Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Kontrola typografie v Adobe InDesign

Plzeň 2011 Marcela Kubová

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracovala samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 12. května 2011

Marcela Kubová

Abstract

The main aim of this work was to create a special tool for automatic typography check in Adobe InDesign. Typography check is based on straight typesetting rules. In order to cope with this task, JavaScript scripting programme was chosen or better said its variant working directly with Adobe InDesign. This paper contains selected typography rules analysis and description of their typical violation. The rules were chosen according to their potential of automation. Automation is being conducted on two levels: automatic detection and replacement of the concrete word or check of the detected word according to a set of predefined rules from which the right replacement is chosen. The paper also handles project details including close description of scripting steps.

Obsah

1 Úvod **1**

2 Teoretická část **3**
2.1 Konzultace 3
2.2 Regulární výrazy 4
2.3 Hladká sazba 4
2.4 Pravidla hladké sazby 5
2.4.1 „Bílé“ znaky 6
2.4.2 Interpunkční znaménka 6
2.4.3 Závorky 10
2.4.4 Spojovník, pomlčka 11
2.4.5 Podtržítko, lomítko 11
2.4.6 Čísla 11
2.5 Možnosti automatických oprav 12

3 Praktická část **14**
3.1 Popis projektu 14
3.2 Skriptování v Adobe InDesign 15
3.2.1 Popdora skriptování 15
3.2.2 Spouštění skriptů v InDesign 15
3.2.3 Práce s InDesignem 16
3.2.4 Text, textové rámečky a práce s textem 16
3.3 InDesign a regulární výrazy 17
3.4 Skript pro automatickou kontrolu typografie 18
3.4.1 Spuštění skriptu a načtení údajů 18
3.4.2 Způsoby úprav textu 19
3.4.3 Rozsah textu určeného ke kontrole 20
3.4.4 Vybrání výrazů pro prohledání 20
3.5 Prohledávání dokumentu skriptem 21
3.5.1 Ukládání dat 21
3.5.2 Způsob prohledávání 22

3.5.3 Vyhledávání výrazů 22
3.5.4 Způsoby kontroly typografie 22
3.5.5 Nahrazení výrazů 24
3.6 Ukončení skriptu 25
3.7 Rozšíření skriptu 25

4 Závěr **27**

A Uživatelský manuál **32**

B Přiložené CD **36**

1 Úvod

Práce se zaměřuje na prostudování pravidel hladké sazby, možností automatického vyhledávání porušení těchto pravidel a automatické opravy. Součástí práce je navržení heuristických metod, které budou vybrané typografické chyby vyhledávat a opravovat. Tyto metody jsou implementovány jako skript pro program Adobe InDesign.

Lidé, kteří se zabývají předtiskovou přípravou, často zpracovávají texty jiných autorů, do kterých bývá někdy i ve značné míře zaneseno mnoho porušení typografických pravidel. Autoři textů nemají znalosti o pravidlech hladké sazby a texty tvoří, řekněme, podle svého uvážení. I když se sama v zaměstnání nepohybuji přímo v oblasti DTP, pracovala jsem s texty na určitá odborná témata dodanými od různých autorů. Součástí mé práce bylo tyto texty sjednotit a zajistit korektury. Během předtiskové přípravy je tedy nutné tyto odchylky vyhledat a zaměnit za správné výrazy. Při ručním procházení textu je tato práce časově náročná a může dojít k přehlédnutí chyb. Pro zjednodušení této činnosti a minimalizování časových nároků na „kontrolní“ část zpracování textu je vhodné zavést automatizaci.

Sázecí program Adobe InDesign umožňuje zavedení skriptů, které je možné v tomto programu spouštět. Jejich využití je různorodé. Přínos mají skripty v hromadném zpracovaní mnoha dokumentů či automatizaci rutinních činností. Richard Krejčí[6] hodnotí možnost skriptování jako jednu z nejzajímavějších vlastností, zároveň ale konstatuje, že i přes svou nespornou užitečnost je skriptování na pre-pressových pracovištích využíváno pouze velmi omezeně. Důvodem je obava uživatelů z nutnosti osvojovat si dovednosti z oblasti programování a nejasná představa, co by jim skriptování mohlo přinést. Cílem práce je vytvoření skriptu pod Adobe InDesign, který bude automaticky vyhledávat typografické chyby. Uživatelé Adobe InDesign pak výsledný skript budou moci využít bez znalosti skriptování.

Navržený skript nabízí uživatelům možnost automatické kontroly dokumentu, tedy vyhledání a záměnu výrazů, které porušují pravidla hladké sazby. Uživatel si vybírá z nabídky definovaných typografických pravidel, která jsou načtena ze souboru. Je tedy možné si definovat další výrazy podle potřeb uživatele. Rozsah prohedávání je nastaven na celý dokument, na označenou část dokumentu či na všechny otevřené dokumenty. Dále skript umožňuje vybrat, co bude s vyhledanými výrazy učiněno. Mohou být automaticky zaměněny,

po záměně vyznačeny buď tagy nebo změnou barvy písma. Ve skriptu je možné zvolit také vyhledání a označení nalezených míst (tagy, změna barvy) a náhrada za správný výraz je ponechán na uživateli. Poslední možností je průběžné upozorňování na vyhledané výrazy, kde je uživateli nabídnut návrh záměny.

2 Teoretická část

Před zahájením práce na skriptu bylo nutné seznámit se s pravidly hladké sazby a navrhnout možnost začlenění do skriptu. Zároveň bylo vhodné zjistit časté typografické chyby, které grafici při zpracování textů řeší.

2.1 Konzultace

Největší problémy se objevují, pokud uživatel dodává texty připravené v některém z textových editorů. Svou práci jsem tedy zaměřila na vyhledávání „prohřešků“, které do textu vloží sám autor, než odevzdá svůj text k typografickému zpracování.

Vyšla jsem z informací od grafičky Alice Neugebauerové:

- nadbytečné mezery,
- špatně umístěné mezery kolem konce odstavců,
- mezery před interpunkčními znaménky,
- konce odstavců několikrát za sebou (toto platí i pro řadu mezer, které uživatelé stále v hojné míře používají) — Alice pro toto používá výraz „kochající se redaktor“,
- měkký enter neboli pevný konec řádku,
- pomlčky — vyhledání,
- závorky — vyhledání,
- tři tečky,
- mezery za lomítkem.

Zároveň ona sama by využila toho, kdyby bylo možné kontroly nebo zpracování provést ve více souborech najednou. Její styl práce je takový, že například časopis rozděluje na více částí, soubory po článcích a pak by pro ni zpracování bylo výhodné ve všech souborech k danému tématu.

2.2 Regulární výrazy

Vyhledávání typografických chyb je z programátorského hlediska problém, který lze řešit regulárními výrazy.

Obecně jsou regulární výrazy silným pomocníkem při práci s textem. Jejich prostřednictvím lze:

- vytahovat z textových dat údaje, které nás zajímají,
- přetvářet je do podoby, kterou potřebujeme,
- vyhledávat a nahrazovat v textových editorech a dalších programech.

Při navrhování jednotlivých regulárních výrazů jsem problematiku rozdělila na dvě části. Pro obě části jsou znaky nebo skupiny znaků vyhledány pomocí regulárních výrazů, dále se s nimi pracuje odlišně. První část lze pomocí regulárního výrazu vyhledat a rovnou automaticky nahradit. Do této části patří např. záměna nadbytečných znaků nebo záměna pevné mezery za jednoznakovou předložkou. V druhé skupině jsou výrazy, které po vyhledání zhodnotí příslušná fuknce a podle jejich významu vybere správný výstup pro nahrazení. Do této části patří např. kontrola všech interpunkčních znamének, kdy je nutné ošetřit jednotlivé situace, které ve výrazu mohou nastat. Pravidla a podmínky shrnuji v kapitole 2.4 a jejich podkapitolách.

