DITEC
Data Information Technology & Expert Consulting

Integračná príručka
D.Signer/XAdES Java, v2.0

Copyright

Popisné charakteristiky dokumentu

Projekt	GOV_ZEP		A3019_002
Dokument	Integračná príručka		
Podnázov	D.Signer/XAdES Java, v2.0		
Ref. číslo	GOV_ZEP.207	Verzia	14

Vypracoval	Vittek Róbert	Podpis	Dátum 24. 10. 2023
Preveril		Podpis	Dátum
Schválil		Podpis	Dátum

Formulár	Dokument		
Ref. číslo	Fo 11	Dátum poslednej aktualizácie	Dátum 14. 10. 2005

Akceptované dňa : <Dátum akceptácie>

Za <Objednávateľa>: Za <Dodávateľa>.:

<Meno zodpovednej osoby> < Meno zodpovednej osoby >

Projekt	GOV_ZEP		A3019_002
Dokument	Integračná príručka		
Referencia	GOV_ZEP.207	Verzia	14

Záznamy o zmenách

Autor	Popis zmien	Dátum	Verzia

Pripomienkovanie a kontrola

Autor	Stanovisko	Dátum	Verzia

Rozdeľovník

	Priezvisko Meno	Firma, Funkcia
Originál		
Kópia		
Kópia		
Kópia		

Obsah

1. Úvod

Tento dokument je určený pre zhotoviteľov, prípadne prevádzkovateľov systémov, v rámci ktorých bude aplikácia D.Signer/XAdES Java pre zaručený/kvalifikovaný elektronický podpis (ZEP/KEP) integrovaná.

Jednotlivé časti dokumentácie aplikácie D.Signer/XAdES Java je možné použiť pri tvorbe dokumentácie týchto systémov po dohode s vlastníkmi autorských práv aplikácie D.Signer/XadES Java.

2. Zoznam použitých skratiek

CRL – Certificate Revocation List; zoznam zneplatnených certifikátov

DHC – Data Hashing Component

DTBS – Data To Be Signed

DTBSF – Data To Be Signed Formatted, resp. komponent Data To Be Signed Formatter

DTBSR – Data To Be Signed Representation

KEP – kvalifikovaný elektronický podpis

NBÚ – Národný bezpečnostný úrad

OCSP – Online Certificate Status Protocol

QSCD – Qualified Signature Creating Device; zariadenie na vytváranie kvalifikovaného podpisu

SAC – Signer's Authentication Component

SCA – Signature Creation Application

SCDev – Signature Creating Device

SCVA – Signature Creation and Validation Application

SDOC – Signed Data Object Composer

SDP – Signer's Document Presentation

SIC – Signer Interaction Component

SLC – Signature Logging Component

SSA – SCDev/SCA Authenticator

SSC – SCDev/SCA Communicator

SSCD – Secure Signature Creating Device; bezpečné zariadenie na vytváranie elektronického podpisu

SVA – Signature Validation Application

XML – eXtended Markup Language

XSD – XML Schema Definition

XSL – eXtensible Stylesheet Language

XSLT – XSL Transformation

XAdES – XML Advanced Electronic Signatures

XAdES_ZEP – profil formátu elektronického podpisu XAdES pre ZEP

XAdES_ZEPbp – profil formátu kvalifikovaného elektronického podpisu na báze XAdES baseline profile

ZEP – Zaručený elektronický podpis

3. Referencie

[1] W3C/IETF Recommendation: "XML-Signature Syntax and Processing" v2002-02-12 (XMLDSIG)

[2] ETSI TS 101 733 – CMS Advanced Electronic Signatures (CAdES) v2.2.1

[3] ETSI TS 101 903 – XML Advanced Electronic Signatures (XAdES) v1.4.2

[4] RFC 3125 – Electronic Signature Policies

[5] RFC 3161 – Internet X.509 Public Key Infrastructure Time-Stamp Protocol

[6] RFC 3279 – Algorithms and Identifiers for the Internet X.509 PKI

[7] RFC 5280 – Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile

[8] RFC 3548 – The Base16, Base32, and Base64 Data Encodings

[9] RFC 5652 – Cryptographic Message Syntax (CMS)

[10] RFC 4051 – Additional XML Security Uniform Resource Identifiers

[11] Zákon č. 215/2002 Z.z. o elektronickom podpise a o zmene a doplnení niektorých zákonov v znení neskorších predpisov

[12] Vyhláška NBÚ č. 131/2009 Z.z. o certifikátoch a kvalifikovaných certifikátoch

[13] Vyhláška NBÚ č. 134/2009 Z.z. o produktoch elektronického podpisu

[14] Vyhláška NBÚ č. 135/2009 Z.z. o vyhotovení a overovaní elektronického podpisu a časovej pečiatky

[15] Vyhláška NBÚ č. 136/2009 Z.z. o spôsobe a postupe používania elektronického podpisu v obchodnom styku a administratívnom styku

[16] NBÚ Formáty certifikátov a kvalifikovaných certifikátov, v4.0 (2014-07-10)

[17] NBÚ Formáty zoznamu zrušených kvalifikovaných certifikátov, v3.0 (2010-01-17)

[18] NBÚ Formáty zaručených elektronických podpisov, v3.0 (2009-08-12)

[19] NBÚ Upresnenia obsahu a formálne špecifikácie formátov dokumentov pre ZEP, v1.0 (2007-07-24)

[20] Zákon č. 272/2016 Z.z. o dôveryhodných službách

[21] CWA 14170:2004 E – Security requirements for signature creation applications

[22] CWA 14171:2004 D/E/F – General guidelines for electronic signature verification

[23] XMLENC – XML Encryption Syntax and Processing", J. Reagle, D. Eastlake, December 2002. http://www.w3.org/TR/2002/REC-xmlenc-core-20021210/

[24] Koncepcia všeobecného formátu XML podpisu a aplikácie SCVA, DITEC, a.s., 2006

[25] Profil XAdES_ZEP – formát ZEP na báze XAdES, v1.0, DITEC, a.s., 2008

[26] Profil XAdES_ZEP – formát ZEP na báze XAdES, v1.1, DITEC, a.s., 2009

[27] Profil XAdES_ZEP – formát ZEP na báze XAdES, v2.0, DITEC, a.s., 2011

[28] Profil XAdES_ZEPbp – formát ZEP na báze XAdES baseline profile, v1.0, DITEC, a.s., 2016

[29] Rozhodnutie komisie 2011/130/EU, ktorým sa ustanovujú minimálne požiadavky na cezhraničné spracovanie dokumentov elektronicky podpísaných príslušnými orgánmi v zmysle smernice Európskeho parlamentu a Rady 2006/123/ES o službách na vnútornom trhu

[30] Rozhodnutie komisie 2014/148/EÚ zo 17. marca 2014, ktorým sa mení rozhodnutie 2011/130/EÚ, ktorým sa ustanovujú minimálne požiadavky na cezhraničné spracovanie dokumentov elektronicky podpísaných príslušnými orgánmi v zmysle smernice Európskeho parlamentu a Rady 2006/123/ES o službách na vnútornom trhu

[31] Nariadenie Európskeho Parlamentu a Rady EÚ č. 910/2014 z 23. júla 2014 o elektronickej identifikácii a dôveryhodných službách pre elektronické transakcie na vnútornom trhu a o zrušení smernice 1999/93/ES

[32] Rozhodnutie komisie 2015/1506/EU, ktorým sa ustanovujú špecifikácie týkajúce sa formátov zdokonalených elektronických podpisov a zdokonalených elektronických pečatí, ktoré môžu subjekty verejného sektora uznávať

[33] ETSI TS 103 171 – Electronic Signatures and Infrastructures (ESI) XAdES Baseline Profile, v2.1.1

[34] ETSI TS 102 918 – Electronic Signatures and Infrastructures (ESI);. Associated Signature Containers (ASiC), v1.3.1

[35] ETSI TS 103 174 – Electronic Signatures and Infrastructures (ESI);. ASiC Baseline Profile, v2.2.1

[36] Zákon č. 305/2013 o elektronickej podobe výkonu pôsobnosti orgánov verejnej moci a o zmene a doplnení niektorých zákonov (zákon o e-Governmente) v znení neskorších predpisov

[37] Výnos MF SR č. 55/2014 o štandardoch pre informačné systémy verejnej správy v znení neskorších predpisov

[38] Schéma dohľadu kvalifikovaných dôveryhodných služieb definovaná orgánom dohľadu, v1.4, NBÚ, 1.8.2019

[39] ETSI EN 319 412-1 – Electronic Signatures and Infrastructures (ESI) Certificate Profiles; Part 1: Overview and common data structures, v1.4.4

4. Popis aplikácie

Aplikácia D.Signer/XAdES Java predstavuje riešenie pre vytváranie zaručeného/kvalifikovaného elektronického podpisu (ZEP/KEP) nad množinou rôznych formátov dokumentov, resp. typov dát (XML dokumenty, PDF dokumenty atď.), prípadne nad ľubovoľnou kombináciou podporovaných formátov dát, ktoré spolu vytvárajú tzv. *multipart* dokument. Povolené formáty podpisovaných elektronických dokumentov v administratívnom styku špecifikuje Výnos MF SR č. 55/2014 o štandardoch pre IS VS [37]. Požiadavky na formát a obsah podpisovaných dát stanovuje dokument NBÚ SR – Upresnenia obsahu a formálne špecifikácie formátov dokumentov pre ZEP, v1.0.

Zaručený/kvalifikovaný elektronický podpis na druhej strane zabezpečuje integritu podpísaných dát a nepopierateľnosť podpisu. Aplikácia D.Signer/XAdES Java môže byť teda nasadená v rámci akéhokoľvek systému, kde je potrebné zabezpečiť jednak integritu prenášaných a spracovávaných dokumentov, ako aj nepopierateľnosť identity ich podpisovateľa.

Aplikácia D.Signer/XAdES Java pred samotnou procedúrou vytvorenia ZEP/KEP v zmysle zákona č. 215/2002 Z.z. o elektronickom podpise a o zmene a doplnení niektorých zákonov:

- zabezpečí podpisovateľovi zobrazenie všetkých podpisovaných dát jednoznačným a adekvátnym spôsobom,
- zaručí, že dáta sa pri podpise nezmenia.

Pre vytvorenie ZEP/KEP musí byť aplikácia použitá len v súlade s platnou podpisovou politikou pre ZEP/KEP, ktorá bola schválená NBÚ SR. Používateľ je pred vytvorením podpisu povinný presvedčiť sa, že podpisová politika, ktorú aplikácia používa, je stále platná a nebola zo strany vydavateľa predčasne zrušená. Výrobca, resp. integrátor aplikácie D.Signer/XAdES Java je povinný zabezpečiť také nastavenie konfigurácie aplikácie a parametrov volania metód rozhrania aplikácie, aby aplikácia vytvárala podpis v súlade so špecifikovanou podpisovou politikou.

Za obsah a sformátovanie vstupných dát (dokumentov), ako aj za dodržanie správneho postupu vytvorenia ZEP/KEP, definovaného v rámci podpisovej politiky, je zodpovedný podpisovateľ. Za správne vyhodnotenie platnosti vytvoreného ZEP/KEP a za špecifikovanie parametrov procesu verifikácie ZEP/KEP v súlade s podpisovou politikou je zodpovedný prijímateľ alebo prevádzkovateľ systému, ktorý tieto dáta spracováva.

Požiadavky NBÚ SR na vytváraný formát ZEP/KEP upravuje dokument – Formáty zaručených elektronických podpisov, v3.0. Minimálne požiadavky EÚ na cezhraničné spracovanie dokumentov elektronicky podpísaných príslušnými orgánmi v zmysle smernice Európskeho parlamentu a Rady 2006/123/ES o službách na vnútornom trhu stanovuje rozhodnutie komisie 2014/148/EU, ktoré nahrádza rozhodnutie komisie 2011/130/EU. Špecifikácie týkajúce sa formátov

zdokonalených elektronických podpisov a zdokonalených elektronických pečatí, ktoré môžu subjekty verejného sektora uznávať ustanovuje Rozhodnutie komisie 2015/1506/EU.

Aplikácia D.Signer/XAdES Java vytvára ZEP v súlade so schválenými formátmi pre zaručený elektronický podpis XAdES_ZEP, v1.0 (http://www.ditec.sk/ep/signature_formats/xades_zep/v1.0), XAdES_ZEP v1.1 (http://www.ditec.sk/ep/signature_formats/xades_zep/v1.1), XAdES_ZEP v2.0 (http://www.ditec.sk/ep/signature_formats/xades_zep/v2.0) a KEP v súlade s profilom pre kvalifikovaný elektronický podpis XAdES_ZEPbp, v1.0 (http://www.ditec.sk/ep/signature_formats/xades_zepbp/v1.0). Aplikácia D.Signer/XAdES Java vytvára typ podpisu XAdES_ZEP-EPES, resp. XAdES_ZEPbp-EPES teda elektronický podpis rozšírený o informáciu o čase vzniku ZEP/KEP, o explicitnú podpísanú referenciu podpisovej politiky a podpísané informácie o typoch a formátoch podpísaných dátových objektov, a tiež XAdES_ZEP-T, resp. XAdES_ZEPbp-T, teda elektronický podpis rozšírený o časovú pečiatku podpisu. Aplikácia D.Signer/XAdES Java umožňuje vytvárať aj typ podpisu XAdES_ZEPbp-BES, to znamená typ elektronického podpisu bez explicitne uvedenej referencie podpisovej politiky, v súlade s príslušnými nariadeniami komisie [29][30][32] a príslušným baseline profilom pre XAdES [33].

Aplikácia D.Signer/XAdES Java môže byť použitá taktiež pre vytváranie tzv. obyčajného elektronického podpisu zmysle zákona č. 215/2002 Z.z. o elektronickom podpise.

Na vytvorenie štruktúr zložených elektronických podpisov v súlade s profilmi XAdES_ZEP, resp. XAdES_ZEPbp, prípadne štruktúr elektronických podaní Registration, resp. Message Container môžu byť použité komponenty D.Sig XAdES Extender a ASiC Factory.

V súlade s §4, odsek (5) zákona o e-Gov [36] je aplikácia D.Signer/XAdES Java implementovaná takým spôsobom, aby poskytovala funkcionalitu vytvorenia ZEP/KEP aj pre osoby so zdravotným postihnutím – pre slabozrakých a nevidiacich pomocou technológie NVDA (http://www.nvaccess.org/).

Aplikácia D.Signer/XAdES Java je lokalizovaná v slovenskom a v anglickom jazyku.

5. Systémové požiadavky

Systémové požiadavky aplikácie D.Signer/XAdES Java sú nasledujúce:

- operačný systém MS Windows 7 / 8 / 10 / 11, Mac OS X: verzia 10.12 – 10.15, 11, 12, procesor (architektúra CPU): x86_64, arm (M1), prekladač Rosetta 2 – v prípade procesora arm (M1), GNU/Linux: Mint verzia 13, 17.x, 18, 19.x, 20.0, 20.1, 20.2, 20.3; Debian verzia 8, Mint Debian Edition 4, 5; Ubuntu verzia 12.04 LTS, 14.04 LTS, 16.04 LTS, 18.04 LTS, 20.04 LTS, 21.04, 21.10, 22.04; Fedora: verzia 23, 24, 25, 33, 34, 35, 36; Manjaro 21.0, 21.2.2,
- ak je aplikácia D.Sig XAdES Extender Java spúšťaná z web portálu pomocou aplikácie D.Launcher v1.x:
  - ⇒ Oracle Java 8 (https://www.java.com/en/download/manual.jsp), pozn. kombinácia OpenJDK a IcedTea nie je podporovaná,
  - ⇒ Java plugin do webového prehliadača, Java Web Start a Java FX verzia 2.1 a vyššia (súčasť inštalácie Oracle Java),
- občiansky preukaz s čipom alebo iné certifikované SSCD/QSCD zariadenie pre generovanie kľúčových párov a vytváranie elektronického podpisu + čítačka a ovládače podľa odporúčaní akreditovanej certifikačnej autority (ACA); prípadne PKCS#12 súbor ako úložisko podpisového certifikátu,
- príslušná CSP implementácia MS CryptoAPI (iba MS Windows) alebo implementácia PKCS #11 rozhrania (32 bit / 64 bit podľa platformy Java); súčasť softvéru dodávaného s SSCD/QSCD zariadením,
- web prehliadač podporujúci spúšťanie Java appletov[1] – MS Internet Explorer v7.0 alebo vyššia (len 32 bit), Mozilla Firefox, v45 – v51, resp. v59 ESR (len 32 bit, s podporou NP API), Safari 14, 15,
- prístup na internet (prípadne správne nastavenia pre proxy),
- správne nastavený aktuálny systémový dátum a čas.

Ak je aplikácia D.Signer/XAdES Java spúšťaná z web portálu pomocou aplikácie D.Launcher v1.x, tak požiadavky na web prehliadač zahŕňajú aj prehliadače:

- MS Internet Explorer verzia 10/11 (aj 64 bit), Mozilla Firefox, v45 a vyššia aj 64-bit, Google Chrome v51 alebo vyššia (prípadne Chromium), Opera v38 alebo vyššia, MS Edge v25 alebo vyššia.

