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

GENERÁTOR KŘÍŽOVEK

CROSSWORD GENERATOR

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — MARTIN POHNER

AUTHOR

VEDOUCÍ PRÁCE — Ing. RUDOLF KAJAN,

SUPERVISOR

BRNO 2010

Abstrakt

Tato bakalářská práce řeší vývoj webové aplikace pro generování švédských křížovek. Je zde popsána a vysvětlena technologie Silverlight od firmy Microsoft, ve které je aplikace vyvíjena, návrh uživatelského rozhraní a implementace generátoru. Výsledná aplikace je spustitelná ve webovém prohlížeči a umožňuje uživateli vytvářet nové křížovky z předlohy např. noviny anebo jejich generování (téměř bez účasti uživatele). Řešení, mazání a různé typy nápověd jsou součástí aplikace.

Abstract

This Bachelor´s thesis deals with development of web application for Swedish crossword generator. There is described and explained Microsoft Silverlight technology, which is application developed in, concept of user interface and implementation of crossword generator. Final application can start in web browser and allows to create a new crossword from copy, for example the newspapers or direct generating (almost without user´s complicity). Solving, deleting and another types of help are a parts of application too.

Klíčová slova

Silverlight, generátor, křížovka, implementace, web.

Keywords

Silverlight, generator, crossword, implementation, website.

Citace

Martin Pohner: Generátor křížovek, bakalářská práce, Brno, FIT VUT v Brně, 2010

Generátor křížovek

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Rudolfa Kajana.

.....
Martin Pohner
18. května 2011

Poděkování

Rád bych poděkoval vedoucímu práce za výbornou spolupráci a za čas, který mi věnoval při konzultačních hodinách.

© Martin Pohner, 2010.
Tato práce vznikla jako školní dílo na Vysokém učení technickém v Brně, Fakultě informačních technologií. Práce je chráněna autorským zákonem a její užití bez udělení oprávnění autorem je nezákonné, s výjimkou zákonem definovaných případů.

Obsah

1 Úvod **4**

2 Švédská křížovka **5**
2.1 Struktura křížovky 5
2.2 Pravidla struktury křížovky 6
2.3 Definice řetězců křížovky 6
2.4 Body křížení 7
2.5 Seznam řetězců 8

3 Požadavky na aplikaci **9**
3.1 Analýza současných generátorů 9
3.2 Shrnutí 10
3.3 Stanovené požadavky na naši aplikaci 10

4 Návrh řešení generátoru **11**
4.1 Backtracking 11
4.2 Metoda obalování slov 12
4.3 Uspořádání dat ve slovníku 13
4.4 Vyhledávání ve slovníku 14
4.5 Pseudo generátor 14

5 Vývojové nástroje a technologie **15**
5.1 .NET Framework 15
5.2 Microsoft Visual Studio 2010 19
5.3 Microsoft Expression Blend 4 19
5.4 Microsoft SQL Server 19

6 Silverlight **21**
6.1 Historie 21
6.2 Silverlight 22
6.3 Silverlight Toolkit 22
6.4 Schéma architektury 22

7 Implementace aplikace **24**
7.1 Uživatelské rozhraní 24
7.2 Vývojový diagram tříd 25
7.3 Implementace funkcionality 26

8 Testování aplikace **30**
8.1 Test rychlosti generátoru 30

9 Závěr **31**

A Obsah DVD **33**

B Vzhled aplikace **34**

C Vygenerovaná křížovka **35**

D Prázdná křížovka **36**

E Vygenerovaná křížovka **37**

F Prázdná křížovka **38**

Seznam obrázků

2.1 Křížovka s mřížkou 4×4. 6
2.2 Návrh křížovky. 7
2.3 Definice křížovky ve formě dynamického pole. 7
2.4 Seznam bodů křížení ve formě dynamického pole. 8

4.1 Princip metody obalování slov. 13
4.2 Princip naindexování slovníku. 13

5.1 Schéma architektury .NET Frameworku převzato z [3]. 16

6.1 Schéma architektury Silverlight převzato z [1]. 23

7.1 Vývojový diagram tříd 25

B.1 Uživatelské rozhraní aplikace. 34

C.1 Vygenerovaná křížovka. 35

D.1 Křížovka připravená k řešení. 36

E.1 Vygenerovaná křížovka. 37

F.1 Křížovka připravená k řešení. 38

Kapitola 1

Úvod

Řešení křížovek je v dnešní době velmi populární. S křížovkami se setkáváme téměř v každém časopise nebo novinách, ale varianta řešit křížovku na internetových stránkách prozatím není příliš rozšířená. Existující internetové aplikace pro řešení křížovek navíc postrádají takové uživatelské prostředí, které bude uživatelům co nejvíce připomínat řešení křížovek v běžném tisku.

Tento poznatek mě dovedl k nápadu vytvořit aplikaci, která bude umožňovat nejenom čisté řešení, ale také propracovaný návrh křížovek a jejich generování pomocí informačních technologií. První křížovky se objevily v 20. letech 20. století, a protože informační technologie byly ještě v té době na bodu mrazu, vytvářel křížovky člověk. S postupným rozšířením informačních technologií se pro řešení různých úloh začaly využívat počítače místo lidského myšlení. Tato nová možnost v řešení úloh přinesla potřebu předat myšlení člověka počítači tak, aby byl schopen zadaný problém správně řešit. Vědecké odvětví, které se zabývá touto činností, se nazývá umělá inteligence.

Umělou inteligenci lze použít také pro řešení různých hlavolamů a křížovek a pomocí vhodných metod nalézt požadované řešení.

Touto prací bych chtěl ukázat možnosti umělé inteligence pro generování švédských křížovek a aplikovat je pro vytvoření webové aplikace umožňující nejen generování samotných křížovek, ale také jejich co nejreálnější řešení, tak jak je známe z běžných novinových křížovek. Na základě již vytvořených aplikací pro generování křížovek, prostuduji principy a algoritmy, které se pro tuto problematiku používají. Z hlediska algoritmů bude prostudován a implementován algoritmus z oblasti umělé inteligence backtracking, a z principů bude zvolen princip obalování slov.

V závěru této práce bych se chtěl zaměřit na uživatelské prostředí aplikace, na jeho efektivitu, použitelnost vůči uživateli a zhodnotit dosažené výsledky a navrhnout další rozšíření této aplikace.

Kapitola 2

Švédská křížovka

V této kapitole se hlouběji seznámíme se švédskými křížovkami a specifikujeme jejich strukturu viz 2.1 a 2.2. Dále bude vysvětlena reprezentace křížovky na základě genetických algoritmů [2], kde si postupněi popíšeme definice řetězců křížovky 2.3, pole bodů křížení 2.4 a seznam řetězců 2.5.

Švédská křížovka je hlavolam, hádanka, ve které se podle určitých pravidel doplňují písmena. Křížovka se skládá ze vzájemně se protínajících (křížících se) řádků a sloupců, které jsou rozdělené na jednotlivé čtverečky (buňky), do kterých se doplňují písmena. Nalezená slova nebo hlásky se doplňují do řádků nebo sloupců tak, aby všechna slova ve svislém i vodorovném směru měla smysl.

Slova a hlásky se doplňují podle nápovědy uvedené v legendě. Legenda u švédské křížovky musí být její součastí. Cílem křížovky je vyřešit tajenku (skrytý text), která se objeví až po doplnění všech ostatních slov podle legendy a záleží pouze na tvůrci křížovky jakou tajenku zašifruje. Může to být např. nějaký citát.

2.1 Struktura křížovky

Struktura křížovky se skládá z několika základních částí a v následujícím výčtu budou rozebrány.

Popis částí křížovky:

- **Mřížka** – místo, které obsahuje zadání křížovky. Rozměr mřížky je běžně omezen dle velikosti papíru, na kterém je křížovka vytištěna a také podle toho, aby byla dobře čitelná. Tvar křížovky může být obdélníkový, čtvercový anebo různorodý. Minimální rozměr mřížky je však 3×3. Mřížka se skládá z buněk.

- **Buňka** – základní kámen mřížky. Velikost buňky je zvolena tak, aby obsah vložený uvnitř byl dobře čitelný. Existují čtyři typy buněk, jejich výčet je uveden níže.

- **Buňka pro písmeno** – speciální typ buňky, který zůstává prázdný. Řešitel při luštění křížovky vyplňuje tyto buňky písmeny. Vepsaná písmena v těchto buňkách udávají slova v jednotlivých sloupcích a řádcích.

- **Buňka pro tajenku** – téměř totožná buňka s buňkou pro písmeno jen s tím rozdílem, že má například jiné podbarvení. Tyto pole odhalí tajenku při vyřešení křížovky.

- **Legenda** – speciální typ buňky, který obsahuje nápovědu pro hledané slovo v daném řádku nebo sloupci. Může také definovat jaký sloupec či řádek je tajenkou.

- **Nápověda** – speciální buňka, která bývá obvykle umístěna v některém z rohů mřížky. Obsahuje teoreticky nejobtížnější části křížovky, zhruba tak tři až pět slov a je na řešiteli, aby je správně umístil do křížovky.

NÁPOVĚDA: AAK	ČLUN RÝNSKÝ	MĚSTO FINSKÉ	AUSTR. TISKOVÁ AGENTURA
AMATÉRSKÁ ATLETASOCIACE	A	A	A
PILOT	A	B	A
TAJENKA	K	O	P

(a) Vyřešená křížovka

NÁPOVĚDA: AAK	ČLUN RÝNSKÝ	MĚSTO FINSKÉ	AUSTR. TISKOVÁ AGENTURA
AMATÉRSKÁ ATLETASOCIACE			
PILOT			
TAJENKA			

(b) Prázdná křížovka

Obrázek 2.1: Křížovka s mřížkou 4×4.

2.2 Pravidla struktury křížovky

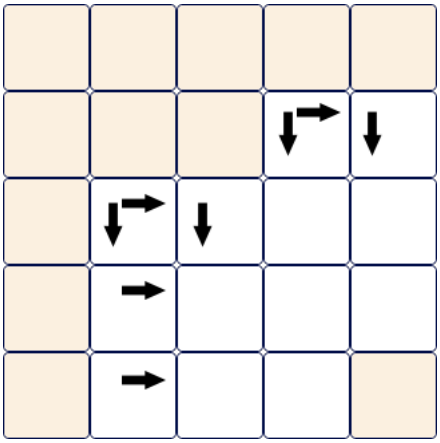
Při návrhu křížovky je nutné dodržovat předem stanovená pravidla. V případě jejich nedodržení by se komplikoval celý návrh a generování řešení.

Rozměry křížovky byly krátce zmíněny v předchozím bodě, ale je nutné doplnit vysvětlení. Minimální rozměr mřížky křížovky je 3×3 a to proto, že nejkratší slova používaná v křížovkách generovaných z českého slovníku jsou délky 2. Maximální rozměr není teoreticky omezen a tvar křížovky není pevně stanoven.

Rozmístění legend v křížovce musí být provedeno tak, aby nikde v mřížce nevznikaly příliš velké nerozdělené bloky buněk, a také aby se nikde v křížovce nevyskytovala samostatná legenda. Nesmíme zapomenout na to, že nejkratší řetězec je délky 2 a proto nemůžeme umístit legendu v mřížce tak, aby vznikl řetězec délky 1. Nedodržení těchto pravidel by mohlo způsobit problémy při generování řešení. Optimální délky řetězců, které vytvoříme rozmístěním legend, by měly být 5 - 6 a to z důvodu, že těchto slov ve slovníku bývá nejvíce.

2.3 Definice řetězců křížovky

U navržené křížovky připravené k vygenerování řešení se neobejdeme bez její kompletní definice řetězců. Pro budoucí generátor je vhodné mít informace o všech řetězcích, které se do křížovky budou vyhledávat. Jedná se o počáteční pozici každého řetězce, který v křížovce bude umístěn, jakým směrem bude orientován (vodorovně nebo svisle) a jeho délka. Pro názorný příklad definic řetězců jednoduché křížovky použijeme následující tabulku 2.1 a obrázek návrhu 2.2. Záznamy v tabulce budeme řadit dle jejich délky vzestupně a pro směr zvolíme 0 – vodorovně, 1 – svisle. V křížovce se budeme zabývat pouze bílými buňkami, kde se řídíme podle pomocných šipek.