2.3 Hladká sazba

Hladká sazba obsahuje plynulý text, je to sazba odstavců na určenou šířku z jednoho typu a stupně písma. Řídí se obecně typografickými pravidly i pravidly českého pravopisu, samozřejmě jsou pro praxi velmi potřebná pravidla pro sazbu různých znaků. Vychází z normy ON 882503, která definovala základní pravidla sazby.

Kromě pravidel hladké sazby existuje norma ČSN 01 6910 Úprava písemností zpracovaných textovými editory, jejíž poslední aktualizace proběhla v dubnu 2007. Ta stanoví pravidla pro grafickou úpravu textů při práci s textovými editory, určuje způsob psaní interpunkčních znamének, zkratek, značek a čísel, stanoví postupy při zvýrazňování a členění textů a zásady for-

málního uspořádání textového sloupce. Tato norma je využívána například v obchodní korespondenci.

Nutno říci, že mezi jednotlivými normami neexistuje jednotnost a dochází k rozporům. Jako příklad uveďme nejednoznačnost při psaní času, tedy hodin, minut a sekund. Pravidla hladké sazby určují pro vyjádření času následující tvar:
`HH.MM:SS` `10.15:30`,

zatímco norma, kterou se řídí obchodní korespondence a evropská norma uvádí tvar:
`HH:MM:SS` `10:15:30`.

Vytvořený skript pro automatickou kontrolu textu pro Adobe InDesign se řídí pravidly hladké sazby, jak vychází ze zadání práce. Práce se zabývá částí hladké sazby věnující se pravidlům používání sazby jednotlivých znaků, a to takových, která mohou být kontrolována automaticky skriptem. Nejsou zde uváděna pravidla pro sazbu odstavců a práce s nimi. Před zahájením práce na skriptu jsem konzultovala s grafičkou v naší společnosti a zjišťovala jsem, s jakými potížemi v textu se často potýká. Dále stručně uvedu vybraná pravidla hladké sazby, která jsem zařadila do kontrolního skriptu. Jedná se o takové znaky a jejich nesprávné použití, které se jednak často objevují v autorských textech a zároveň jsou vhodné pro automatizaci.

2.4 Pravidla hladké sazby

Kontrolní skript využívá vybraných pravidel hladké sazby. Zaměřuje se především na následující skupiny problémů:

- chyby při psaní „bílých“ znaků,
- kontrola interpunkčích znamének,
- pomlčka, spojovník, lomítko, podtržítko,
- kontrola číselných výrazů.

2.4.1 „Bílé“ znaky

Jak jsem již uvedla v kapitole 2.1, konzultací s grafiky v DTP studiu i z vlastní zkušenosti jsem do problematiky prohledávání chyb zařadila kontrolu nesprávně uváděných „bílých“ znaků. V kontrolovaných textech se velmi často objevují nadbytečné znaky jako např. dvě nebo více mezer za sebou, více ostavcových znaků nebo tabelátorů za sebou či mezery před začátkem odstavce. Stále ještě se řada uživatelů nenaučila používat jiné formátování textu než odsazování pomocí „nekonečné“ řady mezer nebo znaků konce odstavce a dodává své texty ke zpracování grafikem výrazně zanesené nevhodnými úpravami. Ve skriptu jsou pomocí regulárních výrazů uvedeny nejčastější nadbytečné a opakující se znaky a ty jsou odstraněny.

2.4.2 Interpunkční znaménka

Praktická typografie[1] uvádí, že interpunkci začal používat benátský nakladatel a tiskař Aldus Pius Manutius, který zavedl čárky, středníky a dvojtečky. Vykřičník vznikl z latinského zvolání Iyo, tedy zkratky Io napsané pod sebe. Podobně je na tom otazník, který svému původu vděčí latinskému slovu Questio, spíše jeho zkratce Qu, opět zapsané pod sebou.

Pro všechna interpunkční znaménka platí, že se sází těsně za (poslední) slovo ve větě a další slovo je oddělené normální mezerou. Pokud se vedle sebe sejdou dvě nebo více interpunkčních znamének, nedělá se mezi nimi mezera. Ta je umístěna až za posledním znaménkem. Ve své praxi jsem se často v textech setkala se záměnou, kdy byla nesprávně uvedena mezera před interpunkčním znaménkem namísto za, případně byly mezery umístěny z obou stran znaménka a podobně. Všechny tyto chyby jsem při návrhu zařadila do realizace skriptu, další možnosti kontroly vychází z pravidel pro jednotlivá znaménka.

Při návrhu skriptu a po konzultacích jsem se rozhodla pro vyhledání všech interpukčních znamének v textu pomocí regulárních výrazů, jejich seznam je předáván dál do funkcí, které provádějí kontrolu správnosti. V případě, že je interpunkce umístěna správně, přechází se na další vyhledanou pozici. Pokud je nalezena typografická chyba, je výraz podle pravidel sazby upraven a zaměněn.

Chybný výraz	**Popis**
␣.␣	Chybně umístěná mezera před tečkou na konci věty.
.	Vynechání mezery za tečkou na konci věty (pokud není věta uvedena v závorce).
.␣)	Chybná mezera mezi interpunkčním znaménkem a uzavírací závorkou.
␣.␣	Chybně umístěná mezera před a za tečkou ve spojení s čísly (hodiny, kapitoly).
..	Dvě tečky, ty mohou znamenat chybné použití dvou teček namísto správné jedné nebo špatně zapsané tři tečky, kdy byla jedna tečka zapomenuta. Automatickou kontrolou toto není možné rozhodnout, uživatel je ke konci prohledávání dokumentu na dvě tečky upozorněn.

Tabulka 2.1: Chybné výrazy s použitím tečky

Tečka

Pokud by za sebou mělo být umístěno více teček, především na konci věty, sází se pouze jedna. Pokud jsou nadpisy číslované (např. 1.1.2), sází se bez mezer.

Tečka má mnohá další využití v různých významech, kdy se zapisuje bez mezer. Používá se při zápisu:

- hodin,
- názvů souborů,
- internetové adresy,
- verzí programů,
- kapitol.

Ve vyhledávacím skriptu je kontrolováno správné umístění tečky v textu, u čísel se tečka vyskytuje vždy bez mezer (u hodin, číslovaných nadpisů, kapitol či verzí programů). Nejčastější chyby shrnuje tabulka 2.1.

Otazník a vykřičník

V textu mohou být uvedeny také v závorkách, otazník jako výraz pochybnosti, vykřičník jako upozorňující výraz. V tomto případě se mezera nepoužívá, vizte kapitolu 2.4.3. Pokud je otazník nebo vykřičník součástí určitého výrazu na konci věty, koncová tečka už se za nimi nepíše. Chyby, které se mohou vyskytnout v textu, opět shrnuje tabulka 2.2, která ukazuje případy s použitím otazníku, stejné případy se definují i pro vykřičník.

Chybný výraz	**Popis**
␣?␣	Chybně umístěná mezera před otazníkem na konci věty.
?	Vynechání mezery za otazníkem (pokud není uveden v závorce).
(␣?␣)	Chybně umístěná mezera před a za otazníkem uvnitř závorek.
?.	Použití koncové tečky, před kterou následuje výraz s otazníkem.

Tabulka 2.2: Chybné výrazy s použitím otazníku a vykřičníku
Pozn: Stejné výrazy lze definovat také pro vykřičník.

