公的個人認証サービス

利用者クライアントソフト API 仕様書

【カード AP ライブラリ CryptoAPI 編】

第 4.3 版

地方公共団体情報システム機構

変更履歴

版数	変更日付	変更内容
1.0 版	平成 16 年 1 月 16 日	新規作成
1.1 版	平成 16 年 10 月 14 日	Windows XP SP2 対応に伴い表 3-1 のプラットフォームを追加。
2.0 版	平成 18 年 5 月 2 日	公的個人認証サービス利用者クライアントソフト Ver2.0 のリリースに伴い、動作環境、ソフトウェア構成図を変更。
2.1 版	平成 18 年 11 月 1 日	MacOS 対応に伴い、第２章　ドキュメント体系を変更
2.2 版	平成 19 年 10 月 4 日	WindowsVista 対応に伴い表 3-1 のプラットフォームを追加。
2.3 版	平成 20 年 10 月 10 日	表 3-1 動作環境のプラットフォームに WindowsVista ServicePack1、WindowsXP ServicePack3 を追加。 表 3-1 動作環境 Web ブラウザに Internet Explorer6.0 ServicePack3 を追加。
2.4 版	平成 23 年 04 月 01 日	図 2-1 ドキュメント体系図に「JavaDoc JPKICryptJNI(64bit)」を追加。 表 3-1 動作環境のプラットフォームを表 3-1 動作環境(Windows)、表 3-2 動作環境(IC カード)に分割しマトリックス形式の記述に変更。 表 3-1 動作環境(Windows)の OS に WindowsVista ServicePack2, Windows7(32/64 bit)を追加、Web ブラウザに Internet Explorer8.0 を追加。 図 4-1 ソフトウェア構成図を変更。
2.5 版	平成 25 年 12 月 01 日	第３章　動作環境　表 3-1 動作環境(Windows)より Windows2000 を削除、Windows8(32/64bit)、Windows8.1(32/64bit)を追加。 第５章　第１節　sha-256 に対応する API の記載を追加 第５章　第２節　（８）CryptCreateHash の ALG_ID に CALG_SHA_256 を追加。 第５章　第３節　（２）RSAPUBKEY の鍵長に 2048(鍵長)を追加。
3.0 版	平成 26 年 04 月 01 日	全体　「地方公共団体情報システム機構」への事業承継により、組織名称を変更する。 全体　「公的個人認証サービス共通基盤事業運用会議」への事業承継により、「公的個人認証サービス都道府県協議会」の組織名称を変更する。

版数	変更日付	変更内容
3.1 版	平成 26 年 07 月 01 日	第３章　表　３-１　動作環境(Windows)の Windows XP/Windows 8 を削除、Windows 8.1 を Windows 8.1 update に変更、Windows7(32/64bit)の Web ブラウザを IE11.0 に変更。 第５章　第１節　表　５-１ サポート API 一覧の注釈に記載された番号に 13 を追加し、SHA256 による署名生成および署名検証について追記 第５章　第４節　第５章　第４節　（３）/第５章　第４節　（４）/第５章　第４節　（５）/第５章　第４節　（６）それぞれの CryptCreateHash の ALG_ID に「(CALG_SHA1 または CALG_SHA256 のいずれかを署名方式に合わせて指定)」を追加
4.0 版	平成 27 年 6 月 30 日	番号制度対応に伴い、以下を修正。 ・第１章 第１節に用語の定義を追加。 ・第２章のドキュメント体系を修正。 ・第３章の動作環境を修正。 ・第４章の機能仕様を修正。 ・第５章の API 仕様を修正。 ・第６章の画面仕様を修正。
4.0.1 版	平成 28 年 10 月 26 日	・第３章システム概要 動作環境　更新プログラムに係る注釈※3、4 の追加 ・第５章　第２節　サポート API 仕様詳細　（１）引数の文言修正及び備考に dwFlags のパラメータに関する内容を追加 ・第５章　第４節　コーリングシーケンス（１）利用者証明書取得処理に CRYPT_VERIFYCONTEXT の記載を追加 ・文末　注意事項の利用用途の文言修正 その他、図の整形及び誤記等の文言修正

版数	変更日付	変更内容
4.1版	平成28年11月30日	PC接続機能追加に伴い、以下を修正。 ・第１章　第１節の「用語の定義」に以下を追加。 ➢ PC/SC ➢ ICカードリーダライタ ➢ 挿入 ➢ NFC ➢ Bluetooth ・第２章　ドキュメント体系図にAndroid版を追加。 ・第３章　表3-1にPC接続機能対応可否追加 ・第３章　表3-1にWindows 10(32/64bit)を追加。 ・第３章　表3-2 動作環境(ICカード)を表3-2動作環境(ICカード)、第１節 PC/SC対応ICカードリーダライタ、第２節 Android端末に分割。 ・第４章 第１節 ソフトウェア構成図にBluetooth通信を追加。
4.2版	平成29年07月31日	Java9対応に伴い、以下を修正。 ・第２章 ドキュメント体系図を改訂。 ・第３章 表 ３-１のOSからWindows Vista(32bit)を削除。 ・第３章 表 ３-１のOSからWindows 8(32bit)を削除。 ・第３章 表 ３-１のOSからWindows 8(64bit)を削除。 ・第３章 表 ３-１のJavaVMにJRE9を追加。 ・第３章 表 ３-４の【PC接続の場合】にAndroid 6.0.1、7.0を追加。 ・第３章 表 ３-４の【Android単体で利用する場合】にAndroid 6.0.1、7.0を追加。 ・その他、記載の見直しを実施。
4.3版	平成31年3月31日	・第３章 表 ３-４の【PC接続の場合】、【Android単体で利用する場合】にAndroid 8.0を追加。 ・その他、記載の見直しを実施。

－ 目次 －

第１章　はじめに

利用者クライアントソフトにおけるカード AP ライブラリは、以下の機能を実現するための Application Program Interface(以下、API)を提供する。

- 証明書取得機能
- 電子署名生成機能
- 電子署名検証機能

以降、本書ではカード AP ライブラリのうち、CryptoAPI の API 仕様について説明する。

第１節　用語の定義

表 1-1 用語の定義

項番	用語・略号	説明
1	IC カード	以下のカードを指す総称。 ・住基カード ・個人番号カード
2	電子証明書	公開鍵及び発行対象を識別する情報を含むデータに、認証局が発行対象の正当性を保証する電子署名を付与して、発行されるデータをいう。データは、日本工業規格 X560-1 の識別符号化規則により符号化された形式で利用される。
3	証明書	電子証明書と同義。
4	利用者証明書	公的個人認証サービスで発行した利用者の証明書。 本書では以下の電子証明書を指す。 ・住基カードに格納された署名用電子証明書 ・個人番号カードに格納された署名用電子証明書 ・個人番号カードに格納された利用者証明用電子証明書
5	利用者秘密鍵	公開鍵暗号方式において用いられる鍵ペアの一方。公開鍵に対する、利用者のみが保有する鍵。 本書では以下の秘密鍵を指す。 ・住基カードに格納された署名用利用者秘密鍵 ・個人番号カードに格納された署名用利用者秘密鍵 ・個人番号カードに格納された利用者証明用利用者秘密鍵
6	認証局の自己署名証明書	自認証局の公開鍵に対して、自認証局の秘密鍵で署名した証明書。 本書では以下の電子証明書を指す。 ・住基カードに格納された都道府県知事の自己署名証明書 ・個人番号カードに格納された署名用認証局の自己署名証明書 ・個人番号カードに格納された利用者証明用認証局の自己署名証明書
7	PC/SC	Personal Computer/Smart Card の略。
8	IC カードリーダライタ	以下の機器を指す総称 ・PC/SC 対応 IC カードリーダライタ ・Android 端末
9	挿入	IC カードリーダライタが IC カードを読み込める状態にすること。 具体的には以下の状態にすることを指す。 ・PC/SC 対応 IC カードリーダライタに IC カードをセットすること ・Android 端末に IC カードをセットすること
10	NFC	Near Field Communication　(近距離無線通信)の略。

