CONFIDENTIAL

Smart-in デベロッパーズガイド

第 1.6 版 2015 年 4 月 13 日

改訂履歴

Ver.	年月日	内容	備考
1.0	2014/10/31	初版 ユーザーズガイドを「導入手順ガイド」と「デベロッパーズガイド」に分離	
1.1	2014/11/21	「5.疎通確認方法について」説明内容を削除。	
1.2	2014/11/26	「5 別紙」追記 ・5.1.1 C#用	
1.3	2014/12/16	サンプルソースの動作確認サイト」を追記	
1.4	2014/12/24	「3.3.1　認証依頼データ（JSON コード）の作成」へ追記 「4.2.2 サンプルコード」へ画面遷移を追記	
1.5	2015/1/27	「3.1 開発リソースについて」 提供リソース一覧内の「2.Smart-in 導入手順ガイド.pdf」の記述を削除。連番を変更。	
1.6	2015/4/13	・「4.1Smart-in 接続サンプル利用の注意点」について 動作環境修正 ・「4.2.2 サンプルコード」コメントの誤記を修正 // result 0:正常 9:異常	

目次

1. はじめに

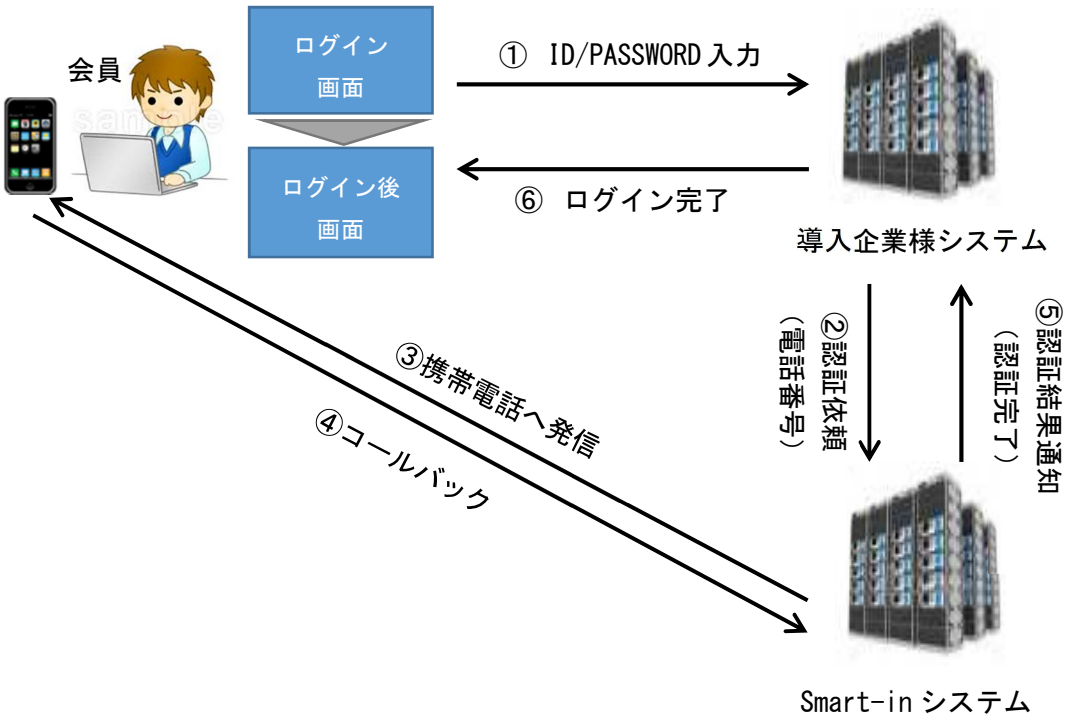
本書は、Smart-in システム導入の為の知識および開発方法について説明しています。

本書は、Smart-in に接続するシステム（以降、導入企業様システム）の開発者を対象に書かれています。

2. Smart-in システム概要

2.1. 概要

導入企業様システムの個人認証に Smart-in 認証を組み込んだ場合のイメージを下記に示します。



① ログイン画面に ID/PASSWORD を入力します

② 導入企業様システム内に予め登録されている電話番号を取り出し、その電話番号に対する認証依頼を Smart-in システムへ送信します
Smart-in システムから認証結果通知が返送されてくるまで数十秒要する為、画面には「認証中です」などを表示します。

③ 依頼された電話番号へ発信する。 携帯電話側では、数回コール音が鳴り自動的に切断されまので応答する必要はありません。

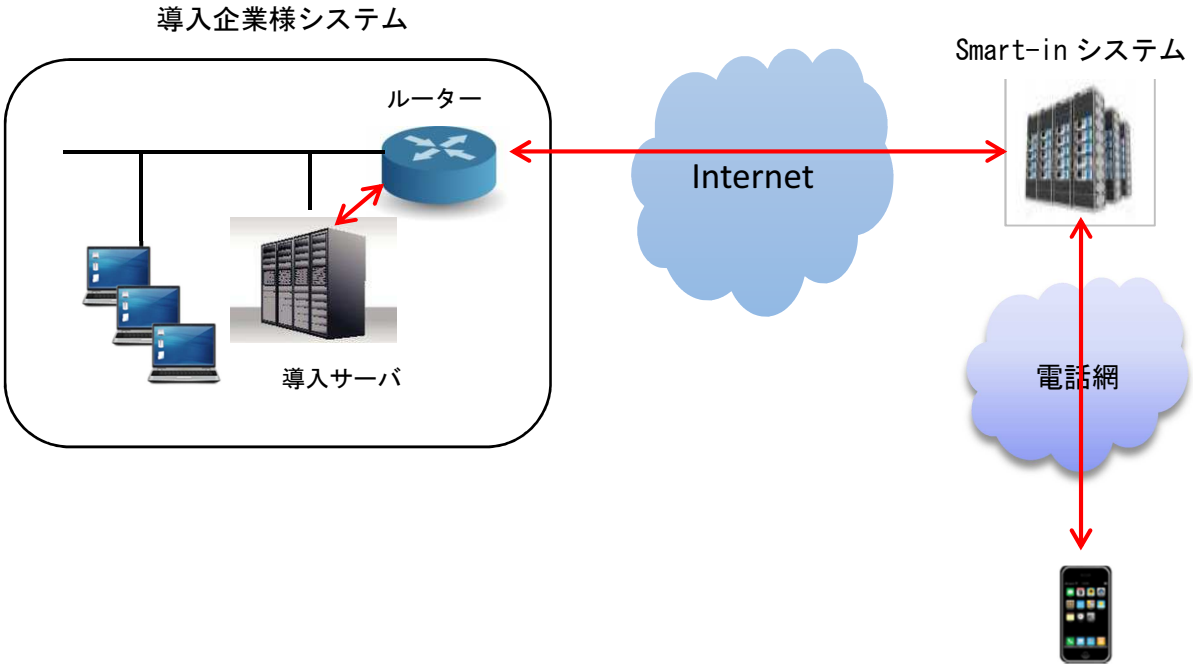
④ 着信履歴などからかかってきた電話番号へコールバックします。Smart-in システム側で発信番号を認識すると応答せず自動的に切断しますので課金されません。