Čárka

Kromě umístění ve větě jako interpunkční znaménko se dále čárka používá na vyjádření desetinné čárky v číslech a matematických výrazech. V těchto případech se sází z obou stran těsně. Při návrhu kontroly čárky bylo potřeba zohledit, zda se v okolí čárky z obou stran objevuje číslo a ošetřit sazbu bez mezer. Pokud je číslo pouze z jedné strany čárky, je číslo součástí textu a sazba bude s mezerou za interpunkcí.

Při zjištění čísel z obou stran a umístěné mezery za čárkou je dále potřeba zjišťovat, jak vypadá okolí čísel a zda použití s mezerou je správné. To se stává v případech, kdy jmenujeme několik čísel oddělených čárkami nebo spojkami *a, i, nebo*, vizte příklad.

Příklad: Ukázka výskytu jedné tečky namísto dvou

Potřebuješ půjčit 1 000, 2 000 nebo 3 000?

Chybný výraz	**Popis**
␣,␣	Chybně umístěná mezera před čárkou.
,	Vynechání mezery za čárkou (čárka nemá význam desetinné čárky v čísle).
1␣,␣1	Při významu desetinné čárky použití mezery před a (nebo) za čárkou.

Tabulka 2.3: Chybné výrazy s použitím čárky

Středník

Sází se podle stejných pravidel jako čárka, tedy bez mezery k předchozímu písmenu a za ním následuje vždy malé písmeno. Při výčtu desetinných čísel se používá k oddělení jednotlivých čísel a tedy opět s mezerou za znaménkem.

Tři tečky

Používáme pro vyjádření vynechané části textu nebo jako naznačení pomlky ve větě. Sází se speciálním znakem a k předcházejícímu výrazu se přisazuje bez mezery. Za třemi tečkami se umisťuje mezera. Končí-li věta třemi tečkami, plní tyto tečky zároveň ukončovací funkci, další tečka se tedy za nimi nepíše. V kontrolním skriptu je tedy kromě vyhledání třech teček a záměny za speciální znak nutno ošetřit také kontrolu další tečky, která zde již být nemá.

Chybný výraz	**Popis**
␣...␣	Chybně umístěná mezera před tečkami.
...	Uvedení třech teček místo speciálního znaku, kde jsou tečky odděleny tenkými mezerami.
.␣.␣.␣. /...␣.	Může být zamýšlen význam třech teček a koncové tečky na konci věty, další tečka se ovšem nepíše.
.␣.␣.	Chybné zapsání normálních mezer mezi tečkami, místo celého výrazu by měl být použit speciální znak pro tři tečky.

Tabulka 2.4: Chybné výrazy s použitím třech teček

Dvojtečka

U dvojtečky probíhá kontrola stejně jako u jiných interpunkčních znamének. Dále jsou v kontrole zohledněny výrazy udávající časové údaje, na tato místa je uživatel upozorněn: použití dvojtečky je podle hladké sazby správné u oddělení minut a sekund, ale často se v textech používá také k vyjádření hodin a minut. Správně podle hladké sazby je zde umístěna tečka. Při matematických výrazech a chemickém poměru je sázena z obou stran s mezerami.

Chybný výraz	**Popis**
`␣:␣`	Chybně umístěná mezera před dvojtečkou (výjimky jsou chemický poměr a matematické výrazy).
`:`	Chybí mezera za dvojtečkou v textu (kromě časových údajů).
`12:54`	Časový údaj, kde ale automaticky nelze rozhodnout, zda se jedná o minuty a sekundy nebo hodiny a minuty.

Tabulka 2.5: Chybné výrazy s použitím dvojtečky

2.4.3 Závorky

V hladké sazbě se používají následující závorky:

- oblé (),
- hranaté [],
- svorky `{}`.

Vždy se píší těsně k výrazům nebo větám, které se do nich vkládají. Při návrhu kontroly jsem se rozhodla vyhledat celý výraz od otevírací závorky až po uzavírací závorku a jejich okolí. Podobně jako u interpunkčních znamének se kontroluje správné umístění mezer (těsně za vložený výraz). Pro vyhledání celého výrazu jsem se rozhodla pro případné další rozšíření skriptu a důkladnější kontrolu výrazu (např. zda za otevírací závorkou následují velké nebo malé písmeno a podle toho je umístěna i interpunkce u uzavírací závorky).

Chybný výraz	**Popis**
...(...)...	Vynechání mezer před otevírací závorkou a za uzavírací závorkou.
...␣(␣...␣)␣...	Chybně umístěné mezery za otevírací závorkou a před uzavírací závorkou.

Tabulka 2.6: Chybné výrazy s použitím závorek

2.4.4 Spojovník, pomlčka

Spojovník se používá k označení dělení slov nebo jako spojovací znaménko ve složených výrazech a sází se těsně. Pomlčka naznačuje větší přestávku v řeči nebo od sebe výrazně odděluje části textu, významově je rovna čárce. Uvádí se s okolními mezerami. Pomlčka se také používá jako náhrada výrazů a, až, od do, versus. Při tomto použití se sází bez mezer a nesmí být na začátku ani na konci řádku. Pomlčka za desetinnou čárkou u čísel vyjadřuje celé peněžní jednotky a v tomto případě se za čárkou sází těsně.

Při kontrole je možné skriptem vyhledat všechna umístění spojovníku nebo pomlček, ale není možné odhalit záměnu a nesprávné používání spojovníku a pomlčky. Na jednotlivá umístění při automatické kontrole není uživatel na tato místa upozorňován, ale je provena typografická kontrola a předpokládá se správné použití.

2.4.5 Podtržítko, lomítko

Oba znaky se sází bez mezer, vyhledání ve skriptu je navrženo pomocí regulárních výrazů a okamžité záměny bez dalších kontrolních funkcí.

2.4.6 Čísla

Zápis čísla

Číslo se celé zapisuje na jeden řádek, nelze jej rozdělit do dvou řádků. Případné mezery mezi čísly jsou nalezeny a zaměněny za úzké pevné mezery, tak, aby nedocházelo k rozdělení čísla na dva řádky. V nabídce vyhledávání je také nalezení čísel větších než tisíc, které se seskupují po třech.

Desetinná část čísla je oddělena čárkou, okolo které se nepíše mezera ani z jedné strany. Jednotlivé části skriptu jsou provázány a čárka u čísel je zhodnocena i ve funkci zaměřené na kontrolu všech vyhledaných čárek v textu.

Chybný výraz	**Popis**
`123456789`	Čísla nejsou rozdělena po řádech.
`123␣456␣789`	Umístění normální mezery mezi řády v číslech, kdy správně má být úzká pevná mezera.
`123␣456␣,␣789`	Desetinná čárka s mezerami kolem ní.

Tabulka 2.7: Chybné výrazy s použitím čísel

Časové údaje

V hladké sazbě se hodiny a minuty sazí oddělené tečkou. Celé hodiny je možné sázet ve formátu 12.00 nebo 12 hodin. Minuty a sekundy pak oddělujeme dvojtečkou, sekundy a desetiny sekund čárkou. Uživatel si bude moci z výběru vybrat pouze vyhledání časových údajů, pokud vyhledává i znaménka tečky, dvojtečky, čárky, pak je tato kontrola provedena již u těchto znamének.

Chybný výraz	**Popis**
`09:48`	Nesprávné vyjádření času s uvedením dvojtečky mezi hodinami a minutami. Toto dovoluje norma ČSN016910.

Tabulka 2.8: Chybné výrazy při zápisu času

2.5 Možnosti automatických oprav

Pro automatickou kontrolu dokumentu může uživatel využít nabídku *Hledat a nahradit*, kterou InDesign nabízí. Zde je možné vybírat ze několika záložek, pro kontrolu hladké sazby nejvíce použitelné *Text* a *GREP*. V záložce *Text* je možné vyhledávat také pomocí zástupných symbolů, které InDesign pro textovou části definuje. Záložka *GREP*, která je dostupná od verze Adobe InDesign CS3, nabízí možnost pracovat s regulárními výrazy. Výhodou je automatické prohledávání textu a rychlá záměna výrazů. Nevýhodou je pro uživatele alespoň částečná znalost použití regulárních výrazů a nemožnost

rozhodování u nejednoznačných situací, jako například interpunkce, jak je uváděno výše.

