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

Bakalářská práce

Generátor a parser formulářů recenzí příspěvků na konferenci TSD

Plzeň 2019 Vojtěch Danišík

Místo této strany bude zadání práce.

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.
V Bakalářské práci jsou použity názvy programových produktů, firem apod., které mohou být ochrannými známkami nebo registrovanými ochrannými známkami příslušných vlastníků.

V Plzni dne 20. března 2019

Vojtěch Danišík

Poděkování

Děkuji panu Ing. Kamilu Ekšteinovi Ph.D. za ochotu při vedení bakalářské práce a rady s jejím vypracováním.

Abstract

Generator and Parser of Submission Review Forms for the TSD Conference. The goal of this thesis is to create PHP module, which will be easily integrate into existing information system for managing TSD conference. First part of the thesis explains standard PDF format and forms created in PDF. Subsequently, there are described existing PHP libraries for generating and parsing PDF form of scientific contribution. Second part of the thesis focuses on the implementation of selected libraries into TSD conference web portal. The module was tested by conference system users and multiple PDF browsers were used. Test results are part of this thesis.

Abstrakt

Cílem bakalářské práce je vytvořit jednoduše integrovatelný PHP modul do již existujícího informačního systému pro správu konference TSD. První část práce důkladně vysvětluje standardní formát PDF a formuláře vytvořené v PDF. Následně jsou popsány existující PHP knihovny pro generování a zpracování PDF formuláře daného vědeckého příspěvku. Druhá část práce se věnuje implementaci vybraných knihoven do webového portálu konference TSD. Modul byl otestován uživateli konferenčního systému a bylo použito více PDF prohlížečů. Výsledky testování jsou součástí této práce.

Obsah

1 Úvod

TSD (**T**ext, **S**peech and **D**ialogue) je konference zabývající se problémy zpracování přirozeného jazyka. Mezi nejčastěji probíraná témata se řadí: rozpoznávání řeči, modelování řeči,textové korpusy, značkování textu a mnoho dalších. Konference se koná každý rok v září a místo konání se střídá mezi Brnem (pořadatelem je Fakulta informatiky Masarykovy Univerzity) a Plzní (pořadatelem je Fakulta aplikovaných věd Západočeské univerzity v Plzni). Tento rok bude konference organizována právě Západočeskou univerzitou, a poprvé se bude konat za hranicemi České Republiky, přesněji na Slovinsku ve městě Ljubljana.

Ke každé konferenci existuje webový portál vytvořený daným pořadatelem, na nějž jsou od uživatelů nahrávány vědecké příspěvky. Tyto příspěvky jsou poté hodnoceny recenzenty (převážně členy programového výboru) formou online formuláře a na základě konečného hodnocení jednotlivých parametrů a na doporučení recenzentů jsou tyto příspěvky schváleny organizátorem a mohou být prezentovány na konferenci, nebo jsou zamítnuty z důvodu nedostatečného hodnocení. Modul vytvářený autorem bude implementován do webového portálu organizovaný Fakultou aplikovaných věd.

Cílem této práce je prostudovat strukturu PDF formátu, který je pro vytváření editovacích formulářů nejvhodnější a byl vybrán zadávajícím jako standard, tak i funkcionalitu volně dostupných PHP knihoven pro generování a parsování PDF souborů obsahujících editovatelný formulář, aby existovala možnost ohodnocení daného vědeckého příspěvku i v místech, kde není dostupné internetové připojení, neboli off-line. Tento PDF soubor musí obsahovat hodnotící formulář se všemi hodnotícími parametry, text vědeckého příspěvku doplněný o vodoznak. Pro generování a parsování musí být použity výhradně knihovny v jazyce PHP, jelikož není vhodné využívat aplikace třetích stran spustitelné z terminálu. Modul musí být nezávislý na platformě a lze ho upravovat v jakémkoliv PDF prohlížeči nezávisle na verzi PDF. Před vytvořením modulu na testovací verzi webového portálu bude potřeba projít zdrojové soubory webového portálu pro seznámení s již existujícími funkcionalitami a zařadit do portálu i náš modul. Z dřívějších let je zde naimplementován totožný modul pro generování a parsování PDF souborů, bohužel tento modul nesplňuje veškeré body zadání právě z důvodu použití nevhodného parseru.

