. Obejít tento problém se dá pomocí rozdělení těla dokumentu do více CDATA elementů tak, aby koncové značky elementu CDATA byly rozděleny do dvou samostatných elementů CDATA [9].

Kapitola 6

Implementace výsledného návrhu

Při implementaci jsem vycházel z navrženého řešení, nicméně během vývoje jsem narazil na několik více či méně závažných problémů, které bylo potřeba vyřešit pro úspěšnou realizaci návrhu.

6.1 Vizuální styly

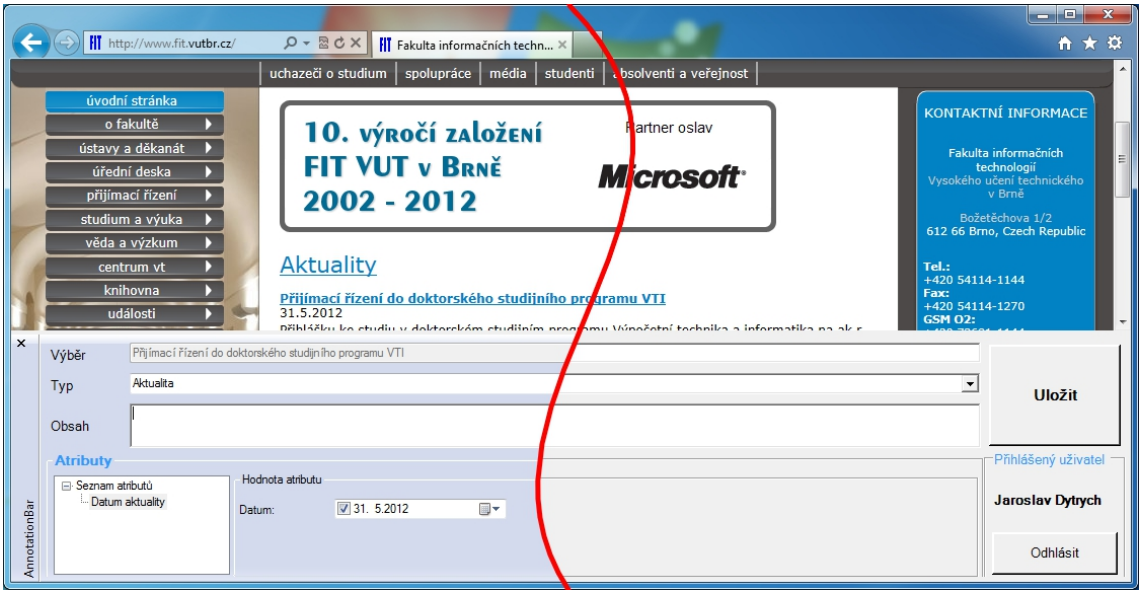
Již od ukázkových příkladů frameworku SpicIE všechna rozšíření vypadala jako vytržená ze starších verzí Windows (95 a 98). Nejvíce to bylo znát na tlačítkách a okrajích ovládacích prvků. V diskuzi na stránkách projektu to již dříve řešilo více programátorů a opravit to měla metoda EnableVisualStyles, která však podle dokumentace na MSDN na prvky v Internet Exploreru nemá vliv [4]. Některým uživatelům ale fungovala, takže jsem její použití na doplňku také vyzkoušel, nicméně jen s částečným úspěchem. Ovládací prvky pak totiž vypadaly jako ostatní prvky prohlížeče, ale samotný panel měl stále našedlou barvu, přestože v návrhu zobrazení, ani jinde v kódu, nebyla nastavena. Nakonec jsem ale zjistil, že pokud sám barvu nezvolím, je předdefinována šedá. Nastavil jsem tedy barvu v kódu ručně na průhlednou a dosáhl tím požadovaného vzhledu doplňku, takže vypadá jako součást prohlížeče. Porovnání vzhledů je pro ukázku na obrázku 6.1.

6.2 Jméno objektu pomocníka prohlížeče a vydavatel panelu aplikace

Anotační doplněk se ve skutečnosti skládá ze dvou komponent - objektu pomocníka prohlížeče a panelu aplikace. Ve správě doplňků jsou tedy vidět obě komponenty. Bohužel ve frameworku byly chyby, které způsobovaly, že objekt pomocníka se zobrazoval s chybným jménem a panel aplikace neměl žádného vydavatele, takže podle správce doplňků nepatřily k sobě, což mohlo být pro běžného uživatele matoucí. Obojí se mi podařilo opravit, jak bude popsáno v podkapitole 6.9.