Pro operační systém Mac OS X je pro automatickou kontrolu dokumentu nabízen skript *Viktor Čistič*. Tento skript byl nabízen již pro AdobeInDesign CS1 a od té doby byl postupně rozšiřován a opravován. Uživateli nabízí vyčištění dokumentu od nadbytečných znaků, nevhodně umístěných mezer např. před znaky závorek, vykřičníků. V nabídce má nyní i kontrolu měrných jednotek či mezery u zkratek a titulů.

Pro operační systém Windows je možné využít skripty Petera Kahlera, které usnadňují práci s nástrojem *GREP* v InDesignu. Na dokument je pak při použití skriptu opět možné aplikovat řadu *GREP* dotazů. Ty se vykonají v pořadí, jak jsou načteny z dialogového okna skriptu, GREP dotazy musí uživatel definovat. Automatizace probíhá tedy bez rozhodování z více správných možností.

Dříve bylo možné zakoupit skript *UseMyTypo*, který nabízel automatickou kontrolu dokumentu, nyní jsem o něm aktuální informace již neobjevila.

Ve svém skriptu se snažím řešit kontrolu vybraných výrazů a znaků nejen automatickou záměnou za jiný výraz, ale také rozhodování v případě, že pro znak existuje více správných řešení, podle toho, jak vypadá text v okolí znaku. Skript pracuje s nástrojem *GREP* a uživatel jej může používat bez znalosti regulárních výrazů.

3 Praktická část

3.1 Popis projektu

Pro automatickou kontrolu typografie jsem se rozhodla využít skriptování, které Adobe InDesign podporuje od verze CS1. Sázecí program navíc podporuje více jazyků, konkrétně *AppleScript* (pro MAC OS), *VBScript* (pro Windows) a *JavaScript* (pro obě platformy). Pro svůj skript jsem si vybrala JavaScript, který je ve velké míře využíván při tvorbě webových aplikací a k dispozici je množství literatury a nástrojů. JavaScript určený pro InDesign má však oproti tomu, který je používán na webu, společnou pouze základní specifikaci, tedy základní syntaxi, programové konstrukty a některé objekty.

JavaScript pro InDesign je přizpůsoben práci v InDesignu, jsou v něm definovány jednotlivé objekty, se kterými InDesign pracuje. Více shrnuje kapitola 3.2. Skript pro automatickou kontrolu typografie jsem vytvořila pro verzi CS3 a vyšší.

Prohledávací skript obsahuje následující části:

- hlavní část skritpu, která spouští další funkce;
- část pro vyhledávání a náhrady v textu dle určeného rozsahu;
- část pro kontrolu vyhledaných výrazů a rozhodnutí o dalším zpracování;
- část s jednotlivými funkcemi, které jsou volány podle vyhledaných regulárních výrazů;
- část pro práci se soubory;
- část pro nastavení rozsahu a možností vyhledávání (podle výsledků nastavených uživatelem);
- část pro dialogová okna;
- část pro pomocné a doplňující funkce.

Kromě samotných skriptů jsou vytvořeny následující textové dokumenty pro podporu skriptů:

- `Config.txt` – konfigurační soubor pro nastavení možností skriptu;
- `Config2.txt` – konfigurační soubor pro nastavení možností skriptu při druhém průchodu dokumentem;
- `FindList.txt` – soubor, který definuje jednotlivé typografické chyby, které mají být vyhledány, pomocí regulárních výrazů;
- `Result[číslo prohledávané části].txt` – soubor s výsledkem prohledávání.

3.2 Skriptování v Adobe InDesign

3.2.1 Popdora skriptování

Jak již bylo v úvodu řečeno, skriptování je v Adobe InDesign umožněno od verze CS1. Skriptování v Adobe InDesign přináší řadu možností, která je s každou verzí InDesignu rozšiřována a více podporována. V jednotlivých verzích jsou nové funkce přidávány, zároveň jsou některé nahrazeny jinými a dochází tak k nekompatibilitě. V řadě případů jsou vytvořené skripty určené pouze pro určitou verzi InDesignu a pro novou verzi musí být aktualizovány.

3.2.2 Spouštění skriptů v InDesign

Skripty, které chceme používat, se ukládají do příslušného podadresáře. Cesta k adresáři od verze CS3 je následující:

- `Adobe InDesign CS3/Scripts/Scripts Panel`,

V samotném programu se skripty umístěné v uvedeném podadresáři zobrazí v paletce *Skripty* InDesignu. Paletka se v jednotlivých verzích aktivuje následujícím způsobem:

- verze CS3: *Okna – Automatizace – Skripty*,
- verze CS4: *Okna – Automatizace – Skripty*,
- verze CS5: *Okna – Pomůcky – Skripty*,

3.2.3 Práce s InDesignem

Pro práci a návrh skriptu bylo nutné seznámit se nejen s fungováním samotného *JavaScriptu*, ale také s jeho rozšířením pod InDesignem. *JavaScript* v InDesignu je navržen tak, že umí zacházet s *objekty* daného programu, jejich *vlastnostmi* a *metodami*. *Objekty* jsou zde například dokument, textový či obrazový rámeček, vrstva či okno dokumentu. *Vlastnosti* jednotlivých objektů jsou definovány podle typu objektu, například u rámečku šířka, výška, pozice na stránce nebo obsah. *Metody* jsou akce, které dokáže objekt provádět, tedy třeba změna velikosti, rotace či pozice rámečku. Tímto *JavaScript* pro InDesign dokáže zastoupit různé činnosti uživatele.

Z výše uvedeného je zřejmé, že se jedná o široké a specifické pole možností. Zaměříme se zde pouze na práci s textem, tedy s textovými rámečky a jejich obsahem.

3.2.4 Text, textové rámečky a práce s textem

Základním objektem je *story* (textový tok). Ten se skládá z kolekcí objektů *chars, words, lines, paragraphs, atd.* Přistupovat můžeme k textovému obsahu daného textového objektu. Automatizovat tak můžeme vkládání textu, nastavení parametrů textových rámečků nebo textových formátovacích charakteristik, kontroly přetečení rámečků a podobně.

Pomocí skriptů je možné realizovat různé automatické vyhledávání nebo záměny. Při skriptování však lze mimo samotného textového obsahu zpracovávat i rozmanité textové atributy jako např. barvu písma, označení částí textu tagy. Toho využívám také ve svém skriptu, kdy označuji vyhledané části textu, tak, aby je mohl uživatel identifikovat. Skriptování v InDesignu nabízí zajímavé možnosti, zároveň jsem se ale během práce setkala s různými úskalími. Například je omezena práce s vyhledaným textem, musela jsem najít několik oklik, jak k tomuto textu přistupovat a získat z něj informace pro další zhodnocení vyhledaného výrazu.

Jedním z problémů byla právě práce se textem. Vyhledané výrazy se ukládají do objektů typu text, který dále neumožňoval snadnou práci s ním. Pokud jsem ale vyhledaný text uložila do proměnné typu `String`, bylo možné pracovat s textem a regulárními výrazy dále. Problém nastal při načtení speciálních znaků, které má InDesign definované pod číselným označením.

Do proměnné typu `String` se speciální znaky nenačetly správně, ale právě ve svém číselném označení. Musela jsem hledat řešení, jak použít regulární výrazy v objektech typu text.

Skriptování v InDesign nabízí podporu regulárních výrazů od verze CS3.