2 Formát PDF

Formát **PDF** (**P**ortable **D**ocument **F**ormat) je souborový formát vyvinutý společností Adobe v roce 1992. PDF formát byl vyvinut za účelem konzistentní prezentace dokumentů (spustitelné na více zařízeních a různých platformách). Díky konzistenci lze dosáhnout toho, že PDF soubor vytvořený a uložený v systému Windows bude zobrazen totožně na systémech Mac, na všech distribucích Linuxu nezávisle na použitém PDF prohlížeči (Adobe Reader, Foxit a další).

V PDF souboru lze uchovávat velice širokou škálu dat, včetně formátovaného textu, vektorové grafiky a rastrových obrazů, nebo například informace o rozložení, velikosti a tvaru stránky. Informace definující umístění jednotlivých položek (jsou zde zahrnuty i editovací objekty pro formuláře) na stránce jsou zde uloženy též. Do dokumentu lze ukládat i metadata. Metadata jsou informace uložené v hlavičce souboru a lze do nich uložit název dokumentu, autora dokumentu, předmět a klíčová slova. Je zde možnost uložit heslo, aby byl dokument přístupný pouze autorizovaným uživatelům. Všechny tyto informace jsou uloženy ve standardním formátu [5, 9].

2.1 Objekty

PDF Objekty jsou základním stavebním kamenem pro uchovávání dat v dokumentu. Množinou PDF objektů lze reprezentovat bitmapové a vektorové objekty, barevné prostory, text, fonty aj. [10].

2.1.1 Základní objekty

V PDF můžeme najít celkem 5 základních objektů:

- **Celá a reálná čísla** - Celá čísla jsou reprezentována jako jedno nebo více desetinných čísel z rozsahu 0..9 se znaménkem + nebo - před číslem. Reálné číslo je celé číslo rozšířené o desetinnou část s ideálně jedním desetinným číslem (reálná čísla nelze popsat exponenciálním způsobem). Přesnost a rozsah celých a reálných čísel je definován jednotlivými implementacemi PDF. V některých implementacích platí pravidlo které přetypuje celé číslo na reálné po přesáhnutí předem daného rozsahu.

- **Řetězce** - Řetězec je reprezentován jako množina po sobě jdoucích bytů vepsaných mezi jednoduché závorky. Jako příklad lze uvést: *(Hello, World!)*. Pro zobrazení zpětného lomítka a jednoduchých závorek je potřeba před tyto znaky přidat zpětné lomítko pro jejich správné zobrazení v dokumentu. V tabulce 2.1 lze vidět využití zpětného lomítka pro zobrazení odřádkovacích znaků:

Sekvence znaků	**Význam**
$\backslash n$	*Line feed (LF)*
$\backslash r$	*Carriage return (CR)*
$\backslash t$	*Tab*
$\backslash b$	Backspace

Tabulka 2.1: Odřádkovací sekvence znaků

  Řetězce můžou být reprezentovány i jako sekvence hexadecimálních čísel vložených mezi znaky $<$ a $>$.
  Jako příklad lze uvést: *<4F6EFF00> →0x4F, 0x6E, 0xFF, 0x00.*

- **Jména** - Jméno je reprezentováno jako sloučení zpětného lomítka a řetězce (př. */Jmeno*). Za jméno se pokládá i zpětné lomítko bez řetězce. Pokud bychom potřebovali nadefinovat v dokumentu jméno, jenž bude obsahovat mezery, musíme do řetězce přidat i sekvenci znaků **#20**, jelikož v ASCII tabulce je hexadecimální hodnota 20 vyjádřena jako prázdný znak. Jména jsou case-sensitive, proto */Jmeno* a */jmeno* jsou rozdílná jména. Jeho využití v PDF je prosté, slouží jako klíče ve slovnících a pro definice složitějších (vícehodnotových) objektů.