6.3 Získání fragmentů z výběru ve stránce

Jednou z největších komplikací, kterou jsem během návrhu nepředpokládal, byla nedostatečná výbava pro práci s dokumentem v samotném rozhraní MSHTML [20]. Teprve až od Internet Exploreru 9 totiž podporuje získání DOM elementů vybraných uživatelem na



Obrázek 6.1: Anotační panel vlevo se zapnutými a vpravo s vypnutými vizuálními styly

stránce. Ve starších verzích je tak možné získat pouze prostý text a úsek HTML kódu, který je ovšem bez kontextu k ničemu a navíc je jeho obsah deformován, v případě že nevybíráme všechen text z určité značky HTML. Z toho důvodu nebylo možné získat anotované fragmenty z HTML kódu přímo v doplňku. K dispozici ale byla možnost spouštět ze zdrojového kódu doplňku JavaScript, který sice umožňuje získání vybraných DOM elementů také až od Internet Exploreru 9, ale je snadno rozšiřitelný a ve výchozím nastavení je ve všech verzích Internet Exploreru zapnutý (přesto je nutné kontrolovat, zda zapnutý opravdu je). Fragmenty tedy pro anotaci získá JavaScriptový kód a předá je zpět metodě doplňku. Protože není možné takto předat více hodnot najednou, bylo před předáním ještě potřeba všechny fragmenty vhodně serializovat do jednoho textového řetězce. Vzhledem k tomu, že starší verze Internet Exploreru neimplementují některé potřebné JavaScriptové funkce, bylo nutné doplnit je pomocí knihovny IERange [29], ve které jsem ještě musel opravit drobné chyby na základě dvou funkcí z TinyMCE knihovny [31], a upravené knihovny TreeWalker z projektu Mozile [25]. Pro správné fungování se tedy po synchronizaci stránky se serverem připojí ke skriptům stránky kód pro získání fragmentů a v případě, že používaná verze Internet Exploreru je nižší než 9, se vloží i podpůrné knihovny pro doplnění chybějících funkcí.

6.4 Vícejazyčnost

Microsoft Visual Studio podporuje tvorbu vícejazyčných aplikací již v návrhu zobrazení. Je v něm možné přepnout na jiný jazyk a automaticky se vygeneruje (pokud již neexistuje) soubor pro danou lokalizaci. Nepříjemnou skutečností ale je, že pokud se dodatečně rozhodnete přejmenovat některý z prvků grafického rozhraní nebo proměnnou využívající překlady, změna se neprojeví v překladových souborech, ale pouze v neutrálním jazykovém prostředí. Výsledkem pak je, že se Vám ztratí propojení překladu na konkrétní prvek a budete jej muset navázat manuálně, a to ve všech přidaných jazycích. Navíc, pokud si tohoto nedostatku nejste vědomi, je vysoce pravděpodobné, že si chybějícího navázání ani

nevšimnete, dokud nepřepnete v návrháři zobrazení na jiný než neutrální jazyk, popřípadě nespustíte aplikaci v jiném jazykovém prostředí, protože samotné Visual Studio Vás o tom nijak neinformuje.