3.3 InDesign a regulární výrazy

V *JavaScriptu* pod InDesignem jsou vytvořeny funkce umožňující vyhledávání a záměny pomocí regulárních výrazů. Na *story* a textové objekty nelze použít metody `search()`, `replace()`, `match()` a `split()`, které je naopak možné aplikovat na objekty `String`. V InDesignu s regulárními výrazy pracují funkce:

- `findGrep()` pro vyhledání výrazů,
- `changeGrep()` pro záměnu a nahrazení výrazu.

Výhodou použití funkcí vytvořenými přímo pod InDesignem jsou jejich rozšířené možnosti, kdy je možné vyhledávat i speciální znaky jako různé druhy mezer či pomčky a podobně. Pro tyto funkce mluví i využití i dalších voleb ve vyhledávání a nahrazení; vyhledaný text může mít různý styl nebo může být po nahrazení označen či změněny jeho vlastnosti.

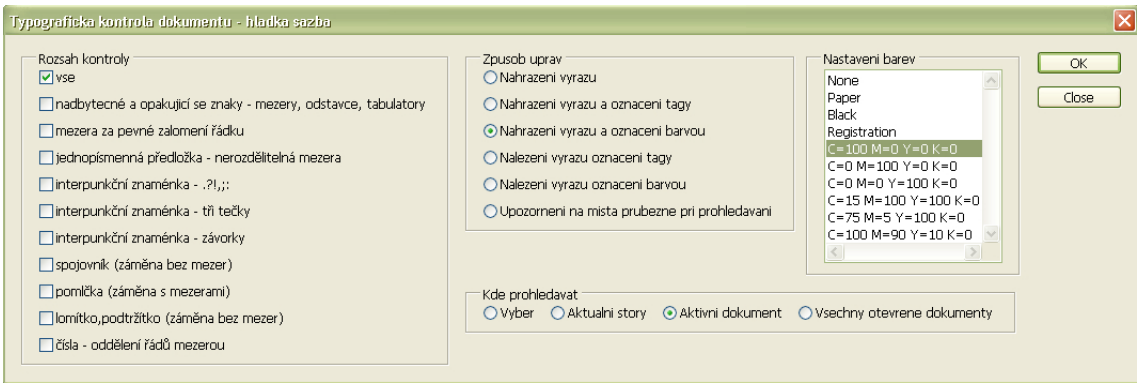
Nevýhodou ve skriptu je, že dokument bude prohledáván v cyklu podle podle počtu vybraných výrazů. V určitých situacích se může stát, že bude výraz nalezen ve více částech prohledávání a pokaždé vyhodnocen. Ve skriptu je snaha o omezení těchto situací, např. tři tečky je možné vyhodnotit v rámci interpunkce i samostatně zvlášť. V případě výběru obou těchto voleb jsou tři tečky vyhledány v pořadí první z nich a druhá je označena, že se prohledávat nebude.

3.4 Skript pro automatickou kontrolu typografie

3.4.1 Spuštění skriptu a načtení údajů

Po spuštění skriptu se uživateli otevře dialogové okno, jak ukazuje obrázek 3.1, s nabídkou možností, které skript nabízí. Možnosti jsou rozděleny do třech částí.

- Výběr z nabízeného rozsahu výrazů, které mají být vyhledány,
- způsob úprav, tedy jak má být nahrazení, případně jenom vyhledání provedeno a označeno,
- rozsah textu, na který se vyhledávání bude vztahovat.



Obrázek 3.1: Dialogové okno po spuštění skriptu

Při prvním spuštění dialogového okna se načte nastavení pro dialogové okno z konfiguračního souboru `Config.txt`. Nastavení dialogového okna, jak jej uživatel definoval, bude před ukončením uloženo zpět do konfiguračního dokumentu a při dalším spuštění skriptu bude použito při zobrazování dialogového okna. Kromě nastavení jednotlivých grafických prvků je součástí konfiguračního souboru i nastavení samotného spuštění dialogového okna. Pro uživatele, kteří budou ve skriptu používat stále stejné nastavení a jsou schopni zasahovat do konfiguračního souboru, je zde možnost vypnout zobrazování dialogového okna. Po spuštění skript načte konfigurační soubor a podle jeho nastavení začne rovnou provádět kontrolu dokumentu.

Pro vypnutí dialogového okna stačí změnit následující proměnnou na `false`:

```
var dialogShow = false;
```

Pokud by se uživateli povedlo omylem smazat konfigurační soubor `Config.txt`, po spuštění skript vytvoří nový a spustí podle něj dialogové okno.

3.4.2 Způsoby úprav textu

Uživateli jsou nabízeny také možnosti prohledávání a označení výsledku prohledání. Jednak mohou být vybrané výrazy vyhledány a nahrazeny, nebo pouze vyhledány a označeny.

První variantou je možnost vyhledat chyby v hladké sazbě a ty automaticky nahradit za správné výrazy bez dalšího označení vyhledaných míst.

Následující varianta nabízí označit nahrazená místa tagy, případně vybranou barvou. Pří výběru barvy je na konci prohledávání nabídnuta uživateli možnost nastavit skript na druhý průchod dokumentem, což blíže popisuje kapitola 3.4.4. Při označování jednotlivých vyhledaných částí barvou je potřeba vzhledem ke změně indexů jednotlivých znaků v textu po náhradě provádět nahrazování od konce dokumentu.

Další varianty nabízejí stejné označení (tagy, barva), ale text se pouze vyhledá a samotná náhrada je ponechána na rozhodnutí uživatele. Při označení barvou je na konci prohledání opět uživateli zobrazena nabídka druhého průchodu dokumentem.

Poslední možností je průběžné upozorňování na vyhledaná místa (v textu se označí výběr místa a otevře se dialogové okno s nabídkou náhrady, jak ukazuje obrázek 3.2). I při této možnosti je nutné (vzhledem ke změně indexů jednotlivých znaků při změně vyhledaného výrazu) prohledávat dokument od konce na začátek.

Barvu, kterou se bude označovat vyhledaný text, si vybírá uživatel ze seznamu v dialogovém okně. Seznam barev je načten ze vzorníku aplikace a liší se podle toho, jak má každý uživatel definované barvy ve vzorníku.

Obrázek 3.2: Dialogové okno pro nahrazení jednotlivých vyhledaných výrazů

3.4.3 Rozsah textu určeného ke kontrole

Po spuštění skript kontroluje, zda v otevřeném dokumentu je textový rámeček obsahující text, který bude možné prohledat. Pokud ne, je uživatel na tuto skutečnost upozorňen a skript je ukončen. V nabídce jednotlivých voleb je možné nastavit, v jak velké části textu se bude prohledávat. Uživatel může volit mezi následujícími možnostmi:

