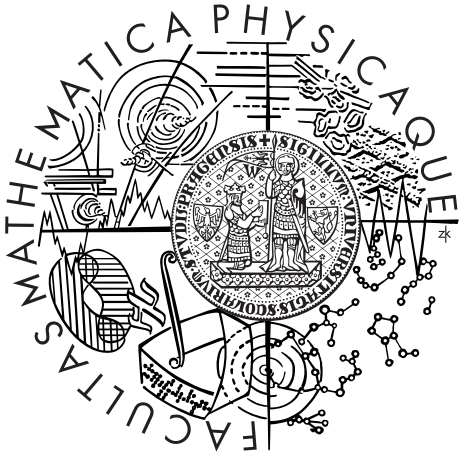
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

BAKALÁŘSKÁ PRÁCE



Jan Pacovský

Rozšíření zim-wiki umožňující plynulé přecházení mezi strukturou stránek a nadpisů

Kabinet sowtware a výuky informatiky

Vedoucí bakalářské práce: Mgr. Daniel Lessner
Studijní program: Informatika
Studijní obor: Programování

Praha 2014

Velice děkuji panu Mgr. Danielu Lessnerovi, vedoucímu mé bakalářské práce, za návrh zadání. Děkuji za čas strávený nad mou prací, za ochotu, laskavost i trpělivost a v neposlední řadě za podnětné návrhy, rady a připomínky.

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ..... Podpis autora

Název práce: Rozšíření zim-wiki umožňující plynulé přecházení mezi strukturou stránek a nadpisů

Autor: Jan Pacovský

Katedra: Kabinet software a výuky informatiky

Vedoucí bakalářské práce: Mgr. Daniel Lessner, Kabinet software a výuky informatiky

Abstrakt: Multiplatformní textový editor Zim umožňuje uchovat a třídit poznámky, propojovat a organizovat texty a spravovat kolekce textových souborů.
Cílem práce bylo vytvoření zásuvného modulu (rozšíření), který dokáže komfortně rozdělovat a spojovat stránky v Zimu, aniž by uživatel ztratil přehled o svých textech a jejich organizační struktuře.
V průběhu řešení se ukázalo nutným zasáhnout i do kódu hlavního programu. Podařila se implementovat nová funkcionalita umožňující převod obsahu stránky na podstrom stránek. Úspěšně se vyřešilo zachování odkazů a nesoulad počtu úrovní nadpisů a stránek v podstromu.
Výsledkem je efektivnější, přehlednější a systematičtější práce s vlastními texty ve stromové struktuře stránek.

Klíčová slova: Zim - Desktopová wiki, Textový editor, Převod nadpisů a stránek, Zásuvný modul, Python

Title: Zim-wiki Plugin for Smooth Transition between Pages and Headings Structure

Author: Jan Pacovský

Department: Department of Software and Computer Science Education

Supervisor: Mgr. Daniel Lessner, Department of Software and Computer Science Education

Abstract: The Multiplatform text editor Zim enables the user to preserve and sort notes, to join and organize texts, and manage a collections of text files. The goal of this thesis was to develop a plugin that would comfortably enable the splitting and merging of pages in Zim without the user losing track of his texts and their organizational structure. While solving the problem it appeared inevitable to modify the main program. New functionality was implemented that allows the transfer of the page content into a page subtree. Links were successfully maintained and the discrepancy in the number of header levels and pages in the subtree was resolved. These modifications result in more effective, more transparent, and more systematic work with texts using the tree structure of the pages.

Keywords: Zim - A Desktop Wiki, Text editor, Transformation between headings and pages, Plugin, Phyton

Obsah

Úvod

Informační společnost, jak je médii nazývána současná doba, oplývá množstvím informací. Vyvstává nutnost informace analyzovat, selektovat a zpracovávat. Počítače mohou za krátkou dobu téměř nahradit papíry a sešity, pokud budou existovat vhodné nástroje na tvorbu a správu poznámek.

Program Zim je jedním z nich. Byl vytvořen pro různé typy poznámek. Umožňuje jejich třídění, organizování a další zpracování. Cílem této bakalářské práce je tento program obohatit o nové funkce, aby se přidalo na jeho praktické využitelnosti a aby došlo ke zvýšení produktivity práce s programem. Zároveň se však snažíme o zachovaní současné intuitivity a jednoduchosti používání.
Motivací pro napsání tohoto pluginu byla skutečnost, že v komunitě uživatelů Zimu byl vznesen požadavek na napsání pluginu, který umožňuje slévání stránek a jejich opětné rozdělení.

Bližšímu seznámení se Zimem věnujeme další kapitolu. Zim dokáže zaznamenat vysoce provázané poznámky. Uchovává je v rámci hierarchie v jednotlivých stránkách. Text poznámky může být rozdělen mezi více stránek. Avšak poznámku rozdělenou do více stránek Zim neumí zobrazit najednou.

Tato práce má umožnit uživateli převádět mezi sebou dvě struktury: Index, jak se nazývá v Zimu strom stránek, a strukturu nadpisů. Tuto strukturu tvoří nadpisy nacházející se v rámci jedné stránky. Přechod se tedy bude uskutečňovat mezi jednou stránkou a skupinou vybraných stránek z Indexu. Tím umožníme zobrazení poznámky rozdělené ve více stránkách v jedné stránce a to tak, aniž by došlo k poškození obsahu a v nich se nacházejících i do nich směřujících odkazů. Přitom by měla být původní stromová struktura patrná z příslušných úrovní nadpisů sloučených stránek (při rozdělení stránky na menší stránky, by pak obsah nových stránek měl odpovídat obsahu pod původními nadpisy). Aby byla stromová struktura patrná i z úrovní nadpisů, je třeba navýšit počet úrovní nadpisů z dosavadních pěti nejlépe na neomezený počet.

Tato bakalářská práce je rozdělena do čtyř kapitol.

V první kapitole se podrobněji zabýváme tím, co je program Zim, co umí, čím je odlišný, jaké jsou jeho principy a jak se používá.

Ve druhé kapitole podrobněji analyzuje problémy, které vyvstaly při psaní tohoto nového programu. Každá její podkapitola obsahuje konkrétní řešenou situaci, specifikuje možná řešení a odůvodňuje vybraný způsob řešení.

Třetí kapitola pojednává o tom, jak se s programem i jeho rozšířením má zacházet a co umožňuje naše nové rozšíření. Zaznamenává změny oproti Zimu, specifikuje uživatelské rozhraní a navrhuje příklady užití.

A v poslední kapitole je popsaná programátorská dokumentace.

Nejprve bylo nutné promyslet, jak zdánlivě jasné a jednoduché zadání konkretizovat. Vyskytla se celá řada otázek. Jak spojovat a rozdělovat stránky, aniž by se narušila jejich struktura. Jakým způsobem přenést a zachytit strukturu Indexu do struktury nadpisů a jak ji přenést tříděním neporušenou zpět. Rovněž bylo nutné promyslet, jak rozšířit strukturu nadpisů tak, aby mohla mít neomezený

počet úrovní. Museli jsme se také vypořádat s odkazy, které vedly v původní struktuře mezi stránkami a po jejich sloučení se ocitly v jediné stránce. Situaci komplikoval samotný Zim, neboť umožňuje výhradně odkazování na celé stránky. Čekalo nás také řešení otázky, nakolik lze nové rozšíření Zimu uskutečnit jako plugin a jestli nebude nutné zasáhnout do zdrojového kódu Zimu. V neposlední řadě bylo nutné se také zabývat běžnou podobou nástrojů, kterými bude uživatel program v rozšířené verzi ovládat.

1. Zim-wiki

V této kapitole se seznámíme s programem, který tato práce rozšiřuje. Zjistíme, proč byl vytvořen, co dělá a jakých zásad se drží. Popíšeme, co vše může stránka obsahovat. Ozřejmíme základy ovládání. Ukážeme si, kde je z uživatelova hlediska vidět dualita v načítání „zdrojového kódu pro vykreslení stránky“. Popíšeme jak se vytváří nadpisy, které v průběhu práce pozměníme.

1.1 Seznámení se Zimem

Zim – Desktopová wiki, zkráceně zim-wiki nebo Zim, je multiplatformní textový editor s jednoduchým vzhledem a podporou zásuvných modulů (pluginů), který umožňuje texty nejen editovat, ale i propojovat a organizovat.

1.1.1 Obecná charakteristika Zimu

Účel programu
Zamýšleným uplatněním Zimu je sloužit jako užitečný nástroj k zachycení osobních poznámek, připomínek, návrhů, nápadů a glos. S úspěchem lze použít pro seznamy úkolů nebo jako pořadač pro organizaci nejrůznějších textů, atd. Lze jej využít i na rozsáhlejší dokumenty. Dokáže nahradit deník, kalendář, poznámkový blok i kalkulačku. Umožňuje efektivní práci s texty, které sdružuje do kolekcí (notebooků). V rámci kolekce i kolekcí se na sebe mohou jednotlivé stránky vzájemně odkazovat. Autorovým záměrem bylo vytvořit program, který by sloužil k rychlému zachycení poznámek tak, aby uživatel nebyl vytrhován od svých myšlenek zbytečnými úkony, které by ho odvedly od obsahu. Neobtěžuje tím, kam se co, jak a kdy má uložit. Stránky se automaticky při změně textu ukládají jako textové soubory na disk.

Historie
Vznik se datuje k roku 2005 nebo 2008. První verze pro veřejnost sice vyšla v září 2005, avšak tehdy se jednalo o verzi napsanou v jazyce Perl. V roce 2008 došlo k rozhodnutí, že se ukončí vývoj perlovské verze. Program se totiž stal populárním a jeho původní architektura nebyla dostatečně modulární, aby umožňovala snadné přidávání pluginů. Proto se celý program přepsal do jazyku Python.

Autorství
Zim je uvolněn pod licencí GNU GPLv2 – má otevřený zdrojový kód. Jeho autorem je nizozemský programátor Jaap Karssenberg a v současné době se na vývoji Zimu podílí podle launchpadu přibližně dvacet programátorů a téměř 500 zainteresovaných lidí. [1].

Wiki
Zim přináší koncept wiki známé z webu do uživatelova počítače, proto je v názvu programu podtitul „Desktopová wiki“. Snaží se uchopit ideu wiki, ne jako www stránek, ale jako kolekce dokumentů, které má uživatel ve svém počítači a může je editovat pomocí WYSIWYG GUI aplikace. Stránky jsou organizovány

do stromové struktury. Soubory se ukládají na disk do strukturovaných složek podle toho, kde se stránky ve stromové hierarchii nalézají. Stránka v této hierarchii téměř vždy odpovídá souboru na disku a strom stránek stromu složek. Pokud vytvoříme odkaz na dosud neexistující stránku, Zim ji v místě, na které odkaz směřuje, připraví.

Syntaxe
Stránky jsou popsány vlastní wiki syntaxí [2] — značkovacím jazykem (markup language). Zobrazují se jako formátovaný text a jejich obsah je společně s wiki syntaxí ukládán do souborů. Veškeré značky (tagy) využívají alespoň dva znaky sloužící k redukci rizika konfliktů s textem.

Soubory s wiki syntaxí lze otevřít i v jiných textových editorech. Formátování je srozumitelné i pro wiki syntaxe neznalého člověka a samotný text zůstává velmi dobře čitelný, protože syntaxe v Zimu přímo vychází z DokuWiki [3], která klade důraz právě na dobrou čitelnost.

1.2 Hlavní myšlenky Zimu

Být přívětivý, předvídatelný, pohodlný, neobtěžovat.

Styly
Nechat uživatele soustředit na samotné zachycení myšlenek a neodvádět zbytečně jeho pozornost. Styly nemají rozptylovat, ať už by to bylo příliš mnoho nebo příliš málo, důležitý je obsah. Uživatel nemá přemýšlet o formátování textu. Styly mají sémantický význam. Jsou nosičem informace.

Databáze
Nemít skryté informace. Databáze jsou podporou pro fungování programu. Nejsou jediným zdrojem informací, které nejsou dostupné jinde. Díky tomuto přístupu lze všechna uživatelská data, která jsou v jednom adresáři jednoduše přenést nebo doplnit. Není třeba nic importovat nebo exportovat. Správa databází při konfliktech, údržbách nebo problémech je jednoduchá, databáze se smaže.

1.3 Vzhled a obsah stránky

V editoru se nedá měnit barva písma, font ani jeho velikost. Globálně je však lze pro jednotlivé tagy nastavit v konfiguračním souboru a navíc velikost písma lze nastavit přes klávesové zkratky: „Ctrl-+“ a „Ctrl-–“.

Text lze pomocí značek, které určují vzhled: zvýraznit, přeškrtnout, napsat tučně, kurzívou nebo strojopisem, napsat jako horní nebo dolní index, změnit na nadpis. Značek pro nadpisy je celkem pět. Tagy se nesmí křížit a ani se do sebe nemohou zanořovat – což například znamená, že nemůžeme mít text psaný kurzívou, který je zároveň tučný. Stejně tak nemůžeme docílit toho, aby část nadpisu byla jinak formátovaná než je jeho zbytek.

V rámci textu navíc mohou být zaškrtávací, číslované nebo odrážkové seznamy. Uvnitř stránky se mohou vyskytovat odkazy na jiné stránky, soubory nebo

speciální odkazy na obrázky, tím se obdobně jako v TEXu či LATEXu obrázky zobrazí ve výsledném naformátovaném textu.

Z tohoto kompletního výčtu je patrné, že uživatele nerozptyluje příliš velké množství možností, ale pro běžnou práci je plně postačující.

1.3.1 Formátování textu

Formátování textu v editoru je intuitivní a lze k němu použít tlačítka na liště, položku v Menu nebo klávesovou zkratku – ty jsou uvedeny v Menu. Případně jsou v nápovědě pod položkou klávesové zkratky. Zde jsou vypsány zkratky typu „Ctrl-tlačítko“. Kromě těchto zkratek lze v Zimu psát rovnou v zim-wiki syntaxi.

V nápovědě, kterou vyvoláme stiskem klávesy F1, je v levém panelu ve stromu položka Wiki Syntax (viz obr. 3.3). Pokud na ni klikneme levým tlačítkem myši, otevře se jí odpovídající stránka v hlavním panelu. A tam na výborně zpracovaných příkladech vidíme kompletní syntaxi.

Pokud bychom chtěli napsat odrážkový seznam, klikneme v liště na „Insert“ a pak „Bullet List“. Tím se vloží odrážka před označené místo. Pomocí kláves Tab a „Shift-Tab“ lze měnit její odsazení. Po stisknutí klávesy Enter, Zim, pokud odrážka obsahuje nějaký text, doplní na další řádek novou odrážku.

Jiný způsob vložení odrážkového seznamu vyžaduje jej pomocí zim-wiki syntaxe zapsat přímo v textu. Lze jej vytvořit tak, že napíšeme hvězdičku a bezprostředně za ní mezeru nebo tabulátor „* “ a Zim nahradí hvězdičku za odrážku. Takto automaticky nahrazování nefunguje pro všechnu syntaxi.

