

Paidy payment processing SFRA

Version<21.1.0>



Table of contents

Details

Table of contents	2
1. Summary.....	4
2. Component summary	5
2-1. Features overview.....	5
2-2. Restrictions and limitations.....	5
2-3. Use cases.....	6
2-3-1. Standard payments.....	6
2-3-2. Subscription payments.....	10
2-4. Privacy when paying.....	15
3. Implementation guide.....	16
3-1. Uploading cartridges (UX Studio).....	16
3-2. BusinessManager setup.....	16
3-2-1. Cartridge assignment	16
3-2-2. Importing metadata.....	18
3-3. Modifying cartridges on the merchant	22
3-3-1. Add payment method.....	22
3-3-2. CheckoutJS installation.....	27
3-3-3. Added paidy.js loading.....	27
3-3-4. Add attribute on order confirmation button.....	28
3-3-5. CheckoutServices.js modifications.....	28
3-3-6. Add payment method to email template.....	32
3-4. Advice	33
4. Cartridge deployment testing at merchant	35
4-1. Before beginning a test	35
4-1-1. Paidy merchant dashboard screen.....	35
4-1-2. Commerce Cloud Business Manager	36
4-2. PaidyCheckout test data	36
4-3. Browser compatibility.....	37

4-4. Test cases.....	37
4-4-1. Paidy standard payments.....	37
4-4-2. Paidy subscription payments.....	38
5. Webhook usage examples.....	39
5-1. Webhook usage.....	39
5-2. Detecting inconsistent payments.....	39
5-3. Handling inconsistencies	40
5-3-1. Processing example 1	40
5-3-2. Processing example 2	44
5-4. Webhook settings.....	44
6. What to do when the service is down.....	45
7. Revision history	45

1. Summary

This implementation guide supports Store Front Reference Architecture (SFRA 6.0.0) . Compatibility Mode up to 21.2 is supported.

Cartridges allow a Commerce Cloud store to make use of Paidy payments.

The developer must follow this text to install the cartridges and set them up on the online store.

Because Paidy allows for creating test payments by making use of a test API key, the user can test the settings before switching over to a live environment. For details on data needed for testing, see [4-1. Before beginning a test..](#)

A cartridges must be implemented in order to begin using Paidy payments.

- "int_paidy_sfra" is the core cartridge that enables use of Paidy payments on the store.

*Other cartridges are used on SiteGenesis based cartridges.

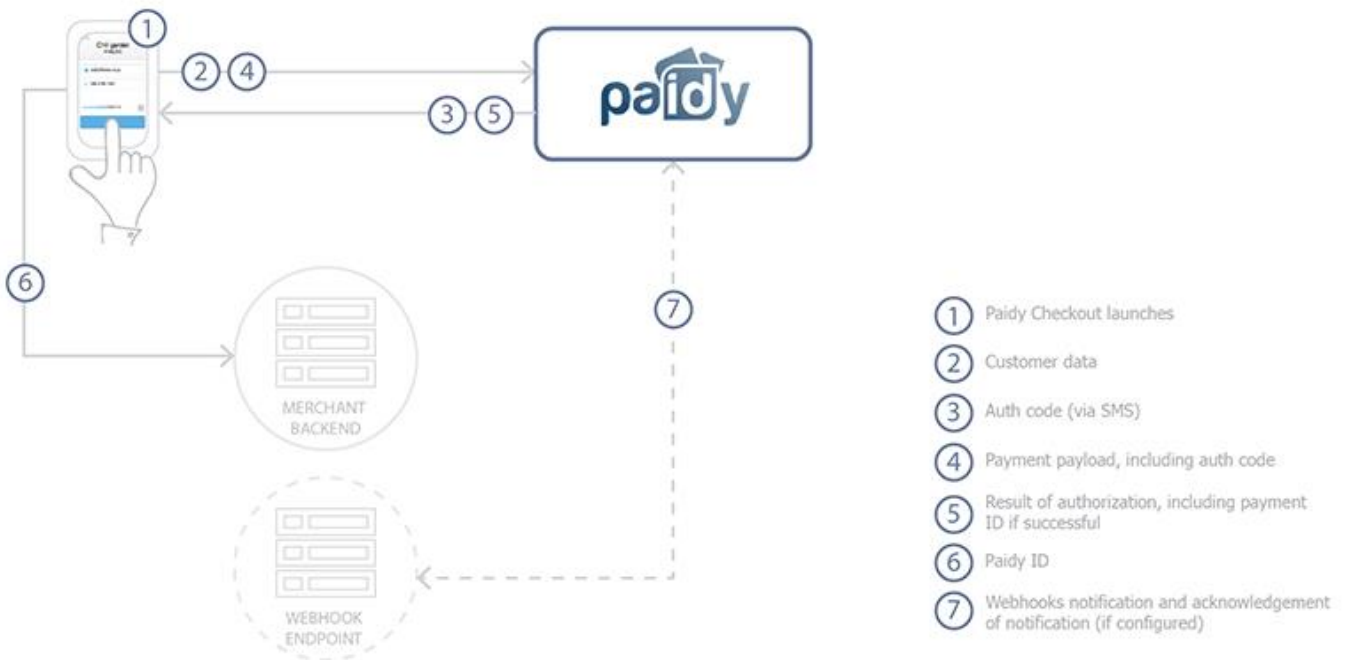
2. Component summary

2-1. Features overview

The process flow for Paidy is explained in the diagram below.

Paidy cartridges correspond to point 1-6 in this workflow.

In step 1-6, the results of authentication with Paidy are saved on the site.



2-2. Restrictions and limitations

- Only a “creation” process is in place for payment. Capture, Refund, Update, and Close must be implemented by the merchant.
- For first-time subscription payments, Commerce Cloud runs a creation process.
Subsequent payments must be configured at the merchant, but the payment process can be implemented by executing the functions found in the Paidy cartridge.
- Subscription payments are designed to only work when the user is logged in.
- Confirm notifications from the Webhook on OMS; if the payment process results are incorrect, you can choose to set the order status to shippable in SFCC, or cancel the payment on Paidy. For details, please see [5. Webhook usage examples.](#)

- This cartridge can only be used on Japanese language sites. (language: Japanese; currency: JPY)

2-3. Use cases

2-3-1. Standard payments

Paidy Checkout launches when a user clicks a payment purchase button on a site.

Consumers enter the requisite information and make a Paidy payment.

If the payment is successful, the process transitions to the product purchase confirmation screen.

- Payment flow

Select Paidy standard payments when selecting a payment method when checking out and proceed to confirming the order.

The screenshot displays the Paidy checkout interface. At the top, there are three payment method icons: CREDIT, Paidy, and Paidy. The Paidy icon is highlighted with a red box. Below the icons, a dropdown menu is open, showing the selected option: 'Paidy 翌月払い (コンビニ/銀行)'. The main content area features a blue banner with the text '今月のお買いもの コンビニ・銀行 まとめて翌月払い' (This month's purchases at convenience stores/banks, paid in full next month). Below the banner, there is a section titled 'Paidy 翌月払い (コンビニ/銀行)' with the tagline '買いたいを、カンタンに' (Buy what you want, easily). The text below states: '月に何回お買い物をしても、お支払いは翌月にまとめて1回。' (No matter how many times you shop in a month, you only pay once next month). It then lists the available payment methods: '下記のお支払い方法がご利用いただけます。' (The following payment methods can be used). The list includes: '・口座振替(支払手数料:無料)' (Direct debit (payment fee: free)), '・コンビニ(支払手数料:356円税込)' (Convenience store (payment fee: 356 yen tax included)), and '・銀行振込(支払手数料:金融機関により異なります)' (Bank transfer (payment fee: varies by financial institution)). At the bottom, there is a link 'Paidyについて詳しくはこちら。' (Click here for more details about Paidy). A blue button at the bottom right says '次へ: 注文を確定する' (Next: Confirm order).

The Paidy Checkout popup appears.

注文手続き

配送

配送先住所:
100-0000
東京都

0123456789
配送方法:
Delivery

支払

請求先住所:
100-0000

paidy

事前登録は不要

メールアドレスと携帯電話番号でお支払い

請求はまとめて月1回

翌月1~3日にご利用1ヶ月分の請求案内をメールでお届け

お支払いは翌月10日まで

コンビニ*、銀行振込、口座振替から選べるお支払い方法
*コンビニ払いの場合は支払手数料350円(税込)が発生します。

次へ

..... ¥ 3,616

..... ¥ 500

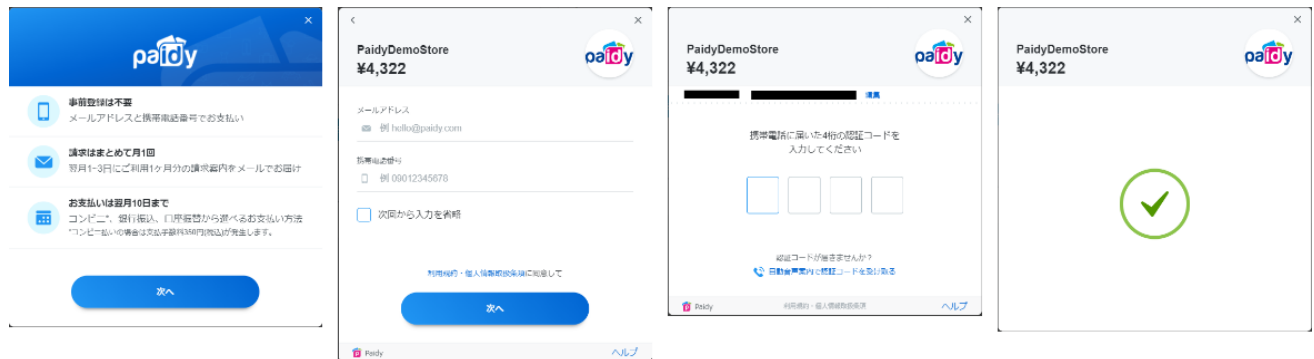
..... ¥ 4,322

..... ¥ 3,616

タイ
ッド
り

単価	数量	合計
¥4,763- ¥	1	¥ 3,616

The requisite information is filled in and Paidy payment is processed.



