クレジットカード決済
トークン方式
3Dセキュア2.0
（自動課金情報変更,停止）

目次

■概要
:自動課金情報変更,停止について ・・・・・・・・・・P.3
:対応ブラウザ・・・・・・・・・・・・・・・・・・・・P.4

■自動課金情報変更
:自動課金情報変更処理(カード情報変更あり) 仕様 ・P.6
:自動課金情報変更処理(カード情報変更なし) 仕様 ・P.7

■自動課金停止
:自動課金停止処理 仕様 ・・・・・・・・・・・・・・・P.9
:自動課金停止処理後のキックバック 仕様 ・・・・・・P.10

■トークン作成、3Dセキュア2.0認証仕様
:ポップアップ方式 実装方法(サンプル)・・・・・・P.12
:カスタマイズ方式 実装方法(サンプル)・・・・・・P.14
:トークン作成(ポップアップ方式) 仕様 ・・・・・・P.16
:トークン作成(カスタマイズ方式) 仕様 ・・・・・・P.17
:3Dセキュア2.0認証 仕様 ・・・・・・・・・・・・・P.18

自動課金情報変更,停止について

自動課金情報変更,停止をAPIで行う場合の仕様書です。

■自動課金変更
自動課金に紐づく情報の変更を行うことができます。

カード情報を変更する際には トークン作成 と3Dセキュア2.0認証 が必要です。
※カード情報以外を変更する場合は不要です

トークン作成と3Dセキュア2.0認証の画面フロー、処理フローについては
以下の仕様書をご覧ください。

「クレジットカード決済 トークン方式 3Dセキュア2.0（概要、処理フロー、画面フロー）」

■自動課金停止
自動課金を停止することができます。

対応ブラウザ

対応ブラウザは以下です。

・Microsoft Edge 最新版
・Google Chrome 最新版
・Firefox　最新版
・Safari　最新版

※JavaScriptの利用できない端末（一部フィーチャーフォン等）では、トークン決済を行う事が出来ません。

自動課金情報変更

自動課金情報変更処理（カード情報変更あり）仕様

自動課金情報変更処理（カード情報変更あり）の仕様は以下の通りです。

■接続先
https://credit.j-payment.co.jp/gateway/accgate_token.aspx

※接続元IP認証を行っておりますため、加盟店様のサーバ（固定されたIPアドレス）からリクエストしてください。
エンドユーザー様のブラウザからリクエストした場合ER003（送信元IPエラー）が発生します。

■リクエスト方式
GET もしくは POST

※カード情報の更新の際には トークン作成 と3Dセキュア2.0認証 が必須です。
後述の「トークン作成、3Dセキュア2.0認証仕様」をご覧ください。

項目	フィールド	詳細	必須	形式
店舗ID	aid	契約時に発行される店舗のID	○	半角数字（6）
自動課金番号	acid	変更対象の自動課金番号	○	半角数字
変更タイプ	cmd	「1」	○	半角数字（1）
トークン情報 （カード情報）	tkn	トークン ※カード情報を変更する場合は トークン作成 と3Dセキュア2.0認証 が必要です	○	半角英数記号
メールアドレス	em	メールアドレス	※1	半角英数（254）
電話番号	pn	電話番号	※1	半角数字（15）
商品金額	am	商品金額		半角数字
税金額	tx	税金額		半角数字
送料	sf	送料		半角数字
次回課金日	nad	変更後の次回課金日（YYYYMMDD形式）		半角数字 （YYYYMMDD）
課金周期	actp	変更後の課金周期 2：毎週課金 3：隔週課金 4：毎月課金 5：隔月課金 6：3ヶ月課金 7：6ヶ月課金 8：1年課金		半角数字（6）

＜※1＞
自動課金情報にメールアドレス、電話番号いずれも登録されていない場合
もしくは登録されているメールアドレス、電話番号の形式にエラーがある場合は
いずれかの指定が必須となります。