Nahrazení například nefunguje pro odkazy, které se zapisují do hranatých závorek (`[[odkaz]]`. Aby se takovýto odkaz změnil z prostého textu na aktivní odkaz bez hranatých závorek, musíme stránku opustit a vrátit se na ni nebo stisknout „Ctrl-R“ pro její obnovení. Tím dojde k uložení na disk a celý text se znovu bude parsovat.

Převod textu z wiki syntaxe na odrážku dokonce nebude fungovat, ani pro již zmíněnou hvězdičku, pokud ji zapíšeme tak, aby bezprostředně za ní nebyl napsán bílý znak, ale až za nebílým znakem. Pokud potom smažeme nebílý znak nedojde k nahrazení za odrážku. Text zůstane v prosté podobě, dokud neprovedeme nějakou výše uvedenou akci.

Tímto způsobem lze obejít „on the fly“ vyhodnocování. Narazili jsme zde na projev duality Zimu při parsování textu v závislosti odkud text přichází – z disku nebo z klávesnice.

1.3.2 Psaní nadpisů

Nadpisy lze vytvářet pomocí Menu, pomocí klávesových zkratek „Ctrl-1“ až „Ctrl-5“ a pomocí wiki syntaxe.

Pro hlavní úroveň nadpisu napíšeme dva znaky rovnítek za sebe mezeru a text nadpisu (`== nadpis první úrovně`) a pro každou nižší úroveň přidáváme vždy jedno další znaménko rovnosti (`====== nadpis páté úrovně`).

Na nadpisech je opět vidět dualita Zimu, použijeme-li obdobný trik jako v předchozím odstavci, dospějeme k zajímavému pozorování. Podle toho, jakou cestu překladu syntaxe zvolíme, budou mít nadpisy různou úroveň. Zvolíme-li

cestu dosažitelnou pomocí „Ctrl-R“, pak je význam počtu rovnítek opačný (`== nadpis Páté úrovně`), navíc se v uloženém souboru nalézá za nadpisem stejný počet znamének rovnosti jako před nadpisem (`=== nadpis ===`).

1.3.3 Podobné programy

K Zimu existuje množství alternativ. Je to dáno také tím, že je velmi univerzální. Existují programy, které jsou specializované na malé části toho, co zim dělá. Zim lze používat jako kalendář, deník, kalkulačku, ale především jako textový editor. A především jako editor poznámek. Nejznámější komerční alternativou je OneNote od Microsoftu, nekomerční Tomboy.
Alternativa k rozšíření, které přináší tato bakalářská práce asi neexistuje.

2. Analýza rozšíření

V této kapitole se budeme věnovat všemu tomu, co je potřeba promyslet a rozhodnout před tím, než se začne programovat. Nejdříve si zopakujeme co je naším úkolem a co potřebujeme vědět, a poté začneme řešit jednotlivé problémy.

Zadání: *„Cílem práce je umožnit uživateli přenést strukturu části stromu stránek do jediné stránky a zpět. Všechen obsah přitom zůstane zachován, původní struktura stránek bude patrná z nadpisů příslušných úrovní. Opačný směr funguje analogicky, ze stránky obsahující nadpisy různých úrovní se vytvoří strom stránek. Jejich obsah bude odpovídat obsahu pod původními nadpisy.“*

2.1 Základní seznámení

Stromová struktura stránek, jak jsme již v úvodu nazývali Index, ve skutečnosti není stromem, ale lesem, protože nemá dosažitelný kořenový uzel. Jelikož se však ve zdrojovém kódu Zimu mluví o stromu, budeme ji tedy nazývat stromem, ale s vědomím toho, že pokud v dalším textu explicitně nezmíníme kořenový uzel, tak o něm nemluvíme. Každá stránka je právě jedním uzlem Indexu a může mít neomezené množství jak bratrů, tak i synů.

Stránkou rozumějme položku Indexu, která se zobrazuje jako formátovaný text v hlavním widgetu. Text sám o sobě může obsahovat pět úrovní nadpisů, odkazy, různé formátovaní obsahu, obrázky a přílohy.

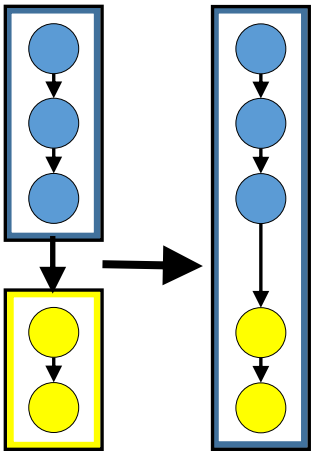
Nadpisy, i když to není na první pohled zjevné, mají jasně danou strukturu, která však není nikde definována, není to třída (jako Index), díky tomu s nimi máme vetší volnost v nakládání. Obdobně jako u Indexu se jedná o lesovou strukturu. Začínají nadpisem první úrovně, ale chybí jim nultá, která by mezi sebou propojovala nadpisy první úrovně. Jsou součástí stránky, od ostatního textu rozlišitelné pouze formátováním. Úrovně nadpisů nejsou náhodné, ale přibývají postupně – další úroveň může být libovolně nižší, stejná jako současná nebo o právě jednu vyšší.

Naším úkolem je přenést část Indexu dovnitř stránky, aby ji vyjadřovaly pouze nadpisy a nazpět. Jinými slovy: chceme umožnit přecházení mezi strukturou nadpisů a strukturou Indexu.

Nejmenší částí Indexu je stránka, tu převádět nemusíme. Další, větší část Indexu, je stránka se svým synem.

Zvolme tuto stránku modrou a se třemi úrovněmi nadpisů a jejího syna žlutého, se dvěma úrovněmi. Nejjednodušším způsobem je sloučit stránky dohromady a nadpisy ze žlutého syna snížit na úroveň pod nadpisy modré stránky (viz obr. 2.1). Tím se nám podařilo přenést část Indexu dovnitř stránky. Pokud bychom chtěli tuto stránku rozdělit, nebudeme mít informaci o tom, kde jedna původní stránka přechází v druhou (barva je zde jen pro naši lepší představu). Nepoznáme ani, o kolik sloučených stránek se jedná. Na tomto příkladu vidíme, že nejsme schopni vrátit původní podobu stránek. Skrytým předpokladem se ukázalo zajištení vratnosti změn.

Rovněž jsme si ujasnili, že převod struktur se kvůli Indexu může týkat pouze celých stránek.



Obrázek 2.1: Příklad slučování. Vlevo vztah mezi stránkami se zobrazenými nadpisy, vpravo již jen jako jedna stránka s nadpisy.

2.1.1 Řešení

K našemu úkolu, přenesení části Indexu dovnitř stránky a zpět, přibyl požadavek: dostat se co nejblíže k originálnímu stavu po provedení obou změn. Chceme vratnost změn. Abychom mohli mezi sebou struktury převádět, musíme se s nimi nejprve důkladně seznámit. Proto se v následujících částech se budeme detailněji zabývat tím, jak dané struktury vypadají, v čem jsou si podobné a v čem se liší a zdali na sebe půjdou převést.

2.1.2 Detailnější seznámení se strukturami

Index
Index automaticky uspořádává stránky podle jejich názvu abecedně. Umožňuje přesun a vytvoření stránek tak, že se stanou synem jakékoli stránky (včetně kořenové). Při přesunu se automaticky přesouvá stránka i s veškerými potomky. Index ale neumožňuje označení více než jedné stránky.

Struktura nadpisů
Nadpisy abecedně seřazeny být nemusí, umožňují přesun a vytvoření nových synovských nadpisů s výjimkou poslední úrovně, protože nižší již nelze vytvořit.

Podstromem současného nadpisu budeme nazývat všechny nadpisy nižší úrovně až do dalšího nadpisu stejné úrovně nebo konce stránky.

Přesun nadpisu lze realizovat nejen s jeho podstromem, ale i se samotným nadpisem za podmínky, že předchozí nadpis je maximálně o úroveň nižší než následující nadpis. Navíc, pro manipulaci lze vybrat více než jeden nadpis, lze libovolně přesunovat nadpisy s jejich celými podstromy v rámci stejné úrovně beze změn. V rámci různých úrovní se musí změnit všechny úrovně nadpisů a musí se vejít do rozsahu úrovní.

Ve struktuře nadpisů, kromě omezení počtu úrovní a neexistence odkazů, máme větší možnosti i volnost. Lze vybrat najednou více nadpisů z různých částí,

jeden nadpis, podstrom nadpisu i více podstromů.

Přesun samotné stránky, bez jejího podstromu, lze realizovat i v Indexu. Nejdříve se přesune i stránka i s podstromem a poté se podstrom vrátí na původní místo.

Zásahy do Indexu se budeme snažit omezit na minimum, protože je Index v současnosti předěláván samotným autorem. Potřeba zásahu bude jen pro případ, že bychom chtěli všechny stránky sloučit do jedné.

Vlastnosti	**Index**	**Nadpisy**
Řazení prvků ve struktuře	Abecední	Libovolné
Hloubka struktury	Neomezená	Pět (5) úrovní
Šířka struktury	Neomezená	Neomezená
Výběr více než jednoho prvku	Ne	Ano
Vytvořit syna lze v úrovních	Ve všech	Do páté (1–4)
Přesun prvku i s potomky	Ano	Ano
Přesun prvku bez potomků	Ne	Někdy
Existuje odkazovat na jiné prvky	Ano	Ne

Tabulka 2.1: Rozdíly struktur.

V dalších částech se budeme postupně zabývat řešením těchto problémů:

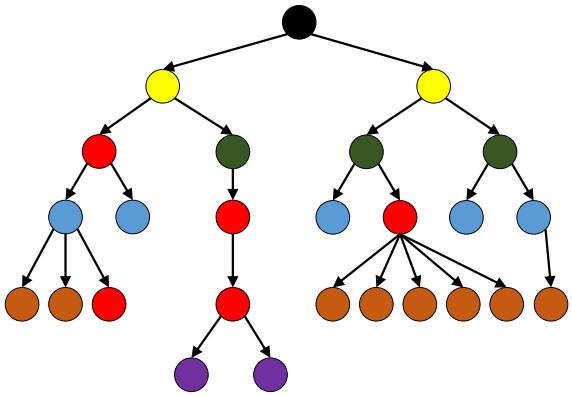
- jak vybrat část Indexu (viz 2.2);
- jak vybrat část struktury nadpisů (viz 2.2);
- jak sjednotit velikost struktur – počet úrovní nadpisů a Indexu (viz 2.3);
- jak převést prvek Indexu na prvek struktury nadpisu –
  – celou stránku (vč. obsahu – i s nadpisy) na jeden nadpis (viz 2.4);
- jak se vyrovnat s různým řazením prvků (viz 2.5);
- jak se chovat k odkazům (viz 2.6).

2.2 Výběr části struktury

Víme, že slučovat můžeme stránky. Nejdříve si ujasníme jak vybrat skupinu stránek ke sloučení. Velikost této skupiny je alespoň dva.

Díky již zmíněným vlastnostem Indexu lze vybrat jednu stránku nebo její potomky, což je podstrom, nebo oboje stránku i její potomky. Na rozdíl od struktury nadpisů neumí Index vybrat více stránek najednou v různých částech stromu. Pokud bychom jej však chtěli připodobnit nadpisům (viz třetí řádek tabulky 2.1), znamenalo by to zásadní zásah do zdrojového kódu Indexu. V tom by ale nebyl pro uživatele velký přínos. Jestliže by takovéto spojení přesto vyžadoval, má možnost toho docílit i bez předělávání Indexu. Uživatel může stránky přesunout pod společný uzel, který už Index vybrat umí.

Samotný požadavek na slučování v různých částech stromu spíše značí špatně uspořádanou strukturu (viz obr. 2.2).



Obrázek 2.2: Příklad uzlů stránek. Červeně jsou označené uzly, pro které vzájemné sloučení nedává smysl. Není jasné, v jakém pořadí tyto stránky sloučit, co udělat se synovskými uzly, ani pod kterého otce by měla sloučená stránka patřit.

Pro lepší představu: v uživatelské dokumentaci na obrázku vidíme obě struktury, jak je zobrazuje Zim (viz obr.3.7).

2.2.1 Hloubka slučování

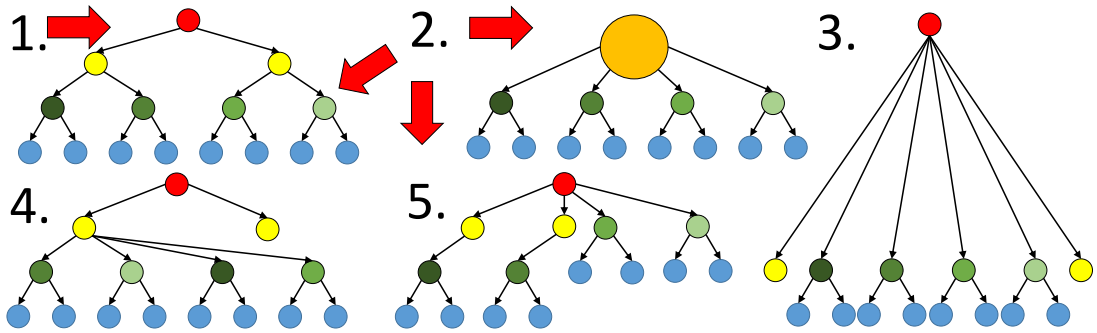
Vybrat umíme vhodně jen stránku a tím i celý její podstrom. Stránky, které chceme slučovat mezi sebou, mají mít vztah. Tím jsme problém mírně upravili. Vztah mají uzly ve společných větvích. A ty mají společný uzel. Nyní již víme jak vybrat místo odkud se bude slučovat, nevíme však jak hluboko, kolik úrovní.

Existují dva přístupy, máme na výběr: buď sloučit celý podstrom, anebo konkrétní počet úrovní podstromu a uzly v první úrovni mimo spojovanou oblast přiřadit uzlu, do kterého je slučujeme.

Pokud bychom chtěli slučovat konkrétní počet úrovní, narazíme na zásadní nedostatek. Jednak by se znesnadnilo používání programu, protože by se od uživatele vždy vyžadovalo číslo konkrétní úrovně. Pro správnou obsluhu by si uživatel musel zapamatovat, jak vypadají téměř všechny stránky ve slučované oblasti. Uživateli by navíc nezbývalo nic jiného, než že by musel ve všech různých větvích spojovat stránky jen na stejné úrovni podstromu. Tato varianta by tedy byla příliš složitá, ale stále by neumožňovala slučování přesně podle uživatelových představ. Dalším důvodem proti tomuto přístupu je, že při opětovném dělení nelze bez použití databáze jednoznačně určit, ke komu tito potomci náleží (viz obr. 2.3). Databáze by podle Zimu (viz 1.2), neměly obsahovat informace nezjistitelné z jiných zdrojů.

