Opis techniczny modułów JPKLib.DLL i JPKLibUI.DLL wersja 1.1.6.1

Wersja opisu / Wer. modułów	z dnia	Zmiany
w.1.1	4.08.2016	• Dodanie przykładu prostej wysyłki
w.1.2	8.08.2016	• Dodanie metody BreakGetUPO • **Zmiana typu Integer na Int64 w TProgressProc (zmiana wymusza modyfikację w kodzie programu jeśli została zdefiniowana procedura tego typu!)**
w.1.3	9.08.2016	• dodanie rozdziału „Wymagania i ograniczenia”.
w.1.4	12.08.2016	• dodanie metody XMLFileValidate • zmiany w rozdziale „Wymagania i ograniczenia”.
w.1.5 / 1.0.1.7	25.08.2016	• **Zmiana w opisie procedury zwrotnej FinishProc.** • Dodanie metody SignSendBreak.
w.1.6 / 1.0.2.0	27.08.2016	• Dodanie metody XMLValidateBreak, ShowCertList i GetCertCurrentIndex. • Zmiany w rozdziale „Wymagania i ograniczenia” w części dotyczącej ograniczeń metod XMLValidate i XMLFileValidate.
w.1.7 / 1.1.3.1	30.03.2017	• Dodanie do zasobów DLL certyfikatów uwierzytelniających połaczenia z bramką MF JPK i usługą Microsoft Azure. Domyślnie użycie tych certyfikatów z pominięciem certyfikatów z magazynu systemowego. • Umożliwienie poprzez metodę SetCertFile dodawania certyfikatów uwierzytelniania połączeń z bramką MF JPK i usługą Microsoft Azure oraz umożliwienie włączenia korzystania z certyfikatów z magazynu systemowego.
w.1.8 / 1.1.4.1	16.10.2017	• Dodanie metody SetRaportDir.
w.1.9 / 1.1.4.2	2.03.2017	• Usunięcie błędu. Zobacz dodatkowe informacje.
w.1.9 / 1.1.5.1	26.08.2019	• Zaktualizowanie klucza publicznego Ministerstwa Finansów dla JPK (https://www.podatki.gov.pl/komunikaty-techniczne/zmiana-certyfikatu-klucza-publicznego-do-srodowiska-produkcyjnego-jpk/)
w.1.9 / 1.1.6.1	5.11.2019	• Zaktualizowanie klucza publicznego bramki testowej Ministerstwa Finansów dla JPK

w.2.0 / 1.3.1.1	27.10.2020	• Dodanie autoryzacji pliku JPK danymi autoryzacyjnymi. • Rozszerzenie metody SetCertFile o przekazanie ciągu z danymi autoryzacyjnymi. • Rozszerzenie metody GetCertyfikate o odczyt danych autoryzacyjnych.

Spis treści

1. Wstęp

Moduły są przeznaczone do operacji na plikach XML JPK (Jednolitych Plików Kontrolnych) dla wymaganych przez Ministerstwo Finansów w Polsce.

Udostępniają metody służące do walidacji i wysyłania XML oraz odbioru UPO (Urzędowego Potwierdzenia Odbioru).

Biblioteka JPKLibUI.DLL udostępnia dodatkowo metody walidacji XML oraz formatki interfejsu użytkownika takie jak:

- okno parametrów połączenia HTTP i HTTPs (ustawienia serwera proxy),
- okno wyboru podpisu kwalifikowanego,
- okno postępu wysyłania.

Wbudowany „User Interface” ułatwia programiście użycie modułu.

*2. Wymagania i ograniczenia (wersja modułu **1.0.2.0**)*

- Moduł wymaga systemu Windows XP, Vista, 7, 8 i 10.
- Moduł nie wymaga instalowania dodatkowych modułów.
- Metoda **AddXMLJPKFile**:
    - **Wersja 32bit**: Użycie metody AddXMLJPKFile gwarantuje wysłanie pliku XML dowolnej wielkości. Test wysłania pliku o wielkości 5GB wypadł pomyślnie. Czas procesu wysyłki (kompresja, podpisanie, wysyłanie) tak dużego pliku wyniósł ok. 6 minut. Czas wysyłki pliku 1,5GB wyniósł 1,5 minuty.
    - **Wersja 64bit**: Jak w wersji 32bit – wielkość pliku bez ograniczeń.
- Metoda **AddXMLJPK**: Należy pamiętać o odpowiednich zasobach komputera. Potrzeba dwukrotnie więcej RAM niż wielkość pliku.
    - **Wersja 32bit**: Wielkość danych przekazanych metodą AddXMLJPK jest ograniczona w wersji 32 bitowej do 2GB.
    - **Wersja 64bit**: Wielkość danych przekazanych metodą AddXMLJPK jest ograniczona tylko ilością wolnej pamięci operacyjnej.
- Metoda **XMLFileValidate:**
    - **Wersja 32bit**: Wielkość testowanego pliku XML bez ograniczeń.
    - **Wersja 64bit:** Wielkość testowanego pliku XML bez ograniczeń.
- Metoda **XMLValidate:**
    - **Wersja 32bit**: Wielkość danych do testowania jest ograniczona w wersji 32 bitowej do 2GB.
    - **Wersja 64bit**: Wielkość danych jest ograniczona jedynie ilością pamięci operacyjnej.

