**VALIDÁTOR**

Uživatelská dokumentace

Verze: 1.01

Datum: 14.2.2013

**Obsah**

[Úvod 3](#_Toc341366656)

[Odkazované dokumenty 3](#_Toc341366657)

[O aplikaci 3](#_Toc341366658)

[Požadavky na systém 3](#_Toc341366659)

[Proces validace 3](#_Toc341366660)

[Funkce aplikace 4](#_Toc341366661)

[Validace struktury balíčku 4](#_Toc341366662)

[Validace hlavního metadatového souboru 4](#_Toc341366663)

[Validace vedlejších metadatových souborů 6](#_Toc341366664)

[Struktura adresáře programu 6](#_Toc341366665)

[Popis konfiguračního souboru DPValidator.properties 7](#_Toc341366666)

[Instalace aplikace 8](#_Toc341366667)

[Spuštění aplikace 9](#_Toc341366668)

[Výstupy validace 11](#_Toc341366669)

[Příklad validace PSP balíčku 11](#_Toc341366670)

[Seznam obrázků 13](#_Toc341366671)

[Seznam tabulek 13](#_Toc341366672)

[Seznam zkratek a pojmů 13](#_Toc341366673)

# Úvod

Dokumentace je určena technickým zpracovatelům dat, která jsou výstupem digitalizačního procesu. Dokumentace vysvětluje instalaci aplikace, nastavení aplikace a jednotlivé funkce aplikace.

# Odkazované dokumenty

[1] specifikace\_periodika\_1\_4.pdf ze dne 4. 4. 2012

[2] specifikace\_monografie\_1-0.pdf ze dne 4. 4. 2012

# O aplikaci

Účelem aplikace je poskytnout jednoduchý nástroj pro kontrolu vytvořených PSP balíčků zda odpovídají definici metadat pro periodika a monografie NDK [1] a [2][[1]](#footnote-1). Podle nastavení parametrů aplikace je aplikace schopna provést kontrolu jednoho balíčku nebo celé dávky balíčků obsažené v dané složce.

Úkolem aplikace je validovat strukturu PSP balíčku, validovat hlavní metadatový soubor a validovat vedlejší metadatové soubory.

## Požadavky na systém

Aplikace vyžaduje nainstalovanou JAVA[[2]](#footnote-2) verze 1.6[[3]](#footnote-3) a vyšší na platformě Windows XP/Vista/2007 a platformě UNIX.

# Proces validace

Obr. 1 - Proces validace

# Funkce aplikace

Aplikace provádí validaci kompletního vstupního PSP balíčku podle pravidel pro periodikum, případně monografii a výsledek zapíše formou reportu obsahujícího, zda balíček vyhověl pravidlům, popřípadě soupis chyb.

Průběh validace lze rozdělit na několik kroků:

* Kontrola struktury PSP balíčku.
* Kontrola hlavního metadatového souboru (hlavní METS).
* Kontrola vedlejších metadatových souborů (vedlejší METS soubory v amdSec složce).

## Validace struktury balíčku

Základní kontrola, která vyhodnotí, zda balíček obsahuje všechny povinné soubory a složky a zda jsou správně pojmenované dle jmenné konvence definované v [1] a [2].

Struktura balíčku:

* info.xml
* masterCopy (složka) – obsahuje obrazy JPEG 2000 lossless
* userCopy (složka) – obsahuje obrazy JPEG 2000 lossy
* ALTO (složka) – obsahuje soubory ALTO.xml pro každou stranu
* TXT (složka) – obsahuje soubory OCR.TXT pro každou stranu
* Hlavní\_METS.xml
* Soubor.md5

## Validace hlavního metadatového souboru

Kontrola, zda hlavní METS záznam má správný formát a obsahuje všechny povinné položky.

Tuto validaci lze rozdělit do následujících kroků:

* Validace platnosti xml pomocí mets.xsd a mods-3-4.xsd.
* Validace fileSec části - kontrola zda odkazované soubory (element FLocat atribut xlink:href) skutečně existují.
* Validace přítomnosti povinných položek a atributů pomocí šablon – na základě typu dokumentu validátor vybere profil, který obsahuje seznam jednotlivých šablon, které mají být použity pro validaci.[[4]](#footnote-4)

## Validace vedlejších metadatových souborů

Kontrola vedlejších metadatových souborů (AMD\_METS).