V tomto prípade je Java plugin vyžadovaný pre MS Internet Explorer 7/8/9, voliteľný pre MS Internet Explorer 10/11; môže byť nutné ho v prehliadači MS Internet Explorer povoliť pomocou voľby Tools/Manage add-ons. Systémové požiadavky pre aplikáciu D.Launcher sú špecifikované v rámci používateľskej príručky pre aplikáciu D.Launcher.

[1] Ak je aplikácia D.Signer/XAdES Java spúšťaná ako Java applet.

Ak je aplikácia D.Sig XAdES Extender Java spúšťaná z web portálu pomocou aplikácie D.Launcher v2.x a rozšírenia D.Bridge 2:

- webový prehliadač – MS Internet Explorer 11 (len 32bit verzia), Mozilla Firefox 78, 89, 91, 101, Google Chrome 91, 100, 101, Chromium 91, 100, 101, Opera 76, 78, Microsoft Edge 91, 96, 97,
- vo webovom prehliadači nainštalované a povolené rozšírenie D.Bridge 2, pre MS Internet Explorer sa vyžaduje vypnutý chránený režim.

Pri vytváraní zaručeného/kvalifikovaného elektronického podpisu pomocou aplikácie D.Signer/XAdES Java sa vyžaduje použitie certifikovaného zariadenia pre generovanie a uloženie privátneho kľúča a pre vytvorenie zaručeného/kvalifikovaného elektronického podpisu (SSCD/QSCD – napr. čipová karta, USB token apod.) a použitie kvalifikovaného certifikátu, vydaného akreditovanou certifikačnou autoritou. Ak príslušný právny úkon vyžaduje autorizáciu mandátnym kvalifikovaným certifikátom, používateľ musí zvoliť mandátny kvalifikovaný certifikát s príslušným oprávnením pre daný úkon podľa osobitného predpisu. Zoznam zobrazených certifikátov, ktoré sú k dispozícii pre vytvorenie elektronického podpisu, je možné riadiť nastavením filtra certifikátov v okne pre výber certifikátu v aplikácii D.Signer/XADES Java. Aplikácia D.Signer/XAdES Java pristupuje k danému SSCD/QSCD zariadeniu prostredníctvom príslušného CSP providera (implementácia MS Crypto API pre dané SSCD/QSCD zariadenie) alebo prostredníctvom príslušnej implementácie PKCS#11 rozhrania.

Pri vytváraní tzv. obyčajného elektronického podpisu pomocou aplikácie D.Signer/XAdES Java nie je potrebné použiť kvalifikovaný certifikát vydaný akreditovanou certifikačnou autoritou, ani certifikované SSCD/QSCD zariadenie. Použitá podpisová politika by mala jasne deklarovať, o aký elektronický podpis ide.

Veľkosť distribučných súborov jednotlivých komponentov aplikácie D.Signer/XAdES Java je uvedená v nasledujúcej tabuľke.

Komponent	**Veľkosť**
D.Signer/XadES Java	10,5 MB
D.Signer/XAdES Java – XML Plugin	450 kB
D.Signer/XAdES Java – PDF Plugin[2]	11,3 MB (MS Windows) 11,6 MB (GNU/Linux) 20,4 MB (Mac OS X)
D.Signer/XAdES Java – TXT Plugin	40 kB

[2] PDF technology in D.Signer/XAdES Java - PDF Plugin is powered by PDFNet SDK copyright © PDFTron™ Systems Inc., 2001-2016, and distributed by DITEC a.s. under license. All rights reserved.

D.Signer/XAdES Java – PNG Plugin	45 kB

Tzn. že pre konkrétnu platformu (OS, 32/64-bit) je veľkosť distribučných súborov cca 22,5 – 32 MB; ak sú skomprimované, tak dokonca len 15 – 25 MB. Potrebné miesto na serveri pre všetky distribučné súbory (skomprimované aj neskomprimované) pre všetky platformy je spolu cca 300 MB.

Podrobný popis požiadaviek na prevádzku aplikácie D.Signer/XAdES Java, teda požiadaviek na SSCD/QSCD zariadenie, požiadaviek na prevádzkové prostredie aplikácie, bezpečnostných požiadaviek apod. je špecifikovaný v rámci dokumentu Požiadavky na prevádzkové prostredie a SSCD/QSCD.

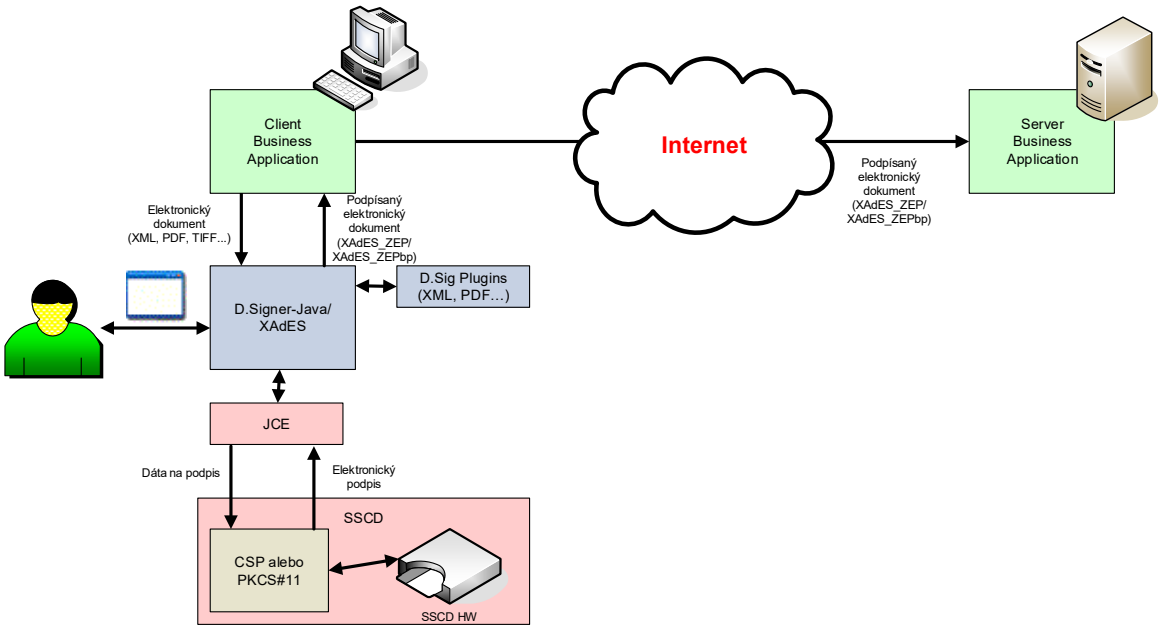
Špecifické systémové požiadavky pre jednotlivé pluginy aplikácie D.Signer/XAdES Java sú uvedené v rámci príslušnej integračnej príručky pre daný plugin.

6. Architektúra aplikácie D.Signer/XAdES Java

V rámci tejto kapitoly je popísaný návrh komponentovej architektúry aplikácie D.Signer/XAdES Java, ktorý vychádza z dokumentov:

- Koncepcia všeobecného formátu XML podpisu a aplikácie SCVA [24]
- CWA14170:2004 E – Security requirements for signature creation applications [21].

6.1. Postavenie aplikácie v rámci prevádzkového prostredia



Aplikácia D.Signer/XAdES Java je realizovaná ako hlavná aplikácia (modul) a sada komponentov (pluginov) pre jednotlivé podporované dátové typy, ktoré môžu byť v súlade s požiadavkami zákazníka nasadené ako súčasť rozsiahlejších aplikácií a informačných systémov napr. pre elektronickú výmenu dokumentov medzi rôznymi subjektami, v rámci ktorých je potrebné zabezpečiť:

- jednoznačnú identifikáciu pôvodcu dokumentu a neodmietnuteľnosť autorstva,
- integritu (prenášaných) dokumentov.

Aplikácia D.Signer/XAdES Java poskytuje pre klientské aplikácie nasledujúce integračné rozhrania – API:

- Java applet API – umožňuje volanie služieb komponentu D.Signer/XAdES Java priamo z prostredia webového prehliadača,
- Java API – umožňuje volanie služieb komponentu D.Signer/XAdES Java z Java aplikácií bežiacich v JRE.

Pre interakciu s podpisovateľom aplikácia D.Signer/XAdES Java poskytuje GUI rozhranie, v rámci ktorého je realizované:

- zobrazenie obsahu podpisovaných dokumentov ako aj všetkých relevantných parametrov ZEP/KEP pred spustením procedúry vytvorenia ZEP/KEP,
- výber kvalifikovaného certifikátu pre vytvorenie ZEP/KEP,
- štandardné ovládacie prvky – potvrdenie procedúry vytvorenia ZEP/KEP, zrušenie procedúry vytvárania ZEP/KEP apod.

Pre kryptografické operácie spojené s výpočtami digitálnych odtlačkov a samotného elektronického podpisu aplikácia využíva:

- implementácie rozhrania Java Cryptography Extension (JCE),
- certifikované SSCD/QSCD zariadenie pre generovanie kľúčových párov a vytváranie elektronického podpisu, ku ktorému pristupuje pomocou CSP implementácie MS Crypto API alebo PKCS#11 knižníc.

6.2. Vnútorná architektúra aplikácie

6.2.1. Funkčný pohľad

Vnútorná architektúra aplikácie D.Signer/XAdES Java vychádza a je v súlade s funkčným komponentovým modelom dokumentu CWA14170:2004 E – Security requirements for signature creation applications [21]. Jednotlivé súčasti aplikácie D.Signer/XAdES Java je teda možné rozdeliť do dvoch skupín:

- dôveryhodné komponenty – povinné komponenty zabezpečujúce základnú požadovanú funkcionalitu SCA,
- aplikačne závislé komponenty – komponenty, ktorých existencia, architektúra a funkcionalita je aplikačne závislá.

Z pohľadu funkčného komponentového modelu SCA sú v rámci aplikácie D.Signer/XAdES Java implementované nasledujúce dôveryhodné komponenty:

- SDP – Signer's Document Presentation Component – zabezpečuje zobrazenie podpisovaných dokumentov podpisovateľovi,
- SAV – Signature Attributes Viewer – zabezpečuje zobrazenie atribútov vytváraného ZEP/KEP podpisovateľovi,
- DTBSF – Data To Be Signed Formatter – zabezpečuje sformátovanie a tranformáciu vstupných dokumentov a ďalších parametrov podpisu do kanonickej formy a vytvorenie štruktúry DTBSF,

- SIC – Signer Interaction Component – rozhranie pre interakciu medzi podpisovateľom a aplikáciou D.Signer/XAdES Java,
- DHC – Data Hashing Component – umožňuje vytvorenie DTBSR z DTBSF pomocou príslušnej hashovacej funkcie,
- SAC – Signer's Authentication Component – umožňuje autentifikáciu podpisovateľa pre použitie SSCD/QSCD zariadenia a na ňom uloženého privátneho kľúča. Dialóg pre zadávanie autentifikačných údajov je zvyčajne realizovaný v rámci CSP alebo PKCS#11 knižnice príslušného certifikovaného SSCD/QSCD zariadenia. Pri prístupe k SSCD/QSCD cez PKCS#11 knižnicu však môže byť pre zadávanie autentifikačných údajov vyvolaný PIN dialóg aplikácie D.Signer/XAdES Java. Pri použití PKCS#12 súboru ako úložiska podpisového certifikátu je pre zadanie hesla pre prístup k PKCS#12 súboru vždy vyvolaný PIN dialóg aplikácie D.Signer/XAdES Java.

Aplikácia D.Signer/XAdES Java obsahuje nasledujúce aplikačne závislé komponenty všeobecnej architektúry SCA:

- SDOC – Signed Data Object Composer – modul pre vytvorenie dokumentu elektronického podpisu vo formáte XAdES_ZEP/XAdES_ZEPbp zo vstupných dokumentov, ďalších vstupných parametrov, DTBSF a vypočítanej hodnoty elektronického podpisu,

Medzi ďalšie implementované súčasti komponentu D.Signer/XAdES Java patria:

- Config Reader – modul pre načítanie konfiguračných údajov aplikácie D.Signer/XAdES Java,

Nasledujúce komponenty netvoria súčasť aplikácie D.Signer/XAdES Java:

- SSA – SCDev/SCA Authenticator – voliteľný modul pre vytvorenie dôveryhodnej cesty medzi aplikáciou D.Signer/XAdES Java a SSCD/QSCD. Keďže aplikácia D.Signer/XAdES Java nesmie byť prevádzkovaná ako verejná služba operátorom, SSA komponent nie je realizovaný,
- SSC – SCDev/SCA Communicator – rozhranie pre komunikáciu medzi aplikáciou D.Signer/XAdES Java a SSCD/QSCD, je realizovaný v rámci CSP alebo PKCS#11 knižnice príslušného certifikovaného SSCD/QSCD zariadenia,
- SDC – Signer's Document Composer – umožňuje podpisovateľovi vytvoriť podpisované dokumenty, má byť realizovaný v rámci klientskej aplikácie,
- SLC – Signature Logging Component. – zabezpečuje vytváranie auditných záznamov o činnosti aplikácie D.Signer/XAdES Java, voliteľný komponent – nie je realizovaný,
- SHI – SCDev Holder Indicator – umožňuje zobraziť meno vlastníka SCDev (SSCD/QSCD) zariadenia, voliteľný komponent – nie je realizovaný.

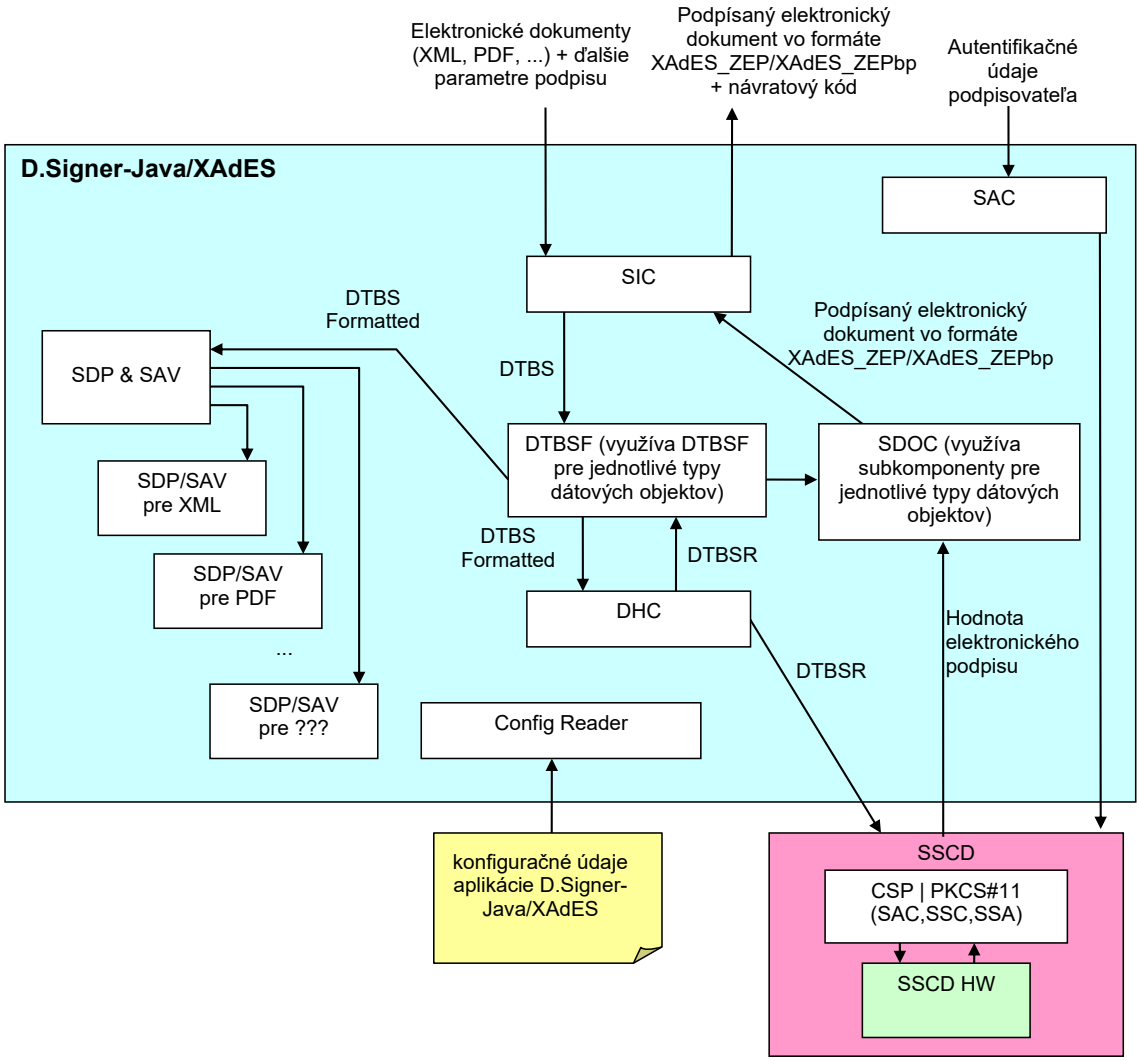
Aplikácia D.Signer/XAdES Java umožňuje vytváranie ZEP/KEP nad komplexnými dátovými štruktúrami, ktoré môžu zahŕňať rôzne typy dátových objektov (XML, PDF, atď.), pričom aplikácia musí byť schopná rozširovania podpory pre nové

typy dátových objektov a jej architektúra musí byť prísne komponentová tak, aby v rámci cieľového prostredia mohli byť nasadené len komponenty (pluginy) s podporou pre relevantné typy dátových objektov.