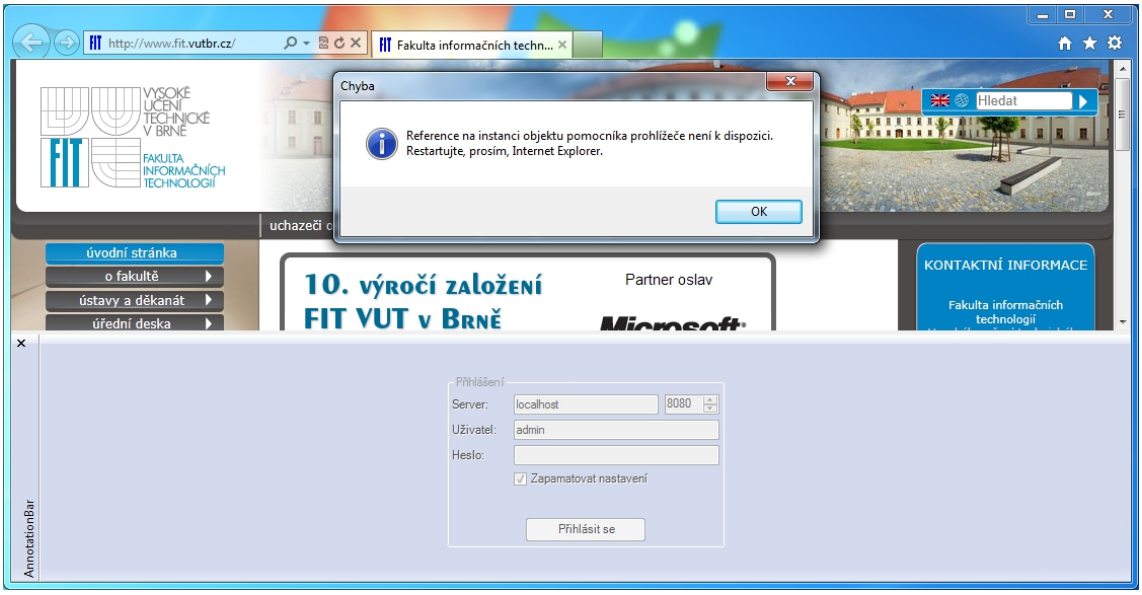
6.5 Klávesy v interaktivních prvcích

Téměř od začátku vývoje doplňku jsem se setkával s nedostatky souvisejícími s nesprávnou funkcí některých speciálních kláves. Chybu klávesy `BackSpace`, která místo mazání vlevo vrátila prohlížeč na předchozí navštívenou stránku, jsem nejdříve obcházel pomocí odchytávání klávesy v každém prvku rozhraní samostatně. Další problém byl s klávesou `Tab`, po jejímž zmáčknutí nepřešel kurzor do dalšího pole podle hodnot `TabIndexu`, ale přeskočil do těla stránky. Obojí se podařilo odstranit opravením chyb ve frameworku SpicIE, jak bude uvedeno v podkapitole 6.9, ovšem u tabulátoru jen částečně. Zatímco u části doplňku fungoval, jak má, u jiné stále přeskakoval mimo. Nakonec jsem přesunul všechny prvky panelu do tabulkového rozložení, což problém vyřešilo, protože předchozí několikanásobné víceúrovňové rozdělení panelů pravděpodobně narušovalo posloupnost hodnot `TabIndex`.

6.6 Ukládání nastavení

Jedním z požadavků na doplněk bylo ukládání nastavení. Vzhledem k tomu, že Microsoft Visual Studio má grafické rozhraní pro vytváření nastavení, rozhodl jsem se jej využít, protože by to znamenalo velké usnadnění práce s nastavením. Bohužel se ukázalo, že kromě usnadnění to přinese i značné komplikace. Původně jsem chtěl využít nastavení aplikace, ale to umožňuje jen načítání nastavení bez jeho ukládání – nastavení aplikace totiž nelze měnit za chodu. Navíc, jak jsem dále zjistil, programy využívající DLL knihovny nemohou z bezpečnostních a jiných důvodů načítat jiná než vlastní nastavení aplikace [3], takže nastavení aplikace knihovny by stejně nepřicházelo v úvahu. Musel jsem tedy přejít k uživatelskému nastavení, ale ani to nebylo bez problémů. Od Windows Vista se totiž v Internet Exploreru objevil chráněný režim, který brání škodlivým aplikacím v instalaci do systému, ale zároveň omezuje i funkce doplňků. Poprvé se uloží uživatelské nastavení bez problémů – vytvoří se příslušný konfigurační soubor a do něj se zapíší data. Během dalšího spuštění se data ze souboru načtou, ale při pokusu o uložení již dojde k chybě. Při dalším ukládání nastavení se totiž nejdříve vytváří dočasný soubor s novými údaji, teprve pak se maže starý konfigurační soubor a nakonec se dočasný soubor přejmenuje na nový konfigurační soubor. K chybě dojde hned při vytváření dočasného souboru, protože aplikace nemá právo k zápisu do jiného souboru než je právě konfigurační soubor. Jediným řešením tedy bylo smazat konfigurační soubor těsně před každým pokusem o uložení nastavení.

