**Paidy決済**

Version<22.1.1>



# 目次

内容

[目次 2](#_Toc88741556)

[1.【概要】 4](#_Toc88741557)

[2.【コンポーネントの概要】 5](#_Toc88741558)

[2-1. 機能概要 5](#_Toc88741559)

[2-2. 制限、制約 5](#_Toc88741560)

[2-3. ユースケース 6](#_Toc88741561)

[2-3-1. 通常決済 6](#_Toc88741562)

[2-3-2. 定期決済 9](#_Toc88741563)

[2-4．決済におけるプライバシー 13](#_Toc88741564)

[3.【実装ガイド】 14](#_Toc88741565)

[3-1．カートリッジアップロード（UX Studio) 14](#_Toc88741566)

[3-2．BusinessManagerセットアップ 14](#_Toc88741567)

[3-2-1. カートリッジの割当て 14](#_Toc88741568)

[3-2-2. メタデータのインポート 15](#_Toc88741569)

[3-3．加盟店様でのカートリッジ修正 20](#_Toc88741570)

[3-3-1. CheckoutJSの設置 20](#_Toc88741571)

[3-3-2. 購入確定ボタンの制御について 23](#_Toc88741572)

[3-3-3. リソースファイルの修正について 25](#_Toc88741573)

[3-4．加盟店様でのカートリッジ修正 26](#_Toc88741574)

[3-4-1. COBillingjsの修正について 26](#_Toc88741575)

[3-4-2. COPlaceOrder.jsの修正について 28](#_Toc88741576)

[3-4-3. COSummary.jsの修正について 30](#_Toc88741577)

[3-4-4. 加盟店様側のJSコントローラカートリッジのpackage.jsonの修正について 32](#_Toc88741578)

[3-4-5. int\_paidy\_controllersカートリッジのpackage.jsonの修正について 32](#_Toc88741579)

[3-5. アドバイス 32](#_Toc88741580)

[4.加盟店様でのカートリッジ導入テスト 34](#_Toc88741581)

[4-1.テストを開始する前に 34](#_Toc88741582)

[4-1-1.Paidy加盟店管理画面 34](#_Toc88741583)

[4-1-2.Commerce Cloud Business Manager 35](#_Toc88741584)

[4-2.PaidyCheckoutのテスト情報 35](#_Toc88741585)

[4-3.ブラウザ互換性 36](#_Toc88741586)

[4-4.テストケース 36](#_Toc88741587)

[4-4-1.Paidy 通常決済 36](#_Toc88741588)

[4-4-2.Paidy定期決済 37](#_Toc88741589)

[5. Webhook運用案 39](#_Toc88741590)

[5-1. Webhook運用について 39](#_Toc88741591)

[5-2. 不整合決済検知の流れ 39](#_Toc88741592)

[5-3. 不整合の対処方法 40](#_Toc88741593)

[5-3-1. 対処案１ 40](#_Toc88741594)

[5-3-2. 対処案２ 44](#_Toc88741595)

[5-4. Webhookの設定方法 44](#_Toc88741596)

[６.サービス停止時の対応について 45](#_Toc88741597)

[７.改訂履歴 46](#_Toc88741598)

# 1.【概要】

このカートリッジは、Commerce CloudストアがPaidy決済を使用できるようにします。

開発者はこの文書の指示に従って、カートリッジをインストールしてオンラインストアに設定する必要があります。

PaidyはテストAPIキーを使用してテスト決済を作成することが出来る為、本番環境に切り替える前に設定をテストすることができます。テストに必要な情報の登録については「4-1.テストを開始する前に」の項を参照ください。

Paidy決済を使用するためには2種類のカートリッジを導入する必要があります。

・「int\_paidy」はストア上でPaidy決済を可能にするCoreカートリッジです。

・「int\_paidy\_controllers」はJSControllerサイト上でPaidy決済を可能にするカートリッジです。

# 2.【コンポーネントの概要】

## 2-1. 機能概要

Paidyにおける処理のフローについては以下の図のようになります。

Paidyカートリッジでは、このフローの内、①〜⑥について対応をしています。

⑥ではPaidyとの認証の結果をサイト側へ保存します。

ダイアグラム

自動的に生成された説明

## 2-2. 制限、制約

・決済については「作成」処理のみを実装しており、「Capture」、「Refund」、「アップデート」、「Close」については加盟店様側で実装頂く必要があります。

・定期決済についてはコマースクラウドでは初回のみ「作成」の処理を実装しております。

　２回目以降の決済については加盟店様側で作成頂く必要がありますが、

　決済処理についてはPaidyカートリッジ内に用意された関数を実行することで実現可能となります。

・定期決済はログイン時のみ決済が可能な実装としております。

・Webhookの通知と受取確認はOMS側で対応頂き、決済の処理結果を正しく受け取れていないSFCC側の注文ステータスを出荷可能とするか、もしくはPaidy側の決済をキャンセル頂く必要があります。詳細は、[5. Webhook運用案](#_5.運用案)をご参照ください。

・このカートリッジは日本語サイトでのみ使用できます。（言語=日本語、通貨=日本円）

## 2-3. ユースケース

### 2-3-1. 通常決済

　サイト内で商品の購入を決定するボタンを押下したタイミングでPaidy Checkoutが立ち上がります。

　コンシューマーは、必要な情報を入力してPaidy決済を行います。

　決済が正常に完了した場合、商品購入完了画面へと遷移します。

・決済の流れ

注文手続きの支払方法の選択でPaidy通常決済を選択し、注文の確定へ進みます。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

Paidy CheckoutがPOPUPで表示されます。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, チャットまたはテキスト メッセージ