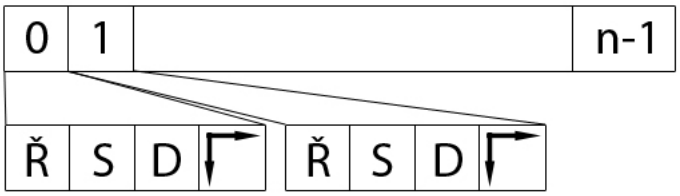


Obrázek 2.2: Návrh křížovky.

Pořadí	Řádek	Sloupec	Délka	Směr
1.	1	3	2	0
2.	4	1	3	0
3.	1	4	3	1
4.	2	1	3	1
5.	2	2	3	1
6.	2	1	4	0
7.	3	1	4	0
8.	1	3	4	1

Tabulka 2.1: Definice řetězců křížovky.

Na principu získání definic z návrhu křížovky, který byl předveden výše, bude navržen algoritmus, který analyzuje navrženou křížovku a vytvoří tak dynamické pole, jehož každý prvek bude obsahovat číslo řádku, číslo sloupce, délku řetězce a směr, kterým je řetězec orientován. Pole bude seřazené dle délky řetězce vzestupně. Princip dynamického pole je předveden na obrázku 2.3.



Obrázek 2.3: Definice křížovky ve formě dynamického pole.

2.4 Body křížení

Body křížení odpovídají buňkám pro písmena a tajenku, jejichž význam byl vysvětlen v 2.1. Bod křížení musí nést informaci o tom, na jaké pozici v řetězci se nachází a která definice řetězce jej zahrnuje. Oba typy informace musíme vždy uvažovat ve směru horizontálním a vertikálním. Pro názorný příklad seznamu bodů křížení využijeme tabulku 2.1 a první

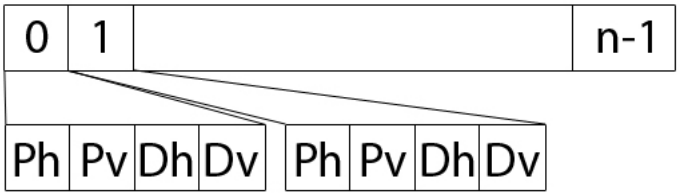
dva řádky křížovky na obrázku 2.2, ve kterém jsou pro nás důležité bílé buňky. „Pozice h.“ a „Definice h.“ odpovídají podle písmene „h“ horizontálnímu směru.

Tyto body společně se seznamem definic poskytují důležité informace, které přesně charakterizují danou křížovku.

Pořadí	**Pozice h.**	**Pozice v.**	**Definice h.**	**Definice v.**
1.	0	0	1	8
2.	1	0	1	3
3.	0	0	6	4
4.	1	0	6	5
5.	2	1	6	8
6.	3	1	6	3

Tabulka 2.2: Seznam bodů křížení.

Na principu získání bodů křížení z návrhu křížovky, který byl předveden výše, bude navržen algoritmus, který analyzuje navrženou křížovku a vytvoří tak dynamické pole, jehož každý prvek bude obsahovat pozice bodu v řetězcích a odpovídající indexy do pole definic řetězců křížovky. Je nutné podotknout skutečnost, že v poli budů křížení budeme potřebovat možnost pohybu po jednotlivých prvcích jako v matici. Proto při vytváření seznamu bodů křížení zahrneme i buňky pro legendy, které budou mít nulové ohodnocení. Princip dynamického pole je předveden na obrázku 2.4.



Obrázek 2.4: Seznam bodů křížení ve formě dynamického pole.

2.5 Seznam řetězců

Pokud máme k dispozici vyřešenou křížovku a vytvoříme všechny definice řetězců dle pravidel v 2.3, můžeme sestavit seznam řetězců postupným procházením seznamu definic. Je důležité, aby pořadí řetězců v seznamu odpovídalo přesnému pořadí jejich definic. Pokud máme k dispozici seznamy definice řetězců, body křížení a řetězce, jsme schopni zpětně sestavit celou křížovku, což je náš budoucí cíl jen s tím rozdílem, že seznam řetezců bude generován.

Kapitola 3

Požadavky na aplikaci

V dnešní době existují aplikace jak pro generování, tak i pro řešení křížovek. Na základě provedených studií, které aplikace jsou dostupné na internetu, bylo zjištěno, že pokud chceme software pro návrh a generování křížovek, jsou pro nás k dispozici desktopové placené aplikace. Jedná se o aplikace, které svou robustností jsou schopny generovat kvalitní křížovky, se kterými se setkáváme v tisku. Křížovky sestavené člověkem jsou nákladné a to z důvodu časové náročnosti, proto je pochopitelné, že organizace sestavující například křížovkářký magazín investují do koupi tohoto softwaru, který jim poskytuje mnohem větší flexibilitu. V podstatě veškeré křížovky, ať už se jedná o klasické švédské křížovky, osmisměrky, sudoku, jsou generované pomocí počítačové techniky.

Vývojáři těchto aplikací se musí soustředit na tři základní požadavky, které aplikace musí splňovat:

- Databáze výrazů, ze které bude křížovka sestavena, zahrnující velkou škálu výrazů (alesoň 500 000 výrazů).
- Algoritmus, který spolehlivě a rychle vygeneruje křížovku. Nejčastějším algoritmem pro generování křížovky je backtracking (4.1) s využitím heuristiky.
- Intuivní uživatelské rozhraní, které usnadní uživatelům práci v aplikaci.

Ve studii jsme se také zabývali aplikacemi, které jsou čistě webového charakteru a umožňují návštěvníkům řešit křížovky. Výsledek není příliš zajimavý. Při hledání takovýchto aplikací byl nalezen pouze server www.lusk.cz, který poskytuje jak placenou tvorbu křížovek, tak i bezplatné řešení z jejich webových stránek. Řešení švédských křížovek není na první pohled nějakým způsobem přitažlivé. Otázky k jednotlivým slovům jsou uvedeny v seznamu vedle křížovky, což neplní takový dojem, jako když máme před sebou položené noviny s křížovkou.

3.1 Analýza současných generátorů

V dnešní době je dostupná celá řada desktopových programů pro generování křížovek. Z českých zástupců lze jmenovat např. CwdStudio, Křižák. CwdStudio kromě švédkých křížovek umožňuje vytváření klasických křížovek a osmisměrek – přičemž si uživatel může zvolit obtížnost generování (aplikace disponuje slovníkem o velikosti 40 000 výrazů). Program křížák využívá nakladatelství SAGAX, na jejichž webových stránkách uvádějí základní informace o generátoru, který vychází z algoritmů umělé inteligence – program využívá sofistikované algortimy na bázi slepého prohledávání stavového prostoru v modifikaci backtracking a využívá heuristiky.

Ze zahraničních aplikací lze uvést FineCrosses. Většina aplikací, které lze běžně na internetu vyhledat, jsou placené a poskytují pouze demoverze pro vyzkoušení s omezenou funkčností.

3.2 Shrnutí

Po provedení studie bylo možné specifikovat jaké nástroje a technologie pro vývoj naší aplikace využijeme a také jaké požadavky na aplikaci stanovíme. Nemohli jsme se vydat cestou vývoje aplikace pomocí HTML a JavaScriptu, jak to bylo řešeno na www.lusk.cz, protože naším hlavním cílem bylo uspokojit uživatele při řešení křížovky komfortním uživatelským rozhraním. Pro nás se nabízely dvě technoligie Adobe Flash a Silverlight. Pro aplikaci jsme zvolili technologii Silverlight, která bude popsána v kapitole 6. Na základě této technologie byly zvoleny vývojové nástroje, které jsou popsány v kapitole 5.

Z desktopových aplikací pro návrh křížovek jsme se ponaučili a inspirovali správným přístupem k tvorbě uživatelského rozhraní, např. jaké možnosti může mít uživatel při tvorbě křížovky a co přitom nesmí porušit. Jelikož se jedná o webovou aplikaci, bylo rozhodnuto, že maximální velikost generované křížovky bude 15×15 políček. Důvodem je, aby se křížovka zobrazila celá a uživatel, který nemá monitor s vysokým rozlišením, nemusel při řešení využívat posuvníků.

3.3 Stanovené požadavky na naši aplikaci

- webová aplikace
- možnost řešit a navrhovat křížovku
- generování křížovky pomocí algoritmu backtracking
- vytvoření křížovky dle předlohy z např. novinové křížovky
- velikost vytvářené křížovky $3 \times 3 - 15 \times 15$ políček
- uložení vytvořené křížovky do databáze
- uložení vytvořené křížovky do formátu png, pdf
- přímý tist křížovky z prohlížeče
- zobrazení správného řešení křížovky při luštění
- zobrazení chyb při luštění křížovky

Kapitola 4

Návrh řešení generátoru

V této kapitole bude vysvětlen návrh generátoru pracujícího metodou obalování slov [12], který využívá algoritmu zpětného navracení backtracking viz 4.1. Před zahájením generování, je třeba mít shromážděné veškeré potřebné informace charakterizující proporce křížovky viz 2.3, 2.4 a připravená vstupní data tak, aby umožnila generátoru vyhledat řešení v co nejkratším čase. Způsob uspořádání dat bude popsán v 4.3 a princip vyhledávání slov ve slovníku bude vysvětlen v 4.4. V závěru této kapitoly bude nastíněno zjednudušené řešení generátoru bez zpětného navracení popsáného pomocí pseudokódu 2.

4.1 Backtracking

Algoritmus backtracking – slepé prohledávání se zpětným navracením, je algoritmus prohledávání do hloubky, patřící do skupiny neinformovaných metod viz [13]. Cílem backtrackingu je nalézt řešení ve stavovém prostoru, přičemž velké množství potenciálních řešení může být vyloučeno bez přímého vyzkoušení. Před zahájením hledání řešení je určen počáteční stav, ze kterého jsou generovány postupně nové stavy, dokud není nalezen stav koncový, kdy algoritmus úspěšně končí. Pokud se algoritmus ocitne v situaci, kdy není možné ze současného stavu vygenerovat stav nový, vrací se vždy na předchozí stav a pokračuje v generování. Pokud nastane situace, že není možné vygenerovat z počátečního stavu stav nový, znamená to, že pro konkrétní počáteční stav neexistuje řešení ve stavovém prostoru. V našem případě se jedná o případ, kdy uživatel zadá nesmyslnou tajenku, ze které algoritmus není schopen vygenerovat křížovku. Časová složitost algoritmu je exponenciální. Pomocí backtrackingu je např. řešen problém osmi dam.

Algoritmus:

1. Inicializuj zásobník Open a vlož do něj počáteční stav.
2. Pokud je zásobník Open prázdný, ukonči prohledávání jako neúspěšné.
3. Pokud lze na stav na vršku zásobníku aplikovat první/další operátor, aplikuj ho. Jinak vyjmi stav ze zásobníku a vrať se na bod 2.
4. Pokud je nově vzniklý stav stavem cílovým, vypiš cestu k řešení a ukonči prohledávání jako úspěšné. Jinak ulož stav do zásobníku Open a vrať se na bod 2.