⑤ 導入企業様システムへ認証結果「認証完了」を返します。
一定時間コールバックがなかった場合は、「認証タイムアウト」を返します。

⑥ Smart-in 認証が完了した場合は、正常ログインとしてログイン後の画面へ遷移します。

2.2. インターネットを介した接続構成

導入企業様システムが Smart-in システムとインターネットを介して接続される場合のイメージ図と諸条件について記述いたします。



2.2.1. 諸条件

① Smart-in システムに接続するサーバにグローバル IP アドレスが必要になります。
Smart-in システム側から認証結果を https で POST する為です。
接続するサーバがイントラネット内に設定されている場合は、ルータにてポートフォワードが必要になります。

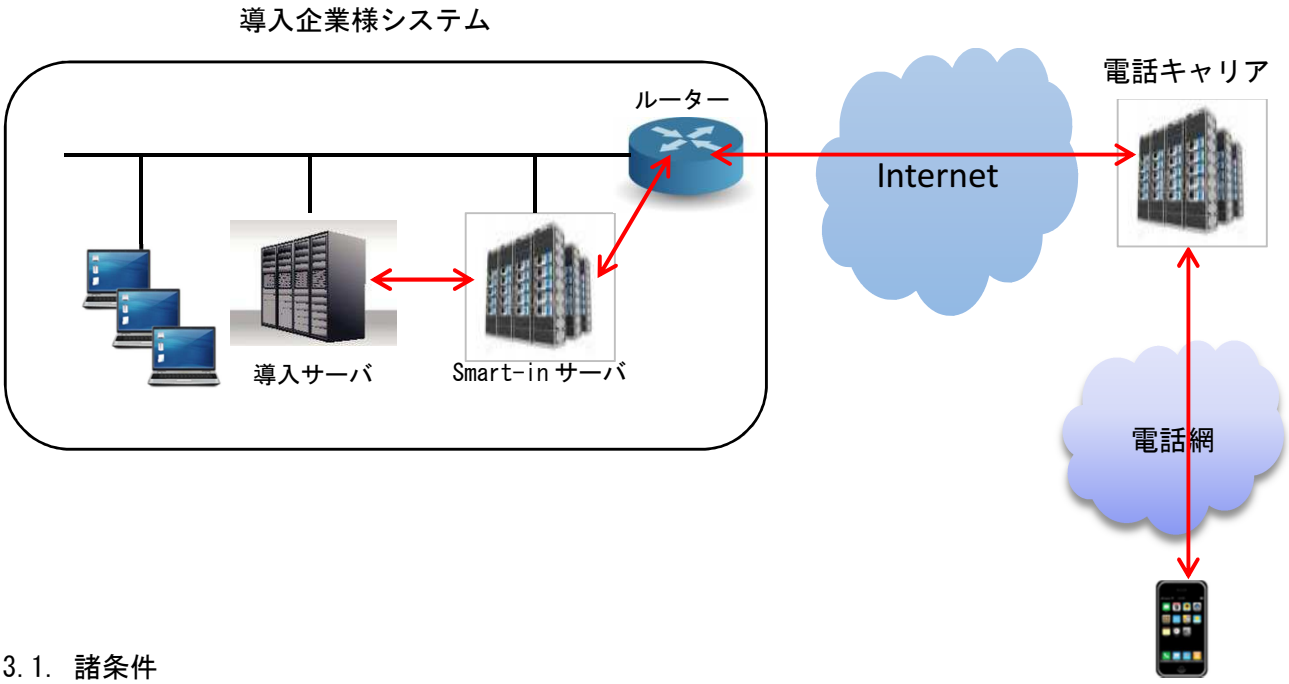
② Smart-in システムに接続するサーバ内に OpenSSL（※）のミドルウェアが必要になります。
暗号化／復号化モジュール（sin_crypt）で必要になります。
※Linux の場合（Windows の場合は、再頒布パッケージが必要になります）

③ Smart-in システムと接続には、下記プロトコル／ポートを使用いたしますので
ファイアウォールを介す場合は、下記プロトコル／ポートを解放して頂く必要があります。
https/443

④ Smart-in システムから携帯電話へ発信する番号（発番号）は、１契約当り原則１番号となりますが、複数番号の設定も可能です。

2.3. 自社ネットワーク内にオンプレミス形式で設置した場合の接続構成

導入企業様システムが Smart-in システムを自社内のネットワークに設置した場合のイメージ図と諸条件について記述いたします。



2.3.1. 諸条件

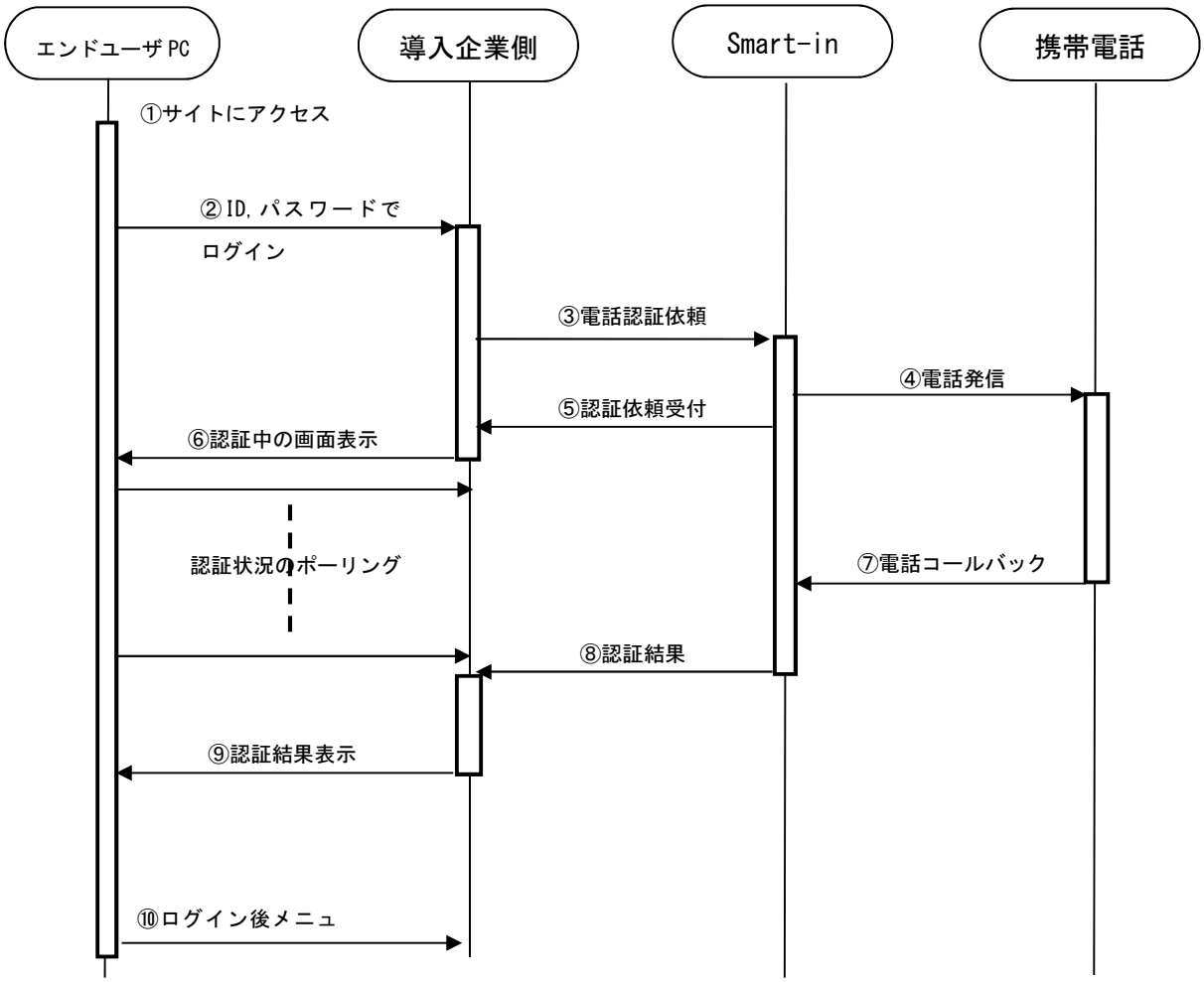
① Smart-in サーバに接続するサーバ内に OpenSSL（※）のミドルウェアが必要になります。
Smart-in との通信パケットの暗号化／復号化処理で必要になります。
暗号化／復号化モジュール（sin_crypt）で必要になります。
※Linux の場合（Windows の場合は、再頒布パッケージが必要になります）