3. Typy parametrów metod:

typ	Wielkość (bajty)		opis
	Win 32	Win 64	
boolean	1	1	0 – false, 1 – true.
byte	1	1	Bajt.
integer	4	4	Liczba całkowita ze znakiem.
nativeUInt	4	8	Liczba całkowita bez znaku.
Int64	8	8	Liczba całkowita ze znakiem.
pointer	4	8	Wskaźnik.
PChar	4	8	Wskaźnik do ciągu znaków unicode (2 bajty na znak) zakończonego znakiem 0000h. **UWAGA!** Wskaźniki PChar zwracane przez metody DLL wskazują na obszar pamięci zarezerwowany przez DLL. Aby zapamiętać ciąg tekstowy należy utworzyć lokalny obszar pamięci i skopiować do niego ciąg znaków.
PAnsiChar	4	8	Wskaźnik do ciągu znaków ansi (1 bajt na znak) zakończonego znakiem 00h.
TProgressProc	4	8	Wskaźnik do procedury typu TProggressProc dostarczonej od klienta.
TFinishProc	4	8	Wskaźnik do procedury typu TFinishProc dostarczonej od klienta.
AUTHDATA			Łańcuch znaków zawierający dane autoryzacyjne (nip lub pesel, pierwsze imię, nazwisko, datę urodzenia i kwotę).

4. Metody JPKLib.DL i JPKLibUI.DLL.

Metoda	Przeznaczenie
Licencja	
SetLic	Wpisanie numeru licencji. Bez numeru licencji wysyłanie XML nie będzie działało.
Parametry połączenia. Należy ustawić parametry połączenia HTTP i HTTPS jeśli ustawiono serwer proxy i parametry połączenia są inne niż domyślnie ustawione w przeglądarce Microsoft'u.	
SetProxyParamsHTTP	Ustawienia parametrów połączenia protokołu HTTP.
SetProxyParamsHTTPS	Ustawienia parametrów połączenia protokołu HTTPS.
InternetSettingsForm (tylko JPKLibUI.DLL)	Otwarcie okna z edycją parametrów połączeń HTTP i HTTPS.
Inne parametry	

SetProgressCaption (tylko JPKLibUI.DLL)	Ustawienie nazwy dla okna progress bar.
SetParentWindowHandle (tylko JPKLibUI.DLL)	Ustawienie uchwytu okna mającego być parent-em dla niektórych okien modułu JPKLibUI.DLL.
SetRaportDir	Ustawienie nazwy katalogu dla plików raportu.
Certyfikatu (podpisy)	
GetCertyficateCount	Ilość dostępnych zarejestrowanych podpisów.
GetCertyficate	dostęp do danych opisowych podpisu.
SetCertFile	Ustawienie wybranego podpisu.
ShowCertList	Otwiera okno z wyborem podpisów elektronicznych zarejestrowanych w Windows.
GetCertCurrentIndex	Oddaje index wybranego podpisu.
Wysyłanie	
AddXMLJPK	Dodanie danych XML.
AddXMLJPKFile	Dodanie nazwy pliku XML.
SignSendJPK	Podpisanie XML i wysłanie.
SignSendBreak	Przerywa wysyłanie zapoczątkowane SignSendJPK. Umożliwia przerwanie niezależnie od użycia metod TProgressbar.
WaitForEnd	Oczekiwanie na zakończenie wysyłania.
GetReferenceNumbers	Odczytanie numerów referencyjnych wysłanych dokumentów.
GetUPO	Pobranie UPO.
BreakGetUPO	Przerwanie pobierania UPO.
Obsługa błędów	
JPKGetLastError	Pobranie ostatniego błędu.
Walidacja	
FileNameValidate (tylko JPKLibUI.DLL)	Walidacja nazwy pliku XML
XMLValidate (tylko JPKLibUI.DLL)	Walidacja poprawności danych XML - JPK
XMLFileValidate (tylko JPKLibUI.DLL)	Walidacja poprawności pliku XML - JPK

XMLValidateBreak (tylko JPKLibUI.DLL)	Przerwanie procesu walidacji.

procedure SetLic(licText:PChar); stdcall;

Procedura SetLic służy do wpisania tekstu będącego licencją na wykorzystanie modułu.

Bez prawidłowego numeru licencji moduł nie pozwala na wysłanie XML do bramki docelowej, a przy wysyłce do bramki testowej wyświetla komunikat o braku licencji.