- **Boolean (pravdivostní) hodnoty** - Logické hodnoty **true**/**false** a vyskytuje se v jednotlivých záznamech ve slovníku jako příznak.

- **Hodnota null** - Nabývá hodnot *f* (free) nebo *n* (use) a vyjadřuje, zda je objekt vyobrazen v dokumentu.

2.1.2 Složené objekty

Složený objekt je takový objekt, který obsahuje seřazenou/neseřazenou množinu základních objektů i množinu složených objektů.

- **Pole** - Pole je v PDF reprezentováno jako seřazená množina základních i složených PDF objektů (v poli může být uložen například i slovník nebo pole) nezávisle na typech (v poli lze uchovávat například řetězec a číslo zároveň). Hodnoty pole jsou vloženy mezi znaky [a].

- **Slovníky** - Slovník se skládá z množiny dvou hodnot: klíče a hodnoty, pomocí kterých se slovník namapuje. Klíč je reprezentován pomocí **jména**, zatímco hodnota může být kterýkoliv PDF objekt, povoleny jsou i slovníky nebo pole. Slovníky jsou uloženy mezi znaky « a ».
- **Datové proudy** - Datové proudy slouží především pro uložení binárních dat a skoro ve všech případech jsou zkomprimovány různými kombinacemi algoritmů, které jsou popsány v kapitole 2.2, proto datové proudy musí být zároveň i nepřímým odkazem. Skládají se ze slovníků a části binárních dat. Slovník je využit pro ukládání parametrů binárních dat, jako například délka binárních dat aj.

2.1.3 Linkovací objekty

PDF objekty můžou být různě velké. Pokud je objekt až příliš veliký, pak jsou v kódu dokumentu využity nepřímé odkazy. Na obrázku 2.1 si lze všimnout využití nepřímých odkazů ve slovníku.

```
<<
/Resources 10 0 R <--- znak R reprezentuje nepřímý odkaz na objekt s ID 10 a gen. číslem 0
/Contents [4 0 R]
>>
```

Obrázek 2.1: Ukázka nepřímého odkazu

2.2 Komprese dat v PDF

PDF soubory moho být poměrně kompaktní, o mnoho menší než ekvivalentní postscriptové soubory. Tato vlastnost je dosažena nejen lepší strukturou dat, ale i díky kompresním algoritmům, které jsou velice efektivní. Typ komprese dat PDF souboru lze zjistit pomocí textového editoru, který dokáže zpracovat binární data, vyhledáním klíčového slova **/Filter**. Níže jsou popsány kompresní algoritmy využívané v PDF [3].

- **CCITT G3/G4** - Algoritmus je bezeztrátový a využívá se pro vykreslení černobílých obrázků.
- **JPEG** - JPEG algoritmus může být jak ztrátový, tak i bezeztrátový. V Acrobatu se využívá pouze ztrátový s 5 stupni komprese. Využívá se pro barevné a šedotónové obrázky.