Rovněž by nešlo postupně slučovat stránky vždy jen o jednu úroveň, protože by se narušilo pořadí přidávaných synů už při slučování, takže by se synovské stránky slučovaly ve špatném pořadí.

Druhou možností je slučovat celý podstrom. Tento přístup nemá nedostatky předchozí možnosti, uživatel si nemusí pamatovat přesnou strukturu stránek a vše uvidí na jedné stránce. Strukturu může upravit pomocí nadpisů, které mají vyšší volnost ji měnit. Poté ji lze převést zpět na strukturu stránek. Navíc strom



Obrázek 2.3: Slučování o jednu úroveň. 1. Strom před spojením červeného uzlu a jeho žlutých synů, 2. Výsledek sloučení, 1.,3.,4.,5. zástupci možných výsledků dělení, protože se ve 2. ztratila informace o zapojení uzlů k jejich původním otcům.

při dělení nemá potomky, a i kdyby v průběhu práce se Zimem přibyly současnému uzlu noví potomci, budou jen jeho a problém s jejich přidělováním vůbec nenastane (viz obr. 2.3).

Pro další zefektivnění práce přidáme nástroj k rychlejší práci se strukturou nadpisů. Nástroj bude zobrazovat všechny nadpisy včetně úrovní, textu a pozice nadpisu na stránce bez jiného obsahu stánky, aby uživatel nemusel posouvat okno. Finální nástroj je vidět na obrázku: 3.4

2.2.2 Řešení

Tyto důvody vedly k rozhodnutí zvolit variantu, která slučuje celý podstrom.

To přineslo komplikaci, protože dává smysl sloučit všechny uzly do jednoho. Jelikož na kořenový uzel nelze přistoupit, přidáme možnost sloučit všechny potomky kořenového uzlu do nové stránky. Na její jméno se zeptáme uživatele. Celý Index se sloučí do jedné stránky.

Obdobu této operace v nadpisech realizujeme přesunem pod nový nadpis, který bude první na stránce.

2.3 Sjednocení hloubky struktur

Dalším problémem je sjednocení rozdílného počtu úrovní nadpisů s počtem úrovní Indexu. Pokud sloučíme moc hluboký podstrom, nelze reprezentovat původní strukturu.

2.3.1 Omezení počtu úrovní

Nejjednodušší možností, jak urovnat tuto nesrovnalost, je omezit počet úrovní Indexu na stejnou hodnotu jako mají nadpisy. Toho bychom ale měli uživatele ušetřit. Uživateli by to totiž při psaní neumožňovalo tvůrčí volnost. Téměř jistě každý pokročilý uživatel používá v Indexu více než pět úrovní. Samotný kalendář zobrazuje den na čtvrté úrovni. Pokud bychom se tedy chtěli dostat o pouhé dvě úrovně hlouběji, nebylo by nám to kvůli nadpisům umožněno.
Musíme tedy nalézt jiný způsob, jak nadpisům přidat libovolné množství úrovní.

2.3.2 Přidání úrovně do názvu nadpisu

Pro libovolně mnoho úrovní by jednou možností jak program upravit, bylo přidat nadpisům úrovně do jejich názvu, ať už by se jednalo jen o jednoúrovňové číslování nebo i vícenásobné, vhodné například při opakovaném slučování stránek. Ovšem nevýhodou by bylo, že by úrovně nadpisů při překročení páté úrovně neodpovídaly svému uživatelem předpokládanému vzhledu. To by ovšem odporovalo základní myšlence Zimu, kde mají mít tagy význam jak grafický, tak sémantický.

2.3.3 Přidání úrovně do odsazení nadpisu

Vhodným řešením by bylo nadpisy vyšší než páté úrovně odsazovat. Jelikož Zim sám o sobě nepodporuje odsazování nadpisů, nemusíme se obávat, že by toto naše případné vylepšení porušilo dokumenty vytvořené v dřívějších verzích. Rovněž se nemusíme obávat, že by se nadpisy (při běžném množství úrovní nadpisů) dostaly ven z viditelné oblasti.

Pro nějaké množství úrovní by bylo možné nadpisy odsazovat o menší a menší vzdálenosti, ale to je pouze dočasné řešení – nemůžeme odsazovat o méně než jeden pixel. Při běžné práci toto překážku nezpůsobí, naopak, odsazení zřetelně odliší úrovně, a pokud by se snad někdo dostal na vyšší úrovně, mohl by se odsazovat i text. Je ale velmi nepravděpodobné, že by někdo obvykle pracoval s více než třiceti – čtyřiceti úrovněmi.

2.3.4 Přidání více úrovní pro vyjádření nadpisů

Problém máme s tím, že Zim omezuje počet úrovní na konkrétní a navíc malé číslo. Pokud bychom toto číslo zvětšili, problém pouze odsuneme dál. Pro dostatečně vysoké číslo by nám to již možná nemuselo vadit. Jestliže bychom však zvolili počet úrovní příliš veliký, Zim by se nemusel ani spustit. Zim na nadpisy používá statické tagy, což jsou tagy, které se po spuštění programu všechny inicializují, a existují po celou dobu běhu programu.

Počítač na kterém bude Zim bežet na rozdíl od Tuningova stroje bude mít omezenou pamět a tu by mu neúměrné množství tagů spotřebovalo. Počáteční inicializace ovšem stojí i čas a čím více takových tagů budeme mít tím déle se program bude spouštět.

Shrňme nevýhody tohoto přístupu: zpomalí to start, zabere zbytečně moc paměti a navíc je pak uživatel z velké části ani nepoužije.

Přidání nových tagů Zim pluginům nemumožňuje. Pro přidání nových tagů bychom museli zasáhnout do zdrojového kódu programu, a tím se vzdát toho, že naše práce bude pouhým pluginem.

Dynamické vytváření tagů až když jsou opravdu potřeba by vyřešilo časovou a prostorovou náročnost a jediným novým úskalím tohoto přístupu je, že dotaz na ještě neexistující úroveň vyústí chybou. Bohužel Zim vytvářet tagy dynamicky neumí, a proto by se do zdrojového kódu muselo zasáhnout mnohem více než v předchozím případě. Bude nezbytná úprava syntaxe a s tím související změna v ukládání a načítání souborů i při formátování wiki během psaní v „on the fly“ režimu.

2.3.5 Řešení

Zde jsme se rozhodli pro kombinaci dvou řešení, odsazování pro lepší odlišení vzhledu i pro přidání nekonečného počtu úrovní nadpisů, jejichž použití bude naznačeno v následující kapitole.

2.4 Vzájemný převod nadpisu a stránky

Jak jsme se již zmínili, text sám o sobě může obsahovat pět úrovní nadpisů, odkazy a různé formátovaní. Oproti tomu nadpis žádné formátování neumožňuje, jedná se jen o řádek textu.

Musíme zobecnit pojem nadpisu, abychom za ním viděli i část stránky. Je třeba se jen rozhodnout, jak velká část stránky k němu bude připojena.

Jedna z možností, které se nabízejí, se dá jednoduše vyjádřit slovy: co nadpis, to stránka. Znamenalo by to sjednotit pojem stránky a pojem nadpisu tím, že omezíme počet nadpisů první úrovně na jedné stránce na jeden nadpis. Opět by se ale jednalo o zbytečné omezování uživatele.

Další variantou je rozdělit nadpisy na dva typy, prvním by byly nadpisy sloužící jako oddělovače stránek a na běžné nadpisy v textu. Nadpisy sloužící jako oddělovače textu budeme dále označovat jako „super-nadpisy“. Stránka bude super-nadpisy rozdělena na stránky tím způsobem, že vše (včetně běžných nadpisů) bude součástí předchozího super-nadpisu, až do dalšího super-nadpisu nebo konce stránky.

Toto řešení volíme proto, že není pro uživatele omezující a nenaruší již vytvořené stránky.

2.4.1 Řešení

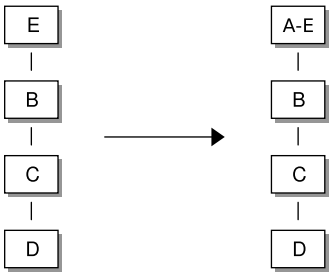
V předcházejícím rozboru jsme dospěli k závěru, že musíme přidat podporu pro libovonlně mnoho úrovní nadpisů. Díky změně, kterou jsme v tomto bodu přinesli, však nebudeme rozšiřovat původní nadpisy v textu, nýbrž nově přidané super--nadpisy. Nadpisy a super-nadpisy kvůli větší přehlednosti barevně odlišíme. Nové super-nadpisy se budou vzájemně velikostí, stylem a odsazením lišit podle toho, na jaké jsou úrovni. Stránku obsahující super-nadpisy budeme dále označovat jako „super-stránku“ čili stránku rozdělitelnou podle super-nadpisů na více stránek.

2.5 Zachování pořadí stránek v Indexu

Zatímco stránky v Indexu jsou řazeny abecedně, super-nadpisy mohou být na super-stránce řazeny v libovolném pořadí. Pokud bychom super-nadpisy dělením super-stránky převedli na stránky, zobrazily by se v Indexu v nesprávném pořadí. Stránky se v Indexu zobrazují podle názvu, jak je seřadí lehce upravená SQL funkce „ORDER BY“. Abychom zajistili správné pořadí stránek, musíme je nějak přejmenovat, aby byly správně seřazeny. Přejmenování lze provádět dvěma následujícími způsoby:

2.5.1 Minimální přejmenování

Možným pokročilým řešením je určit co nejmenší množinu nutnou k přejmenování. Například: mějme čtyři stránky pojmenované v pořadí: „E“,„B“,„C“,„D“ (viz obr. 2.4). Není potřeba přejmenovat všechny čtyři, ale pro zachování pořadí stačí přejmenovat pouze první stránku na „A-E“ a tím zachováme i název stránky, byť s prefixem. Pokud se na názvy stránek a jejich pořadí podíváme jako na permutace, lze poměrně jednoduše najít vhodnou funkci, pomocí níž bychom mohli tyto stránky přejmenovávat. V tomto případě by uživatel mohl nabýt dojmu, že se mu některé stránky zcela nesmyslně přejmenovaly, zvláště při více cyklech permutací. Navíc by se při opětovném dělení spojené super-stránky vyskytl nemalý problém, jak poznat, které stránky přejmenovat a které nikoli. Rozhodli jsme se proto toto řešení nepoužít a zvolit řešení vhodnější.



Obrázek 2.4: Skupina stránek, která by v Indexu změnila pořadí. Před a po provedení minimálního přejmenování.

2.5.2 Prostá prefixace

Druhá metoda přidá číselný prefix před všechny stránky. Toto přejmenování se odvíjí od počtu stránek a má výhodu v tom, že bude jednoduše detekovatelné při spojování na super-stránce, takže půjde snadněji odstranit a bude pro uživatele srozumitelnější. Další nezanedbatelnou výhodou je také menší výpočetní náročnost.

Tento přístup je kompatibilní i s již zmíněnou upravenou SQL „ORDER BY“ funkcí. Funkce sama o sobě funguje normálně, ale dostává upravený vstup. Ten je převeden na malá písmena a zároveň se autor programu pokusil vyřešit problém s přirozeným tříděním čísel tak, že krátká čísla zleva dorovnává nulami na celkových pět míst (čísla vyšší než deset tisíc se nemění). To znamená, že například název „21STráNKa2“ převede na „00021stránka00002“.

2.5.3 Řešení

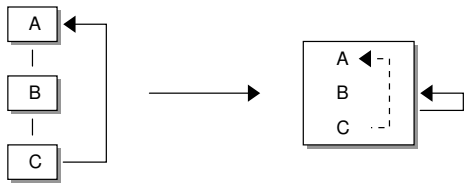
Zvolili jsme prefixaci, protože je intuitivnější.

2.6 Změna cíle odkazu

Zajímavým problémem se ukázaly odkazy na jiné stránky, které je potřeba měnit při slučování i štěpení stránek. Musíme vždy měnit odkazy vedoucí na stránky a velmi často odkazy vedoucí ze stránky.

2.6.1 Jak se odkázat na konkrétní místo ve stránce

Rozeberme si nejprve případ, kdy stránku spojujeme s jejím podstromem stránek. Spojíme-li několik stránek pod jeden uzel, může se stát, pokud existoval mezi původními stránkami odkaz, že se nově vytvořená super-stránka bude odkazovat sama na sebe (obr. 2.5).



Obrázek 2.5: Odkaz z C směřující na A, který po sloučení ukazuje na super-stránku složenou ze stránek A,B,C. Odkaz v rámci super-stránky na původní stránku A je ztracen.

Zim nepodporuje odkazy na konkrétní místo ve stránce, tak jako to umí například HTML (přes `<a href=\#kotva\>Návěští</a>`), ale pouze na stránku jako na celek. Musíme se proto rozhodnout, co uděláme s odkazy, které by po sloučení směřovaly do super-stránky. Máme několik následujících možností:

2.6.1.1 Zachování odkazů

První možností je odkazy po sloučení zachovat v původním stavu. Pokud ale odkazy po sloučení stránek vůbec nezměníme, budou nadále odkazovat na stránky, které jsme sloučením již zrušili. Odkazy tedy budou po sloučení směřovat na neexistující stránky. Ovšem Zim odkazy ukazující na neexistující stránku interpretuje tak, že tyto stránky vytváří, sice se na disk fyzicky nezapíší, ale chovají se tak – lze na ně přistupovat a Index je zobrazuje, takže by toto řešení narušovalo strukturu Indexu a ani odkazy by nefungovaly podle uživatelových představ.

2.6.1.2 Smazání odkazů

Smazání odkazů by vyřešilo výše zmíněný problém s narušením struktury Indexu. Uživatel by tím však zároveň trvale přišel o dané odkazy, a to i po případném budoucím obnovení jednotlivých stránek do původního rozděleného stavu (před sloučením do super-stránky).

2.6.1.3 Textové odkazy

Jiným řešením je přepsat odkaz tak, že bychom z funkčního odkazu udělali pouze textový odkaz. Na takový odkaz by se nedalo kliknout. Jednalo by se pouze o text nahrazující odkaz, který by uživatele informoval o existenci zrušeného odkazu, aby uživatel věděl, kam zrušený odkaz směřoval. Toto řešení je konzistentní s chováním Zimu, když stránka na kterou odkaz ukazuje zanikne. Zde však stránka nepřestává existovat, ale pouze se stává součástí jiné stránky.