- **LicText**: PChar – tekst z numerem licencji.

procedure SetProxyParamsHTTP (UserName,Password,Serwer: PChar; Port:Integer); stdcall;

Procedura ustawia nazwę użytkownika, hasło, adres serwera proxy i numer portu dla połączenia HTTP.

- **UserName**: PChar – nazwa użytkownika proxy.
- **Password**: PChar – hasło użytkownika dostępu do proxy.
- **Serwer**: PChar – tekst z adresem IP serwera proxy.
- **Port**: Integer – numer portu serwera proxy.

procedure SetProxyParamsHTTPS (UserName,Password,Serwer: PChar; Port:Integer); stdcall;

Procedura ustawia nazwę użytkownika, hasło, adres serwera proxy i numer portu dla połączenia HTTPS.

- **UserName**: PChar – nazwa użytkownika proxy.
- **Password**: PChar – hasło użytkownika dostępu do proxy.
- **Serwer**: PChar – tekst z adresem IP serwera proxy.
- **Port**: Integer – numer portu serwera proxy.

procedure InternetSettingsForm(ParentWnd:NativeUInt); stdcall;

(tylko JPKLibUI.DLL)

Procedura otwierająca okno do edycji parametrów połączenia HTTP i HTTPS.

Zobacz: Wygląd formatek

Ustawione parametry będą zapisane w rejestrze windows w gałęzi **HKEY_CURRENT_USER\Software\IPSPI\Internet** w kluczu **Settings**. Stąd także moduł JPKLibUI.DLL (tylko) je odczyta po inicjalizacji.

- ParentWnd: Cardinal – uchwyt okna parent'a.

function SetProgressCaption (SessionID:Pointer; Caption:PChar):Boolean; stdcall;

(tylko JPKLibUI.DLL)

Funkcja ustawia nazwę, która będzie widoczna w pasku postępu wysyłania (progress bar).

Zobacz: Wygląd formatek

- **SessionID**: Pointer – identyfikator sesji otrzymany w Result funkcji SignSendJPK.
- **Caption**: PChar – Ustawiany tekst.
- **Result**: Boolean – false w przypadku niepowodzenia, np. podania nieprawidłowego SessionID.

procedure SetParentWindowHandle(Handle:NativeUInt); stdcall;

(tylko JPKLibUI.DLL)

Procedura ustawia domyślne okno parent'a dla wyskakujących okien wyboru certyfikatu i progress bar.

- **Handle**: NativeUInt – uchwyt okna parent'a.

procedure SetRaportDir(Dir:PChar); stdcall;

Procedura ustawia katalog do którego będą zapisywane pliki z kolejnych etapów przetwarzania: po kompresji, po podpisaniu i podpisany nagłówek XML.

Do nazwy pliku JPKXML będą dodawane rozszerzenia: .zip, .zip.aes. Plik nagłówka ma nazwę Initupload_.xml

- **Dir**: PChar – nazwa katalogu. Jeśli NIL to pliku raportu nie będą zapisywane. Katalog musi istnieć, w przeciwnym wypadku pliku nie będą zapisywane.

function GetCertyficateCount:Integer; stdcall;

Funkcja oddające ilość dostępnych w systemie zarejestrowanych certyfikatów mogących służyć jako podpis.

- **Result**:Integer – ilość certyfikatów

function GetCertyficate(Index:Integer; out cert:PChar):Boolean; stdcall;

Funkcja oddająca listę danych tekstowych certyfikatu.

- **Index** – indeks dostępnego certyfikatu w zakresie **od 0 do GetCertyficateCount-1** lub **-1** w celu nadanych wcześniej odczytania danych autoryzacyjnych**.**
- Out **cert**: PChar – zwracany wskaźnik do ciągu tekstowego:

  a) dla index >= 0 i index< GetCertyficateCount zawierającego listę w formacie „ini" danych o certyfikacie.
  Dostępne dane w liście:

  - Name=nazwa właściciela podpisu. Od wersji modułu 1.0.1.8 obok nazwy mogą być dodane w nawiasach informacje o nieważności podpisu lub ewentualnych innych błędach.
  - NameFull=pełne dane właściciela podpisu w formacie CSV.
  - Issuer=nazwa wystawcy podpisu.
  - IssuerFull=pełne dane wystawcy podpisu w formacie CSV.
  - DateOf=data początku okresu obowiązywania podpisu.
  - DateTo=data końca okresu obowiązywania podpisu.
  - IsQualified=czy podpis kwalifikowany (1-tak).