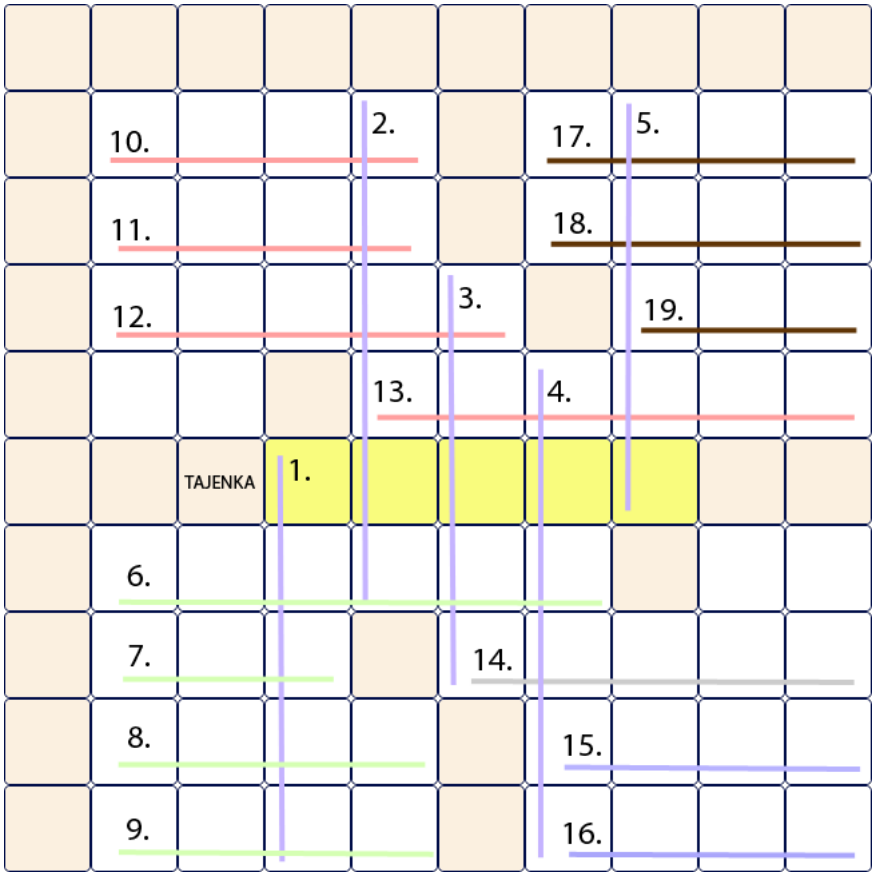
Algoritmus 1: Backtracking - pseudokód

```
InitStack(Open);
Push(Open, firstState);
while Open is not empty do
    Top(Open, expandedState);
    newState = ApplyNextOp(expandedState);
    if newState == NULL then
        Pop(Open);
    else if Goal(newState) then
        PrintResult(newState);
        return true;
    else
        Push(Open,newState);
end
return false;
```

`InitStack` – Inicializuje zásobník Open.
`Top/Pop/Push` – Ukáže/vyjme/uloží stav z/do zásobníku Open.
`ApplyNextOperator` – Aplikuje první/další operátor a vrací ukazatel na nový stav.
`Goal` – Vrací `true` v případě, že jsme narazili na cílový uzel.
`PrintResult` – Vytiskne cestu z cílového do počátečního uzlu.

4.2 Metoda obalování slov

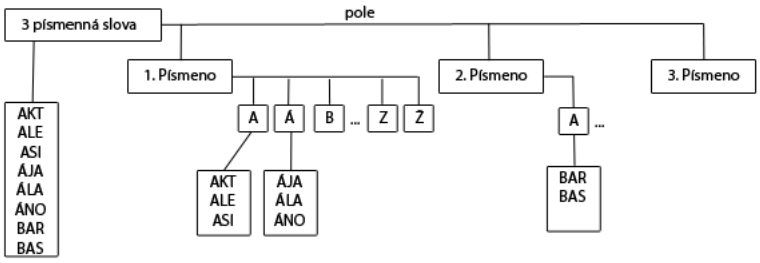
Předpokládejme křížovku o velikosti 10×10 buňek s rozmístěnými legendami dle obrázku 4.1. Zvolíme startovní slovo – tajenku, která se obvykle volí jako nejdelší slovo v křížovce. Generování řešení křížovky začíná od prvního písmene tajenky, kdy je vytvořena kolmá maska, dle které je následně připraven seznam odpovídajících řetězců. Jednotlivé řetězce vytvořeného seznamu jsou postupně dosazovány do křížovky a u každého písmene vloženého řetězce je opět zkontrolována kolmá maska, aby nedocházelo k vytváření slov, které se nevyskytují ve slovníku. Nalezený vhodný řetězec je vložen do křížovky a bude dalším, který bude tzv. obalován po vyřešení písmen tajenky. Aby generování pokračovalo po obalení tajenky tímto řetězcem, uložíme si všechny jeho křížící se pozice (body křížení) do struktury typu fronta. Při každém vkládání nových pozic do fronty musí být provedena kontrola, zdali se nová pozice již ve frontě nenachází. Takto budeme postupovat, dokud nebude vyplněna celá křížovka. Zkrácený postup generování principem obalování slov je uveden na obrázku 4.1.



Obrázek 4.1: Princip metody obalování slov.

4.3 Uspořádání dat ve slovníku

Při generování křížovky potřebujeme co nejrychleji vyhledávat potřebný seznam řetězců, který odpoývídá zadané masce. Prohledávání vždy celého slovníku by bylo příliš časově náročné a neefektviní, proto si celý slovník naindexujeme dle délek slov a písmen na jednotlivých pozicích ve slovech. Můžeme tak přistoupit např. k seznamu slov délky „3“ s druhým písmenem „B“ nebo se třetím písmenem „R“ ve slově. Názorný příklad indexování slovníku je uveden na obrázku 4.2.



Obrázek 4.2: Princip naindexování slovníku.

4.4 Vyhledávání ve slovníku

Díky naindexovanému slovníku je vyhledávání řetězců odpovídající hlavní masce mnohem jednodušší. Dle masky, která určuje jaká písmena se nachází na daných pozicích v hledaném slově, postupně vybereme všechny seznamy slov ze slovníku a porovnáme s maskou. Výrazy shodující se s maskou vkládáme do nového seznamu, který pro nás bude klíčovým seznamem, ze kterého budeme vybírat slovo do křížovky. Pokud slovo splňuje podmínku, že jeho všechny ortogonální masky mají své zástupce ve slovníku, je vloženo do křížovky. Kontrolu ortogonálních masek provádíme opět dle pricipu vyhledávání řetězců dle hlavní masky.

4.5 Pseudo generátor

Algoritmus 2: Generátor - pseudokód bez zpětného navracení

```
InitQueue(CrossPointQueue);
while CrossPointQueue is not empty do
    Get(CrossPointQueue, CrossPoint);
    MainMask = CreateMainMask(CrossPoint);
    WordList = GetListBy(Dictionary, MainMask);
    while All ortogonal masks are not ok do
        InsertWord(WordList.Item);
    end
    Insert(passedCrossPoints, CrossPoint);
    InsertNewCrossPoints(CrossPointQueue, CrossPoint);
end
```

`InitQueue` – Inicializuje frontu bodů křížení a vloží do ní počáteční bod.
`Insert` – Vloží bod křížení do seznamu vyřešených bodů.
`Get` – Vyjme bod křížení z fronty.
`InsertNewCrossPoints` – Vloží nové body křížení do fronty.
`GetList` – Vyhledá seznam řetězců odpovídajících masce ve slovníku.
`InsertWord` – Vloží nové slovo do křížovky.

Kapitola 5

Vývojové nástroje a technologie

Hlavním požadavkem na aplikaci bylo, aby byla webového charakteru. Proto se pro vývoj aplikace nabízely dvě sobě konkurenční technologie Flash od firmy Adobe a Silverlight od firmy Microsoft, které umožňují uživateli interaktivní práci s aplikací. Pro vývoj byla zvolena technologie Silverlight z důvodu volného přístupu k placeným vývojovým nástrojům a také proto, abych se naučil pracovat s touto pro mě doposud neznámou technologií.

V této kapitole budou stručně rozebrány technologie a vývojové nástroje firmy Microsoft viz Visual Studio 5.2, Microsoft Expression Blend 5.3 a Microsoft SQL Server 5.4. Technologii Silverlight bude věnována samostatná kapitola 6. Tato kapitola čerpá z [4], [7], [8], [10], [11].

5.1 .NET Framework

.NET Framework je rozsáhlá softwarová platforma, jejíž první verze se objevila v roce 2002. Platforma je převážně určena pro vývoj různých druhů aplikací. Pomocí této platformy můžeme vytvářet klasické desktopové aplikace pro Windows, webové aplikace, nebo aplikace pro mobilní zařízení.

Architektura

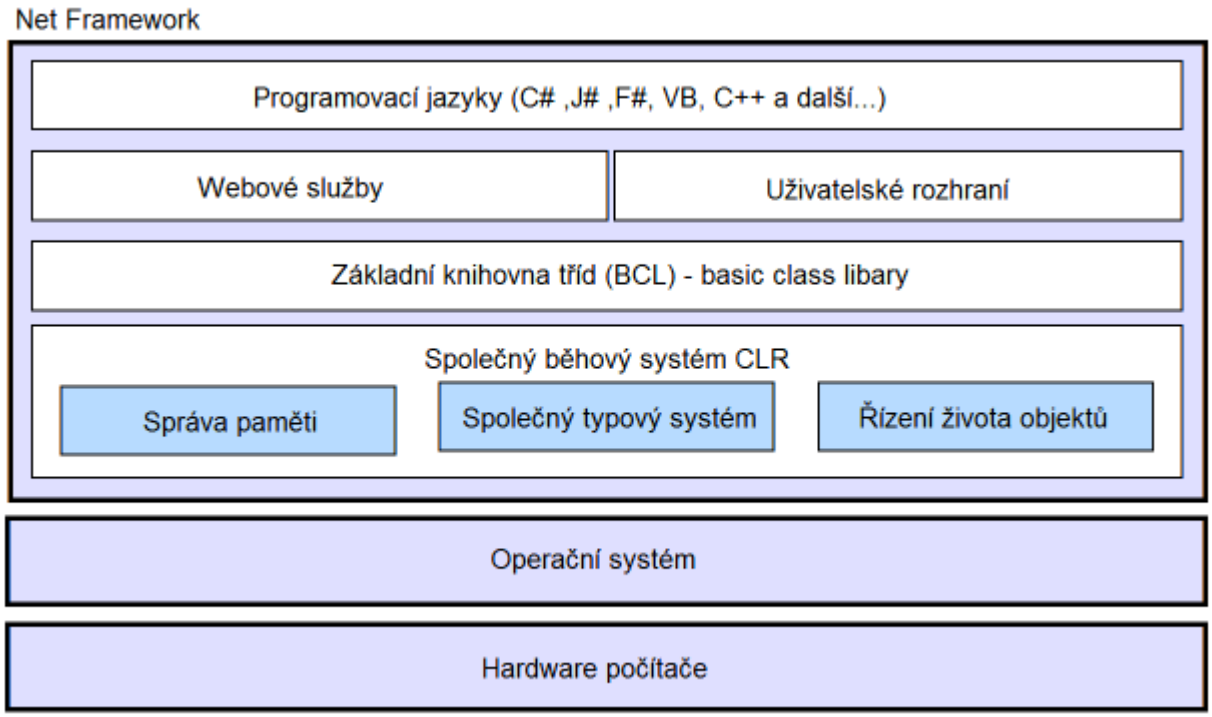
Jádro platformy .NET nazvané .NET framework, je nadstavbou nad operačním systémem. V současné době existuje ve verzích 1.0, 1.1, 2.0, 3.0, 3.5, 4,0. Nejnovější verze 4.0 byla představena v roce 2010. Architekturu .NET frameworku tvoří tři základní vrstvy. První vrstvou na nejnižší úrovni je *Common Language Runtime* (CLR), která realizuje základní infrastrukturu, na které je celý .NET Framework postaven. Nad CLR se nachází vrstva *Basic Class Library* BCL – knihovna tříd systému. Na nejvyšší úrovni jsou dvě vrstvy, které usnadňují vývoj nejen webových aplikací, ale i klasické desktopové aplikace s uživatelským rozhraním. Schéma architektury frameworku je ukázáno na obrázku 5.1.

Podrobnější popis dvou hlavních vrstev:

- **CLR** – běhový systém, který si můžeme představit jako virtuální stroj, ve kterém pracují aplikační funkce platformy .NET, kde mají všechny jazyky k dispozici knihovnu tříd systému. Tzn. na počítači beží další systém, který spravuje aplikace pro .NET. Pro tento systém existuje obdoba zdrojového kódu *Microsoft Intermediate Language* MSIL. Aplikace napsané pro .NET jsou omezeny výkonností prostředí, ve kterém běží a to z důvodu, že kód není kompilován přímo do zdrojového kódu, ale do MSIL. Tímto

nedostatkem se ovšem objevují mnohé výhody, např. řízený kód, jednotný typový systém a správa paměti.