2.6.1.4 Uložení odkazů do databáze

Dalším možným řešením je uložit odkazy do databáze (a jejich originál smazat) a v případě, že se stránka bude někdy dodatečně štěpit, je z databáze opět obnovit. Databáze však nejsou bezstavové a filozofií Zimu je, aby všechen obsah šel obnovit z textových souborů i v případě smazání databází. Bez ohledu na filozofii Zimu je zde další nepříjemnost a sice, že na dobu spojení stránek ztratíme původní cíl odkazů. Tato nepříjemnost přetrvá i tehdy, když originál odkazu nesmažeme (což by znamenalo, že by odkazy úplně zmizely a že bychom po celou dobu sloučení stránek vůbec nepoznali, že zde kdy nějaký odkaz existoval), ale pouze převedeme tak, aby ukazoval alespoň na novou super-stránku. Mimoto předpokládáme, že se obsah stránky nezmění a tak při opětovném dělení správně poznáme původní zdroj i cíl odkazů. Pokud bychom tento předpoklad neuvažovali, pak bychom museli každou změnu na stránce i v Indexu hlídat a podle toho upravovat záznamy v databázi

2.6.1.5 Odkazy na nadpisy

Lepší možností je zasáhnout do již existujícího kódu, nacházejícího se mimo zásuvný modul, a upravit ho tak, aby začal podporovat odkazy dovnitř stránek na nadpisy. Umožnilo by to odkazovat se na nadpisy jak v rámci jednoho dokumentu, tak napříč různými dokumenty.

Toto řešení by ale vyžadovalo zavedení nové syntaxe a upravení stávajícího kódu, starajícího se o otevírání, parsování a ukládání stránek. Odkazování na nadpisy by nám tak nejen umožnilo převádět odkazy, ale navíc by rozšířilo funkcionalitu programu. Uživatel by tak získal možnost se odkazovat dovnitř libovolné stránky.

2.6.1.6 Řešení

Na základě tohoto výčtu se jeví jako nejrozumnější a navíc uživatelsky nejpřívětivější možnost pro spojování stránek právě poslední z navrhovaných řešení. Právě možnost odkazovat se přímo na nadpisy uvnitř stránek byla mnoha lidmi žádána na oficiální webové stránce Zimu.

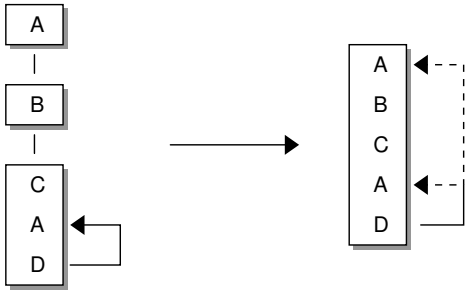
2.6.2 Slučování stránek do super-stránek

Již jsme vyřešili, v jaké formě uchovávat nadpisy na stránce. Nyní musíme vyřešit jak je do této podoby převést.

Převod všech odkazů nebude stejný, rozdělíme je do skupin podle toho, na které stránce se nacházejí a na kterou stránku směřují. S odkazy, které ukazují mimo slučovanou oblast, nic dělat nemusíme, protože jejich adresa zůstává stále stejná. Oproti tomu odkazy směřující do podstromu slučované stránky změnit musíme, a to na nadpis reprezentující původní stránku se směrovací adresou super-stránky, do které se slučuje. U odkazů vedoucích na původní stránku (po sloučení do super-stránky) máme volnost, zde se můžeme rozhodnout, do jaké skupiny je zařadíme, výsledek bude v obou případech korektní.

Při slučování je třeba si dát pouze pozor na to, aby se do vznikající super-stránky nepřidal stejný nadpis, který již na stránce existuje (obr.2.6). Jinak

by došlo ke kolizi v nadpisech a případný odkaz by tak mohl směřovat na jiný nadpis, než který byl původně zamýšlen. Řešit to lze přejmenováním. Tím však způsobíme, že bude potřeba opravit odkazy vedoucí na přejmenovávaný nadpis. Jako schůdnější možnost se jeví pouze na tento problém uživatele upozornit. A nechat ho rozhodnout zda chce odkazy přejmenovat.



Obrázek 2.6: Odkaz z D směřující na A před a po sloučení

2.6.3 Štěpení super-stránek na stránky

V této části již také počítáme s tím, že umíme odkazovat na nadpisy.

Rozdělme odkazy podle jejich umístění a směřování:

- Odkazy mimo štěpenou stránku
  - Odkazující mimo štěpené stránky
  - Odkazující dovnitř štěpené stránky
- Odkaz uvnitř štěpené stránky
  - Odkazující mimo štěpenou stránku 2.6.3.2
    - Absoulutní
    - Relativní
  - Odkazující dovnitř štěpené stránky
    - Do části, ve které zůstane
    - Do jiné části, než ve které skončí

2.6.3.1 Odkaz mimo štěpenou super-stránku

V případě, že nějaký odkaz ukazuje na super-nadpis uvnitř super-stránky, kterou štěpíme, pro správnou změnu odkazu stačí změnit lokaci na nově vytvořenou rozštěpenou stránku. Odkaz ukazující na samotnou super-stránku nebo mimo ní, se nemění.

2.6.3.2 Odkaz uvnitř štěpené super-stránky

Mějme stránku s množstvím spojených stránek a odkazů, kterou chceme dělit. Odkazy můžeme rozdělit podle toho, jestli ukazují mimo náš nově vznikající podstrom nebo dovnitř něho.

Když odkazy ukazují mimo štěpenou stránku, není potřeba, pokud jsou absolutní (tzn. začínají v kořenovém uzlu (syntaxe odkazu začíná znakem: „:“)), abychom s nimi cokoli dále prováděli. Pokud nejsou absolutní (a jsou tedy relativní) může vyvstat další problém, což bude vysvětleno podrobněji v následujících odstavcích.

Relativní odkazy
Relativní odkazy nezačínají v kořenovém uzlu. Zim má dva typy relativních odkazů, jednodušší odkaz, který ukazuje do potomků uzlu ve kterém se nachází odkaz a začíná znakem: „+“. A složitější kterému musíme dohledat počáteční uzel.

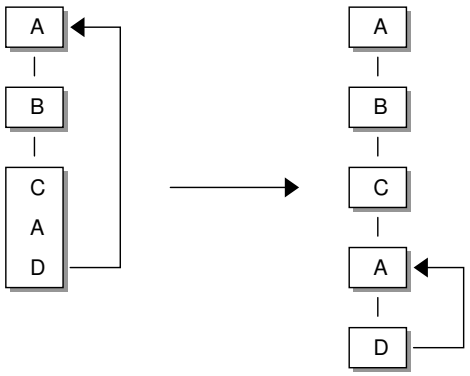
Vede-li odkaz ze štěpené super-stránky ven a je-li relativní, musíme ověřit, že jej nenarušíme. Náš úkol spočívá v nalezení tohoto uzlu, který musíme hledat stejným vyhledávacím algoritmem, který používá samotný Zim. Tento typ odkazu se jménem odkazuje na název uzlu v cestě mezi sebou a kořenovým uzlem nebo na název uzlu, který je bratrem některého z uzlů na této cestě. To znamená, že takto určený uzel musíme sami nejprve dohledat, a od něj začíná již absolutní cesta k cíli. Když už máme správný uzel tak musíme zajistit, že mezi něj a štěpenou stránku se nedostane jiný uzel se stejným jménem.

Vyhledávání uzlu
Algoritmus vyhledávání odkazovaného uzlu v Zimu vrátí vždy nějaký uzel. Buď odkaz na bratra současného uzlu, pokud se nikde v hledaných oblastech uzel nenašel, nebo vrátí první správně se jmenující uzel, který algoritmus našel. Je zřejmé, že je důležité zachovat postup hledání odkazu stejný, jinak bychom při shodě jmen došli ke špatnému uzlu. Vyhledávání začíná od uzlu s relativním odkazem (včetně) a jde až ke kořenovému uzlu (mimo). Poté hledání pokračuje v bratrech uzlů na cestě v původním (vzestupném) pořadí.

Vyhledávání uzlu – komplikace
Při štěpení by se mohlo přihodit, když by se cestou k odkazovanému uzlu objevil stejně pojmenovaný uzel, že by vyhledávací algoritmus označil za výchozí uzel jiný než ten původní. K tomu by došlo, pokud by dělená stránka obsahovala novou stránku se stejným jménem jako stránka, na kterou z potomka vede relativní odkaz. Například při situaci na obr. 2.7 dělíme část Indexu se třemi stránkami A-B-C, s tím, že stránka C obsahuje dva super-nadpisy A a D (C je „super--stránka“ obsahující stránky C, A a D). Při dělení C by se bez kontroly odkazů stalo to, že ukazatel z podstránky D by začal ukazovat jinam než je zamýšleno.

Řešením může být uživatele upozornit na kolizi jmen nebo některou stránek přejmenovat.

Protože se jedná o velmi okrajový případ, řešením bude varovné okno upozorňující na všechny nastalé kolize s dotazem, zda se má či nemá pokračovat. V případě, že se uživatel rozhodne pokračovat všechny odkazy s nejasným cílem budou nahrazeny všemi možnými variantami.



Obrázek 2.7: Problém relativních odkazů vyskytujících se na dělené stránce. Odkaz z D směřující na stránku A před rozštěpením a po rozštěpení.

2.6.3.3 Odkaz v rámci super-stránky

Zjištění cíle odkazu
Odkazy v nově přidané syntaxi, kompatibilní s dohodnutým formátem [4], ukazují pouze na nadpis bez toho, aby v něm byla informace o tom, v jakém podstromu se nalézá.

Což znamená, že pokud chceme vědět, kam do podstromu stránka patří, abychom mohli vytvořit odkaz, musíme najít nadpis, na který odkazuje, zjistit do jaké stránky se dostane, jak bude ta stránka finálně pojmenovaná a jak budou pojmenovány všechny předchozí stránky v cestě ve stromu od uzlu s nadpisem až po uzel, který štěpíme.

Může se stát, že se na stránce před dělením ocitnou stejně pojmenované nadpisy.

Tento případ nastane:

- Pokud slučujeme stránky, které mají nezávisle na sobě stejný nadpis. Tento případ popisujeme již výše.
- Když si je uživatel sám vytvoří na jedné stránce. Tomu zabránit nemůžeme, a tak opět stojíme před problémem, jak rozlišit, který nadpis patří příslušnému odkazu. V tomto případě opět uživatele upozorníme a necháme ho vybrat, jak chce dále pokračovat.

2.7 Okrajové problémy

Další otázky, na které bylo třeba odpovědět, byly maličkosti.

Jak uložit název stránky, pokud se liší od názvu nadpisu.

Do teď jsme nepředpokládali, že by se takovýto problém mohl vyskytnout. Jediným rozumným řešením se ukázalo použít druhý nadpis, který by následoval přímo za super-nadpisem. Při dělení se nadpis smaže a použije se na jméno stránky.

Pokud slučujeme stránku a víme, že ji budeme opět dělit a má přílohy, dostáváme se do situace, kdy nevíme co s nimi udělat, pokud bychom je přesunuli

a nic jiného s nimi nedělali, pak při dělení zůstanou v super-stránce a stránka ztratí přílohy. Pokud se ovšem stránka na přílohu ze svého textu odkazuje, pak lze text analyzovat a soubor přesunout zpět spolu se stránkou. Ovšem na soubor se stránka nemusí vůbec odkazovat a nebo se na něj nemusí odkazovat jediná. V případě, že se neodkazuje sama, je jedno, které stránce připadne. Pouze musí zůstat funkční odkaz ze druhé stránky.

A v případě, že se neodkazuje vůbec obdobně jako v předchozím řešení, nezbude než informaci o tom, že stránka vlastní přílohu, připsat do stránky a při dělení ji odebrat.

Slučujeme-li více stránek dohromady a ty obsahují stejně pojmenovaný soubor, je potřeba zajistit, aby se soubory nepřepsaly, ale pouze přejmenovaly, abychom nepřišli o žádnou informaci.

Zim obsahuje zpětné odkazy (backlinks), jsou to odkazy, které se automaticky vytváří v opačném směru než uživatelské odkazy. Stránka má možnost zjistit, kdo na ni odkazuje. Při každé operaci sloučení nebo dělení musíme opravit odkazy i na stránkách, kterých se tyto operace jinak netýkají.

Ačkoli se tyto věci jevily jako maličkosti, daly nakonec nejvíce práce.

3. Uživatelská dokumentace

V této kapitole seznámíme uživatele s programem a detailně popíšeme jeho ovládání. Nejprve provedeme uživatele jednoduchou instalací.

3.1 Instalace, spuštění, ovládání

3.1.1 Závislosti

Pro správný běh programu je potřeba splnit následující systémové závislosti:

- **Operační systém:**
  - Linux
  - Windows ve verzi XP nebo vyšší
  - Mac OS X
- **Programové vybavení:**
  - gtk2
  - Python ve verzi 2.7 a vyšší, ale nižší než 3.
  - python-gtk
- **Konkrétní závislosti dle platforem:**
  - **Závislosti pro Linux**:
    - gtk+ >= 2.6
    - python >= 2.7
    - python-gtk
    - python-gobject
  - **Závislosti pro deriváty Debianu (např. Ubuntu)**:
    - python >= 2.7
    - libgtk2.0-0
    - python-gtk2
  - **Závislosti pro Windows**:
    - python >= 2.7
    - pygtk-all-in-one-2.24.2.win32-py2.7.msi nebo novější
  - **Závislosti pro Mac OS X**:
    - python27
    - py27-gtk
    - py27-simplejson
    - py27-xdg

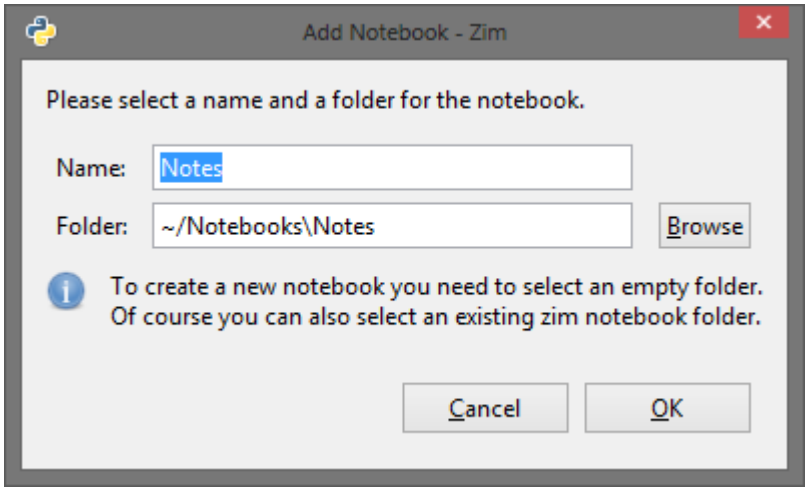
3.1.2 Instalace a spuštění

Samotný program Zim vůbec nevyžaduje instalaci, stačí pouze rozbalit soubor „`zim.zip`“ do nové složky na lokálním disku. V případě, že jsou dobře nakonfigurované systémové cesty, se aplikace spustí otevřením souboru „`zim.py`“ v kořenovém adresáři rozbalených souborů. V opačném případě, pokud je Python ve standardním umístění a v příkazovém řádku je otevřena složka s rozbalenými soubory, lze soubor otevřít z příkazové řádky příkazem „`C:\Python27\Python.exe zim.py`“.