項番	用語・略号	説明
11	Bluetooth	機器間の近距離無線通信 IEEE 802.15.1 の規格名称。

第2章　ドキュメント体系

利用者クライアントソフトのドキュメント体系図を以下に示す。本書は以下の体系図の網掛け部分に該当する。

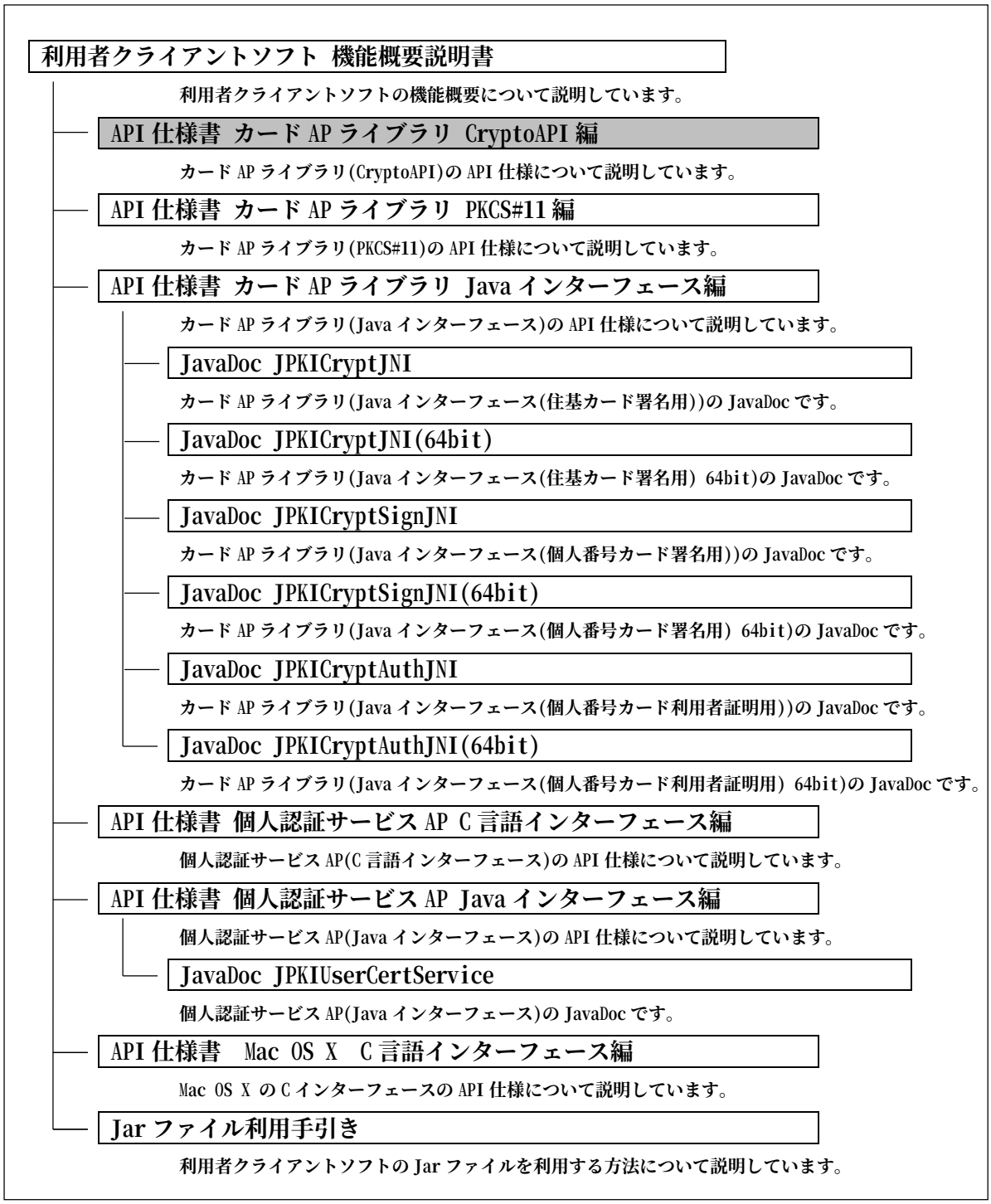


図 2-1 ドキュメント体系図

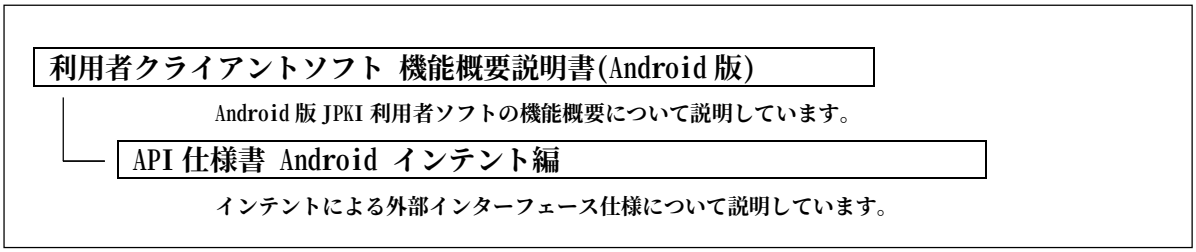


図 2-2 ドキュメント体系図(Android 版)

第３章　動作環境

カード AP ライブラリ(CryptoAPI)の動作環境は以下の通りとする。

表 3-1 動作環境(Windows)

OS(※1)	Web ブラウザ (※1, ※2)	JavaVM(※1,※6)		PC 接続機能対応可否 (※5)
		JRE8.0	JRE9.0	
Windows 7(32bit) (Service Pack 1) (※3)	IE11.0	○	×	○
Windows 7(64bit) (Service Pack 1) (※3)	IE11.0	○	○	○
Windows 8.1 update (32bit) (※4)	IE11.0	○	×	○
Windows 8.1 update (64bit) (※4)	IE11.0	○	○	○
Windows 10(32bit 版)	IE11.0	○	×	○
Windows 10(64bit 版)	IE11.0	○	○	○

※1 本仕様書で定めるバージョンの開発時点の環境。最新の動作環境の情報は、JPKI ポータルサイトに掲載するものとする。

※2 プラットフォームがWindowsの場合､暗号機能等の利用のためにInternet Explorerが必要。

※3 Windows の更新プログラム(KB3033929)をインストールする必要がある。
(なお、更新プログラム(KB3033929)のインストール前に、
関連する更新プログラム(KB3035131)が必要になる為、注意すること。)

※4 Windows の更新プログラム(KB2919355)をインストールする必要がある。

※5 PC 接続機能については「利用者クライアントソフト 機能概要説明書 第３章 第３節 PC 接続機能について」を参照。

※6 JRE9.0 の 32bit 版は Oracle 社より提供されていないため非対応。

IC カードの動作環境は以下の通りとする。

表 3-2 動作環境(IC カード)

項目	条件
IC カード	住基カードまたは個人番号カードであること。 ※PC 接続機能を使用する場合は個人番号カードのみ対応。

第１節　PC/SC 対応 IC カードリーダライタ

PC/SC 対応 IC カードリーダライタの動作環境は以下の通りとする。

表 3-3 動作環境(PC/SC 対応 IC カードリーダライタ)

項目	条件
PC/SC 対応 IC カードリーダライタ	以下の条件を満たす PC/SC 対応 IC カードリーダライタとする。(「個人番号カード対応適合性検証済み IC カードリーダーライタ一覧」「住基カード対応適合性検証済み IC カードリーダーライタ一覧」(※1)を参照のこと。) ・ IC カードのインターフェース(非接触型、接触非接触両対応型)に対応していること。 ・ PC/SC 対応 IC カードリーダライタであること。 ・ USB など、パソコンに接続するためのインターフェースを有すること。 ・ PC/SC 対応 IC カードリーダライタと通信するためのドライバソフトウェアが提供されていること。 ・ IC カードの搬送方式が手動挿入/手動排出タイプまたは自動挿入/自動排出タイプであること。 ・ IC カードを挿入するスロットの数は１つとし、１度に挿入できる IC カードは１枚であること。

※1 最新の「個人番号カード対応適合性検証済み IC カードリーダーライタ一覧」「住基カード対応適合性検証済み IC カードリーダーライタ一覧」の情報は、JPKI ポータルサイトに掲載するものとする。

第２節　Android 端末

Android 端末の動作環境は以下の通りとする。

表 3-4 動作環境(Android 端末)