If the payment succeeds, the order is complete.



Order data created on Commerce Cloud

全般 属性 支払 メモ 履歴

注文 '00000506' の詳細

情報:

1 件の配送先への 1 項目を含む。合計金額は ¥ 4,322.00 です。

受取日:

10/31/19 8:50:19 AM Etc/UTC

サイト:

PaidyDEMO

作成者:

顧客

顧客:

顧客ナンバー:

1001

IP アドレス:

該当なし

Eメール:

電話:

注文ステータス:

未処理

確認ステータス:

確認済み

配送ステータス:

未配送

エクスポートステータス:

エクスポート可能

配送 00002503

個数	商品 ID	名前	Manufacturer (製造元)	税率	小売単価	税基準	項目合計
1	793775362380M	ストライプ地シルクタイ		5.00 %	¥ 3,616.00	¥ 3,616.00	¥ 3,616.00
						荷物配送料: :	¥ 500.00
						合計配送料 (Delivery):	¥ 500.00
						配送合計:	¥ 500.00
						税金合計:	¥ 206.00
						合計:	¥ 4,322.00

Eメールの送信

注文の印刷

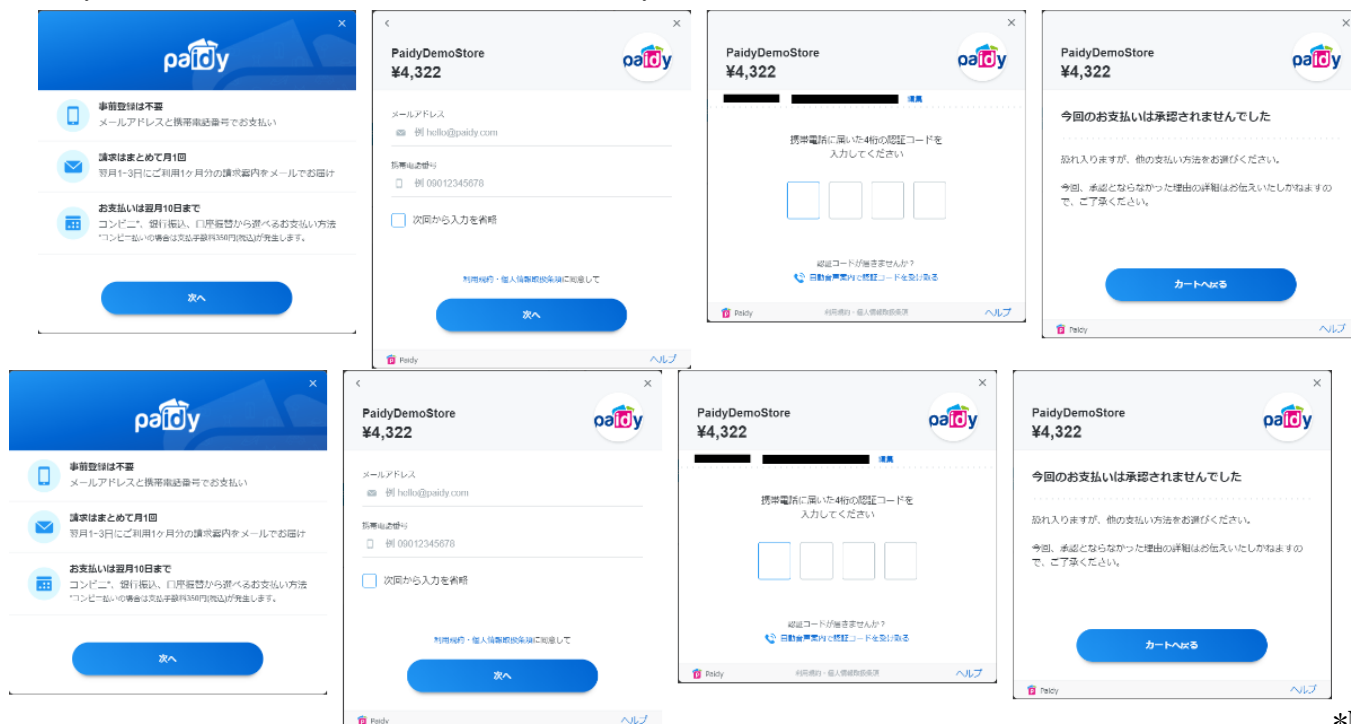
Payment data created on Paidy



決済ID	顧客名	金額	オーソリ日時	ステータス	キャプチャー期限
pay_Xbqi_VIAAEcARLUA TEST	*****	¥ 4,322	2019-10-31 18:01:49	AUTHORIZED 取引ID: 00000506	2019-11-30 18:02:18
pay_Xbqi_ZVIAAEARKr3 TEST	*****	¥ 4,322	2019-10-31 17:46:29	AUTHORIZED 取引ID: 00000505	2019-11-30 17:46:31
pay_XbqeuVIAACBARKJ TEST	*****	¥ 4,322	2019-10-31 17:43:37	AUTHORIZED 取引ID: 00000504	2019-11-30 17:43:41

If the payment fails, Paidy Checkout displays a payment error message and order failure data is created on Commerce Cloud.

Payment data will not be created on Paidy.



*Note

that if the top-right X button is clicked on the popup, order failure data is created. Payment data will not be created on Paidy.

Order data created on Commerce Cloud (order failure)

全般 属性 支払 メモ 履歴

注文 '00000601' の詳細

情報:	1 件の配送先への 1 項目を含む。合計金額は ¥ 4,322.00 です。							
受取日:	11/1/19 1:15:11 AM Etc/UTC							
サイト:	Paidy DEMO							
作成者:	顧客							
顧客:	*****							
顧客ナンバー:	1001							
IP アドレス:	該当なし							
Eメール:	[REDACTED]							
電話:	[REDACTED]							
注文ステータス:	失敗	確認ステータス:	未確認					
配送ステータス:	未配送	エクスポートステータス:	未エクスポート					
配送								
個数	商品 ID	名前	ステータス	Manufacturer (製造元)	税率	小売単価	税基準	項目合計
1	793775362380M	ストライプ地シルクタイ	CANCELLED		5.00 %	¥ 3,616.00	¥ 3,616.00	¥ 3,616.00
							荷物配送料: CANCELLED	¥ 500.00
							合計配送料 (Delivery):	¥ 500.00
							配送合計:	¥ 500.00
							税金合計:	¥ 206.00
							合計:	¥ 4,322.00
							Eメールの送信	注文の印刷

2-3-2. Subscription payments

Only usable by logged in users.

When clicking on the order confirmation button, Paidy Checkout launches. Consumers enter the requisite information and authenticate with Paidy. If the authentication completes correctly, a token is issued and is assigned to the order.

- Payment flow

Select Paidy subscription payments when selecting a payment method when checking out and proceed to confirming the order.

CREDIT

paidy

paidy
Paidy翌月払い (コンビニ/銀行) ※定期購入

今月のお買いもの **コンビニ・銀行**

paidy
買いたいを、
カンタンに

まとめて翌月払い

月に何回お買い物をしても、お支払いは翌月にまとめて1回。

下記のお支払い方法がご利用いただけます。

- ・口座振替 (支払手数料: 無料)
- ・コンビニ (支払手数料: 356円税込)
- ・銀行振込 (支払手数料: 金融機関により異なります)

Paidyについて詳しくは [こちら](#)。

次へ: 注文を確認する

An order confirmation screen is displayed and the order is confirmed.

編集

Paidy翌月払い（コンビニ/銀行）※定期購入



単価	数量	合計
¥4,763- ¥	1	¥3,616
3,616		

注文を確定する

The Paidy Checkout popup appears.



Enter the requisite information on Paidy Checkout and a token is issued.

*If the popup is closed during the process by the user clicking the X button on the top right, the popup will no longer display, but a token can be reissued and the payment processed.



After issuing a token, the process moves to payment processing. If the payment succeeds, we transition to the order confirmation screen.