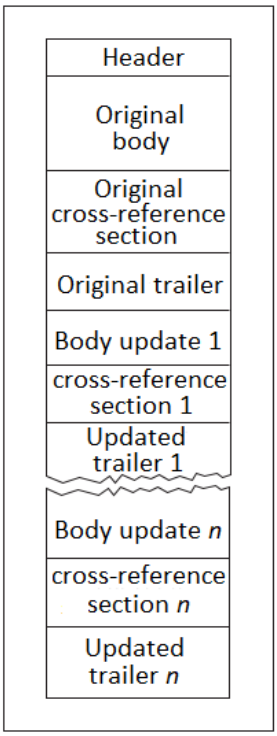
- **JPEG2000** - Rychlejší algoritmus na bázi JPEGu. Víceméně se nepoužívá, jelikož není kompatibilní se staršímy systémy a vysokými nároky na procesor.
- **Flate** - Bezeztrátový algoritmus, vychází z kompresních algoritmů LZ77 a Huffmanova kódování.
- **JBIG2** - Alternativní k CCITT. V Dnešní době se nevyužívá z důvodu pomalejší komprese než je u jeho protějšku.
- **LZW** - Komprimací LZW algoritmem lze dosáhnout až o polovinu menší velikosti díky komprimaci veškerého textu a operátorů v souboru.
- **RLE** - Bezeztrátový algoritmus pro vykreslování černobílých obrázků. Nahrazen efektivnějším algoritmem CCITT.
- **ZIP** - Bezeztrátový algoritmus, učinější než jeho protějšek LZW.

2.3 Vnitřní struktura PDF

Vnitřní reprezentace PDF souboru je rozdělena na sekce, které jsou znázorněny na obrázku 2.2.

Z obrázku lze vyčíst, že se zde vyskytují 4 hlavní sekce: *Header, Body, Cross-reference a trailer*. Díky jedné z vlastností PDF formátu se při úpravě souboru staré sekce neodstraní, místo toho se pouze na jeho konci vytvoří nové sekce [7].

- **Header** - Hlavička souboru je uložena na první řádce, obsahující primárně použitou verzi PDF.
- **Body** - V těle dokumentu jsou uložena veškerá data objektů reprezentující celý dokument. Objekty jsou referencovány v tabulce Cross-reference z důvodu rozprostření částí dat patřících k danému objektu po celé sekci.Pokud se v dokumentu vyskytuje jeden obrázek/zvukový záznam vícekrát než jednou, tak se poté všechny objekty reprezentující obrázky odkazují na jednu množinu dat [6].
- **Cross-reference table** - Jinak nazývána **xref** je tabulka obsahující reference na veškeré objekty uložené v těle a v kódu začíná řetězcem *xref*. Reference uložená v tabulce je reprezentována na 2 řádcích pomocí řetězce a skládá se z 5 částí o celkové velikosti 20 bytů včetně oddělovačů *CRLF*:



Obrázek 2.2: Interní struktura PDF souboru

```
%PDF-1.4%âãÐÓ <--- hlavička souboru
```

Obrázek 2.3: Ukázka hlavičky

```
4 0 obj     <--- start objektu
<</Filter /FlateDecode /Length 1882>>
stream      <--- start dat
xśíśßoÚ6□ĄąWżî□ °‡´ŘÂŠ□ő□□öÐ
iÐ□CÓĆŘ□Ö=ŘŽăx‰“ĹVšv ýŽ§□–Ďrě¸>#0□#’,Qüňî H~
(S’üµ-_ĽŃRk□□□lźËŁv+ß□´□\□·^□źj9□Ř÷(Nä}
endstream   <--- konec dat
endobj      <--- konec objektu
```

Obrázek 2.4: Ukázka dat objektu

- *Číslo objektu* - Jednoznačný číselný identifikátor objektu.
- *Počet subobjektů* - Počet částí daného objektu vyskytujícího se v dokumentu.
- *Začátek objektu* - Tvoří většinu řetězce (prvních 10 bytů) a určuje offset od začátku PDF dokumentu až po začátek daného objektu.
- *Generační číslo objektu* - Vyjadřuje jak často byl objekt vymazán při úpravě dokumentu.