- **BCL** – standardní knihovna k dispozici všem .NET jazykům. BCL zapouzdřuje velké množství funkcí jako je čtení a zápis do souboru, zobrazování grafiky, interakce s databází a manipulace s XML dokumenty, které usnadňují práci vývojářům. Mezi základní třídy patří ADO.NET, XML.



Obrázek 5.1: Schéma architektury .NET Frameworku převzato z [3].

Vývoj platformy

V této sekci bude popsán vývoj platformy od verze 2.0 na základě vzniku nových technologií, které byly postupně začleněny.

Verze 2.0

- ASP.NET – technolgie pro vývoj webových aplikací.

Verze 3.0 a 3.5 byly rozšířeny o tyto knihovny:

- Windows Communication Foundation (WCF) – technologie využívána pro vzájemnou komunikaci mezi komunikačními protokoly a technologiemi.
- Windows Presentation Foundation (WPF) – slouží pro vytváření hardwarově akcelerovaného uživatelského rozhraní.
- Windows Sync Framework – sada objektů pro synchronizaci různých zdrojů.

- Windows Workflow Foundation (WF) – technologie pro definování heterogeních sekvenčních procesů.
- Windows CardSpace – obsahuje implementaci standardu Information Cards.
- LINQ (Language INtegrated Query) – slouží k objektovému přístupu k datům v databázi.

Verze 4.0

- Managed Extensibility Framework – usnadňuje vytváření rozšiřitelných aplikací a aplikací, které jsou schopny složení.
- PLINQ – Paralelní implementace LINQ to Objects.

ASP.NET

Technologie ASP.NET je jednotný model vývoje webu, který zahrnuje služby nezbytné k sestavení rozsáhlých webových aplikací s minimálním množstvím kódu. Při vytváření aplikací má vývojář přistup k třídám rozhraní .NET Framework a může navíc psát aplikace v libovolném jazyce, který je kompatibilní s modulem CRL. Mezi tyto jazyky patří např. Visual Basic a C#. Firma Microsoft nabízí pro vývoj aplikací pomocí této technologie zdarma vývojové prostředí Visual Web Developer Express.

Pomocí sady Visual Studia 2010 můžeme vytvářet různé typy projektů této technologie. Jedná se o projekty zahrnující weby, webové aplikace, webové služby a AJAX serverové ovládací prvky. Pokud chceme vytvořit MVC (model-view-controller) aplikaci, musíme použít projekt webové aplikace. Hlavní rozdíly mezi projekty webů a projekty webových aplikací lze nalézt v [9].

Webové stránky technologie ASP.NET mohou být požadovány z jakéhokoliv prohlížeče nebo mobilního zařízení. Jedná se o zcela objektově orientovaný přístup. V rámci webových stránek můžeme pracovat s elementy jazyka HTML pomocí vlastností, metod, událostí. Veškerý kód technologie je kompilován, což umožňuje silné typování, optimalizaci výkonu a statickou vazbu. Nechybí zde ani infrastuktura rozšířeného zabezpečení pro ověřování a autorizaci přístupu uživatelů, či provádění jiných operací v souvislosti se zabezpečením.

Windows Communication Foundation

WCF je zaměřen na vytváření servisně orientovaných aplikací. Díky WCF je možné zasílat data jako asynchronní zprávy z jednoho koncového bodu služby k jinému. Koncovým bodem může být např. jiná služba unvnitř aplikace nebo část běžně dostupné služby hostované IIS. Model WCF rozděluje aplikace na klienty a na služby, kde klient je aplikace, která iniciuje komunikaci, služba je aplikace, která čeká na požadavky klientů.

Služba

Služby (services) jsou základním stavebním prvkem. Jedná se o systém, který poskytuje jeden nebo více koncových bodů, které slouží pro příjem a odeslání SOAP (Simple Object Access Protocol) zpráv. Skládá se ze tří základních částí:

- třída služby (její implementace),
- hostovací prostředí (místo), na kterém služba poběží,

- jeden nebo více koncových bodů.

Koncový bod
Koncový bod je místo uvnitř služby, které slouží k přijímání a odesílání zpráv a skládá se ze tří částí:

- adresa – místo na které budou zasílány zprávy,
- binding – definice komunikace služby (komunikační protokol, kódování, druh zabezpečení),
- kontrakt – specifikace rozhraní, které služba poskytuje (např. metody).

Windows Presentation Foundation

Podsystém WPF je nový přístup pro vytváření Windows klientských aplikací. S WPF je možné vytvářet širokou škálu desktopových a webových aplikací. Jádro WPF je založeno na idependece-resolution[1] a vector-based[2] rendrovacím stroji, který využívá výhod moderních grafických modulů. Podsystém rozšiřuje základní komplexní sadu funkcí o jazyk XAML (Extensible Markup Langugage), ovládací prvky, datové vazby, rozložení, 2D a 3D grafiku, animace, styly, šablony, dokumenty, média, text a typografii.

XAML je značkovací jazyk, vystavěn na základě jazyka XML, umožňující deklarativní přístup pro implementaci vzhledu aplikací. Typicky je využíván pro vytváření oken, vyskakovacích oken, stránek, řízení prostředí a obecně grafiky. Uživatelské rozhraní tvoří stromovou hierarchii vnořených prvků.

Hlavní vlastností aplikace je implenetace funkcionality, která zodpovídá za interakci s uživatelem včetně zpracování událostí (např. kliknutí na nabídku, talčítko) a volání logiky přístupu k datům. Toto chování je obecně implementováno v .NET jazyce např. c# a typ kódu je označován code-behind[3].

Jazyk C#

C# je určen pro vytváření aplikací založených na platformě .NET Framework. První verze byla vydána společně s NET. Frameworkem 1.0 v roce 2002 a obsahovala pouze základní podporu objektového programování. V současné době je již na trhu verze C# 4.0.

Vývojáři, kteří programují v jazycích C, C++ nebo Javě jsou obvykle schopni začít efektivně pracovat v tomto jazyce za velmi krátkou dobu. Syntaxe jazyka C# zjednodušuje mnoho složitostí C++ a poskytuje výkonné prvky např. výčty, delegáty, lambda výrazy, přímý přístup do paměti, které nejsou k dispozici v Javě. Podporuje generické metody, typy, které zvyšují bezpečnost a výkon, a iterátory, které umožňují vývojářům kolekcí tříd definovat vlastní chování.

Zapouzdření, polymorfismus a dědičnost jsou bezpochyby samozřejmostí. Všechny proměnné a metody jsou zapouzdřeny v rámci definice třídy. Třídy mohou dědit pouze z jedné nadřazené třídy, ale mohou implementovat libovolný počet rozhraní. Metody, které přepisují virtuální metody nadřazené třídy vyžadují klíčové slovo „override“, aby nedošlo k náhodnému předefinování. C# usnadňuje vývoj softwarových komponent díky inovativních konstrukcí např. LINQ.

[1] Elementy grafického rozhraní mohou být vykreslovány ve velikosti nezávislé na mřížce pixelů.
[2] Typ grafického rozhraní jehož elementy jsou vykreslovány vektorově.
[3] Oddělení implementace chování od implementace vzhledu, kód běží na pozadí.

5.2 Microsoft Visual Studio 2010

Visual Studio 2010 Professional je integrované prostředí, které zjednodušuje tvorbu, ladění a nasazování aplikací. Psaní kódu je rychlejší, je možné zužitkovat své dosavadní dovednosti a prostředí je možné přizpůsobit podle svých zvyklostí. Z hlediska vývoje pro naši aplikaci je důležitý vývoj pomocí technologie Silverlight. Visual Studio obsahuje mnoho prvků, některé z nich jsou popsány níže.

- **Editor kódu** – Standardem editoru je zvýraznění syntaxe a automatické dokončování. Mezi navigační prvky editoru patří nastavování záložek, sbalování bloků kódu, hledání s podporou regulárních výrazů.
- **Debbuger** – Umožňuje spustit aplikaci v debbug módu, ve kterém je možné krokování programu a kontrola obsahu proměnných.
- **WPF Designer** – Designer umožňuje snazší návrh aplikace pomocí uživatelského rozhraní bez znalostí XAML jazyka, který je automaticky generován.
- **Designer dat** – Tento designer dat je určený pro grafickou úpravu databázových schémat, včetně psaných tabulek, primárních a cizích klíčů a omezení.
- **Designer mapování** – Designer mapování je používán funkcí LINQ to SQL k zobrazení mapování mezi databázovými schématy a třídami, které zabalují data.

5.3 Microsoft Expression Blend 4

Microsoft Expression Blend verze 4, je flexibilní a produktivní grafické vývojářské prostředí. Pomáhá při tvorbě moderních a graficky propracovaných aplikací. Umožňuje vytvářet a editovat soubory typu XAML, které popisují prezentační vrstvu webových aplikací. Vývojář se tedy nemusí učit celý XAML jazyk a může si pohodlně vytvořit vzhled aplikace v tomto nástroji. XAML jazyk vychází z jazyka XML.

Při návrhu prezentační vstvy je možné využít již předdefinovaných prvků např. rámec, tlačítko, blok pro text ..., kde každý prvek má své vlastní nastavení. Pokud ovšem nemůžeme prvek změnit do podoby dle našich představ, je zde druhá možnost, vytvořit styl, který pak můžeme aplikovat na více prvků stejného typu. Usnadní to tak vytváření např. graficky propracované nabídky menu. Tyto styly můžeme dále využít při vytváření dynamických prvků pomocí jazyka c#, kde stačí pouze styl přiřadit a vzhled prvku není třeba dále definovat. Nástroj umožňuje návrh stylu dvěma způsoby, kterými jsou vytvoření pomocí uživatelského prostředí nebo čistý zápis v jazyce XAML. Další propracovanou části nástroje je přiřazování akcí na různé typy událostí. Jednoduše a rychle může uživatel přiřadit akci (např. tlačítku) při přejetí myši a nastavit tak změnu pozadí či různou animaci dle vlastní fantazie.

5.4 Microsoft SQL Server

Microsoft SQL Server 2008, je produkt, který byl vyvinut v souladu s vizí společnosti Microsoft pro datovou platformu. Organizace pak můžou kdykoli spravovat libovolná data z jakéhokoliv místa. Data je možné ukládat ze strukturovaných, částečně strukturovaných a nestrukturovaných dokumentů, jako jsou obrázky, multimediální soubory, přímo v rámci

databáze. Tento produkt nabízí širokou škálu integrovaných služeb, které umožňují provádět další operace s daty, jako například vyhledávání, dotazy, synchronizaci, generování sestav a vytváření analýz.

SQL Server poskytuje také vysokou úrověň zabezbečení, spolehlivosti a škálovatelnosti. Umožňuje využívat data z vlastních aplikací vyvinutých na platformách Microsoft .NET a Visual Studio.

Kapitola 6

Silverlight

V posledních několika letech jsme zaznamenali odklon od webových aplikací využívajících JavaScript z důvodu maximální kompatibility pro prohlížeče. Dnes se však už u většiny nových aplikací předpokládá začlenění technologie AJAX založené na JavaScriptu a vytváření tak tzv. RIA (Rich Internet application) aplikací. Každý den tak vidíme spoustu webových „WEB 2.0“ aplikací. Opuštění od tvorby tzv. „chudých aplikací“, které postrádají jakoukoliv interaktivitu s uživatelem, vedlo k nevyhnutelnému přechodu k tzv. „bohatým aplikacím“. Uživatelské rozhraní webových aplikací se v dnešní době dostává na tak vysokou úroveň, že svou propracovaností převyšuje některé desktopové aplikace.

Příchod dlouho očekávaného podsystému WPF (Windows Presentation Foundation) umožnil jednotný přístup k uživatelskému rozhraní Microsoft desktopových aplikací. Tento podsystém vychází z technologií, které jsou již na webu dlouho využívany (např. deklarace uživatelskéo rozhraní pomocí značkování), sjednocuje programování uživatelského prostředí s jeho grafickým návrhem a představuje nové možnosti pro okenní aplikace.