Order data created on Commerce Cloud

全般 属性 支払 メモ 履歴

注文 '00000603' の詳細

情報:

1 件の配送先への 1 項目を含む。合計金額は ¥ 4,322.00 です。

受取日:

11/1/19 2:03:20 AM Etc/UTC

サイト:

PaidyDEMO_sfra

作成者:

顧客

顧客:

顧客ナンバー:

1001

IP アドレス:

該当なし

Eメール:

電話:

注文ステータス:

未処理

確認ステータス:

確認済み

配送ステータス:

未配送

エクスポートステータス:

エクスポート可能

配送 00003002

個数	商品 ID	名前	Manufacturer (製造元)	税率	小売単価	税基準	項目合計
1	793775362380M	ストライプ地シルクタイ		5.00 %	¥ 3,616.00	¥ 3,616.00	¥ 3,616.00
						荷物配送料: :	¥ 500.00
						合計配送料 (Delivery):	¥ 500.00
						配送合計:	¥ 500.00
						税金合計:	¥ 206.00
						合計:	¥ 4,322.00

Eメールの送信

注文の印刷

Payment data created on Paidy

ホーム

支払管理

レポート

設定

検索

payto

(developer)

ログアウト

ホーム

最近の支払

決済ID	顧客名	金額	オーソリ日時	ステータス	キャプチャー期限
pay_XbuSaFIAAFQASAJa	TEST	¥ 4,322	2019-11-01 11:03:20	AUTHORIZED 取引ID: 00000603	2019-12-01 11:03:20
pay_XbuR8VIAAEgASAEs	TEST	¥ 4,322	2019-11-01 11:01:21	AUTHORIZED 取引ID: 00000602	2019-12-01 11:01:21
pay_Xbqi_VIAAEcARLUA	TEST	¥ 4,322	2019-10-31 18:01:49	AUTHORIZED 取引ID: 00000506	2019-11-30 18:02:18

If payment fails, a payment error will be displayed on the order confirmation page and order failure data is created on Commerce Cloud.

Payment data will not be created on Paidy.

salesforce commerce cloud

注文手続き

サポートをご利用になりますか? こちらまでお電話ください:
1-800-555-0199

申し訳ございませんが、ご注文を完了することができませんでした。ご注文が殺到しているため、または一時的な接続エラーが原因だと思われます。数分後にご注文を再度送信してください。ご注文が正しく完了するまで、お支払いが処理されることはありません。その他にもご不明がございましたら、弊社までお問い合わせください。

注文の概要

小計	¥ 3,616
配送	¥ 500
合計	¥ 4,322

配送

編集

配送先住所:
100-0000
東京都 *****

1点の商品

¥ 3,616

ストライプ地シルクタイ



色: レッド
在庫あり

Order data created on Commerce Cloud (order failure)

マーチャントツール > 注文 > 注文: 00002101(PaidyDEMO_sfra)

全般 属性 支払 メモ 履歴

注文 '00002101' の詳細

情報:

受取日: 11/15/19 7:09:15 AM Etc/UTC

サイト: PaidyDEMO_sfra

作成者: 顧客

顧客: *****

顧客ナンバー: 1001

IP アドレス: 該当なし

Eメール:

電話:

注文ステータス: 失敗

確認ステータス: 未確認

配送ステータス: 未配送

エクスポートステータス: 未エクスポート

個数	商品 ID	名前	ステータス	Manufacturer (製造元)	税率	小売単価	税基準	項目合計
1	793775362380M	ストライプ地シルクタイ	CANCELLED		5.00 %	¥ 3,616.00	¥ 3,616.00	¥ 3,616.00
荷物配送料: CANCELLED								¥ 500.00
合計配送料 (Delivery):								¥ 500.00
配送合計:								¥ 500.00
税金合計:								¥ 206.00
合計:								¥ 4,322.00

[Eメールの送信](#)
[注文の印刷](#)

2-4. Privacy when paying

Paidy payment authentication requires entering a telephone number and e-mail address, but these are not saved on Commerce Cloud.

However, information entered upon registration and shipment and billing details are saved on Commerce Cloud.

3. Implementation guide

3-1. Uploading cartridges (UX Studio)

The cartridge is uploaded to Commerce Cloud.

- int_paidy
- int_paidy_sfra

The cartridge is for Store Front Reference Architecture (SFRA).

*If your website is based SiteGenesis, please refer to SiteGenesis implementation guide.

Use Commerce Cloud UX-studio to upload the int_paidy_sfra cartridges to Commerce Cloud.

3-2. BusinessManager setup

Paidy payment settings are configured on the Commerce Cloud management screen (Business Manager).

3-2-1. Cartridge assignment

A Paidy cartridge is assigned to the site.

Management > Sites > Site Management > Sites Under Management: Settings

RefArch - 設定

適用をクリックして詳細を保存します。前回保存された状態に戻すには、リセットをクリックします。

インスタンスタイプ: Sandbox (サンドボックス)/Development (開発) ▼

廃止予定。HTTP と HTTPS のホスト名の構成では、サイトのエイリアス構成の新しい機能を使用することをお勧めします ("SEO > エイリアス構成")。既定する HTTP/HTTPS ホスト名の値は、エイリアス構成でホスト名が定義されていない場合に、古い設定スタイルをサポートする目的でのみ使用されます。

HTTP ホスト名:

HTTPS ホスト名:

インスタンスタイプ: すべて

カートリッジ:

int_paidy_sfra:int_paidy:app_storefront_base

有効なカートリッジのパス:

int_paidy_sfra
int_paidy
app_storefront_base
plugin_apple_pay
plugin_facebook
plugin_payments
plugin_pinterest_commerce
plugin_web_payments
bc_content
core

int_paidy_sfra:int_paidy:merchant cartridge

3-2-2. Importing metadata

Meta_data_21.1.0.zip is provided in the package.

By importing this zip file, you can add services, add payment methods / payment processors, change system objects and custom objects to use Paidy payment. Details will be described later.

Administration > Site Development > Site Import & Export

Click [Choose File].

The screenshot shows the 'インポート' (Import) page. Under 'アーカイブのアップロード:' (Archive Upload:), the 'ローカル' (Local) radio button is selected. The 'ファイルを選択' (Choose File) button is highlighted with a red box. To its right is a button that says '選択されていません' (Not selected). Further right is the 'アップロード' (Upload) button.

Select meta_data_21.1.0.zip in the package and click the [Upload] button.

The screenshot shows the 'インポート' (Import) page. Under 'アーカイブのアップロード:' (Archive Upload:), the 'ローカル' (Local) radio button is selected. The 'ファイルを選択' (Choose File) button is now disabled, and the text 'meta_data_21.1.0.zip' is displayed next to it. The 'アップロード' (Upload) button is highlighted with a red box.

After uploading the zip file, check meta_data_21.1.0.zip and click [Import].

選択	名前 ▲	場所	削除可能?	ファイルサイズ	最終更新日時
<input checked="" type="radio"/>	instance/meta_data_21.1.0.zip	ローカル	はい	3,96 KB	6/21/21 6:01:11 AM
<input type="radio"/>	SiteGenesis デモサイト				
<input type="radio"/>	Storefront Reference Architecture デモサイト				

インポート 削除

Click [OK].

The screenshot shows the 'インポート' (Import) page. At the bottom, there is a confirmation message: '選択したアーカイブをインポートしますか?' (Do you want to import the selected archive?). To the right of this message, the 'OK' button is highlighted with a red box, and the 'キャンセル' (Cancel) button is also visible.

-Configuration when importing service metadata

Describes the service configuration when importing metadata.

Note that for services added after import
Please do not change the information described.

< Service Credentials >

Name : paidy.http.payment

URL : https://api.paidy.com

< Service Profiles >

Name : paidy.api.prof

< Services >

Name : paidy.api.payment

Type : HTTP

Enabled : check on

Service Mode : Live

Profile : paidy.api.prof

Credentials : paidy.http.payment

・ Payment method ・ Configuration when importing payment processor metadata

Describes the payment method and payment processor configuration when metadata is imported.

As a point of caution, for payment methods and payment processors added after import

Please do not change the information described.

<Payment Processors >

ID : PAIDY_NORMAL

<Payment Processors >

ID : PAIDY_REGULAR

< Payment Methods >

ID : PAIDY_NORMAL

Name : Paidy翌月払い (コンビニ/銀行)

Enabled : Yes

< Payment Methods >

ID : PAIDY_REGULAR

Name : Paidy翌月払い (コンビニ/銀行) ※定期購入

Enabled : Yes

- Object layout for metadata importing

Here we include instructions on the layout of objects when importing metadata.

One word of caution is that system and custom objects added after importing should not be modified.

<Order>

Management > Site Development > System Object Types> Order - Attribute Definitions

System objects: Order is an object that corresponds to order data

Order is given the below definitions as custom attributes.

paidyPaymentId: saves the ID issued when making a Paidy payment.

paidyToken: saves the token value used when using subscription payments.

すべて選択	ID	名前
<input type="checkbox"/>	paidyPaymentId	
<input type="checkbox"/>	paidyToken	

<Site Preferences>

Management > Site Development > System Object Types > Site Preferences > Attribute Definitions

System object: Site Preferences is an object that corresponds to site settings.

The below is defined as a custom attribute in Site Preferences.

paidy_api_key: public key data used to connect to the Paidy API.

This can be confirmed on the Paidy merchant management screen.

For a test, save the public key used for testing purposes.

paidy_secret_key: secret key data used to connect to the Paidy API.