Podobný problém také nastane, pokud omylem spustíte Internet Explorer jako „Správce“. V takovém případě se konfigurační soubor automaticky vytvoří mimo chráněný režim a při následujících standardních spuštěních nebude možné toto nastavení přepsat, protože v chráněném režimu nemáte práva ani ke smazání tohoto souboru, natož k vytváření jiného. Nicméně pokud tato situace nastane, zobrazí Vám doplněk cestu ke konfiguračnímu souboru, který je potřeba ručně smazat, aby byla obnovena původní správná funkčnost.



Obrázek 6.2: Problém při prvním spuštění Internet Exploreru 9 po instalaci rozšíření

6.7 Instalátor doplňku

Pokud chcete distribuovat jakýkoliv program mezi běžné uživatele, je pro něj vhodné vytvořit instalátor, aby uživatel nemusel přemýšlet nad tím, kam má který soubor nakopírovat, atd. Microsoft Visual Studio umožňuje vytvoření takového instalátoru a díky podpoře ve frameworku SpicIE je možné při instalaci automaticky zaregistrovat DLL knihovnu jako rozšíření Internet Exploreru. Možností nastavení instalátoru není mnoho, je jen jednojazyčný, ale kladem je, že při vytváření vzniká také balíček pro Windows Installer (MSI soubor) [34], který lze použít pro bezobslužnou instalaci, případně integrovat do instalačního CD/DVD systému Windows.

6.8 První spuštění

V doplňku zůstává chyba, a to problém při prvním spuštění Internet Exploreru 9 a 10 po instalaci rozšíření. Panel aplikace při něm nemá referenci na objekt pomocníka prohlížeče a není tak s ním schopen komunikovat. Proč tomu tak je, souvisí s vyvářením instancí jednotlivých objektů samotným Internet Explorerem [17]. Zatím je to řešeno požádáním uživatele o restartování prohlížeče, protože od dalšího spuštění je již reference k dispozici. Náhled prozatímního řešení je na obrázku 6.2.

6.9 Modifikace frameworku SpicIE

Během práce s frameworkem jsem bohužel zjistil, že obsahuje několik zásadních chyb, které brání správnému fungování vytvářených doplňků.

- V metodě pro detekci stisku kláves byl parametr špatného datového typu, takže klávesa `BackSpace` (pro mazání vlevo) nebyla zachycena příslušným prvkem grafického rozhraní a místo toho způsobila přechod prohlížeče na předchozí navštívenou stránku.

- V těle třídy pro panel prohlížeče chyběly metody pro změnu aktivního prvku, takže v žádném z prvků uživatelského rozhraní, jako jsou textové pole, výběr data a času, přepínače, atd., nefungoval tabulátor pro přepínání mezi nimi na základě `TabIndex` hodnot (pořadí aktivních prvků při použití klávesy tabulátor) a místo toho se stiskem tabulátoru uživatel dostal mimo panel doplňku, tedy do těla zobrazené stránky.

- Třída instalátoru měla zakomentované metody pro registraci doplňku, takže se při instalaci nezapsaly hodnoty do registrů, a rozšíření se tak v prohlížeči vůbec neobjevilo.

- Jméno objektu pomocníka prohlížeče se při registraci doplňku nezapisovalo do registrů, takže se ve správě doplňků zobrazoval pod jménem odpovídajícím vzoru „JmennýProstor.JménoTřídy“ („NameSpace.ClassName“).

- Rozšiřující panel aplikace neměl žádné jméno vydavatele, protože při registraci se k němu přiřadila instance třídy pro zobrazování HTML stránky, která vydavatele opravdu nemá. Jelikož zobrazovat HTML stránku v panelu nepotřebuje každý, umožnil jsem tuto vlastnost vypnout a získat tak správné jméno vydavatele, které odpovídá jménu vydavatele celé DLL knihovny.

Na některé ze zmíněných problémů si stěžovali i mnozí další lidé v diskuzi na stránkách projektu a neexistovalo pro ně většinou žádné řešení. Z toho důvodu jsem se rozhodl zveřejnit opravenou verzi frameworku SpicIE i pro všechny ostatní uživatele. Případní zájemci tak nyní přímo v diskuzi k projektu[1] najdou odkaz na stažení spolu s jednoduchým návodem, jak nahradit standardní verzi touto s opravami, a seznamem provedených změn ve frameworku. Přestože odezva od uživatelů je v diskuzi zatím nulová, byl celý archiv už několikrát stažen, takže věřím, že bude užitečný nejen mně, ale i budoucím uživatelům.