Kroky validace pro jednotlivé soubory:

* Validace přítomnosti povinných položek a atributů pomocí šablon.
* Validace platnosti xml jednotlivých událostí (premis:event) pomocí premis.xsd.
* Validace platnosti xml jednotlivých agentů (premis:agent) pomocí premis.xsd.
* Validace platnosti xml technických metadat v elementech MIX001 a MIX002 pomocí mix.xsd.

## Struktura adresáře programu

**validate** – *kořenový adresář programu*

**bin –** *spustitelný soubor programu*

**validate.bat –** *spustitelný soubor pro platformu Windows*

**validate.sh –** *spustitelný soubor pro platformu UNIX*

**config –** *konfigurační soubory*

**DPValidator.properties –** *hlavní konfigurační soubor – popis viz dále*

**log4j.xml –** *standardní konfigurační soubor pro logování průběhu programu*

**\*.xsd -** *schémata pro validaci XML souborů*

**lib –** *programové knihovny aplikace java*

**templates** *– šablony validace dle specifikace [1] a [2]*

# Popis konfiguračního souboru DPValidator.properties

|  |  |
| --- | --- |
| **Klíč** | **Popis** |
| **validation.output.location** | Definuje umístnění chybového výstupu a jméno výstupního souboru. |
| **metsdata.templates.location** | Umístnění XML šablon pro validaci. |
| **validation.output.append** | Hodnoty True/False. V případě hodnoty True a existence výstupního souboru do tohoto souboru připsán výsledek validace. V případě hodnoty False a existence výstupního souboru je tento soubor při běhu programu smazán a vytvořen nový. |
| **validation.input.dirmode** | Hodnoty True/False. V případě True je na vstupu očekáván nadřazený adresář několika PSP balíčků. V případě hodnoty False, je na vstupu očekáván adresář jenom jednoho PSP balíček. |
| **mandatory.directory.X[[5]](#footnote-5).dirname** | Definuje povinnost validace pod-složky (kde dirname určuje jméno složky) v digitalizovaném záznamu. |
| **crossvalidation.on** | Hodnoty True/False. V případě hodnoty True je zapnuta křížová validace souborů. |
| **mets.xsdfile.location** | Umístnění XSD definice pro Mets. |
| **mods.xsdfile.location** | Umístnění XSD definice pro Mods. |
| **premis.xsdfile.location** | Umístnění XSD definice pro Premis. |
| **mix.xsdfile.location** | Umístnění XSD definice pro Mix. |
| **metsdata.templates.version** | Určuje verzi šablony, která se použije, verze souboru je určená příponou v jeho názvu. Např. pro verzi 1 se jméno šablony složí následovně monographic\_volume***\_1.xml*** |
| **metsdata.profile.X.name** | Určuje jméno profilu X. Např. „monograph“ |
| **metsdata.profile.X.tests** | Určuje náležící validační testy k profilu X |

Tab. 1 – Konfigurační soubor

# Instalace aplikace

Aplikace se instaluje následujícím způsobem:

1. Ověření správné verze javy: ***java –version .* (**V případě chybějící instalace javy je nutné nejprve nainstalovat, popř. povýšit verzi na 1.6 a vyšší). V případě, že nejsou přidělěna uživatelská práva k instalaci, je nutné požádat o asistenci administrátora, případně jiné pověřené osoby.

*Příklad výpisu ve Windows:*

*D:\\_validator>****java -version***

*java version "1.6.0\_25"*

*Java(TM) SE Runtime Environment (build 1.6.0\_25-b06)*

*Java HotSpot(TM) Client VM (build 20.0-b11, mixed mode, sharing)*

1. Ověření existence nástroje pro extrakci ZIP souboru (např. WINZIP, Total Commander, gzip, apod.)
2. Rozbalení instalačního balíčku  do  cílového kořenového adresáře aplikace, implicitně ***validate*** .
3. Nastavení konfigurace pro standardní logovaní javy. Konfigurační soubor se nachází:

<***instalační\_adresář>\config\log4j.xml****.* Zde je nutné nastavit parametr **value** na hodnotu, kam je požadováno směrování logů běhu aplikace. Např.:

<appender name="R" class="org.apache.log4j.RollingFileAppender">

**<param name="file" value="*d:/Logs/NDK/testbatch/app.log*"/>**

<param name="MaxFileSize" value="1000KB"/>

<layout class="org.apache.log4j.PatternLayout">

<param name="ConversionPattern" value=" %d %p %t %c - %m%n"/>

</layout>

</appender>

Cesta k souboru (v tomto případě„***d:/Logs/NDK/testbatch/****“)*  **musí existovat**, samotný soubor **„*app.log“***existovat nemusí.

