クレジットカード トークン方式
3Dセキュア2.0(通常決済)
接続仕様書

ROBOT PAYMENT

目次

通常決済について

■通常決済とは
通常（都度課金）決済は、単発の決済を行います。

※定期的に課金したい場合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（自動課金決済）」

■トークン作成、3Dセキュア2.0認証
決済にはトークン作成と3Dセキュア2.0認証 が必要です。

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）」

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。

通常決済

決済処理（商品登録なし）仕様

決済処理（商品登録なし）の仕様は以下の通りです。

「商品登録なし」の場合は【商品金額、税金額、送料】の合計値をリクエストします
決済金額の指定を動的に行える為、ショッピングカートとの連携等を行う際にご利用ください。

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

※決済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）
ジョブタイプ	jb	決済処理方法 「AUTH」（仮売上） ※1 「CAPTURE」（仮実同時売上）	○	半角英字
決済結果返信方法	rt	「0」キックバック 「1」レスポンス	○	半角数字（1）
店舗オーダー番号	cod	オーダーを識別する為に任意で設定するパラメータ		50バイト以内
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
メールアドレス	em	メールアドレス	※4	半角英数（254）
電話番号	pn	電話番号	※4	半角数字（15）
商品金額	am	商品金額 ※2	○	半角数字
税金額	tx	税金額 ※2	○	半角数字
送料	sf	送料 ※2	○	半角数字
その他データ		key=value形式で任意で設定するパラメータ ※3		1024バイト以内

＜※1＞
AUTH（仮売上）を指定した場合、決済後90日以内に実売上を行ってください。
実売上を行って頂かないとお客様への請求が確定いたしません。
CAPTURE（仮実同時売上）を指定した場合、決済時に自動的に実売上が行われます。

＜※2＞
【決済金額＋税金額＋送料】の値がカード会社に送信されます。
商品金額に送料、税額を含む場合、tx sf は0を指定してください。

＜※3＞
「改行コード」「スペース」を送信される場合、必ずエンコード処理を行ってください

＜※4＞
メールアドレス、電話番号のどちらかが必須となります。

■レスポンス
後述の「決済結果通知のパラメータ 仕様」を参照

決済処理（商品登録あり）仕様

決済処理（商品登録あり）の仕様は以下の通りです。

「商品登録あり」の場合は、事前に管理画面上にて商品情報の登録を行い、登録した商品の商品コードをリクエストします。
商品の金額が固定の場合にご利用ください。

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

※決済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）
決済結果返信方法	rt	「0」キックバック 「1」レスポンス	○	半角数字（1）
店舗オーダー番号	cod	オーダーを識別する為に任意で設定するパラメータ		50バイト以内
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
メールアドレス	em	メールアドレス	※2	半角英数（254）
電話番号	pn	電話番号	※2	半角数字（15）
商品コード	iid	商品コード	○	半角英数（50）
その他データ		key=value形式で任意で設定するパラメータ　※1		1024バイト以内

＜※1＞
「改行コード」「スペース」を送信される場合、必ずエンコード処理を行ってください

＜※2＞
メールアドレス、電話番号のどちらかが必須となります。

■レスポンス
後述の「決済結果通知のパラメータ 仕様」を参照

決済処理（有効性チェック）仕様

決済処理（有効性チェック）の仕様は以下の通りです。

「有効性チェック」の場合は、クレジットカードの有効性の確認を行うため
0円での決済処理を行います。

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

※決済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	○	半角英数（6）
ジョブタイプ	jb	「CHECK」	○	半角英字
決済結果返信方法	rt	「0」キックバック 「1」レスポンス	○	半角数字（1）
店舗オーダー番号	cod	オーダーを識別する為に任意で設定するパラメータ		50バイト以内
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
メールアドレス	em	メールアドレス	※2	半角英数（254）
電話番号	pn	電話番号	※2	半角数字（15）
その他データ		key=value形式で任意で設定するパラメータ　※1		1024バイト以内

＜※1＞
「改行コード」「スペース」を送信される場合、必ずエンコード処理を行ってください

＜※2＞
メールアドレス、電話番号のどちらかが必須となります。

■レスポンス
後述の「決済結果通知のパラメータ 仕様」を参照

決済結果通知のパラメータ 仕様

決済結果通知の仕様は以下の通りです。

■キックバック(rt=0)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。

・サンプル
決済結果通知URL?gid=10000001&rst=1&ap=0001112&ec=&god=10000002&cod=test &am=1000&tx=80&sf=500&ta=1580

■レスポンス(rt=1)仕様

・形式
HTML出力 カンマ区切り

・サンプル
1095729,1,TestMod,,205685,,1000,2,4,1006,,,
0,2,,ER900,,,0,0,0,0,,,0,

項目	フィールド	詳細	レスポンス出力順	形式
決済番号	gid	決済ごとに発行される番号	1	半角数字
決済結果	rst	1:決済成功 2:決済失敗	2	半角数字(1)
カード会社承認番号	ap	カード会社から発行される承認番号	3	半角英数(6～7)
エラーコード	ec	エラーコード	4	半角英数(12)
オーダーコード	god	決済ごとに発行されるコード	5	半角数字
店舗オーダー番号	cod	決済リクエスト時に指定したcod	6	50バイト以内
決済金額	am	決済リクエスト時に指定したam	7	半角数字
税金額	tx	決済リクエスト時に指定したtx	8	半角数字
送料	sf	決済リクエスト時に指定したsf	9	半角数字
合計金額	ta	am,tx,sfの合計値	10	半角数字
発行ID	id	弊社システムより発行されたID ※1	11	半角英数(50字以内)
発行パスワード	ps	弊社システムより発行されたパスワード ※1	12	半角英数(50字以内)
その他		決済リクエスト時に指定したその他データ	13	1024バイト以内