This can be confirmed on the Paidy merchant management screen.

When testing, save the secret key used for testing purposes.

- Paidy merchant management screen (details found in section 4-1-1)

本番用APIキー	
パブリックキー	<input type="text" value="pk_live_"/>
シークレットキー	<input type="text" value="sk_live_"/>
テスト用APIキー	
パブリックキー	<input type="text" value="pk_test_"/>
シークレットキー	<input type="text" value="sk_test_"/>

paidy_logo_url: the URL of a logo image to be displayed on the Paidy Checkout application.

If nothing is set, the Paidy logo will be shown.

paidy_service_name: key data used to call the information for communicating with Paidy as registered in BM.

This is fixed to paidy.api.payment.

paidy_store_name: the shop name displayed on Paidy Checkout application headers, MyPaidy, and the merchant dashboard.

paidy_token_description (optional): specifies descriptions of tokens defined by the merchant.

すべて選択	ID	名前
<input type="checkbox"/>	paidy_api_key	
<input type="checkbox"/>	paidy_logo_url	
<input type="checkbox"/>	paidy_secret_key	
<input type="checkbox"/>	paidy_service_name	
<input type="checkbox"/>	paidy_store_name	
<input type="checkbox"/>	paidy_token_description	

<Customer Profile>

Management > Site Development > System Object Types > Customer Profile > Attribute Definitions
System object: Customer Profile is an object relating to customer data.

The below is defined as a custom attribute on the Customer Profile.

paidyToken: token data used for subscription payments. Saved when issuing a token.

すべて選択	ID	名前
<input type="checkbox"/>	paidyToken	

<Custom site environment settings>

Merchant Tools > Site Environment Settings > Custom Site Environment Settings Group
Once metadata is successfully imported, a group with the Paidy ID will be created in the custom environment settings group.

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

サイト全体で表示

新規

Q

▼

1 ~ 1 / 1

ID	名前	説明	環境設定	サイト全体で表示
Paidy			6	表示

Clicking Paidy displays configuration screens for each value. Enter the requisite information and click the Save button at the top right.

Instances Type: Sandbox (サンドボックス)

ID で検索...

名前	値	デフォルト値
paidy_api_key	<input type="text"/>	サイト全体で編集
paidy_logo_url	<input type="text"/>	サイト全体で編集
paidy_secret_key	<input type="text"/>	サイト全体で編集
paidy_store_name	<input type="text"/>	サイト全体で編集
paidy_service_name	<input type="text"/>	サイト全体で編集
paidy_token_description	<input type="text"/>	サイト全体で編集

© 2020 salesforce.com, inc. All Rights Reserved. RefArch タイムゾーン: 協定世界時 | インスタンスのタイムゾーン: アメリカ東部夏時間 | バージョン: 20.4 最終更新日時: 3月 30, 2020 (互換モード: 18.10)

3-3. Modifying cartridges on the merchant

3-3-1. Add payment method

In order to display Paidy by payment method, write it in the template of the payment page.

By writing this script, you can:

- Selection tab added to payment methods
- The explanation for consumers is displayed when Paidy payment is selected
- The type of Paidy payment selected by the payment method is displayed on the order confirmation screen when selecting Paidy payment.

To embed a template, modify the following file.

• [app_storefront_base]/cartridge/templates/default/checkout/billing/paymentOptions/paymentOptionsTabs.isml

Add the below after </isif>.

```
.....
<isif condition="{paymentOption.ID === 'PAIDY_NORMAL'}">
    <include template="checkout/billing/paymentOptions/paidyNormalTab" />
</isif>
<isif condition="{paymentOption.ID === 'PAIDY_REGULAR'}">
```

```
<isinclude template="checkout/billing/paymentOptions/paidyRegularTab" />
</isif>
```

```
.....
<isloop items="{pdict.order.billing.payment.applicablePaymentMethods}" var="paymentOption">↓
  <isif condition="{paymentOption.ID == 'CREDIT_CARD'}">↓
    <isinclude template="checkout/billing/paymentOptions/creditCardTab" />↓
  </isif>↓
  <isif condition="{paymentOption.ID == 'PAIDY_NORMAL'}">↓
    <isinclude template="checkout/billing/paymentOptions/paidyNormalTab" />↓
  </isif>↓
  <isif condition="{paymentOption.ID == 'PAIDY_REGULAR'}">↓
    <isinclude template="checkout/billing/paymentOptions/paidyRegularTab" />↓
  </isif>↓
</isloop>↓
```

The screenshot shows a checkout form with two input fields: "*Eメール" (Email) and "*電話番号" (Phone Number). Below the fields are three payment options: "CREDIT", "paidy", and "paidy". The "paidy" options are highlighted with a blue bar.

• [app_storefront_base]/cartridge/templates/default/checkout/billing/paymentOptions/paymentOptionsContent.isml

Add the below after </isif>.

```
.....
<isif condition="{paymentOption.ID == 'PAIDY_NORMAL'}">
  <isinclude template="checkout/billing/paymentOptions/paidyNormalContent" />
</isif>
<isif condition="{paymentOption.ID == 'PAIDY_REGULAR'}">
  <isinclude template="checkout/billing/paymentOptions/paidyRegularContent" />
</isif>
```

```
.....
<isloop items="{pdict.order.billing.payment.applicablePaymentMethods}" var="paymentOption">↓
  <isif condition="{paymentOption.ID == 'CREDIT_CARD'}">↓
    <isinclude template="checkout/billing/paymentOptions/creditCardContent" />↓
  </isif>↓
  <isif condition="{paymentOption.ID == 'PAIDY_NORMAL'}">↓
    <isinclude template="checkout/billing/paymentOptions/paidyNormalContent" />↓
  </isif>↓
  <isif condition="{paymentOption.ID == 'PAIDY_REGULAR'}">↓
    <isinclude template="checkout/billing/paymentOptions/paidyRegularContent" />↓
  </isif>↓
</isloop>↓
```

今月のお買いもの **コンビニ・銀行**

まとめて翌月払い

買いたいを、
カンタンに

月に何回お買い物をして、お支払いは翌月にまとめて1回。

下記のお支払い方法がご利用いただけます。

- ・口座振替(支払手数料:無料)
- ・コンビニ(支払手数料:356円税込)
- ・銀行振込(支払手数料:金融機関により異なります)

Paidyについて詳しくは[こちら](#)。

次へ: 注文を確定する

・[app_storefront_base]/cartridge/templates/default/checkout/billing/paymentOptions/paymentOptionsSummary.isml

Add the below after `</isif>` and `</div>`.

.....

(omission)

```

<isif condition="{payment.paymentMethod === 'CREDIT_CARD'}">
  <isinclude template="checkout/billing/paymentOptions/creditCardSummary" />
</isif>
<isif condition="{payment.paymentMethod === 'PAIDY_NORMAL'}">
  <isinclude template="checkout/billing/paymentOptions/paidyNormalSummary" />
</isif>
<isif condition="{payment.paymentMethod === 'PAIDY_REGULAR'}">
  <isinclude template="checkout/billing/paymentOptions/paidyRegularSummary" />
</isif>
</isloop>
</div>
<div class="payment-details-summary paidy-normal" style="display: none;">
  <isinclude template="checkout/billing/paymentOptions/paidyNormalSummary" />
</div>
<div class="payment-details-summary paidy-regular" style="display: none;">
  <isinclude template="checkout/billing/paymentOptions/paidyRegularSummary" />

```


</div>

```
.....
<div class="payment-details">↓
  <isloop items="{pdict.order.billing.payment.selectedPaymentInstruments}" var="payment">↓
    <isif condition="{payment.paymentMethod === 'CREDIT_CARD'}">↓
      <isinclude template="checkout/billing/paymentOptions/creditCardSummary" />↓
    </isif>↓
    <isif condition="{payment.paymentMethod === 'PAIDY_NORMAL'}">↓
      <isinclude template="checkout/billing/paymentOptions/paidyNormalSummary" />↓
    </isif>↓
    <isif condition="{payment.paymentMethod === 'PAIDY_REGULAR'}">↓
      <isinclude template="checkout/billing/paymentOptions/paidyRegularSummary" />↓
    </isif>↓
  </isloop>↓
</div>↓
<div class="payment-details-summary paidy-normal" style="display: none;">↓
  <isinclude template="checkout/billing/paymentOptions/paidyNormalSummary" />↓
</div>↓
<div class="payment-details-summary paidy-regular" style="display: none;">↓
  <isinclude template="checkout/billing/paymentOptions/paidyRegularSummary" />↓
</div>↓
```

支払

編集

請求先住所:
100-0000

支払:
Paidy翌月払い (コンビニ/銀行)

注文を確定する

• [app_storefront_base]/cartridge/scripts/checkout/checkoutHelpers.js
Add the below function.

```
.....
/**
 * Selected payment instrument
 * @param {dw.order.Order} order - The order object to be placed
 * @returns {string|null} selected payment instrument
 */
function selectedPaymentInstrument(order) {
  if (order.billing.payment && order.billing.payment.selectedPaymentInstruments
    && order.billing.payment.selectedPaymentInstruments.length > 0) {
    return order.billing.payment.selectedPaymentInstruments[0].paymentMethod;
  } else {
    return null;
  }
}
```

```

}
(omission)
  setGift: setGift,
  selectedPaymentInstrument: selectedPaymentInstrument
};