– *Identifikátor využití* - Nabývá hodnot *f* (free) nebo *n* (use) a vyjadřuje, zda je objekt vyobrazen v dokumentu.

```
xref   <--- start tabulky
0 1    <--- ID objektu a počet subobjektů
0000000001 65535 f
31 1
0000423765 00000 n
```

Obrázek 2.5: Ukázka jednoduché xref tabulky

- **Trailer** - Trailer je seznam informací, ze kterých lze snadno zjistit například velikost nebo umístění xref tabulky. Trailer může obsahovat tyto elementy:
  - *Size* - Udává počet objektů referencovaných v xref tabulce.
  - *Prev* - Offset od začátku dokumentu k předchozí xref tabulce.
  - *Root* - Odkazuje na objekt obsahující informace ohledně katalogu xref tabulek.
  - *Encrypt* - Specifikuje komprimující algoritmus použití pro daný dokument.
  - *Info* - Obsahuje dodatečné informace ohledně katalogu xref tabulek.
  - *ID* - 2-bytový identifikátor PDF dokumentu.
  - *XrefStm* - Offset od začátku dokumentu až k dekódovanému xref streamu. Využívá se pouze u hybridně-referencovaných souborů pouze tehdy, kdy hledaný objekt není nalezen v xref tabulce (před tím, než se volá element *Prev*).

```
trailer         <--- start traileru
<<
/Size 742       <--- velikost xref tabulky
/Root 741 0 R   <--- odkaz na objekt odkazující na objekt katalogu xref tabulek
/Info 740 0 R   <--- odkaz na informační slovník xref tabulek
/ID [<009feb05c3e899ac1d26612f86bb56aa> <009feb05c3e899ac1d26612f86bb56aa>] --->
<--- indentifikátor souboru
>>
startxref
408764          <--- offset tabulky xref
%%EOF
```

Obrázek 2.6: Ukázka traileru

2.4 PDF formuláře

Pod pojmem formulář si lze představit dokumenty, které od svých uživatelů vyžadují vyplnění určitých údajů. Mezi nejznámější dokumenty lze například uvést daňová přiznání, oznamovací tiskopisy, dotazníky, složenky aj. Ruční vyplňování i jejich následné zpracování bývá obvykle pracné a zdlouhavé, proto je v dnešní době výhodnější využívat interaktivní elektronické formuláře. Základní výhoda těchto formulářů spočívá ve snažším vyplňování, zpracování lze jednoduše zautomatizovat a také se díky elektronické podobě zvedne úspora papíru a financí vynaložených na tisk formulářů. Mezi nejčastější formuláře, které lze potkat na internetu, jsou ve formátu HTML a lidé se s nima setkávají každodenně (ať už to jsou jednoduché přihlašovací formuláře stránek nebo různé dotazníky na určitá témata). Nevýhoda těchto formulářů je v jejich závislosti na internetovém připojení.

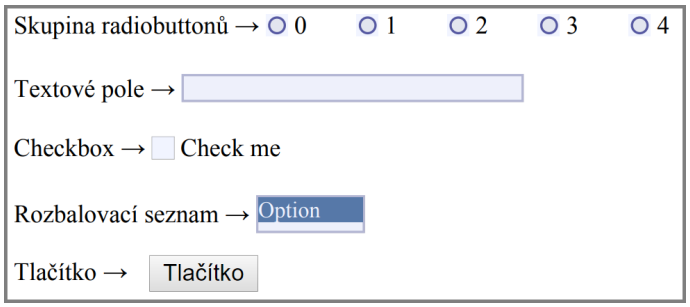
Proto firma Adobe přišla se svým řešením, interaktivním PDF formulářem, který lze vyplňovat kdekoliv nezávisle na internetovém připojení. Mezi další výhody PDF formulářů patří elektronický podpis (lze s ním potvrzovat smlouvy z domova), zabezpečení (dokument se otevře až po zadání správného hesla, neautorizovaným uživatelům je přístup zamítnut) aj. Tyto formuláře obsahují stejné interaktivní prvky jako mají HTML formuláře viz kapitola 2.4.1. Pro generování PDF formulářů lze využít kterýkoliv programovací jazyk, který podporuje práci s PDF soubory (například *PHP*, *Java*), produkty firmy Adobe (například *Adobe Acrobat*) nebo lze použít i nekomerční aplikace typu *TeX* nebo *pdfmarks*.