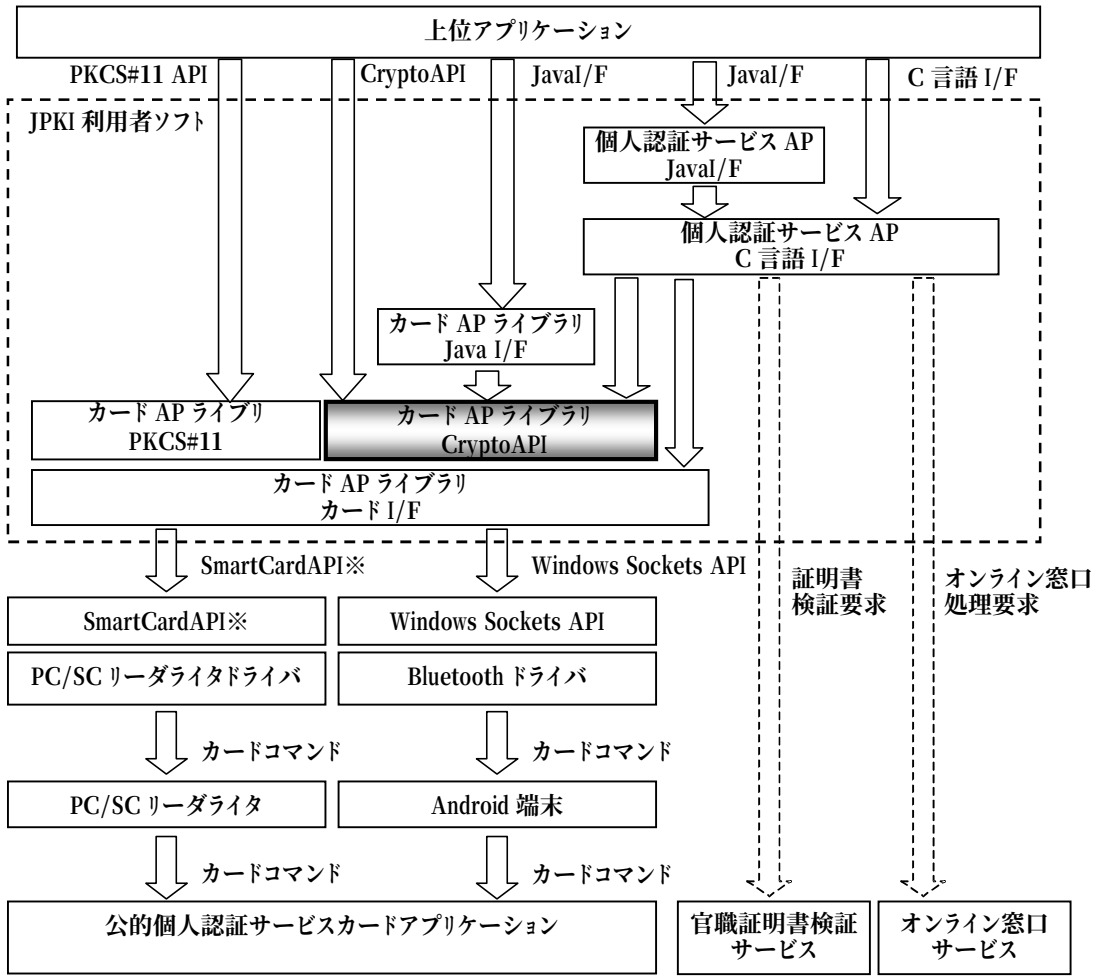
項目	条件
Android 端末	以下の条件を満たす Android 端末とする。(「個人番号カード対応適合性検証済み Android 端末一覧」(※)を参照のこと。) ・【PC 接続の場合】Android 4.3、5.1、6.0.1、7.0 または 8.0 を搭載していること。 ・【Android 単体で利用する場合】Android 5.1、6.0.1、7.0 または 8.0 を搭載していること。 ・Bluetooth 4.0 を搭載していること。 ・ISO/IEC 14443 Type B に対応している NFC を搭載していること。

※ 最新の「個人番号カード対応適合性検証済み Android 端末一覧」の情報は、JPKI ポータルサイトに掲載するものとする。

第４章　機能仕様

第１節　ソフトウェア構成図

本仕様書では、利用者クライアントソフトのうち、下図の太枠に示すカード AP ライブラリ(CryptoAPI)の仕様をまとめる。



※ Smart Card Resource Manager API の略

図 4-1 ソフトウェア構成図

第２節　実現可能な機能の一覧

カード AP ライブラリ(CryptoAPI)で実現可能な機能の一覧を表 ４-１に示す。

表 ４-１ 実現可能な機能の一覧

NO	機能	概要
1	利用者証明書取得	IC カードに格納された利用者証明書を取得する。
2	認証局の自己署名証明書取得	IC カードに格納された認証局の自己署名証明書を取得する。
3	署名生成(署名対象データを渡すパターン)	署名対象データからハッシュ値を計算し、IC カードに格納された利用者秘密鍵を使用して電子署名を生成する。
4	署名生成(ハッシュ値を渡すパターン)	ハッシュ値に対して、IC カードに格納された利用者秘密鍵を使用して電子署名を生成する。
5	署名検証(検証対象データを渡すパターン)	検証対象データからハッシュ値を計算し、ハッシュ値、電子署名、公開鍵を使用して電子署名を検証する。
6	署名検証(ハッシュ値を渡すパターン)	ハッシュ値、電子署名、公開鍵を使用して電子署名を検証する。
7	繰り返し署名生成(署名対象データを渡すパターン)	NO3 の処理を繰り返し実行し、複数の署名対象データに対する電子署名を生成する。
8	繰り返し署名生成(ハッシュ値を渡すパターン)	NO4 の処理を繰り返し実行し、複数のハッシュ値に対する電子署名を生成する。
9	繰り返し署名検証(検証対象データを渡すパターン)	NO5 の処理を繰り返し実行し、複数の電子署名を検証する。
10	繰り返し署名検証(ハッシュ値を渡すパターン)	NO6 の処理を繰り返し実行し、複数の電子署名を検証する。

第５章　API 仕様

第１節　サポート API 一覧

利用者クライアントソフト用 Cryptographic Service Provider(以下、CSP)がサポートする CryptoAPI の一覧を表 ５-１に示す。なお、サポートしていない API を呼び出した場合には、戻り値が常に FALSE(失敗)となる。

表 ５-１ サポート API 一覧

NO	CryptoAPI	概要
1	CryptAcquireContext	鍵コンテナのハンドルを生成する。
2	CryptReleaseContext	鍵コンテナのハンドルを開放する。
3	CryptGetProvParam	CSP のパラメータの値を取得する。
4	CryptDestroyKey	鍵の破棄を行う。
5	CryptGetKeyParam	鍵のパラメータの値を取得する。
6	CryptImportKey	外部から鍵を取り込む。
7	CryptGetUserKey	鍵コンテナ内の鍵ハンドルを取得する。
8	CryptCreateHash	ハッシュオブジェクトの生成を行う。
9	CryptDestroyHash	ハッシュオブジェクトの破棄を行う。
10	CryptSetHashParam	ハッシュオブジェクトのパラメータを設定する。
11	CryptGetHashParam	ハッシュオブジェクトのパラメータを取得する。
12	CryptHashData	ハッシュオブジェクトにデータを与えハッシュ値を計算する。
13	CryptSignHash	ハッシュ値に署名を行う。
14	CryptVerifySignature	署名を検証する。

※NO.8,9,10,11,12,13,14 について sha-256 の利用者証明書による署名生成および署名検証、sha-256 の官職証明書・職責証明書による署名検証に対応済み。

第２節　サポート API 仕様詳細

各関数の戻り値は BOOL 型で、エラーが発生した場合は FALSE を返す。上位アプリケーションでエラー情報を取得する際は、GetLastError()関数をコールしてエラーコードを取得すること。なお、本仕様書に記述のないエラーコードが OS から返る場合があるが、それらのエラーコードについては MSDN を参照のこと。

（1） CryptAcquireContext

API 名	CryptAcquireContext		
概要	鍵コンテナのハンドルを生成する。		
関数インターフェース	CryptAcquireContext(HCRYPTPROV* phProv, LPCTSTR pszContainer, LPCTSTR pszProvider, DWORD dwProvType, DWORD dwFlags);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTPROV*	OUT	ハンドル格納場所のポインタ
	LPCTSTR	IN	NULL 文字で終了するコンテナ名称 NULL を指定すること。
	LPCTSTR	IN	NULL 文字で終了する CSP 名称 ”JPKI Crypto Service Provider” ”JPKI Crypto Service Provider for Sign” ”JPKI Crypto Service Provider for Auth” のいずれかを指定する。 詳細は備考(A)を参照。
	DWORD	IN	プロバイダタイプ PROV_RSA_FULL PROV_RSA_AES のいずれかを指定すること。 詳細は備考(B)を参照。