自動課金情報変更処理（カード情報変更なし）仕様

自動課金情報変更処理（カード情報変更なし）の仕様は以下の通りです。

■接続先
https://credit.j-payment.co.jp/gateway/accgate.aspx

※接続元IP認証を行っておりますため、加盟店様のサーバ（固定されたIPアドレス）からリクエストしてください。
エンドユーザー様のブラウザからリクエストした場合ER003（送信元IPエラー）が発生します。

■リクエスト方式
GET もしくは POST

項目	フィールド	詳細	必須	形式
店舗ID	aid	契約時に発行される店舗のID	○	半角数字（6）
自動課金番号	acid	変更対象の自動課金番号	○	半角数字
変更タイプ	cmd	「1」	○	半角数字（1）
メールアドレス	em	メールアドレス	※1	半角英数（254）
電話番号	pn	電話番号	※1	半角数字（15）
商品金額	am	商品金額		半角数字
税金額	tx	税金額		半角数字
送料	sf	送料		半角数字
次回課金日	nad	変更後の次回課金日（YYYYMMDD形式）		半角数字 （YYYYMMDD）
課金周期	actp	変更後の課金周期 2：毎週課金 3：隔週課金 4：毎月課金 5：隔月課金 6：3ヶ月課金 7：6ヶ月課金 8：1年課金		半角数字（6）

＜※1＞
自動課金情報にメールアドレス、電話番号いずれも登録されていない場合
もしくは登録されているメールアドレス、電話番号の形式にエラーがある場合は
いずれかの指定が必須となります。

自動課金停止

自動課金停止処理 仕様

自動課金停止処理の仕様は以下の通りです。

■接続先
https://credit.j-payment.co.jp/gateway/acsgate.aspx

※接続元IP認証を行っておりますため、加盟店様のサーバ（固定されたIPアドレス）からリクエストしてください。
エンドユーザー様のブラウザからリクエストした場合ER003（送信元IPエラー）が発生します。

■リクエスト方式
GET もしくは POST

項目	フィールド	詳細	必須	形式
店舗ID	aid	契約時に発行される店舗のID	○	半角数字（6）
自動課金番号	acid	停止対象の自動課金番号	○	半角数字
処理タイプ	cmd	「1」	○	半角数字（1）

自動課金停止処理後のキックバック 仕様

・形式
URLパラメータ

・通知先
管理画面の「決済結果通知設定」の「決済結果通知URL」で設定したURL
※通知の成功判定を行っているため、通知先ではContentLengthが0以上が出力されるよう実装してください。

・通知タイミング
対象の自動課金の「次回課金日」に通知されます。

※リンク方式での自動課金停止した場合の通知タイミングは 「即時」・「次回課金日」を選択できますが
API（このページで案内している方式）では選択できず 「次回課金日」固定ですのでご注意ください。

・サンプル
決済結果通知URL?gid=10000001&rst=4&cod=test&acid=10000001&ec=

項目	フィールド	詳細	形式
初回決済番号	gid	停止対象の自動課金の初回決済時の決済番号	半角数字
結果返却	rst	「4」	半角数字
店舗オーダー番号	cod	初回決済時に指定したcod	50バイト以内
自動課金番号	acid	停止対象の自動課金番号	50バイト以内
エラーコード	ec	エラーコード	半角英数

トークン作成、3Dセキュア2.0認証仕様

ポップアップ方式　実装方法（サンプル）

HTMLの実装

①jQuery, CPToken.js, EMV3DSAdapter.jsを読み込みます。

```
<head>
<meta charset="utf-8" />
<script type="text/javascript"
src="https://credit.j-payment.co.jp/gateway/js/jquery.js" ></script> <!--※1-->
<script type="text/javascript"
src="https://credit.j-payment.co.jp/gateway/js/CPToken.js" ></script><!--※2-->
<script type="text/javascript"
src="https://credit.j-payment.co.jp/gateway/js/EMV3DSAdapter.js" ></script><!--※3-->
</head>
```