自動的に生成された説明

Paidy Checkoutへ必要な情報を入力し、Paidy決済を行います。

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明

決済が成功すると、注文が確定します。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, メール

自動的に生成された説明

コマースクラウドへ作成された注文データ



Paidyへ作成された決済データ

グラフィカル ユーザー インターフェイス

自動的に生成された説明

決済が失敗した場合は、Paidy Checkoutで決済エラーのメッセージが表示され、コマースクラウドへ

注文失敗のデータが作成されます。Paidyへ決済データは作成されません。

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明  
※なお、ポップアップ右上の×で中断した際も、注文失敗のデータが作成されます。Paidyへ決済データは作成されません。

コマースクラウドへ作成された注文データ（注文失敗）

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

### 2-3-2. 定期決済

　ログインしているユーザのみが利用できます。

サイト内で商品の購入を決定する画面へ遷移したタイミングでコンシューマーがトークンを保持しているかを確認します。

　購入を決定するボタンを押下すると、初回購入時はトークンを保持していないため、Paidy Checkoutが立ち上がります。コンシューマーは、必要な情報を入力してPaidyとの認証を行い、認証が正常に完了した場合、トークンが発行され、コンシューマーに紐付けられます。

トークンの保持が完了した、またはトークンを保持している場合は、トークン情報を元にPaidy決済が行われます。

・決済の流れ

注文手続きの支払方法の選択でPaidy定期決済を選択し、注文の確定へ進みます。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

注文確定画面が表示されるので、注文を確定します。



Paidy Checkoutが初回のみPOPUPで表示されます。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, チャットまたはテキスト メッセージ

自動的に生成された説明

Paidy Checkoutへ必要な情報を入力し、トークン発行を行います。

※途中でポップアップ右上の×で中断した場合は、ポップアップが非表示になりますが、再度トークン発行と決済処理が可能です。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

トークン発行の後、決済処理に移ります。決済が成功すると、注文完了画面へ遷移します。

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, メール

自動的に生成された説明

コマースクラウドへ作成された注文データ



Paidyへ作成された決済データ

グラフィカル ユーザー インターフェイス

自動的に生成された説明

決済が失敗した場合は、注文の確定時に決済エラーのメッセージが表示され、コマースクラウドへ

注文失敗のデータが作成されます。Paidyへ決済データは作成されません。



コマースクラウドへ作成された注文データ（注文失敗）

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明

## 2-4．決済におけるプライバシー

Paidy決済の認証では電話番号とメールアドレスを入力しますが、認証に使用した電話番号とメールアドレスについてはコマースクラウドでは保存されません。

ただし、会員登録時に入力した情報、配送、請求先として入力した情報はコマースクラウドに保存されます。

# 3.【実装ガイド】

## 3-1．カートリッジアップロード（UX Studio)

CommerceCloudへカートリッジのアップロードを行います。

・int\_paidy

・int\_paidy\_controllers（JSControllerサイトの場合）

PaidyカートリッジはSiteGenesisベースJSControllerサイトで使用できます。

Commerce Cloud UX-studioを使用して 'int\_paidy'および ‘int\_paidy\_controllers’カートリッジをCommerceCloudにアップロードします。

## 3-2．BusinessManagerセットアップ

Paidy決済の設定を、CommerceCloudの管理画面、BusinessManagerにて行います。

### 3-2-1. カートリッジの割当て

サイトにPaidyカートリッジを割当てます。

管理 > サイト > サイトの管理 > [管理しているサイト] - 設定



int\_paidy\_controllers:int\_paidy:加盟店様カートリッジ

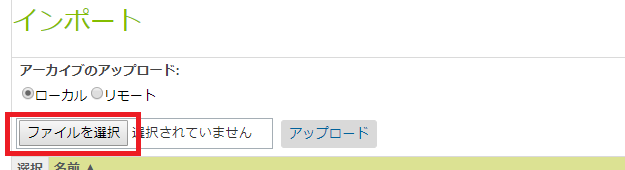
### 3-2-2. メタデータのインポート

パッケージ内にmeta\_data\_22.1.0\_sg.zipを用意しています。

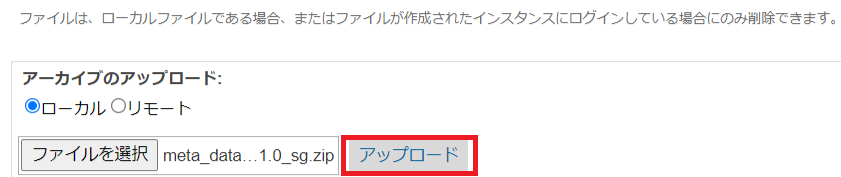
このzipファイルをインポートすることで、Paidy決済を使用する為の、サービスの追加、支払方法・支払処理業者の追加、システムオブジェクト、カスタムオブジェクトの構成の変更をします。詳細については後述します。