	型	I/O	内容
引数	DWORD	IN	動作に関するパラメータ 0 CRYPT_VERIFYCONTEXT のいずれかを指定すること。 詳細は備考(C)を参照。
	値	内容	
エラーコード	NTE_BAD_FLAGS	dwFlags に不正な値が設定されている。	
	NTE_KEYSET_NOT_DEF	鍵コンテナが見つからない。	
	NTE_NO_MEMORY	メモリが不足している。	
	SCARD_W_CANCELLED_BY_USER	パスワード入力が利用者によってキャンセルされた。	
	SCARD_E_NOT_READY	R/W 接続不備、IC カード未挿入等のため IC カードに接続できない。	
	SCARD_E_UNKNOWN_CARD	IC カード種別が相違している。 ※IC カード種別相違と判定されるケースは以下の通り。 (1)CSP 名称が"JPKI Crypto Service Provider"かつ住基カード以外の IC カードが挿入されている場合 (2)CSP 名称が"JPKI Crypto Service Provider for Sign"かつ個人番号カード以外の IC カードが挿入されている場合 (3)CSP 名称が"JPKI Crypto Service Provider for Auth"かつ個人番号カード以外の IC カードが挿入されている場合	
	SCARD_W_CHV_BLOCKED	利用者パスワードがロックしている。	

備考

(A) pszProvider に指定した CSP 名称によって、CryptAcquireContext で取得したハンドルを利用する処理の対象となる利用者証明書、利用者秘密鍵、認証局の自己署名証明書は以下のように決定される。

"JPKI Crypto Service Provider"の場合	
利用者証明書	住基カードに格納された署名用電子証明書
利用者秘密鍵	住基カードに格納された署名用利用者秘密鍵
認証局の自己署名証明書	住基カードに格納された都道府県知事の自己署名証明書
"JPKI Crypto Service Provider for Sign"の場合	
利用者証明書	個人番号カードに格納された署名用電子証明書
利用者秘密鍵	個人番号カードに格納された署名用利用者秘密鍵
認証局の自己署名証明書	個人番号カードに格納された署名用認証局の自己署名証明書
"JPKI Crypto Service Provider for Auth"の場合	
利用者証明書	個人番号カードに格納された利用者証明用電子証明書
利用者秘密鍵	個人番号カードに格納された利用者証明用利用者秘密鍵
認証局の自己署名証明書	個人番号カードに格納された利用者証明用認証局の自己署名証明書

(B) dwProvType に指定するプロバイダタイプは、pszProvider に指定した CSP 名称に応じて以下の値を設定する。

"JPKI Crypto Service Provider"の場合	
プロバイダタイプ	PROV_RSA_FULL
"JPKI Crypto Service Provider for Sign"または "JPKI Crypto Service Provider for Auth"の場合	
プロバイダタイプ	PROV_RSA_AES

<table>
<tr><td>備考</td><td>(C) dwFlags に指定する動作に関するパラメータは、pszProvider に指定した CSP 名称に応じて以下の値を設定する。
<table>
<tr><td colspan="2">"JPKI Crypto Service Provider"または
"JPKI Crypto Service Provider for Sign"の場合</td></tr>
<tr><td>動作に関するパラメータ</td><td>0: 利用者証明書、利用者秘密鍵使用時に指定する。
CRYPT_VERIFYCONTEXT: 認証局の自己署名証明書取得、署名検証の際に指定する。</td></tr>
<tr><td colspan="2">"JPKI Crypto Service Provider for Auth"の場合</td></tr>
<tr><td>動作に関するパラメータ</td><td>0: 利用者証明書、利用者秘密鍵使用時に指定する。
CRYPT_VERIFYCONTEXT:認証局の自己署名証明書取得、署名検証の際に指定する。</td></tr>
</table>
pszProvider に指定した CSP 名称に応じて以下の画面が表示される。
<table>
<tr><td colspan="2">"JPKI Crypto Service Provider"の場合</td></tr>
<tr><td>表示画面</td><td>住基カード署名用パスワード入力画面</td></tr>
<tr><td colspan="2">"JPKI Crypto Service Provider for Sign"の場合</td></tr>
<tr><td>表示画面</td><td>個人番号カード署名用パスワード入力画面</td></tr>
<tr><td colspan="2">"JPKI Crypto Service Provider for Auth"の場合</td></tr>
<tr><td>表示画面</td><td>個人番号カード利用者証明用パスワード入力画面</td></tr>
</table>
</td></tr>
</table>

（2） CryptReleaseContext

API 名	CryptReleaseContext		
概要	鍵コンテナのハンドルを開放する。		
関数インターフェース	CryptReleaseContext(HCRYPTPROV hProv, DWORD dwFlags);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTPROV	IN	CryptAcquireContext で取得したハンドル
	DWORD	IN	動作に関するパラメータ 0 を指定すること。
	値	内容	
エラーコード	NTE_BAD_UID	hProv に不正な値が設定されている。	

（3） CryptGetProvParam

API 名	CryptGetProvParam		
概要	CSP のパラメータの値を取得する。		
関数インターフェース	CryptGetProvParam(HCRYPTPROV hProv, DWORD dwParam, BYTE* pbData, DWORD* pdwDataLen, DWORD dwFlags);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTPROV	IN	CryptAcquireContext で取得したハンドル
	DWORD	IN	値を取得するためのパラメータ サポートする値を表 5-2 に示す。
	BYTE*	OUT	値を格納するバッファのポインタ NULL を指定した場合は値の書き込みは行われず、pdwDataLen に値の格納に必要な長さが設定される。

	型	I/O	内容
引数	DWORD*	IN/OUT	値の長さを保持するバッファへのポインタ 関数呼び出し時には、pbData バッファに割り当てられたメモリサイズを指定する。関数終了時には、値の格納に必要な長さが設定される。
	DWORD	IN	動作に関するパラメータ 0 を設定すること。
	値	内容	
エラーコード	ERROR_MORE_DATA	pbData バッファが小さすぎる。	
	NTE_BAD_TYPE	dwParam に不正な値が設定されている。	
	NTE_BAD_UID	hProv に不正な値が設定されている。	

表 5-2 CryptGetProvParam の dwParam でサポートする値

#	値	内容
1	PP_IMPTYPE	CSP の実装形を DWORD 型で返す。 本 CSP は、CRYPT_IMPL_MIXED ¦ CRYPT_IMPL_REMOVABLE を返す。
2	PP_NAME	CSP 名称を CHAR 型の NULL ターミネート文字列で返す。 本 CSP は、CryptAcquireContext 実行時に引数 pszProvider で指定された CSP 名称を返す。
3	PP_VERSION	CSP のバージョンを DWORD 型で返す。 本 CSP は、1.0(0x00000100)を返す。
4	PP_PROVTYPE	CSP のプロバイダタイプを DWORD 型で返す。 本 CSP は、CryptAcquireContext 実行時に引数 dwProvType で指定されたプロバイダタイプを返す。
5	PP_JPKI_CA_CERTIFICATE	本 CSP は、IC カードに格納された認証局の自己署名証明書(DER 形式)を返す。

（4） CryptDestroyKey

API 名	CryptDestroyKey
概要	鍵の破棄を行う。
関数インターフェース	CryptDestroyKey(HCRYPTKEY hKey);

戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTKEY	IN	破棄する鍵のハンドル
	値	内容	
エラーコード	NTE_BAD_KEY	hKey に不正な値が設定されている。	

（5） CryptGetKeyParam

API 名	CryptGetKeyParam		
概要	鍵のパラメータの値を取得する。 (IC カードに格納された利用者証明書(DER 形式)を返す。)		
関数インターフェース	CryptGetKeyParam(HCRYPTKEY hKey, DWORD dwParam, BYTE* pbData, DWORD* pdwDataLen, DWORD dwFlags);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTKEY	IN	値を取得する鍵のハンドル
	DWORD	IN	値を取得するパラメータ KP_CERTIFICATE: IC カードに格納された利用者証明書を返す。
	BYTE*	OUT	値を格納するバッファのポインタ NULL を指定した場合は値の書き込みは行われず、pdwDataLen に値の格納に必要な長さが設定される。
	DWORD*	IN/OUT	値の長さを保持するバッファへのポインタ 関数呼び出し時には、pbData バッファに割り当てられたメモリサイズを指定する。関数終了時には、値の格納に必要な長さが設定される。
	DWORD	IN	動作に関するパラメータ 0 を設定すること。

	値	内容
エラーコード	ERROR_MORE_DATA	pbData バッファが小さすぎる。
	NTE_BAD_KEY	hKey に不正な値が設定されている。
	NTE_BAD_TYPE	dwParam に不正な値が設定されている。