1. Nastavení konfigurace aplikace v konfiguračním souboru ***validate\config\*** ***DPValidator.properties*** dle vlastních potřeb (viz Popis konfiguračního souboru). Povinné je alespoň nastavit parametry:

* validation.output.location
* metsdata.templates.location

Poznámky:

* Při specifikaci cest v konfiguračních souborech pro javu je nutné uvádět znak “/” (dopředné lomítko) na obou platformách.
* Cestu ke konfiguračním souborům a šablonám je možné uvést:
* absolutně (např.: C:/validator/templates/)
* relativně (např.: ./templates/) - v tomto případě je však nutné spouštět program z příslušného adresáře, aby nalezl šablony pro běh programu)

# Spuštění aplikace

Aplikace na svém vstupu může přijmout dva parametry. Pořadí parametrů musí být zachováno.

1. První parametr určuje umístnění konfiguračního souboru. Parametr je zadán formou plné cesty, včetně názvu konfiguračního souboru např. *„****d:\Test\Config\DPValidator.properties****“*
2. Druhý parametr určuje umístnění testované složky/skupiny složek. Parametr je zadán formou plné cesty končící názvem složky např. *„****d:\Test\TestovaciData****“*

Aplikace se spouští pomocí skriptu umístěném v složce ***validate\bin*** a to ve dvou režimech:

* **Dávkový (zpracování několika PSP balíčků najednou)**

Z příkazového řádku např. pomocí Windows ***cmd*** z adresáře validate:

***bin\validate <nadřazený\_adresář>***

a nastavení konfiguračního souboru ***DPValidator.properties*** položky ***validation.input.dirmode = true***

Např. V adresáři C:\pokus jsou umístěny dva PSP balíčky. Jejich dávkové zpracování lze spustit takto:

C:\validate> ***bin\validate C:\pokus***

Tento způsob zajistí dávkovou validaci obou PSP balíčků.

* **Samostatný režim (zpracování jednoho PSP balíčku)**

Z příkazového řádku např. pomocí Windows ***cmd*** z adresáře validate:

***bin\validate <adresář PSP balíčku>***

a nastavení konfiguračního souboru ***DPValidator.properties*** položky ***validation.input.dirmode = false***

Např. Zpracování PSP balíčku fa1fefa0-33cf-11e2-9612-0050568209d3 se spustí takto:

C:\validate> ***bin\validate fa1fefa0-33cf-11e2-9612-0050568209d3***

Tento způsob zajistí validaci výše zmíněného PSP balíčku.

Aplikaci je možné také spustit v obou režimech s explicitně specifikovaným souborem konfigurace:

***bin\validate <soubor\_konfigurace> <nadřazený\_adresář>***

*Např. v režimu dávkové validace:*

***bin\validate C:\Temp\DPValidator.properties C:\pokus***

*Program validace se bude řídit nastaveném ze souboru* ***C:\Temp\DPValidator.properties****.*

Poznámka pro platformu UNIX:

Na platformě UNIXu se při spouštění programu validace musí respektovat odlišnosti UNIXu: např.: ***./home/validator/validate ./home/data/fa1fefa0-33cf-0050568209d3***

# Výstupy validace

Hlavním výstupem aplikace je report popisující výsledek validace. Nejprve je v sumáři uveden celkový výsledek validace s počtem chyb. V případě existence chyb následují jejich detailní informace výpisem na samostatný řádek pro každou chybu. Umístnění souboru a jeho jméno je možné nastavit pomocí konfiguračního souboru (*validation.output.location*). Před název souboru se ve výsledku automaticky přidá prefix odpovídající názvu testované složky (např. testovaná složka se jmenuje *„XYZ“* výstupní soubor je nastaven na *„****D:\test\output.log****“* výsledný soubor se bude jmenovat „*XYZ\_output.log“* a bude ve složce *„D:\test\“*).

Dalším výstupem aplikace je standardní java log popisující detailně běh aplikace. Umístnění logu je možné nastavit pomocí konfiguračního souboru (viz Instalace aplikace). Tento výstup je rovněž vypsán do konzole, ze které je aplikace spuštěna.

## Příklady validace PSP balíčku

Následující příklad validace PSP balíčku fa1fefa0-33cf-11e2-9612-0050568209d3, kde chybí adresář ALTO.

Jsou vytvořeny dva soubory:

**fa1fefa0-33cf-11e2-9612-0050568209d3\_output.log:**

Processing PSP in directory fa1fefa0-33cf-11e2-9612-0050568209d3 2 ERRORS!