管理 > サイトの開発 > サイトのインポート & エクスポート

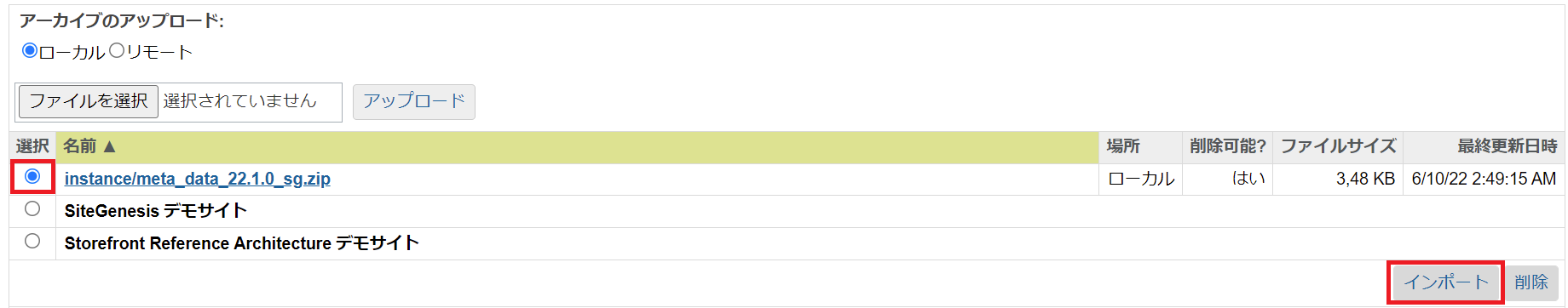
[ファイルを選択]をクリックします。



パッケージ内のmeta\_data\_22.1.0\_sg.zipを選択し、[アップロード]ボタンをクリックします。



zipファイルアップロード後、meta\_data\_22.1.0\_sg.zipにチェックを入れ、[インポート]をクリックします。



[OK]をクリックします。



・サービスメタデータインポート時の構成

メタデータをインポートした際のサービスの構成について記載します。

注意点として、インポート後に追加されるサービスについては

記載している情報の変更を行わないでください。

<サービス認証情報>

名前：paidy.http.payment

URL：https://api.paidy.com

<サービスプロフィール>

名前：paidy.api.prof

<サービス>

名前：paidy.api.payment

タイプ：HTTP

有効：check on

サービスモード：Live

プロフィール：paidy.api.prof

認証情報：paidy.http.payment

・支払方法・支払処理業者メタデータインポート時の構成

メタデータをインポートした際の支払方法・支払処理業者の構成について記載します。

注意点として、インポート後に追加される支払方法・支払処理業者については

記載している情報の変更を行わないでください。

<支払処理業者>

ID：PAIDY\_STANDARD

<支払処理業者>

ID：PAIDY\_SUBSCRIPTION

<支払方法>

ID：PAIDY\_STANDARD

Name：あと払い（ペイディ）

有効：はい

<支払方法>

ID：PAIDY\_SUBSCRIPTION

Name：あと払い（ペイディ）※定期購入

有効：はい

・メタデータインポート時のオブジェクト構成

メタデータをインポートした際のオブジェクトの構成について記載します。

注意点として、インポート後に追加されるシステムオブジェクトやカスタムオブジェクトについては

変更などを行わないでください。

<Order>

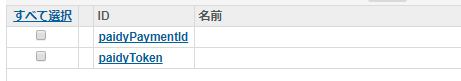
管理 > サイトの開発 > システムオブジェクトタイプ > Order - 属性定義

システムオブジェクト：Orderは注文情報に関するオブジェクトです。

Orderにはカスタム属性として以下を定義しています。

paidyPaymentId：Paidyと決済した際に発行されるIDを保持します。

paidyToken：定期決済を利用した場合に、使用したトークンの値を保持します。



<Site Preferences>

管理 > サイトの開発 > システムオブジェクトタイプ > Site Preferences - 属性定義

システムオブジェクト：Site Preferencesはサイトの設定に関するオブジェクトです。

Site Preferencesにはカスタム属性として以下を定義しています。

paidy\_api\_key：PaidyのAPIと接続するための公開鍵（パブリックキー）情報です。

　　　　　　　 加盟店用のPaidy加盟店管理画面で確認可能です。

　　　　　　　テストの場合はテスト用の公開鍵（パブリックキー）を登録します。

paidy\_enabled：Paidyカートリッジを使用するかしないかを設定します。オフにすると、  
　　　　　　　　決済手段としてpaidyの決済は利用できなくなります。

paidy\_secret\_key：PaidyのAPIと接続するための秘密鍵（シークレットキー）情報です。

　　　　　　　　 加盟店用のPaidy加盟店管理画面で確認可能です。

　　　　　　　　 テストの場合はテスト用の秘密鍵（シークレットキー）を登録します。

・Paidy加盟店管理画面(Paidy加盟店管理画面については4-1-1に説明を記載しています)



paidy\_logo\_url：Paidy Checkoutアプリケーションの表示時に表示するロゴ画像のURLです。

　　　　　　　 指定がない場合はPaidyのロゴが表示されます。

paidy\_service\_name：BM内で登録している「Paidyと通信する為の情報」を呼び出す為のキー情報です。

　　　　　　　　　　"paidy.api.payment"固定となります。

paidy\_store\_name：Paidy Checkoutアプリケーションのヘッダー部分、MyPaidy、

　　　　　　　　　マーチャントダッシュボードに表示される店舗名です。

paidy\_token\_description：（任意項目）加盟店が定義するトークンの説明を指定します。

テーブル

自動的に生成された説明

<Customer Profile>

管理 > サイトの開発 > システムオブジェクトタイプ > Customer Profile - 属性定義

システムオブジェクト：Customer Profileはカスタマー情報に関するオブジェクトです。

Customer Profileにはカスタム属性として以下を定義しています。

paidyToken：定期決済時に利用するトークン情報です。トークンを発行したタイミングで登録します。



<カスタムサイト環境設定>

マーチャントツール > サイト環境設定 > カスタムサイト環境設定グループ

メタデータのインポートが正常に完了すると、カスタム環境設定グループに「Paidy」のIDでグループが作成されています。



[Paidy]リンクをクリックすると、各値の設定画面が表示されますので、必要な情報を入力し、右上の[保存]ボタンをクリックして保存します。

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明

## 3-3．加盟店様でのカートリッジ修正

### 3-3-1. CheckoutJSの設置

Paidy Chekoutを呼び出し、決済を行うために

Checkout JSスクリプトを支払いページのテンプレートに埋め込みます。

このスクリプトを記述することで、Paidy Chekoutが呼び出されます。

テンプレートの埋め込むには、下記ファイルに修正を加えます。

・[storefront\_core]\cartridge\templates\default\components\footer\footer\_UI.isml

<script src="${URLUtils.staticURL('/js/app.js')}"></script>の直後に以下を追加

...........................