② Smart-in サーバは、電話キャリアと接続する為、下記プロトコル／ポートを使用いたします。
ルーター・ファイアウォールを介す場合は、下記プロトコル／ポートを解放して頂く必要があります。

- 外部向け通信（DMZ から Internet）
  UDP/5060
- 内部向け通信（Internet から DMZ）
  UDP/9051

2.4. Smart-in サービスシーケンス

2.4.1. 認証依頼シーケンス



3. 開発概要

3.1. 開発リソースについて

開発用に提供するリソースついて説明いたします。

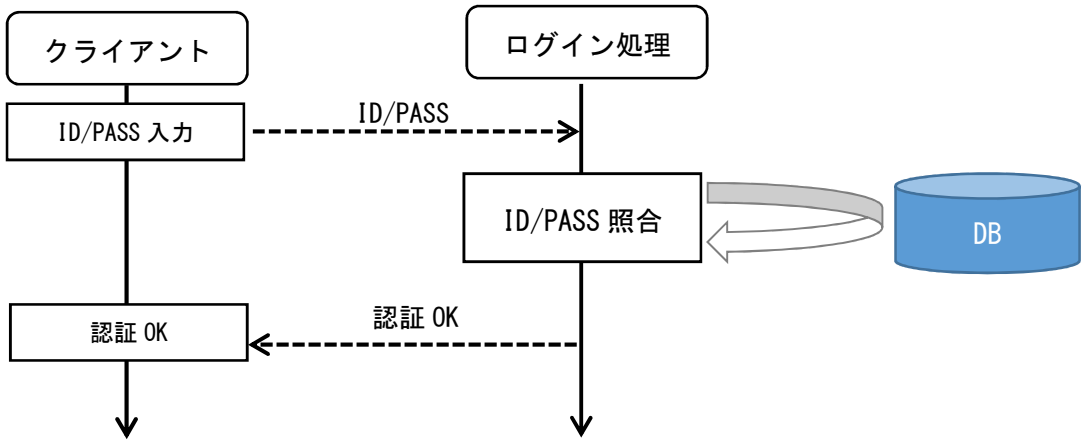
提供リソース一覧（Smart-in フォルダ内）

No.	ディレクトリ名/ファイル名	概要
1	Smart-in デベロッパーズガイド.pdf	本資料です。 開発者向けに、インタフェース仕様の説明、サンプルコードなどを記載しています。
2	インタフェース仕様書.pdf	Smart-in システムとのインタフェース仕様を記載しています。
3	Lib/Java /smart-in-api.jar	アプリケーションレベルでの暗号化/復号化を実施する為のモジュールです。 導入企業様システムの環境に合わせてご利用ください No.3 Java 用 No.4,5 CentOS 用 No.6 Windows Server （2003R2 以降） 用 ※OS の種類が異なる場合は別途プログラムソースからのコンパイルの必要があるため、サポートセンターまでお問い合わせください。
4	Lib /Linux/32bit/sin_crypt	
5	Lib /Linux/64bit/sin_crypt	
6	Lib/Windows/sin_crypt.exe	
7	sample/php/	PHP 向けサンプルファイルが格納されているディレクトリです。 利用方法は、「4 実装例」を参照下さい。
8	sample/C#/	Windows 向けサンプルファイルが格納されているディレクトリです。 利用方法は、「5.1.1　C#用」を参照下さい。