※1 加盟店様側でjQueryの読み込み部分を実装されている場合は不要です。
※2 トークン決済用
※3 3Dセキュア2.0用

②トークン用Hidden要素、トークンポップアップ用要素、3Dセキュア用要素を追加します。

```
<form id="mainform" method="POST" action="https://credit.j-payment.co.jp/gateway/accgate_token.aspx"   >
    <input type="hidden" value="" id="aid" name="aid">
    <input type="hidden" value="" id="acid" name="acid">
    <input type="hidden" value="" id="cmd" name="cmd">
    <input type="text" value="" id="em" name="em">
    <input type="text" value="" id="pn" name="pn">
    <!--トークン作成処理後にid:tkn要素に値がセットされます-->
    <input type="hidden" value="" id="tkn" name="tkn" >
    <!- トークンポップアップ表示用 -->
    <div id="CARD_INPUT_FORM" ></div>
    <!-- 3Dセキュアポップアップ表示用 -->
    <div id="EMV3DS_INPUT_FORM" ></div>
    <input type="button" value="変更する" onclick="doPurchase()" />
</form>
```

次のページへ続く

ポップアップ方式　実装方法（サンプル）

Javascriptの実装

```
/**
 * １．クレジットカードトークンの作成
 * トークン作成用の関数を準備します。
 */
function doPurchase() {

    // トークン作成処理を呼び出します。
    CPToken.CardInfo({
        aid: '000000' // 店舗IDを設定します。
    }, execAuth);      // 3Dセキュア2.0認証用関数をコールバックにセットします。
}
/**
 * ２．3Dセキュア2.0の認証を実行
 * 3Dセキュア2認証用関数の準備をします。
 * 引数はresultCode, errMsgの2つを受け取れるようします。
 */
function execAuth(resultCode, errMsg) {
    if (resultCode != "Success") {
        // 戻り値がSuccess以外の場合はエラーメッセージを表示します
        window.alert(errMsg);
    } else {

        // 3Dセキュア2.0認証処理を呼び出します。
        ThreeDSAdapter.authenticate({
            aid: '000000',         // 店舗IDを設定します。
            tkn: $("#tkn").val(), // トークン作成後にtkn要素に値が入力されます
            acid: '000000',        // 自動課金番号を設定します。
            em: 'test@test.jp',    // 変更処理にリクエストする値と同じ値を設定します。(不一致の場合はエラー)
            pn: '000000',          // 変更処理にリクエストする値と同じ値を設定します。(不一致の場合はエラー)
        }, execPurchase);   // 決済実行用の関数をコールバックにセットします。
    }
}
/**
 * ３．決済処理の実行
 * 決済実行用の関数を準備します。
 * 引数はresultCode, errMsgの2つを受け取れるようにします。
 */
function execPurchase(resultCode, errMsg) {
    if (resultCode != "Success") {
        // 戻り値がSuccess以外の場合はエラーメッセージが返却されます。
        window.alert(errMsg);
    } else {
        // 決済リクエストを実行する処理の実装をします。
        $("#mainform").submit();
    }
}
```

カスタマイズ方式　実装方法（サンプル）

HTMLの実装

①jQuery, CPToken.js, EMV3DSAdapter.jsを読み込みます。

```
<head>
<meta charset="utf-8" />
<script type="text/javascript"
src="https://credit.j-payment.co.jp/gateway/js/jquery.js" ></script> <!--※1-->
<script type="text/javascript"
src="https://credit.j-payment.co.jp/gateway/js/CPToken.js" ></script><!--※2-->
<script type="text/javascript"
src="https://credit.j-payment.co.jp/gateway/js/EMV3DSAdapter.js" ></script><!--※3-->
</head>
```