ERROR: Violation type: Cross validation. Violation message: Dir alto does not exist!

**app.log (extrakt)**

………

2012-11-22 12:50:04,195 DEBUG main com.logica.ndk.tm.validation.validator.core.MetsXSDValidator - Validation Mets aginst XSD has end

2012-11-22 12:50:04,196 INFO main com.logica.ndk.tm.validation.validator.parts.MetsValidator - Mets validator has end

2012-11-22 12:50:04,196 INFO main com.logica.ndk.tm.validation.validator.parts.MetadataValidator - Metadata validator is starting

2012-11-22 12:50:04,233 INFO main com.logica.ndk.tm.validation.validator.parts.MetadataValidator - Start validating main mets document

2012-11-22 12:50:04,235 INFO main com.logica.ndk.tm.validation.validator.core.MetadataValidatorCore - Validation mets document started.22.11.2012 12:50:04

2012-11-22 12:50:04,236 INFO main com.logica.ndk.tm.validation.validator.core.MetadataValidatorCore - Validation profile: Monograph

…….

Následující složitější příklad validace PSP balíčku **KPW01192129**

**KPW01192129\_output.log (extrakt):**

Processing PSP in directory KPW01192129 505 ERRORS!

ERROR: Violation type: Main mets validation using template: monographic\_volume. Violation message: Validation error in : //mets:mets[1]/mets:dmdSec[@ID='MODSMD\_VOLUME\_0001'][1]/mets:mdWrap[1]/mets:xmlData[1]/mods:mods[1]/mods:identifier[4]/@type Mandatory attribute type have bad value: pole1, enable values: [doi, hdl, issn, isbn, ccnb, permalink, urnnbn, uuid, local, barcode, oclc, URN, http, uri]

…

…

…

Krátkým rozborem uvedené chyby je zjištěno toto:

Soubor: *Hlavní mets*

Šablona: *monographic\_volume*

Cesta k chybě v souboru: *//mets:mets[1] /mets:dmdSec[@ID='MODSMD\_VOLUME\_0001'][1] /mets:mdWrap[1] /mets:xmlData[1] /mods:mods[1] /mods:identifier[4]/@type*

Chyba: Povinný atribut ***type*** obsahuje špatnou hodnotu ***pole1***

Povolené hodnoty: *[doi, hdl, issn, isbn, ccnb, permalink, urnnbn, uuid, local, barcode, oclc, URN, http, uri]*

# Seznam obrázků

Obr. 1 – Proces validace

# Seznam tabulek

Tab. 1 – Konfigurační soubor

# Seznam zkratek a pojmů

|  |  |
| --- | --- |
| **Zkratka / Pojem** | **Vysvětlení** |
| MIX | **Metadata for Images in XML Standart**  Je XML schéma pro technické informace potřebné pro správu digitálních obrazů. |
| Premis | Je mezinárodní standard pro metadata. Jeho účelem je podpora zachování digitálních objektů a zajištění jejich dlouhodobé použitelnosti. |
| PSP | **Producer submission package**  Balíček dat a metadat, který přichází od producenta dat. |
| METS | **Metadata Encoding & Transmission Standard**  Schéma METS je standardem pro kódování popisných, administrativních a strukturálních metadata týkajících se objektů v digitální knihovně. |
| Amd mets | **Administrative Metadata**  Obsahuje technická metadata o jednotlivých souborech, přistupová práva, apod. |
| MODS | **Metadata Object Description Schema**  Je schéma pro bibliografické informace. |
| XML | **Extensible Markup Language**  Je značkovací jazyk určený především pro výměnu informací mezi aplikacemi a publikaci dokumentů. |
| XSD | **XML Schema Definition**  Je XML schéma, které popisuje strukturu XML dokumentu. |

1. Dostupné například zde <http://ndk.cz/digitalizace/vystavena-nova-verze-definice-metadat-pro-periodika-a-monografie> [↑](#footnote-ref-1)
2. Dostupné například: <http://www.java.com/en/download/index.jsp>. [↑](#footnote-ref-2)
3. Lze zjistit například zadáním příkazu **java –version** do příkazové řádky. [↑](#footnote-ref-3)
4. V případě chyby formátu šablony, je vypsána tato chyba ale validace pokračuje pomocí následující šablony. [↑](#footnote-ref-4)
5. X je unikátní číselný idenfikátor v rámci konfiguračního souboru. Používá se pro opakovatelné klíče. [↑](#footnote-ref-5)