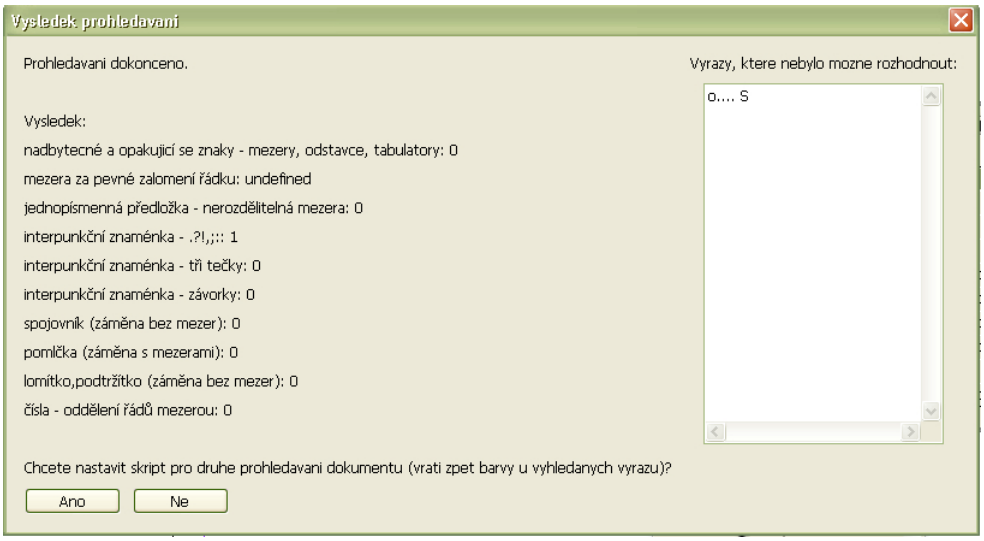
- **Výběrem** – označená část textu, pokud nebyl výběr označen a je vybrána tato možnost, skript na toto uživatele upozorní a ukončí se.
- **Aktuální story** – InDesign nabízí možnost rozdělit texty do jednotlivých story a je možné prohledat pouze určitou story, pokud není v dokumentu žádná story, opět na toto skript upozorní.
- **Aktuálním dokumentem** – prohledají se všechny story v aktuálním dokumentu.
- **Ve všech otevřených dokumentech** – dokumenty jsou postupně aktivovány a následně prohledány.

3.4.4 Vybrání výrazů pro prohledání

Součástí dialogového okna je samozřejmě nabídka pro výběr výrazů, které se mají prohledat. Pro správné zobrazení všech nabízených výrazů se načítá konfigurační soubor regulárních výrazů `Findlist.txt`. Jak dále po načtení souboru probíhá prohledávání ve skriptu popisuje kapitola 3.5.

Druhý průchod prohledávání

Pokud uživatel v úvodním nabídkovém okně zvolí možnost barevného označení vyhledaných míst, je mu po ukončení hledání nabídnuta možnost druhého průchodu dokumentem (obázek 3.3). Znamená to, že při dalším spuštní skriptu budou vyznačené části textu obarveny opět původní barvou. Informace potřebné pro tento „návrat“ jsou uloženy v dokumentu `Result[číslo prohledávané části].txt`, kam se uloží vyhledané a nahrazené výrazy.



Obrázek 3.3: Dialogové okno s nabídkou nastavení druhého průchodu při obarvování textu

3.5 Prohledávání dokumentu skriptem

3.5.1 Ukládání dat

Skriptování pod InDesignem mi neumožňuje vytvořit si datový objekt, do kterého by se ukládaly všechny potřebné informace k jednotlivým regulárním výrazům, jako samotný výraz, jeho nahrazení, jeho příznak a index a další. Tuto „nevýhodu“ řeším pomocí několika polí, do kterých se výše uvedené informace ukládají. Veškeré informace související s jedním výrazem jsou pak uloženy v pod jedním indexem a je možné k nim přes tento index přistupovat.

3.5.2 Způsob prohledávání

Při návrhu skriptu a testování možností, jsem se rozhodla využít definované prohledávací funkce, které nabízí skriptování pod InDesignem. Jejich výhodou je přímá podpora InDesignu a nejrůznější možnosti nastavení preferencí prohledávání. Nevýhodou je, že dokument je prohledáván několikrát v cyklu podle počtu regulárních výrazů, které si uživatel vybral.

Využití preferencí, tedy vlastností souvisejících s prohledáváním, umožňuje nejenom nastavení toho, co se má prohledávat nebo nahrazovat, ale bylo výhodné použít jej i při označování tagy nebo barvou.

Funkce `findGrep()` a `changeGrep()` se aplikují na textovou oblast (výběr, story, dokument), kterou podle nastavení dialogu vyhodnotí funkce `myFindPlace_CS3()`. Ta dále volá funkci `myFindAndReplaceInText_CS3(mySearchSelection)` a jako argument má právě odkaz na vybraný text.

3.5.3 Vyhledávání výrazů

Pomocí preferencí se nastaví prohledávání, kdy se postupně v cyklu zadávají regulární výrazy načtené z konfiguračního souboru `FindList.txt`. Prohledaná místa jsou uložena do pole a podle výběru způsobu úprav textu jsou volány další funkce.

```
app.findGrepPreferences.findWhat = myExp;
myFoundText = mySearchSelection.findGrep();
```

3.5.4 Způsoby kontroly typografie

V konfiguračním souboru `FindList.txt` jsou jednotlivé regulární výrazy označeny příznakem pro zpracování.

- Příznak `r` – označuje výrazy, které po vyhledání a zavolání příslušné funkce pro záměnu jsou ihned nahrazeny.
- Příznak `f` – označuje výrazy, které jsou po uložení do pole postupně dále procházeny pomocí definovaných funkcí (podle výběru, co se má

prohledat). V nich se rozhodne, zda je vyhledaný výraz správně a nebude se zaměňovat a nebo na jakém místě je ve výrazu chyba. Podle typu chyby se nastaví vhodná náhrada.

Regulární výrazy s příznakem `r` jsou použity na chybové části jako nadbytečné znaky nebo záměna mezery za jednopísmennou předložkou.

Pomocí regulárních výrazů s příznakem `f` jsou vyhledány všechny výskyty znaků v dokumentu, např. interpunkčních znamének. Kontroly ve funkci je sledují typografické chyby, pokud nejsou nalezeny, je výraz považován za správný. Jednotlivé funkce jsou definovány podle pravidel hladké sazby a vychází z tabulek uvedených v kapitole 2.4.

Kontrolní funkce výrazů

Pro znaky, jako například interpunkční znaménka, není možné kontrolu řešit jako vyhledání všech výskytů a jejich náhradu jedním jiným výrazem. Náhrada naopak musí vycházet z výběru různých variant řešení. Při analýze pravidel hladké sazby jsem si definovala znaky, které se mají nebo naopak nemají vyskytovat v okolí hledaného znaku. Dále vše vysvětlím na příkladech interpunkčních znamének.

Při vyhledávání znaku je společně se znaménkem načteno i jeho okolí, tedy bílé znaky a z každé strany jeden znak textu. Regulární výrazy v konfiguračním souboru `FindList.txt` jsou navrženy zdánlivě složitěji než by mohly být, ale zvolila jsem tuto variantu právě z důvodu zachování nebo odstranění jednotlivých částí nalezeného výrazu.

U interpunkčních znamének je neprve, pokud nebyla tato volba zvolena uživatelem ve výběru, kontrolována přítomnost více bílých znaků za sebou a ponechán je jeden (namísto dvou mezer jedna). Je zde také ošetřen případ, kdy se za znaménkem může vyskytovat mezera a za ní konec odstavce (případně více konců odstavců). V tomto případě má být zachován až znak konce odstavce a ne mezery hned za znaménkem.

Dále je sledováno zda se ve výrazu vyskytuje jedno interpunkční znaménko nebo více a je přesně určeno, o jaký znak se jedná. Podle toho jsou volány funkce, které sledují již každé interpunkční znaméneko zvlášť. Samostatná funkce řeší výskyt více interpunkčních znamének.

V těchto funkcích určených pro znaménka je opět používána metoda v InDesignu `findGrep()` s nastavenými preferencemi. Metoda `findGrep()` je tentokráte aplikována pouze na výraz nalezený v textu a v něm hledá možné kombinace výskytu v souvislosti se znaménkem.

Nejprve jsou ošetřeny případy, kdy se v okolí interpunkce objevuje číslo (případ čárky, tečky, dvojtečky) a funkce je ukončena s tím, že výraz je správně nebo je opraven a nastaven příznak pro nahrazení výrazu. V souvislosti s čísly jsou dále sledovány případy, kdy za číslem čárka zůstat má, jak ukazuje příklad v kapitole 2.4.2. Pro tuto kontrolu slouží samotstatná funkce, která je volána v případě čárky a vkládání nerozdělitelných mezer u čísel v řádech tisíců.