<isscript>

if (!empty(pdict.Basket)) {

var paymentMethods = pdict.Basket.getPaymentInstruments();

var paymentMethod = '';

if (paymentMethods != null && !paymentMethods.isEmpty() ) {

for( var i = 0; i < paymentMethods.length ; i++ ){

if( paymentMethods[i] != null && paymentMethods[i].paymentMethod != "GIFT\_CERTIFICATE" )

paymentMethod = paymentMethods[i].paymentMethod;

}

}

}

</isscript>

<isif condition="${paymentMethod=='PAIDY\_STANDARD'}">

<script type="text/javascript" src="https://apps.paidy.com/"></script>

<script src="${URLUtils.staticURL('js/paidyStandard.js')}"></script>

<iselseif condition="${paymentMethod=='PAIDY\_SUBSCRIPTION'}">

<script type="text/javascript" src="https://apps.paidy.com/"></script>

<script src="${URLUtils.staticURL('js/paidySubscription.js')}"></script>

<iselse/>

<script>

(function(){

var button = $('.submit-order button[type="submit"]');

if(button.length) {

button.prop('disabled', false);

button.removeClass('disabled');

}

}());

</script>

</isif>

...........................

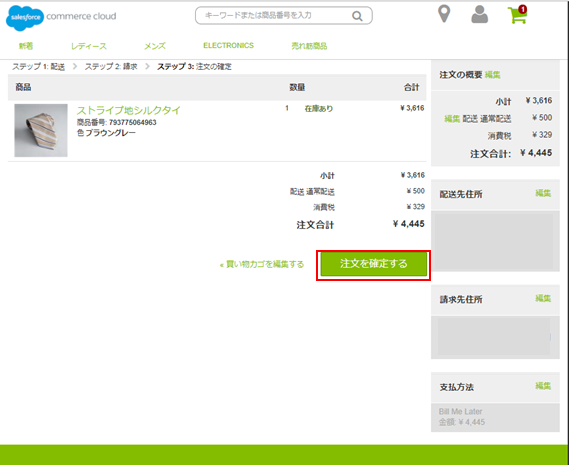


グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, チャットまたはテキスト メッセージ

自動的に生成された説明

### 3-3-2. 購入確定ボタンの制御について

購入確定画面を表示した際に、HTMLソースが完全に読み込まれる前に購入確定ボタンをクリックした際、Paidy Chekoutのポップアップが正常に動作しない可能性があります。



その為、購入確定ボタンのスタイルについて初期表示時はボタンを押せないようにする対応が必要です。

テンプレートでの対応は以下のようになります。

・[storefront\_core]\cartridge\templates\default\checkout\summary\summary.isml

クラス「button-fancy-large」が付与されているbuttonにdisabledクラス、disabled属性を追加

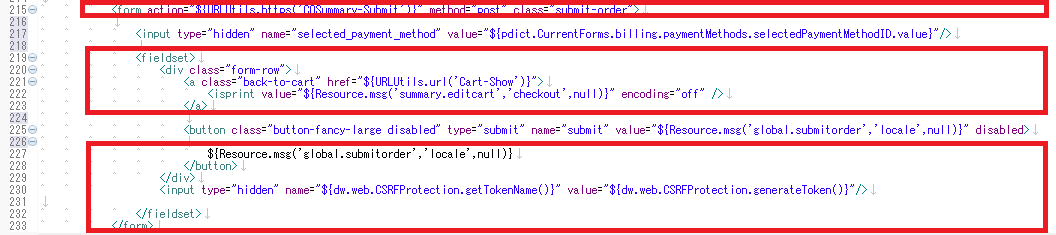
<button class="button-fancy-large disabled" type="submit" name="submit" value="${Resource.msg('global.submitorder','locale',null)}" disabled>

<form action="${URLUtils.https('COSummary-Submit')}" method="post" class="submit-order">のフォームタグ直後にタグを追加

<input type="hidden" name="selected\_payment\_method" value="${pdict.CurrentForms.billing.paymentMethods.selectedPaymentMethodID.value}"/>

また、上記formタグに”submit-order”クラスが指定してありますが、このクラスについては削除しないようにしてください。

購入確定ボタンはPaidy Checkoutの準備が完了したタイミングで押せるようになります。



### 3-3-3. リソースファイルの修正について

Paidy決済を行うにあたり、URLに関する設定をリソースファイルと呼ばれる設定ファイルに記載する必要があります。

・[storefront\_core]\cartridge\scripts\util\Resource.ds

　ResourceHelper.getUrlsのurls内の最後に以下を追加

...........................

getPaidyStandardPaidyConfig : URLUtils.https("PaidyStandard-GetPaidyConfig").toString(),

doPaidyStandardCOSummarySubmit: URLUtils.https("COSummary-Submit").toString(),

doPaidyStandardFailOrder : URLUtils.https("PaidyStandard-FailOrder").toString(),

doPaidyStandardPlaceOrder : URLUtils.https("PaidyStandard-PlaceOrder").toString(),

getPaidyStandardOrderToken : URLUtils.https("PaidyStandard-GetOrderToken").toString(),

getPaidySubscriptionConfig : URLUtils.url(PaidySubscription-GetPaidyConfig').toString(),

setPaidySubscriptionToken : URLUtils.url(PaidySubscription -SetPaidyToken').toString()

...........................