    Przykład:

    Caption=Marcin Konopka >>>(Błąd)[Podpis nieważny]
    Name=Marcin Konopka
    NameFull=Kraj = PL,Nazwa powszechna = Marcin Konopka,Imię = Marcin Sławomir
    Issuer=CERTUM QCA
    IssuerFull=Kraj = PL,Organizacja = Unizeto Technologies S.A.,Nazwa powszechna = CERTUM QCA,Numer seryjny = Nr wpisu: 1
    DateOf=2014-12-23
    DateTo=2016-12-22
    IsQualified=1

  b) dla index=-1 zawierającego dane autoryzacyjne w formacie AUTHDATA, które zostały wcześniej wpisane metodą SetCertFile lub wpisane poprzez interfejs użytkownika (tylko JPKLibUI.DLL).

- **Result**: Boolean – True – jeśli certyfikat jest dostępny. W przeciwnym wypadku Result=False

procedure SetCertFile(CertFileNameOrIndexOrAuthData,CertPass:PChar); stdcall;

Procedura ustawia indeks certyfikatu służącego do podpisania XML lub dodaje certyfikaty uwierzytelniania połączeń.

- **CertFileNameOrIndexOrAuthData**: PChar -
  1. Tekst z numerem indeksu w zakresie od 0 do GetCertyficateCount-1**.**
  2. Nazwa pliku z certyfikatem do uwierzytelniania połączenia z bramką MF JPK i usługą Microsoft AZURE.
  3. Dane autoryzacyjne AUTHDATA. W tym przypadku parametr CertPass musi być nil;

- **CertPass**: PChar –
  1. hasło certyfikatu podpisu**.**
  2. identyfikator rodzaju połączenia dla którego jest dodawany plik certyfikatu: UWAGA! Dodanie certyfikatu wyłącza dla określonego połączenia użycie certyfikatu z zasobów modułu DLL! Użyte zostaną tylko dodane certyfikaty.
     - HTTPSCERT – dla połączenia z usługą Microsoft Azure (plik certyfikatu w formacie tekstowym Base64 - „PEM")
     - MFCERT – dla połączenia z usługą REST bramki MF JPK (plik certyfikatu w formacie tekstowym Base64 - „PEM")
  3. Identyfikator wymuszający użycie certyfikatów z systemowego magazynu certyfikatów. **UWAGA! Włączenie użycia systemowych certyfikatów wyłącza dla określonego połączenia użycie certyfikatów z zasobów modułu DLL.** Oprócz certyfikatów systemowych mogą zostać użyte dodane certyfikaty.
     - HTTPS_USESYSTEMCERT - dla połączenia z usługą Microsoft Azure
     - MF_USESYSTEMCERT – dla połączenia z usługą REST bramki MF JPK.

Uwagi

(tylko JPKLibUI.DLL) Jeśli certyfikat nie będzie ustawiony to w procesie wysyłki (SignSendJPK) zostanie otwarte okno wyboru certyfikatów.

function ShowCertList(ParentHandle:NativeUInt):Integer; stdcall;

Funkcja umożliwia wybór certyfikatu służącego do podpisania XML zarejestrowanego w magazynie certfikatów Windows. Funkcja otwiera okno wyboru certyfikatu.

- **ParentHandle**: NativeUInt – Uchwyt do okna nadrzędnego lub 0.
- **Result**: Integer – Indeks wybranego certyfikatu lub -1 w przypadku rezygnacji z wyboru.

function GetCertCurrentIndex:Integer; stdcall;

Funkcja oddaje index aktualnego wybranego certyfikatu, o ile został wybrany przez użytkownika z listy certyfikatów lub ustalony metodą SetCertFile.

- **Result**: Integer – Index aktualnego certyfikatu lub -1 jeśli nie ustalony.

function AddXMLJPK (XML:PAnsiChar; DocType:Byte; DefFileName:PChar=nil):Integer; stdcall;

Funkcja dodaje dane XML do walidacji i wysłania.

UWAGA! Zobacz Wymagania i ograniczenia.

- **XML**: PAnsiChar – dane XML
- **DocType**: Byte – 0 – **JPK** (dokumenty przesyłane cyklicznie), 1 – **JPKAH** (dokumenty przesyłane na żądanie)
- **DefFileName**: PChar – Nazwa pliku pod którą dane zostaną przesłane. W przypadku niepodania tego parametru moduł podstawi nazwę *XMLFile_%n.XML*, gdzie %n o numer kolejny pliku w sesji wysyłki.
- **Result**: Integer – 0 – jeśli dane zostaną dodane prawidłowo, -1 w przypadku niepowodzenia. Tekst błędu można odczytać funkcją JPKGetLastError.

function AddXMLJPKFile (XMLFileName:PChar; DocType:Byte):Integer; stdcall;

Funkcja dodaje dane XML z zewnętrznego pliku.

UWAGA! Zobacz Wymagania i ograniczenia.