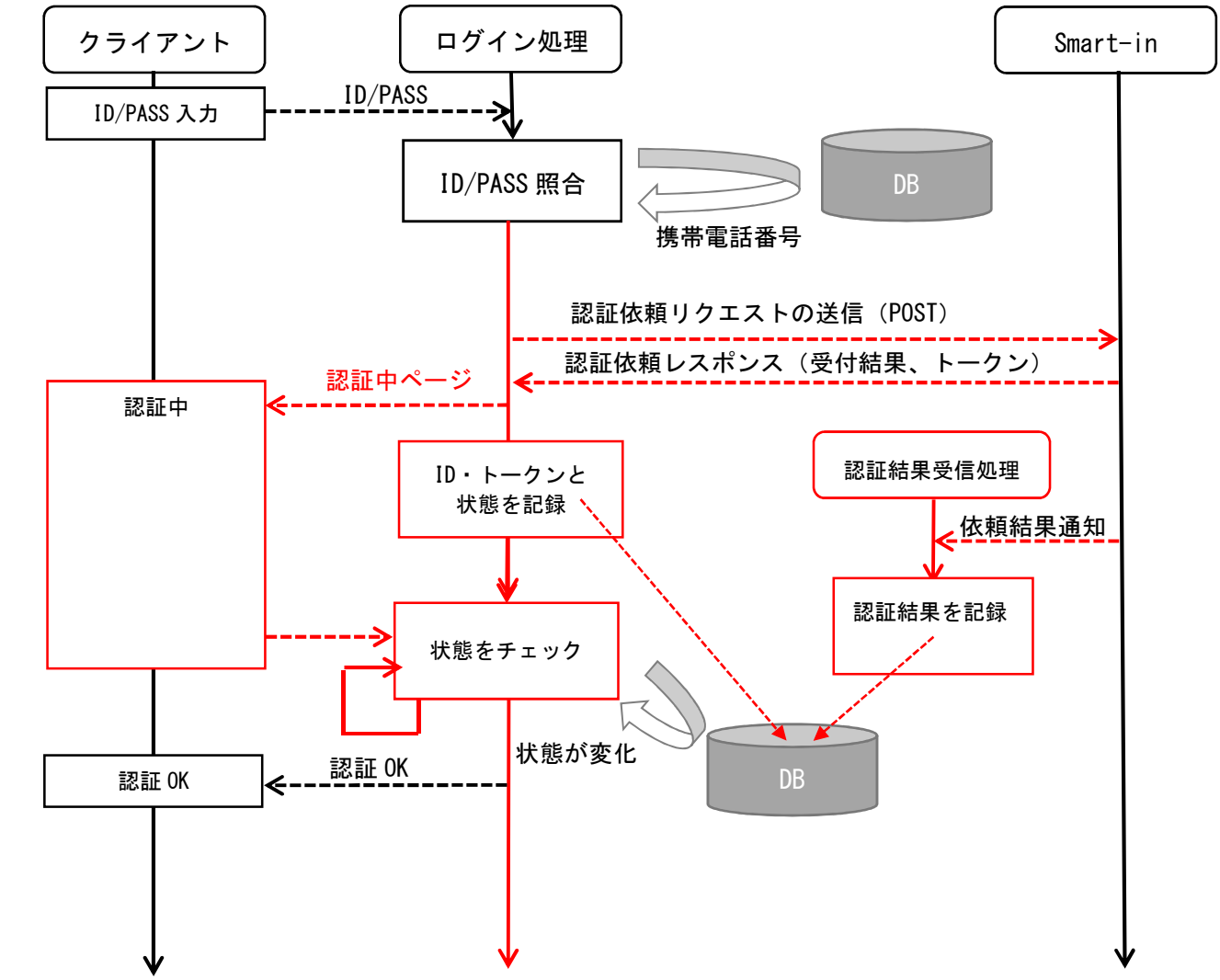
3.2. 開発項目について

Smart-in システムを利用するためには、既存のシステムに機能追加が必要となります。

既存処理イメージ



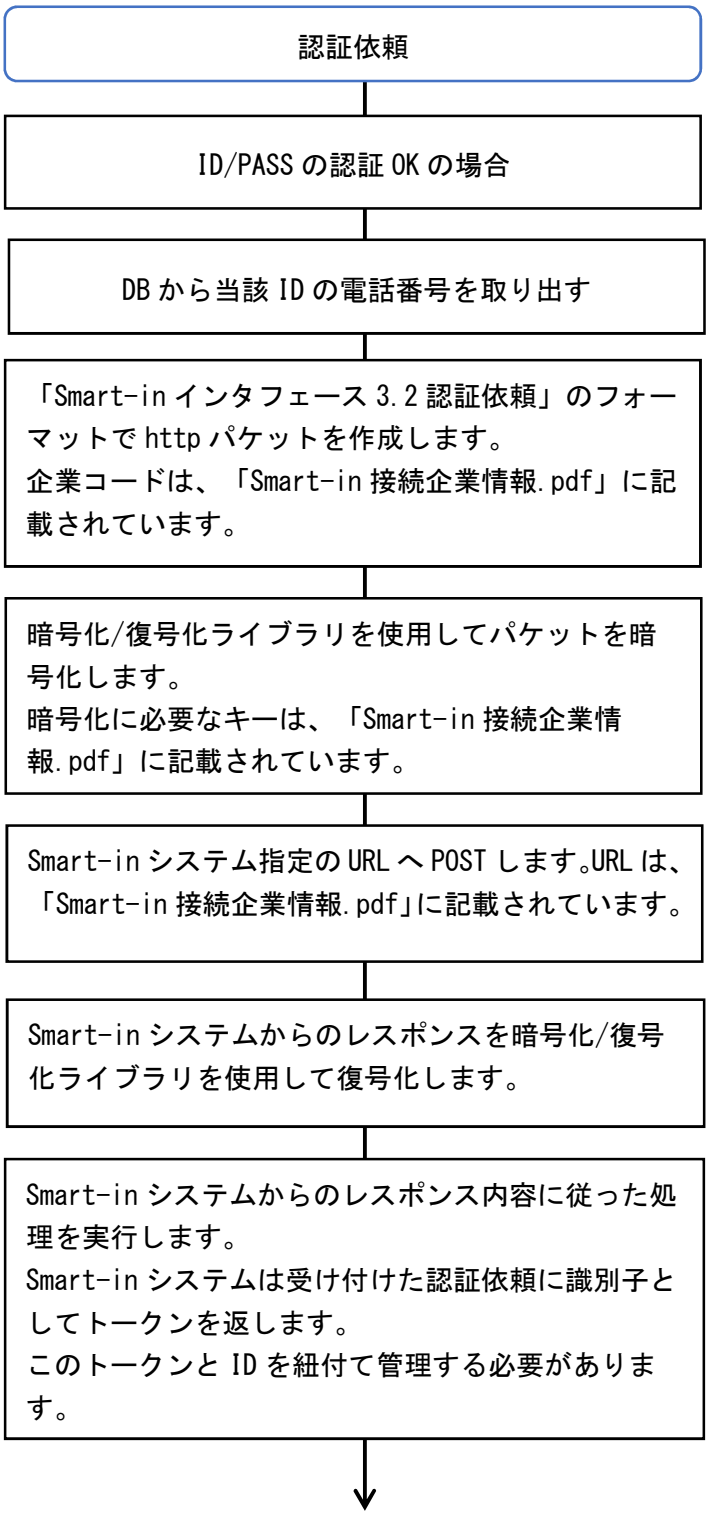
Smart-in 認証処理イメージ



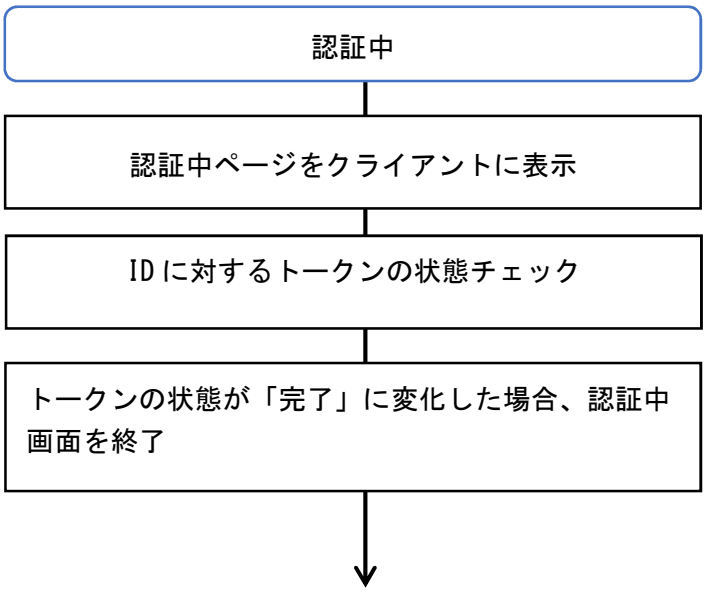
追加開発機能概要一覧

No.	機能	処理概要
1	認証依頼送信	Smart-in に対して電話認証依頼をする。 ・http リクエストパケット作成 ・http リクエストパケット暗号化 ・http リクエストパケットの POST 送信（SSL） ・http レスポンスパケットの受信 ・http レスポンスパケットの復号化
2	認証中を表示する画面	認証中の画面をクライアントに表示する。 ・認証中 ID＆トークンの状態チェック ・認証中 ID＆トークンの状態が「完了」に変化した場合 認証中画面を終了し、結果画面を表示する
3	認証中状態の管理	認証依頼中の ID とトークン、その状態を管理する。 ・認証依頼のレスポンスで返されるトークンを記録 状態：認証中
4	認証結果通知受信	Smart-in から認証結果通知を受信する。 ・認証結果通知パケットの復号化 ・http レスポンスパケットを送信 ・認証結果通知パケット中に含まれるトークンに紐づく ID の状態を「完了」に書き換える。 異常結果の場合は、詳細コードも記録する。

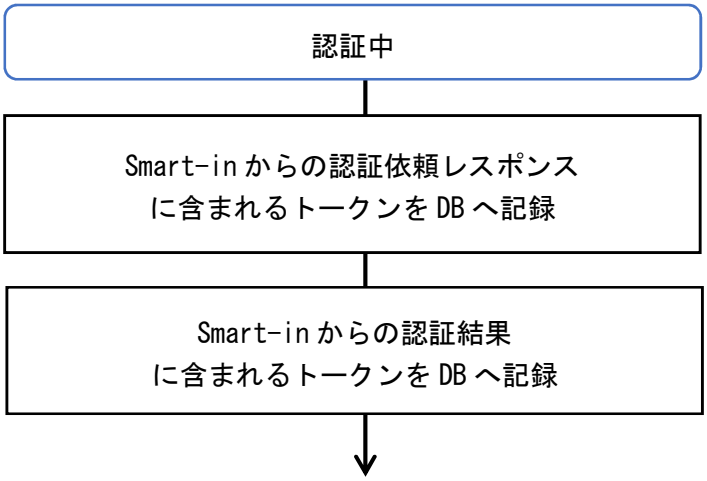
3.2.1. 認証依頼送信機能について



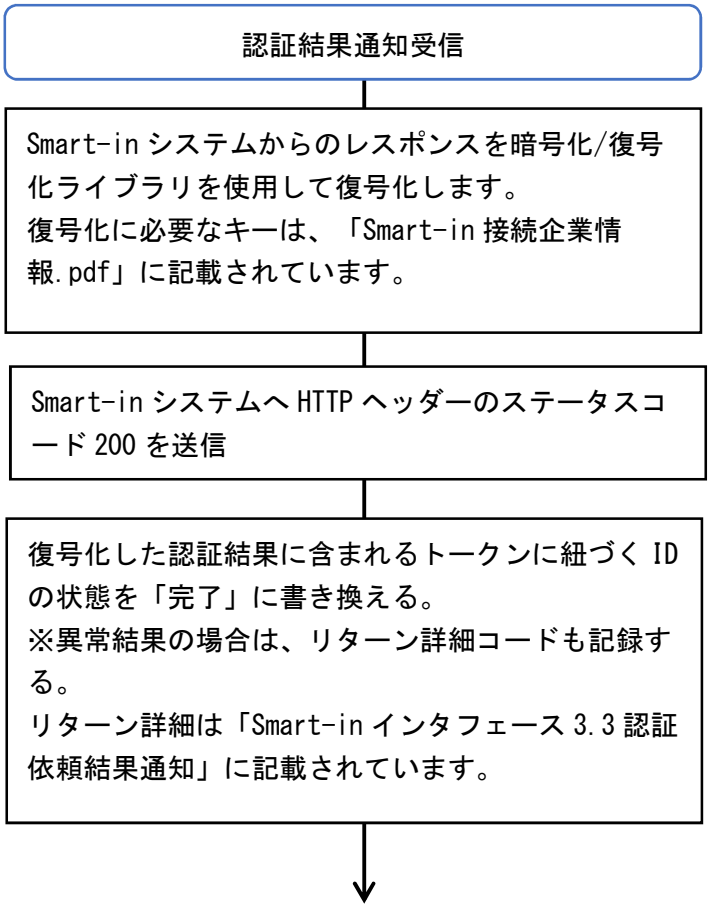
3.2.2. 認証中を表示する画面機能について



3.2.3. 認証中状態の管理機能について



3.2.4. 認証結果通知受信機能について



3.3. プログラミング例

3.3.1. 認証依頼データ（JSON コード）の作成

インタフェース仕様に沿って、JSON データを生成します。
※電話番号はハイフンや空白など、数字以外を取り除いて下さい。

依頼データ例：

```
{
     “company”: “0001”