Tvorba formulářů je jedna věc, druhá věc je jejich zpracování (získání dat vyplněných od uživatele). Mezi nejznámější nástroje pro zpracování vyplněných dat patří určitě nástroj *FDF Toolkit* od firmy Adobe. Tento nástroj je zcela zdarma a umožňuje vytvářet orientovaná řešení pro zpracování dat v jazycích *C/C++*, *ActiveX*, *Java* a *Perl*. Jsou-li data odeslána v HTML, lze k jejich zpracování využít nástroje určené pro formáty *CGI*, *PHP* aj. [1].

2.4.1 Základní prvky

Jednotlivé formulářové prvky mohou mít přiřazeny nejrůznější atributy a jsou reprezentovány jako PDF objekty. Tyto atributy lze rozdělit do následujících skupin: **Vzhled** (definovaný vzhled prvku), **Akce** (po kliknutí na prvek se provede daná akce), **Formát** (typ fontu textu aj.), **Ověřování dat** (akceptovatelný formát vstupu) a **Výpočty** (matematické operace použité při práci se vstupy z jiných prvků) [2].

Ve formuláři se může vyskytovat až 7 různých prvků viz obrázek 2.7:



Obrázek 2.7: Základní prvky vyskytující se v PDF

- **Textové pole** - Slouží k vyplnění textu. Jako příklad lze uvést například klasický přihlašovací formulář, který obsahuje 2 textové pole, jedno pro zadání uživatelského jména a druhé (upravené, místo textu se zobrazují pouze speciální znaky pro zakrytí zadaného textu) pro zadání hesla. Při vytváření lze předvyplnit toto pole výchozím textem, lze omezit maximální počet znaků vkládaných do pole a jejich formát. Pole může být uzamčeno a může sloužit i jako informační položka.
- **Tlačítko** - Účel tohoto prvku je spouštění zvolených akcí, které se po kliknutí na tlačítko mají provést, tudíž se označují jako hlavní řídící prvek každého formuláře. Tlačítko se skládá převážně z ikonky a textu, případně mu může být nastaven externí obrázek.
- **Seznam** - Zobrazuje seznam položek v rolovacím okně, ze kterého lze současně označit jeden nebo více položek (s využitím klávesy *Shift* nebo *Ctrl*). Pro seznamy lze nastavit filtry, které budou seznam třídit podle předem daných parametrů a zobrazí položky na základě těchto filtrů.
- **Kombinované pole** - Kombinované pole je ve své podstatě seznam prvků, ale liší se ve výběru položek. V kombinovaném poli lze vybrat pouze jeden aktivní prvek, ostatní budou zakázány. Platí zde pravidla s tříděním pprvků odle filtrů.
- **Přepínací tlačítka** - Jinak označované jako **radio-buttony** je seznam tlačítek, ve kterém uživatel vybírá pouze jednu z nabízených hodnot.
- **Zaškrtávací pole** - Jedná se o indikační prvek umožňující současný výběr více položek. V odborném prostředí se označuje jako **Checkbox**.
- **Podpis** - Pomocí tohoto prvku lze do dokumentu vložit elektronický podpis.

3 Knihovny

V programování můžeme knihovnu definovat jako kolekci předem zkompilovaných procedur, funkcí (v objektovém programování i třídy a objekty), konstant a datové typy. Knihovna by měla být následně i dobře zdokumentována pro její snadnější zakomponování do již existujících modulů (při používání nezdokumentovaných knihoven se musí provádět takzvaný reverse engineering pro zjištění všech procedur a funkcí, nebo vyhledávat už hotová řešení na internetu).