Na základě WPF a RIA aplikací vzikla technologie Silverlight, kterou dnes známe v její nejnovější verzi 4. Zjednodušeně je možné Silverlight aplikaci přirovnat k interaktivnímu zobrazovacímu a pracovnímu prostoru, který je zobrazen na straně klienta (webový prohlížeč nebo samostané okno), přičemž aplikační logika pracuje na straně serveru. Tato kapitola čerpá z [1], [5] a [6].

6.1 Historie

V září roku 2007 se objevila první verze technologie Silverlight s podporou jediného programovacího jazyka JavaScript. JavaScript je využíván k interkakci mezi Silverlight objekty, které jsou vykonávány pomocí pluginu ve webovém prohlížeči. Později v roce 2008 následovala verze 2, která s sebou přinesla možnost vytváření již bohatých aplikací a bylo možné využívat v plné míře .NET jazyky. V polovině roku 2009 vyšla další verze Silverlight 3, která přináší některé významené novinky a vylepšení. Jednou z nejvýznaměnjších novinek je možnost běhu aplikace na klientském počítači mimo webový prohlížeč. Nalezneme zde i také rozšířenou datovou podporu, podporu pro bussines objekty a možnost přehřávání audia a videa ve vysoké kvalitě. V nynější době již vyšla doposud zatím poslední verze Silverlight 4, která bude popsána v 6.2.

6.2 Silverlight

Tato platforma je určena pro vytváření a provoz internetových RIA aplikací. Aby bylo možné spustit aplikaci ve webovém prohlížeči, je nutné mít nainstalovaný speciální plug-in, který je velmi podobný plug-inu Adobe Flash a instaluje se stejným způsobem. Silverlight plug-in podporuje vektorovou grafiku, audio, video a HD video, které jsou zabezpečeny systémem *Digital Rights Management* (DRM). Charakteristika technologie je popsána v následujících bodech:

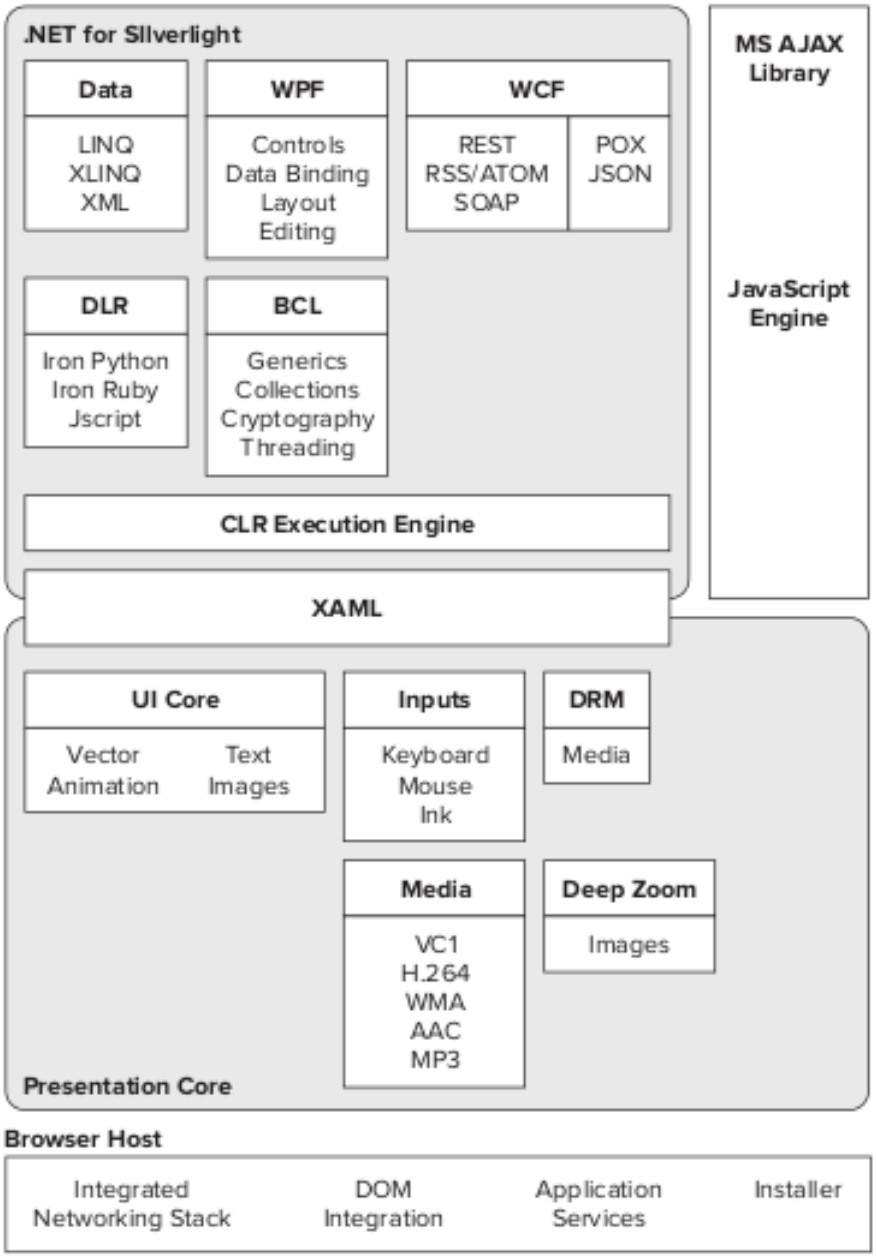
- Silverlight je multiplatformí technologie podporovaná ve všech webových prohlížečích pro tvoření bohatých, interaktivních aplikací.
- Silverligh 4 aplikace mohou být vytvořeny pomocí nástrojů Expression Blend, Microsoft Visual Studio a Eclipse.
- Silverlight podporuje přehrávání Windows Media VC-1/WMA a 720p+ full-screen HD Video.
- Užitím XAML, HTML, JavaScriptu, C# nebo VB Silverlight přináší bohatá multimédia, vektorovou grafiku, animace a interaktivitu.
- Knihovny BCL poskytují přístup k obecným třídám kolekcí, generics a vláken, které jsou obvyklé pro vývoj Windows klient aplikací.
- Jakákoliv Silverlight aplikace může běžet jako klasická desktopová aplikace tzv. out-of-browser.
- Začlenění nových business prvků – navigační framework, tiks, podpora drag-and-drop, událost kliknutí na pravé tlačítko myši atd.
- Šablony obchodních aplikací.
- Veškerá aplikační logika a XAML kód vytvořený v Silverlightu může být použit u WPF alikací.

6.3 Silverlight Toolkit

Pro rozšíření ovládacích prvků a nových utilit lze využít balík Silverlight Toolkit, který obsahuje další prvky (např. grafy), které mohou být vývojáři užitečné. Toolkit není součástí Silverlight runtime a to z důvodu nabytí velikosti instalačního balíku Silverlight. Oddělení toolkitu od základní verze umožňuje Microsoftu aktualizace nástrojů mimo vydání cyklu Silverlight.

6.4 Schéma architektury

Kompletní architekturu technologie Silverlight můžeme vidět na obrázku 6.1.



Obrázek 6.1: Schéma architektury Silverlight převzato z [1].

Kapitola 7

Implementace aplikace

Aplikace byla vyvíjena nejprve pomocí technologie Silverlight 3, protože pro tehdejší verzi 4 nebyl dostupný nástroj Expression Blend pro implementaci vzhledu. Když se vývoj aplikace blížil ke konci a byly doimplementovány poslední části, objevily se nedostatky této verze. Verze 3 neumožňovala přímý tisk z prohlížeče, což dalo impuls k přechodu na verzi 4 již s dostupným Expression Blendem.

V této kapitole bude v úvodu předveden vývojový diagram tříd 7.1, které spolu se svými metodami tvoří jádro funkcionality aplikace. Implementace uživatelského rozhraní, implementovaného v jazyce XAML bude popsáno v 7.1 . Implementace funkcionality aplikace bude popsána v 7.3.

7.1 Uživatelské rozhraní

Vzhled uživatelského rozhraní byl kompletně navržen v nástroji Expression Blend, které automaticky generuje XAML kód. Nejprve byl navržen startovní vzhled, aby bylo možné pracovat i průběžně na funkcionalitě aplikace. Při postupném seznámení se s nástrojem a nastudováním tutoriálů, byl navržen originální moderní vzhled uživatelského rozhraní viz B.1. Každá stránka aplikace nebo vyskakovací okno je třídou, která má svoji vlastní definici vzhledu ve formě XAML kódu.

Šablona stránek

Vzhled aplikace je možné rozdělit na dynamické a statické části. Statické části jsou ty, které se po celou dobu běhu aplikace nemění. V našem případě se jedná o záhlaví aplikace s hlavním menu a zápatí aplikace. Tělo aplikace je dynamickou částí, která se pokaždé mění (přechod na jinou stránku). Vzhled třídy `MainWindow` byl implementovan tak, aby obsahoval staticke záhlaví, zápatí a do těla byl umístěn objekt `ContentFrame`, který na základě URL adresy dokáže zobrazovat obsah jiných stránek.

Efekty

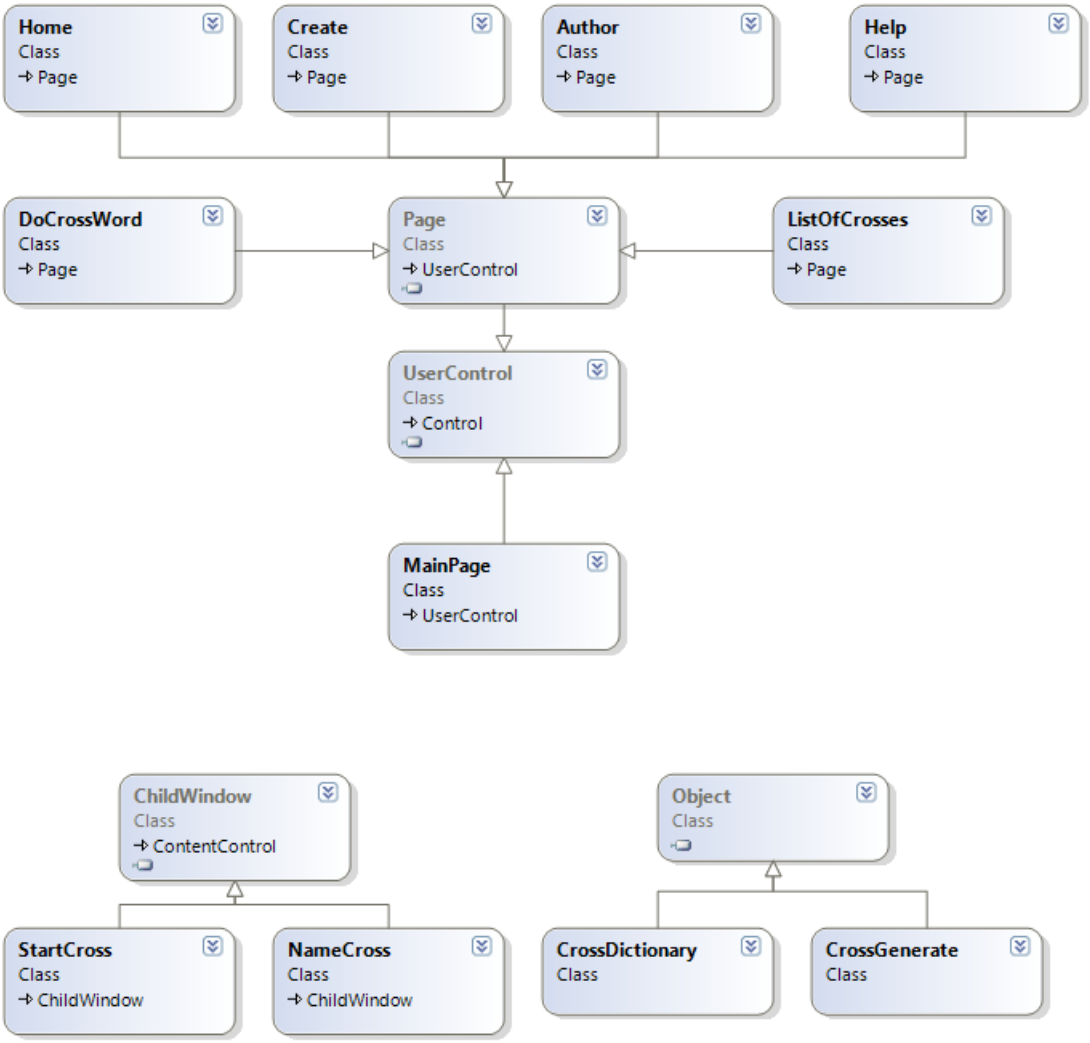
Pokud si spustíme aplikaci, můžeme si všimnout tlačítek s odlesky, či změny v barvách při přejetí myši přes tlačítko. Expression Blend je silný nástroj, ve kterém se tyto efekty rychle a snadno navrhují. Při výrobě tlačítek s odleskem byly použity standardní prvky nástroje, kde bylo nutné správně nastavit jejich vlastnosti (barevné přechody, rámečky, podbarvení, zaoblení rámce atd.). Při přiřazení efektu změny podbarvení při přejetí kursorem myši bylo

využito speciálního editoru nástroje, ve kterém jsme tlačítku přiřadili typ události a efekt, který se má provést. Při vytváření aplikace, můžeme využít již předdefinovaných prvků, ovšem pokud chceme vytvářet vlastní grafiku, musíme jít od základu stavby.