    ″code″: ″C51″,
    ″telno″: “09011112222”,
     “response_url” : “https://202.11.22.33/response/”,
     “timer” : 120
}
```

3.3.2. 認証依頼データ暗号化

前項で生成した JSON データを暗号化/復号化モジュールを使用し暗号化します。

※暗号化の方法は使用するライブラリによって異なります。
また、java ライブラリを使用する際は、暗号化と同時に送信処理までを行います。

暗号化コマンド例（Linux 用 sin_crypt 使用時）
※下記は、Linux コマンドライン例です。
※コマンドラインには改行を含めません。

```
./sin_crypt -e 1234567890abcdef1234567890abcdef '{ ″company″:″0001″, ″code″:″C51″, ″
telno″:″09011112222″, ″response_url″:″https://202.11.22.33/response.php″, ″timer″:12
0 } '
```

↓暗号化
暗号化されたデータ例：

```
2216eaf2a7fccaf729dab94817d72456934b6a8e9903c3257d0127863a0e4c3092b94a514763920a99fa
6942a651d58e698c79dceb27a4b2cd912b75b8bdbb91ecebcb58d4f6b8efae4ee93a73ea182ea26605b1
e10170da973874ab81f099b3b2dd3bf66b574de9044422d39498a19ff2042fdf057b81a4d89dd5bc9939
b28b
```

3.3.3. 暗号化データの送信

暗号化したデータを Smart-in サーバに送信します。

以下の内容で HTTP 送信してください。

URL	https:// {接続サーバドメイン名} /request.cgi ※接続サーバドメイン名は、契約時に提供された Key ファイルの中身を参照ください。	
送信タイプ	POST	
パラメータ	company	企業コード 4 桁
	data	暗号化されたデータ