※1 加盟店様側でjQueryの読み込み部分を実装されている場合は不要です。
※2 トークン決済用
※3 3Dセキュア2.0用

②トークン用Hidden要素、トークンポップアップ用要素、3Dセキュア用要素を追加します。

```
<form id="mainform" method="POST" action="https://credit.j-payment.co.jp/gateway/accgate_token.aspx"   >
    <input type="hidden" value="" id="aid" name="aid">
    <input type="hidden" value="" id="acid" name="acid">
    <input type="hidden" value="" id="cmd" name="cmd">
    <input type="text" value="" id="em" name="em">
    <input type="text" value="" id="pn" name="pn">
    <!-- カード番号 -->
    <input type="text" value="" id="cn name="cn">
    <!-- カード有効期限 -->
    <input type="text" value="" id="ed_year" name="ed_year">
    <input type="text" value="" id="ed_month" name="ed_month">
    <!-- カード名義 -->
    <input type="text" value="" id="fn" name="fn">
    <input type="text" value="" id="ln" name="ln">
    <!--トークン作成処理後にid:tkn要素に値がセットされます-->
    <input type="hidden" value="" id="tkn" name="tkn" >
    <!-- 3Dセキュアポップアップ表示用-->
    <div id="EMV3DS_INPUT_FORM" ></div>
    <input type="button" value="購入する" onclick="doPurchase()">
</form>
```

次のページへ続く

カスタマイズ方式　実装方法（サンプル）

Javascriptの実装

```
/**
 * １．クレジットカードトークンの作成
 * トークン作成用の関数を準備します。
 */
function doPurchase() {
    // トークン作成処理を呼び出します。
    CPToken.TokenCreate ({
        aid: '000000', // 店舗IDを設定します。
        cn: $("#cn").val(),
        ed: $("#ed_year").val() + $("#ed_month").val(),
        fn: $("#fn").val(),
        ln: $("#ln").val()
    }, execAuth); // 3Dセキュア2.0認証用関数をコールバックにセットします。
}

/**
 * ２．3Dセキュア2.0の認証を実行
 * 3Dセキュア2認証用関数の準備をします。
 * 引数はresultCode, errMsgの2つを受け取れるようします。
 */
function execAuth(resultCode, errMsg) {
    if (resultCode != "Success") {
        // 戻り値がSuccess以外の場合はエラーメッセージを表示します
        window.alert(errMsg);
    } else {

        // 3Dセキュア2.0認証処理を呼び出します。
        ThreeDSAdapter.authenticate({
            aid: '000000',          // 店舗IDを設定します。
            tkn: $("#tkn").val(), // トークン作成後にtkn要素に値が入力されます
            acid: '000000',         // 自動課金番号を設定します。
            em: 'test@test.jp',     // 変更処理にリクエストする値と同じ値を設定します。(不一致の場合はエラー)
            pn: '000000',           // 変更処理にリクエストする値と同じ値を設定します。(不一致の場合はエラー)
        }, execPurchase);   // 決済実行用の関数をコールバックにセットします。
    }
}

/**
 * ３．決済処理の実行
 * 決済実行用の関数を準備します。
 * 引数はresultCode, errMsgの2つを受け取れるようにします。
 */
function execPurchase(resultCode, errMsg) {
    if (resultCode != "Success") {
        // 戻り値がSuccess以外の場合はエラーメッセージが返却されます。
        window.alert(errMsg);
    } else {
        // カード情報を消去
        $("#cn").val("");
        $("#ed_year").val("");
        $("#ed_month").val("");
        $("#fn").val("");
        $("#ln").val("");

        // 決済リクエストを実行する処理の実装をします。
        $("#mainform").submit();
    }
}
```

トークン作成（ポップアップ方式）仕様

トークン作成（ポップアップ方式）の仕様は以下の通りです　。

■接続先
https://credit.j-payment.co.jp/gateway/js/CPToken.js

■ファイル
CPToken.js

■関数
CPToken.CardInfo(tokenRequest,callback)