- **XMLFileName**: PChar – pełna ścieżka z nazwą pliku na dysku.
- **DocType**: Byte – 0 – **JPK** (dokumenty przesyłane cyklicznie), 1 – **JPKAH** (dokumenty przesyłane na żądanie)
- **Result**: Integer – 0 – jeśli plik zostanie dodany prawidłowo, -1 w przypadku niepowodzenia. Tekst błędu można odczytać funkcją JPKGetLastError.

function SignSendJPK (TestMode:Boolean; Progress:TProgressProc; Finish:TFinishProc; SyncFinish:Boolean):Pointer; stdcall;

Funkcja podpisuje i wysyła dane XML dodane poprzez AddXMLJPK lub AddXMLJPKFile.

Wywołanie uruchamia wątek działający w tle. Aby uzyskać wynik działania należy utworzyć i dostarczyć w parametrze Finish procedurę typu TFinishProc lub poczekać na zakończenie używając metody WaitForEnd.

- **TestMode**: Boolean – jeśli true dane zostaną wysłane na bramkę **testową** JPK, false – na bramkę produkcyjną.
- **Progress**: TProgressProc – Wskaźnik do procedury zwrotnej, dostarczonej przez klienta modułu. Procedura daje możliwość klientowi obrazowania postępu wysyłki oraz przerwania wysyłania. Procedura będzie wywoływana wielokrotnie w czasie wysyłania danych w celu uaktualnienia informacji o postępie wysyłania. Jeśli **Progress=nil** to w przypadku modułu JPKLibUI.DLL zostanie otwarte wewnętrzne okno postępu wysyłania. W Przypadku JPKLib.DLL wysyłka będzie trwała bez wywoływania Progress.
- **Finish**: TFinishProc – Wskaźnik do procedury zwrotnej, dostarczonej przez klienta modułu. Procedura zostanie wywołana raz po zakończeniu wysyłki niezależnie od powodzenia. Jeśli **Finish=nil** to klient nie otrzyma powiadomienia o zakończeniu wysyłania. W takim przypadku w oczekiwaniu na zakończenie wysyłki można wykorzystać metodę WaitForEnd, a następnie odczytać status funkcją GetUPO. Gdy WaitforEnd nie zostanie wykorzystana można to do odczytania numerów referencyjnych należy użyć funkcję GetReferenceNumbers.
- **SyncFinish**: Boolean – Jeśli true to nastąpi synchronizacja wywoływania procedury Finish, jeśli uruchomiono więcej niż jedną sesję SingSendJPK.
- **Result**: Pointer – Identyfikator sesji **SessionID**.

TProgressProc = procedure (SessionID:Pointer; FilesCount, CurrentFile, FileSize, CurrentPosition : Int64; //Zmiana z Integer od wersji 1.0.0.24! var UserBreak:Boolean; Err:Integer; ErrMsg:PChar); stdcall;

Procedura typu TProgressProc może być utworzona po stronie klienta modułu i dostarczona do funkcji SignSendJPK. Procedura będzie wywoływana w trakcie wysyłania danych XML w sesji rozpoczętej przez SignSendJPK.

- **SessionID**: Pointer – Identyfikator sesji, która wywołuje procedurę, ten sam jaki został oddany w Result funkcji SignSendJPK.
- **FilesCount**: Int64 – Ilość wysyłanych plików w sesji (danych XML).
- **CurrentFile**: Int64 – numer aktualnie wysyłanego pliku w sesji (w zakresie od 1 do FilesCount).
- **FileSize**: Int64 – Wielkość aktualnie wysyłanego pliku (Danych XML).
- **CurrentPosition**: Int64 – Aktualnie wysłana ilość bajtów wysyłanego pliku.

- Var **UserBreak**:Boolean – Wartość dostarczana zwrotnie od klienta. Jeśli true to wysyłka danych w sesji zostanie przerwana.
- **Err**: Integer – Numer ewentualnego błędu.
- **ErrMsg**: PChar – Opis tekstowy ewentualnego błędu lub null lub pusty tekst.

TFinishProc = procedure(SessionID:Pointer; ReferenceNumber:PChar; Status:Integer; StatusText:PChar; UPO:PChar; UserBreak:Boolean; Err:Integer; ErrMsg:PChar); stdcall;

Procedura typu TFinishProc może być utworzona po stronie klienta modułu i dostarczona do funkcji SignSendJPK. Procedura będzie wywołana po zakończeniu wysyłania danych w sesji rozpoczętej przez SignSendJPK tyle razy ile wysłano dokumentów dodanych metodami AddXMLJPK lub AddXMLJPKFile i w takiej samej kolejności jak je dodawano. Wywoływanie procedury kończy się na pierwszym niewysłanym (błąd lub przerwanie wysyłki) dokumencie.

UWAGA! W wersji modułów <1.0.0.18 w przypadku przerwania wysyłki procedura była wywoływana tylko jeden raz.