3.3.4. 送信結果の復号化と分析

前項で送信した結果は暗号化されたテキストで返されます。

このテキストを復号化すると、依頼結果が JSON 形式で取得できます。

レスポンス例：

```
099c51fd3dfaaf86dd596e31dfea8c2a61a322a743d6bcdf1433e4647a735c92ebad775531b5a1404e7
684d167db341621def73e29e0bbf25840d5977573297f388828e0f214d29b44b24428ee897fb6
```

復号化例

※下記は、Linux コマンドライン例です。

```
./sin_crypt -d 1234567890abcdef1234567890abcdef
099c51fd3dfaaf86dd596e31dfea8c2a61a322a743d
6bcdf1433e4647a735c92ebad775531b5a1404e7684d167db341621def73e29e0bbf25840d597757329
7f388828e0f214d29b44b24428ee897fb6
```

↓

復号化後

レスポンスには改行が含まれません。

```
{ "result": "0", "token": "717b63612d9b423bbef554a544f65a5b", "detail": "" }
```

この JSON 内の result の値が 0 であれば正常に依頼を受け付けたことになります。
指定した電話番号宛に電話が発信されることを確認ください。

0 以外の値が返ってきた場合は 依頼を受け付けられなかったことを意味します。
detail に理由のコードが記載されています。

3.3.5. 認証結果通知の受信（受け口を用意する）

コールバック認証を行う場合、Smart-in からの認証結果を受け取るための仕組みが
導入企業様システム側に必要となります。

Smart-in が認証完了した場合、もしくはタイムアウトした場合などに導入企業様システムに対し、
結果データを HTTP 送信します。

URL	認証依頼に設定した response_url	
送信タイプ	POST	
パラメータ	data	暗号化されたデータ

データを復号化すると、JSON 形式で結果が取得できます。

復号化前のテキスト：

```
bdf887be7d7fbcfee78d07c054540c2f7816cbfb84a3b150255dd3d9588223fb7dc230d58a401e03a0d5
c0b0529f50bc0a8f807f1b29e21e76fb6f07686994fc876a8f4fdca48e31575e63ccd5a1362e
```

復号化コマンド例（汎用ライブラリ使用時）

```
./sin_crypt -d 1234567890abcdef1234567890abcdef bdf887be7d7fbcfee78d07c054540c2f7816
cbfb84a3b150255dd3d9588223fb7dc230d58a401e03a0d5c0b0529f50bc0a8f807f1b29e21e76fb6f07
686994fc876a8f4fdca48e31575e63ccd5a1362e
```

↓

復号化後

レスポンスには改行が含まれません。

```
{ "token": "717b63612d9b423bbef554a544f65a5b", "code": "C51", "detail": "00" }
```

ここまで確認できれば、ひととおりの疎通が確認できたことになります。

3.3.6. エラー応答

依頼データの JSON 形式に不備がある場合、レスポンスの JSON データの暗号化ができません。
その場合は、以下のエラーメッセージを返却します。
上記エラーメッセージを返却時は、HTTP ヘッダーのステータスコードを **450** で返却します。

原因 1：POST のパラメータが間違っている、もしくは POST パラメータがない場合
間違った例 1、company ではない　com=XXXX&data=YYYYYY
間違った例 2:data ではない　company=XXXX&datas=YYYYY

返却例 1：

```
ParseRequest Error
```

原因 2：POST パラメータの値が間違っている場合
company=XXXX の XXXX が間違っている場合。
data=YYYYYY の YYYYY が間違っている場合。
YYYY 部分を復号化したデータが JSON 形式ではない場合。

```
DecryptRequest Error
```

4. 実装例

導入企業様システムにおける接続機能の実装例として、サンプルコードを以下に示します。
サンプルコードは PHP 言語で記載しています。

4.1. Smart-in 接続サンプル利用の注意点

・CentOS6.x(64bit) ※6.3 以上
・PHP5.3.3
・MySQL5.6
※OS の種別が異なる場合や、32bitOS の場合は動作しません。
その際はお客様環境に合わせた sin_crypt に差し換えてください。

Smart-in 接続サンプルファイル一覧

No.	ディレクトリ名/ファイル名	概要
1	calling.gif	接続確認時のダイアログ画像
2	index.html	画面表示用 HTML
3	readme.txt	Smart-in 接続サンプル簡易説明
4	sin_crypt	暗号化モジュール（※要権限設定 755）
5	smartin.php	サンプル PHP（※要設定）
6	smartin_response.php	サンプル PHP（※要設定）
7	textdb.db	ファイル使用時のデータファイル（※要権限設定 666）

4.2. 設置手順

4.2.1.1.下記のファイル内の設定情報を変更
下記を参照し、ファイルの編集をします。

smartin.php 変更箇所

No.	ディレクトリ名/ファイル名	概要
1	{Smart-in キー32 桁}	「3.1　接続リソースの入手」 Smart-in 接続企業情報.pdf を参照
2	{Smart-in 企業コード 4 桁}	「3.1　接続リソースの入手」 Smart-in 接続企業情報.pdf を参照
3	{DB 名}	データベース名を設定
4	{DB ユーザ名}	データベース接続ユーザ名
5	{DB パスワード}	データベース接続パスワード
6	{設置 DIR}	php ファイル格納先ディレクトリ

smartin_response.php 変更箇所

No.	ディレクトリ名/ファイル名	概要
1	{Smart-in キー32 桁}	「3.1 接続リソースの入手」 Smart-in 接続企業情報.pdf を参照
3	{DB 名}	データベース名を設定
4	{DB ユーザ名}	データベース接続ユーザ名
5	{DB パスワード}	データベース接続パスワード

4.2.1.2.ファイル格納

FTP もしくは SCP にてファイル一式をサーバに配置します。

4.2.1.3.パーミッションの設定

以下のファイルのパーミッションを設定します。

コマンドライン例

```
$ chmod 755 sin_crypt
$ chmod 666 textdb.db
```

4.2.1.4.データベースの作成

データベースをご利用の場合、token 管理用のテーブルを作成します。

※MySQL の場合

例

```
CREATE TABLE `tokens` (
     `token` varchar(32) NOT NULL,
     `status` int(11) NOT NULL,
      `updated` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
CURRENT_TIMESTAMP
    ) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

4.2.1.5.動作確認

ブラウザを開いて、設置した index.html にアクセスします。

電話番号を入力するフィールドがありますので、入力し、「Smart-in 認証」ボタンを押すと

電話がかかってきます。

4.2.2. サンプルコード

PHP と汎用ライブラリ（sin_crypt）で実装した場合のサンプルコードは以下となります。

ファイル名：index.html

```
<html>
<head>
<title>Smart-in 認証サンプル</title>
</head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.8.0/jquery.min.js"></script>
<script>
$(document).ready(function(){
    $("#send").click(function(){
        $("#done").hide();
        $("#fail").hide();
        callSmartin();
    });
});