（6） CryptImportKey

API 名	CryptImportKey		
概要	外部から鍵を取り込む。		
関数インターフェース	CryptImportKey(HCRYPTPROV hProv, BYTE* pbData, DWORD dwDataLen, HCRYPTKEY hPubKey, DWORD dwFlags, HCRYPTKEY* phKey);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTPROV	IN	CryptAcquireContext で取得したハンドル
	BYTE*	IN	鍵データのバッファへのポインタ 公開鍵を取り込む際のデータ構造は構造体「第 3 節　（3）Public-key BLOBs」を参照のこと。
	DWORD	IN	鍵データのサイズ(バイト)
	HCRYPTKEY	IN	鍵データの各種パラメータ NULL を設定すること。
	DWORD	IN	動作に関するパラメータ 0 を設定すること。
	HCRYPTKEY*	OUT	取り込んだ鍵のハンドルをコピーするバッファのアドレス
	値	内容	
エラーコード	NTE_BAD_TYPE	サポートされていない BLOB タイプまたは不正な鍵をインポートしようとした。	
	NTE_BAD_UID	hProv に不正な値が設定されている。	

	値	内容
エラーコード	NTE_BAD_VER	インポートしようとした鍵 BLOB のバージョンはサポートしていない。
備考	本関数では、暗号化されていない PUBLICKEYBLOB のみをサポートする。	

（7） CryptGetUserKey

API 名	CryptGetUserKey		
概要	鍵コンテナ内の鍵ハンドルを取得する。		
関数インターフェース	CryptGetUserKey(HCRYPTPROV hProv, DWORD dwKeySpec, HCRYPTKEY* phUserKey);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTPROV	IN	CryptAcquireContext で取得したハンドル
	DWORD	IN	鍵の種類 AT_KEYEXCHANGE: 鍵交換用鍵 AT_SIGNATURE: 署名用鍵 AT_SIGNATURE を設定すること。
	HCRYPTKEY*	OUT	鍵ハンドルをコピーするバッファのアドレス
	値	内容	
エラーコード	NTE_BAD_KEY	hKey に不正な値が設定されている。	
	NTE_BAD_UID	hProv に不正な値が設定されている。	
	NTE_NO_KEY	dwKeySpec に指定された種類の鍵が存在しない。	

（8） CryptCreateHash

API 名	CryptCreateHash
概要	ハッシュオブジェクトの生成を行う。

関数インターフェース	CryptCreateHash(HCRYPTPROV hProv, ALG_ID Algid, HCRYPTKEY hKey, DWORD dwFlags, HCRYPTHASH* phHash);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTPROV	IN	CryptAcquireContext で取得したハンドル
	ALG_ID	IN	ハッシュアルゴリズム CALG_SHA: SHA1 アルゴリズム CALG_SHA1: SHA1 アルゴリズム CALG_SHA_256: SHA256 アルゴリズム CALG_SSL3_SHAMD5: SSL クライアント認証用アルゴリズム
	HCRYPTKEY	IN	キードハッシュの場合の鍵ハンドル 0 を設定すること。(本 CSP ではキードハッシュは未サポート)
	DWORD	IN	動作に関するパラメータ 0 を設定すること。
	HCRYPTHASH*	OUT	生成したハッシュオブジェクトのハンドルをコピーするバッファのポインタ
	値	内容	
エラーコード	NTE_BAD_ALGID	指定されたアルゴリズムはサポートしていない。	
	NTE_BAD_UID	hProv に不正な値が設定されている。	
	NTE_NO_MEMORY	メモリが不足している。	

（9） CryptDestroyHash

API 名	CryptDestroyHash
概要	ハッシュオブジェクトの破棄を行う。

関数インターフェース	CryptDestroyHash(HCRYPTHASH hHash);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTHASH	IN	破棄するハッシュオブジェクトのハンドル
	値	内容	
エラーコード	NTE_BAD_HASH	hHash に不正な値が設定されている。	

（１０） CryptSetHashParam

API 名	CryptSetHashParam		
概要	ハッシュオブジェクトのパラメータを設定する。		
関数インターフェース	CryptSetHashParam(HCRYPTHASH hHash, DWORD dwParam, BYTE* pbData, DWORD dwFlags);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTHASH	IN	パラメータを設定するハッシュオブジェクトのハンドル
	DWORD	IN	設定するパラメータ HP_HASHVAL: pbData バッファに格納された値をハッシュ値としてハッシュオブジェクトに設定する。
	BYTE*	IN	パラメータに設定するデータのポインタ
	DWORD	IN	動作に関するパラメータ 0 を設定すること。
	値	内容	
エラーコード	NTE_BAD_HASH	hHash に不正な値が設定されている。	
	NTE_BAD_TYPE	dwParam に不正な値が設定されている。	

（１１） CryptGetHashParam

API 名	CryptGetHashParam

概要	ハッシュオブジェクトのパラメータを取得する。		
関数インターフェース	CryptGetHashParam(HCRYPTHASH hHash, DWORD dwParam, BYTE* pbData, DWORD* pdwDataLen, DWORD dwFlags);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTHASH	IN	パラメータを取得するハッシュオブジェクトのハンドル
	DWORD	IN	取得するパラメータ HP_ALGID: アルゴリズム ID を DWORD 型で返す。 HP_HASHSIZE: ハッシュサイズを DWORD 型で返す。 HP_HASHVAL: ハッシュ値を返す。
	BYTE*	OUT	値を格納するバッファのポインタ NULL を指定した場合は値の書き込みは行われず、pdwDataLen に値の格納に必要な長さが設定される。
	DWORD*	IN/OUT	値の長さを保持するバッファへのポインタ 関数呼び出し時には、pbData バッファに割り当てられたメモリサイズを指定する。関数終了時には、値の格納に必要な長さが設定される。
	DWORD	IN	動作に関するパラメータ 0 を設定すること。
	値	内容	
エラーコード	ERROR_MORE_DATA	pbData バッファが小さすぎる。	
	NTE_BAD_HASH	hHash に不正な値が設定されている。	
	NTE_BAD_TYPE	dwParam に不正な値が設定されている。	

（１２） CryptHashData

API 名	CryptHashData		
概要	ハッシュオブジェクトにデータを与えハッシュ値を計算する。		
関数インターフェース	CryptHashData(HCRYPTHASH hHash, BYTE* pbData, DWORD dwDataLen, DWORD dwFlags);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTHASH	IN	ハッシュオブジェクトのハンドル
	BYTE*	IN	ハッシュ値を計算するデータのポインタ
	DWORD	IN	ハッシュ値を計算するデータの長さ
	DWORD	IN	動作に関するパラメータ 0 を設定すること。
	値	内容	
エラーコード	NTE_BAD_ALGID	hHash で指定されたアルゴリズムはサポートしていない。	
	NTE_BAD_HASH	hHash に不正な値が設定されている。	
	NTE_BAD_HASH_STATE	ハッシュ計算は既に完了しているため、データを追加できない。	
	NTE_FAIL	予期しないエラーが発生した。	
	NTE_NO_MEMORY	メモリが不足している。	

（１３） CryptSignHash

API 名	CryptSignHash
概要	ハッシュ値に署名を行う。
関数インターフェース	CryptSignHash(HCRYPTHASH hHash, DWORD dwKeySpec, LPCTSTR sDescription, DWORD dwFlags, BYTE* pbSignature, DWORD* pdwSigLen);

戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTHASH	IN	署名を行うハッシュオブジェクトのハンドル
	DWORD	IN	鍵の種類 AT_KEYEXCHANGE: 鍵交換用鍵 AT_SIGNATURE: 署名用鍵 AT_SIGNATURE を設定すること。
	LPCTSTR	IN	ハッシュの概要についての NULL ターミネート文字列 NULL を設定すること。
	DWORD	IN	署名時のフラグ 0 を設定すること。
	BYTE*	OUT	署名データを格納するバッファのポインタ NULL を指定した場合は値の書き込みは行われず、pdwSigLen に値の格納に必要な長さが設定される。
	DWORD*	IN/OUT	署名データの長さを保持するバッファへのポインタ 関数呼び出し時には、pbSignature バッファに割り当てられたメモリサイズを指定する。関数終了時には、署名データの格納に必要な長さが設定される。
	値	内容	
エラーコード	ERROR_MORE_DATA	pbSignature バッファが小さすぎる。	
	NTE_BAD_ALGID	hHash で指定されたアルゴリズムはサポートしていない。	
	NTE_BAD_FLAGS	dwFlags に不正な値が設定されている。	
	NTE_BAD_HASH	hHash に不正な値が設定されている。	
	NTE_NO_KEY	dwKeySpec に指定した種類の鍵が存在しない。	
	NTE_NO_MEMORY	メモリが不足している。	