## 3-4．加盟店様でのカートリッジ修正

### 3-4-1. COBillingjsの修正について

クレジットカードやBMLからPaidyに支払方法を変更した際、カード情報などの入力値をクリアする処理を追加します。

...........................

function resetPaymentForms() {

var cart = app.getModel('Cart').get();

var status = Transaction.wrap(function () {

if (app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals('PayPal')) {

app.getForm('billing').object.paymentMethods.creditCard.clearFormElement();

app.getForm('billing').object.paymentMethods.bml.clearFormElement();

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_CREDIT\_CARD));

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_BML));

cart.removePaymentInstruments(cart.getPaymentInstruments('PAIDY\_STANDARD'));

cart.removePaymentInstruments(cart.getPaymentInstruments('PAIDY\_SUBSCRIPTION'));

} else if (app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals(PaymentInstrument.METHOD\_CREDIT\_CARD)) {

app.getForm('billing').object.paymentMethods.bml.clearFormElement();

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_BML));

cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));

cart.removePaymentInstruments(cart.getPaymentInstruments('PAIDY\_STANDARD'));

cart.removePaymentInstruments(cart.getPaymentInstruments('PAIDY\_SUBSCRIPTION'));

} else if (app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals(PaymentInstrument.METHOD\_BML)) {

app.getForm('billing').object.paymentMethods.creditCard.clearFormElement();

if (!app.getForm('billing').object.paymentMethods.bml.ssn.valid) {

return false;

}

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_CREDIT\_CARD));

cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));

cart.removePaymentInstruments(cart.getPaymentInstruments('PAIDY\_STANDARD'));

cart.removePaymentInstruments(cart.getPaymentInstruments('PAIDY\_SUBSCRIPTION'));

} else if (app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals('PAIDY\_STANDARD')) {

app.getForm('billing').object.paymentMethods.creditCard.clearFormElement();

app.getForm('billing').object.paymentMethods.bml.clearFormElement();

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_CREDIT\_CARD));

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_BML));

cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));

cart.removePaymentInstruments(cart.getPaymentInstruments('PAIDY\_SUBSCRIPTION'));

} else if (app.getForm('billing').object.paymentMethods.selectedPaymentMethodID.value.equals('PAIDY\_SUBSCRIPTION')) {

app.getForm('billing').object.paymentMethods.creditCard.clearFormElement();

app.getForm('billing').object.paymentMethods.bml.clearFormElement();

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_CREDIT\_CARD));

cart.removePaymentInstruments(cart.getPaymentInstruments(PaymentInstrument.METHOD\_BML));

cart.removePaymentInstruments(cart.getPaymentInstruments('PayPal'));

cart.removePaymentInstruments(cart.getPaymentInstruments('PAIDY\_STANDARD'));

}

return true;

});

return status;

}

...........................



### 3-4-2. COPlaceOrder.jsの修正について

PaidyからAjaxでOrderを作成した場合は、決済処理の実行前に中断する処理を追記します。

...........................

function handlePayments(order) {

(中略)

} else {

var authorizationResult = PaymentProcessor.authorize(order, paymentInstrument);

if (authorizationResult.not\_supported || authorizationResult.error) {

return {

error: true**,**

**PlaceOrderError: authorizationResult.PlaceOrderError**

};

}

}

}

}

return {};

}

function start() {

(中略)

var handlePaymentsResult = handlePayments(order);

if (handlePaymentsResult.error) {

return Transaction.wrap(function () {

OrderMgr.failOrder(order);

**if (handlePaymentsResult.PlaceOrderError != null) {**

return {

error : true,

**PlaceOrderError : handlePaymentsResult.PlaceOrderError**

**};**

**} else {**

**return {**

**error : true,**

PlaceOrderError : new Status(Status.ERROR,'confirm.error.technical')

};

**}**

});

} else if (handlePaymentsResult.missingPaymentInfo) {

(中略)

}

**if (request.httpParameterMap.format.stringValue == 'ajax') {**

**return {**

**Order: order,**

**paidy\_order\_created: true**

**};**

**}**

var orderPlacementStatus = Order.submit(order);

if (!orderPlacementStatus.error) {

clearForms();

}

return orderPlacementStatus;

}

...........................

### 3-4-3. COSummary.jsの修正について

PaidyからAjaxでOrderを作成した場合は、決済処理の実行前に中断した後、JSONをレスポンスする処理を追記します。

...........................

function submit() {

var placeOrderResult = app.getController('COPlaceOrder').Start();

if (placeOrderResult.error) {

start({

PlaceOrderError: placeOrderResult.PlaceOrderError

});

} else if (placeOrderResult.order\_created) {

showConfirmation(placeOrderResult.Order);

**} else if (placeOrderResult.paidy\_order\_created) {**

**var paidyCartridge = require('~/package.json').paidyCartridge;**

**var currentForms = session.forms;**

**app**

**.getView({**

**JSONResponse: require(paidyCartridge +**

**'/cartridge/scripts/paidy/standard/authorize').getConfirmationPaidyJSON(**

**currentForms.billing.paymentMethods.selectedPaymentMethodID.value,**

**customer,**

**placeOrderResult.Order**

**)**

**})**

**.render('util/responsejson');**

}

}

...........................



### 3-4-4. 加盟店様側のJSコントローラカートリッジのpackage.jsonの修正について

加盟店様側のJSコントローラカートリッジのpackage.jsonに、Paidyのカートリッジ参照を追加します。

...........................

{

"hooks": "./cartridge/scripts/hooks.json"**,**

**"paidyCartridge": "int\_paidy"**

}

...........................