Na Linuxu a Mac OS X se spouští obdobně příkazem „`python ./zim`“

3.1.3 Ovládání

3.1.3.1 První spuštění – složka pro Zim



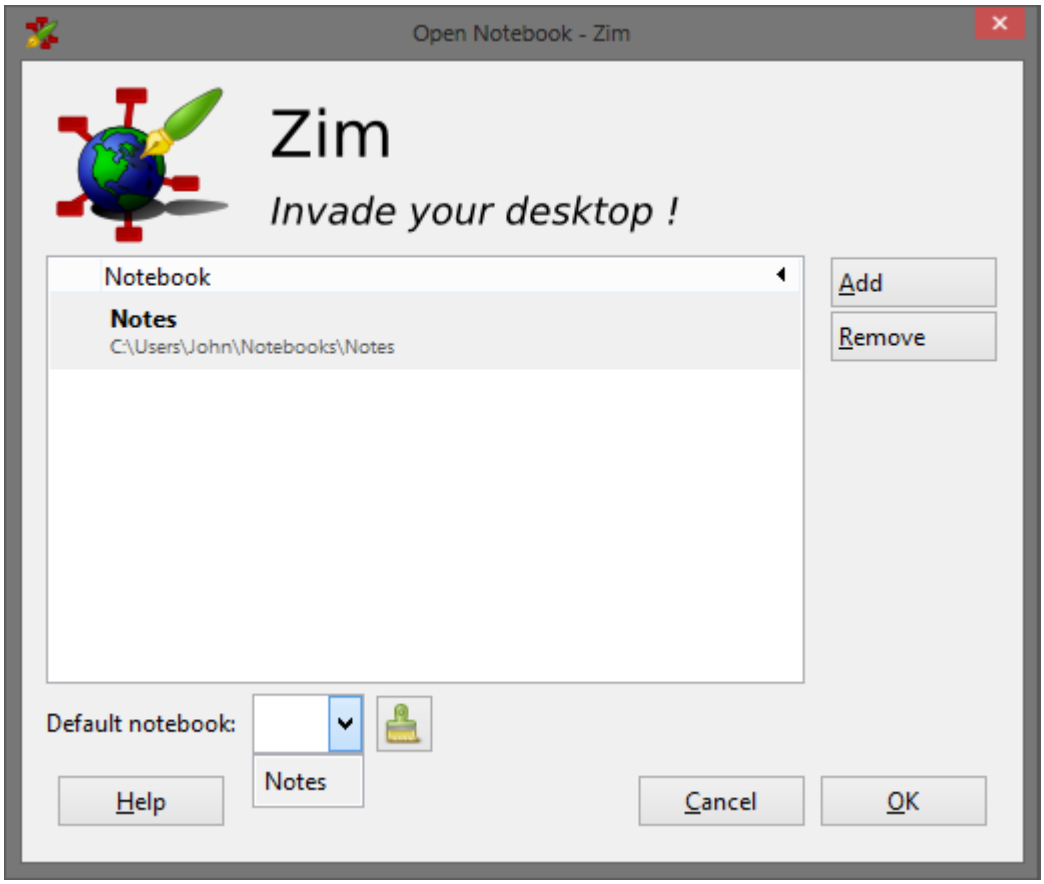
Obrázek 3.1: Okno při prvním spuštění programu.

Po spuštění programu se objeví okno s nabídkou pro výběr složky na disku (viz obr. 3.1). Tu si pak Zim bude sám spravovat. Do této složky se bude ukládat obsah jedné kolekce (sešitu). Pro každou další kolekci zvolíme novou složku přes nástrojovou lištu v sekci File (Soubor) –> Open another notebook (Otevřít jiný sešit). To nám otevře podobné okno jako na začátku, v němž můžeme sešity spravovat (viz obr. 3.2).

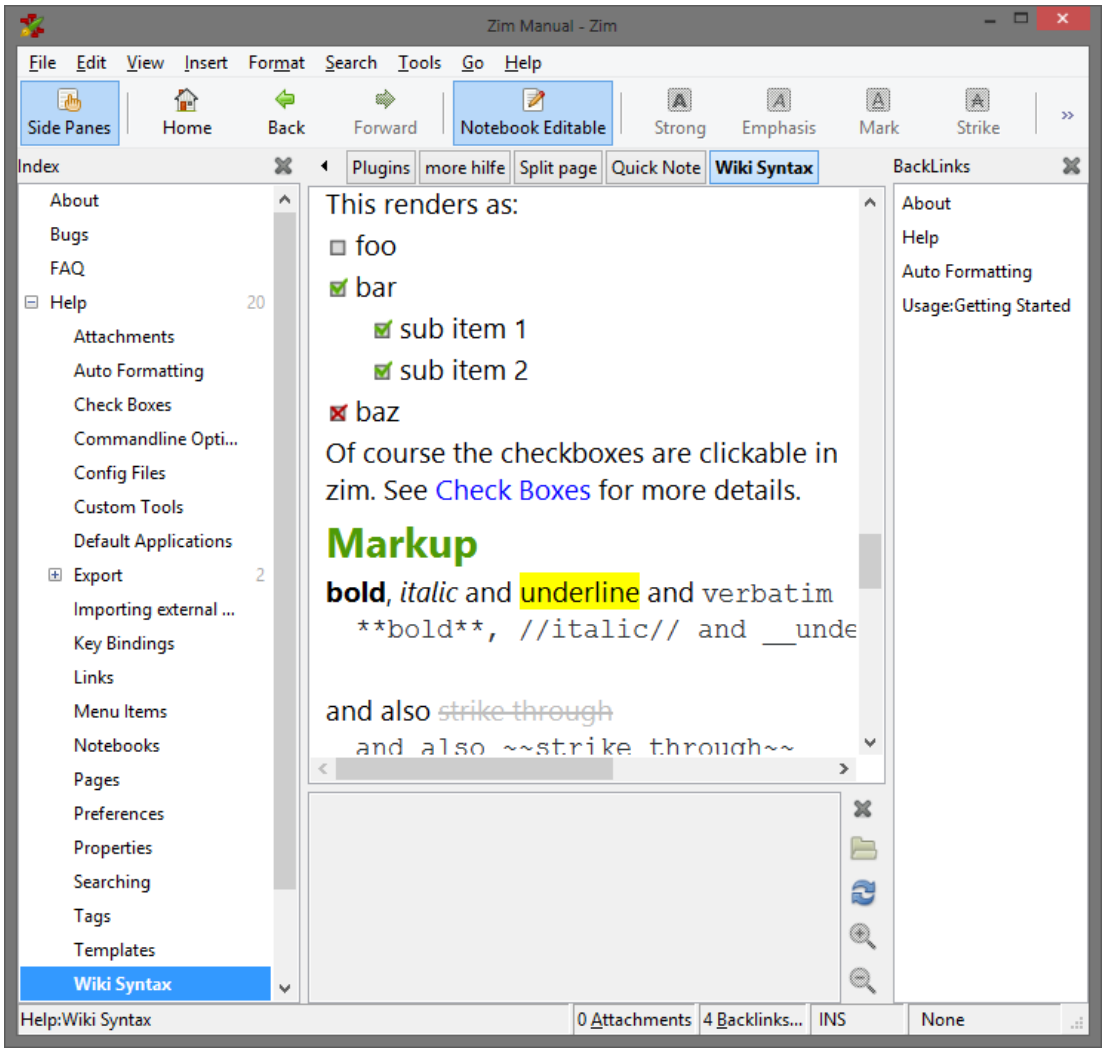
Po vybrání sešitu se otevře hlavní okno programu, které se skládá z nástrojové lišty, z lišty s tlačítky, z lišty s historií navštívených stránek a z několika postranních panelů (viz obr. 3.3). V levém panelu se zobrazuje strom stránek, v největším panelu se nachází obsah stránky, další panely zobrazují přílohy a zpětné odkazy. Panely se dají schovat a zobrazit klávesami F9 a „Ctrl-F9“.

3.1.4 Zapnutí pluginu

Pro zapnutí pluginu je zapotřebí v Edit (Úpravy) –> Preferences (Nastavení) –> Plugins –> Split pages zaškrtnout políčko a potvrdit tlačítkem OK. Tím se plugin aktivuje a na hlavní liště by měla přibýt nová tlačítka. Program samotný není plně počeštěn, i proto byla za jazyk pluginu zvolena angličtina.



Obrázek 3.2: Okno pro výběr kolekce.



Obrázek 3.3: Hlavní okno programu se zobrazenou nápovědou a ukázka možností syntaxe.

3.2 Změny oproti Zimu

V této části se pojednává o změnách, které v programu Zim nastaly díky novému rozšíření programu, prezentovaného v této bakalářské práci. Nejdříve uvedeme seznam nových uživatelských prvků, a poté představíme ovládání pluginu.

3.2.1 Nové nadpisy

Do Zimu jsme přidali nový typ nadpisu. Tento nadpis slouží i jako oddělovač stránek, pokud bychom je chtěli dělit. Budeme mu říkat super-nadpis. Přidali jsme rovněž nový způsob, jakým zapsat původní nadpisy. Pro zápis jednoho takového (původního) nadpisu slouží dvě rovnítka, číslo určující jeho úroveň, mezera a samotný text nadpisu (`==1 H1` a `==5 H5`). Super-nadpis zapíšeme podobně jako v předchozím případě, ale samotnou úroveň pro odlišení s normálním nadpisem v textu prefixujeme dalším rovnítkem nebo jedničkou (`==11 super-nadpis první úrovně` nebo (`===1 super-nadpis první úrovně`). Super-nadpis má jinou barvu, než mají standardní nadpisy, navíc je pro snazší odlišení vyšších úrovní odsazen a toto odsazení je fixní. Normální nadpisy mají zelenou barvu, zatímco super-nadpisy modrou.

3.2.2 Odkazy na nadpisy

Zajímavou vlastností Zimu je koncept odkazování. Je podobný tomu, jak jej známe z internetových stránek. Zatímco v nerozšířeném Zimu jsme se nemohli odkázat dovnitř nějaké stránky, ať už byla jakkoli dlouhá, tak v naší rozšířené verzi se již lze odkázat na konkrétní nadpis. Toto rozšíření se využívá tehdy, když spojujeme na sebe navzájem odkazující stránky a nechceme přijít o dané odkazy. Tímto způsobem se odkážeme na nadpis stránky představující původní, nespojenou stránku.

Tímto novým odkazem se lze odkázat na jakoukoli stránku a na jakýkoli nadpis, tedy nejenom na super-nadpis. Vytvořit takovýto nadpis v textu lze velmi jednoduše. Od normálního odkazu se totiž liší až v poslední části adresy.

Odkaz na stránku se zapisuje tak, že mezi názvy stránek od společného uzlu až k cíli našeho odkazu vložíme dvojtečky. Za takovouto adresou stránky, která může být prázdná, napíšeme křížek (#) a opíšeme nadpis. Pokud by nadpis nebo stránky obsahovaly mezery, musíme je v odkazu buď nahradit podtržítky, nebo odkaz zapsat jiným způsobem.

Jiným způsobem lze nadpis napsat pomocí wiki syntaxe, a to tak, že napíšeme dvakrát dvojici hranatých závorek a dovnitř umístíme odkaz. Tedy odkaz na stránku, která se nachází pod kořenovým uzlem a jmenuje se MFF s nadpisem Malostranské náměstí lze napsat jako: „"[[:MFF#Malostranské náměstí]]. Nejjednodušeji se odkaz na jinou stránku vytvoří tím způsobem, že se z Indexu(Obsahu) přetáhne stránka dovnitř textu. Tento odkaz můžeme pozměnit buď přes nabídku myši – edit link, nebo se můžeme textovým kurzorem postavit před poslední písmeno textu, připsat co potřebujeme a nepotřebný znak smazat.

3.3 Popis uživatelského rozhraní

V této části podrobněji popíšeme nové ovládací prvky, které přidáváme do uživatelského rozhraní. Nejprve budou vyjmenovány a popsány a v následujících sekcích je postupně detailněji představíme.

Naše rozšíření přidává do uživatelského rozhraní tyto nové ovládací prvky:

- novou klávesovou zkratku:
  - Shift + pravá šipka
- nové položky v nastavení:
  - Automatically turn "#Tag"into links to current page.
    (Automaticky převádět „#Tag“ na odkazy do současné stránky.)
  - Besides standard types e.g. ":Page"and "+Subpage:"turn automatically into links also:
    (Kromě standardních typů („:Stránky“ a „+Podstránky:“) se mají převádět na odkazy také:)
- nová tlačítka do hlavní lišty:
  - Join pages (Spoj stránky)
  - Split pages (Rozděl stránky)
- nové tlačítko do položky Edit (Úpravy) v nástrojové liště:
  - Change headings (Uprav nadpisy)
- novou položku do vyskakovacího okna, které se objeví při stisknutí pravého tlačítka myši v okně, kde se zobrazuje Index:
  - Join pages (Spoj stránky)

Mimo tyto prvky obsahuje plugin nastavení v sekci Configure v nabídce, kde jsme aktivovli plugin (v Edit –> Preferences –> Plugins –> Split pages)

V nastavení pluginu jsou položky zobrazeny tím způsobem, jakým jsou shrnuty v tabulce 3.1

3.3.1 Nová klávesová zkratka

Do programu jsme přidali novou klávesovou zkratku „shift + šipka doprava“, umožňující rozbalit podstrom stránek Indexu.

Separator between page content and list of attachements created during join.
Oddělovač obsahu stránky a seznamu příloh vytvořených během slučování.
Delimiter in list, between attachmenets created during join „\n“ or „\t“ are accepted
Oddělovač položek v seznamu příloh vytvořeném během slučování. Lze použít „\n“ i „\t“.
Experimental support for attachment browser reload after split/join
Experimentální podpora pro znovunačtení okna pro prohlížeč příloh.
In potentionaly dangerous operation dialogs pre-select continue
Při potencionálně nebezpečných operacích v dialogu předvybrat tlačítko pokračovat.
Ambiguous links delimiter
Oddělovač pro víceznačné odkazy.

Tabulka 3.1: **Nastavení pluginu.** Položky nastavení pluginu s překladem.

3.3.2 Položky v nastavení

Obě položky nastavují automatické převody textu na odkazy.

Nastavení u #Tag je zatrhávací tlačítko, které určuje, zda se text má či nemá převést, pokud je zapsán tak, že první písmeno slova je křížek (#).