Knihovny jsou z technického hlediska rozděleny do 2 skupin, které se následně rozdělují do 2 podskupin:

- **Rozdělení z hlediska způsobu propojení s programem:**
  - *Statická knihovna* - Zdrojový kód knihovny je v průběhu překládání zkopírován do výsledného programu pomocí kompilátoru. Největší výhoda statických knihoven spočívá v jistotě, že všechny potřebné knihovny budou přítomny ve výsledném programu, proto nikdy nemůže nastat situace nazvaná **dependency hell (DLL Hell)**, která značí nepřítomnost jedné nebo více knihoven, které jsou využívány jinou knihovnou, nebo také může značit nadbytečné závislosti knihoven, které nejsou ve výsledku využity.
  - *Dynamická knihovna* - Oproti statickým knihovnám, zdrojové kódy dynamických knihoven nejsou zakomponovány ve výsledném programu, ale pomocí linkeru jsou vytvořeny záznamy na funkce použité v programu, které jsou následně uloženy do tabulky symbolů vyskytující se ve výsledném programu.
- **Rozdělení z hlediska sdílení kódu mezi programy:**
  - *Sdílená knihovna* - Zdrojový kód sdílených knihoven je možné sdílet mezi více programy. Tímto způsobem jsou efektivně sníženy nároky na velikost operační paměti, protože úseky kódu využívané více procesory jsou uloženy ve sdílené paměti (namapovány do adresních prostorů všech procesů, které ji využívají).
  - *Nesdílená knihovna* - Nesdílené knihovny neumožňují sdílet úseky kódu více procesorům z důvodu kopírování kódu z knihoven do souborů při linkování souborů.

3.1 PHP Knihovny pro generování PDF

3.1.1 FPDF

IN PROGRESS

3.1.2 dompdf

IN PROGRESS

3.1.3 mPDF

IN PROGRESS

3.1.4 FPDF

IN PROGRESS

3.1.5 TCPDF

Knihovna **TCPDF** je open-source PHP knihovna sloužící pro práci s PDF soubory. Její vývoj odstartoval už v roce 2002 kdy vznikla jako odnož knihovny FPDF. Díky její rozmanitosti funkcí pro vytváření PDF souborů si jí oblíbilo mnoho uživatelů a je využívána i na mnoha webových portálech. Mezi hlavní funkcionality lze zařadit: podpora UTF-8 kódování, komprese stránek, vkládání zdrojových souborů, šifrování celého dokumentu, vkládání čárových kódů aj. Protože je psána pouze v jazyce PHP a nevyužívá žádné externí knihovny, pak ji lze brát jako vhodnou knihovnu pro vyvíjený modul.

3.2 PHP Knihovny pro zpracování PDF

3.2.1 pdf-to-html

Knihovna **pdf-to-html** má za úkol překonvertovat veškerý obsah PDF souboru do HTML struktury, ze které lze snadno vyextrahovat obsah souboru a předat ho ke zpracování. Požadavky pro správné fungování této knihovny je mít v PHP konfiguraci mít povolen přístup k příkazové řádce systému a mít na serveru nainstalovaný **Poppler** (program napsaný v jazyce C++ sloužící k renderování PDF dokumentů) [8]. Protože je tato knihovna závislá na externím programu (Poppler), pak ji nelze brát jako vhodnou pro vyvíjený modul.

3.2.2 TCPDF parser

Knihovna **TCPDF parser** je součást knihovny **TCPDF** (viz 3.1.5), která se soustředí na zpracování PDF souboru. Pro svůj běh nepotřebuje žádné externí knihovny a je psána pouze v jazyce PHP, ale stále se nachází ve fázi vývoje a při jejím použití nemusíme vždy dojít ke správnému výsledku. Proto z tohoto důvodu není nejvhodnější pro vyvíjený modul a bude lepší se ohlédnout po jiné knihovně.