### 3-4-5. int\_paidy\_controllersカートリッジのpackage.jsonの修正について

“controllers”で指定されるカートリッジ名を、加盟店様側のJSコントローラカートリッジの名前に変更します。

...........................

{

"paidyCartridge": "int\_paidy",

"hooks": "./cartridge/scripts/hooks.json",

**"controllers": "paidy\_storefront\_controllers"**

}...........................

## 3-5. アドバイス

・定期購入利用の注意点

　3-2-3. 支払方法の追加 に従って支払方法を設定した場合、標準では"通常決済"と"定期決済"が別々に存在します。

テキスト

中程度の精度で自動的に生成された説明

支払方法の選択画面に遷移してきたタイミングで、「定期決済」を行う為の手続きであることが判別可能であれば、css等を利用して支払方法を「あと払い(ペイディ)」のみの表示に制御することができます。

・トークンについての注意事項

マーチャントツール > 顧客 > 顧客の管理 > [顧客ナンバー] - 全般

トークンについては顧客情報で確認することが出来ます。



何らかの理由によりトークンを再発行したい場合は、paidyTokenの値を空で入力し更新を行い、

コンシューマーに再度定期購入を頂くことで可能となります。

・定期購入時の注文完了メールについて  
int\_paidyカートリッジでは、注文完了メールは定期購入であっても、通常の注文完了メールを送信する仕組みとなっております。カスタマイズによって送信されるメールを変更したい場合は、int\_paidyカートリッジの変更を行うことで対応が可能です。

・定期購入の決済否決時のエラーメッセージについて

定期購入の決済否決時のメッセージは、以下の文言をご参考ください。

------------------------------------------------------

今回の決済は承認されませんでした

申し訳ございませんが、オンラインショップが提供する他の支払方法をご利用ください。

なお、審査結果の詳細につきましては開示できませんのであらかじめご了承ください。

Paidyに関するお問い合わせ

0120-971-918

------------------------------------------------------

# 4.加盟店様でのカートリッジ導入テスト

## 4-1.テストを開始する前に

### 4-1-1.Paidy加盟店管理画面

・テスト用APIキーの確認

カスタムサイト環境設定の入力のため、「テスト用APIキー」のパブリックキー、シークレットキーを確認します。



### 4-1-2.Commerce Cloud Business Manager

・APIキーの設定

カスタムサイト環境設定にて、paidy\_api\_keyとpaidy\_secret\_keyに、4-1-1.Paidy加盟店管理画面で確認した、テスト用APIキーを入力します。



## 4-2.PaidyCheckoutのテスト情報

・mail addressとphone number

テストケースと利用可能なメールアドレス、電話番号グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明

|  |  |  |
| --- | --- | --- |
| テストケース | mail address | phone number |
| 成功 | successful.payment@paidy.com | 080-0000-0001 |
| 失敗 | rejected.payment@paidy.com |

※会員情報・配送先情報・請求先情報のmail address、phone numberは上記と異なっても問題ありません。

・認証番号

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション, メール

自動的に生成された説明8888を入力します。

## 4-3.ブラウザ互換性

当カートリッジでは以下のOS、ブラウザでの実行を対象とします。

|  |  |
| --- | --- |
| Device - OS | Browser |
| PC - Windows | Edge最新版 |
| Chrome 最新版 |
| FireFox 最新版 |
| PC - MacOS | Chrome 最新版 |
| Safari 最新版 |
| スマートフォン - iOS | iPhone (iOS最新版) Safari  iPhone (iOS最新版) Chrome |
| スマートフォン - Android | Android4.4以上 Chrome |

## 4-4.テストケース

カートリッジ設定後に実行すべきテストケースを以下に記載します。

### 4-4-1.Paidy 通常決済

通常決済はゲスト、会員でのテストを行います。

(1)ゲストでの注文成功

・ログインせず、Paidy通常決済の支払方法を選択し、PaidyCheckoutのポップアップが表示されたら、成功するmail addressとphone numberを入力して商品購入完了画面が表示されること

・Paidyマーチャントサイトで今回作成されたCommerceCloudの注文番号が、取引IDとして表示されること

・Commerce CloudのBusiness Managerで、対象の注文番号の注文ステータスが、新規となっていること

(2)会員での注文成功

・ログイン後、Paidy通常決済の支払方法を選択し、PaidyCheckoutのポップアップが表示されたら、成功するmail addressとphone numberを入力して商品購入完了画面が表示されること

・Paidyマーチャントサイトで今回作成されたCommerceCloudの注文番号が、取引IDとして表示されること

・Commerce CloudのBusiness Managerで、対象の注文番号の注文ステータスが、新規となっていること

(3) ゲストでの注文失敗

・ログインせず、Paidy通常決済の支払方法を選択し、PaidyCheckoutのポップアップが表示されたら、失敗するmail addressとphone numberを入力すると、エラーメッセージが表示されること