・CardInfo引数仕様

引数名	説明	備考
tokenRequest	トークン作成に必要なパラメータ	Json形式
callback	実行後にコールバックされる関数 ※ThreeDSAdapter.authenticate関数を指定してください。	

・tokenRequest仕様

項目	フィールド	詳細	必須	形式
店舗ID	aid	契約時に発行される店舗のID	○	半角数字（6）

■レスポンス
なし（処理完了時にHTMLのid=tkn要素のvalueにトークンの文字列が設定されます）

■その他
弊社のJavaScript内で使用しているため、実装の際に「rpemvtds」クラスを使用しないでください。

トークン作成（カスタマイズ方式）仕様

トークン作成（カスタマイズ方式）の仕様は以下の通りです。

■接続先
https://credit.j-payment.co.jp/gateway/js/CPToken.js

■ファイル
CPToken.js

■関数
CPToken.TokenCreate(tokenRequest,callback)

・TokenCreate引数仕様

引数名	説明	備考
tokenRequest	トークン作成に必要なパラメータ	Json形式
callback	実行後にコールバックされる関数 ※ThreeDSAdapter.authenticate関数を指定してください。	

・tokenRequest仕様

項目	フィールド	詳細	必須	形式
店舗ID	aid	契約時に発行される店舗のID	○	半角数字（6）
カード番号	cn	カード番号	○	半角数字
カード有効期限	ed	カードの有効期限（YYMM形式）	○	半角数字
カード名義（名）	fn	カードの名義（名）	○	半角英数記号（43） ※1
カード名義（姓）	ln	カードの名義（姓）	○	半角英数記号（43） ※1
セキュリティコード	cvv	VISA・MASTER・JCB・DINERSは カード裏面の3桁の番号 AMEXのみカード表面の4桁の番号	※2	半角数字

＜※1＞
・記号はピリオド、ハイフン（いずれも半角）
・カード名義（名）と（姓）の合計が44文字以内
＜※2＞
セキュリティーコードを利用の場合は必須（ご利用には契約が必要です）

■レスポンス
なし（処理完了時にHTMLのid=tkn要素のvalueにトークンの文字列が設定されます）

■その他
弊社のJavaScript内で使用しているため、実装の際に「rpemvtds」クラスを使用しないでください。

3Dセキュア2.0認証 仕様

3Dセキュア2.0認証の仕様は以下の通りです 。

■接続先
https://credit.j-payment.co.jp/gateway/js/EMV3DSAdapter.js

■ファイル
EMV3DSAdapter.js

■関数
ThreeDSAdapter.authenticate(authenticateRequest,callback)

・authenticate引数仕様

引数名	説明	備考
authenticateRequest	3DS認証に必要なパラメータ	Json形式
callback	実行後にコールバックされる関数 ※決済処理を行う関数を指定してください。	

・authenticateRequest仕様

項目	フィールド	詳細	必須	形式
店舗ID	aid	契約時に発行される店舗のID	○	半角数字(6)
トークン情報	tkn	トークン(トークン作成処理で取得)	○	半角英数記号
自動課金番号	acid	変更対象の自動課金番号	○	半角数字
メールアドレス	em	メールアドレス 変更処理と同じ値を設定してください。	※1	半角英数(254)
電話番号	pn	電話番号 変更処理と同じ値を設定してください。	※1	半角数字(15)

＜※1＞
メールアドレス、電話番号を変更する場合は必須です。

・callback仕様

項目	説明	備考
結果コード	成功時:Success 失敗時:エラーコード	“Success”はAPI呼び出しの成功を意味し、3DS認証の成功を表すものではありません。 3DS認証の成否は、後続で実施する決済処理のレスポンスで出力されます。
メッセージ	結果コードの詳細メッセージ	

■レスポンス
なし

■その他
弊社のJavaScript内で使用しているため、実装の際に「rpemvtds」クラスを使用しないでください。