※本関数にて署名された署名データは OID を含みます。

（１４） CryptVerifySignature

API 名	CryptVerifySignature		
概要	署名を検証する。		
関数インターフェース	CryptVerifySignature(HCRYPTHASH hHash, BYTE* pbSignature, DWORD dwSigLen, HCRYPTKEY hPubKey, LPCTSTR sDescription, DWORD dwFlags);		
戻り値	BOOL（TRUE:成功　FALSE:失敗）		
	型	I/O	内容
引数	HCRYPTHASH	IN	検証するハッシュオブジェクトのハンドル
	BYTE*	IN	署名データのバッファへのポインタ
	DWORD	IN	署名データの長さ
	HCRYPTKEY	IN	署名の検証に使用する公開鍵のハンドル
	LPCTSTR	IN	ハッシュの概要についての NULL ターミネート文字列 NULL を設定すること。
	DWORD	IN	署名検証時のフラグ 0 を設定すること。
	値	内容	
エラーコード	NTE_BAD_FLAGS	dwFlags に不正な値が設定されている。	
	NTE_BAD_HASH	hHash に不正な値が設定されている。	
	NTE_BAD_KEY	hPubKey に不正な値が設定されている。	
	NTE_BAD_SIGNATURE	署名の検証に失敗した。	
	NTE_BAD_UID	hProv に不正な値が設定されている。	
	NTE_NO_MEMORY	メモリが不足している。	

第３節　構造体仕様

（1） PUBLICKEYSTRUC

構造体名		PUBLICKEYSTRUC		
概要		公開鍵の BLOB ヘッダを格納する構造体。		
NO	変数名	型	値	備考
1	bType	BYTE	BLOB タイプ	PUBLICKEYBLOB を指定する。
2	bVersion	BYTE	BLOB バージョン	CUR_BLOB_VERSION を指定する。
3	reserved	WORD	未使用	-
4	aiKeyAlg	ALG_ID	アルゴリズム ID	CALG_RSA_KEYX または CALG_RSA_SIGN を指定する。

（2） RSAPUBKEY

構造体名		RSAPUBKEY		
概要		公開鍵の BLOB ヘッダを格納する構造体。		
NO	変数名	型	値	備考
1	magic	DWORD	鍵のタイプ	RSA1(0x31415352)を指定する。
2	bitlen	DWORD	鍵長	1024(鍵長)または、2048(鍵長)を指定する。
3	pubexp	DWORD	public exponent	public exponent を指定する。

（3） Public-key BLOBs

構造体名		なし(構造体としては定義されていない)		
概要		公開鍵 BLOB		
NO	変数名	型	値	備考
1	publickeystruc	PUBLICKEYSTRUC	PUBLICKEYSTRUC 構造体	PUBLICKEYSTRUC 構造体を指定する。
2	rsapubkey	RSAPUBKEY	RSAPUBKEY 構造体	RSAPUBKEY 構造体を指定する。
3	BYTE	DWORD	Modulus	Modulus を指定する。

第４節　コーリングシーケンス

「第４章　第２節　実現可能な機能の一覧」を実現するためのコーリングシーケンスを以下に示す。上位アプリケーションは、このコーリングシーケンスに沿って実装すること。

注意事項

・複数のコーリングシーケンスを並行して実行しないこと。
　１つのシーケンスを開始したなら、そのシーケンスを完了してから次のシーケンスを開始してください。

（1） 利用者証明書取得処理

CryptAcquireContext	鍵コンテナ生成 phProv: ハンドル格納領域アドレス pszContainer: NULL pszProvider: CSP 名称 dwProvType: プロバイダタイプ dwFlags: 0
↓	
CryptGetUserKey	利用者鍵ハンドル取得 hProv: CryptAcquireContext で取得したハンドル dwKeySpec: AT_SIGNATURE phUserKey: 利用者鍵ハンドル格納領域アドレス
↓	
CryptGetKeyParam	利用者証明書サイズ取得 hKey: 利用者鍵ハンドル dwParam: KP_CERTIFICATE pbData: NULL pdwDataLen: 利用者証明書長格納領域アドレス dwFlags: 0
↓	
CryptGetKeyParam	利用者証明書取得 hKey: 利用者鍵ハンドル dwParam: KP_CERTIFICATE pbData: 利用者証明書格納領域アドレス pdwDataLen: 利用者証明書長格納領域アドレス dwFlags: 0
↓	
CryptDestroyKey	利用者鍵ハンドル破棄 hKey: 破棄する鍵ハンドル

↓

CryptReleaseContext	鍵コンテナ開放 hProv: CryptAcquireContext で取得したハンドル dwFlags: 0

（2） 認証局の自己署名証明書取得処理

CryptAcquireContext	鍵コンテナ生成 phProv: ハンドル格納領域アドレス pszContainer: NULL pszProvider: CSP 名称 dwProvType: プロバイダタイプ dwFlags: CRYPT_VERIFYCONTEXT

↓

CryptGetProvParam	認証局の自己署名証明書サイズ取得 hProv: CryptAcquireContext で取得したハンドル dwParam: PP_JPKI_CA_CERTIFICATE pbData: NULL pdwDataLen: 認証局の自己署名証明書長格納領域アドレス dwFlags: 0

↓

CryptGetProvParam	認証局の自己署名証明書取得 hProv: CryptAcquireContext で取得したハンドル dwParam: PP_JPKI_CA_CERTIFICATE pbData: 認証局の自己署名証明書格納領域アドレス pdwDataLen: 認証局の自己署名証明書長格納領域アドレス dwFlags: 0

↓

CryptReleaseContext	鍵コンテナ開放 hProv: CryptAcquireContext で取得したハンドル dwFlags: 0

（3） 署名生成処理(署名対象データを渡すパターン)[*1]

CryptAcquireContext	鍵コンテナ生成 phProv: ハンドル格納領域アドレス

[*1] 署名生成処理(署名対象データを渡すパターン)のコーリングシーケンスを実行することで、IC カード内の利用者証明書、署名対象データに対するハッシュ値および署名値を取得することができる。

	pszContainer: NULL pszProvider: CSP 名称 dwProvType: プロバイダタイプ dwFlags: 0

↓

CryptGetUserKey	利用者鍵ハンドル取得 hProv: CryptAcquireContext で取得したハンドル dwKeySpec: AT_SIGNATURE phUserKey: 利用者鍵ハンドル格納領域アドレス

↓

CryptGetKeyParam	利用者証明書サイズ取得 hKey: 利用者鍵ハンドル dwParam: KP_CERTIFICATE pbData: NULL pdwDataLen: 利用者証明書長格納領域アドレス dwFlags: 0

↓

CryptGetKeyParam	利用者証明書取得 hKey: 利用者鍵ハンドル dwParam: KP_CERTIFICATE pbData: 利用者証明書格納領域アドレス pdwDataLen: 利用者証明書長格納領域アドレス dwFlags: 0

↓

CryptDestroyKey	利用者鍵ハンドル破棄 hKey: 利用者鍵ハンドル

↓

CryptCreateHash	ハッシュオブジェクト生成 hProv: CryptAcquireContext で取得したハンドル ALG_ID: アルゴリズム ID(CALG_SHA1 または CALG_SHA256 のいずれかを署名方式に合わせて指定) hKey: 0 dwFlags: 0 phHash: ハッシュオブジェクトのハンドル格納領域アドレス