・Commerce CloudのBusiness Managerで、対象の注文番号の注文ステータスが、失敗となっていること

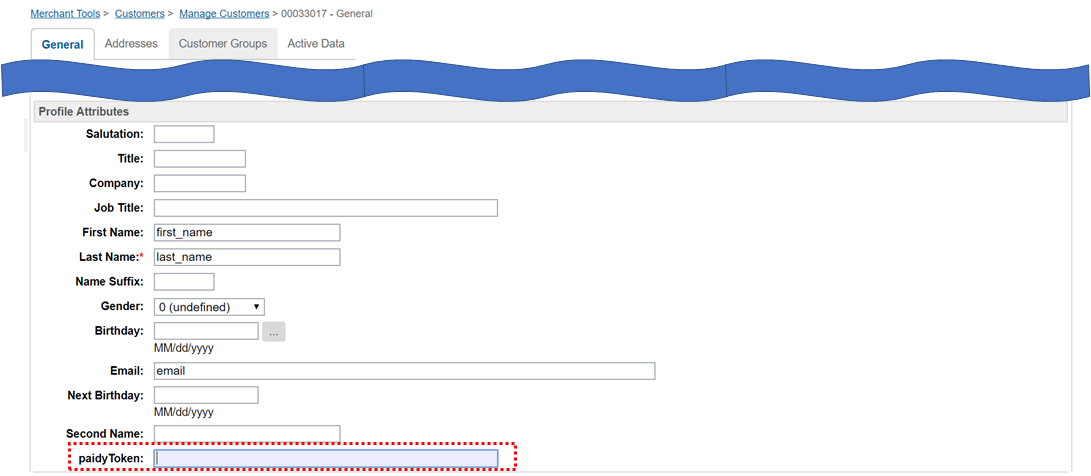
(4)会員での注文失敗

・ログイン後、Paidy通常決済の支払方法を選択し、PaidyCheckoutのポップアップが表示されたら、失敗するmail addressとphone numberを入力すると、エラーメッセージが表示されること

・Commerce CloudのBusiness Managerで、対象の注文番号の注文ステータスが、失敗となっていること

### 4-4-2.Paidy定期決済

定期決済を利用する場合はログインが必ず必要となる。定期決済で1度認証に成功すると、会員情報のpaidyTokenにトークン情報が保存され、2回目以降はPaidyCheckoutのポップアップが表示されなくなるため、テストの必要に応じてCommerce CloudのBusiness ManagerよりpaidyTokenを削除すること。



(1)トークンなしの成功

・トークンをまだ保持していない会員でログインし、Paidy定期決済の支払方法を選択し、「注文を確定する」ボタンを押下し、PaidyCheckoutのポップアップが表示されたら、成功するmail addressとphone numberを入力して処理を進め、商品購入完了画面が表示されること

・Paidyマーチャントサイトで今回作成されたCommerceCloudの注文番号が、取引IDとして表示されること

・Commerce CloudのBusiness Managerで、対象の注文番号の注文ステータスが、新規となっていること

(2)トークンなしの失敗

・Paidy定期決済の支払方法を選択し、「注文を確定する」ボタンを押下し、PaidyCheckoutのポップアップが表示されたら、失敗するmail addressとphone numberを入力して処理を進め、エラーメッセージが画面に表示されること

・Commerce CloudのBusiness Managerで、対象の注文番号の注文ステータスが、失敗となっていること

(3)トークンありの注文の成功

・1度Paidy定期決済を成功しているアカウントでログインし、Paidy定期決済の支払方法を選択し、「注文を確定する」ボタンを押下すると、商品購入完了画面が表示されること

・トークンが存在するため、PaidyCheckoutのポップアップが表示されないこと

・Paidyマーチャントサイトで今回作成されたCommerceCloudの注文番号が、取引IDとして表示されること

・Commerce CloudのBusiness Managerで、対象の注文番号の注文ステータスが、新規となっていること

# 5. Webhook運用案

## 5-1. Webhook運用について

Paidy決済は、JavaScriptアプリケーションを介してコンシューマーとPaidyが直接通信をして与信を実行するものです。したがって、コンシューマーの離脱や通信の断絶によって、決済完了の結果をECサイトが受け取れず、Paidy側にのみ決済データが残るケースが、仕様上の制約として存在します。