＜※1＞
コンテンツ(ID/PW)の商品登録した商品で、商品登録あり決済をした場合のみ返却されます。

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/gateway_token.aspx"   >
    <input type="hidden" value="000000" id="aid" name="aid">
    <input type="hidden" value="" id="rt" name="rt">
    <!-- 商品登録なしの場合：am, tx, sfが必要 -->
    <input type="hidden" value="1000" id="am" name="am">
    <input type="hidden" value="0" id="tx" name="tx">
    <input type="hidden" value="0" id="sf" name="sf">
    <!-- 商品登録ありの場合：iidが必要 -->
    <input type="hidden" value= "ItemCode001" id= "iid" name="iid">
    <input type="hidden" value="sample@sample.com" id="em" name="em">
    <input type="hidden" value="0300000000" 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 id="tkn" name="tkn" type="hidden" value="">
    <!-- 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="購入する" 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({
            tkn: $("#tkn").val(),     // トークン作成後にtkn要素に値が入力されます　。
            aid: '000000',            // 店舗IDを設定します。
            // 商品登録なしの場合：am, tx, sfが必要
            am: 1000,                 // 決済処理と同じ値を設定します。(不一致の場合はエラー)
            tx: 0,                    // 決済処理と同じ値を設定します。(不一致の場合はエラー)
            sf: 0,                    // 決済処理と同じ値を設定します。(不一致の場合はエラー)
            // 商品登録ありの場合：iidが必要
            iid: 'ItemCode001',       // 決済処理と同じ値を設定します。(不一致の場合はエラー)
            em: 'sample@sample.com', // 決済処理と同じ値を設定します。(不一致の場合はエラー)
            pn: '0300000000',         // 決済処理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/gateway_token.aspx"   >
    <input type="hidden" value="000000" id="aid" name="aid">
    <input type="hidden" value="" id="rt" name="rt">
    <!-- 商品登録なしの場合：am, tx, sfが必要 -->
    <input type="hidden" value="1000" id="am" name="am">
    <input type="hidden" value="0" id="tx" name="tx">
    <input type="hidden" value="0" id="sf" name="sf">
    <!-- 商品登録ありの場合：iidが必要 -->
    <input type="hidden" value= "ItemCode001" id="iid" name="iid">
    <input type="hidden" value="sample@sample.com" id="em" name="em">
    <input type="hidden" value="0300000000" id="pn" name="pn">
    <!-- カード番号 -->
    <input type="text" value="" name="cn" id="cn" />
    <!-- カード有効期限 -->
    <input type="text" value="" name="ed_year" id="ed_year" /> /
    <input type="text" value="" name="ed_month" id="ed_month" />
    <!-- カード名義 -->
    <input type="text" value="" name="fn" id="fn" />
    <input type="text" value="" name="ln" id="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 id="tkn" name="tkn" type="hidden" value="">
    <!-- 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({
            tkn: $("#tkn").val(),     // トークン作成後にtkn要素に値が入力されます　。
            aid: '000000',            // 店舗IDを設定します。
            // 商品登録なしの場合：am, tx, sfが必要
            am: 1000,                 // 決済処理と同じ値を設定します。(不一致の場合はエラー)
            tx: 0,                    // 決済処理と同じ値を設定します。(不一致の場合はエラー)
            sf: 0,                    // 決済処理と同じ値を設定します。(不一致の場合はエラー)
            //商品登録ありの場合：iidが必要
            iid: 'ItemCode001',       // 決済処理と同じ値を設定します。(不一致の場合はエラー)
            em: 'sample@sample.com', // 決済処理と同じ値を設定します。(不一致の場合はエラー)
            pn: '0300000000',         // 決済処理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（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
支払方法	md	未指定の場合は「一括払い」となります 「10」一括払い 「61」分割払い ※3 「80」リボ払い ※3		半角数字
分割回数	pt	「3,5,6,10,12,15,18,20,24」から選択	※4	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）
＜※3＞
分割払い・リボ払いは契約時のみ指定可
＜※4＞
md=61（分割払い）指定時は必須

■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（商品登録なし）仕様

3Dセキュア2.0認証（商品登録なし）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
商品金額	am	商品金額※1	○	半角数字
税金額	tx	税金額※1	○	半角数字
送料	sf	送料※1	○	半角数字
メールアドレス	em	メールアドレス※1	※2	半角英数（254）
電話番号	pn	電話番号※1	※2	半角数字（15）

＜※1＞
決済処理と同じ値を指定してください。
＜※2＞
メールアドレス、電話番号のどちらかが必須となります。

・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。

3Dセキュア2.0認証（商品登録あり）仕様

3Dセキュア2.0認証（商品登録あり）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
商品コード	iid	商品コード※1	○	半角英数（50）
メールアドレス	em	メールアドレス※1	※2	半角英数（254）
電話番号	pn	電話番号※1	※2	半角数字（15）

＜※1＞
決済処理と同じ値を指定してください。
＜※2＞
メールアドレス、電話番号のどちらかが必須となります。

・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。