Styly prvků

Při vytváření tlačítek a nabídky menu byly vytvořeny vlastní grafické styly, které byly použity na více prvků. Došlo tak k zefektivnění a usnadnění práce. Styly prvků jsou také používaný při vytváření dynamických prvků, kterým chceme dodat moderní grafickou podobu (vytvoření pole křížovky). Definice stylů jsou součástí XAML kódu.

7.2 Vývojový diagram tříd



Obrázek 7.1: Vývojový diagram tříd

7.3 Implementace funkcionality

Implementace tříd byla uskutečněna v nástroji Microsoft Visual Studio. Každé stránce s navrženým vzhledem byla doimplementována její funkcionalita.

Pohyb v aplikaci

Přechod mezi jednotlivými stránkami a dynamickou změnu těla řídí třída `MainWindow`. V konstruktoru třídy je nastavena objektu `ContentFrame` adresa k úvodní stránce, která se zobrazuje při startu aplikace a dále jsou nastaveny posuvníky tak, aby reagovaly na otočení kolečkem myši. Metody reagující na událost „kliknutí na tlačítko v hlavním menu“ zařídí změnu adresy objektu `ContentFrame` na adresu požadované stránky, kterou chceme zobrazit.

Návrh křížovky

Při kliknutí na tlačítko „Vytvořit křížovku“ v hlavním menu, je vytvořena instance třídy `StartCross`, která způsobuje vyvolání dialogového okna pro zadání rozměrů křížovky. Metody třídy kontrolují validitu vstupu. V případě, že jsou hodnoty zadány správně, uloží je do veřejných proměnných (výška, šířka) a vrátí řízení metodě třídy `MainWindow`, která okno vyvolala. Tato metoda přečte hodnoty veřejných proměnných z instance okna a vytvoří URL adresu obsahující rozměry, kterou nastaví objektu `ContentFrame` a dojde tak k přesměrování na stránku s návrhem křížovky.

Při načtení stránky návrhu křížovky, kterou řídí třída `Create`, je jako první spuštěna metoda detekující parametry URL adresy a protože se tam vyskytují rozměry křížovky, zpracuje je a uloží do privátních proměnných. Poté je spuštěna metoda pro sestavení prázdného pole křížovky. Metoda vytvoří objekt `Grid` (mříž) o rozměrech zadaných uživatelem, který vyplní objekty `TextBox`. Každému `TextBoxu` je přiřazen styl definovaný v souboru s XAML kódem. Zbylé metody třídy `Create` řídí interaktivitu s uživatelem při návrhu křížovky. Při kliknutí na tlačítko spustit generování je vytvořena pracovní matice, která obsahuje tvar křížovky. Dále je vytvořena instance `generate` třídy `CrossGenerate`, ve které nainicializujeme rozměry křížovky, pracovní matici, výstupní objekty (mříž, bussy box). Nakonec vytvoříme nové vlákno, kterému přiřadíme startovní metodu `prepareGenerating` instance `generate` a spustíme vlákno. Nyní již je proces generování řízen metodami objektu `generate` a bude podrobně popsán v 7.3.

Generování křížovky

Jak už bylo řečeno v 7.3, generování beží v samostatném vlákně, které jako první spustí metodu `prepareGenerating`, která provede postupně tyto části:

1. Vytvoření definic řetězců křížovky 2.3.
2. Vytvoření pole délek řetězců, které se v křížovce vyskytují.
3. Vytvoření pole bodů křížení 2.4.
4. Inicializace seznamu řetězců (pouze řetězec tajenky).
5. Vytvoření instance třídy `CrossDictionary` a naindexovaní slovníku.

6. Nastavení odchycení události „slovník naindexován“ a následné spuštění generování křížovky – metoda `generateCrossWordHandler`.

Metoda `generateCrossWordHandler` je řídící metoda procesu generování, která inicializuje zásobník stavů (ukládání stavů křížovky) a frontu bodů křížení. Do fronty jsou na počátku vloženy ty body křížení, které náleží tajence. Poté je cyklicky volána metoda `generateCrossWord`, která generuje křížovku. Pokud nastane situace, kdy není možné vygenerovat nový stav křížovky, řídící metoda zastaví generování, načte předchozí stav a opět jej spustí.

Metoda `generateCrossword` provadí již samotné generování křížovky. Tělo této metody generuje řetězce, dokud není fronta bodů křížení prázdná (generování dokončeno). Generování probíha postupně v těcho bodech:

1. Vytvoření nového stavu (záloha fronty bodů křížení, seznamu řetězců a seznamu již vyřešených bodů křížení).
2. Vyjmutí prvního bodu křížení z fronty.
3. Na základě bodu křížení je vypočtena ortogonální maska, dle které je ze slovníku vybrán seznam odpovídajích adeptů.
4. Postupné vkládání adeptů do seznamu řetězců, kontrola ortogonálních masek každého písmene vkládaného řetězce a originality řetězce.
5. Do nového stavu je přidána zbylá část seznamu řetězců, která nebyla kontrolována (úspěšné nalezení řetězce).
6. Vložení stavu na zásobník stavů.
7. Vložení bou křížení do seznamu vyřešených bodů.
8. Vložení nových bodů křížení do fronty.
9. V případě, že při generování nastala chyba, návrat na předchozí stav.

Jakmile metoda generování `generateCrossWord` dokončí svoji práci, je vráceno řízení řídíci metodě, která spustí metody pro vygenerování legend k jednotlivým řetězcům. Nakonec je vráceno řízení hlavnímu vláknu, kde proběhne překopírování vygenerovaných řetězců a legend do mříže textových bloků. Tím je generování křížovky dokončeno. Může také nastat situace, kdy křížovku není možné vygenerovat (špatný návrh křížovky). Potom je generování ukončeno s chybným hlášením.

Slovník výrazů

Vytvoření naindexovaného slovníku výrazů zajišťuje třída `CrossDictionary`, která v konstruktoru inicializuje seznam délek řetězců. Při vytváření instance této třídy musíme tedy zadat parametr seznam délek. Indexovaní slovníku předchází veřejná metoda `createVocabulary`, která požádá databázi o poskytnutí všech řetězců, které svou délkou odpovídají položce v seznamu délek. Jakmile databáze provede potřebné výběry a dokončí je, odešle data metodě `gen_createVocabularyCompleted`, která spustí indexování slovníku. Metoda pro indexování pracuje na principu 4.3, který byl uveden v textu výše.

Mezi další veřejné metody, které tato třída implementuje patří:

- Vyhledání slov ve slovníku dle regulárního výrazu (masky) a sestavení seznamu.
- Kontrola ortogonálních masek (využití předchozí metody).
- Vytvoření seznamu legend dle výrazu, který se nachází v křížovce.

Seznam křížovek

Stránku se seznamem křížovek řídí třída `ListOfCrosses`. Konstruktor této třídy spouští metodu pro načtení uložených křížovek z databáze. Jakmile jsou data k dispozici, spouští se metody, které vytvoří grafický seznam křížovek, kde u každé křížovky je popis a tlačítka „Jít lušit“, „Smazat“. Kliknutím na tlačítko „Smazat“, dojde k přechodu na metodu, která odešle požadavek smazání konkrétní křížovky. Kliknutím na tlačíko „Jít lušit“ dojde k přesměrovní na stránku s řešením křížovky. V URL adrese se přenese parametr obsahující identifikátor křížovky v databázi.

Řešení křížovky

Stránku s řešením křížovky řídí třída `DoCrossWord`. Jako první je spuštěna metoda detekující parametry URL adresy. Tato metoda zpracuje identifikátor křížovky v parametru a zavolá metodu `loadCrossWord` pro stažení křížovky z databáze. Je provedena analýza křížovky a podle zjištěných informací je vytvořena grafická podoba křížovky. Opět se křížovka skládá z mříže a textových bloků. Buňky legend, tajenky a prazdnych poliček jsou barevně odlišeny. Při řešení křížovky může uživatel využít možnosti zobrazit celé řešení nebo zobrazit chyby v křížovce. Při zobrazení řešení se provede záloha vyplněné křížovky a křížovka se vyplní řešením, při návratu se provede zpětný postup. Při zobrazení chyb se chybně vyplněná pole obarví na červeno. Abychom zjistili, zdali se chyby v křížovce vyskytují, postupně porovnáváme originál s řešením uživatele a chybné buňky obarvujeme.

Publikování křížovky

Aplikace nabízí tři možnosti jak lze křížovku publikovat:

- přímý tisk,
- obrázek ve formátu png,
- dokument ve formátu pdf.

Kliknutí na tlačítko „tisk“ obsluhuje metoda, která vyvolá dialogové okno pro výběr tiskárny a nastavení tisku. Uložení křížovky jako obrázku nebo dokumentu pdf řídí metody, které vyvolají dialogové okno pro zadání umístění a názvu souboru. Abychom docílili toho, že se bude tisknou pouze křížovka a nikoli i vzhled aplikace, musíme v jazyce XAML použít objekt `Canvas`, který bude rodičovským objektem mříže s textovými bloky a dále pracovat pouze s ním. Dále tento objekt převedeme na obrázek, který buď uložíme do souboru nebo vložíme do vytvářeného pdf. Pro převod objektu `Canvas` na obrázek a pro vytvoření pdf jsou využity dodatečné knihovny *ImageTools* a *SilverPDF*.

Práce s databází

V aplikaci nebylo možné s databází komunikovat napřímo. Bylo proto nutné vytvořit servisní službu, která bude implementovat dotazy a požadavky směřované databázi. Toto omezení je z důvodu bezpečnosti při práci s databází. Ve službě byly implementovány tyto metody:

- `createVocabulary` – pro výběr slov výrazů z databáze.
- `LoadCrosses` – načtení křížovek z databáze.
- `SaveCross` – uložení křížovky do databáze.
- `DeleteCross` – odstranění křížovky z databáze.
- `TakeCross` – vybere konkrétní křížovku z databáze.

Kapitola 8

Testování aplikace

V této kapitole budou popsány testy, které aplikace podstoupila při dokončení implementace.

8.1 Test rychlosti generátoru

Postupně byly testovány všechny čtvercové velikosti křížovky s různými tajenkami. Každé měření se skládalo z deseti dílčích měření, kde byla generována křížovka pokaždé s rozdílnou tajenkou. Všechny časy byly naměřeny a výsledky byly zapsány do tabulky. Všech třináct souborů měření proběhlo úspěšně. Doby generování můžeme vidět v tabulce 8.1. Je těžké určit přesné doby generování a tím upozornit uživatele jak dlouho se bude křížovka vytvářet. Doba generování je závislá na návrhu křížovky a na vhodně zvolené tajence. Pokud při návrhu křížovky rozmístíme legendy tak, aby se v křížovce vyskytovaly všechny možné délky řetězců, doba generování se značně zvýší! Pokud do křížovky doplníme nevhodnou tajenku, může se stát, že nebude nalezeno řešení křížovky.