```

```

728 /**
729  * Selected payment instrument
730  * @param {dw.order.Order} order - The order object to be placed
731  * @returns {string|null} selected payment instrument
732  */
733 function selectedPaymentInstrument(order) {
734   if (order.billing.payment && order.billing.payment.selectedPaymentInstruments
735       && order.billing.payment.selectedPaymentInstruments.length > 0) {
736     return order.billing.payment.selectedPaymentInstruments[0].paymentMethod;
737   } else {
738     return null;
739   }
740 }
741
742 module.exports = {
743   getFirstNonDefaultShipmentWithProductLineItems: getFirstNonDefaultShipmentWithProductLineItems,
744   ensureNoEmptyShipments: ensureNoEmptyShipments,
745   getProductLineItem: getProductLineItem,
746   isShippingAddressInitialized: isShippingAddressInitialized,
747   prepareCustomerForm: prepareCustomerForm,
748   prepareShippingForm: prepareShippingForm,
749   prepareBillingForm: prepareBillingForm,
750   copyCustomerAddressToShipment: copyCustomerAddressToShipment,
751   copyCustomerAddressToBilling: copyCustomerAddressToBilling,
752   copyShippingAddressToShipment: copyShippingAddressToShipment,
753   copyBillingAddressToBasket: copyBillingAddressToBasket,
754   validateFields: validateFields,
755   validateCustomerForm: validateCustomerForm,
756   validateShippingForm: validateShippingForm,
757   validateBillingForm: validateBillingForm,
758   validatePayment: validatePayment,
759   validateCreditCard: validateCreditCard,
760   calculatePaymentTransaction: calculatePaymentTransaction,
761   recalculateBasket: recalculateBasket,
762   handlePayments: handlePayments,
763   createOrder: createOrder,
764   placeOrder: placeOrder,
765   savePaymentInstrumentToWallet: savePaymentInstrumentToWallet,
766   getRenderedPaymentInstruments: getRenderedPaymentInstruments,
767   sendConfirmationEmail: sendConfirmationEmail,
768   ensureValidShipments: ensureValidShipments,
769   setGift: setGift,
770   selectedPaymentInstrument: selectedPaymentInstrument
771 };

```

• [app_storefront_base]/cartridge/templates/default/checkout/billing/paymentOptions.isml

Add the below the first line of the file (after div.form-nav) and attribute in div.form-nav

```

<isscript>
  <isset name="paymentMethod" value="$ {require('*/cartridge/scripts/checkout/checkoutHelpers
  ').selectedPaymentInstrument(pdict.order)}" scope="page"/>
</isscript>
<div class="form-nav billing-nav payment-information"
  data-payment-method-id="CREDIT_CARD"
  data-selected-payment-method="$ {paymentMethod}"

```

```
data-is-new-payment="{dict.customer.registeredUser && pdict.customer.customerPaymentInstr
uments.length ? false : true}"
>
```

```
.....
1 <isset name="paymentMethod" value="{require('*/cartridge/scripts/checkout/checkoutHelpers').selectedPaymentInstrument(pdict.order)}" scope="page"/>|
2 <div class="form-nav billing-nav payment-information"|
3   data-payment-method-id="CREDIT_CARD"|
4   data-selected-payment-method="{paymentMethod}"|
5   data-is-new-payment="{dict.customer.registeredUser && pdict.customer.customerPaymentInstruments.length ? false : true}"|
6 >|
7   <ul class="nav nav-tabs nav-fill payment-options" role="tablist">|
8     <include template="checkout/billing/paymentOptions/paymentOptionsTabs" />|
9   </ul>|
10 </div>|
11 <div class="credit-card-selection-new">|
12   <div class="tab-content">|
13     <include template="checkout/billing/paymentOptions/paymentOptionsContent" />|
14   </div>|
15 </div>
```

3-3-2. CheckoutJS installation

In order to process payment, the Checkout JS script is embedded in the payment page template. Including this script will call Paidy Checkout.

To embed the template, modify the following file.

- [app_storefront_base]/cartridge/templates/default/common/layout/checkout.isml

Add the below after <head>

```
.....
<!DOCTYPE html>
<html lang="en">
  <head>
    <script type="text/javascript" src="https://apps.paidy.com/"></script>
    <!--[if gt IE 9]><!-->
```

```
.....
<!DOCTYPE html>↓
<html lang="en">↓
  <head>↓
    <script type="text/javascript" src="https://apps.paidy.com/"></script>↓
    <!--[if gt IE 9]><!-->↓
    <include sf-toolkit="off" template="/common/scripts" />↓
```

3-3-3. Added paidy.js loading

Add the paidy.js required for the payment process below.

- [app_storefront_base]/cartridge/templates/default/checkout/checkout.isml

Add the below after assets.addJs('/js/checkout.js');

```
.....
<isscript>
  var assets = require('*/cartridge/scripts/assets.js');
  assets.addJs('/js/checkout.js');
  assets.addJs('/js/paidy.js');
  assets.addCss('/css/checkout/checkout.css');
</isscript>
```

```

.....
<isdecorate template="common/layout/checkout">↓
  <!-- Load Static Assets -->↓
  <!-- Load Static Assets -->↓
  <script>↓
    var assets = require('#/cartridge/scripts/assets.js');↓
    assets.addJs('/js/checkout.js');↓
    assets.addJs('/js/paidy.js');↓
    assets.addCss('/css/checkout/checkout.css');↓
  </script>↓

```

3-3-4. Add attribute on order confirmation button

Define the information required for the payment process as the attribute of the settlement confirmation button.

- [app_storefront_base]/cartridge/templates/default/checkout/checkout.isml

Add the below in button.place-order

```

.....
<button class="btn btn-primary btn-block place-order" data-action="{URLUtils.url('CheckoutServices-PlaceOrder')}"
  data-get-paidy-normal-config="{URLUtils.url('PaidyNormal-GetPaidyConfig')}"
  data-paidy-normal-fail-order="{URLUtils.url('PaidyNormal-FailOrder')}"
  data-paidy-normal-place-order="{URLUtils.url('PaidyNormal-PlaceOrder')}"
  data-get-paidy-regular-config="{URLUtils.url('PaidyRegular-GetPaidyConfig')}"
  data-paidy-regular-set-token="{URLUtils.url('PaidyRegular-SetPaidyToken')}"
  type="submit" name="submit" value="place-order">{Resource.msg('button.place.order', 'checkout', null)}
</button>

```

```

.....
<button class="btn btn-primary btn-block submit-payment" type="submit" name="submit" value="submit-payment">
  {Resource.msg('button.next.place.order', 'checkout', null)}
</button>
<button class="btn btn-primary btn-block place-order" data-action="{URLUtils.url('CheckoutServices-PlaceOrder')}"
  data-get-paidy-normal-config="{URLUtils.url('PAIDY_NORMAL-GetPaidyConfig')}"
  data-paidy-normal-fail-order="{URLUtils.url('PAIDY_NORMAL-FailOrder')}"
  data-paidy-normal-place-order="{URLUtils.url('PAIDY_NORMAL-PlaceOrder')}"
  data-get-paidy-regular-config="{URLUtils.url('PAIDY_REGULAR-GetPaidyConfig')}"
  data-paidy-regular-set-token="{URLUtils.url('PAIDY_REGULAR-SetPaidyToken')}"
  type="submit" name="submit" value="place-order">{Resource.msg('button.place.order', 'checkout', null)}
</button>
</div>
</div>
</div>

```

3-3-5. CheckoutServices.js modifications

Select Paidy as the payment method, and do not process credit card payment in Ajax processing when transitioning to the order confirmation screen.

- [app_storefront_base]/cartridge/controllers/CheckoutServices.js

Add the below.

```

.....

```

```

server.post(
    'SubmitPayment',
    server.middleware.https,
    csrfProtection.validateAjaxRequest,
    function (req, res, next) {
(omission)
        var validationHelpers = require('*/cartridge/scripts/helpers/basketValidationHelpers');
        var cartHelpers = require('*/cartridge/scripts/cart/cartHelpers');
        var paidyHelpers = require('*/cartridge/scripts/paidy/paidyHelpers');
        var currentBasket = BasketMgr.getCurrentBasket();
(omission)
    }

    paidyHelpers.resetPaymentForms(currentBasket);
    var billingAddress = currentBasket.billingAddress;
    var billingForm = server.forms.getForm('billing');
    .....