Z pohľadu rozdelenia funkcionality SCA do samostatných modulov, ktoré je možné pri nasadení aplikácie D.Signer/XAdES Java kombinovať podľa požiadaviek zákazníka, je aplikácia tvorená nasledujúcimi komponentami:

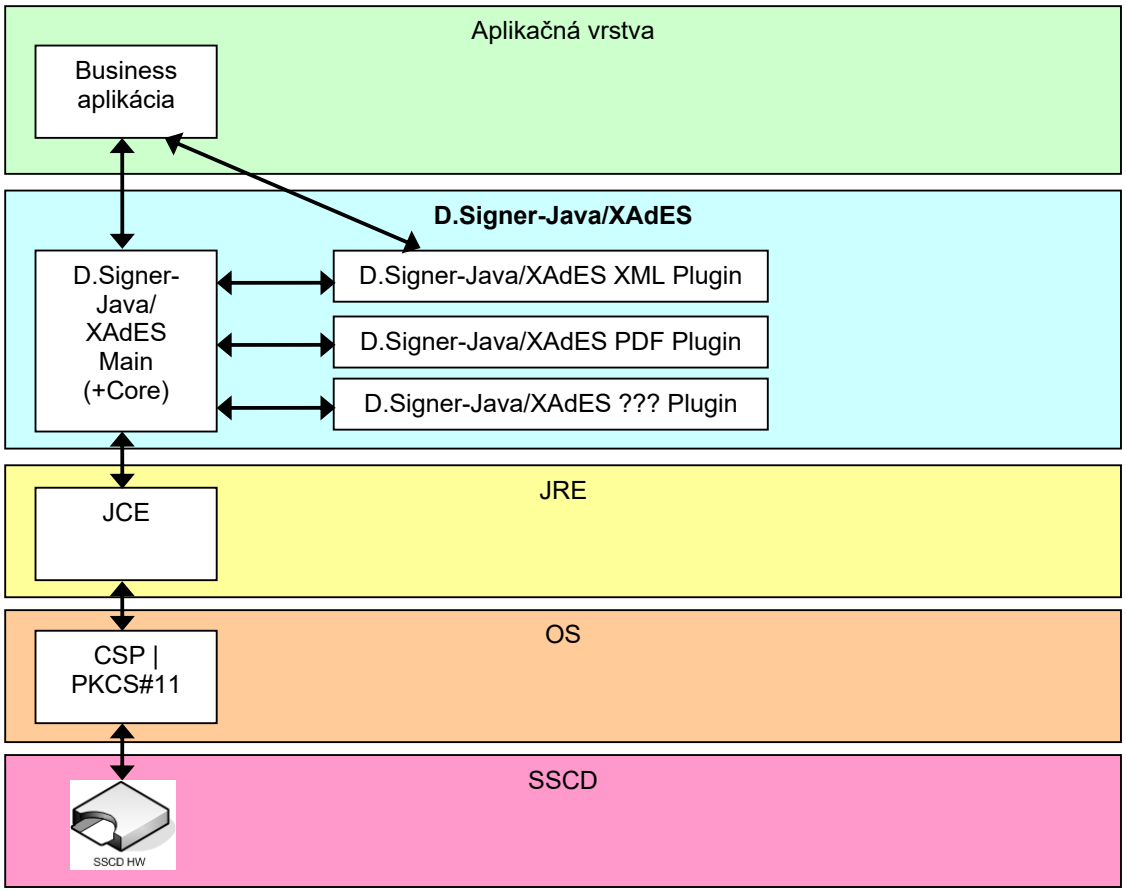
- D.Signer/XAdES Java Main – hlavný modul:
  - ⇒ poskytuje integračné API pre klientské aplikácie,
  - ⇒ spracovanie tých parametrov vytvorenia ZEP/KEP, ktoré nie sú závislé na typoch podpisovaných dátových objektov,
  - ⇒ poskytuje hlavné prezentačné GUI pre podpisovateľa,
  - ⇒ má na starosti vytvorenie ZEP/KEP a formátu podpisu podľa profilov XAdES_ZEP/XAdES_ZEPbp,
  - ⇒ pre svoju činnosť využíva rozhranie pluginov pre jednotlivé typy dátových objektov (vizualizácia, vytvorenie príslušných DTBSF apod.),
- pluginy pre jednotlivé typy dátových objektov – poskytujú funkcie:
  - ⇒ pre spracovanie tých parametrov vytvorenia ZEP/KEP, ktoré sú závislé od typu podpisovaného dátového objektu,
  - ⇒ pre vytvorenie dátových objektov pre podpisované dáta a príslušné verifikačné parametre,
  - ⇒ pre vytvorenie príslušných XML štruktúr pre jednotlivé spracovávané dátové objekty v rámci vytváraného ZEP/KEP podľa profilov XAdES_ZEP/XAdES_ZEPbp,
  - ⇒ pre vizualizáciu daného typu dátového objektu,
- D.Signer/XAdES Java Core – poskytuje funkcie, ktoré sú spoločné pre hlavnú aplikáciu a jednotlivé pluginy (kanonikalizácia XML, výpočet digitálnych odtlačkov apod.)

Na nasledujúcom obrázku je zobrazená bloková schéma dekompozície aplikácie D.Signer/XAdES Java na jednotlivé popísané súčasti a tok informácií medzi jednotlivými komponentami aplikácie.



6.2.2. Pohľad na vrstvy architektúry

Na nasledujúcom obrázku je zobrazený pohľad na jednotlivé vrstvy architekúry aplikácie, ktorá využíva služby vytvárania ZEP/KEP aplikácie D.Signer/XAdES Java, a postavenie komponentu D.Signer/XAdES Java v rámci tejto architektúry.



D.Signer/XAdES Java poskytuje integračné API rozhranie pre aplikačnú vrstvu, teda pre aplikácie, ktoré potrebujú vytvárať ZEP/KEP. Pre svoju činnosť využíva knižnice prostredia Java Runtime Environment a prostredníctvom nich pristupuje k implementácii CSP alebo k PKCS#11 knižnici príslušného SSCD/QSCD zariadenia.

7. Popis činnosti aplikácie

Aplikácia (modul) D.Signer/XAdES Java bude nasadená ako súčasť klientských systémov a aplikácií, v rámci ktorých je potrebné implementovať vytváranie ZEP/KEP. Ak chce klientská aplikácia využívať služby modulu D.Signer/XAdES Java, musí vytvoriť jeho inštanciu. V rámci vytvorenia inštancie modulu prebehne zároveň jeho inicializácia (pozri ďalej).

Následne môže klientská aplikácia pomocou metód integračného API predať modulu D.Signer/XAdES Java vstupné dokumenty a ďalšie parametre, potrebné pre vytvorenie ZEP/KEP. Výsledok procesu vytvorenia ZEP/KEP je klientskej aplikácii prístupný cez funkcie modulu D.Signer/XAdES Java: getErrorMessage a getSignedXmlWithEnvelope, resp. getSignatureWithASiCEnvelope.

Činnosť aplikácie (modulu) D.Signer/XAdES Java pre vytváranie ZEP/KEP je možné popísať nasledovne:

- po vytvorení inštancie modulu D.Signer/XAdES Java klientskou aplikáciou si modul načíta zo svojich konfiguračných dát aktuálnu konfiguráciu a zoznam podporovaných pluginov D.Signer/XAdES Java pre jednotlivé typy dátových objektov,
- klientská aplikácia ďalej vytvorí inštancie jednotlivých pluginov pre požadované dátové typy a pomocou volaní metód pluginov createObject vytvorí príslušné dátové objekty pre jednotlivé vstupné dokumenty, ktoré majú byť podpísané,
- následne klientská aplikácia zavolá pre jednotlivé vytvorené dátové objekty metódu hlavného modulu addObject, ktorá pridá jednotlivé vstupné dátové objekty do kolekcie dátových objektov na podpísanie (DTBS),
- keď sú pripravené všetky dátové objekty na podpis, klienská aplikácia zavolá metódu sign (resp. sign11, sign20, ďalej len sign) hlavného modulu, ktorá vykoná validáciu vstupných dokumentov a ich spracovanie v rámci jednotlivých pluginov na DTBSF (aplikovanie príslušných transformácií, napr. kanonikalizácia)
- zobrazí sa hlavné okno aplikácie D.Signer/XAdES Java, pričom vizualizácia jednotlivých podpisovaných dátových objektov je realizovaná prostredníctvom príslušných funkcií pluginov pre jednotlivé typy dátových objektov,
- používateľ má možnosť si cez GUI aplikácie D.Signer/XAdES Java prezrieť podpisované dátové objekty a ďalšie parametre podpisu,
- v ďalšom kroku používateľ vyberie pomocou GUI podpisový certifikát,
- po výbere certifikátu, modulu pripraví vstupné dáta (ds:SignedInfo) pre výpočet DTBSR a sprístupní objekt zvoleného poskytovateľa pre výpočet digitálneho odtlačku,

- v ďalšom kroku prebehne autentifikácia používateľa pre použitie príslušného SSCD/QSCD zariadenia, na ktorom je uložený privátny kľúč pre zvolený podpisový certifikát. Autentifikácia prebehne podľa nastavení daného zariadenia,
- ak je autentifikácia pre použitie SSCD/QSCD úspešná, SSCD/QSCD výpočíta a vráti modulu D.Signer/XAdES Java hodnotu elektronického podpisu,
- modul D.Signer/XAdES Java následne algoritmicky overí hodnotu elektronického podpisu pomocou JCE knižníc, čím sa zároveň overí dôveryhodná cesta medzi D.Signer/XAdES Java a SSCD/QSCD,
- modul D.Signer/XAdES Java nakoniec vytvorí XML štruktúru podľa profilu XAdES_ZEP, resp. XAdES_ZEPbp a uloží ju do príslušnej premennej SignedXmlWithEnvelope, resp. SignatureWithASiCEnvelope,
- v prípade, že došlo pri vytváraní ZEP/KEP k chybe, modul D.Signer/XAdES Java nastaví návratovú premennú getErrorMessage (hodnota premennej SignedXmlWithEnvelope, resp. SignatureWithASiCEnvelope bude nastavená na prázdny reťazec),
- používateľ následne potvrdí (tlačidlo Ok) alebo zruší (tlačidlo Zrušiť) vytvorenie ZEP/KEP a modul D.Signer/XAdES Java vráti riadenie klientskej aplikácii.

Klientská aplikácia môže následne získať informáciu o výsledku vytvorenia ZEP/KEP pomocou modulu D.Signer/XAdES Java a samotný ZEP/KEP prostredníctvom funkcií getErrorMessage a getSignedXmlWithEnvelope, resp. getSignatureWithASiCEnvelope.

8. Integrácia s klientskými aplikáciami

Funkcionalita SCA je v rámci aplikácie D.Signer/XAdES Java rozdelená do samostatných modulov, ktoré je možné pri nasadení aplikácie kombinovať podľa požiadaviek zákazníka. Aplikáciu D.Signer/XAdES Java tvorí sada JAR knižníc, ktoré poskytujú pre klientské aplikácie nasledujúce integračné rozhrania:

- Java applet API – umožňuje volanie služieb komponentu D.Signer/XAdES Java priamo z prostredia webového prehliadača,
- Java API – umožňuje volanie služieb komponentu D.Signer/XAdES Java z Java aplikácií bežiacich v JRE.

Ak je pre integráciu aplikácie D.Signer/XAdES Java použité Java API, tak pre správne fungovanie logovania je potrebné, aby integrátor poskytol implementáciu SLF4J (https://www.slf4j.org/).

Aby bolo možné postupne budovať podporu pre ďalšie typy dátových objektov, medzi hlavným modulom D.Signer/XAdES Java a pluginmi je navrhnuté abstraktné API, ktoré musí každý plugin implementovať. Hlavný modul komunikuje s jednotlivými pluginmi prostredníctvom tohto rozhrania.

Každý plugin musí navyše definovať triedu pre typ dátového objektu, pre ktorý je určený. Metódy a atribúty tejto triedy sú závislé na type dátového objektu a musia byť definované v samostatnom dokumente špecifikácie daného pluginu.

V nasledujúcich kapitolách sú popísané jednotlivé rozhrania.

8.1. Integračné API hlavnej aplikácie

Hlavný modul aplikácie D.Signer/XAdES Java publikuje pre Java aplikácie nasledujúce rozhranie:

Package:

```
sk.ditec.zep.dsigner.xades
```

Triedu:

```
XadesSig
```

Konštanty:

```
public static final String LANG_SK = "SK";
public static final String LANG_EN = "EN";
```

Metódy a premenné:

```
public void setWindowSize(int width, int height);
public void setSigningTimeProcessing
(     boolean displayGui
,     boolean includeSigningTime
);
public String setLanguage(String language);
public void setCertificateFilter(String filterID);
public void setRevocationChecking(boolean ocspCheck, boolean crlCheck,
String ocspCertIdHashAlgorithm);
public int loadConfiguration(String configsZipBase64);


public int sign
(     String signatureId
,     String digestAlgUri
,     String signaturePolicyIdentifier
);
public int sign
(     String signatureId
,     String digestAlgUri
,     String signaturePolicyIdentifier
,     Callback callback
);


public int sign11
(     String signatureId
,     String digestAlgUri
,     String signaturePolicyIdentifier
,     String dataEnvelopeId
,     String dataEnvelopeURI
,     String dataEnvelopeDescr
);
public int sign11
(     String signatureId
,     String digestAlgUri
,     String signaturePolicyIdentifier
,     String dataEnvelopeId
,     String dataEnvelopeURI
,     String dataEnvelopeDescr
,     Callback callback
);
```

```
public int sign20
(     String signatureId
,     String digestAlgUri
,     String signaturePolicyIdentifier
,     String dataEnvelopeId
,     String dataEnvelopeURI
,     String dataEnvelopeDescr
);
public int sign20
(     String signatureId
,     String digestAlgUri
,     String signaturePolicyIdentifier
,     String dataEnvelopeId
,     String dataEnvelopeURI
,     String dataEnvelopeDescr
,     Callback callback
);
public int addObject(DataObject dataObject);


public byte[] getSignatureTimeStampRequest
(     String reqPolicy
,     String digestAlgUri
);
public byte[] getSignatureTimeStampRequest
(     String reqPolicy
,     String digestAlgUri
,     Long nonce
,     Boolean certReq
,     String extensions
);
public byte[] getSignatureTimeStampRequest
(     String reqPolicy
,     String digestAlgUri
,     Long nonce
,     Boolean certReq
,     Extension[] extensions
);
public String getSignatureTimeStampRequestBase64
(     String reqPolicy
,     String digestAlgUri
);
public String getSignatureTimeStampRequestBase64
(     String reqPolicy
,     String digestAlgUri
,     Long nonce
,     Boolean certReq
,     String extensions
);
```

```
public int createXAdESZepT
(     String tsResponseB64
,     String tsCertB64
);
public int createXAdESZepT
(     byte[] tsResponse
,     byte[] tsCert
);
public String getVersion();


public String getErrorMessage();
public String getSignedXmlWithEnvelope();
public String getSignedXmlWithEnvelopeBase64();
public String getSignedXmlWithEnvelopeGZipBase64();
public Date getSigningTime();
public String getSignerIdentification();
public String getSigningCertificate();


public String getSignedXmlWithEnvelopeAndTimeStamp();
public String getSignedXmlWithEnvelopeAndTimeStampBase64();
public String getSignedXmlWithEnvelopeAndTimeStampGZipBase64();
public byte[] getSignatureTimeStampToken();
public String getSignatureTimeStampTokenBase64();
public String getSignatureTimeStampCert();
public Date getSignatureTimeStampTime();
public String getTSAIdentification();


public Object getPlugin(String dataType);
public int getReturnCode();
public boolean isBusy();
public void reset();
public void installLookAndFeel();
public void installSwingLocalization();
public void installProxySelector();
public static interface Callback
{     public void onClose(XadesSig source);}
```

Package:

```
sk.ditec.zep.dsigner.xades.bp
```

Triedu:

```
XadesBpSig
```

Konštanty:

```
public static final String LANG_SK = "SK";
public static final String LANG_EN = "EN";
```