Číslo měření	**Rozměr křížovky**	**Nejlepší čas**	**Nejhorší čas**	**Průměrný čas**
1.	3×3	3.3 s	4.1 s	3.76 s
2.	4×4	23.5 s	27.1 s	25.05 s
3.	5×5	1 min 33 s	2 min 13 s	1 min 42 s
4.	6×6	4 min	6 min 41 s	4 min 28 s
5.	7×7	4 min 29 s	6 min 37 s	5 min 40 s
6.	8×8	5 min 7 s	7 min 2 s	5 min 51 s
7.	9×9	6 min 25 s	12 min	9 min 15 s
8.	10×10	7 min 15 s	12 min 59 s	10 min 11 s
9.	11×11	8 min 25 s	13 min 55 s	9 min 34 s
10.	12×12	11 min 6 s	14 min 31 s	12min 22 s
11.	13×13	9 min 38 s	16 min 54 s	13 min 30 s
12.	14×14	9 min 23 s	17 min 11 s	12 min 45 s
13.	15×15	13 min 30 s	20 min 42 s	15 min 32 s

Tabulka 8.1: Test rychlosti generátoru.

Kapitola 9

Závěr

Cílem této práce bylo vytvořit webovou aplikaci pro generování a řešení křížovek. V dnešní době je řešení křížovek vysoce populární zábava, obzvláště u starší generace. Tato internetová aplikace může pomoci lidem, kteří našli zalíbení právě v řešní křížovek, přispět k obohacení jejich života. Pro lepší využití je možné vytisknout křížovku a pokud se rozhodneme pro online řešení křížovky, lze využít variability velikosti křížovky pro lepší řešení.

Pro generování křížovky byl využit algoritmus umělé inteligence backtracking a princip obalování slov. Aplikace byla vytvořena pomocí technologie Silverlight 4 a jazyka C#. Při návrhu a implementaci uživatelského rozhraní byl kladen důraz na jeho použitelnost, efektivnost a interaktivitu. Proto byl také zhotoven moderní grafický návrh. Studium technologie Silverlight mi dodalo nový pohled na webové aplikace a díky volně dostupných materiálů bylo uvedení do problematiky jednodušší.

Při pokračování ve vývoji aplikace by bylo možné se zaměřit na optimalizaci generátoru s hlubším teoretickým základem. Dále by bylo vhodné v aplikaci doimplementovat funkci, která by umožňovala generovat křížovky s danou obtížností, doposud generované křížovky jsou značně složité. Komunikace s databází je problematická s přenášením velkých množství dat, další vývoj by se mohl ubírat i tímto směrem a zefektivnit tak rychlost přenosu dat.

Po dokončení aplikace byl založen webový hosting, kde byla aplikace umístěna a je volně dostupná k vyzkoušení (http://www.crossword.asp2.cz). Aplikace zde pouze neumožňuje generování, protože hosting poskytuje příliš malý prostor pro databáze a s malým počtem výrazů nelze vygenerovat křížovku. Uživatel může křížovku navhrnout například podle novin a opsat tak legendové výrazy, i tak se dají křížovky vytvářet v této aplikaci.

Práce na tomto projektu byla velkým přínosem nových znalostí a zkušeností s doposud pro mě nevyzkoušenou technologií.

Literatura

[1] Beres, J.; Evjen, B.; Rader, D.: *Professional Silverlight 4*. Wiley Publishing Indianapolis, 2010, ISBN 978-0-470-65092-9, 842 s.

[2] Fábera, V.: Genetické algoritmy [online]. `http://www.fd.cvut.cz/personal/xfabera/HNS/prednasky/genet_algoritmy.pdf`, 2010-11-2 [cit. 2011-4-16].

[3] Horváth, T.: .NET Framework [online]. `http://programujte.com/?akce=clanek&cl=2008120700net-framework`, 2008-10-12 [cit. 2011-4-16].

[4] Košťál, M.: WCF - Základné pojmy [online]. `http://www.vyvojar.cz/Articles/454-wcf-zakladne-pojmy.aspx`, 2007-2-11 [cit. 2011-4-16].

[5] Lacko, L.: Silverlight 3.0 [online]. `http://msdn.microsoft.com/cs-cz/dd727769`, 2009-8-10 [cit. 2011-4-16].

[6] Lacko, L.: Silverlight 4.0 [online]. `http://msdn.microsoft.com/cs-cz/dd727769`, 2010-4-7 [cit. 2011-4-16].

[7] Microsoft: VISUAL STUDIO 2010 [online]. `http://www.microsoft.com/visualstudio/en-us/`, 2010-11-2 [cit. 2011-4-16].

[8] Microsoft: Introduction to WPF [online]. `http://msdn.microsoft.com/en-us/library/aa970268.aspx`, 2011 [cit. 2011-4-16].

[9] Microsoft: Projekty webových aplikací versus webové projekty [online]. `http://msdn.microsoft.com/cs-cz/library/dd547590.aspx`, 2011 [cit. 2011-4-16].

[10] Microsoft: Přehled technologie ASP.NET [online]. `http://msdn.microsoft.com/cs-cz/library/4w3ex9c2.aspx`, 2011 [cit. 2011-4-16].

[11] Microsoft: What Is Windows Communication Foundation [online]. `http://msdn.microsoft.com/cs-cz/library/ms731082.aspx`, 2011 [cit. 2011-4-16].

[12] Skalický, I.: *Program pro automatické generování křížovek*. Diplomová práce, FIT VUT v Brně, 2008.

[13] Zbořil, F.; Zbořil, F.: Základy umělé inteligence IZU [online]. `https://www.fit.vutbr.cz/study/courses/IZU/private/oporaizu-esf-4.pdf`, 2007-5-31 [cit. 2011-4-16].

Dodatek A

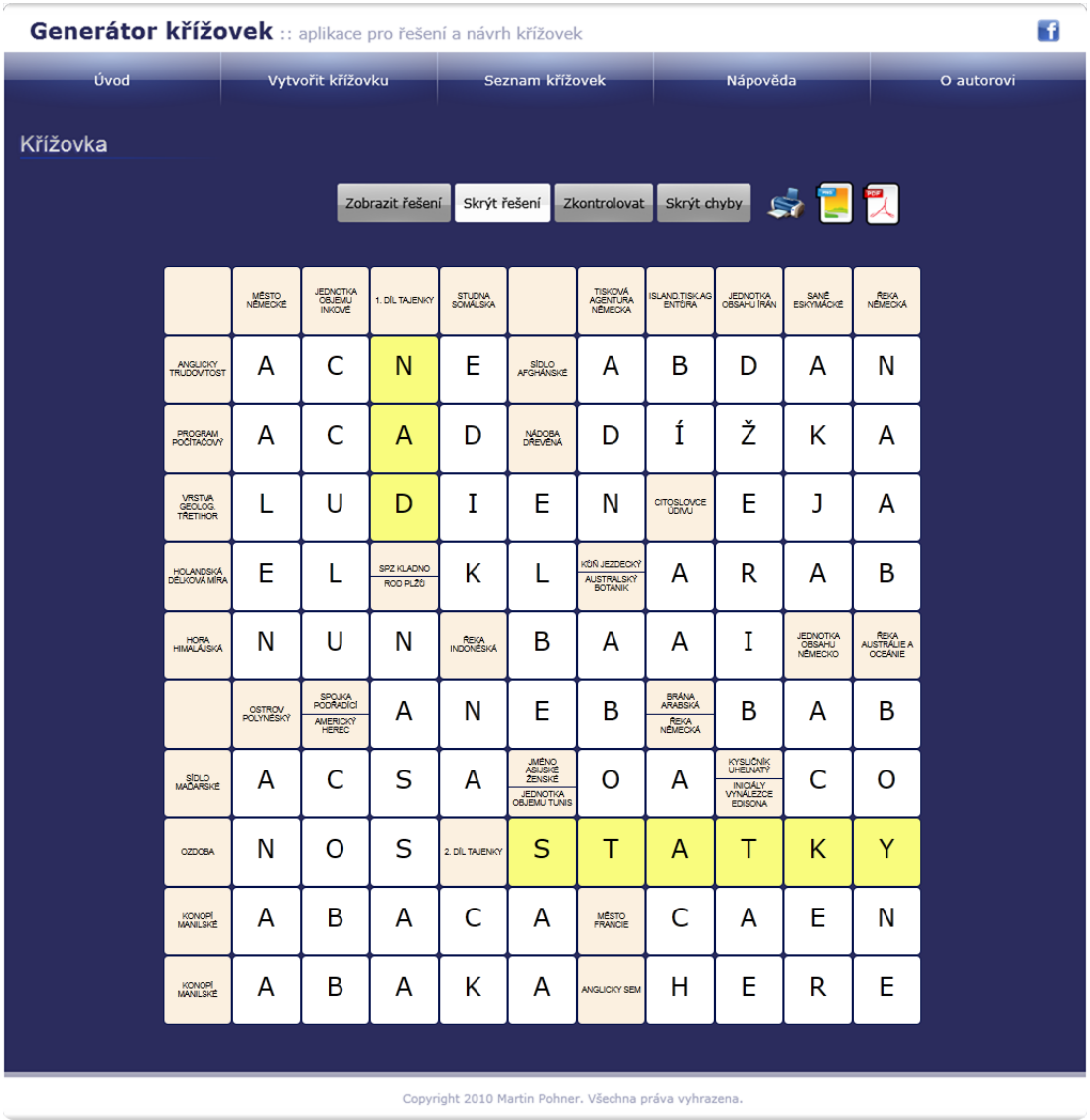
Obsah DVD

Na přiložený disk byly vypáleny všechny soubory, které byly během vývoje aplikace vytvořeny (zdrojové soubory, obrázky atd.). V kořenovém adresáři jsou data přehledně rozčleněna do cíleně pojmenovaných složek. Přehled obsahu disku:

- Silverlight projekt vytvořený ve Visual Studiu 2010.
- Projekt databáze vytvořený ve Visual Studiu 2010.
- SQL skript obsahující slovník výrazů.
- Zdrojové soubory textové části bakalářské práce.
- Textová část bakalářská práce ve formátu pdf.

Dodatek B

Vzhled aplikace



Obrázek B.1: Uživatelské rozhraní aplikace.