3.2.3 PDF Parser

PDF Parser je další z mnoha knihoven sloužících pro zpracování PDF souborů. Tato knihovna je založena na již existující knihovně **TCPDF parser**, která je navíc doplněna o nové funkcionality jako je například extrakce metadat a komprimovaných souborů aj. Na oficiálních stránkách lze najít demo verzi, která demonstruje funkčnost, kdy po nahrání kteréhokoliv PDF souboru se na stránkách zobrazí data extrahovaná z nahraného souboru. Vzhledem k tomu, že PDF Parser je plně vyvinutá knihovna využívaná na mnoha webových portálech pro zpracování PDF souborů, pak ji lze brát jako vhodnou knihovnu pro vyvíjený modul.

3.2.4 php-pdftk

Nástroj **PDF Toolkit** (zkráceně pdftk) je multiplatformní nástroj pro manipulaci s PDF soubory, který navazuje na starší verzi nástroje **iText library**. PDF Toolkit lze najít ve tře verzích. Mezi neplacené verze patří *PDFtk Server*, což je open-source tool v příkazové řádce a verze *PDFtk Free*, která je úplně zdarma), zatímco mezi placené verze patří verze *PDFtk Pro* (patří mezi proprietární software, jehož zdrojové soubory nejsou volně dostupné). Pomocí tohoto nástroje (převážně v placené verzi) lze oddělovat/ spojovat/šifrovat PDF soubory, měnit jeho vlastnosti, metata, vyplňovat formuláře *FDF daty* (Forms Data Format) a mnoho dalších funkcionalit [4]. Díky rozsáhlé funkcionalitě byla vyvinuta knihovna v PHP s názvem **php-pdftk**, pomocí které lze využívat veškerou funkcionalitu tohoto nástroje v jazyce PHP. Bohužel díky závislosti na externím programu ji nelze brát jako vhodnou pro vyvíjený modul.

3.2.5 pdf-to-text

IN PROGRESS

4 Výsledky testování modulu

5 Závěr

Literatura

[1] *PDF formuláře: obecný úvod* [online]. 2002. [cit. 2002/04/25]. Dostupné z: http://www.grafika.cz/rubriky/pdf---adobe-acrobat/pdf-formulare-obecny-uvod-130460cz.

[2] *PDF formuláře: Popis formulářových prvků* [online]. 2002. [cit. 2002/05/15]. Dostupné z: http://www.grafika.cz/rubriky/pdf---adobe-acrobat/pdf-formulare-popis-formularovych-prvku-130502cz.

[3] *Compression in PDF files* [online]. Prepressure, 2017. [cit. 2017/01/05]. Dostupné z: https://www.prepressure.com/pdf/basics/compression.

[4] *PHP PDFTK* [online]. 2017. [cit. 2017/12/17]. Dostupné z: https://www.drupal.org/project/phppdftk.

[5] Christenssion, P. *PDF Definition* [online]. TechTerms. Sharpened Productions, 2018. [cit. 2018/04/05]. The Tech Terms Dictionary. Dostupné z: https://techterms.com/definition/pdf.

[6] King, J. *Introduction to the Insides of PDF* [online]. Adobe, 2005. [cit. 2005/04/29]. Dostupné z: https://www.adobe.com/technology/pdfs/presentations/KingPDFTutorial.pdf.

[7] Lukan, D. *PDF File Format: Basic Structure* [online]. InfoSec, 2018. Dostupné z: https://resources.infosecinstitute.com/pdf-file-format-basic-structure.

[8] Nikolaev, A. *PHP PDF to HTML: Convert PDF to HTML using Poppler* [online]. 2018. [cit. 2018/06/29]. Dostupné z: https://www.phpclasses.org/package/9423-PHP-Convert-PDF-to-HTML-using-Poppler.html.

[9] Rouse, M. *Portable Document Format (PDF)* [online]. TechTarget, 2010. [cit. 2010/05/20]. Dostupné z: https://whatis.techtarget.com/definition/Portable-Document-Format-PDF.

[10] Whitington, J. *PDF Explained, Chapter 3. File Structure.* O'Reilly Media, Inc., 2011. ISBN 9781449310028.