/* Ajax で Smart-in サーバへリクエストを送信 */
function callSmartin(){
    data_tel_no = $("#tel_no").val();
    req_url = "smartin.php?mode=request";
    $.ajax({
        url: req_url,
        type:'POST',
        data: { tel_no: data_tel_no },
    }).done(function( token_str ) {
        $("#progress").show();

        // Smart-in サーバからのコールバックを確認開始
        checkSmartin( token_str );
    }).fail(function( data ) {
        $("#fail").show();
        $("#fail").html("ネットワークエラー (0)");
        $("#progress").hide();
    });
}

/* Ajax で Smart-in サーバからのコールバックを確認
   5 秒間隔で結果が得られるまで繰り返し
*/
function checkSmartin( token_str ){
    req_url = "smartin.php?mode=check";
    timer = setInterval(function(){
        $.ajax({
            url: req_url,
            type:'POST',
            data: { token: token_str },
        }).done(function( result ) {

            // 認証成功
            if(result == "true"){
                $("#done").show();
                $("#progress").hide();
                clearInterval(timer);

            // 認証中
            }else if(result == "waiting"){
```

次ページへ続く・・・

・・・前ページからの続き

```
            // 認証失敗
            }else{
                $("#fail").show();
                $("#fail").html("認証失敗 (" + result + ")");
                $("#progress").hide();
                clearInterval(timer);
            }
        }).fail(function( data ) {
            $("#fail").show();
            $("#fail").html("ネットワークエラー (1)");
            $("#progress").hide();
            clearInterval(timer);
        });
    }, 5000);       // 5 秒間隔で確認
}
</script>
<body>

<div id="container">

    認証電話番号<br/>
    <input type="text" name="tel_no" id="tel_no"><br/>
    <br/>
    <input type="button" id="send" value="Smart-in 認証"><br/>
    <br/>

    <div id="progress" style="display:none">
        <img id="calling" src="calling.gif"><br/>
        本人確認中
    </div>

    <div id="done" style="display:none">
        認証成功
    </div>

    <div id="fail" style="display:none">
    </div>

</div>

</body>

</html>
```

ファイル名：smartin.php

```
<?php
/*   Smart-in への接続要求、およびステータスチェックを行います。
*    以下に必要情報を記載してからご利用ください。
*    {Smart-in キー32 桁}
*    {Smart-in 企業コード 4 桁}
*    {DB 名}
*    {DB ユーザ名}
*    {DB パスワード}
*    {設置 DIR}
*/

// sin_crypt 暗号化キー（半角英数 32 桁）
define('KEY_CODE', '{Smart-in キー32 桁}');
// 企業コード
define('COMPANY_CODE', '{Smart-in 企業コード 4 桁}');
// 依頼区分(要求された電話番号に発信を行い、コールバック認証を行う。)
define('REQUEST_CODE', 'C51');
// Smart-in 依頼用サーバ
define('SMARTIN_SVR', 'api.smart-in.biz');   // こちらは試験用です。提供環境に応じ、適宜変更ください。
// sin_crypt 格納パス
define('SIN_CRYPT_PATH', './sin_crypt');
// 確認結果 POST 用 URL
define('RESPONSE_URL', 'https://' . $_SERVER['HTTP_HOST'] . '{設置 DIR}/smartin_response.php');// UR
L はご利用の環境に応じ、ディレクトリを追記ください。

// DB 設定
define('PDO_DSN', 'mysql:dbname={DB 名}; host=localhost');
define('PDO_USER', '{DB ユーザ名}');
define('PDO_PASS', '{DB パスワード}');

if ($_SERVER['REQUEST_METHOD'] == 'POST') {

     $mode = $_REQUEST['mode'];

     /*
          Smart-in へコールバックリクエストを行う
     */
     if ($mode == 'request') {

          // 接続先電話番号
          $tel_no = $_POST['tel_no'];

          // smart-in へのリクエストパラメータを生成
          $request = array(
               'company' => COMPANY_CODE,
               'code' => REQUEST_CODE,
               'telno' => $tel_no,
               'response_url' => RESPONSE_URL
          );
          $requestJson = json_encode($request);

          // json を専用 API で暗号化する
          $execShellCommand = SIN_CRYPT_PATH." -e ".KEY_CODE." '${requestJson}'";
          exec($execShellCommand, $output, $encodeResult);

          // 暗号化成功
          if($encodeResult == 0){

               // Smart-in への送信データに暗号化した json を代入
               $authJson = $output[0];
               $url = 'http://'.SMARTIN_SVR.'/request.cgi';
               $postdata = http_build_query(
                    array(
                         'company' => COMPANY_CODE,
                         'data' => $authJson
                    )
               );
```

次ページへ続く・・・

ファイル名：smartin.php

・・・前ページからの続き

```
            // http 設定
            $opts = array('http' =>
                array(
                    'method'  => 'POST',
                    'header'  => 'Content-type: application/x-www-form-urlencoded',
                    'content' => $postdata
                )
            );
            $context  = stream_context_create($opts);

            // Smart-in にリクエスト
            $response = file_get_contents($url, false, $context);

            // レスポンスをデコードし、画面に表示
            $execShellCommand = SIN_CRYPT_PATH." -d ".KEY_CODE." ${response}";
            $responseJson = exec($execShellCommand, $output, $decodeResult);
            if($decodeResult == 0){
                $responseDecodeStr = "";
                for ($i = 1; $i < count($output)-1; $i++) {
                    $responseDecodeStr .= $output[$i];
                }

                // デコードすると token と result が出てくる
                // result 0:正常 9:異常
                $responseArray = json_decode($responseDecodeStr, true);

                echo $responseArray['token'];

            // デコード失敗
            }else{
                echo 'デコード失敗';
            }

        // 暗号化失敗
        }else{
            echo '暗号化失敗';
        }

        exit;
    }

    /*
        Smart-in のコールバックから登録した情報を参照する
    */
    if ($mode == 'check'){
        $token = $_POST['token'];
        if (strlen($token) != 32)
            exit;