[1] http://archive.msdn.microsoft.com/SpicIE/Thread/View.aspx?ThreadId=5251

Kapitola 7

Testování

V průběhu vývoje doplňku (a zejména k jeho konci) bylo nutné podrobit jej různému testování pro ověření správného fungování. K otestování správnosti komunikace se ukázala být značnou výhodou lokální kopie anotačního serveru projektu 4A Framework, díky které bylo možné okamžitě získat odpověď serveru na požadavek z doplňku. Zásluhou procesu testování jsem našel a opravil několik chyb, z nichž některé by pravděpodobně zůstaly dlouho neodhalené a mohl by na ně narazit až některý koncový uživatel.

7.1 Ověření kompatibility v oficiálně nepodporovaných prostředích

Vzhledem k tomu, že framework SpicIE oficiálně nepodporoval poslední verze nejnovějších vývojových prostředí a platforem (Microsoft Visual Studio 2010, .NET4 Framework a Internet Explorer 9), bylo potřeba nejdříve ověřit, jestli v nich, a zejména v Internet Exploreru 9, bude fungovat. Přenesení doplňku do nového vývojového prostředí i registrace do prohlížeče proběhly úspěšně a bez chyb, ale přechod na platformu .NET4 byl komplikovanější, protože framework SpicIE byl kompilován pomocí starší verze .NET2.0 a nemohl tak být základem projektu doplňku. Bylo tedy potřeba změnit cílovou platformu frameworku, zkompilovat jej pomocí dostupných zdrojových kódů a teprve poté ho bylo možné přidat k projektu.

7.2 Testování vícejazyčnosti a zachování funkcí v různých prostředích

Pro ověření zachování stejné funkčnosti a plné kompatibility v různých verzích Windows s odlišnou jazykovou lokalizací a s různými verzemi Internet Exploreru jsem testoval následující kombinace prostředí:

- Windows XP + CZ + Internet Explorer 6
- Windows XP + EN + Internet Explorer 6
- Windows XP + CZ + Internet Explorer 7
- Windows XP + EN + Internet Explorer 7
- Windows XP + CZ + Internet Explorer 8

- Windows XP + EN + Internet Explorer 8
- Windows Vista + CZ + Internet Explorer 7
- Windows Vista + CZ + Internet Explorer 8
- Windows Vista + CZ + Internet Explorer 9
- Windows 7 + CZ + Internet Explorer 8
- Windows 7 + CZ + Internet Explorer 9
- Windows 8 Consumer Preview + EN + Internet Explorer 10

Na základě výsledků testů v Internet Exploreru 6-8 bylo potřeba několikrát opravovat převzaté JavaScriptové knihovny pro tyto verze, aby výsledný kód odpovídal zabudovaným funkcím v Internet Exploreru 9 a 10. Dále jsem díky testování v Internet Exploreru 10 objevil chybu ve vkládání JavaScriptu k ostatním skriptům do hlavičky dokumentu, stejně jako v jeho zpětném odebírání. Zatímco funkční stránku tedy bylo potřeba upravovat, aby doplněk pracoval ve všech prostředích stejně, uživatelské rozhraní fungovalo všude bez úprav, jen s drobnými rozdíly ve vzhledu, který vždy vycházel z konkrétní verze použitého operačního systému. Stejně tak testování v jiném jazykovém prostředí proběhlo úspěšně.

7.3 Testování lokálních funkcí

Než začal doplněk komunikovat s anotačním serverem, bylo potřeba ověřit, zda fungují korektně funkce a interaktivní prvky, pomocí kterých se získávají výsledná data pro odesílání na server.

Nejdříve bylo nutné zjistit, jestli fragmenty, získávané pomocí volání JavaScriptu, skutečně odpovídají vybraným úsekům textu na stránce. Kontroly jsem dosáhl pomocí porovnání výpisu fragmentů získaných kódem s ručně vyhodnocenými fragmenty (cesta XPath, odpovídající textový řetězec, jeho délka a posunutí) ze stromové struktury dokumentu.