```

```

40 /** ↓
41 * Handle Ajax payment (and billing) form submit ↓
42 */ ↓
43 server.post( ↓
44     'SubmitPayment', ↓
45     server.middleware.https, ↓
46     csrfProtection.validateAjaxRequest, ↓
47     function (req, res, next) { ↓
48         var PaymentManager = require('dw/order/PaymentMgr'); ↓
49         var HookManager = require('dw/system/HookMgr'); ↓
50         var Resource = require('dw/web/Resource'); ↓
51         var COHelpers = require('*/cartridge/scripts/checkout/checkoutHelpers'); ↓
52     }
(中略)
141     var validationHelpers = require('*/cartridge/scripts/helpers/basketValidationHelpers'); ↓
142     var cartHelpers = require('*/cartridge/scripts/cart/cartHelpers'); ↓
143     var paidyHelpers = require('*/cartridge/scripts/paidy/paidyHelpers'); ↓
144     ↓
145     var currentBasket = BasketMgr.getCurrentBasket(); ↓
146     var validatedProducts = validationHelpers.validateProducts(currentBasket); ↓
147     ↓
148     var billingData = res.getViewData(); ↓
149     ↓
150     if (!currentBasket || validatedProducts.error) { ↓
151         delete billingData.paymentInformation; ↓
152         ↓
153         res.json({ ↓
154             error: true, ↓
155             cartError: true, ↓
156             fieldErrors: [], ↓
157             serverErrors: [], ↓
158             redirectUrl: URLUtils.url('Cart-Show').toString() ↓
159         }); ↓
160         return; ↓
161     } ↓
162     ↓
163     paidyHelpers.resetPaymentForms(currentBasket); ↓
164     ↓
165     var billingAddress = currentBasket.billingAddress; ↓
166     var billingForm = server.forms.getForm('billing'); ↓
167     var paymentMethodID = billingData.paymentMethod.value; ↓

```

Add a process to be interrupted before proceeding the payment process when creating an order from Paidy with Ajax

Add the below.

```

.....
server.post('PlaceOrder', server.middleware.https, function (req, res, next) {
    var BasketMgr = require('dw/order/BasketMgr');

```

```

(omission)
    var addressHelpers = require('*/cartridge/scripts/helpers/addressHelpers');
    var paidyHelpers = require('*/cartridge/scripts/paidy/paidyHelpers');
    var currentBasket = BasketMgr.getCurrentBasket();
(omission)
    if (handlePaymentResult.error) {
        res.json({
            error: true,
            errorMessage: Resource.msg('error.technical', 'checkout', null)
        });
        return next();
    }

    if (paidyHelpers.isPaidyNormal(currentBasket.paymentInstrument.paymentMethod)) {
        var responseJson = require('*/cartridge/scripts/paidy/normal/authorize')
            .getConfirmationPaidyJSON(currentBasket.paymentInstrument.paymentMethod, order.getCustomer(), order);
        res.json(responseJson);
        return next();
    } else if (paidyHelpers.isPaidyRegular(currentBasket.paymentInstrument.paymentMethod)) {
        var paidyRegularAuthorize = require(
            '*/cartridge/scripts/paidy/regular/authorize')
            .Authorize(order, req.querystring.paidyToken);
        if (paidyRegularAuthorize.error) {
            Transaction.wrap(function () { OrderMgr.failOrder(order); });
            res.json({
                error: true,
                errorMessage: Resource.msg('error.technical', 'checkout', null)
            });
            return next();
        }
    }

    var fraudDetectionStatus = hooksHelper('app.fraud.detection', 'fraudDetection', currentBasket, require('*/cartridge/scripts/hooks/fraudDetection').fraudDetection);
    .....

```

```

40 server.post('PlaceOrder', server.middleware.https, function (req, res, next) {↓
41     var BasketMgr = require('dw/order/BasketMgr');↓
42     var OrderMgr = require('dw/order/OrderMgr');↓
43     var Resource = require('dw/web/Resource');↓
44     var Transaction = require('dw/system/Transaction');↓
45     var URLUtils = require('dw/web/URLUtils');↓
46     var basketCalculationHelpers = require('*/cartridge/scripts/helpers/basketCalculationHelpers');
47     var hooksHelper = require('*/cartridge/scripts/helpers/hooks');↓
48     var COHelpers = require('*/cartridge/scripts/checkout/checkoutHelpers');↓
49     var validationHelpers = require('*/cartridge/scripts/helpers/basketValidationHelpers');↓
50     var addressHelpers = require('*/cartridge/scripts/helpers/addressHelpers');↓
51     ↓
52     var paidyHelpers = require('*/cartridge/scripts/paidy/paidyHelpers');↓
53     ↓
54     var currentBasket = BasketMgr.getCurrentBasket();↓
55     ↓

```

(中略)

```

81         errorMessage: Resource.msg('error.technical', 'checkout', null)↓
82     });↓
83     return next();↓
84 }↓
85 ↓
86     if (paidyHelpers.isPaidyNormal(currentBasket.paymentInstrument.paymentMethod)) {↓
87         var responseJson = require('*/cartridge/scripts/paidy/normal/authorize')↓
88         .getConfirmationPaidyJSON(currentBasket.paymentInstrument.paymentMethod, order.getCustomer(), order);↓
89         res.json(responseJson);↓
90         return next();↓
91     } else if (paidyHelpers.isPaidyRegular(currentBasket.paymentInstrument.paymentMethod)) {↓
92         var paidyRegularAuthorize = require('*/cartridge/scripts/paidy/regular/authorize')↓
93         .Authorize(order, req.querystring.paidyToken);↓
94         if (paidyRegularAuthorize.error) {↓
95             Transaction.wrap(function () { OrderMgr.failOrder(order); });↓
96             res.json({↓
97                 error: true,↓
98                 errorMessage: Resource.msg('error.technical', 'checkout', null)↓
99             });↓
100             return next();↓
101         }↓
102     }↓
103 ↓
104     var fraudDetectionStatus = hooksHelper('app.fraud.detection', 'fraudDetection', currentBasket, require('*/car
105     if (fraudDetectionStatus.status === 'fail') {↓

```

• [app_storefront_base]/cartridge/scripts/cart/cartHelpers.js

Add the below function.

.....

(omission)

/**

* Deletes multiple payment instruments.

*

* @transactional

* @alias module:models/CartModel~CartModel/removePaymentInstruments

* @param {dw.order.Basket} basket - the target Basket object

* @param {dw.util.Collection} paymentInstruments - The payment instruments to remove.

*/

```

function removePaymentInstruments(currentBasket, paymentInstruments) {
    collections.forEach(paymentInstruments, function(item) {
        currentBasket.removePaymentInstrument(item);
    });
}

```

(omission)

```

    updateBundleProducts: updateBundleProducts,
    removePaymentInstruments: removePaymentInstruments
};

.....

/** ↓
 * Deletes multiple payment instruments. ↓
 * ↓
 * @transactional ↓
 * @alias module:models/CartModel~CartModel/removePaymentInstruments ↓
 * @param {dw.order.Basket} basket - the target Basket object ↓
 * @param {dw.util.Collection} paymentInstruments - The payment instruments to remove. ↓
 */
function removePaymentInstruments(currentBasket, paymentInstruments) { ↓
    collections.forEach(paymentInstruments, function(item) { ↓
        currentBasket.removePaymentInstrument(item); ↓
    }); ↓
} ↓

module.exports = { ↓
    addLineItem: addLineItem, ↓
    addProductToCart: addProductToCart, ↓
    checkBundledProductCanBeAdded: checkBundledProductCanBeAdded, ↓
    ensureAllShipmentsHaveMethods: ensureAllShipmentsHaveMethods, ↓
    getQtyAlreadyInCart: getQtyAlreadyInCart, ↓
    getNewBonusDiscountLineItem: getNewBonusDiscountLineItem, ↓
    getExistingProductLineItemInCart: getExistingProductLineItemInCart, ↓
    getExistingProductLineItemsInCart: getExistingProductLineItemsInCart, ↓
    getMatchingProducts: getMatchingProducts, ↓
    allBundleItemsSame: allBundleItemsSame, ↓
    hasSameOptions: hasSameOptions, ↓
    BONUS_PRODUCTS_PAGE_SIZE: BONUS_PRODUCTS_PAGE_SIZE, ↓
    updateBundleProducts: updateBundleProducts, ↓
    removePaymentInstruments: removePaymentInstruments ↓
}; ↓

```

3-3-6. Add payment method to email template

Add a payment method to the email sent when payment is completed.

• [app_storefront_base]/cartridge/templates/default/checkout/confirmation/confirmationPaymentInfo.isml

Add the below after </isif> in isloop of Payment information

```

.....
(omission)

        <span>${Resource.msg('msg.card.type.ending', 'confirmation', null)} ${payment.expirationMonth}/${payment.expirationYear}</span>
    </div>
</isif>
<isif condition="${payment.paymentMethod == 'PAIDY_NORMAL'}">
    <include template="checkout/billing/paymentOptions/paidyNormalSummary" /
>
</isif>
<isif condition="${payment.paymentMethod == 'PAIDY_REGULAR'}">
    <include template="checkout/billing/paymentOptions/paidyRegularSummary"
/>
</isif>

```



```

        </isloop>
    </div>
</div>
.....