        // 結果を取得（DB 使用時）
        $dbh = new PDO(PDO_DSN, PDO_USER, PDO_PASS);
        $sql = "SELECT * FROM tokens WHERE token = '${token}'";
        $stmt = $dbh->query($sql);
        $array = $stmt->fetchAll(PDO::FETCH_ASSOC);
        if(count($array) > 0)
            $smartin_res = $array[0]['status'];
        $dbh = null;
```

次ページへ続く・・・

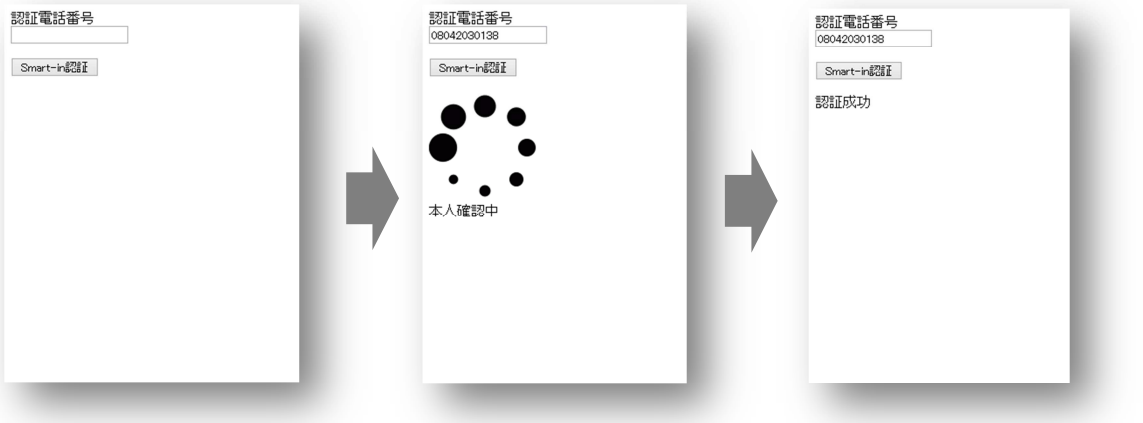
ファイル名：smartin.php

・・・前ページからの続き

```
        // 結果を取得（ファイル使用時）
        //$db = parse_ini_file('textdb.db');
        //$smartin_res = $db[$token];


        switch ($smartin_res) {
            case 100:
                $result = 'true';
                break;
            case 101:
                $result = '本人話中';
                break;
            case 102:
                $result = '着信時拒否';
                break;
            case 103:
                $result = 'タイムアウト';
                break;
            case 109:
                $result = '例外';
                break;
            default:
                $result = 'waiting';
                break;
        }

        echo $result;
        exit;
    }

}
```

ファイル名：smartin_response.php

```
<?php
/*   Smart-in から POST される認証結果情報を取得し、DB に登録します。
*    以下に必要情報を記載してからご利用ください。
*    {Smart-in キー32 桁}
*    {DB 名}
*    {DB ユーザ名}
*    {DB パスワード}
*/

// sin_crypt 暗号化キー（半角英数 32 桁）
define('KEY_CODE', '{Smart-in キー32 桁}');
// sin_crypt 格納パス
define('SIN_CRYPT_PATH', './sin_crypt');

// DB 設定
define('PDO_DSN', 'mysql:dbname={DB 名}; host=localhost');
define('PDO_USER', '{DB ユーザ名}');
define('PDO_PASS', '{DB パスワード}');

if (!isset($_POST['data'])) {
    exit;
}

// 暗号化解除
$encryptedJson = $_POST['data'];
$execShellCommand = SIN_CRYPT_PATH." -d ".KEY_CODE." '${encryptedJson}'";
exec($execShellCommand, $output, $decodeResult);

// JSON のデコード
if($decodeResult == 0){

    // デコードすると token と detail(結果)が出てくる
    // detail 00:正常, 01:ビジー, 02:着信時拒否, 03:タイムアウト
    $joinvar = join('', $output);
    $responseArray = json_decode($joinvar, true);
    $token = $responseArray['token'];
    // 正常（規定時間内に接続）
    if($responseArray['detail'] === '00'){
        $status = 100;
    // ビジー(本人話中)
    }elseif($responseArray['detail'] === '01'){
        $status = 101;
    // 着信時拒否（スマホのみ）
    }elseif($responseArray['detail'] === '02'){
        $status = 102;
    // タイムアウト（応答無し、接続不可）
    }elseif($responseArray['detail'] === '03'){
        $status = 103;
    // 例外
    }else{
        $status = 109;
    }

    // 結果を保存（DB 使用時）
    // ※注意：定期的に古いレコードを削除する必要があります
    $dbh = new PDO(PDO_DSN, PDO_USER, PDO_PASS);
    $stmt = $dbh->prepare("INSERT INTO tokens (token, status) VALUES (?, ?)");
    $stmt->execute(array($token, $status));
    $dbh = null;

    // 結果を保存（ファイル使用時）
    //$fp = fopen('textdb.db', 'a');
    //fwrite($fp, "${token} = ¥"${status}¥"");
    //fclose($fp);

}
```



サンプルを動作させた際の画面の流れ

5. 別紙

5.1.1. C#用

5.1.1.1.ソースガイド

5.1.1.1.1.Smart-in 接続サンプル利用の注意点

動作環境

・Windows7、WindowsServer2008R2
※動作確認は Windows7(64bit)で実施しました。
・IIS7.5
・Microsoft SQL Server 2012
・.NETFramework 4.5

Smart-in 接続サンプルファイル一覧

No.	ディレクトリ名/ファイル名	概要
1	DB 作成の SQL.txt	データベースのテーブルの SQL 文
2	calling.gif	接続確認時のダイアログ画像
3	sin_crypt.exe	暗号化モジュール
4	smart-in-website.sln	プロジェクトファイル
5	index.aspx	メインページの ASPX ファイル
6	index.aspx.cs	メインページの CS ファイル
7	smartin_response.aspx	サンプル ASPX
8	smartin_response.aspx.cs	サンプル CS（※要設定）
9	Web.config	設定ファイル
10	App_Code/smartin.cs	サンプル CS（※要設定）
11	App_Code/Utility.cs	サンプル CS

5.1.1.1.2.設置手順

下記のファイルの編集をします。

Web.config 変更箇所

No.	ディレクトリ名/ファイル名	概要
1	connectionString	・Data Source=******* ・User ID=********** ・Password=*******

smartin.cs 変更箇所

No.	ディレクトリ名/ファイル名	概要
1	{Smart-in キー32 桁}	「3.1　接続リソースの入手」 Smart-in 接続企業情報.pdf を参照
2	{Smart-in 企業コード 4 桁}	「3.1　接続リソースの入手」 Smart-in 接続企業情報.pdf を参照
3	{設置 DIR}	ファイル格納先ディレクトリ

smartin_response.aspx.cs 変更箇所

No.	ディレクトリ名/ファイル名	概要
1	{Smart-in キー32 桁}	「3.1　接続リソースの入手」 Smart-in 接続企業情報.pdf を参照
2	{設置 DIR}	ファイル格納先ディレクトリ

5.1.1.1.3.Web サイト構築

Visual Studio でビルドして Web サイトをサーバに発行します。

5.1.1.1.4.データベースの作成

DB をご利用の場合、token 管理用のテーブルを作成します。

例、Microsoft SQL Server の場合

```
CREATE TABLE [dbo].[tokens](
[token] [varchar](60) NOT NULL,
[status] [bigint] NOT NULL,
[updated] datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
```

5.1.1.1.5.動作確認

ブラウザを開いて、設置した index.aspx にアクセスします。

電話番号を入力するフィールドがありますので、入力し、「Smart-in 認証」ボタンを押すと

電話がかかってきます。

5.2. 用語について

No.	用語	説明
1	テストサーバ	導入企業側サイトを構築する際に、接続試験を実施する為に使用します。
2	本番サーバ	本番運用に使用します。