Metódy a premenné:

```
public void setWindowSize(int width, int height);
public String setLanguage(String language);
public void setCertificateFilter(String filterID);
public void setRevocationChecking(boolean ocspCheck, boolean crlCheck,
String ocspCertIdHashAlgorithm);
public int loadConfiguration(String configsZipBase64);


public int sign
(     String signatureId
,     String digestAlgUri
,     String signaturePolicyIdentifier
,     Callback callback
);
public int addObject(DataBpObject dataObject);


public byte[] getSignatureTimeStampRequest
(     String reqPolicy
,     String digestAlgUri
);
public byte[] getSignatureTimeStampRequest
(     String reqPolicy
,     String digestAlgUri
,     Long nonce
,     Boolean certReq
,     String extensions
);
public byte[] getSignatureTimeStampRequest
(     String reqPolicy
,     String digestAlgUri
,     Long nonce
,     Boolean certReq
,     Extension[] extensions
);
public String getSignatureTimeStampRequestBase64
(     String reqPolicy
,     String digestAlgUri
);
public String getSignatureTimeStampRequestBase64
(     String reqPolicy
,     String digestAlgUri
,     Long nonce
,     Boolean certReq
,     String extensions
);
```

```
public int createXAdESZepBpT
(     String tsResponseB64
,     String tsCertB64
);
public int createXAdESZepBpT
(     byte[] tsResponse
,     byte[] tsCert
);
public String getVersion();


public String getErrorMessage();
public byte[] getSignatureWithASiCEnvelope();
public String getSignatureWithASiCEnvelopeBase64();
public Date getSigningTime();
public String getSignerIdentification();
public String getSigningCertificate();


public byte[] getSignatureAndTimeStampWithASiCEnvelope();
public String getSignatureAndTimeStampWithASiCEnvelopeBase64();
public byte[] getSignatureTimeStampToken();
public String getSignatureTimeStampTokenBase64();
public String getSignatureTimeStampCert();
public Date getSignatureTimeStampTime();
public String getTSAIdentification();


public Object getPlugin(String dataType);
public int getReturnCode();
public boolean isBusy();
public void reset();
public void installLookAndFeel();
public void installSwingLocalization();
public void installProxySelector();
public static interface Callback
{     public void onClose(XadesSig source);}
```

Pozn. Aby sa aj pre inštancie pluginov a triedy XMLDataContainer použilo správne nastavenie jazyka aplikácie D.Signer/XAdES Java, je potrebné vždy najprv vytvoriť inštanciu hlavnej triedy (XadesSig alebo XadesBpSig) aplikácie D.Signer/XAdES Java a až následne vytvoriť inštanciu príslušnej triedy pluginu alebo triedy XMLDataContainer.

Hlavný modul aplikácie D.Signer/XAdES Java publikuje pre webové aplikácie využívajúce Java applet API nasledujúce rozhranie:

Package:

```
sk.ditec.zep.dsigner.xades.applet
```

Triedu:

```
XadesSigApplet extends DtcApplet
```

Konštanty:

```
public static final String LANG_SK = "SK";
public static final String LANG_EN = "EN";
```

Metódy a premenné:

```
public void setWindowSize(int width, int height);
public void setSigningTimeProcessing
(     final boolean displayGui
,     final boolean includeSigningTime
);
public boolean setLanguage
(     final String language
,     final JSObject callback
);
public void setCertificateFilter(String filterID);
public void setRevocationChecking
(     final boolean ocspCheck
,     final boolean crlCheck
,     final String ocspCertIdHashAlgorithm
);
public boolean loadConfiguration
(     final Object configsZipBase64
,     final JSObject callback
);
```

```
public boolean sign
(     final Object signatureId
,     final Object digestAlgUri
,     final Object signaturePolicyIdentifier
,     final JSObject callback
);
public boolean sign11
(     final Object signatureId
,     final Object digestAlgUri
,     final Object signaturePolicyIdentifier
,     final Object dataEnvelopeId
,     final Object dataEnvelopeURI
,     final Object dataEnvelopeDescr
,     final JSObject callback
);
public boolean sign20
(     final Object signatureId
,     final Object digestAlgUri
,     final Object signaturePolicyIdentifier
,     final Object dataEnvelopeId
,     final Object dataEnvelopeURI
,     final Object dataEnvelopeDescr
,     final JSObject callback
);
public boolean addObject
(     final DataObject dataObject
,     final JSObject callback

);


public boolean getSignatureTimeStampRequestBase64
(     final Object reqPolicy
,     final Object digestAlgUri
,     final JSObject callback
);
public boolean getSignatureTimeStampRequestBase64
(     final Object reqPolicy
,     final Object digestAlgUri
,     final Long nonce
,     final Boolean certReq
,     final Object extensions
,     final JSObject callback
);


public boolean createXAdESZepT
(     final Object tsResponseB64
,     final Object tsCertB64
,     final JSObject callback
);
public boolean getVersion(final JSObject callback);
```

```
public boolean getErrorMessage(final JSObject callback)
public boolean getSignedXmlWithEnvelope(final JSObject callback);
public boolean getSignedXmlWithEnvelopeBase64(final JSObject callback);
public boolean getSignedXmlWithEnvelopeGZipBase64
(     final JSObject callback
);
public boolean getSigningTime(final JSObject callback);
public boolean getSignerIdentification(final JSObject callback);
public boolean getSigningCertificate(final JSObject callback);



public boolean getSignedXmlWithEnvelopeAndTimeStamp
(     final JSObject callback
);
public boolean getSignedXmlWithEnvelopeAndTimeStampBase64
(     final JSObject callback
);
public boolean getSignedXmlWithEnvelopeAndTimeStampGZipBase64
(     final JSObject callback
);
public boolean getSignatureTimeStampTokenBase64
(     final JSObject callback
);
public boolean getSignatureTimeStampCert(final JSObject callback);
public boolean getSignatureTimeStampTime(final JSObject callback);
public boolean getTSAIdentification(final JSObject callback);

public boolean getPlugin
(     final String dataType
,     final JSObject callback
);
public boolean getReturnCode(final JSObject callback);
public void reset();
```

Package:

```
sk.ditec.zep.dsigner.xades.bp.applet
```

Triedu:

```
XadesBpSigApplet extends DtcApplet
```

Konštanty:

```
public static final String LANG_SK = "SK";
public static final String LANG_EN = "EN";
```

Metódy a premenné:

```
public void setWindowSize(int width, int height);
public boolean setLanguage
(     final String language
,     final JSObject callback
);
public void setCertificateFilter(String filterID);
public void setRevocationChecking
(     final boolean ocspCheck
,     final boolean crlCheck
,     final String ocspCertIdHashAlgorithm
);
public boolean loadConfiguration
(     final Object configsZipBase64
,     final JSObject callback
);


public boolean sign
(     final Object signatureId
,     final Object digestAlgUri
,     final Object signaturePolicyIdentifier
,     final JSObject callback
);
public boolean addObject
(     final DataBpObject dataBpObject
,     final JSObject callback
);


public boolean getSignatureTimeStampRequestBase64
(     final Object reqPolicy
,     final Object digestAlgUri
,     final JSObject callback
);
public boolean getSignatureTimeStampRequestBase64
(     final Object reqPolicy
,     final Object digestAlgUri
,     final Long nonce
,     final Boolean certReq
,     final Object extensions
,     final JSObject callback
);


public boolean createXAdESZepT
(     final Object tsResponseB64
,     final Object tsCertB64
,     final JSObject callback
);
public boolean getVersion(final JSObject callback);
```

```
public boolean getErrorMessage(final JSObject callback)
public boolean getSignatureWithASiCEnvelopeBase64
(     final JSObject callback
);
public boolean getSigningTime(final JSObject callback);
public boolean getSignerIdentification(final JSObject callback);
public boolean getSigningCertificate(final JSObject callback);


public boolean getSignatureAndTimeStampWithASiCEnvelopeBase64
(     final JSObject callback
);
public boolean getSignatureTimeStampTokenBase64
(     final JSObject callback
);
public boolean getSignatureTimeStampCert(final JSObject callback);
public boolean getSignatureTimeStampTime(final JSObject callback);
public boolean getTSAIdentification(final JSObject callback);


public boolean getPlugin
(     final String dataType
,     final JSObject callback
);
public boolean getReturnCode(final JSObject callback);
public void reset();
```

Tieto triedy XadesSigApplet a XadesBpSigApplet rozširujú Java triedu JApplet, aby bolo možné aplikáciu D.Signer/XAdES Java spúšťať aj z webových aplikácií v rámci webového prehliadača.

8.1.1. Princípy integračného rozhrania Java API

8.1.1.1. Notifikácia o ukončení Java API funkcie pomocou callback

Integračné rozhranie Java API umožňuje volanie služieb komponentu D.Signer/XAdES Java z Java aplikácií bežiacich v JRE. V prípade, že posledným parametrom funkcie Java API sign (resp. niektorej z jej alternatív) je parameter:

- Callback callback,

tak tento parameter určuje, či pri zavolaní funkcie sign bude dialóg aplikácie D.Signer/XadES Java modálny alebo nemodálny. Vstupom je buď trieda, ktorá musí implementovať interface Callback a jeho metódu onClose alebo null. Ak je parameter callback null, tak dialóg bude modálny a funkcia sign skončí až keď používateľ zavrie okno aplikácie D.Signer/XAdES Java. Ak parameter callback nie je null, tak dialóg aplikácie D.Signer/XAdES Java bude nemodálny a funkcia sign skončí okamžite. Po zavretí okna dialógu používateľom D.Signer/XAdES Java zavolá metódu onClose implementácie rozhrania Callback. Výsledok operácie vytvorenia podpisu je možné získať následne volaním funkcií getReturnCode a getErrorMessage.

Pri volaní funkcií Java API rozhrania je možné volať vždy len jednu metódu. Kým nie je v asynchrónnom režime ukončené vykonávanie funkcie sign spätným zavolaním funkcie onClose špecifikovaného callback objektu, nie je možné volať iné funkcie Java API rozhrania okrem funkcie isBusy. V prípade, že toto nastane, funkcia sign vráti chybový kod -11 bez vykonania svojej činnosti, ale už neoznámi svoje ukončenie prostredníctvom príslušnej metódy onClose callback obejktu. Prebiehajúce vykonávanie funkcie sign je možné detegovať pomocou funkcie isBusy (počas vykonávania sign vrácia True).

8.1.1.2. Vizualizácia formátu HTML

Vizulizácia formátu HTML je riešená cez Java FX, ktorý beží v samostatnom vlákne. Ak je ukončené GUI volajúcej aplikácie (t.j. Swing Event Dispatch Thread), tak komponent pre vizualizáciu deteguje jeho ukončenie a ukončí aj vlákno s Java FX. Od tohto okamihu nie je možné používať API aplikácie D.Signer/XAdES Java.

8.1.2. Princípy integračného rozhrania Java applet API

8.1.2.1. Detekcia pripravenosti appletu

Integračné rozhranie Java applet API umožňuje volanie služieb komponentu D.Signer/XAdES Java priamo z prostredia webového prehliadača. Ak je potrebné v rámci web stránky detegovať pripravenosť appletu, tak je potrebné do objektu window priradiť callback funkciu s jedným parametrom a pri nasadzovaní appletu treba pridať parameter onLoadCallbackName s názvom vytvorenej funkcie:

```
<APPLET ....>
    <PARAM name="onLoadCallbackName" value="<meno funkcie">
</APPLET>
```

Po úspešnej inicializácii applet zavolá túto funkciu s hodnotou svojej inštancie.

8.1.2.2. Spracovanie dlhých reťazcov

Z dôvodu spracovania dlhých reťazcov, applet pri svojej inicializácii vloží do objektu window prototyp objektu ditec.WrappedString, ktorý slúži na obaľovanie dlhých (rádovo 1MB) JavaScript reťazcov. V prípade že Applet API predpisuje dátový typ Object, tak skutočný typ môže byť buď JavaScript String (do 1MB) alebo ditec.WrappedString (ľubovoľná dĺžka).

Pre konverziu JavaScript Stringu na ditec.WrappedString je potrebné vytvoriť jeho inštaciu pomocu:

```
var s = new ditec.WrappedString(<hodnota JavaScript retazca>);
```

Pre získanie honoty ditec.WrappedString je potrebné zavolať jeho metódu: `s.str()`.

8.1.2.3. Notifikácia o ukončení Java applet API funkcie pomocou callback

Rozhranie jednotlivých objektov je koncipované ako asynchrónne. Ak funkcia vracia návratovú hodnotu, tak funkcia Java applet API je navrhnutá podľa schémy:

```
public boolean funkcia
(     <parametre zodpovedajucej funkcie v Java API>
,     final JSObject callback
);
```

Posledný parameter musí byť JavaScript objekt reprezentujúci callback funkciu, ktorá je definovaná podľa nasledujúceho vzoru:

```
var callback_object = {
      onComplete : function(value, instace) {
            //value [lubovolny typ]: navratova hodnota volanej metody
            //instance [applet]: instancia apletu, ktory zavolala
callback
      },

      onException : function(msg, stackTrace, instance) {
            //msg [String]: chybova hlaska, ktora vznikla pri volani
metody
            //stackTrace [String]: detial chyby
            //instance [applet]: instancia apletu, ktory zavolala
callback
      }
}
```

Callback funkcia musí byť objekt, ktorý obsahuje metódy onComplete a onException. V prípade, že daná operácia skončila korektne (teda aj v prípade, ak jej návratový kód reprezentuje chybu), je zavolaná metóda onComplete, pričom výsledok operácie je poskytnutý ako parameter value tejto metódy. V prípade neošetrenej vynimky je zavolaná metóda onException s technickými detailami o vzniknutej chybe.

Dátový typ value zodpovedá JavaScript reprezentácii prislušného typu návratovej hodnoty v Jave. Java dátový typ java.lang.String sa vždy mapuje na JavaScript objekt ditec.WrappedString, z ktorého je možné získať jeho hodnotu zavolaním metódy `s.str()`.

Pri volaní funkcií Java applet API rozhrania je možné volať vždy len jednu metódu. Kým nie je ukončené jej vykonávanie spätným zavolaním funkcií onComplete alebo onException callback objektu, nie je možné volať iné funkcie Java applet API rozhrania. V prípade, že toto nastane, funkcia vráti okamžite false bez vykonania svojej činnosti, ale už neoznámi svoje ukončenie prostredníctvom príslušných funckií callback obejktu.

8.1.3. Popis funkcií a premenných API hlavnej aplikácie

8.1.3.1. setWindowSize

Parametre:

- width – šírka okna v pixeloch,
- height – výška okna v pixeloch.

Popis:

Nastavuje veľkosť okna aplikácie D.Signer/XAdES Java.

Štandardná veľkosť okna aplikácie D.Signer/XAdES Java je 600x450 bodov. Metóda umožňuje programovo nastaviť inú veľkosť okna aplikácie. Metóda však nedovolí nastaviť veľkosť okna menšiu ako 450x350 bodov a väčšiu ako je rozlíšenie obrazovky používateľa.

8.1.3.2. setSigningTimeProcessing

Parametre:

- displayGui – ak displayGui = True, používateľ bude mať k dispozícii štandardné Windows GUI, ktoré mu zobrazí aktuálny systémový dátum a čas pre overenie aktuálnej hodnoty systémového času PC a jej prípadnú korekciu pred zahrnutím elementu xades:SigningTime do štruktúry vytváraného elektronického podpisu.
- includeSigningTime – v prípade, že podpisová politika definuje tento element ako povinný alebo ak includeSigningTime = True, tak element xades:SigningTime bude zahrnutý do štruktúry vytváraného elektronického podpisu a nastavený na aktuálnu hodnotu systémového času PC. V tomto prípade bude hodnota elementu xades:SigningTime zobrazená používateľovi takisto v rámci parametrov podpisu.

Popis:

Nastavuje spracovanie elementu xades:SigningTime pri vytváraní elektronického podpisu podľa profilu XAdES_ZEP, v1.1 [26].

Informácia o povinnosti alebo voliteľnosti elementu xades:SigningTime v rámci podpisovej politiky bude používateľovi zobrazená v rámci parametrov podpisu.

Nastavenie includeSigningTime nemá žiadny význam pri vytváraní elektronického podpisu podľa profilu XAdES_ZEP, v1.0 [25] a v2.0 [27], pretože v rámci týchto profilov je element xades:SigningTime povinný.

8.1.3.3. setLanguage

Parametre:

- langCode – kód jazyka aplikácie D.Signer/XAdES Java; povolené hodnoty: LANG_SK, LANG_EN,
- final JSObject callback – viď kapitolu 8.1.2.3.

Návratová hodnota:

Kód nastaveného jazyka aplikácie.

Popis:

Nastaví jazyk aplikácie D.Signer/XAdES Java.

Pri vytvorení inštancie aplikácie D.Signer/XAdES Java sa štandardne jazyk aplikácie nastaví na základe konfigurácie aplikácie; ak sa nedá nastaviť na základe konfigurácie, tak sa nastaví default "SK".