Pokud nejsou v okolí znaménka čísla, pak funkce pokračuje další kontrolou a sledují se jednotlivé definované případy jako:

- chybějící mezera za interpunkcí, pokud za znakem není umístěno další znaménko nebo uzavírací závorky,
- nadbytečná mezera za interpunkcí, pokud za znaménkem následuje další znaménko, uzavírací závorky,
- nadbytečná mezera před znaménkem ve všech případech.

Pro každý znak jsou definovány vlasní kontrolní funkce ošetřující situace, které mohou nastat, jak bylo popsáno výše. Kromě toho, jsou ve funkcích nastaveny případy, které není možné rozhodnout, jejich příznak je nastaven tak, aby nedošlo k záměně. Tyto výrazy jsou v závěrečném shrnutí uvedeny v přehledu a vždy vyznačeny barvou ve skriptu.

Pokud výraz není v žádné části funkce zaměněn, je považován za správný a k náhradě nedojde. To určuje číselný příznak, který se při nalezení chyby zvyšuje. Do nahrazovací funkce `changeGrep()` jsou pak zařazeny pouze výrazy, u kterých se příznak změnil.

3.5.5 Nahrazení výrazů

Při nahrazovanání určených výrazů (všech u příznaku `r` a vybraných chybových u příznaku `f`) využívám opět preferencí. Kromě nastavení regulárního

výrazu pro nahrazení se zde nastavují tagy nebo barva, kterou uživatel vybral ze vzorníku. Pokud mají být výrazy nalezeny a označeny bez změny samotného výrazu, je v preferencích záměny uveden prázdný výraz.

Jednotlivé preference vypadají následovně:

```
app.changeGrepPreferences.changeTo = myNewText;
app.changeGrepPreferences.changeTo = "";

app.changeGrepPreferences.markupTag = "Zmeneny_text";
app.changeGrepPreferences.markupTag = "Nalezeny_text";
app.changeGrepPreferences.fillColor =
String(app.activeDocument.swatches[colorIndex].name);
```

Samotná funkce `changeGrep()` je volána buď na celou textovou část (podle rozsahu prohledávaného textu) nebo na jednotlivé položky pole podle vyhodnocení funkcí.

```
myChangedText = myFoundText[y].changeGrep();
myChangedText = mySearchSelection.changeGrep();
```

3.6 Ukončení skriptu

Po skončení skriptu je uživateli zobrazeno dialogové okno s výsledkem průchodu dokumentem, vizte obrázek 3.3. Jsou zde shrnuty počty zaměněných výrazů v jednotlivých skupinách a dále je v dialogu uvedena tabulka s výrazy, které skript nalezl, ale nemohl rozhodnout o jejich správnosti (např. hodiny a minuty).

3.7 Rozšíření skriptu

Skript pro automatickou kontrolu ošetřuje nejčastější a základní typografické chyby, jak byly jmenovány výše. Zároveň nabízí uživateli možnost doplnit si skript o další regulární výrazy s příznakem `r`. Regulární výrazy stačí zapsat do konfiguračního souboru `FindList.txt`, stejně jako jsou tam zapsány stávající regulární výrazy.

U této verze skriptu je potřeba po doplnění regulárních výrazů ošetřit jednu skutečnost, aby skript neskončil chybovým hlášením. Po uložení změn v souboru `FindList.txt` musí být ve stejném adresáři smazán konfigurační soubor `Config.txt`. Po opětovném spuštění skriptu je vytvořen nový konfigurační soubor, ve kterém jsou již zaznamenány údaje k nově doplněným částem prohledávání. Toto bylo ve skriptu zohledněno i vzhledem k tomu, že by uživatel soubor `Config.txt` smazal. V dalším zpracování skriptu by bylo přínosné tuto situaci ošetřit v rámci samotného skriptu.

Skript je navržen tak, aby bylo možné jej dále rozšířit úpravou uvnitř skriptu. Stačí doplnit regulární výrazy s příznakem `f` do konfiguračního souboru a k nim příslušné ošetřující funkce přímo do skriptu. Jednotlivé ošetřující funkce jsou číslovány a číslo odpovídá číslu části v konfiguračním souboru. Nová ošetřující funkce musí mít název `myControl'číslo části'()` a použijí se parametry, které jsou stejné pro všechny kontrolní funkce.

Pro praktické použití by bylo vhodné dialogové volby rozšířit o nabídku vytvoření kopie prohledáváného dokumentu a samotné prohledávání provádět v této kopii. Dále by bylo možné nabídnout uživateli výběr označovací barvy pro výrazy, které skript neumí rozhodnout. V tuto chvíli je ve skriptu pro tyto situace barva pevně nastavena.

Pro příjemnější uživatelský přístup při dopňování vlastních regulárních výrazů (v režimu nalezení a automatické náhrady) by bylo užitečné doplnit další skript, který by nabízel dialogové okno, kam by uživatel mohl své regulární výrazy zadat. Skript by je pak ve správném formátu a řazení zapsal do konfiguračního souboru `FindList.txt`.

4 Závěr

V průběhu realizace projektu jsem postupně musela řešit řadu dílčích úkolů, které znesnadňovaly tvorbu skriptu. Různé komplikace vyvstávaly při používání funkcí *JavaScriptu* pod InDesignem. Při testech jsem narazila na to, že původní myšlenku, jak kontrolovat jednotlivé výrazy ve funkcích, musím pozměnit, například kvůli používání speciálních znaků v InDesignu. S tím se objevila řada dalších úkolů, které bylo nutné v rámci inovovaného návrhu kontroly provést. Nakonec se mi podařilo vytvořit funkční skript, který nejenom automaticky zaměnuje vyhledané výrazy, ale podle potřeby je analyzuje a vybírá vhodné řešení pro konkrétní situace.

Při snaze nabídnout uživateli možnosti výběru, jak má skript s dokumentem pracovat, jsem musela řešit situace, jak propojovat fungování samotného vyhledávání s ostatní nabídkou. Také samotné kontrolní funkce ve vyhledávání s sebou přinesly řadu úkolů k řešení.

Kromě seznámení se samotným skriptováním v *JavaScriptu* jsem si prostudovala práci v InDesignu. Zajímavé pro mě bylo prozkoumat, jakým způsobem instrukce skritpu zacházejí s objekty v InDesignu, s jejich vlastnostmi a metodami. Dále jsem si vyzkoušela práci s regulárními výrazy, navíc doplněnou o používání regulárních výrazů ve funkcích vypracovaných přímo pro InDesign.

Literatura

[1] Kočička, Pavel – Blažek, Filip. *Praktická typografie*. Brno: Computer Press, 2004.

[2] Suehring, Steve. *JavaScript: krok za krokem*. Brno: Computer Press, 2004.

[3] Adobe System Incorporated. *InDesign CS Scripting Guide*, California: Adobe Systems Incorporated, 2003.

[4] Adobe System Incorporated. *Adobe InDesign CS3 Scripting Guide: JavaScript*. California: Adobe Systems Incorporated, 2007.