```

```

<!-- Payment information -->
<div>
  <strong>${Resource.msg('label.order.payment.info', 'confirmation', null)}</strong>
  <br/>
  <div>
    <isloop items="${pdict.order.billing.payment.selectedPaymentInstruments}" var="payment">
      <isif condition="${payment.paymentMethod == 'CREDIT_CARD'}>
        <div>
          <span>${Resource.msg('msg.payment.type.credit', 'confirmation', null)}
            ${payment.type}</span>
          </div>
          <div>
            ${payment.maskedCreditCardNumber}
          </div>
          <span>${Resource.msg('msg.card.type.ending', 'confirmation', null)}
            ${payment.expirationMonth}/${payment.expirationYear}</span>
          </div>
        </isif>
        <isif condition="${payment.paymentMethod == 'PAIDY_NORMAL'}>
          <include template="checkout/billing/paymentOptions/paidyNormalSummary" />
        </isif>
        <isif condition="${payment.paymentMethod == 'PAIDY_REGULAR'}>
          <include template="checkout/billing/paymentOptions/paidyRegularSummary" />
        </isif>
      </isloop>
    </div>
  </div>

```

3-4. Advice

- Notes on use of subscription purchases

If you followed the steps to add a payment method as indicated in section 3-3-1, by default, you will have two separate entries -- standard payments and subscription payments.

If there are means to detect whether subscription payments have been selected when transitioning to the payment selection screen, CSS or other means can be used to restrict the display to only show 「Paidy翌月払い(コンビニ/銀行)」.

- Notes on tokens

int_paidy_sfra does not use PaidyToken for customer information.

Tokens can be checked via order data.

Merchant Tools > Order > Order > Order Numbers: Attribute

PaidyPayments	
paidyToken:	<input type="text"/>
paidyPaymentId:	<input type="text"/>

- Order confirmation e-mails for subscription purchases

int_paidy_sfra cartridge is designed such that order confirmation e-mails are sent in the same manner as standard e-mails even for subscription purchases. To customize the e-mail sent, you can modify the int_paidy_sfra cartridge.

4. Cartridge deployment testing at merchant

4-1. Before beginning a test

4-1-1. Paidy merchant dashboard screen

- Confirming the test API key

Check the public and secret keys used for the test API; these will be entered into the custom site environment settings screen.

本番用APIキー	
パブリックキー	<input type="text" value="pk_live_"/>
シークレットキー	<input type="text" value="sk_live_"/>
テスト用APIキー	
パブリックキー	<input type="text" value="pk_test_"/>
シークレットキー	<input type="text" value="sk_test_"/>

- API key settings

名前	値	デフ
paidy_api_key	pk_test_	
paidy_logo_url		
paidy_secret_key	sk_test_	
paidy_store_name		
paidy_service_name		
paidy_token_description		

-mail address and phone number

E-mail address and telephone usable for test cases

Test cases	mail address	phone number
Success	successful.payment@paidy.com	080-0000-0001
Failure	rejected.payment@paidy.com	

*Note that the e-mail address and phone number for the above member data and shipping and billing address can differ.

-Authentication number



Enter 8888

4-3. Browser compatibility

This cartridge supports the following OSes and browsers.

Device - OS	Browser
PC - Windows	IE11: latest version
	Edge: latest version
	Chrome: latest version
	FireFox: latest version
PC - MacOS	Chrome: latest version
	Safari: latest version
Smartphone - iOS	iPhone (latest version of iOS) Safari
	iPhone (latest version of iOS) Chrome
Smartphone - Android	Android4.4 and later, Chrome

4-4. Test cases

Test cases to run after configuring the cartridge are listed below.

4-4-1. Paidy standard payments

Standard payment is tested using a guest and a member.

1. Successful order as guest

- If not logging in and selecting a payment method for Paidy standard payments, the PaidyCheckout popup appears. Entering a successful e-mail address and phone number will display the product purchase confirmation screen.
- The Commerce Cloud order number created on the Paidy merchant site should be displayed as a transaction ID
- The order status for the corresponding order number on Business Manager on Commerce Cloud will be new

2. Order success as member

- After logging in, when selecting a payment method for Paidy standard payments, the PaidyCheckout popup appears. Entering a successful e-mail address and phone number will display the product purchase confirmation screen.
- The Commerce Cloud order number created on the Paidy merchant site should be displayed as a transaction ID
- The order status for the corresponding order number on Business Manager on Commerce Cloud will be new

3. Failed order as guest

- If not logging in and selecting a payment method for Paidy standard payments, the PaidyCheckout popup appears. Entering a failing e-mail address and phone number will display an error message.
- The order status for the corresponding order number on Business Manager on Commerce Cloud will have failed

4. Order failure as member

- After logging in, when selecting a payment method for Paidy standard payments, the PaidyCheckout popup appears. Entering a failing e-mail address and phone number will display an error message.
- The order status for the corresponding order number on Business Manager on Commerce Cloud will have failed

4-4-2. Paidy subscription payments

Test as a member.

1. Success as a member

- When logging in as a member and selecting a payment method for Paidy subscription payments, the Confirm Order button is clicked and the PaidyCheckout popup appears. Entering a successful e-mail address and phone number and proceeding to processing the payment will display a product purchase confirmation screen
- The Commerce Cloud order number created on the Paidy merchant site should be displayed as a transaction ID
- The order status for the corresponding order number on Business Manager on Commerce Cloud will be new

2. Failure as a member

- When selecting a payment method for Paidy subscription payments and clicking the Confirm Order button, the PaidyCheckout popup appears. Entering a failing e-mail address and phone number and proceeding to processing the payment will display an error message
- The order status for the corresponding order number on Business Manager on Commerce Cloud will have failed

5. Webhook usage examples

5-1. Webhook usage

Paidy payments run credit holds for a purchase by means of communication directly between a consumer and Paidy over JavaScript. Therefore, one limitation of the system is that if the consumer bounces or the connection is cut, the e-commerce site may fail to obtain the results of payment confirmation and the stale payment data will remain on Paidy.

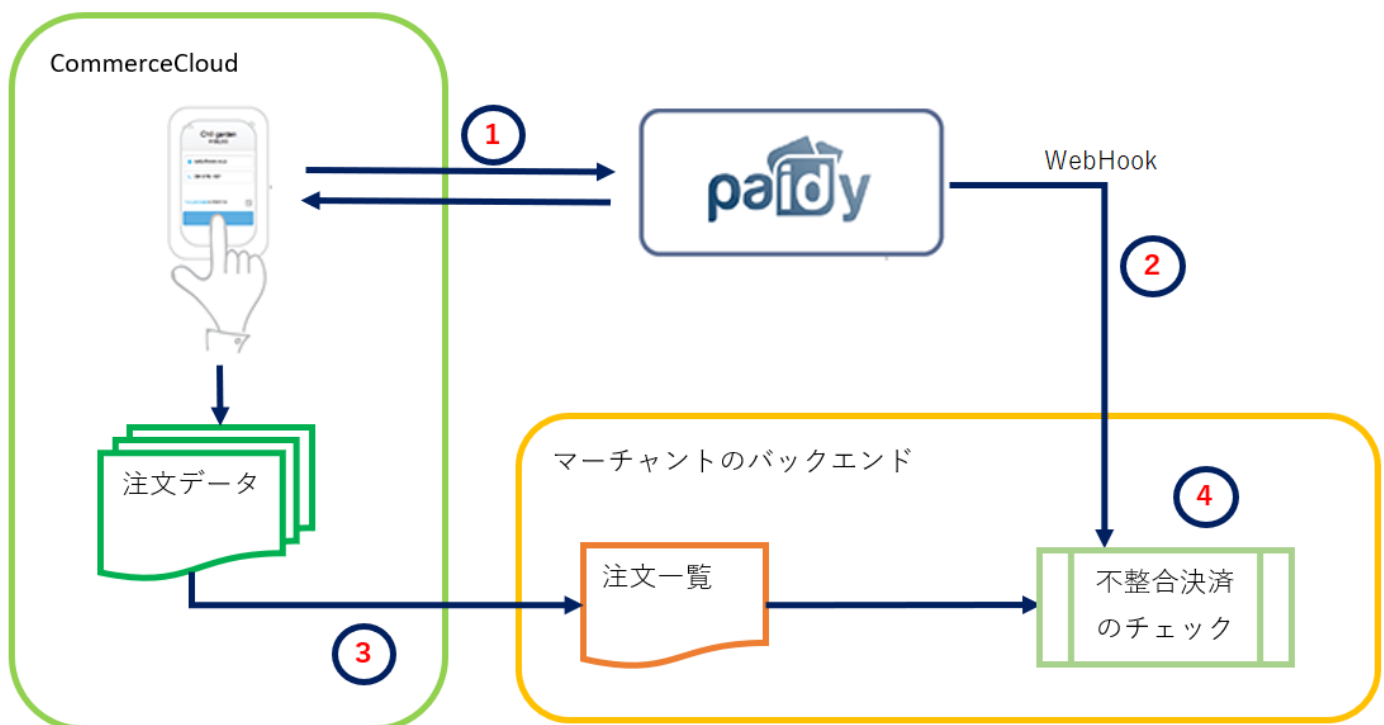
In the event of an inconsistency like the above, there are two problems.

1. The shop loses the opportunity to complete the order.
2. If the order does not have to be completed, the consumer has a credit hold placed against their Paidy account.

To accommodate these cases, a Webhook is provided.

The merchant can use the Webhook to detect inconsistent payments.

5-2. Detecting inconsistent payments



1. Payment

Payment is made via Commerce Cloud using Paidy. A callback from Paidy's payment results returns one of the three order statuses below.