この不整合が発生した場合の問題は２点あります。

１．その注文を成立させる機会をショップが失う。

２．その注文を成立させなくてよい場合、コンシューマーがPaidyの与信枠を取られたままとなる。

これに対処するための機能としてWebhookがあります。

加盟店はWebhookを使用し、不整合決済を検知できます。

## 5-2. 不整合決済検知の流れ

ダイアグラム

自動的に生成された説明

①．決済

Commerce CloudからPaidyを利用して決済を行う。Paidyからの決済結果のコールバックにより、下記3種類の注文ステータスが発生する。

・決済成功：New（新規） →注文一覧へ出力される。支払い可能な状態

・決済失敗：Failed（失敗） →注文一覧へ出力されない。支払い不可能な状態

・決済途中：Created（作成済） →注文一覧へ出力されない。支払い不可能な状態、決済のUPDATEを行うことにより支払い可能な状態へ遷移する。

②．Webhook

マーチャントのバックエンドへWebhookエンドポイントを設定することにより、Paidyからの決済成功が通知される。

③．注文一覧

Commerce Cloudのバッチ処理により決済成功した注文一覧が、マーチャントのバックエンドへxmlファイルで送信される。

④．不整合決済チェック処理

Webhookのデータ（②）は存在するが、Commerce Cloudの決済データ（③）が存在しない決済をチェックする。

## 5-3. 不整合の対処方法

検知した不整合決済については、以下の対処を行うことにより、上記にあげた問題を解消することが可能です。

### 5-3-1. 対処案１

不整合となった決済をリストに出力し、加盟店の担当者が確認する案です。

確認後、しかるべき対処（決済成立、または決済キャンセル）を行います。

**１）決済を成立させる方法**

Commerce CloudのBusinessManagerよりOrder機能を利用し、ステータスを更新する。

・Generalタブ

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

上記3か所のステータスを以下のように変更

Order Status：Created→Open

Confirmation Status：Not Confirm→Confirmed

Export Status:Not Exported→Ready for Export

・Attributeタブ

グラフィカル ユーザー インターフェイス, テキスト, アプリケーション

自動的に生成された説明

paidyPaymentIdにWebhookで届いたpaidyPaymentIDをセット

※注文完了メールをStoreFrontと同じようにメール送信するためには、以下の様にフォーマットが異なるため、カスタマイズが必要となります。

・注文完了メールのイメージ

■SiteGenesisのStoreFrontのメール（通常使用されるメール）

グラフィカル ユーザー インターフェイス, アプリケーション

自動的に生成された説明

■BusinessManagerのOrders、Generalタブの下部、SendMailボタンから送信したメール

グラフィカル ユーザー インターフェイス

自動的に生成された説明

**２）決済をキャンセルする方法**

Paidyの加盟店管理画面より対象の決済を表示し、Cancel（キャプチャーせずにクローズ）を行う。

グラフィカル ユーザー インターフェイス, アプリケーション, Teams

自動的に生成された説明

### 5-3-2. 対処案２

システム的に判断し、不整合決済を自動キャンセルする案です。

CommerceCloudからの注文データとWebhookのデータを照合し、CommerceCloudでは決済が成立せず、Paidy側だけでオーソリデータができていると判断できればクローズAPIをリクエストし、決済をクローズします。

https://paidy.com/docs/jp/payments.html#close

## 5-4. Webhookの設定方法

Paidy加盟店管理画面にて、バックエンドでPaidyからの通知を受け取るために、Paidy加盟店管理画面にWebhook URLの登録を行います。

URL：https://merchant.paidy.com

本番環境に対しての設定は、

“Webhooks - 「本番用APIキー」および「旧バージョンAPIアクセスキー」”

にマーチャントのバックエンド本番URLにて設定を行います。

本番設定前のテスト決済時に疎通を確認する場合は、

“Webhooks - 「テスト用APIキー」”

にマーチャントのバックエンドテストURLにて設定を行います。

Paidy加盟店管理画面



# ６.サービス停止時の対応について

◼︎サービス停止時  
Paidy決済を選択するとエラー画面が表示されます、またはポップアップが動作しなくなります。

◼︎加盟店の対応  
まずは、Paidyテクニカルサポートにお問い合わせください。  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
Paidy テクニカルサポート  
Mail： [tech-support@paidy.com](mailto:tech-support@paidy.com)TEL ： 03-5545-5099 （平日 10:00-18:00）  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

復旧までに時間がかかる場合は、カスタム環境設定でpaidy\_enabledの有効フラグを「いいえ」にしてください。  


# ７.改訂履歴

|  |  |  |
| --- | --- | --- |
| Version | 日付 | 改訂内容 |
| 17.10.0 | 2017/10/13 | 初版 |
| 18.01.0 | 2018/01/09 | ・JSControllerでのカートリッジ開発、およびPipelineサイトとJSControllerサイト両方からの利用のため、手順を修正した。  ・Commerce CloudでのWebHook廃止に伴い改定した。  また、5.運用案を追加した |
| 19.01.0 | 2019/10/25 | ・meta情報のファイルを修正、利用手順の更新。  ・JScontrollerで利用されている、function『ServiceRegistry』を廃止、『LocalServiceRegistry』を利用。  ・JScontrollerで呼び出されている、service処理に、『getResponseLogMessage』を実装。  ・HTTPClientの呼び出しを、全てserviceの利用に対応。  ・各functionにDocCommentを追加。 |
| 20.1.0 | 2020/01/15 | ページ：3-2-2. メタデータのインポート  変更内容：「meta\_data\_19.1.0.zip」を「meta\_data\_20.1.0.zip」に変更 |
| 20.1.0 | 2020/4/5 | ファイル名変更に伴い、記載内容を修正 項目：3-3-1. CheckoutJS installation  変更前： <script src="${URLUtils.staticURL('js/paidy\_standard.js')}"></script> <script src="${URLUtils.staticURL('js/paidy\_regular.js')}"></script> 変更後： <script src="${URLUtils.staticURL('js/paidyNormal.js')}"></script>  <script src="${URLUtils.staticURL('js/paidyRegular.js')}"></script>  サービス停止時に起こることと、その際に加盟店がどのように対応すべきかを記載。  Pipelineサイトに関する表記を削除。 |
| 20.1.0 | 2020/5/15 | ページ：１概要  互換モード「19.1」を追記。  ページ：3-2-2 メタデータのインポート 変更内容： 「paidy\_enabled」の設定内容を追記  ページ:6 サービス停止時の対応について  変更内容： 「paidy\_enabled」にて支払いをオフにする方法を記載。 |
| 21.1.0 | 2021/6/21 | ページ:表紙  versionを変更  ページ: 16  変更内容：「meta\_data\_20.1.0.zip」を「meta\_data\_21.1.0\_sg.zip」に変更 |
| 22.1.0 | 2022/6/21 | ページ:表紙  versionを変更    ページ：全体  Paidy通常決済、定期決済のIDを以下のように変更  ・normal->standard  ・regular->subscription    ページ:15  インポートmetaファイル名更新    ページ：37  4-3.ブラウザ互換性からIE11を削除 |
| 22.1.1 | 2022/11/24 | ページ：全体  Paity通常決済、定期決済のサービス名変更に伴い、以下変更  ●通常決済  あと払い (ペイディ) ●定期決済  あと払い (ペイディ) ※定期購入  ●支払方法説明  ・クレジットカード、事前登録不要。 ・メールアドレスと携帯番号だけで、今すぐお買い物。 ・1か月に何度お買い物しても、お支払いは翌月まとめて1回でOK。 ・お支払いは翌月10日までに、コンビニ払い・銀行振込・口座振替で。  ペイディについて詳しくはこちら。  上記変更に伴い、キャプチャも変更 | |
|  |  |  |