Zavolaním metódy setLanguage je možné vynútiť nastavenie jazyka napr. v súlade s nastavením jazyka v nadradenej aplikácii. Ak však špecifikovaný jazyk v parametri langCode nie je podporovaný, alebo vstupný parameter je null alebo prázdny, tak sa nastaví jazyk na základe konfigurácie aplikácie D.Signer/XAdES Java. Ak sa nedá nastaviť jazyk na základe konfigurácie, tak sa nastaví default "SK". Metóda vráti ako návratovú hodnotu kód nastaveného jazyka.

Metódu je potrebné zavolať pred zobrazením GUI aplikácie – teda pred zavolaním niektorej z metód Sign pre vytvorenie podpisu.

V prípade vynútenia jazyka aplikácie D.Signer/XAdES Java pomocou metódy setLanguage nebude pre používateľa prístupné konfiguračné nastavenie jazyka aplikácie.

8.1.3.4. setCertificateFilter

Parametre:

- filterID – ID filtra certifikátov z konfigurácie.

Popis:

Nastaví filter certifikátov, ktorý bude použitý v okne pre výber certifikátu v procese podpisovania.

V rámci konfigurácie aplikácie D.Signer/XAdES Java môžu byť definované filtre pre rôzne typy certifikátov, napr. kvalifikované certifikáty, mandátne certifikáty a pod. Pri vytvorení inštancie aplikácie D.Signer/XAdES Java sa na základe konfigurácie aplikácie nastaví filter, ktorý má definovanú hodnotu elementu Default = true.

Zavolaním metódy SetCertificateFilter môže nadradená aplikácia riadiť, aké certifikáty budú používateľovi zobrazené v okne pre výber certifikátu v procese podpisovania. Ak nastavený filter znemožňuje výber požadovaného certifikátu na podpisovanie, je možné filtrovanie certifikátov vypnúť nastavením combo boxu na hodnotu "Žiadny filter".

8.1.3.5. setRevocationChecking

Parametre:

- ocspCheck – príznak zapnutia/vypnutia kontroly zneplatnenia podpisového certifikátu pomocou OCSP; povolené hodnoty: true, false,

- crlCheck – príznak zapnutia/vypnutia kontroly zneplatnenia podpisového certifikátu pomocou CRL; povolené hodnoty: true, false,
- ocspCertIDhashAlgorithm – hashovací algoritmus pre položku CertID.hashAlgorithm v OCSP Requeste. Pozn. pomocou tohto algoritmu sa budú vypočítavať hodnoty položiek CertID.issuerNameHash a CertID.issuerKeyHash v OCSP Requeste podľa RFC 6960; povolené hodnoty: OID hashovacieho algoritmu.

Popis:

Pri vytvorení inštancie aplikácie D.Signer/XAdES Java sa kontrola zneplatnenia podpisového certifikátu štandardne nastaví na základe konfigurácie aplikácie; ak sa parametre kontroly zneplatnenia podpisového certifikátu nedajú nastaviť na základe konfigurácie, tak sa:

- príznak zapnutia/vypnutia kontroly zneplatnenia podpisového certifikátu pomocou OCSP nastaví na hodnotu false,
- príznak zapnutia/vypnutia kontroly zneplatnenia podpisového certifikátu pomocou CRL nastaví na hodnotu false,
- hashovací algoritmus pre položku CertID.hashAlgorithm v OCSP Requeste nastaví na default hodnotu SHA1.

Metóda umožňuje nadradenej aplikácii programovo nastaviť parametre kontroly zneplatnenia podpisového certifikátu pred vytvorením elektronického podpisu. Ak sa však parametre kontroly zneplatnenia podpisového certifikátu nedajú nastaviť na základe vstupných parametrov metódy setRevocationChecking, ani na základe konfigurácie, tak platí vyššie uvedený algoritmus.

Metódu je potrebné zavolať pred zobrazením GUI aplikácie – teda pred zavolaním niektorej z metód sign pre vytvorenie podpisu.

V prípade vynútenia nastavení kontroly zneplatnenia podpisového certifikátu pomocou metódy setRevocationChecking nebudú pre používateľa prístupné konfiguračné nastavenia kontroly zneplatnenia podpisového certifikátu.

8.1.3.6. loadConfiguration

Parametre:

- configsZipBase64 – base64 kódovaný ZIP súbor s konfiguračnými súbormi aplikácie D.Signer/XAdES Java,
- final JSObject callback – viď kapitolu 8.1.2.3.

Návratová hodnota:

V prípade úspechu 0; inak chybový kód.

Popis:

Metóda umožňuje nadradenej aplikácii poskytnúť aplikácii D.Signer/XAdES Java jej konfiguračné súbory pre Certificate Filters, Signature Policies a TSA policies, Plugins a Providers (pozri kapitolu 10).

Poskytnuté konfiguračné súbory nebudú uložené na disk, ale len načítané do pamäte aplikácie D.Signer/XAdES Java. Konfiguračné súbory, ktoré sa nenachádzajú v ZIP súbore, budú štandardne načítané z disku používateľa aplikácie pri vytvorení jej inštancie. (Pozn. pri opakovanom volaní metódy loadConfiguration sa aktualizujú v pamäti aplikácie len konfiguračné súbory, ktoré sa nachádzajú v ZIP súbore v parametri configsZipBase64.)

8.1.3.7. sign (trieda XadesSig, resp. XadesSigApplet)

Parametre:

- signatureId – jednoznačné XML Id elementu ds:Signature, povolené znaky: a..z, A..Z, 0..9, „.“ (bodka), „-“ (pomlčka), „_“ (podčiarkovník); XML Id musí začínať písmenom alebo podčiarkovníkom,
- digestAlgUri – identifikátor algoritmu pre výpočet digitálnych odtlačkov v rámci vytváraného elektronického podpisu; nepovinný parameter; ak je null alebo prázdny, použije sa algoritmus špecifikovaný v rámci konfigurácie aplikácie,
- signaturePolicyIdentifier – jednoznačný identifikátor podpisovej politiky použitej pri vytváraní elektronického podpisu,
- Callback callback – viď kapitolu 8.1.1.1.
- final JSObject callback – viď kapitolu 8.1.2.3.

Návratová hodnota:

V prípade úspechu 0; inak chybový kód.

Popis:

Metóda sign spúšťa samotnú procedúru vytvorenia ZEP podľa profilu XAdES_ZEP, v1.0 [25]. Pri zavolaní metódy Sign sa vykonajú nasledujúce činnosti:

- zobrazenie GUI aplikácie D.Signer/XAdES Java,
- spracovanie a vizualizácia všetkých dátových objetov, ktoré boli pridané do kolekcie dátových objektov na podpis, pomocou funkcií príslušných pluginov pre príslušné typy dátových objektov,
- spracovanie a vizualizácia ostatných parametrov vytvárania ZEP (napr. verifikačných údajov),
- umožnenie výberu podpisového certifikátu,
- po výbere podpisového certifikátu používateľom spustenie procedúry pre výpočet hodnoty eletronického podpisu,
- matematické overenie elektronického podpisu pomocou JCE knižnice,
- vytvorenie XML štruktúry ZEP podľa profilu XAdES_ZEP, v1.0 [25].

8.1.3.8. sign11

Parametre:

- signatureId – jednoznačné XML Id elementu ds:Signature, povolené znaky: a..z, A..Z, 0..9, „.“ (bodka), „-“ (pomlčka), „_“ (podčiarkovník); XML Id musí začínať písmenom alebo podčiarkovníkom,
- digestAlgUri – identifikátor algoritmu pre výpočet digitálnych odtlačkov v rámci vytváraného elektronického podpisu; nepovinný parameter; ak je null alebo prázdny, použije sa algoritmus špecifikovaný v rámci konfigurácie aplikácie,
- signaturePolicyIdentifier – jednoznačný identifikátor podpisovej politiky použitej pri vytváraní elektronického podpisu,
- dataEnvelopeId – jednoznačné XML Id elementu xzep:DataEnvelope, povolené znaky: a..z, A..Z, 0..9, „.“ (bodka), „-“ (pomlčka), „_“ (podčiarkovník),
- dataEnvelopeURI – URI atribút elementu xzep:DataEnvelope,
- dataEnvelopeDescr – Description atribút elementu xzep:DataEnvelope,
- Callback callback – viď kapitolu 8.1.1.1.
- final JSObject callback – viď kapitolu 8.1.2.3.

Návratová hodnota:

V prípade úspechu 0; inak chybový kód.

Popis:

Metóda sign11 spúšťa samotnú procedúru vytvorenia ZEP podľa profilu XAdES_ZEP, v1.1 [26]. Pri zavolaní metódy Sign11 sa vykonajú nasledujúce činnosti:

- zobrazenie GUI aplikácie D.Signer/XAdES Java,
- spracovanie a vizualizácia všetkých dátových objetov, ktoré boli pridané do kolekcie dátových objektov na podpis, pomocou funkcií príslušných pluginov pre príslušné typy dátových objektov,
- spracovanie a vizualizácia ostatných parametrov vytvárania ZEP (napr. verifikačných údajov),
- umožnenie výberu podpisového certifikátu,
- po výbere podpisového certifikátu používateľom spustenie procedúry pre výpočet hodnoty eletronického podpisu,
- matematické overenie elektronického podpisu pomocou JCE knižnice,
- vytvorenie XML štruktúry ZEP podľa profilu XAdES_ZEP, v1.1 [26].

8.1.3.9. sign20

Parametre:

- signatureId – jednoznačné XML Id elementu ds:Signature, povolené znaky: a..z, A..Z, 0..9, „.“ (bodka), „-“ (pomlčka), „_“ (podčiarkovník); XML Id musí začínať písmenom alebo podčiarkovníkom,
- digestAlgUri – identifikátor algoritmu pre výpočet digitálnych odtlačkov v rámci vytváraného elektronického podpisu; nepovinný parameter; ak je null alebo prázdny, použije sa algoritmus špecifikovaný v rámci konfigurácie aplikácie,
- signaturePolicyIdentifier – jednoznačný identifikátor podpisovej politiky použitej pri vytváraní elektronického podpisu,
- dataEnvelopeId – jednoznačné XML Id elementu xzep:DataEnvelope, povolené znaky: a..z, A..Z, 0..9, „.“ (bodka), „-“ (pomlčka), „_“ (podčiarkovník),
- dataEnvelopeURI – URI atribút elementu xzep:DataEnvelope,
- dataEnvelopeDescr – Description atribút elementu xzep:DataEnvelope,
- Callback callback – viď kapitolu 8.1.1.1.
- final JSObject callback – viď kapitolu 8.1.2.3.

Návratová hodnota:

V prípade úspechu 0; inak chybový kód.

Popis:

Metóda sign20 spúšťa samotnú procedúru vytvorenia ZEP podľa profilu XAdES_ZEP, v2.0 [27]. Pri zavolaní metódy Sign20 sa vykonajú nasledujúce činnosti:

- zobrazenie GUI aplikácie D.Signer/XAdES Java,
- spracovanie a vizualizácia všetkých dátových objetov, ktoré boli pridané do kolekcie dátových objektov na podpis, pomocou funkcií príslušných pluginov pre príslušné typy dátových objektov,
- spracovanie a vizualizácia ostatných parametrov vytvárania ZEP (napr. verifikačných údajov),
- umožnenie výberu podpisového certifikátu,
- po výbere podpisového certifikátu používateľom spustenie procedúry pre výpočet hodnoty eletronického podpisu,
- matematické overenie elektronického podpisu pomocou JCE knižnice,
- vytvorenie XML štruktúry ZEP podľa profilu XAdES_ZEP, v2.0 [27].

8.1.3.10. sign (trieda XadesBpSig, resp. XadesBpSigApplet)

Parametre:

- signatureId – jednoznačné XML Id elementu ds:Signature, povolené znaky: a..z, A..Z, 0..9, „.“ (bodka), „-“ (pomlčka), „_“ (podčiarkovník); XML Id musí začínať písmenom alebo podčiarkovníkom,
- digestAlgUri – identifikátor algoritmu pre výpočet digitálnych odtlačkov v rámci vytváraného elektronického podpisu; nepovinný parameter; ak je null alebo prázdny, použije sa algoritmus špecifikovaný v rámci konfigurácie aplikácie,
- signaturePolicyIdentifier – jednoznačný identifikátor podpisovej politiky použitej pri vytváraní elektronického podpisu; nepovinný parameter; ak je null alebo prázdny, bude vytvorená tzv. BES forma elektronického podpisu. BES forma elektronického podpisu bude vytvorená bez ohľadu na hodnotu parametra signaturePolicyIdentifier aj v prípade, ak je v rámci konfiguračného súboru s podpisovými politikami (signaturepolicies.xml) nastavený element PresentIdentifierInASiC na hodnotu false, prípadne ak nie je uvedený (viď kapitola 10.1). Pre vytvorenie podpisu vo formáte XAdES_ZEPbp-EPES musí byť hodnota tohto elementu nastavená na true,
- Callback callback – viď kapitolu 8.1.1.1.
- final JSObject callback – viď kapitolu 8.1.2.3.

Návratová hodnota:

V prípade úspechu 0; inak chybový kód.

Popis:

Metóda sign spúšťa samotnú procedúru vytvorenia KEP podľa profilu XAdES_ZEPbp, v1.0 [28]. Pri zavolaní metódy sign sa vykonajú nasledujúce činnosti:

- zobrazenie GUI aplikácie D.Signer/XAdES Java,
- spracovanie a vizualizácia všetkých dátových objetov, ktoré boli pridané do kolekcie dátových objektov na podpis, pomocou funkcií príslušných pluginov pre príslušné typy dátových objektov,
- spracovanie a vizualizácia ostatných parametrov vytvárania KEP (napr. verifikačných údajov),
- umožnenie výberu podpisového certifikátu,
- po výbere podpisového certifikátu používateľom spustenie procedúry pre výpočet hodnoty eletronického podpisu,
- matematické overenie elektronického podpisu pomocou kódu z knižníc Bouncy Castle Crypto,
- vytvorenie štruktúry KEP podľa profilu XAdES_ZEPbp, v1.0 [28].

8.1.3.11. addObject

Parametre:

- dataObject – dátový objekt vytvorený pomocou metódy CreateObject príslušného pluginu aplikácie D.Signer/XAdES Java,
- final JSObject callback – viď kapitolu 8.1.2.3.

Návratová hodnota:

V prípade úspechu 0; inak chybový kód.

Popis:

Umožňuje pridať dátový objekt vytvorený pomocou metódy createObject príslušného pluginu pre daný dátový typ do kolekcie dátových objektov určených na podpis.

8.1.3.12. getSignatureTimeStampRequest, getSignatureTimeStampRequestBase64

Parametre:

- reqPolicy – OID požadovanej TSA politiky; nepovinný parameter – ak je null alebo prázdny, tak sa v štruktúre TimeStampRequest neuvedie,
- digestAlgUri – URI algoritmu digitálneho odtlačku pre výpočet hodnoty messageImprint; nepovinný parameter – ak je null alebo prázdny, tak sa pre výpočet hodnoty messageImprint použije algoritmus digitálneho odtlačku uvedený v príslušnom elemente DefaultTimeStampDigestAlg v rámci konfigurácie TSA policies,
- nonce – veľké náhodné číslo umožňujúce skontrolovať aktuálnosť odpovede vydavateľa časových pečiatok; nepovinný parameter,
- certReq – indikátor, či sa v odpovedi vydavateľa časových pečiatok požaduje podpisový certifikát TSA; nepovinný parameter, default hodnota je true,
- extensions – umožňuje do štruktúry TimeStampRequest zahrnúť ďalšie informácie reprezentované štruktúrou Extensions, ktorá je definovaná v rámci RFC 2459; nepovinný parameter; môže byť špecifikovaný ako pole objektov triedy org.bouncycastle.asn1.x509.Extension alebo ako textový reťazec obsahujúci XML štruktúru v súlade so schémou extensions.v1.0.xsd, ktorá tvorí prílohu tohto dokumentu.
- final JSObject callback – viď kapitolu 8.1.2.3.

Návratová hodnota:

V prípade úspechu štruktúra Time Stamp Request (ako pole bytov, resp. ako base64 kódovaný textový reťazec); inak prázdne pole alebo prázdny reťazec.

Popis:

Metóda umožňuje nadradenej aplikácii získať pre vytvorenú BES/EPES formu podpisu binárnu štruktúru TimeStampRequest (v súlade s RFC 3161 [5]) pre získanie časovej pečiatky od vydavateľa časových pečiatok (TSA).

8.1.3.13. createXAdESZepT, createXAdESZepBpT

Metóda umožňuje nadradenej aplikácii vytvoriť XAdES_ZEP-T, resp. XAdES_ZEPbp-T formu podpisu, teda elektronický podpis s časovou pečiatkou. V prípade úspechu (návratový kód 0) bude T forma podpisu uložená v premennej SignedXmlWithEnvelopeAndTimeStamp, resp. SignatureAndTimeStampWithASiCEnvelope; inak bude do premennej ErrorMessage uložená príslušná chybová hláška.