Druhé nastavení určuje, jaké texty se mají převést na odkazy, pokud neporušují pravidla stránek, a tak by mohly být odkazem. Jedná se o nabídku možností, kde se může vybrat právě jedna položka. V odpověď na otázku „Kromě standardních typů („:Stránky“ a „+Podstránky:“) se mají převádět na odkazy také:“ jsou na výběr možnosti „Nic dalšího, „Stránku:“, „+Stránku“, „Stránku:“ i „+Stránku“, Nepřevádět vůbec nic“, anglicky (Nothing else, ”Page:”, ”+Page”, ”Page:”and ”+page”, All words, Do not turn anything).

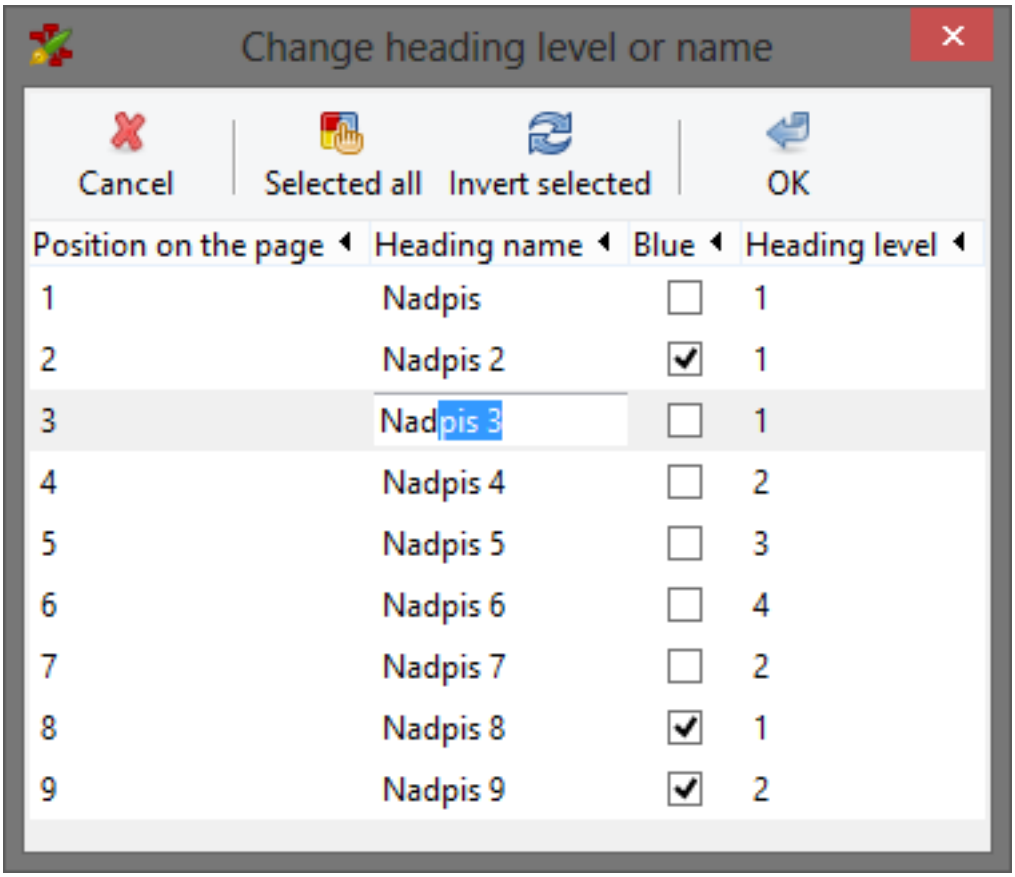
Tímto nastavením řešíme více požadavků, které byly vzneseny na stránce věnované vývoji Zimu.

3.3.3 Tlačítka na hlavní liště

Na hlavní liště se nacházejí dvě nová tlačítka: Join pages (Spoj stránky) a Split pages (Rozděl stránky). Pokud uživatel klikne na tlačítko Join pages, tak se stránky, které se nacházejí v podstromě současné stránky spolu s ní sloučí do jediné stránky. Naopak když uživatel klikne na tlačítko Split pages, rozdělí se stránka podle nadpisů na několik menších.

3.3.4 Nástroj na úpravu nadpisů

Pro úpravu více nadpisů najednou jsme do Zimu přidali nástroj, pojmenovaný Change headings. S tímto nástrojem lze měnit typy nadpisů mezi normálními a super-nadpisy. Měnit lze úroveň nadpisů i jejich text. Všechny nadpisy, i pro dlouhé stránky, přehledně zobrazí v jednom okně (viz obr. 3.4).



Obrázek 3.4: Okno pro změnu typu nadpisů, jejich úrovně i názvu, s vybranou buňkou pro editaci názvu.

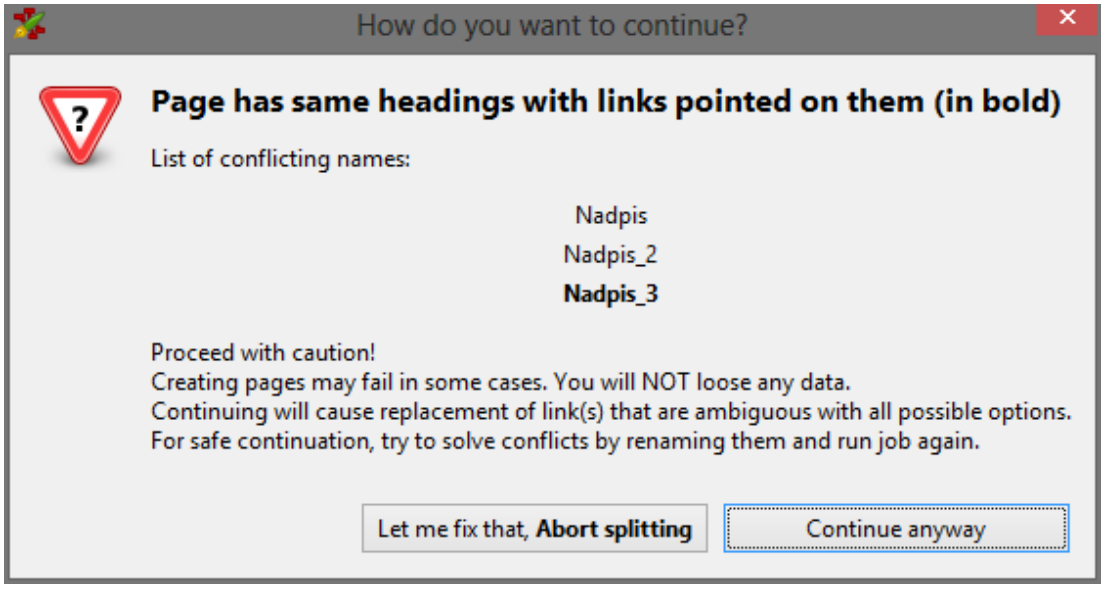
3.3.5 Položka v Indexu

Ovládací prvek Join pages se od stejnojmenného tlačítka na liště mírně liší. Má sice stejnou funkci, ale navíc lze kliknutím na nevybranou stránku v Indexu sloučit stránku, na které se nenacházíme. Největším přínosem tohoto prvku je to, že umožňuje spojit všechny stránky dohromady. Stránky se spojí do nové stránky, její název se zapíše do dialogového okna, které se objeví, když v Indexu klikneme na část, na níž se nenacházejí žádné stránky, a klikneme na Join pages. Jinými slovy: stránky se spojí do nového uzlu, na jehož jméno se zeptá dialogové okno.

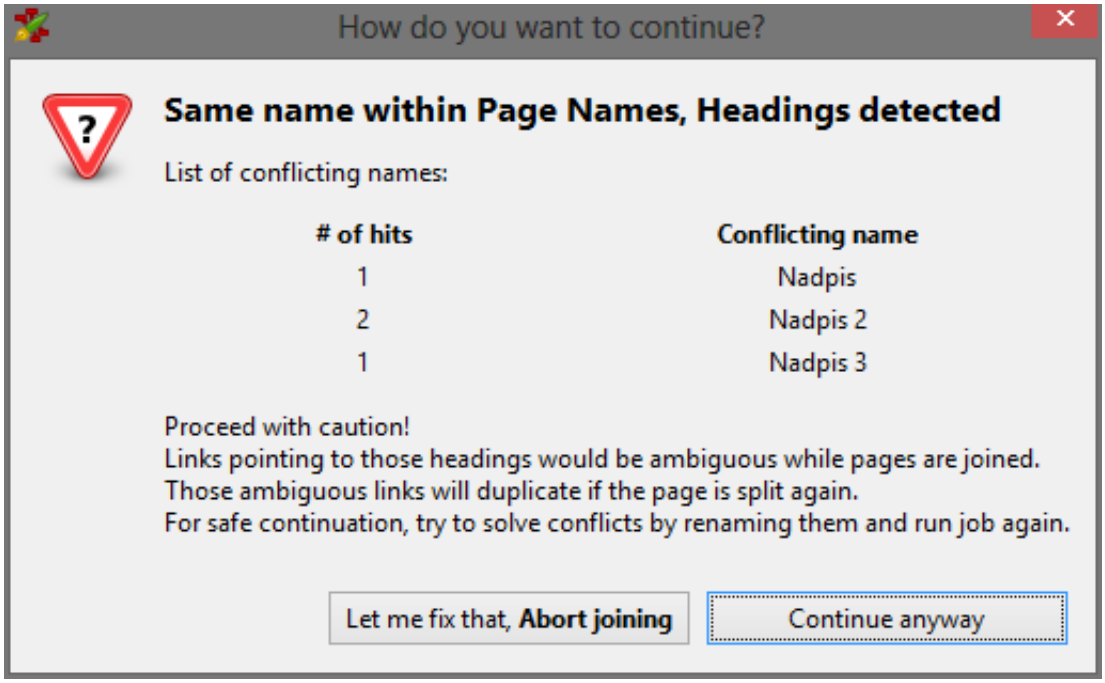
3.4 Ovládání, příklady použití

Začneme operací, kvůli které se bude toto rozšíření používat asi nejčastější. Tou operací je dělení stránek. Text přibývá, až přeroste pro uživatele únosnou mez. Uživatel proto bude chtít stránku dělit. Naše ukázková stránka bude obsahovat pouze nadpis první úrovně, text nadpis první úrovně další text (na cd je to soubor 1_uvod.txt). Pokud nyní klikneme na tlačítko Split page, otevře se nám dialogové okno s dotazem ‚jestli opravdu chceme stránku dělit, protože se na ní nenalézají modré nadpisy, ale přesto se na stránce nalézá dostatek prvků, aby šla rozdělit. Zvolíme možnost dělit podle všech nadpisů a stránka až do místa, kde se předtím nalézal druhý nadpis, zůstane nezměněna a obsah druhé části zmizí. Přesune se do nově vytvořené stránky. Název této stránky bude korespondovat s textem nadpisu a obsah bude rovný obsahu původní stránky od druhého nadpisu včetně do konce stránky.

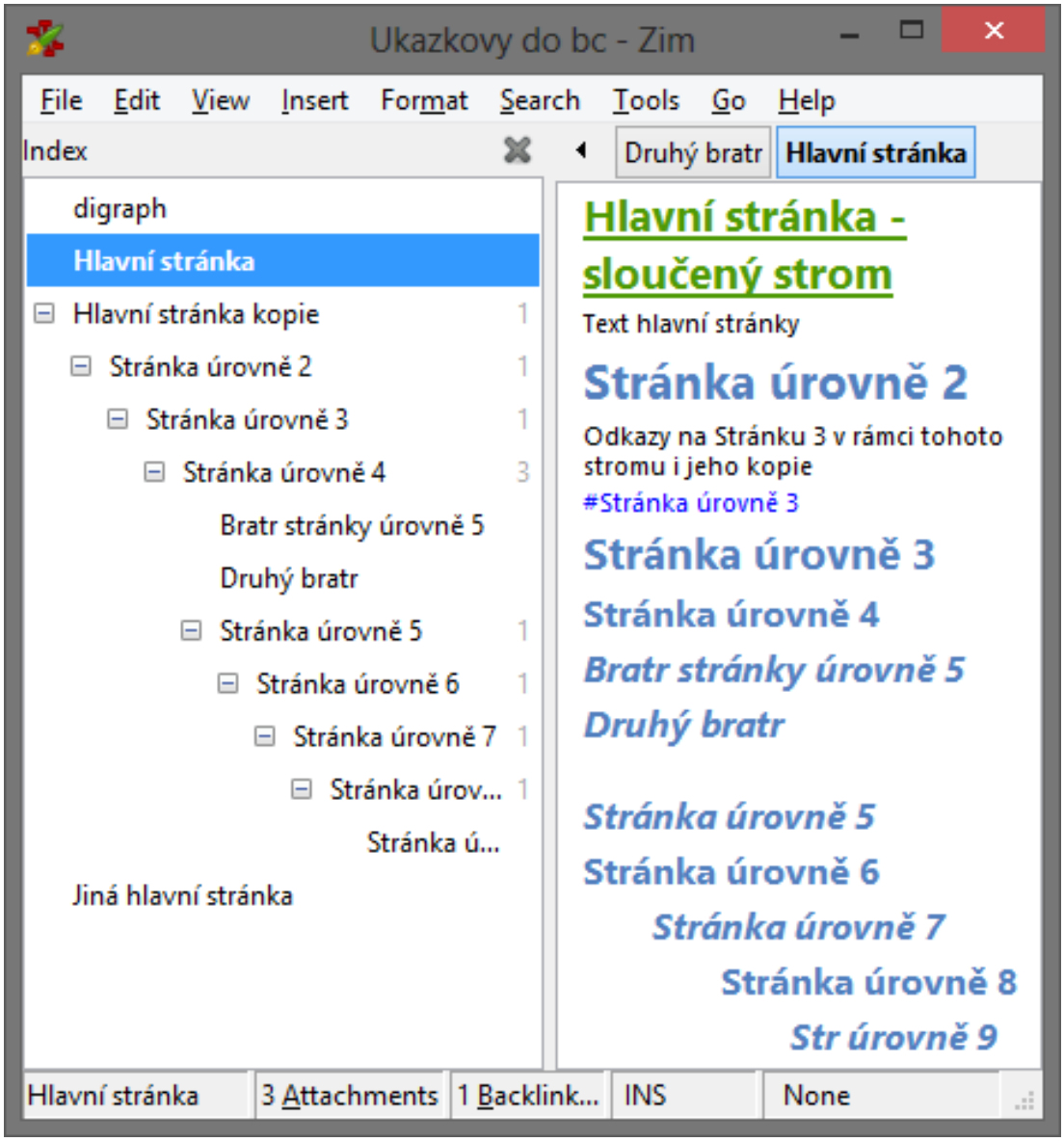
Druhá ukázka se soubory (2_Ukázka_1.txt a 2_Ukázka_2.txt) pomůže přiblížit rozdíl mezi možností dělit podle nadpisů první úrovně a dělení podle všech nadpisů. Soubory jsou totožné, obsahují za sebou jdoucí nadpisy první až páté úrovně a znovu nadpis první až páté úrovně. Mezi některými je vložen text pro lepší orientaci. Pokud nyní opět klikneme na tlačítko Split pages a pro oba soubory zvolíme různou možnost, tak prvním případě přibude jeden soubor, ve druhém případě



Obrázek 3.5: Chybové okno při dělení.