Dalším krokem byla kontrola validace polí pro zadávání hodnot atributů. Pro každý typ atributu platí pravidla uvedená v kapitole 5.4. Tyto kontroly mě upozornily na problém s národním prostředím – pole pro výběr data a času totiž nemají předdefinované nastavení pro zobrazení data i času zároveň, takže bylo nezbytné nastavit vlastní formát, který, jak jsem zjistil, nebyl použitelný například pro anglicky mluvící země. Nakonec tedy využívám pro nastavení vlastního formátu přímo informace z aktuálního národního prostředí operačního systému.

Posledním testem pouze v rámci klientského doplňku pak bylo, zda doplněk správně zkontroluje vyplněnost všech potřebných polí. Každá anotace totiž musí mít zvolený typ, vybrané anotované fragmenty z textu dokumentu a vyplněné ty atributy, které mají nastavené, že musí být uživatelem zadány.

7.4 Sledování komunikace mezi doplňkem a serverem

Velmi podstatným krokem při ověřování správného fungování doplňku bylo sledování komunikace mezi vytvářeným klientským doplňkem a anotačním serverem. Proto jsem nechal doplňkem průběžně ukládat všechny příchozí i odchozí zprávy. Při každé akci doplňku jsem tak měl přehled o tom, jaké informace se zasílají na server, mohl jsem ověřit jejich správnost

a následně vyhodnotit příchozí odpověď serveru. Pro identifikaci chyby se totiž ve většině případů stačilo podívat na odezvu serveru, který již chybu rozpoznal a informoval o ní ve své odpovědi. Mimo jiné sledování komunikace také umožňovalo zkontrolovat, jestli se opravdu odesílají na server všechna data, která se odeslat mají – například při ukládání anotace by se totiž mohlo stát, že server anotaci v pořádku uloží i navzdory tomu, že doplněk neodeslal uživatelem vyplněné volitelné atributy, protože i bez nich se jedná o platnou anotaci. Těmito postupy jsem si tedy ověřil správnost sestavení požadavků pro přihlašování a odhlašování uživatele, synchronizaci dokumentu, získávání a ukládání typů anotace, přidávání, odebírání a modifikaci atributů a nakonec i ukládání nové anotace.

Na druhou stranu analýza ukládané komunikace oddálila nalezení jiné chyby. Uložené záznamy komunikace se serverem byly sice ve správném kódování Unicode (konkrétně UTF-8), přesto se však anotace na serveru ukládaly se špatnou diakritikou. Problém byl nakonec v přenosu dat přes protokol HTTP, který probíhal nezávisle na uložení do souboru a který nebyl zakódovaný v Unicode, ale ve výchozím kódování systému Windows (windows-1250, případně cp1250).

Kapitola 8

Závěr

Počátkem mé práce bylo shromáždění co nejvíce informací o možnostech vývoje rozšíření pro Internet Explorer. V příštím kroku jsem se pak musel rozhodnout, které nástroje budou tím nejlepším kompromisem mezi využitelností a dostupností. Zároveň se sbíráním poznatků o tvorbě doplňků jsem také zjišťoval, co je to anotace, co je strukturovaná anotace a co musí takové anotace obsahovat. Následovalo seznámení se s projektem 4A Framework, pro který byl doplněk vyvíjen a s jehož anotačním serverem měl doplněk spolupracovat. Z toho také plynula nutnost prostudovat komunikační protokol, který je rovněž součástí projektu 4A Framework a který je nezbytný pro správné dorozumívání mezi anotačním serverem a klientským anotačním doplňkem pro prohlížeč. Další částí pak byl návrh doplňku, aby splňoval požadavky dané zadáním, příprava validačních pravidel pro formulářové prvky rozhraní a promyšlení některých potenciálních problémů v budoucí implementaci.

Jakmile byly všechny přípravné kroky dokončené, začal jsem s implementací navrženého řešení. Během ní jsem narazil na množství problémů, které jsem v návrhové části nepředpokládal. Kromě drobného problému po instalaci doplňku při prvním spuštění Internet Exploreru 9 a 10, jsem však všechny úspěšně vyřešil. Na základě úprav v kódu frameworku SpicIE jsem také zveřejnil pro ostatní uživatele opravenou verzi tohoto frameworku.

Již první implementace doplňku bylo potřeba podrobit testování a na základě výsledků testů je zpětně opravovat. Další testování, tentokrát už důkladnější, podstoupil doplněk v závěrečné fázi, a to zejména se zaměřením na různá prostředí, pod kterými by měl pracovat. Po dokončení testování doplněk splňoval všechny body zadání.