Parametre:

- tsResponse – štruktúra TimeStampResponse v súlade s RFC 3161 [5],
- tsCert – podpisový certifikát štruktúry Time Stamp Token pre matematické overenie platnosti podpisu časovej pečiatky (povinný v prípade, ak tsResponse neobsahuje podpisový certifikát časovej pečiatky).
- final JSObject callback – viď kapitolu 8.1.2.3.

Návratová hodnota:

V prípade úspechu 0; inak chybový kód.

Popis:

Metóda spracuje štruktúru Time Stamp Response a vykoná nasledujúce kroky:

- validácia syntaxe a štruktúry Time Stamp Token,
- overenie podpisu Time Stamp Token pomocou tsCert,
- overenie hodnoty digitálneho odtlačku v messageImprint (voči hodnote v žiadosti o časovú pečiatku),
- overenie použitého algoritmu digitálneho odtlačku voči množine povolených algoritmov príslušnou TSA politikou,
- pre XAdES_ZEP, v1.x sa kontroluje hodnota elementu xades:SigningTime voči času z časovej pečiatky, pričom deklarovaný čas vytvorenia elektronického podpisu musí byť menší, ako čas z časovej pečiatky,
- rozšírenie BES/EPES formy na XAdES_ZEP-T, resp. XAdES_ZEPbp-T, teda elektronický podpis s pripojenou časovou pečiatkou,
- nastavenie hodnôt výstupných premenných SignedXmlWithEnvelopeAndTimeStamp, resp. SignatureAndTimeStampWithASiCEnvelope a ErrorMessage.

8.1.3.14. getVersion

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

Vráti informáciu o verzii aplikácie D.Signer/XAdES Java.

8.1.3.15. getErrorMessage

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade výskytu chyby v rámci procesu vytvárania ZEP/KEP bude vracať príslušnú chybovú správu uloženú v premennej ErrorMessage.

8.1.3.16. getSignedXmlWithEnvelope, getSignatureWithASiCEnvelope

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP/KEP bude vracať výslednú štruktúru podľa profilu XAdES_ZEP, resp. XAdES_ZEPbp.

8.1.3.17. getSignedXmlWithEnvelopeBase64, getSignatureWithASiCEnvelopeBase64

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP bude vracať výslednú štruktúru podľa profilu XAdES_ZEP, resp. XAdES_ZEPbp zakódovanú do Base64.

8.1.3.18. getSignedXmlWithEnvelopeGZipBase64

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP bude vracať výslednú XML štruktúru vytvorenú podľa profilu XAdES_ZEP skomprimovanú algoritmom gzip v súlade s RFC 1952 a zakódovanú do Base64.

8.1.3.19. getSigningTime

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP/KEP bude vracať hodnotu elementu xades:SigningTime, teda deklarovaný čas vytvorenia podpisu v UTC.

8.1.3.20. getSignerIdentification

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP/KEP bude vracať Distinguished Name položky Subject podpisového certifikátu.

8.1.3.21. getSigningCertificate

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP/KEP bude vracať base64 kódovanú DER formu podpisového certifikátu.

8.1.3.22. getSignedXmlWithEnvelopeAndTimeStamp, getSignatureAndTimeStampWithASiCEnvelope

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP/KEP s časovou pečiatkou bude vracať výslednú štruktúru podpisu s časovou pečiatkou.

8.1.3.23. getSignedXmlWithEnvelopeAndTimeStampBase64, getSignatureAndTimeStampWithASiCEnvelopeBase64

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP/KEP s časovou pečiatkou bude obsahovať výslednú XML štruktúru podpisu s časovou pečiatkou zakódovanú do Base64.

8.1.3.24. getSignedXmlWithEnvelopeAndTimeStampGZipBase64

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP s časovou pečiatkou bude obsahovať výslednú XML štruktúru podpisu s časovou pečiatkou skomprimovanú algoritmom gzip v súlade s RFC 1952 a zakódovanú do Base64.

8.1.3.25. getSignatureTimeStampToken, getSignatureTimeStampTokenBase64

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP/KEP s časovou pečiatkou bude obsahovať štruktúru Time Stamp Token (časovú pečiatku) z Time Stamp Response.

8.1.3.26. getSignatureTimeStampCert

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP/KEP s časovou pečiatkou bude obsahovať podpisový certifikát časovej pečiatky.

8.1.3.27. getSignatureTimeStampTime

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP/KEP s časovou pečiatkou bude obsahovať čas z časovej pečiatky.

8.1.3.28. getTSAIdentification

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

V prípade úspešného vytvorenia ZEP/KEP s časovou pečiatkou bude vracať Distinguished Name položky Subject podpisového certifikátu časovej pečiatky.

8.1.3.29. getPlugin

Parametre:

- String dataType – textový identifikátor typu dátového objektu, ktorý je definovaný v rámci špecifikácie pluginu ako návratová hodnota implementácie funkcie rozhrania IPlugin getTypeName.
- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

Vráti inštanciu pluginu pre príslušný typ dátového objektu.

8.1.3.30. getReturnCode

Parametre:

- final JSObject callback – viď kapitolu 8.1.2.3.

Popis:

Vráti návratový kód výsledku operácie vytvorenia elektronického podpisu. Funkcia je určená na zistenie výsledku operácie vytvorenia elektronického podpisu pri asynchrónnom volaní funkcií sign, sign11 alebo sign20.

8.1.3.31. isBusy

Návratová hodnota:

True/False

Popis:

Vráti true ak je zobrazený dialóg aplikácie D.Signer/XAdES Java. Funkcia je určená na zistenie stavu dialógu aplikácie D.Signer/XAdES Java pri asynchrónnom volaní funkcií sign, sign11 alebo sign20.

8.1.3.32. reset

Inicialziácia aplikácie D.Signer/XAdES Java a všetkých jej premenných pre vytvorenie nového elektronického podpisu.

8.1.3.33. installLookAndFeel

Nastaví pre swing aplikáciu vlastnú grafickú tému firmy DITEC, a.s. Volanie funkcie ovplyvní vzhľad celej swing aplikácie.

Ak je aplikácia D.Signer/XAdES Java spustená z web portálu pomocou knižníc D.Bridge JS, tak je táto metóda zavolaná automaticky.

8.1.3.34. installSwingLocalization

Nastaví slovenskú lokalizáciu štandardných swing dialógov (napr. pre uloženie súboru). Volanie funkcie ovplyvní lokalizáciu štandardných swing dialógov v celej swing aplikácii.

Ak je aplikácia D.Signer/XAdES Java spustená z web portálu pomocou knižníc D.Bridge JS, tak je táto metóda zavolaná automaticky.

8.1.3.35. installProxySelector

Povolí aplikácii D.Signer/XAdES Java spravovať sieťové proxy nastavenia. Volanie funkcie ovplyvní celú aplikáciu.

Ak je aplikácia D.Signer/XAdES Java spustená z web portálu pomocou knižníc D.Bridge JS, tak je táto metóda zavolaná automaticky.

Ak je aplikácia D.Signer/XAdES Java zavolaná z nadradenej Java aplikácie a integrátor túto metódu nezavolá, tak sa pri prístupe na internet (napr. pri

kontrole zneplatnenia podpisového certifikátu) použijú sieťové proxy nastavenia z Javy, alebo z nadradenej Java aplikácie.

8.2. Integračné API pluginu

Pre Java aplikácie musí každý plugin aplikácie D.Signer/XAdES Java publikovať nasledujúce rozhrania:

Package:

```
<názov_package_pluginu>
```

kde <názov_package_pluginu> je skutočný názov package, napr.

```
sk.ditec.zep.dsigner.xades.plugins.xmlplugin
sk.ditec.zep.dsigner.xades.bp.plugins.xmlplugin.
```

Triedu:

```
<názov_triedy_pluginu>
```

kde <názov_triedy_pluginu> je skutočný názov triedy, napr. XmlPlugin, XmlBpPlugin.

Metódy a premenné:

```
DataObject createObject(<parametre>);
String getErrorMessage();
```

resp.

```
DataBpObject createObject(<parametre>);
String getErrorMessage();
```

kde <parametre> sú skutočné parametre metódy createObject pre daný typ dátového objektu. Parametre tejto metódy sú závislé na type dátového objektu, pre ktorý je plugin určený a musia byť definované v samostatnom dokumente špecifikácie daného pluginu.

8.2.1. Popis funkcií a premenných API pluginu

8.2.1.1. createObject

Umožňuje vytvoriť dátový objekt pre daný dátový typ. Parametre tejto metódy sú závislé na type dátového objektu, pre ktorý je plugin určený a musia byť definované v samostatnom dokumente špecifikácie daného pluginu.

Všetky podpisované informácie o dátovom objekte musia byť pred vytvorením podpisu zobrazené používateľovi a pri overení podpisu musia byť overené voči referenčným údajom evidovaným v rámci IS spracovateľa elektronického podpisu.

8.2.1.2. getErrorMessge

V prípade výskytu chyby v rámci procesu vytvárania dátového objektu bude vracať príslušnú chybovú správu uloženú v premennej ErrorMessage.

8.3. Abstraktné API pre pluginy

Každý plugin, ktorý má byť integrovaný ako súčasť aplikácie D.Signer/XAdES Java musí implementovať nasledujúce abstraktné rozhrania.

```
public interface Plugin extends IPlugin {
}

public interface BpPlugin extends IPlugin {
    public BpFileObject getObjectFile();
}
```

ktoré rozširujú nasledujúce rozhranie IPlugin:

```
public interface IPlugin {

      PluginDescriptionAttribute pluginDescription = null;

      AbstractVisualizer getVisualizer(Window owner);

      String getErrorMessage();

      boolean setData
      (     Object data
      ,     String hashAlg
      ,     String envelopeNS
      ) throws PluginDataValidationException;

      String getTypeName();

      String getPluginVersion();

      List<String> getDSObjects();

      List<String> getDSManifests();

      List<String> getXadesDataObjectFormats();

      List<String> getDSReferences();

      void cleanUp();
}
```

Triedy, ktoré implementujú definované abstraktné rozhrania, musia mať zároveň definovaný nasledujúci atribút:

```
public class PluginDescriptionAttribute {

	private String description;

	public PluginDescriptionAttribute(String description) {
		this.description = description;
	}

	public String getDescription() {
		return description;
	}

	public void setDescription(String description) {
		this.description = description;
	}
}
```

Typ návratovej hodnoty funkcie getVisualizer je trieda AbstractVisualizer, ktorá rozširuje triedu JPanel a umožňuje pluginu aplikácie D.Signer/XAdES Java informovať jadro aplikácie, aby zablokovalo podpisovanie, ak sa pri vizualizácii dátového objektu vyskytnú problémy, a teda hrozí riziko, že by podpisovaný dátový objekt nebol správne zobrazený. Jadro aplikácie D.Signer/XAdES Java sleduje (cez listener) nastavenie property SigningAllowed. Property SigningAllowed určuje, či ovládač pre vizualizáciu dovoľuje používateľovi uskutočniť podpis nad dátovým objektom, ktorý zobrazuje. Pokiaľ je true (predvolená hodnota), podpisovanie je povolené, v opačnom prípade je podpisovanie zakázané.

```
public abstract class AbstractVisualizer extends JPanel {

	public static final String SIGNING_ALLOWED_PROPERTY
		= "SigningAllowed";
	protected void setSigningAllowed(boolean signingAllowed);
	public boolean isSigningAllowed();
}
```

8.3.1. Popis metód triedy AbstractVisualizer

8.3.1.1. setSigningAllowed

Nastaví hodnotu property SigningAllowed podľa vstupného parametra signingAllowed.

8.3.1.2. isSigningAllowed

Vráti hodnotu property SigningAllowed.

8.3.2. Popis metód abstraktného API pre pluginy

8.3.2.1. getVisualizer

Parametre:

owner – okno nadradenej inštancie aplikácie D.Signer/XAdES Java (typ Window),

Popis:

Vráti GUI ovládač pre vizualizáciu dát a verifikačných parametrov pre daný typ dátového objektu (typu AbstractVisualizer).

8.3.2.2. getErrorMessage

V prípade výskytu chyby v rámci vykonávania metódy pluginu bude vracať príslušnú chybovú správu (typu String).

8.3.2.3. setData

Parametre:

data – dátový objekt (typ DataObject),

hashAlg – algoritmus pre výpočet digitálneho odtlačku (typ String),

envelopeNS – namespace obálky vytváranej XML štruktúry podpisu, teda XAdES_ZEP, v1.0, v1.1, resp. XAdES_ZEP, v2.0 alebo XAdES_ZEPbp, v1.0 (typ String).

Popis:

Pridá dátový objekt do kolekcie dátových objektov na podpis, spracuje dátový objekt (aplikovanie príslušných transformácií, vytvorenie DTBSF). V prípade úspechu vráti true, inak false.

8.3.2.4. getTypeName

Vráti úplný názov dátového objektu pre dáta a verifikačné parametre pre daný dátový typ (typu String).

8.3.2.5. getPluginVersion

Vráti informáciu o verzii pluginu (typu String).

8.3.2.6. getDSObjects

Vráti zoznam XML štruktúr (typu List<String>) ds:Object pre jednotlivé dátové objekty a príslušné verifikačné údaje z kolekcie dátových objektov daného typu, ktoré sú určené na podpis, v súlade s príslušným profilom XAdES_ZEP.

8.3.2.7. getDSManifests

Vráti zoznam XML štruktúr (typu List<String>) ds:Manifest pre jednotlivé dátové objekty a príslušné verifikačné údaje z kolekcie dátových objektov daného typu, ktoré sú určené na podpis, v súlade s príslušným profilomXAdES_ZEP v1.0, resp. v1.1.

8.3.2.8. getXadesDataObjectFormats

Vráti zoznam XML štruktúr (typu List<String>) xades:DataObjectFormat pre jednotlivé dátové objekty a príslušné verifikačné údaje z kolekcie dátových objektov daného typu, ktoré sú určené na podpis, v súlade s príslušným profilomXAdES_ZEP, resp. XAdES_ZEPbp.

8.3.2.9. getDSReferences

Vráti zoznam XML štruktúr (typu List<String>) ds:Reference do ds:SignedInfo pre jednotlivé dátové objekty a príslušné verifikačné údaje z kolekcie dátových objektov daného typu, ktoré sú určené na podpis, v súlade s príslušným profilomXAdES_ZEP, resp. XAdES_ZEPbp.

8.3.2.10. cleanUp

Obsahom implementácie na strane pluginu by malo byt vyčistenie pamäte od zdrojov, ktoré nemusia byť automaticky uvoľnené Java garbage collectorom.

8.3.2.11. getObjectFile (len rozhranie BpPlugin)

Vráti informácie o podpísanom dátovom objekte a jeho obsah v objekte triedy BpFileObject, ktorá má nasledujúce atribúty:

- byte[] filename – názov súboru,
- private byte[] data – obsah podpísaného dátového objektu,
- private byte[] mediaType - MimeType dátového objektu.

8.4. Príklad použitia

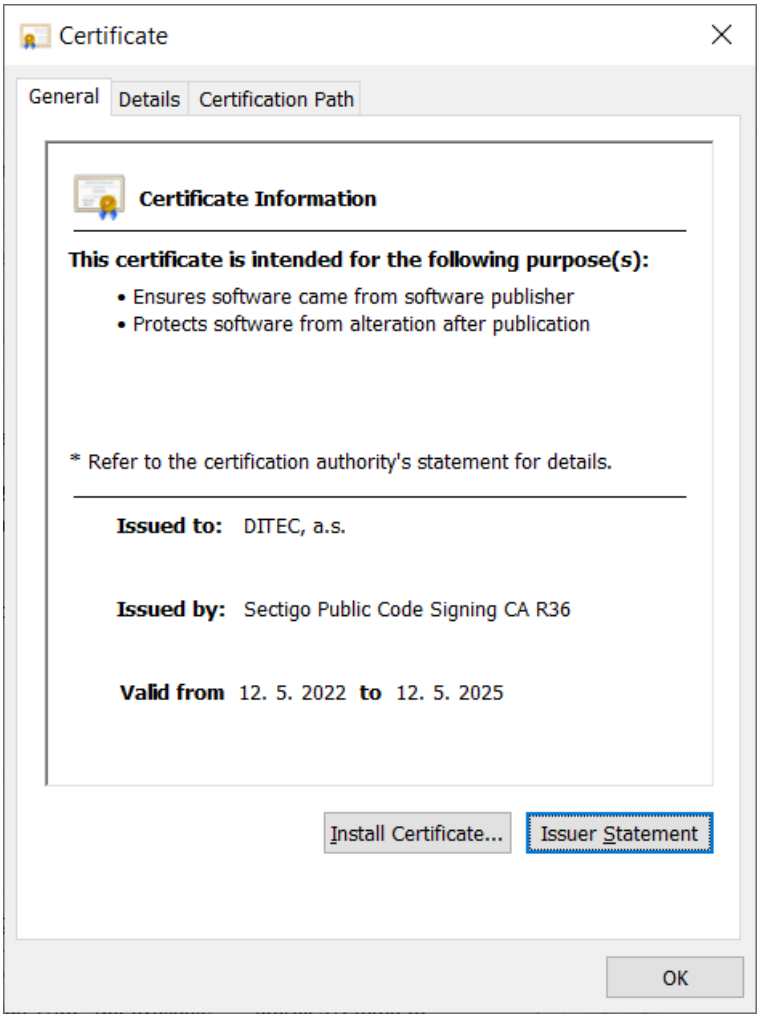
Výrobca aplikácie D.Signer/XAdES Java má k dispozícii pre integrátorov aplikácie tiež sample HTML stránky demonštrujúce použitie komponentu D.Signer/XAdES Java a jednotlivých pluginov pre dátové objekty typu XML, PDF, TXT a PNG ako appletu v rámci webovej aplikácie v jazyku JavaScript.