↓

CryptHashData	ハッシュ値計算 hHash: ハッシュオブジェクトのハンドル

	pbData: 署名対象データ dwDataLen: 署名対象データ長 dwFlags: 0
↓	
CryptGetHashParam	ハッシュ値取得 hHash: ハッシュオブジェクトのハンドル dwParam: HP_HASHVAL pbData: NULL pdwDataLen: ハッシュデータ長格納領域アドレス dwFlags: 0
↓	
CryptGetHashParam	ハッシュ値取得 hHash: ハッシュオブジェクトのハンドル dwParam: HP_HASHVAL pbData: ハッシュデータ格納領域アドレス pdwDataLen: ハッシュデータ長格納領域アドレス dwFlags: 0
↓	
CryptSignHash	署名値長取得 hHash: 署名対象ハッシュオブジェクトのハンドル dwKeySpec: AT_SIGNATURE sDescription: NULL dwFlags: 0 pbSignature: NULL pdwSigLen: 署名データ長格納領域アドレス
↓	
CryptSignHash	署名 hHash: 署名対象ハッシュオブジェクトのハンドル dwKeySpec: AT_SIGNATURE sDescription: NULL dwFlags: 0 pbSignature: 署名データ格納領域アドレス pdwSigLen: 署名データ長格納領域アドレス
↓	
CryptDestroyHash	ハッシュオブジェクト破棄 hHash: 破棄するハッシュオブジェクトハンドル
↓	
CryptReleaseContext	鍵コンテナ開放

	hProv: CryptAcquireContext で取得したハンドル dwFlags: 0

（4） 署名生成処理(ハッシュ値を渡すパターン)*2

CryptAcquireContext	鍵コンテナ生成 phProv: ハンドル格納領域アドレス pszContainer: NULL pszProvider: CSP 名称 dwProvType: プロバイダタイプ dwFlags: 0
↓	
CryptGetUserKey	利用者鍵ハンドル取得 hProv: CryptAcquireContext で取得したハンドル dwKeySpec: AT_SIGNATURE phUserKey: 利用者鍵ハンドル格納領域アドレス
↓	
CryptGetKeyParam	利用者証明書サイズ取得 hKey: 利用者鍵ハンドル dwParam: KP_CERTIFICATE pbData: NULL pdwDataLen: 利用者証明書長格納領域アドレス dwFlags: 0
↓	
CryptGetKeyParam	利用者証明書取得 hKey: 利用者鍵ハンドル dwParam: KP_CERTIFICATE pbData: 利用者証明書格納領域アドレス pdwDataLen: 利用者証明書長格納領域アドレス dwFlags: 0
↓	
CryptDestroyKey	利用者鍵ハンドル破棄 hKey: 利用者鍵ハンドル
↓	
CryptCreateHash	ハッシュオブジェクト生成 hProv: CryptAcquireContext で取得したハンドル ALG_ID: アルゴリズム ID(CALG_SHA1 または

*2 署名生成処理(ハッシュ値を渡すパターン)のコーリングシーケンスを実行することで、IC カード内の利用者証明書、ハッシュ値に対する署名値を取得することができる。

	CALG_SHA256 のいずれかを署名方式に合わせて指定) hKey: 0 dwFlags: 0 phHash: ハッシュオブジェクトのハンドル格納領域アドレス

↓

CryptSetHashParam	ハッシュ値設定 hHash: ハッシュオブジェクトのハンドル dwParam: HP_HASHVAL pbData: ハッシュ値の格納領域アドレス dwFlags: 0

↓

CryptSignHash	署名値長取得 hHash: 署名対象ハッシュオブジェクトのハンドル dwKeySpec: AT_SIGNATURE sDescription: NULL dwFlags: 0 pbSignature: NULL pdwSigLen: 署名データ長格納領域アドレス

↓

CryptSignHash	署名 hHash: 署名対象ハッシュオブジェクトのハンドル dwKeySpec: AT_SIGNATURE sDescription: NULL dwFlags: 0 pbSignature: 署名データ格納領域アドレス pdwSigLen: 署名データ長格納領域アドレス

↓

CryptDestroyHash	ハッシュオブジェクト破棄 hHash: 破棄するハッシュオブジェクトハンドル

↓

CryptReleaseContext	鍵コンテナ開放 hProv: CryptAcquireContext で取得したハンドル dwFlags: 0

（5） 署名検証処理(検証対象データを渡すパターン)

CryptAcquireContext	鍵コンテナ生成 phProv: ハンドル格納領域アドレス

	pszContainer: NULL pszProvider: CSP 名称 dwProvType: プロバイダタイプ dwFlags: CRYPT_VERIFYCONTEXT
↓	
CertCreateCertificateContext (本 CSP の対象外。OS 標準機能を使用する。)	証明書コンテキスト生成 dwCertEncodingType: X509_ASN_ENCODING pbCertEncoded: X.509 DER 形式の証明書データ cbCertEncoded: 証明書長
↓	
CryptImportPublicKeyInfo (CryptImportKey がこの関数の内部でコールされる)	公開鍵インポート hCryptProv: CryptAcquireContext で取得したハンドル dwCertEncodingType: X509_ASN_ENCODING pInfo: 公開鍵情報 CertCreateCertificateContext の戻り値を pcCert とした時、&pcCert->pCertInfo->SubjectPublicKeyInfo phKey: 公開鍵ハンドル格納領域アドレス
↓	
CertFreeCertificateContext (本 CSP の対象外。OS 標準機能を使用する。)	証明書コンテキスト破棄 pCertContext: 破棄する証明書コンテキスト
↓	
CryptCreateHash	ハッシュオブジェクト生成 hProv: CryptAcquireContext で取得したハンドル ALG_ID: アルゴリズム ID(CALG_SHA1 または CALG_SHA256 のいずれかを署名方式に合わせて指定) hKey: 0 dwFlags: 0 phHash: ハッシュオブジェクトのハンドル格納領域アドレス
↓	
CryptHashData	ハッシュ値計算 hHash: ハッシュオブジェクトのハンドル pbdata: ハッシュ値計算対象データ dwDataLen: ハッシュ値計算対象データ長 dwFlags: 0
↓	

CryptVerifySignature	署名検証 hHash: ハッシュオブジェクトのハンドル pbSignature: 署名データ格納領域のアドレス dwSigLen: 署名データ長 hPubKey: 公開鍵ハンドル sDescription: NULL dwFlags: 0
↓	
CryptDestroyHash	ハッシュオブジェクト破棄 hHash: 破棄するハッシュオブジェクトハンドル
↓	
CryptDestroyKey	公開鍵破棄 hKey: 破棄する公開鍵ハンドル
↓	
CryptReleaseContext	鍵コンテナ開放 hProv: CryptAcquireContext で取得したハンドル dwFlags: 0

（6） 署名検証処理(ハッシュ値を渡すパターン)

CryptAcquireContext	鍵コンテナ生成 phProv: ハンドル格納領域アドレス pszContainer: NULL pszProvider: CSP 名称 dwProvType: プロバイダタイプ dwFlags: CRYPT_VERIFYCONTEXT
↓	
CertCreateCertificateContext (本 CSP の対象外。OS 標準機能を使用する。)	証明書コンテキスト生成 dwCertEncodingType: X509_ASN_ENCODING pbCertEncoded: X.509 DER 形式の証明書データ cbCertEncoded: 証明書長
↓	
CryptImportPublicKeyInfo	公開鍵インポート hCryptProv: CryptAcquireContext で取得したハンドル dwCertEncodingType: X509_ASN_ENCODING pInfo: 公開鍵情報 CertCreateCertificateContext の戻り値を pcCert とした時、&pcCert->pCertInfo->SubjectPublicKeyInfo

	phKey: 公開鍵ハンドル格納領域アドレス

↓

CertFreeCertificateContext (本 CSP の対象外。OS 標準機能を使用する。)	証明書コンテキスト破棄 pCertContext: 破棄する証明書コンテキスト

↓

CryptCreateHash	ハッシュオブジェクト生成 hProv: CryptAcquireContext で取得したハンドル ALG_ID: アルゴリズム ID(CALG_SHA1 または CALG_SHA256 のいずれかを署名方式に合わせて指定) hKey: 0 dwFlags: 0 phHash: ハッシュオブジェクトのハンドル格納領域アドレス

↓

CryptSetHashParam	ハッシュ値設定 hHash: ハッシュオブジェクトのハンドル dwParam: HP_HASHVAL pbData: ハッシュ値の格納領域アドレス dwFlags: 0

↓

CryptVerifySignature	署名検証 hHash: ハッシュオブジェクトのハンドル pbSignature: 署名データ格納領域のアドレス dwSigLen: 署名データ長 hPubKey: 公開鍵ハンドル sDescription: NULL dwFlags: 0

↓

CryptDestroyHash	ハッシュオブジェクト破棄 hHash: 破棄するハッシュオブジェクトハンドル

↓

CryptDestroyKey	公開鍵破棄 hKey: 破棄する公開鍵ハンドル

↓

CryptReleaseContext	鍵コンテナ開放 hProv: CryptAcquireContext で取得したハンドル dwFlags: 0

（7） 繰り返し署名生成処理(署名対象データを渡すパターン)

「（3）署名生成処理(署名対象データを渡すパターン)」の網掛け部分を署名対象データの個数分だけ繰り返して呼び出す。

（8） 繰り返し署名生成処理(ハッシュ値を渡すパターン)