- **SessionID**: Pointer – Identyfikator sesji, która wywołuje procedurę, ten sam jaki został oddany w Result funkcji SignSendJPK.
- **ReferenceNumber**: PChar – Tekst z numerem referencyjnym wysłanego dokumentu.
- **Status**: Integer – numer statusu zwracany przez bramkę JP, wg dokumentacji JPK.
- **StatusText**: PChar – tekst z opisem statusu zwracany przez bramkę JPK.
- **UPO**: PChar – tekst z Urzędowym Potwierdzeniem Odbioru – jeśli został dostarczony przez bramkę JPK.
- **UserBreak**: Boolean – true – jeśli wysyłanie zostało przerwane przez użytkownika.
- **Err**: Integer – 0 - jeśli wysyłka zakończyła się powodzeniem. W przeciwnym wypadku numer błędu;
- **ErrMsg**: PChar – opis błędu jeśli wystąpił lub null lub pusty tekst.

AUTHDATA – ciąg znaków typu CSV.

Dane autoryzacyjne do autoryzacji pliku JPK umożliwiające wysyłkę bez użycia certyfikatu.

Tekst składa się z prefiksu „AUTHDATA:” oraz pól oddzielonych przecinkami:

- NIP= (numer nip osoby podpisującej - bez kresek)

  lub
- PESEL= (numer pesel osoby podpisującej)
- IMIEPIERWSZE= (pierwsze imię osoby podpisującej)
- NAZWISKO= (nazwisko osoby podpisującej)
- DATAURODZENIA= (data w formacie rok-miesiąc-dzień)
- KWOTA = (kwota z ostatniego wysłanego JPK z dwoma miejscami po przecinku i kropką dziesiętną)

Przykład:

AUTHDATA:NIP=1234567890,IMIEPIERWSZE=JAN,NAZWISKO=TESTOWY,DATAURODZENIA=1990-01-01,KWOTA=123.45

procedure SignSendBreak(SessionID:Pointer); stdcall;

Ustawia polecenie przerwania trwającego podpisywania i wysyłania.

- **SessionID**:Pointer – Identyfikator sesji otrzymany w Result funkcji SignSendJPK. Jeśli SessionID=nil to przerywa wszystkie sesje.

procedure WaitForEnd(SessionID:Pointer; out ReferenceNumbers:PChar); stdcall;

Metodę używa się w celu oczekiwania na zakończenie SignSendJPK. Metoda WaitForEnd uruchamia pętlę komunikatów i nie blokuje działania kontrolek aplikacji.

Wywołanie WaitForEnd nie jest wymagane ani konieczne jeśli do SignSendJPK dostarczono procedurę zwrotną Finish typu TFinishProc.

- **SessionID**:Pointer – Identyfikator sesji otrzymany w Result funkcji SignSendJPK.
- **ReferenceNumbers**: PChar - Tekst z numerami referencyjnymi wysłanych w jednej sesji dokumentów XML. Zobacz: Format tekstu z numerami referencyjnymi.

function GetReferenceNumbers(SessionID:Pointer):PChar; stdcall;

Funkcja zwraca tekst z numerem lub numerami referencyjnymi dokumentów XML wysłanych w jednej sesji. Zobacz: Format tekstu z numerami referencyjnymi.

- **SessionID**:Pointer – Identyfikator sesji otrzymany w Result funkcji SignSendJPK.

function GetUPO (TestMode:Boolean; RefNumber : PChar; out UPO : PChar; out Status : integer; out StatusText : PChar) : Boolean; stdcall;

Funkcja łączy się z JPK i odczytuje aktualny status i UPO (Urzędowe Potwierdzenie Odbioru) jeśli jest. UPO jest dostępne gdy status=200 (zgodnie ze specyfikacją bramki JPK).

- **TestMode**:Boolean – Określenie z której bramki – testowej czy produkcyjnej – ma nastapić odczyt. Wartość true spowoduje odczyt z bramki testowej.
- **RefNumber**: PChar – Tekst z jednym numerem referencyjnym wysłanego dokumentu XML.
- out **UPO**: PChar – Tekst z Urzędowym Potwierdzenie Odbioru – jeśli istnieje. W przeciwnym wypadku null lub tekst pusty.
- out **Status**: Integer – Numer statusu dokumentu. Numer 200 oznacza, że dokument jest przyjęty i przetworzony (wg dokumentacji JPK).
- out **StatusText**: PChar – Tekst z opisem statusu.
- **Result**: Boolean – true jeśli GetUPO przebiegło pomyślnie, w przeciwnym wypadku informacje o błędzie można odczytać używając JPKGetLastError.

UWAGA

Funkcję GetUPO należy wywołać dopiero po zakończeniu działania SignSendJPK (po opuszczeniu procedury Finish lub WaitForEnd).

function BreakGetUPO(ReferenceNumber : PChar):Boolean; stdcall;

Funkcja przerywa proces pobierania UPO. To może być istotne w przypadku dużego obciążenia bramki Min.Fin. i zbyt długiego czasu oczekiwania na połaczenie.