Pozor! Z príkladov je zrejmé, že aplikácia D.Signer/XAdES Java nie je *thread safe*. Tvorca klientskej aplikácie musí zabezpečiť, že jednotlivé volania funkcií rozhrania aplikácie D.Signer/XAdES Java sú realizované tak, aby nedošlo k vytvoreniu elektronického podpisu nad nesprávnou kombináciou vstupných dokumentov.

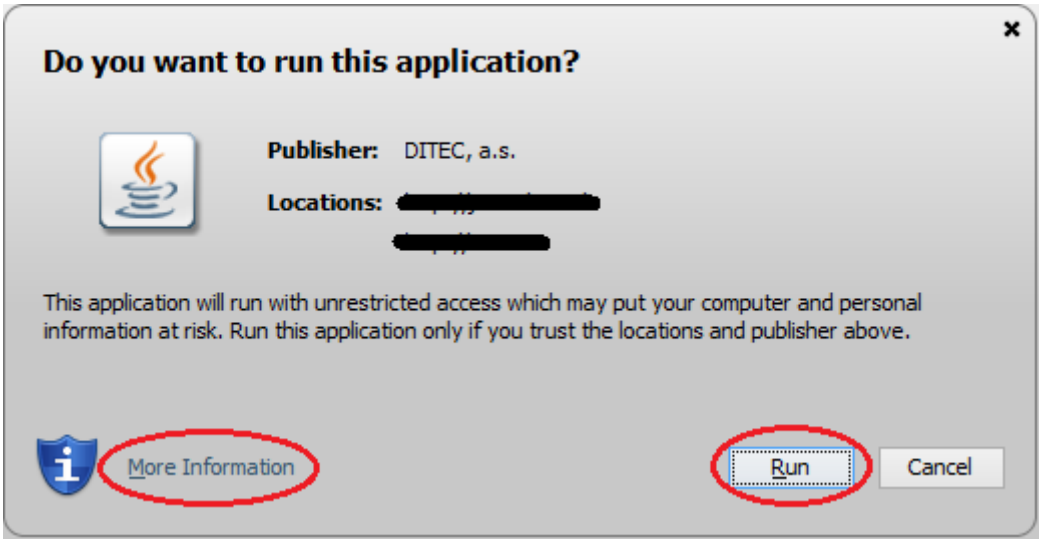
9. Distribúcia a inštalácia

Aplikácia D.Signer/XAdES Java môže byť integrovaná ako applet v rámci web aplikácie alebo ako komponent v rámci klientskej Java aplikácie bežiacej v JRE. Odporúčame, aby distribúcia a inštalácia aplikácie D.Signer/XAdES Java na PC používateľa bola zabezpečená pomocou technológie webstart. V takom prípade integritu súborov aplikácie overuje technológia webstart pri spustení aplikácie. Jednotlivé JAR knižnice sú podpísané certifikátom výrobcu aplikácie (spoločnosť Ditec, a.s.) a je na ne vyžiadaná časová pečiatka.

Na nasledujúcom obrázku je zobrazený náhľad na aktuálny podpisový certifikát spoločnosti DITEC, a.s.



Používateľ si môže skontrolovať podrobnosti a platnosť certifikátu výrobcu kliknutím na link "More information" (prekl. Viac informácií) a potvrdiť spustenie aplikácie kliknutím na tlačidlo "Run" (prekl. Spustiť).



Alternatívnou možnosťou je distribúcia aplikácie D.Signer/XAdES Java spolu s klientskou aplikáciou, v rámci ktorej je integrovaná, z dôveryhodného zdroja na CD médiu v rámci inštalačných súborov klientskej aplikácie. V tomto prípade je integrita súborov aplikácie D.Signer/XAdES Java zabezpečená samotným spôsobom distribúcie.

10. Konfiguračné parametre

Činnosť aplikácie D.Signer/XAdES Java je riadená pomocou konfiguračných parametrov. Tieto konfiguračné parametre zahŕňajú:

- informácie o podporovaných (akceptovaných) podpisových politikách,
- informácie o podporovaných pluginoch pre jednotlivé typy dátových objektov,
- nastavenia filtrov pre podpisové certifikáty,
- informácie o podporovaných (akceptovaných) TSA politikách,
- informácie o poskytovateľoch kryptografických služieb,
- užívateľské nastavenia:
  - ⇒ nastavenie jazyka aplikácie,
  - ⇒ nastavenia kontroly zneplatnenia podpisového certifikátu pred podpisom,
  - ⇒ predvolený spôsob prístupu k SSCD/QSCD a podpisovým certifikátom
  - ⇒ sieťové nastavenia.

10.1. Podpisové politiky

Výrobca, resp. integrátor aplikácie D.Signer/XAdES Java je povinný zabezpečiť také nastavenie konfigurácie aplikácie a parametrov volania metód rozhrania aplikácie, aby aplikácia vytvárala podpis v súlade so špecifikovanou podpisovou politikou.

Informácie o podporovaných podpisových politikách sú distribuované spolu s aplikáciou D.Signer/XAdES Java a uložené v rámci vyhradeného JAR súboru config.jar v XML súbore sk/ditec/zep/dsigner/xades/config/signaturepolicies.xml alebo poskytnuté aplikácii D.Signer/XAdES Java pomocou metódy loadConfiguration. Súbor signaturepolicies.xml musí byť vytvorený v súlade s XML schémou signaturepolicies.v2.0.xsd, ktorá tvorí prílohu tohto dokumentu.

Význam jednotlivých konfiguračných parametrov pre podporované podpisové politiky je nasledujúci:

- PresentIdentifierInASiC – ak je nastavený na false (prípadne ak nie je uvedený), tak sa pri vytváraní podpisu pomocou funkcie Sign (trieda XadesBpSig, viď kapitola 8.1.3.10) vo formáte XAdES_ZEPbp, v1.0 [28] nevloží do podpisu referencia podpisovej politiky a vytvorí sa BES forma podpisu. Pre vytvorenie podpisu vo formáte XAdES_ZEPbp-EPES musí byť hodnota tohto elementu nastavená na true,
- NoSignaturePolicyInfo? – konfiguračné nastavenia pre BES formu podpisu vo formáte XAdES_ZEPbp:

⇒ DefaultDigestAlg – predvolený algoritmus pre výpočet digitálnych odtlačkov v rámci štruktúry podpisu XAdES_ZEPbp vo forme BES,

- SignaturePoliciesInfo* – zoznam podporovaných podpisových politík, v súlade s ktorými je možné pomocou aplikácie D.Signer/XAdES Java vytvárať elektronický podpis; v rámci každej SignaturePolicyInfo budú uvedené nasledujúce položky:
  - ⇒ Identifier – jednoznačný identifikátor podpisovej politiky,
  - ⇒ SigPolicyHash+ – identifikátor algoritmu pre výpočet odtlačku a hodnota odtlačku tejto podpisovej politiky, vypočítaná pomocou špecifikovaného algoritmu a zakódovaná do base64; početnosť 1..n,
  - ⇒ NotBefore – dátum a čas začiatku platnosti podpisovej politiky,
  - ⇒ NotAfter – dátum a čas konca platnosti podpisovej politiky,
  - ⇒ SigningTime – povinnosť zahrnutia elementu xades:SigningTime do vytváraného elektronického podpisu, 1 = xades:SigningTime je v rámci tejto podpisovej politiky povinný element, 0 = xades:SigningTime je v rámci tejto podpisovej politiky voliteľný element, teda bude zahrnutý do podpisu na základe parametrov volania metódy SetSigningTimeProcessing,
  - ⇒ URL – URL, na ktorom je táto podpisová politika k dispozícii, prípadne na ktorom je možné overiť, či podpisová politika nebola predčasne zrušená,
  - ⇒ DefaultDigestAlg – predvolený algoritmus pre výpočet digitálnych odtlačkov v rámci štruktúry podpisu XAdES_ZEP/XAdES_ZEPbp vytvorenej v súlade s danou podpisovou politikou,
  - ⇒ SignerAlgorithmConstraints – obmedzenia pre algoritmy použité podpisovateľom
    - ♦ AlgAndLength*
      - Algorithm – OID algoritmu
      - MinKeyLength? – minimálna dĺžka kľúča.

Pri vytváraní elektronického podpisu si aplikácia D.Signer/XAdES Java načíta informácie o príslušnej podpisovej politike, ktorá bola špecifikovaná pri volaní funkcie sign.

10.2. Podporované pluginy

Informácie o podporovaných pluginoch pre jednotlivé typy dátových objektov sú distribuované spolu s aplikáciou D.Signer/XAdES Java a uložené v rámci vyhradeného JAR súboru config.jar v XML súbore sk/ditec/zep/dsigner/xades/config/plugins.xml alebo poskytnuté aplikácii D.Signer/XAdES Java pomocou metódy loadConfiguration. Súbor plugins.xml

musí byť vytvorený v súlade s XML schémou plugins.v1.0.xsd, ktorá tvorí prílohu tohto dokumentu.:

Význam jednotlivých konfiguračných parametrov pre podporované pluginy je nasledujúci:

- Plugins – zoznam podporovaných pluginov pre jednotlivé typy dátových objektov; pre každý plugin budú uvedené nasledujúce položky:
  - ⇒ ClassName* – plne kvalifikovaný názov triedy príslušného pluginu.

10.3. Nastavenia filtra pre podpisové certifikáty

Aplikácia D.Signer/XAdES Java primárne slúži na vytvorenie zaručeného/kvalifikovaného elektronického podpisu. Zákon č. 215/2002 Z.z. o elektronickom podpise a o zmene a doplnení niektorých zákonov v znení neskorších predpisov však umožňuje použiť v styku s orgánmi verejnej moci aj obyčajný elektronický podpis. Používateľ môže mať na svojom SSCD/QSCD zariadení vygenerovaných viacero kľúčových párov, na ktoré má vystavené kvalifikované certifikáty, mandátne certifikáty s príslušnými oprávneniami podľa osobitných predpisov alebo nekvalifikované certifikáty.

Aby používateľ pri vytvorení elektronického podpisu omylom nepoužil nesprávny typ certifikátu, aplikácia D.Signer/XAdES Java umožňuje konfiguráciu filtrov pre podpisové certifikáty, ktoré sa majú používateľovi zobraziť pri výbere podpisového certifikátu. Predpokladá sa, že nastavenia filtrov certifikátov budú špecifikované integrátorom aplikácie D.Signer/XAdES Java do portálu príslušného orgánu verejnej moci a budú sa teda distribuovať spolu s aplikáciou D.Signer/XAdES Java. Preto aplikácia samotná neposkytuje žiadne GUI pre nastavenie (konfiguráciu) filtrov certifikátov.

Aplikácia D.Signer/XAdES Java umožňuje používateľovi len nastaviť príslušný filter alebo úplne vypnúť filtrovanie certifikátov v okne pre výber podpisového certifikátu.

Nastavenia pre filtre certifikátov budú distribuované spolu s aplikáciou D.Signer/XAdES Java a uložené v rámci vyhradeného JAR súboru config.jar v XML súbore sk/ditec/zep/dsigner/xades/config/certificatefilter.xml alebo poskytnuté aplikácii D.Signer/XAdES Java pomocou metódy loadConfiguration. Súbor certificatefilter.xml musí byť vytvorený v súlade s XML schémou certificatefilter.v2.0.xsd, ktorá tvorí prílohu tohto dokumentu.

S aplikáciou D.Signer/XAdES Java sú distribuované nasledujúce preddefinované filtre certifikátov:

#	ID	Default	Name
1	EUQCForESig[3]	true	sk:Všetky kvalifikované certifikáty pre

[3] Nahradzuje pôvodný filter "SK QC" a zahŕňa certifikáty z filtrov č. 2, 3, 4.

#	ID	Default	Name
			podpis
			en:All qualified certificates for e-signature
2	EUQCForAdESig-QC	false	sk:Len kvalifikované certifikáty pre uznaný spôsob autorizácie
			en:Only qualified certificates for recognized method of authorization
3	EUQCForQESig	false	sk:Len kvalifikované certifikáty pre kvalifikovaný el. podpis
			en:Only qualified certificates for qualified e-signature
4	SK MQC	false	sk.Len mandátne kvalifikované certifikáty
			en:Only mandate qualified certificates
5	EUQCForQESeal	false	sk:Len kvalifikované certifikáty pre kvalifikovanú el. pečať
			en:Only qualified certificates for qualified e-seal

Filter "Len kvalifikované certifikáty pre kvalifikovaný el. podpis" zobrazí len kvalifikované certifikáty vydané pre fyzickú osobu s príznakmi QcCompliance a QcSSCD (uložené na certifikovanom HW zariadení). Na druhej strane filter "Len kvalifikované certifikáty pre uznaný spôsob autorizácie" zobrazí len kvalifikované certifikáty s príznakom QcCompliance, ale bez príznaku QcSSCD. V oboch prípadoch sú pravidlá pre identifikátor fyzickej osoby v atribúte serialNumber (OID 2.5.4.5) položky Subject certifikátu (PNO, IDC, PAS, TIN, TAX) vyhodnocované v súlade s požiadavkami dokumentu NBÚ Schéma dohľadu [38] a štandardu ETSI EN 319 412-1 [39].

Filter "Len mandátne kvalifikované certifikáty" kontroluje v certifikáte okrem príznakov QCCompliance a QcSSCD aj prítomnosť OID certifikačnej politiky NBÚ a prítomnosť OID oprávnenia (1.3.158.36061701.1.1.xyz) v rámci rozšírenia certificatePolicies (OID 2.5.29.32).

Filter "Všetky kvalifikované certifikáty pre podpis" je nadmnožina vyššie uvedených filtrov.

Filter "Len kvalifikované certifikáty pre kvalifikovanú el. pečať" zobrazí len kvalifikované certifikáty vydané pre právnickú osobu s príznakmi QcCompliance a QcSSCD. Ostatné pravidlá pre atribúty položky Subject certifikátu, ako aj pravidlá pre identifikátor právnickej osoby v atribúte organizationIdentifier (OID 2.5.4.97) položky Subject certifikátu (NTR, PSD, VAT, LEI) sú vyhodnocované v súlade s požiadavkami dokumentu NBÚ Schéma dohľadu [38] a štadardu ETSI EN 319 412-1 [39].

Význam konfiguračných parametrov pre jednotlivé filtre podpisových certifikátov je nasledujúci:

- atribút ID – ID filtra certifikátov,
- Name – názov filtra certifikátov,
  - ⇒ atribút lang – jazyk názvu filtra certifikátov,
- Default – indikátor, či má byť v rámci GUI aplikácie D.Signer/XAdES Java filter certifikátov zapnutý/vypnutý; povolené hodnoty: true/false,
- Rules – množina pravidiel pre vyhodnotenie certifikátov; jednotlivé pravidlá (elementy <Rule>) sa vyhodnocujú cez logické OR,
  - ⇒ Rule* – jedno pravidlo pre vyhodnotenie certifikátov; jednotlivé položky pravidla sa vyhodnocujú cez logické AND,
    - ♦ KeyUsage* – certifikát musí mať v rámci KeyUsage nastavenú príslušnú hodnotu (digitalSignature, nonRepudiation, atď.), viď [7],
    - ♦ CertificatePolicyOID* – certifikát musí mať v rámci zoznamu certifkačných politík uvedenú certifikačnú politiku s daným OID,
    - ♦ CertificatePolicyOIDRegex* – v rámci zoznamu certifikačných politík musí vyhovovať danému regulárnemu výrazu OID aspoň jednej certifikačnej politiky,
    - ♦ QCStatementOID* – certifikát musí mať v rámci položky QCStatements uvedené OID príslušného QCStatementu,
    - ♦ SubjectAttrValue* – v rámci poľa Subject certifikátu prevedeného do textovej formy sa musí nachádzať výraz: <názov/OID attribútu DN>=<regulárny výraz>; regulárny výraz sa bude vyhľadávať v hodnote daného atribútu; vyhľadávanie je case-insensitive
    - ♦ IssuerAttrValue* – v rámci poľa Issuer certifikátu prevedeného do textovej formy sa musí nachádzať výraz: <názov/OID attribútu DN>=<regulárny výraz>; regulárny výraz sa bude vyhľadávať v hodnote daného atribútu; vyhľadávanie bude case-insensitive,
    - ♦ ExtendedKeyUsageOID* – certifikát musí mať v rámci rozšírenia Extended Key Usage uvedený príslušný OID, viď [7].