Obrázek 3.6: Chybové okno při slučování.



Obrázek 3.7: Vlevo struktura stránek před sloučením a vpravo struktura nadpisů po sloučení.

4. Programátorská dokumentace

Tato bakalářská práce staví na základech již existujícího programu napsaného v jazyce Python. Znamená to, že jsme se museli v mnoha ohledech držet předem stanovených způsobů. Nemohli jsme se rozhodovat bez přihlédnutí ke kontextu zbylé aplikace. Bylo naší snahou do původního kódu aplikace zasahovat v co nejméně případech, avšak ne za každou cenu. Mnohdy bylo lepší upravit některé drobnosti v původním programu, než jej složitě napodobovat v rámci pluginu. Ve výsledku jsme dospěli k tomu, že zásahy jsou nezbytné, a opustili jsme izolaci pluginu.

Seznam upravených souborů s jejich umístěním.

- zim/plugins/splitpage.py
- zim/formats/__init__.py
- zim/formats/wiki.py
- zim/gui/__init__.py
- zim/gui/pageview.py
- zim/gui/widgets.py
- zim/gui/config.py
- zim/notebook.py
- zim/parsing.py

Soubor splitpage.py je hlavním souborem naší práce. Jedná se o část, která pro své specifické funkce zůstala oddělena od hlavního programu. Neobsahuje všechny přidané funkce. Některé najdou uplatnění i v jiných částech aplikace.

4.1 Úprava metody z text to tree

Při vytváření nové stránky bylo potřeba volat funkci, která náš strom reprezentující stránku uloží. Tato metoda však jako vstup vyžaduje textový řetězec, proto by bylo nutné převádět pomocí třídy Dumper strom na text. Při podrobnější analýze fungování programu jsme však zjistili, že daná metoda by text v našem případě pouze pomocí třídy Parser opět převedla na strom a se samotným textem by již dále nepracovala. Tato funkce se nachází v souboru zim/gui/__init__.py, její název je new_page_from_text. V zájmu zrychlení programu jsme vytvořili identickou metodu a uložili do stejného místa, pojmenovali jsme ji new_page_from_tree. Tato metoda plně zastupuje části původní metody pracující se stromem. Poté jsme původní metodu upravili, aby po převodu textu na strom volala novou metodu from tree a zbylou část jsme smazali, abychom zajistili lepší spravovatelnost kódu. Pokud původní metoda neobdrží při volání název stránky nebo pokud je příslušná proměnná vyhodnocena jako prázdná, tak se z textu maximálně prvních čtyřicet znaků z textu uloží jako název, převede text a zavolá novou metodu, která již pracuje výhradně se stromem.

4.2 Popis tříd

V této části se budeme věnovat tomu, co která třída dělá.

4.2.1 Třída conditions

Třída conditions slouží ke stanovení podmínek. Pomocí podmínek se později filtrují soubory ve složce, kam Zim ukládá přílohy a synovské stránky. Rozlišuje, zda se jedná o soubor, který je určen ke smazání nebo zda se jedná o přílohu a bude se přesouvat.

4.2.2 Třída UserCanceled

Třída UserCanceled je potomek třídy exceptions. Slouží k ukončení výpočtu i z nejhlubších částí kódu. Ušetří mnoho kontrol návratových hodnot, proto je toto řešení efektivní.

4.2.3 Třída SplitDialog

Třída SplitDialog zastřešuje komunikaci s uživatelem pomocí dialogového okna. Od uživatele zjistí jak chce pokračovat v případě, že by dělil stránku, na které není žádný super-nadpis.

4.2.4 Třída DuplicatedDialog

Třída DuplicatedDialog uživatele informuje o problému shodných názvů prvků struktur a jejich možných následcích. Od uživatele vyžaduje odpověď, zda se má či nemá pokračovat v dělení či slučování. Největším problémem, na který třída upozorňuje, je víceznačný odkaz vedoucí na nadpis ve stránce. Ve stránce se na rozdíl od Indexu může vyskytnout stejný prvek. A pokud na jeden ze stejně pojmenovaných nadpisů vede odkaz, tak po rozpojení tuto situaci řešíme, že daný odkaz rozmnožíme a každou jeho kopii nastavíme, aby směřovala na jiný nadpis nebo stránku.

4.2.5 Třída MoveErrorDialog

Třída MoveErrorDialog zobrazuje uživateli upozornění, pokud se nezdaří přesun příloh.

4.2.6 NewRootPageDialog

Při slučování celého Indexu do jedné stránky se třída NewRootPageDialog zeptá uživatele, zda si opravdu přeje vše sloučit do nové stránky a případně na jméno stránky.

4.2.7 Třída Attachements

Při slučování stránek by se mohlo stát, že více stránek má stejně pojmenovaný soubor s různým obsahem. Třída Attachements zabezpečí, aby všechny přílohy měly jedinečné jméno, tak aby se soubory nepřepsaly.

4.2.8 Třída SplitPagePlugin

Hlavní třída pluginu – SplitPagePlugin je potomkem třídy PluginClass. Obsahuje třídy Split a Join a obaluje funkce využívané oběma třídami.

4.2.9 Třída Join

Spojování stránek je jednodušší než jejich dělení, již na začátku přesně víme, kterých stránek se spojování týká. Spojování se skládá z těchto operací:

Načtení souborů do paměti, ověření, že lze slučovat, zálohy stránky do které slučujeme, vykonání funkce join_pages, zkontrolování, že v podstromu nebyly stejně pojmenované nadpisy, uložení sloučení na disk, opravy zpětných souborů, převodu příloh a smazání původních stránek.

Funkce join_pages sjednocuje vstup pro funkci _join_pages, která již rekurzivně prochází stromem stránek a v každém uzlu volá funkci _consume_page, která stránku zpracuje. Tato funkce upravuje nadpisy, odkazy, přidává na konec stránky seznam souborů, jež nebyly zmíněny, aby při opětovném dělení třída Split poznala, kam danou přílohu přesunout.

4.2.10 Třída Split

Třída Split je podobná třídě Join, provádí inverzní operace. Při dělení se nejdříve celá super-stránka prozkoumá. Prochází se její strom a ve funkce conditional_heading_page_creator se podle nadpisů super-stránka rozdělí na stránky, které jsou uloženy do PagesOrderStructure. Poté, co jsou všechny stránky zpracovány se zjistí, kde bude jaká stránka finálně umístěna a jaké bude její jméno. Umístění souboru se může změnit, protože některé stránky se z důvodu zachování pořadí v Indexu přejmenují a tak vznikne jiná cesta. Určí se, kam se přesunou přílohy v případě vícenásobných odkazů na přílohu v rámci jedné super-stránky. Najdou se cíle odkazů, které předtím končily ve spojené super-stránce a opraví se všechny odkazy dotčené přejmenováváním. Odstraní se extra nadpisy na stránce, které umožňují uchovat jméno stránky během spojení. Odstraní se přidaný seznam neodkazovaných souborů přidaný při dělení. Přesunou se přílohy. A opraví se odkazy vedoucí ze třetích stránek na bývalou super-stránku.

4.2.11 Třída PagesOrderStructure

Třída PagesOrderStructure zajišťuje správnost pořadí stránek při dělení jejich přejmenováváním 2.4. Rovněž vytvoří strukturu souborů, jak ji dokáže zpracovat Index.

sectionNová syntaxe Pro přidání nové syntaxe – nadpisů a odkazů na konkrétní místo dovnitř stránky pomocí „#“ bylo potřeba upravit zdrojový kód

značkovacího jazyka wiki.py a jeho metod na převod z textu na xml strom a opačně.

4.2.12 Neomezeně úrovní nadpisů

Bylo potřeba upravit stávající syntaxi značkovacího jazyka Zimu, wiki, opět ve stejném zdrojovém souboru, aby umožnil ukládat a načítat nadpisy vyšších úrovní. Jak jsme již zmínili v odstavci 1.3.2, pro nadpisy úrovní 1–5 se pro uložení používalo značky dvou až šesti znamének rovnosti (2–6 „=“). Pro nejvyšší úroveň nejvíce a pro zkrácené psaní v „on the fly“ režimu, využitelného přímo z editoru měly opačný význam. V případě ukládání do souboru navíc nadpisy končí stejným počtem znamének rovnosti jakým začínají. V obou případech jsou mezi nadpisem samotným a znaménky bílé znaky, lze použít mezeru nebo tabulátor. Aby byla nová syntaxe kompatibilní se stávajícím a umožnila více než pět úrovní, bylo nutné zavést nový způsob. Nová syntaxe se zapisuje ve tvaru dvě nebo tři rovnítka, číslo, mezera a pak text nadpisu. Číslo určuje, o jakou úroveň nadpisu se jedná. Nadpisy prvních úrovní v nové syntaxi zapisujeme: „==1 Nadpis první úrovně“ a „===1 Nadpis, oddělující stránky, první úrovně“. Tato syntaxe netrpí dualitou režimů.

Vnitřně jsou super-nadpisy jako nadpis vyšších úrovní. Všechny funkce programu s nimi umí pracovat a nebylo potřeba nic přidívat. Úrovně byly obsazeny, ale omezení jejich velikosti bylo pouze v wiki.py, kterou jsme upravili. Nechali jsme rezervu normálním nadpisům až do deváté úrovně, nyní využívají rozsah 1–5 (6) šestá jde zapsat, ale uloží se jako pátá. A vyšší než deset využíváme pro potřeby super-nadpisů. Výhodou je využití starého tagu v xml stromu, takže lze dostat iterátor na všechny nadpisy. Nevýhodou je, že plugin tableofcontent neočekává vyšší nadpisy, a proto nefunguje správně.

4.2.13 Dynamické tagy pro nadpisy

Všechny tagy jsou v Zimu inicializovány staticky. Což nám činilo problém, jak jsme v kapitole 2.3.4 zmínili. Potřebovali jsme totiž nagenerovat takové množství úrovní nadpisů, kolik úrovní uživatel během práce použije. Jelikož jsme přidali podporu pro neomezené množství úrovní, museli jsme najít jiný způsob, jak tagy inicializovat.

Funkce touch heading, kontroluje zda tag pro danou úroveň již existuje. V případě potřeby ho vytvoří.

Všechny linky jsou uloženy dohromady, s výjimkou speciálních linků na obrázky img, které se pak zobrazují v textu.
Při kliknutí na link se vyhodnocuje jeho lokace – stránka, která se má otevřít. Metodu bylo třeba upravit tak, aby podporovala #hashtag odkazy a tento přesah dovnitř stránky ignorovala – má vrátit stránku a ta se # nezmění.

4.3 Regulární výrazy

V této části popíšeme speciální koncepty regulárních výrazů Pythonu [5], použitých v této práci. Na příkladech nastíníme, k čemu byly použity. Speciální koncepty jsou koncepty, které nejsou v UNIX normě (viz [6] česky shrnuté [7]).

Regulární výrazy se v Zimu používají k parsování uživatelem psaného textu. Kdykoli uživatel napíše bílý znak (mezeru, nový řádek, tabulátor), obslužná metoda pošle část textu, nejvíce celý řádek, s odebraným posledním znakem, skupině regulárních výrazů. Ty v daném pořadí testují tento textový řetězec a podle toho, který regulární výraz první uspěje, se pak určí zda a jaká speciální syntaxe byla použita a vytvoří se příslušný uzel (nadpis, odkaz...) v Pythoní třídě ElementTree – xml stromu [9], jenž reprezentujícím stránku.

Tato ukázka obsahuje regulární výraz rozpoznávající validní odkaz směřující na konkrétní místo ve stránce.

```
(?::|\+)?\w[\w\.\-\(\)]+(?::\w[\w\.\-\(\)]+)*:?\#\w[\w\.\-\(\)]+$
```

Předcházející výraz je zde doplněn o bílé znaky, aby byl lépe čitelný.

```
(?:    :|\+       )?
     \w[\w\.\-\(\)]+
(?:    :
     \w[\w\.\-\(\)]+
)*
:?\# \w[\w\.\-\(\)]+
$
```

Prvním konstruktem ve výrazu jsou neodkazovatenlé závorky „`(?: )`“. Na závorky v regulárních výrazech se lze odkazovat. To ovšem není vždy to, co od nich vyžadujeme. Python má ve své implementaci i závorky na něž reference nevznikne, anglicky „non–capturing“.

Výraz „`\w`“ je zkratkou za jakékoli písmeno nebo číslo.

Opakující se výraz `\w[\w\.\-\(\)]+` obsahuje znaky, které se mohou vyskytnout v názvu validní stránky či nadpisu. Stránka musí začínat alfa-numerickým znakem, ale na dalších pozicích může být kulatá závorka, tečka nebo pomlčka. Pro zachování rozumné kompatibility tohoto výrazu a výrazu „page_re“, obsahuje tento výraz podmínku pro dvoupísmenné názvy stránek, jinak by stačilo na konci použít „`*`“ místo „`+`“. Zim sám totiž umožňuje i jednopísmenné názvy stránek, ale v odkazech pro automatické rozpoznávání preferuje více znaků, aby se omezilo riziko konfliktů.

Dalším speciálním konceptem pythoních regulárních výrazů použitých v práci je `(?! )`. Tento výraz uspěje, pokud selže výraz uvnitř závorky, anglicky „lookahead negative asertation“.

Příklad: sekvence `(?![-_.)(])` zajistí, že se další znak nebude ze skupiny uvnitř hranatých závorek. Výraz pro rozpoznání všech typů nadpisů.
`^(?:(?:(===?)([0-9]+))|(=2,7))\s*(.*?)\s*(\1|\3)?` obsahuje oba typy závorek, odkazy na závorky (přes „\číslo“), „`\s`“, vybírá všechny bílé znaky.

Závěr

Touto bakalářskou prací bylo vytvořeno nové rozšíření pro zim-wiki, které umožňuje plynulé přecházení mezi strukturou jeho stránek a nadpisů. Smyslem této práce bylo doplnit stávající program. Zaměřili jsme se na to, aby rozšíření umožnilo uživateli s programem lépe, efektivněji a pokud možno i pohodlněji pracovat.

Ačkoli původním záměrem bylo vytvoření pluginu do programu Zim, postupem času se ukázalo, že cíl nelze pouhým pluginem uskutečnit, zvláště pokud by měl program zůstat uživatelsky přívětivým a vůbec spravovatelným. Plugin samotný by sice umožňoval spojovat a rozdělovat stránky, avšak neměl by možnost odkázat se dovnitř stránek a měl by omezený počet úrovní nadpisů, které by vedly na základě zvoleného řešení buď k nemožnosti zachovat strukturu stránek, nebo k deformaci nadpisů nutným prefixem. Proto bylo nutné nakonec přistoupit k vytvoření nové větve programu, která podporuje a doplňuje od počátku plánovaný a současně vytvořený plugin. Dohromady pak tyto dvě části (plugin a nová větev programu) představují nové rozšíření programu Zim prezentované v této bakalářské práci.