[5] *InDesign Scripting Reference*. [online] Dostupné z: http://www.indesignscriptingreference.com/CS3/JavaScript/

[6] Krejčí, Richard. Skriptování InDesingu. Richard Krejčí, 2006. [online] Dostupné z: http://www.grafika.cz/art/sazba/skriptid1.html

[7] Krejčí, Richard. *Adobe InDesign CS3 zblízka: GREP*. Richard Krejčí, 2007. [online] Dostupné z: http://www.grafika.cz/art/sazba/idcs3grep.html

[8] Satrapa, Pavel. *Regulární výrazy*. Vytvořeno pro on-line magazín root. Pavel Satrapa, 2000. [online] Dostupné z: http://www.kit.tul.cz/ satrapa/docs/regvyr/

[9] Pecka, Miroslav. *Regulární výrazy a JavaScript*. Miroslav Pecka, 2005. [online] Dostupné z: http://interval.cz/clanky/regularni-vyrazy-a-javascript-uvod/

Seznam tabulek

2.1 Chybné výrazy s použitím tečky 7
2.2 Chybné výrazy s použitím otazníku a vykřičníku 8
2.3 Chybné výrazy s použitím čárky 9
2.4 Chybné výrazy s použitím třech teček 9
2.5 Chybné výrazy s použitím dvojtečky 10
2.6 Chybné výrazy s použitím závorek 11
2.7 Chybné výrazy s použitím čísel 12
2.8 Chybné výrazy při zápisu času 12

Seznam obrázků

3.1 Dialogové okno po spuštění skriptu 18

3.2 Dialogové okno pro nahrazení jednotlivých vyhledaných výrazů 20

3.3 Dialogové okno s nabídkou nastavení druhého průchodu při obarvování textu 21

Přílohy

Příloha A. Uživatelský manuál
Příloha B. Přiložené CD

A Uživatelský manuál

Skript `MyTypo_CS3` je určen pro automatickou kontrolu typografie podle pravidel hladké sazby v programu Adobe InDesign. Skript je vytvořen pro verzi CS3 a vyšší.

Uložení skriptu

Celou složku se skriptem i konfiguračními soubory je potřeba uložit do složky, kterou InDesign používá pro práci se skripty. Cesta ke složce je následující:

- **Mac OS** Users/[jméno uživatele]/Library/Preferences/Adobe InDesign/-[verze]/([jazyk])/Scripts/Scripts Panel,
- **Windows XP** Documents and Settings\[jméno uživatele]\Application-Data\Adobe\InDesign\[verze]\([jazyk]\)Scripts\Scripts Panel,
- **Windows Vista a Windows 7** Users\[jméno uživatele]\AppData\-Roaming\Adobe\InDesign\[verze]\([jazyk]\)Scripts\Scripts Panel.

Spuštění skriptu

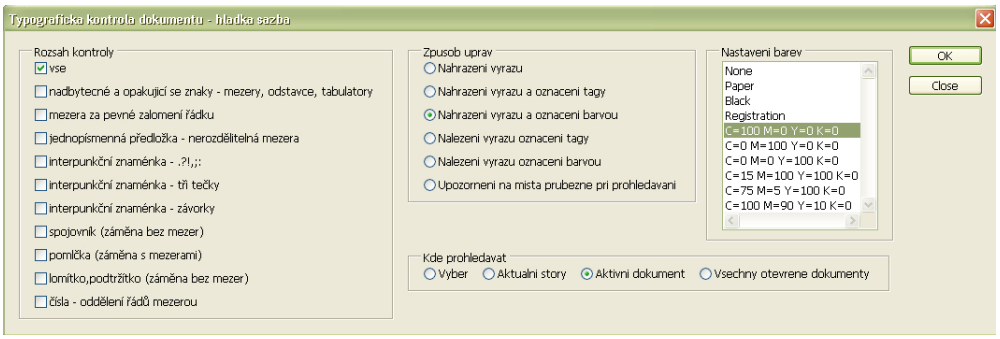
V InDesignu se skripty zobrazí v paletce *Skripty* InDesignu. Panel s nabídkou dostupných skriptů je možné v různých verzích aktivovat následujícím způsobem:

- verze CS3: *Okna – Automatizace – Skripty*,
- verze CS4: *Okna – Automatizace – Skripty*,
- verze CS5: *Okna – Pomůcky – Skripty*,

Samotný skript se spouští souborem `MyTypo_CS3`.

Dialogové okno skriptu

Po spuštění se zobrazí dialogové okno skriptu s nabídkou výběru.



Obr..: Dialogové okno skriptu

Upozornění: Vaše nastavení dialogového okna bude po spuštění skriptu zapamatováno a uloženo. Při dalším spuštění skriptu je v dialogovém okně zobrazeno vaše poslední nastavení.

Rozsah kontroly

V části rozsahu kontroly můžete vybírat z jednotlivých nabídek jaké typografické problémy mají být vyhledány. Vybírat můžete jak jednotlivé části, tak je zde umístěna nabídka pro prohledání všech případů.

Způsob úprav

Skript pro kontrolu typografie vám umožňuje vyhledané výrazy označovat různým způsobem.

Výrazy je možné automaticky nahradit a neoznačovat je žádným způsobem.

Vyhledané výrazy skript může nahradit a označit tagem s názvem `Zmeneny_text`.

Pokud vyberete nahrazení výrazu a označení barvou, aktivuje se vám panel s výběrem barvy podle vašeho vzorníku barev.

Kromě automatického nahrazování výrazů je možné problematická místa textu nechat skript pouze vyhledat. Opět je možné použít označení pomocí tagu s názvem `Nelezeny_text`.

I v případě pouhého vyhledání výrazů je můžete označit barvou podle vašeho výběru ze vzorníku.

Jako poslední vám skript nabízí vyhledávání výrazů a průběžné upozorňování na nalezená místa. Při upozornění je vyhledané místo označeno výběrem a zobrazeno je dialogové okno s návhrem, jak má být výraz nahrazen. Po potvrtení je záměna provedena, při stornování zůstane výraz původní.

Kde prohledávat

I u míst prohledávání máte možnost si vybrat z několika nabídek. Ve všech případech musí otevřený dokument obsahovat text, jinak se skript ukončí. Při volbě *Výběr* je potřeba mít vybranou a označenou část textu pro prohledávání. Všechny otevřené dokumenty jsou prohledávány postupně.

Rozšířená práce se skriptem

Skript vám nabízí možnost rozšířit prohledávání o další regulární výrazy, které slouží k prohledávání textu. Všechny regulární výrazy jsou uložené v samostatném souboru s názvem `FindList.txt`. Tento soubor spolu s ostatními naleznete ve složce skriptu (MyTypo).

Otevřete soubor `FindList.txt`.

Na konec souboru umístěte své nové regulární výrazy podle následujících pravidel:

- Na první řádek zapište číslo části s popisem, který se objeví v zaškrtávací nabídce v dialogovém okně po spuštění skriptu v následujícím tvaru:
  - `cast`*`mezera`*`[číslo regulárního výrazu]`*`tabulator`*`[popis čeho se regulární výraz týká]`

- Na nový řádek zapište samotný regulární výraz a výraz, za který bude nahrazen podle následujích pravidel:

  - jako první musí být označení výrazu písmenem r (nesmí být označení f, skript by potom nefungoval)
  - za tabulátorem je v uvozovkách uveden text nebo regulární výraz pro prohledání
  - za dalším tabulátorem je opět v uvozovkách uveden text nebo regulární výraz, kterým má být vyhledaná část nahrazena
  - opět umístit tabulátor a napsat si popis, o čem regulární výraz je

Příklad:
cast *xx* záměna pozdravu
r "Ahoj" "Dobrý den" popis výrazů

Pozn: číslo regulárního výrazu nastavte jako další následující číslo, které je v dokumentu uvedeno.

Pozor, v této verzi skriptu je potřeba po doplnění nových regulárních výrazů do souboru `FindList.txt` smazat konfigurační soubor `Config.txt`. Ten bude po spuštění skriptu znovu vytvořen a budou v něm zohledněny nově vložené výrazy.

B Přiložené CD

Obsah přiloženého CD:

- Zdrojový kód skriptu včetně potřebných konfiguračních dokumentů,
- text bakalářské práce ve formátu `pdf`.