「（4）署名生成処理(ハッシュ値を渡すパターン)」の網掛け部分をハッシュ値の個数分だけ繰り返して呼び出す。

（9） 繰り返し署名検証処理(検証対象データを渡すパターン)

「（5）署名検証処理(検証対象データを渡すパターン)」の網掛け部分を検証対象データの個数分だけ繰り返して呼び出す。(但し、すべての電子署名が同一の秘密鍵で生成された場合とする。)

（１０） 繰り返し署名検証処理(ハッシュ値を渡すパターン)

「（6）署名検証処理(ハッシュ値を渡すパターン)」の網掛け部分をハッシュ値の個数分だけ繰り返して呼び出す。(但し、すべての電子署名が同一の秘密鍵で生成された場合とする。)

第６章　画面仕様

第１節　画面一覧

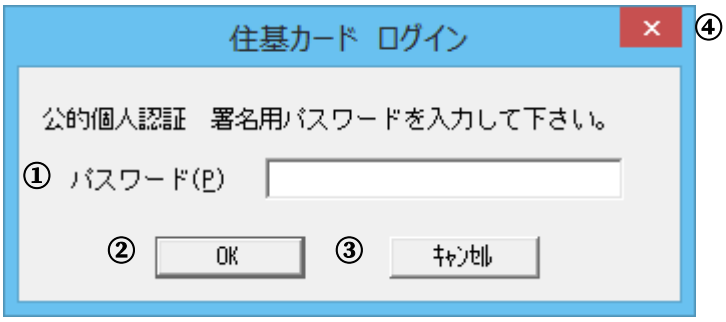
NO	機能名	画面名	機能概要
1	ログイン機能	住基カード署名用パスワード入力画面	住基カードの公的個人認証　署名用パスワード入力要求を行う。
2		個人番号カード署名用パスワード入力画面	個人番号カードの公的個人認証　署名用パスワード入力要求を行う。
3		個人番号カード利用者証明用パスワード入力画面	個人番号カードの公的個人認証　利用者証明用パスワード入力要求を行う。

第２節　画面仕様詳細

（1） 住基カード署名用パスワード入力画面

画面名	住基カード署名用パスワード入力画面
概要	住基カードの公的個人認証　署名用パスワード入力を要求し、入力されたパスワードで公的個人認証サービスカード AP にログインを行う。

画面レイアウト



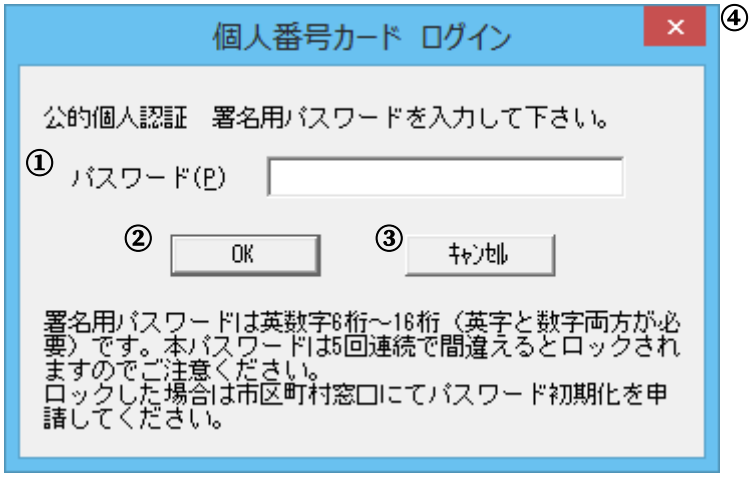
画面項目説明

NO	項目名	概要
①	パスワード	住基カードの公的個人認証　署名用パスワードを入力する。
②	OK	入力したパスワードで IC カードにログインを行う。 ログインできない場合は以下のメッセージを表示する。 ・ パスワードが誤っている場合 →「パスワードが違います。」 ・ ロックしている場合 →「パスワードの確認に失敗しました。」 ・ ログイン時に予期しないエラーが発生した場合 →「予期しないエラーが発生しました。(エラーコード:XXXXX)」
③	キャンセル	パスワード入力を行わずにダイアログを閉じる。
④	×	パスワード入力を行わずにダイアログを閉じる。

（2） 個人番号カード署名用パスワード入力画面

画面名	個人番号カード署名用パスワード入力画面
概要	個人番号カードの公的個人認証　署名用パスワード入力を要求し、入力された署名用パスワードで公的個人認証サービスカード AP にログインを行う。

画面レイアウト



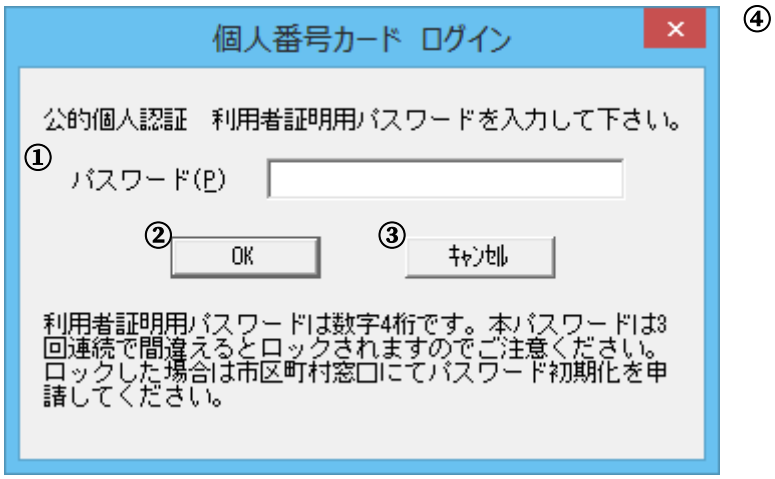
画面項目説明

NO	項目名	概要
①	パスワード	個人番号カードの公的個人認証　署名用パスワードを入力する。
②	OK	入力した署名用パスワードで IC カードにログインを行う。 ログインできない場合は以下のメッセージを表示する。 ・ パスワードが誤っている場合 →「パスワードが違います。」 ・ ロックしている場合 →「パスワードの確認に失敗しました。」 ・ ログイン時に予期しないエラーが発生した場合 →「予期しないエラーが発生しました。(エラーコード:XXXXX)」
③	キャンセル	署名用パスワード入力を行わずにダイアログを閉じる。
④	×	署名用パスワード入力を行わずにダイアログを閉じる。

（3） 個人番号カード利用者証明用パスワード入力画面

画面名	個人番号カード利用者証明用パスワード入力画面
概要	個人番号カードの公的個人認証　利用者証明用パスワード入力を要求し、入力された利用者証明用パスワードで公的個人認証サービスカード AP にログインを行う。

画面レイアウト



画面項目説明

NO	項目名	概要
①	パスワード	個人番号カードの公的個人認証　利用者証明用パスワードを入力する。
②	OK	入力した利用者証明用パスワードで IC カードにログインを行う。 ログインできない場合は以下のメッセージを表示する。 ・ パスワードが誤っている場合 →「パスワードが違います。」 ・ ロックしている場合 →「パスワードの確認に失敗しました。」 ・ ログイン時に予期しないエラーが発生した場合 →「予期しないエラーが発生しました。(エラーコード:XXXXX)」
③	キャンセル	利用者証明用パスワード入力を行わずにダイアログを閉じる。
④	×	利用者証明用パスワード入力を行わずにダイアログを閉じる。

禁・無断転載

公的個人認証サービス

利用者クライアントソフト API 仕様書
【カード AP ライブラリ CryptoAPI 編】

第 4.3 版

（注意事項）

※利用者クライアントソフトの著作権は、総務省、地方公共団体情報システム機構が保有しており、国際著作権条約及び日本国の著作権関連法令によって保護されています。

※利用者クライアントソフトの利用に当たっては､次に掲げる行為を禁止します。

（1） 利用者クライアントソフトを電子署名に係る地方公共団体情報システム機構の認証業務に関する法律において制限されている電子証明書の用途で利用すること。

（2） 利用者クライアントソフトに対し、総務省、地方公共団体情報システム機構に許可なく改造等を行うこと。

※総務省、地方公共団体情報システム機構は、利用者が利用者クライアントソフトを利用したことにより発生した利用者の損害及び利用者が第三者に与えた損害について、一切の責任を負いません。

※商標については次の通りです。

（1） Microsoft Windows および Internet Explorer は、米国 Microsoft Corporation の米国およびその他の国における登録商標または商標です。

（2） Android は、Google Inc.の米国およびその他の国における登録商標です。

（3） その他、記載されている会社名、製品名等は、各社の登録商標または商標です。