Dodatek C

Vygenerovaná křížovka

	BAVLNA EGYPTSKÁ	PROSTOR CHRÁMU ANTICKÉHO	ANGLICKY ŘÍDKÝ	EUNUCH KRÁLE ASVERA		ŘEKA V HORNÍ VOLTĚ	ODCHÝLENÍ OD TĚLNÍ OSY	ÚTVAR NA MARSU	JEŽÍŠ V KORÁNU		POHOŘÍ NĚMECKÉ	ATOL INDONÉSIE	HRAVÉ	ANGLICKY KLEKNOUT SI
ŘEKA ŠVÝCARSKÁ	A	A	R	A	ŘEKA INDONÉSKÁ	B	A	A	I	ANGLICKY PŘEKÁŽKA	B	A	L	K
ANGLICKY ŽVANIT	B	L	A	B	SKUPINA HUDEBNÍ ŠVÉDSKÁ	A	B	B	A	UHLOVODÍK AROMATICKÝ / VĚC SKUTEČNÁ, HMOTNÁ	A	R	E	N
JEDNOTKA DÉLKY DÁNSKO	S	T	R	A	A	ANO (RUS.)	D	A	MĚSTO FINSKÉ	R	A	A	H	E
ŠPANĚLSKY ŽNĚ	S	I	E	G	A	TVOŘENÍ VŘEDŮ	U	L	C	E	R	A	C	E
MPZ ISLANDU	I	S	1. DÍL TAJENKY	T	R	E	K	O	V	Á	ČÍSLOVKA YAPSKÁ	N	E	L
	ČLUN RÝNSKÝ	ZNAČKA ABAMPERU / PILOTA	A	A	FIRMA OTROKOVICKÁ / SYN DCERY ŠEŠANOVY	A	C	S	SLABIKA SOLMIZAČNÍ / SPZ PRAHA MĚSTO	L	A	INICIÁLY DIRIGENTA ANČERLA	ORGANISACE AFRICKÉ JEDNOTY	LET.SPOL. JIHO AFRICKÉ REP.
AMATÉRSKÁ ATLET.ASOCIAC E	A	A	A	ESO (ANGL.) / MĚSTO FINSKÉ	A	C	E	SYN BOHA DIA A AIGINY / MĚSTO PERU	A	I	A	K	O	S
AUTOMOBIL ITALSKÝ (ZN.)	A	B	A	R	T	H	HORA INDICKÁ POSVATNÁ / MĚSTO ROVNÍKOVÉ GUINEY	A	B	U	AMATÉRSKÁ ATLET.ASOCIAC E / PROSTOR CHRÁMU	A	A	A
MODEL FORDU	K	A	ŘEKA FRANCOUZSKÁ / MĚSTO ARMÉNIE	A	A	PŘÍPRAVEK NA NEHTY (ZN.) / OŘECH (DRUH)	A	B	BOHYNĚ KAPADOCKÁ / ZANIKLÝ PRVEK NITRON	M	A	MPZ JAMAJKY / ETIOPSKÝ BĚŽEC	J	A
	ŘEKA NĚMECKÁ	STROM ASIJSKÝ	P	A	J	A	S	A	N	KÓD BOSNY A HERCEGOVINY / RHENIUM	B	A	ŠPANĚLSKY LAVINA	ŘEKA BÝVALÉ JUGOSLÁVIE
AUSTRALSKÝ BOTANIK	A	B	O	T	BASA	K	O	N	T	R	A	B	A	S
MĚSTO DÁNSKÉ	A	A	L	E	AMER.TELEVIZ NÍ SPOLEČNOST	A	B	C	SLOVENSKÉ ŽENSKÉ JMÉNO	E	T	E	L	A
KOLEM	C	C	A	MĚSTO IZRAELSKÉ	E	J	L	A	T	2. DÍL TAJENKY	O	B	U	V
BOA	H	A	D	PLATIDLO MALAJSIE	B	U	A	Y	A	CIZÍ ŽENSKÉ JMÉNO	N	E	D	A

Obrázek C.1: Vygenerovaná křížovka.

Dodatek D

Prázdná křížovka

	BAVLNA EGYPTSKÁ	PROSTOR CHRÁMU ANTICKÉHO	ANGLICKY ŘÍDKÝ	EUNUCH KRÁLE ASVERA		ŘEKA V HORNÍ VOLTĚ	ODCHÝLENÍ OD TĚLNÍ OSY	ÚTVAR NA MARSU	JEŽÍŠ V KORÁNU		POHOŘÍ NĚMECKÉ	ATOL INDONÉSIE	HRAVĚ	ANGLICKY KLEIKNOUT SI
ŘEKA ŠVÝCARSKÁ					ŘEKA INDONÉSKÁ					ANGLICKY PŘEKÁŽKA				
ANGLICKY ŽVANIT					SKUPINA HUDEBNÍ ŠVÉDSKÁ					UHLOVODÍK AROMATICKÝ / VĚC SKUTEČNĚ HMOTNÁ				
JEDNOTKA DÉLKY DÁNSKO						ANO (RUS.)			MĚSTO FINSKÉ					
ŠPANĚLSKY ŽNĚ						TVOŘENÍ VŘEDŮ								
MPZ ISLANDU			1. DÍL TAJENKY								ČÍSLOVKA YAPSKÁ			
	ČLUN RÝNSKÝ	ZNAČKA ABAMPÉRU / PILOTA			FIRMA OTROKOVICKÁ / SYN DCERY ŠEŠANOVY				SLABIKA SOLMIZAČNÍ / SPZ PRAHA MĚSTO			INICIÁLY DIRIGENTA ANČERLA	ORGANISACE AFRICKÉ JEDNOTY	LET.SPOL.JIHO AFRICKÉ REP.
AMATÉRSKÁ ATLET.ASOCIAC E				ESO (ANGL.) / MĚSTO FINSKÉ				SYN BOHA DIA A AIGINY / MĚSTO PERU						
AUTOMOBIL ITALSKÝ (ZN.)							HORA INDICKÁ POSVATNÁ / MĚSTO ROVNÍKOVÉ GUINEY				AMATÉRSKÁ ATLET.ASOCIAC E / PROSTOR CHRÁMU			
MODEL FORDU			ŘEKA FRANCOUZSKÁ / MĚSTO ARMÉNIE			PŘÍPRAVEK NA NEHTY (ZN.) / OŘECH (DRUH)			BOHYNĚ KAPADOCKÁ / ZANIKLÝ PRVEK NITRON			MPZ JAMAJKY / ETIOPSKÝ BĚŽEC		
	ŘEKA NĚMECKÁ	STROM ASIJSKÝ								KÓD BOSNY A HERCEGOVINY / RHENIUM			ŠPANĚLSKY LAVINA	ŘEKA BÝVALÉ JUGOSLÁVIE
AUSTRALSKÝ BOTANIK					BASA									
MĚSTO DÁNSKÉ					AMER.TELEVIZ NÍ SPOLEČNOST				SLOVENSKÉ ŽENSKÉ JMÉNO					
KOLEM				MĚSTO IZRAELSKÉ						2. DÍL TAJENKY				
BOA				PLATIDLO MALAJSIE						CIZÍ ŽENSKÉ JMÉNO				

Obrázek D.1: Křížovka připravená k řešení.

Dodatek E

Vygenerovaná křížovka

	AMATÉRSKÁ ATLET.ASOCIACE	POČÍTAČ AMERICKÝ (ZN.)	1. DÍL TAJENKY	MĚSTO AUSTRÁLIE	POLVIDRO NA TYČINKY	ANGLICKY PŘÍSAHA	DENÍK SLOVENSKÝ		TURECKÁ TISKOVÁ AGENTURA	ČADSKÁ STUDNA	POKRM JAPONSKÝ	HORA NA SUMATŘE	NEVOLNÍK (ZAST.)	ZÁLIV JAPONSKA
SYN KRÁLE ELATA	A	I	P	Y	T	O	S	VELBLOUD	A	A	Š	A	R	I
PRAOTEC IZRAELITŮ	A	B	R	A	H	A	M	PROROK APOŠTOLSKÉ DOBY	A	G	A	B	O	S
BŮŽEK LÁSKY	A	M	O	R	E	T	E	K	USAZENINY SPRAŠOVÉ V USA	A	B	O	B	E
	MĚSTO NĚMECKÉ	VÝPACH / IZRAELSKÝ BÁSNÍK	P	A	C	H	KANCELÁŘSKÁ ZKRATKA / OSTROV BRITSKÝ	A	A	ANGLICKÁ ZKRATKA PRO OSN / ANGLICKY UŽITEK	U	N	BAKCHANTKA	HLAVNÍ MĚSTO SRBSKA
KONOPÍ MANILSKÉ	A	B	A	K	A	ŘEKA ŠVÝCARSKÁ / BŮH VOD	A	A	R	A	KÓD GAMBIE / JEZERO DŽIBUTSKA	G	M	B
BOHYNĚ ŽIVOTNÍHO OSUDU	A	I	S	A	ZÁBAVA (NÁŘ.)	P	O	B	A	V	A	TĚLESO KOSMICKÉ AMERICKÉ / ROSTLINA TROPICKÁ	A	E
OSTROV SOUOSTROVÍ BAHAMY	C	A	T	OSTROV HEBRIDSKÝ / PŘIPOMÍNKA	A	O	R	A	LÁZNĚ ITALSKÉ	A	B	A	N	O
ZKR.DUTÉ MÍRY	H	L	MLHAVÉ / MĚSTO NIZOZEMSKÉ	M	D	L	E	NĚMECKÝ CHEMIK	L	I	E	B	I	G
MÍR	E	I	R	E	N	E	ÚHOŘ (NĚM.)	A	A	L	MĚSTO SOMÁLSKA / ETIOPSKÝ BĚŽEC	U	A	R
HORA GABONSKÁ	N	K	A	M	POHOŘÍ BÝVALÉ JUGOSLÁVIE / ŘEKA IRSKÁ	B	A	B	A	AMER.CÍLOVÁ RAKETA / POHOŘÍ ARMÉNSKÉ	A	T	D	A
	KÓD BURKINY FASO	JAZYK PROGRAMOVACÍ / ROZHL.SPOL.E CUADORU	A	E	D	2. DÍL TAJENKY / DRUH PAPOUŠKA	J	E	VLNA EGYPTSKÁ	A	B	I	A	D
KANCELÁŘ CESTOVNÍ	B	A	L	N	E	A		BIBLICKÁ POSTAVA	A	B	E	L	KILOCYKL	DVOJHLÁSKA LATINSKÁ
ČESKÝ ANTROPOLOG	F	E	T	T	E	R	3. DÍL TAJENKY	H	L	U	B	O	K	O
MEZERA V PLETIVU	A	R	E	O	L	A	NEPŘÍTOMNOST	A	B	S	E	N	C	E

Obrázek E.1: Vygenerovaná křížovka.

Dodatek F

Prázdná křížovka

	AMATÉRSKÁ ATLET.ASOCIACE	POČÍTAČ AMERICKÝ (ZN.)	1. DÍL TAJENKY	MĚSTO AUSTRÁLIE	POUZDRO NA TYČINKY	ANGLICKY PŘÍSAHA	DENÍK SLOVENSKÝ		TURECKÁ TISKOVÁ AGENTURA	ČADSKÁ STUDNA	POKRM JAPONSKÝ	HORA NA SUMATŘE	NEVOLNÍK (ZAST.)	ZÁLIV JAPONSKA
SYN KRÁLE ELATA								VELBLOUD						
PRAOTEC IZRAELITŮ								PROROK APOŠTOLSKÉ DOBY						
BŮŽEK LÁSKY									USAZENINY SPRAŠOVÉ V USA					
	MĚSTO NĚMECKÉ	VÝPACH / IZRAELSKÝ BÁSNÍK					KANCELÁŘSKÁ ZKRATKA / OSTROV BRITSKÝ			ANGLICKÁ ZKRATKA PRO OSN / ANGLICKY UŽITEK			BAKCHANTKA	HLAVNÍ MĚSTO SRBSKA
KONOPÍ MANILSKÉ						ŘEKA ŠVÝCARSKÁ / BŮH VOD					KÓD GAMBIE / JEZERO DŽIBUTSKA			
BOHYNĚ ŽIVOTNÍHO OSUDU					ZÁBAVA (NÁR.)							TĚLESO KOSMICKÉ AMERICKÉ / ROSTLINA TROPICKÁ		
OSTROV SOUOSTROVÍ BAHAMY				OSTROV HEBRIDSKÝ / PŘIPOMÍNKA					LÁZNĚ ITALSKÉ					
ZKR.DUTÉ MÍRY			MLHAVÉ / MĚSTO NIZOZEMSKÉ					NĚMECKÝ CHEMIK						
MÍR							ÚHOŘ (NĚM.)				MĚSTO SOMÁLSKA / ETIOPSKÝ BĚŽEC			
HORA GABONSKÁ					POHOŘÍ BÝVALÉ JUGOSLÁVIE / ŘEKA IRSKÁ					AMER.CÍLOVÁ RAKETA / POHOŘÍ ARMÉNSKÉ				
	KÓD BURKINY FASO	JAZYK PROGRAMOVACÍ / ROZH. SPOL. E CUADORU				2. DÍL TAJENKY / DRUH PAPOUŠKA			VLNA EGYPTSKÁ					
KANCELÁŘ CESTOVNÍ								BIBLICKÁ POSTAVA					KILOCYKL	DVOJHLÁSKA LATINSKÁ
ČESKÝ ANTROPOLOG							3. DÍL TAJENKY							
MEZERA V PLETIVU							NEPŘÍTOMNOST							

Obrázek F.1: Křížovka připravená k řešení.