10.4. TSA politiky

Aplikácia D.Signer/XAdES Java umožňuje rozšírenie BES/EPES formy elektronického podpisu na tzv. T formu, teda na elektronický podpis s časovou pečiatkou pomocou metód createXAdESZepT a createXAdESZepBpT. Pre vytvorenie validnej T formy je však potrebné, aby pri spracovaní štruktúry Time Stamp Response a Time Stamp Token boli vykonané kontroly voči konfiguračným nastaveniam vytvoreným v súlade s použitou TSA politikou.

Nastavenia podporovaných TSA politík budú distribuované spolu s aplikáciou D.Signer/XAdES Java a uložené v rámci vyhradeného JAR súboru config.jar v XML súbore sk/ditec/zep/dsigner/xades/config/tsapolicies.xml alebo poskytnuté aplikácii D.Signer/XAdES Java pomocou metódy loadConfiguration. Súbor tsapolicies.xml bude vytvorený v súlade s XML schémou tsapolicies.v1.0.xsd, ktorá tvorí prílohu tohto dokumentu.

Význam jednotlivých konfiguračných parametrov pre podporované TSA politiky je nasledujúci:

- TSAPolicies – zoznam podporovaných TSA politík, v súlade s ktorými je možné pomocou aplikácie D.Signer/XAdES Java vytvoriť elektronický podpis s časovou pečiatkou; v rámci každej TSA politiky budú uvedené nasledujúce položky:
    - ⇒ TSAPolicy* – informácie o podporovanej TSA politike:
        - ♦ TSAPolicyID – OID TSA politiky,
        - ♦ TSAPolicyIDRegex – regulárny výraz pre skupinu OID TSA politík,
        - ♦ NotBefore – dátum a čas začiatku platnosti TSA politiky,
        - ♦ NotAfter – dátum a čas konca platnosti TSA politiky,
        - ♦ MessageImprintAlgorithmConstraints – obmedzenia pre algoritmy digitálnych odtlačkov; algoritmy povolené danou TSA politikou pre položku messageImprint,
        - ♦ DefaultTimeStampDigestAlg – predvolený algoritmus digitálneho odtlačku pre výpočet hodnoty položky messageImprint.

10.5. Poskytovatelia kryptografických služieb

Aplikácia D.Signer/XAdES Java bude umožňovať konfiguráciu informácií o poskytovateľoch kryptografických služieb prostredníctvom PKCS#11 knižníc alebo konfiguráciu prístupu k certifikátom uloženým v rámci PKCS#12 súborov.

Nastavenia podporovaných poskytovateľov kryptografických služieb budú môcť byť distribuované spolu s aplikáciou D.Signer/XAdES Java a uložené v rámci vyhradeného JAR súboru config.jar v XML súbore sk/ditec/zep/dsigner/xades/config/providers.xml alebo budú poskytnuté aplikácii D.Signer/XAdES Java pomocou metódy loadConfiguration. Nastavenia pre poskytovateľov kryptografických služieb definovaných používateľom budú uložené v adresári <homedir>\.ditec\dsigner2\providers v súbore user.xml.

Význam jednotlivých konfiguračných parametrov a informácií o poskytovateľoch kryptografických služieb je nasledovný:

- Providers – zoznam definovaných poskytovateľov kryptografických služieb:
    - ⇒ Provider* – informácie o poskytovateľovi kryptografických služieb:

- Type – typ poskytovateľa kryptografických služieb; povolené hodnoty "PKCS11", "PKCS12",
- Name – názov poskytovateľa kryptografických služieb,
  - atribút lang – jazyk názvu poskytovateľa kryptografických služieb,
- File – cesta k súboru PKCS#11 knižnice alebo k PKCS#12 súboru,
  - atribút os – operačný systém; povolené hodnoty "Windows", "Linux", "Mac OS X"; nemusí byť uvedený,
  - atribút arch – architektúra; povolené hodnoty "x86", "i386", "x86_64", "amd64" (prípadne ich kombinácie oddelené čiarkou); nemusí byť uvedená,
- atribút id – jedinečné ID poskytovateľa kryptografických služieb (napr. GUID alebo reverzné doménové meno).

10.6. Užívateľské nastavenia

10.6.1. Všeobecné nastavenie aplikácie

Všeobecné nastavenia aplikácie D.Signer/XAdES Java zahŕňajú:

- nastavenie jazyka aplikácie,
- nastavenia kontroly zneplatnenia podpisového certifikátu pred podpisom.

Jazyk aplikácie D.Signer/XAdES Java môže používateľ nastaviť z GUI aplikácie D.Signer/XAdES Java. Nastavenie nového jazyka sa aplikuje pri ďalšom spustení aplikácie D.Signer/XAdES Java.

Aplikácia D.Signer/XAdES Java pred vytvorením elektronického podpisu kontroluje, či zvolený podpisový certifikát nebol zneplatnený, resp. revokovaný. V prípade, že aplikácia nemá prístup na internet, je možné túto kontrolu v GUI Nastavenia aplikácie D.Signer/XAdES Java vypnúť.

Všeobecné nastavenie aplikácie D.Signer/XAdES Java sú uložené v rámci súboru <homedir>\.ditec\dsigner2\config2.xml. Súbor config2.xml bude vytvorený v súlade s XML schémou config.v2.0.xsd, ktorá tvorí prílohu tohto dokumentu.

Význam elementov pre nastavenie jazyka v rámci súboru config2.xml je nasledujúci:

- CurrentLanguage – kód jazyka aplikácie D.Signer/XAdES Java; povolené hodnoty: SK, EN,
- RevocationChecking – nastavenia kontroly zneplatnenia podpisového certifikátu pred podpisom:
  - ⇒ OCSPCheck – príznak zapnutia/vypnutia kontroly zneplatnenia podpisového certifikátu pomocou OCSP; povolené hodnoty: true, false,

⇒ CRLCheck – príznak zapnutia/vypnutia kontroly zneplatnenia podpisového certifikátu pomocou CRL; povolené hodnoty: true, false,

⇒ OCSPCertIDhashAlgorithm – predvolený hashovací algoritmus pre položku CertID.hashAlgorithm v OCSP Requeste. Pozn. pomocou tohto algoritmu sa budú vypočítavať hodnoty položiek CertID.issuerNameHash a CertID.issuerKeyHash v OCSP Requeste podľa RFC 6960.

- ♦ AlgOID+ – OID hashovacieho algoritmu,

⇒ ThisUpdateStillAcceptablePeriod – časový úsek v sekundách, počas ktorého je aplikáciou akceptovaná OCSP odpoveď ako platná v prípade, ak OCSP odpoveď nemá uvedený NextUpdate.

10.6.2. Spôsob prístupu k SSCD/QSCD a podpisovým certifikátom

Aplikácia D.Signer/XAdES Java využíva pri vytváraní zaručeného/kvalifikovaného elektronického podpisu certifikované SSCD/QSCD zariadenie pre generovanie kľúčových párov a vytváranie elektronického podpisu, ku ktorému pristupuje pomocou CSP implementácie MS Crypto API alebo príslušnej PKCS#11 knižnice. Zároveň umožňuje vytvoriť aj obyčajný elektronický podpis napr. pomocou certifikátu uloženom v rámci PKCS#12 súboru. Predvolený spôsob prístupu k SSCD/QSCD, resp. k PKCS#12 súboru (a teda aké podpisové certifikáty bude mať používateľ k dispozícii), je uložený v rámci konfiguračného súboru <homedir>\.ditec\dsigner2\config2.xml.

Po vytvorení inštancie modulu D.Signer/XAdES Java sa aplikácia v rámci inicializácie pokúsi načítať nastavenia uložené v súbore config2.xml. Ak takýto súbor neexistuje, tak otvorí používateľovi dialóg, v ktorom mu umožní nastaviť:

- buď prístup k SSCD/QSCD pomocou MS Crypto API – v tomto prípade bude mať používateľ pri výbere certifikátu k dispozícii všetky platné certifikáty uložené v MS Personal Certificate Store, ku ktorým existuje privátny kľúč,
- alebo pomocou PKCS#11 knižnice – používateľ bude môcť špecifikovať cestu k PKCS#11 knižnici, ktorú má nainštalovanú v systéme. V tomto prípade bude mať používateľ pri výbere certifikátu k dispozícii všetky platné certifikáty uložené na príslušnom SSCD/QSCD zariadení, ktoré je prístupné pomocou špecifikovanej PKCS#11 knižnice a ku ktorým existuje privátny kľúč,
- alebo prístup k PKCS#12 (PFX) súboru, ktorý má uložený na disku. V tomto prípade bude mať používateľ pri výbere certifikátu k dispozícii všetky platné certifikáty uložené v špecifikovanom PFX súbore, ku ktorým existuje privátny kľúč.

Na platforme Windows sa dialóg pre nastavenie prístupu k SSCD/QSCD neotvorí, ale sa štandardne nastaví prístup k SSCD/QSCD prostredníctvom MS Crypto API.

Po potvrdení konfigurácie prístupných SSCD/QSCD zariadení a podpisových certifikátov aplikácia D.Signer/XAdES Java vytvorí súbor <homedir>\.ditec\dsigner2\config2.xml v súlade s XML schémou config.v2.0.xsd, ktorá tvorí prílohu tohto dokumentu.

Význam jednotlivých konfiguračných parametrov v rámci súboru config2.xml pre predvolený prístup k SSCD/QSCD zariadeniu a podpisovým certifikátom je nasledujúci:

- ProviderType – spôsob predvoleného prístupu k SSCD/QSCD zariadeniu, resp. k PKCS#12 súboru:
  - ⇒ MSCAPI – aplikácia D.Signer/XAdES Java sprístupní pri výbere certifikátu používateľovi platné podpisové certifikáty z jeho MS Personal Certificate Store, ku ktorým je dostupný privátny kľúč,
  - ⇒ PKCS – aplikácia D.Signer/XAdES Java sprístupní pri výbere certifikátu používateľovi platné podpisové certifikáty zo zvoleného SSCD/QSCD zariadenia, ku ktorému má prístup prostredníctvom špecifikovanej PKCS#11 knižnice alebo platné podpisové certifikáty zo špecifikovaného súboru vo formáte PKCS#12.
- PkcsProviders – informácie o predvolenom a o povolených poskytovateľoch kryptografických služieb:
  - ⇒ DefaultProvider – ID predvoleného poskytovateľa kryptografických služieb,
  - ⇒ DefaultSlot – identifikácia predvoleného PKCS#11 slotu:
    - ♦ Label – popis SSCD/QSCD,
    - ♦ ManufacturerId – ID výrobcu SSCD/QSCD zariadenia,
    - ♦ Model – model SSCD/QSCD zariadenia,
    - ♦ SerialNumber – sériové číslo SSCD/QSCD zariadenia,
  - ⇒ EnabledProviders – zoznam ID povolených poskytovateľov kryptografických služieb,
    - ♦ Provider* – ID povoleného poskytovateľa kryptografických služieb.

Správu prístupných SSCD/QSCD zariadení a podpisových certifikátov je možné vykonávať takisto prostredníctvom GUI z prostredia aplikácie D.Signer/XAdES Java.

10.6.3. Sieťové nastavenia

Konfigurácia sieťových nastavení umožňuje používateľovi správne nastaviť prístup k sieti internet. Možnosti nastavenia prístupu sú nasledujúce:

- automatická detekcia,
- priame spojenie,

- manuálne nastavenie proxy servera,
- automatická konfigurácia proxy pomocou súboru PAC.

Odporúčame ponechať štandardné nastavenie – Automatická detekcia a len v prípade potreby neštandardnej konfigurácie prístupu k sieti internet sa obrátiť na administrátora lokálnej siete, ktorý detaily nastavenia prístupu k sieti internet internet cez proxy server poskytne.

Sieťové nastavenia aplikácie D.Signer/XAdES Java sú uložené v rámci súboru network.xml v adresári %USERPROFILE%\.ditec. Súbor network.xml bude vytvorený v súlade s XML schémou network.v1.0.xsd, ktorá tvorí prílohu tohto dokumentu.

Význam nastavení v rámci súboru network.xml je nasledujúci:

- Proxy – nastavenia proxy:
  - ⇒ Type – typ pripojenia k sieti internet; možné hodnoty: AUTO, DIRECT, MANUAL, PAC,
  - ⇒ Server – IP adresa proxy servera,
  - ⇒ Port – port proxy servera,
  - ⇒ Bypass – výnimky, teda URL adresy, pre ktoré sa nepoužíva proxy,
  - ⇒ PacUrl – URL alebo cesta k súboru PAC,
  - ⇒ Username – predvolené meno používateľa pre proxy,
  - ⇒ Password – predvolené heslo (pozn. šifrované algoritmom AES).

11. Návratové kódy aplikácie

V nasledujúcej tabuľke sú uvedené návratové kódy funkcií sign, sign11 a sign20 (triedy XadesSig a XadesSigApplet) a sign (triedy XadesBpSig a XadesBpSigApplet).

Návratový kód	Popis
0	Volanie funkcie sign (sign11, sign20) skončilo úspešne.
1	Užívateľ stlačil v dialógu aplikácie tlačidlo "Storno".
-1	Neznámy algoritmus digitálneho odtlačku alebo neznáma/neplatná podpisová politika.
-2	Počet objektov na podpis je 0.
-3	Parameter SignatureId je prázdny.
-4	SignatureId nevyhovuje regulárnemu výrazu pre Id.
-5	Nejednoznačnosť vstupných XML Id (v rámci signatureId a objectId kolekcie podpisovaných dátových objektov).
-6	DataEnvelopeId nevyhovuje regulárnemu výrazu pre Id.
-7	DataEnvelopeUri nezodpovedá validnému URI.
-8	Nejednoznačnosť DataEnvelopeId a SignatureId.
-10	Odchytená výnimka v aplikácii. Popis chyby je možné získať pomocou funkcie getErrorMessage.
-11	Funkcia sign (sign11, sign20) je už spustená.
-12	Nepodarilo sa nájsť plugin prislúchajúci danému dátovému typu.
-13	Pred opakovaným volaním funkcie sign() je potrebné zavolať funkciu reset().

V nasledujúcej tabuľke sú uvedené návratové kódy funkcie addObject (všetky triedy).

Návratový kód	Popis
0	Volanie funkcie addObject skončilo úspešne.
-1	Nejednoznačnosť objectId v kolekcii dátových objektov.
-2	Neznámy typ dátového objektu.
-3	Kolekcia pluginov pre dátové objekty je prázdna.

-4	Vstupný objekt je prázdny (null).
-10	Odchytená výnimka v aplikácii. Popis chyby je možné získať pomocou funkcie getErrorMessage.

V nasledujúcej tabuľke sú uvedené návratové kódy funkcie loadConfiguration (všetky triedy).

Návratový kód	**Popis**
0	Načítanie konfigurácie prebehlo úspešne.
-10	Odchytená výnimka v aplikácii. Popis chyby je možné získať pomocou funkcie getErrorMessage.

V nasledujúcej tabuľke sú uvedené niektoré z možných chybových hlášok funkcií getSignatureTimeStampRequest, resp. getSignatureTimeStampRequestBase64.

Chybová hláška
Nieje možné vytvoriť TS request pred úspešným volaním SignXX operácie.
Nemôžem nájsť platnú TSA politiku pre reqPolicy: <OID TSA politiky>
Neznáma digestAlgUri: <URI hashovacieho algoritmu>
Špecifikovaný digestAlgUri parameter nespĺňa požiadavky TSA politiky: <URI hashovacieho algoritmu>

V nasledujúcej tabuľke sú uvedené návratové kódy funkcií createXAdESZepT, resp. createXAdESZepBpT.

Návratový kód	**Popis**
0	Volanie funkcie skončilo úspešne.
-1	TS odpoveď je prázdna
-2	EPES podpis nebol úspešne vytvorený.
-3	Nonce z tsRequest sa nezhoduje s nonce z tsResponse.
-4	Nepodarilo sa pridať časovú pečiatku do podpisu. Popis chyby je možné získať pomocou funkcie getErrorMessage.
-5	Deklarovaný čas vytvorenia elektronického podpisu nie je menší ako čas z časovej pečiatky.
-10	Odchytená výnimka v aplikácii. Popis chyby je možné získať

	pomocou funkcie getErrorMessage.

Ostatné funkcie vrátia v prípade chyby prázdny string, resp. hodnotu Null (v závislosti od typu návratovej hodnoty).

Ďalšie návratové kódy pre jednotlivé pluginy aplikácie D.Signer/XAdES Java sú popísané v príslušných integračných príručkách pre jednotlivé pluginy aplikácie D.Signer/XAdES Java.

12. Trademarks