Ponieważ funkcja GetUPO nie blokuje pętli komunikatów Windows można okna i kontrolki aplikacji nie są blokowane. Przed uruchomieniem BreakGetUPO można zainicjować okno niemodalne z opcją przerwania pobierania UPO i wyświetlić je po upływie jakiegoś czasu aby użytkownik mógł zdecydować o przerwaniu oczekiwania.

- **ReferenceNumber**: PChar – Tekst z jednym numerem referencyjnym wysłanego dokumentu XML.
- **Result**: Boolean – true jeśli istnieje proces pobierania UPO dla numeru określonego w ReferenceNumber.

function JPKGetLastError(out ErrText:PChar):Integer; stdcall;

Funkcja zwraca kod i tekstu ostatniego błędu.

- out **ErrText**: PChar – tekst z treścią błędu.
- **Result**: Integer – numer błędu.

Format tekstu z numerami referencyjnymi

Tekst z numerami referencyjnymi zawiera numer lub wiele numerów.

W przypadku zawierania wielu numerów są one rozdzielone znakami o kodach CR LF (13 i 10).

W przypadku wysyłki testowej [*SignSendJPK(TestMode=true)*] tekst z numerem/numerami referencyjnymi będzie poprzedzony przedrostkiem „**test:**”.

function FileNameValidate (const FileName:PChar; out errText: PChar):Boolean; stdcall;

Funkcja sprawdzająca poprawność nazwy XML – jej długość oraz użyte znaki.

- **FileName**: PChar – Tekst ze sprawdzaną nazwą.
- out **errText**: PChar – Tekst z opisem znalezionego błędu.
- **Result**: Boolean = true jeśli nazwa jest poprawna.

function XMLValidate (const xmlData: PAnsiChar; out errText: PChar): Boolean; stdcall;

(tylko JPKLibUI.DLL)

Funkcja sprawdzająca poprawność danych XML na podstawie schematów xsd Min.Fin.

Funkcja pobiera z internetu potrzebne schematy i zapisuje do ponownego wykorzystania w katalogu Temp użytkownika. Podstawowe *.XSD JPK są pobierane z katalogu \katalog_modułu\XSD powstałego podczas instalacji programu JotPeK.

Działanie uruchomionego testu można przerwać metoda XMLValidateBreak.

UWAGA! Zobacz Wymagania i ograniczenia.

- **XmlData**: PAnsiChar – Tekst z dokumentem XML do sprawdzenia.
- **ErrText**: PChar – Tekst z objaśnieniem znalezionego błędu lub null lub tekst pusty.
- **Result**: Boolean = true jeśli nie znaleziono błędu.

function XMLFileValidate (const XMLFileName: PChar; out errText: PChar): Boolean; stdcall;

(tylko JPKLibUI.DLL)

Funkcja sprawdzająca poprawność pliku XML na podstawie schematów xsd Min.Fin.

Funkcja pobiera z internetu potrzebne schematy i zapisuje do ponownego wykorzystania w katalogu Temp użytkownika. Podstawowe *.XSD JPK są pobierane z katalogu \katalog_modułu\XSD powstałego podczas instalacji programu JotPeK.

Działanie uruchomionego testu można przerwać metoda XMLValidateBreak.

- **XMLFileName**: PChar – Pełna ścieżka z nazwą pliku z dokumentem XML do sprawdzenia.
- **ErrText**: PChar – Tekst z objaśnieniem znalezionego błędu lub null lub tekst pusty.
- **Result**: Boolean = true jeśli nie znaleziono błędu.

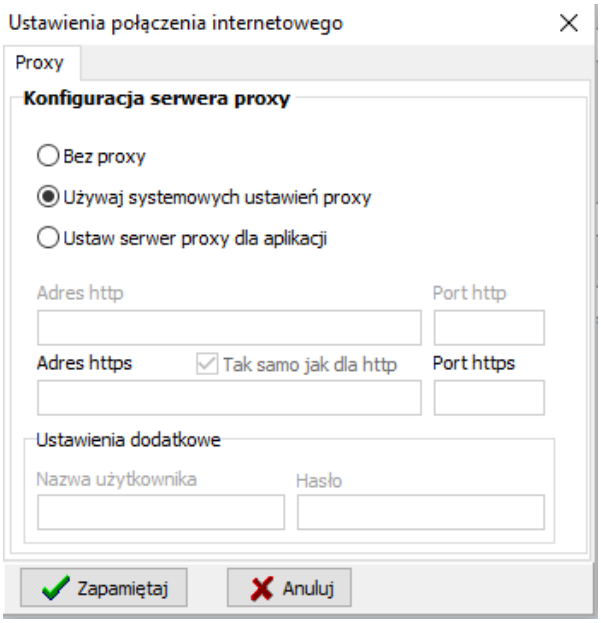
procedure XMLValidateBreak; stdcall;

(tylko JPKLibUI.DLL)

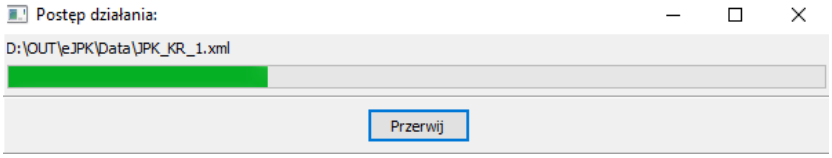
Procedura przerywająca proces testowania XML. Przydatna w przypadku zbyt długiego testowania wielkich plików.