Přínos této práce spočívá v obohacení stávajícího programu Zim o nové funkce, především o možnost plynule přecházet mezi stromovou strukturou stránek a strukturou nadpisů. Struktura stránek má neomezené množství úrovní, avšak struktura nadpisů měla doposud úrovní pouze pět. Díky novému rozšíření bylo množství úrovní nadpisů zvýšeno na neomezený počet, a tak vyrovnáno s množstvím úrovní ve struktuře stránek, což zvyšuje komfort pokročilejšího uživatele.

Přínosem je i nový způsob pohledu na stránku a na obsah stránek. Stránka nyní nezobrazuje pouze obsah jedné stránky, ale umožňuje pohled na více stránek současně včetně pro ně relevantní části struktury Indexu.

Dalším přínosem tohoto rozšíření Zimu je v možnosti slučovat a rozdělovat stránky bez narušení jejich odkazů. Právě s odkazy by mohly nastat komplikace ve chvíli, kdy by se uživatel rozhodl své stránky různě štěpit či spojovat. Díky tomuto rozšíření programu se však uživatel nemusí obávat, že by mu odkazy zmizely, přestaly fungovat, či se začaly chovat nestandardně.

V novém rozšíření programu je uživateli umožněno, aby se mohl odkazovat přímo na nadpisy uvnitř stránek, neboť Zim dosud umožňoval pouze odkazy na celé stránky.

Přínos této práce pro autora

Prozkoumal jsem mnoho slepých větví ve vývoji, abych se dostal k finálnímu programu. Jedním z největších přínosů, které mi tato práce dala, je kromě toho, že jsem se naučil pro mě nový, zajímavý jazyk i to, že jsem se naučil pracovat s cizím programem a dotvářet jej. Toto byl první projekt na kterém jsem se podílel, kdy kód který běží, není mnou plně řízený/ napsaný. Byla to pro mě veliká výzva.
Ze začátku práce jsem cizí kód až přehlížel. Nyní bych již postupoval jinak. Jako příklad uvedu, že jsem si napsal vlastní parser, který byl sice rychlejší a vracel mi text jak jsem si představoval, ale s každou změnou jazyka by bylo potřeba ho upravit, aby fungoval. Jenže v tu chvíli jsem si uvědomil, jak hloupé řešení je

vyrábět vlastní parser, když už jiný někde musí existovat. Prací na parseru jsem si ale oblíbil regulární výrazy a naučil jsem se i jejich pythonní rozšíření.

Zajímavější konstrukty, které nakonec přežily až do finální verze, uvádím v kapitole 4.3. Toto řešení jsem tedy opustil a hledal jsem kde se nachází a jak se používá alternativa k mému pareru.

Pro zkoumání kódu jsem ze začátku nepoužíval zdrojové soubory, ani jsem totiž nevěděl jak v nich hledat to, co potřebuji. Byla to pro mě spousta neznámého textu, do kterého jsem nevěděl ani kudy vniknout. Rozhodl jsem se použít debuger, ve kterém jsem program krokoval, abych zjistil, co se kde a jak používá. Z něj jsem se snažil vyčíst vše potřebné. Zobrazil mi obsah proměnných, jejich názvy, názvy tříd a z nich jsem se snažil vydedukovat co a jak funguje. Některé věci se takto daly odhalit snadno, jiné hůř, pro značnou část to pro mě bylo nadmíru obtížné. Přesto jsem získal poměrně dobrou představu co a jak pracuje. Později, když už jsem uměl přečíst zdrojový kód a porozuměl mu, jsem tuto představu upřesnil.

Když jsem v práci dospěl takto daleko a kriticky si prohlédl vlastní kód, usoudil jsem, že ho musím přepsat. A mnoho věcí jsem kompletně změnil. Bez zásahu snad nepřežila jediná funkce. V tomto místě jsem i opustil koncept pluginu. V té době, kdy jsem měl vlastní parser, mohl —Split page— fungovat samostatně, pouze by místo nadpisu zobrazil svůj značkovací tag.

Tato práce mne též naučila plánovat. Měl jsem již sice určitou představu z některých přednášek a cvičení jak správně postupovat, ale nedokázal jsem tyto teoretické znalosti dobře aplikovat. V této práci se mi v další slepé cestě ukázalo, že je nutností si předem všechny podstatné věci promyslet a na detaily se zaměřit až později, oproti pro mě přirozenějšímu postupu - načíst problém, vyřešit ho a posunout se dále.

V rámci práce jsem se několikrát dostal do kontaktu s původním autorem programu, což mne velmi obohatilo.

Naučil jsem se pracovat se systémem kontroly verzí (Version control), kterou jsem do té doby nikdy nepoužil.

Díky tomuto programu jsem začal sledovat dění na Launchpadu a opravil jsem několik chyb, jelikož jsem věděl, kde se ve zdrojovém kódu Zimu nacházejí. Přidal jsem několik vylepšení do této verze Zimu, protože mě zaujaly.

V praxi jsem si vyzkoušel ukládání dat do SQL databáze i práci s ním. Řešení používající databázi jsem zavrhl z důvodu uvedených v kapitole 1.2.

Pronikl jsem do wiki syntaxe, která je sice jednoduchá, ale přesto užitečná. Použil jsem ji při psaní textu bakalářské práce na jiném počítači připojeném přes SSH v editoru Nano.

Dalším přínosem, který opět není vidět v práci samotné, protože jsem nalezl vhodnější řešení, pro mě bylo naučení se technologie XML Path. Používal jsem ji pro hledání uzlů v XML stromu.

Na začátku práce pro mě byl Zim úkol, program, který jsem moc neznal a zdál se mi příliš prostý. Nyní ho vnímám jako neocenitelného pomocníka, ve kterém se navíc vyznám a vím, jak pracuje.

Pro mně nejtěžší bylo proniknutí do Zimu. Je to největší nejkomplexnější program, jehož zdrojový kód jsem dosud studoval.

Změny provedené v programu se v budoucnu stanou součástí hlavní vývojové větve. Již nyní je v ní využito řešení vytvořené na základě mého kódu, řešícího

nepodporu národních abeced při převádění slov v CamelCase tvaru na odkazy.

Další možná rozšíření

Návrhů na další rozšíření Zimu je mnoho. Zde předkládám několik doporučení pro jeho další vývoj:

- Zajímavými možnými vylepšeními by mohl projít Index stránek. Do původního Indexu stránek by se mohlo přidat zobrazování jednotlivých nadpisů.

  - Minimalistické řešení, které by ani nemuselo zasáhnout příliš do zdrojového kódu programu by mohlo fungovat tak, že by se po kliknutí na nadpis v Indexu zobrazila příslušná stránka v místě nadpisu tak, jako to dělá mnou přidané odkazování na nadpisy.
  - Smazání rozdílu mezi stránkou a nadpisem. Vše by se interpretovalo jako nadpis a kliknutím na nadpis by se zobrazilo buď vše, až do nadpisu další takové úrovně, a nebo jen současný nadpis, do nejbližšího dalšího nadpisu.
  - Rozdělit Index na stránky a podstromy. Stránka by se zobrazovala dál stejně, ale pokud by se kliklo na podstrom, Zim by zobrazil sloučené stránky stejně, jako je zobrazilo toto rozšíření.

    V obou posledně jmenovaných případech by struktura složek a souborů na disku ztratila přímou návaznost na strukturu Indexu a pravděpodobně by bylo třeba předělat i celý systém zacházení se soubory, protože při změně pohledu by se provedlo příliš mnoho diskových operací na to, aby práce byla komfortní – na přepnutí stránky by se totiž chvíli čekalo. Jiným řešením by bylo předělat načítání. Vložit nějakou virtuální mezivrstvu mezi soubory a jejich zobrazení v programu, které by vhodně na sebe mapovalo tyto vrstvy. Pro významnou část by sice šlo použít nástroje založené na mém rozšíření, ale v podstatě by se jednalo o sice příbuzný, ale nový program.

- Pokud v současnosti odkazy dovnitř stránek nenaleznou správný nadpis, otevřou správnou stránku v místě jejího opuštění. Avšak v budoucnu by takové odkazy mohly na konci stránky vytvořit nový nadpis daného jména. Obdobně fungují odkazy na stránky, kdy je pouhé odkázání vytvoří. Pokud odkaz zanikne a stránka nebyla editována a zůstala pouze původní prázdná stránka, tak stránka zanikne spolu s odkazem. Stejně tak by se mohly chovat nadpisy na stránce. Pokud by se nadpis na stránce změnil, bylo by nutné opravit odkaz tak, jako se děje u stránek.

- Při spojení stránek do super-stránky se datum zobrazuje u každé původní stránky. Mohla by se doplnit volba, která by dala uživateli na výběr, zda chce při spojení stránek data vytvoření původních stránek smazat či ponechat. Při případném pozdějším dělení by uživatel měl na výběr mezi možnostmi: obnovit původní datum, použít stejné datum jako má super-stránka nebo neobnovovat. Uchování původních dat (dnů, kdy byly stránky vytvořeny) by mohlo být implementováno pomocí databáze.

- V Zimu funguje tlačítko Vrátit (Undo) tím způsobem, že se po přechodu na jinou stránku zapomenou předchozí úpravy na původní stránce a již se k nim nelze vrátit. Tlačítko Vrátit pro převod mezi strukturami by ale podle očekávání mělo fungovat tak, že vrátí stav do podoby, ve které byl před stisknutím tlačítka.

- Uživatel bude častěji stránky spíše dělit (text přibývá), než aby je slučoval. Důvod ke sloučení bude mít, pokud se bude chtít podívat na obsah bratrovských stránek najednou. Pokud by se mu jednalo pouze o dočasné sloučení, při kterém nebude chtít měnit samotnou strukturu, pak by bylo možné přidat funkci, která by uživateli umožnila náhled z jiné úrovně, aniž by se jakkoli měnila struktura stránek. Takovéto náhledové sloučení by mělo být mnohem rychlejší, než plnohodnotné dělení.

- Přidání zpětných odkazů i pro textové odkazy vedoucí na přílohy v jiných stránkách. Pokud se nyní stránka přejmenuje, tak se může stát, že takovéto odkazy již nebudou správně určovat cílovou přílohu. S tím by zároveň měla přibýt podpora pro opravu odkazů při přesunu stránky (jak zdrojové, tak cílové).

Zim je v současné době dle mého názoru nejlepším programem sloužícím jako mnohostranný nástroj k tvorbě, doplňování a třídění osobních poznámek, úkolů, jako diář, organizér, sofistikovaný zápisník pro studenty, aj. Je dobrým a prospěšným pomocníkem ke správě poznámek.

Přál bych si, aby má práce našla uplatnění mezi uživateli Zimu, aby se pro ně mé vylepšení stalo užitečným nástrojem a aby se jim s ním příjemně a úspěšně pracovalo.

Seznam použité literatury

[1] Members of "Zim": Active members. *Launchpad: "Zim" team* [online]. © 2004-2014 [cit. 2014-05-22].
Dostupné z: `http://launchpad.net/~zim-wiki/+members#active`

[2] Zim User Manual: Wiki Syntax. *Zim - A Desktop Wiki* [online]. [cit. 2014-05-22].
Dostupné z: `http://zim-wiki.org/manual/Help/Wiki_Syntax.html`

[3] Formatting Syntax. *DokuWiki* [online]. Last modified: 2014-05-04 19:31 [cit. 2014-05-22].
Dostupné z: `https://www.dokuwiki.org/wiki:syntax`

[4] Link to anchors within pages. *Launchpad: Zim desktop wiki* [online]. 2009-05-27 [cit. 2014-05-22].
Dostupné z: `http://bugs.launchpad.net/zim/+bug/380844`

[5] Regular expression operations. *Python* [online]. © 1990-2014, last updated on May 18, 2014 [cit. 2014-05-22].
Dostupné z: `https://docs.python.org/2/library/re.html`

[6] The Open Group Base Specifications Issue 7 IEEE Std 1003.1™, 2013 Edition (The Single UNIX). *The Open Group Publications Server* [online]. © 2001-2013 [cit. 2014-05-22].
Dostupné z: `http://pubs.opengroup.org/onlinepubs/9699919799/`

[7] FORST, Libor. *Shell v příkladech, aneb, Aby váš UNIX skvěle Shell*. 1. vyd. Praha: Matfyzpress, 2010. s. 346. ISBN 978-807-3781-521.

[8] Zim: Desktopová wiki, outliner, program na poznámky nebo pomůcka pro studenta. OTT, Vlastimil. *Elektronický Ott* [online]. 15. prosinec 2010 [cit. 2014-05-22].
Dostupné z: `http://www.e-ott.info/2010/12/15/zim`

Zim jako poznámkovník – jak pracovat se stránkami, jak odkazovat jinam a vkládat obrázky. OTT, Vlastimil. *Elektronický Ott* [online]. 9. leden 2011 [cit. 2014-05-22].
Dostupné z: `http://www.e-ott.info/2011/01/09/zim`

Zim jako nástroj pro vaši osobní agendu: Kalendář, úkoly. OTT, Vlastimil. *Elektronický Ott* [online]. 17. březen 2011 [cit. 2014-05-22].
Dostupné z: `http://www.e-ott.info/2011/03/17/zim`

Zim 0.55 vydán, podporuje číslované seznamy, Markdown a obsah. OTT, Vlastimil. *Elektronický Ott* [online]. 4. březen 2012 [cit. 2014-05-22].
Dostupné z: `http://www.e-ott.info/2012/03/04/zim`

[9] The ElementTree XML API. *Python* [online]. © 1990-2014, last updated on May 18, 2014 [cit. 2014-05-22].
Dostupné z:
`https://docs.python.org/2/library/xml.etree.elementtree.html`

Seznam použitých zkratek

GUI	Graphic User Interface	grafické uživatelské rozhraní
GNU GPLv2	GNU General Public License version 2	
GNU	GNU's Not Unix!	
WYSIWYG	What you see is what you get	„co vidíte, to obdržíte“.
HTML	HyperText Makup Language	
XML	eXtensible Makup Language	
SQL	Structured Query Language	

Super-nadpis	nadpis sloužící jako oddělovač stránek
Super-stránka	stránka obsahující alespoň jeden super-nadpis

Přílohy

Obsah přiloženého CD

Na CD se nacházejí následující soubory a složky:

- **prace.pdf** – obsahuje text této práce
- **zim.zip** – obsahahuje zdrojové soubory
- **doc/** – obsahuje programátorskou dokumentaci
- **WWW/** – obsahuje citované webové stránky
- **priklad.zip** – obsahuje příklad použití