Možností pro budoucí rozšíření doplňku je více, nicméně se dají rozdělit do dvou kategorií, a to rozšíření funkcí anotačního panelu a vizualizace existujících anotací. Rozšíření funkcí by mohlo zahrnovat například práci s uživatelskými skupinami, možnost vytvářet vnořené anotace, načítání nastavení ze serveru nebo přidání dalších jednoduchých typů atributů. Vizualizace anotací by musela spoléhat na vkládání přídavného HTML kódu do těla stránky pomocí JavaScriptu, protože jiné vizualizační prostředky nejsou pro práci s dokumentem v rozšíření k dispozici. Tato část by ale pravděpodobně mohla být s modifikacemi převzata z již téměř dokončeného doplňku pro prohlížeč Mozilla Firefox, protože rozšíření pro něj jsou z velké části také psány právě pomocí JavaScriptu.

Literatura

[1] .NET Framework Conceptual Overview. Microsoft Corporation, [Online; navštíveno 26.04.2012].
URL http://msdn.microsoft.com/en-us/library/zw4w595w.aspx

[2] Add-in Express™ for Internet Explorer® and Microsoft® .net. Add-in Express Ltd., [Online; navštíveno 26.04.2012].
URL http://www.add-in-express.com/programming-internet-explorer/

[3] Application Settings. Microsoft Corporation, [Online; navštíveno 26.04.2012].
URL http://msdn.microsoft.com/en-us/library/a65txexh.aspx

[4] Application.EnableVisualStyles Method. Microsoft Corporation, [Online; navštíveno 25.04.2012].
URL http://msdn.microsoft.com/en-us/library/system.windows.forms.application.enablevisualstyles.aspx

[5] Browser Extensions. Microsoft Corporation, [Online; navštíveno 24.04.2012].
URL http://msdn.microsoft.com/en-us/library/aa753587.aspx

[6] C# Language Specification. Ecma International, červen 2006, [Online; navštíveno 26.04.2012].
URL
http://www.ecma-international.org/publications/standards/Ecma-334.htm

[7] COM: Component Object Model Technologies. Microsoft Corporation, [Online; navštíveno 26.04.2012].
URL http://www.microsoft.com/com/default.mspx

[8] Document Object Model (DOM). World Wide Web Consortium (W3C), [Online; navštíveno 26.04.2012].
URL http://www.w3.org/DOM/

[9] Dunn, N.: Nesting CDATA Blocks. Webucator, Inc., 20. listopadu 2010, [Online; navštíveno 26.04.2012].
URL
http://web-design.blogs.webucator.com/2010/11/20/nesting-cdata-blocks/

[10] Dytrych, J.: 4A Framework (Annotations Anywhere, Annotations Anytime) created in NLP@FIT. [Online; navštíveno 24.04.2012].
URL http://pcnlp9.fit.vutbr.cz/annotations/

[11] Dytrych, J.: 4A Protocol Specification. [Online; navštíveno 24.04.2012].
URL http://pcnlp9.fit.vutbr.cz/annotations/4A_protocol_1_1_en.html

[12] ECMAScript Language Specification. Ecma International, červen 2011, [Online; navštíveno 26.04.2012].
URL
http://www.ecma-international.org/publications/standards/Ecma-262.htm

[13] Extensible Markup Language (XML) 1.0 (Fifth Edition). World Wide Web Consortium (W3C), 6. listopadu 2008, [Online; navštíveno 26.04.2012].
URL http://www.w3.org/TR/REC-xml/

[14] Extensible Markup Language (XML) 1.1 (Second Edition). World Wide Web Consortium (W3C), 16. srpna 2006, aktualizováno 29. září 2006, [Online; navštíveno 26.04.2012].
URL http://www.w3.org/TR/2006/REC-xml11-20060816/

[15] Flanagan, D.: *JavaScript: the definitive guide*. Sebastopol: O'Reilly, páté vydání, 2006, ISBN 0-596-10199-6, 994 s.