5. Wygląd wewnętrznych formatek JPKLibUI.DLL

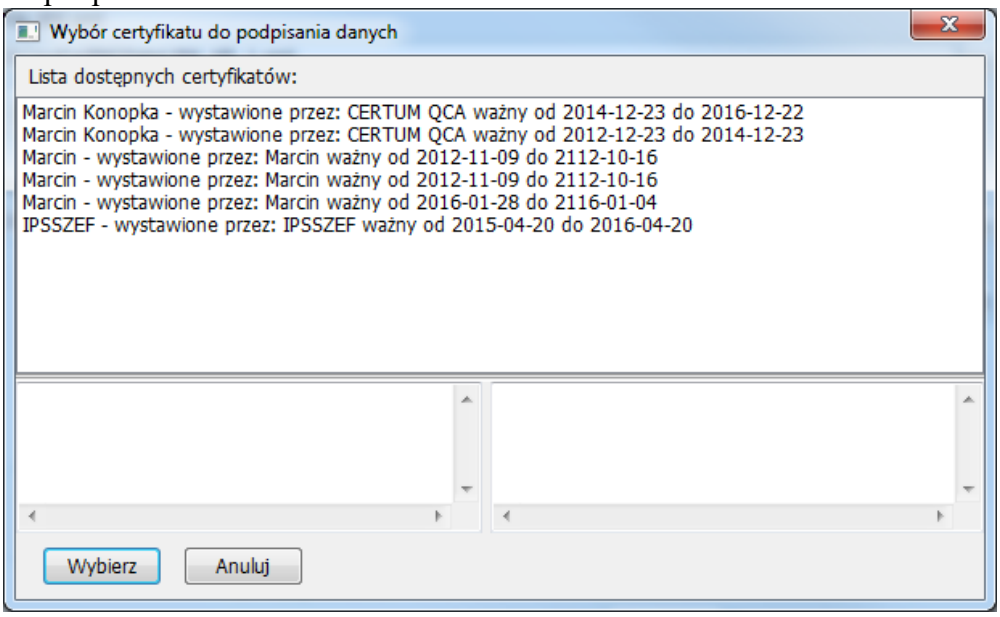
Okno ustawień połączenia HTTP i HTTPs.



Okno progress bar



Okno wyboru podpisów:



6. Przykłady

Wysyłka

```
procedure ShowError;
var
  errTx: PChar;
begin
  if JPKGetLastError(errTx)<0 then
    ShowMessage(errTx);
end;

Const
  cCzyTest = true;
  dt_JPK = 0; //JPK cykliczny
  dt_JPKAH = 1; //JPK na żądanie urzędu
Var
  idSession: Pointer;
  vRefsNum: PChar
  vUPO: PChar;
  vStatusText: PChar;
begin
  SetLic('numer licencji');

  if AddXMLJPKFile('nazwa pliku JPK-XML',dt_JPK)<0 then //Dodanie pliku XML
  begin
    ShowError;
    Exit;
  end;

  idSession:=SignSendJPK(cCzyTest,nil,nil,false); //Inicjalizacja wysyłki.
  if idSession<>nil then
  begin
    SetProgressCaption(idSession,'Mój XML'); //Ustawienie tekstu na progress-bar
    WaitForEnd(idSession,vRefsNum); //Oczekiwanie na zakończenie procesu wysyłki
    if vRefsNum<>nil then //Jeśli wysyłka przebiegła prawidłowo, tzn. jest numer referencyjny.
    begin

      ..Tutaj zapisanie numeru referencyjnego …

      Sleep(1000); //Dać trochę czasu serwerowi M.F.
      if GetUPO(cCzyTest,vRefsNum,vUPO,vStatus,vStatusText) then //Odczyt statusu
```

```
      begin

        ..Tutaj zapisanie statusu i upo. //UPO prawdopodobnie jeszcze nie będzie

      end else
        ShowError;
    end else
      ShowError;
  end else
    ShowError;
end;
```

7. Dodatkowe informacje

Z dnia 2.03.2018

Został usunięty błąd, który polegał na możliwości wystąpienia w jednej instancji modułu JPKLib i JPKLibUI takich samych SessionID zwracanych przez metodę **SignSendJPK**.

Powtórzenie SessionID z kolei powodowało pobieranie numeru referencyjnego uzyskanego w pierwszym wystąpieniu SessionID.