- Payment success: New Outputted to the order list. Payment possible status
- Payment failure: Failed Not outputted to the order list. Payment not possible status
- Processing: Created Not outputted to the order list. When in a payment not possible status, updating the payment transitions to a payable status.

2. Webhook

Setting Webhook endpoints on the merchant backend allows for receiving notifications from Paidy of payment completion.

3. Order list

A list of successful orders processed through batch processing on Commerce Cloud is sent as an xml file to the merchant's backend.

4. Inconsistent payment check processing

Checks for payments where Webhook data (2) is found, but no payment data (3) exists on Commerce Cloud.

5-3. Handling inconsistencies

Inconsistent payments that have been detected can be handled as follows, allowing for resolving the above issues.

5-3-1. Processing example 1

In this solution, a list of inconsistent/invalid payments is outputted and can be checked by the merchant.

After a check is performed, the necessary steps are taken (approving or cancelling payment).

1. Establishing payment

Use the Order function on Business Manager on Commerce Cloud to update the status.

- General tab

General Attributes Payment Notes History

Details for Order 'WBSC00005100'

Information:	Contains 1 line item to 1 shipping location The total price is ¥14,506.00.		
Date Received:	11/24/17 6:27:23 am Etc/UTC		
Site:	Paidy_Master		
Created By:	Customer		
Customer:	純玉水		
Customer No.:	00033017		
IP Address:	n/a		
Email:	jun.tamamizu@wiredbeans.co.jp		
Phone:	022-380-8680		
Order Status:	Open	Confirmation Status:	Confirmed
Shipping Status:	Not Shipped	Export Status:	Ready for Export

The status of the above three points is changed per below

Order Status : Created→Open

Confirmation Status : Not Confirm→Confirmed

Export Status: Not Exported→Ready for Export

- Attribute tab

The screenshot shows the Salesforce Commerce Cloud Merchant Tools interface. The top navigation bar includes the Salesforce logo, the user 'Sandbox - wiredbeans01 Paidy_Master', and tabs for 'Merchant Tools', 'Administration', and 'Store'. The breadcrumb trail is 'Merchant Tools > Ordering > Orders > Order: WBSC00005100(Paidy_Master)'. The 'Attributes' tab is selected, showing the title 'Attributes for Order 'WBSC00005100''. Below the title, a note states: 'On this page you can edit the attributes of the order. Fields with a red asterisk (*) are mandatory. Click Apply to save changes. Click Res...'. The form contains several fields: 'deliveryDate' (with a calendar icon), 'deliveryTimeSpan', 'paymentCode', 'Channel Type' (dropdown), 'Replacement Order Number', 'Cancel Code' (dropdown), 'Shipping Status*' (dropdown, set to '0 (NOTSHIPPED)'), 'Customer Order Reference' (text area), 'Replaced Order No (replacedOrderNo)' (text area, containing 'mizu@wiredbeans.co.jp'), 'Replaced Order No' (text area), 'Replace Code' (dropdown), 'External Order Status', and 'Payment Status*' (dropdown, set to '0 (NOTPAID)'). A large grey box with the text '省略' (Omission) covers the bottom part of the form. Below this, the 'PaidyPayments' section is visible, containing 'paidyToken' and 'paidyPaymentId' (highlighted with a red box, containing 'pay_20171124_001').

Set the paidyPaymentID that arrived via the Webhook for paidyPaymentId

*In order to e-mail an order confirmation e-mail as you would with StoreFront, it must be customized, as the format differs (see below).

- Order confirmation e-mail (concept)

■SiteGenesis StoreFront e-mail (normally used e-mail)

Salesforce Commerce Cloud SiteGenesis	Salesforce Commerce Cloud SiteGenesis 5 Wall Street Burlington, MA 01803 salesforce.com 電話 +1.800.555.0199
---	---

ご注文いただきありがとうございました

ご注文に関してご質問などがございましたら、お電話 (012-345-XXXX) にてご遠慮なくお問い合わせください
(月曜日～金曜日、午前 8 時～午後 8 時)。

注文日: 24-11-2017
注文番号: WBSC00005102

支払情報		
請求先住所 日本 980-0014 宮城県 仙台市青葉区 本町 玉水 純	支払方法 Paidy翌月払い(コンビニ銀行) 金額: ¥ 2,492	支払金額合計 小計 ¥ 1,808 配送 通常配送 ¥ 500 消費税 ¥ 166

■E-mails sent from SendMail button on bottom of General tab on BusinessManager > Orders

SiteGenesis	Demo SiteGenesis 5 Wall Street Burlington, MA 01803 SiteGenesis Phone: +1.888.553.9216
--------------------	---

Order Confirmation: Order Number: WBSC00005101 - Thank you for your order!

This email confirms we received your order at [SiteGenesis](#). We will email your shipping confirmation with your tracking number when your items have been shipped. Your order will generally be shipped within 24 hours after we've received it. You can check your order's status on your account pages at [SiteGenesis](#)

Thank you for shopping with us!

Please print and save a copy for reference.

Please review the order below and verify that everything is correct.

PLEASE NOTE: Since we begin processing your order shortly after you submit it on our web site, if any changes are necessary you **MUST** contact us via PHONE ONLY, at **+1.888.553.9216**

SHIPMENT 1

PRODUCT DETAILS	QTY.	TOTAL	SHIPPING DETAILS
Samsung Series 6 22" LCD High Definition Television Item Number:samsung-ln22a650 全品セール中	1	¥ 66,318 ¥ - 33,159	純 玉水 本町 仙台市青葉区、宮城県 980-0014 通常配送 - ¥ 500

PAYMENT METHOD	AMOUNT
Payment Total	¥ 36,351

2. Cancelling payments

Displaying the transaction in question on the Paidy merchant management screen and Canceling it (do not capture and close).

ホーム

支払管理

レポート

設定

検索

支払詳細

ステータス

AUTHORIZED

取引ID

WBSC00005640

Authorized金額

¥ 5,487 (注文金額: ¥4,581 税金: ¥406 配送: ¥500)

キャプチャー金額

¥0 (注文金額: ¥0 税金: ¥0 配送: ¥0 リファンド: ¥0)

日付

2017-12-01 14:44:40

キャプチャー期限

2017-12-31 14:44:41

ストア名

株式会社ワイヤードビーンズ (サンプル店)

決済ID

pay_WiDsSEoAAFYAdAYR TEST

顧客情報

氏名	フリガナ	メールアドレス	住所
山田 太郎 ヤマダ タロウ		successful.payment@paidy.com	青葉区つみちよう111 仙台市 宮城県 988-8888

オーダー内容

商品番号	商品名	価格	数量	小計
701642843610	ワイドウエストベンシルスカート	¥4,581	1	¥4,581

キャプチャーせずにクローズ

キャプチャー

5-3-2. Processing example 2

In this example, the system makes a determination and automatically cancels inconsistent payments.

If, after referencing the order data and Webhook data on Commerce Cloud, the payment does not complete on Commerce Cloud, and authorization data is found only on Paidy, the close API is requested and the payment is closed.

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

5-4. Webhook settings

In order to receive notifications from Paidy on the backend, the Webhook URL must be saved to the Paidy merchant management screen settings.

URL: <https://merchant.paidy.com>

You can enable these settings for a live environment by setting the merchant's live backend URL in the Webhooks: Live API Key and Old Version API Access Key sections.

In order to check connectivity with test payments before configuring the live environment, set the merchant's backend test URL in Webhooks: Test API Key.

Paidy merchant management screen

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

☒ Webhookの設定 (※オフにすると現在設定されているWebhook URLは削除されます)

Webhook URL

現在設定されているWebhook URL

設定

Webhooks - 「テスト用APIキー」

☒ テスト用Webhookの設定 (※オフにすると現在設定されているWebhook URLは削除されます)

テスト用Webhook URL

現在設定されているテスト用Webhook URL

更新

6. What to do when the service is down

- what happens when the service is down

If you select Paidy payment, you will get an error screen or the pop-up will stop working.

- what the merchant should do to fix it

To get started, please contact Paidy Technical Support.

Paidy Technical Support

Mail: tech-support@paidy.com

TEL : 03-5545-5099 (weekdays 10:00-18:00)

7. Revision history

Version	Date	Revisions
1.1.0	2019/12/06	First edition
20.1.0	2020/01/15	3-2-2. Importing metadata

		-Modify 「meta_data_19.1.0.zip」 to 「meta_data_20.1.0.zip」
20.1.0	2020/4/5	<ul style="list-style-type: none"> -SFRAのバージョン情報を追記。 -Business Managerの互換モードを記載。 -Add Version of SFRA and Compatibility Mode -Add 6. What to do when the service is down
21.1.0	2021/6/21	<ul style="list-style-type: none"> -Update 1. Cartridges version update -Update 4. SFRA support version update -Update 16-17. int_paidy post script -Update 18. Import meta file name update Deleted description related to SiteGenesis and common meta file -Update 25. Corrected the description of function selected Payment Instrument - Update 30. Embedded code fix - Update 31. Embedded code fix - Update 32. File name Change confirmationEmail.isml to confirmationPaymentInfo.isml -Removed 35. Deleted the description of package.json