[16] Goldberg, J.: What's new in .NET Framework 4 Client Profile RTM. Microsoft Corporation, [Online; navštíveno 27.04.2012].
URL http://blogs.msdn.com/b/jgoldb/archive/2010/04/12/
what-s-new-in-net-framework-4-client-profile-rtm.aspx

[17] uživatel Gunnar-D: SpicIE: Understanding the browser extension creation model. Microsoft Corporation, [Online; navštíveno 26.04.2012].
URL http://blogs.msdn.com/b/mtcmuc/archive/2009/04/09/
spicie-understanding-the-browser-extension-creation-model.aspx

[18] HTML vs XHTML. W3Schools, [Online; navštíveno 26.04.2012].
URL http://www.w3schools.com/html/html_xhtml.asp

[19] Hypertext Transfer Protocol – HTTP/1.1. The Internet Engineering Task Force, [Online; navštíveno 26.04.2012].
URL http://tools.ietf.org/html/rfc2616

[20] Internet Explorer Architecture. Microsoft Corporation, [Online; navštíveno 25.04.2012].
URL http://msdn.microsoft.com/en-us/library/aa741312.aspx

[21] ISO/IEC 23270:2006. ISO - International Organization for Standardization, 2006, [Online; navštíveno 26.04.2012].
URL http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_
detail.htm?csnumber=42926

[22] uživatel JohnSpicIE: SpicIE - Simple Plug-In Creator for Internet Explorer. Microsoft Corporation, 15. dubna 2009, [Online; navštíveno 25.04.2012].
URL http://archive.msdn.microsoft.com/SpicIE

[23] JScript (ECMAScript3). Microsoft Corporation, [Online; navštíveno 04.05.2012].
URL http://msdn.microsoft.com/en-us/library/hbxc2t98.aspx

[24] Kosek, J.: *XML pro každého: podrobný průvodce*. Praha: Grada, první vydání, 2000, ISBN 80-7169-860-1, 163 s.

[25] Lapy, R.: Mozile (xhtml editing in your browser). Mozdev Community Organization, [Online; navštíveno 25.04.2012].
URL http://mozile.mozdev.org/

[26] Microsoft Visual Studio 2010. Microsoft Corporation, [Online; navštíveno 27.04.2012].
URL http://www.microsoft.com/cze/msdn/vstudio/2010/

[27] Microsoft Windows SDK for Windows 7 and .NET Framework 4. Microsoft Corporation, [Online; navštíveno 06.05.2012].
URL http://www.microsoft.com/en-us/download/details.aspx?id=8279

[28] Programming Languages for the .NET Framework. Microsoft Corporation, [Online; navštíveno 27.04.2012].
URL http://msdn.microsoft.com/en-us/vstudio/dd643383

[29] Ryan, T. C.: W3C DOM Ranges for IE. Google Project Hosting, duben 2009, [Online; navštíveno 25.04.2012].
URL http://code.google.com/p/ierange/

[30] Sharp, J.: *Microsoft Visual C# 2010: krok za krokem*. Brno: Computer Press, první vydání, 2010, ISBN 978-80-251-3147-3, 696 s.

[31] TinyMCE - Javascript WYSIWYG Editor. Moxiecode Systems AB., [Online; navštíveno 25.04.2012].
URL http://www.tinymce.com/

[32] Troelsen, A. W.: *Pro C# 2010 and the .NET 4 platform*. New York: Apress, páté vydání, 2010, ISBN 978-1-4302-2549-2, 1712 s.

[33] What is Unicode? Unicode, Inc., [Online; navštíveno 26.04.2012].
URL http://www.unicode.org/standard/WhatIsUnicode.html

[34] Windows Installer. Microsoft Corporation, [Online; navštíveno 26.04.2012].
URL
http://msdn.microsoft.com/en-us/library/windows/desktop/cc185688.aspx

[35] XHTML™ 1.1 - Module-based XHTML - Second Edition. World Wide Web Consortium (W3C), [Online; navštíveno 26.04.2012].
URL http://www.w3.org/TR/2010/REC-xhtml11-20101123/

[36] XML Path Language (XPath) 2.0 (Second Edition). World Wide Web Consortium (W3C), 14. prosince 2010, aktualizováno 3. ledna 2011, [Online; navštíveno 26.04.2012].
URL http://www.w3.org/TR/xpath20/

[37] XML Path Language (XPath) Version 1.0. World Wide Web Consortium (W3C), 16. listopadu 1999, [Online; navštíveno 26.04.2012].
URL http://www.w3.org/TR/xpath/