DGFT
DG Financial Technology

mpos エムポス

mPOS SDK 開発ガイド

（iOS 版）

Ver.1.9.0

目次

第1章　本ガイドについて

1-1 本ガイドの内容

mPOS はお手持ちのスマートフォン・タブレット端末で簡単にクレジット決済やコード決済などができる株式会社 DG フィナンシャルテクノロジー（以下、DGFT）が提供するスマホ決済サービスです。また、mPOS SDK は mPOS が持つ決済に関する機能を提供するライブラリです。加盟店様はこの mPOS SDK を加盟店様が開発されるスマホアプリに組み込むことで、加盟店様が開発されるアプリから mPOS SDK を介して決済処理が可能となります。

mPOS SDK にて、クレジットカードやバーコードの読み取り、IC・磁気情報の暗号化、mPOS 決済サーバーへの決済情報送信を行うため、加盟店様は mPOS SDK をご利用していただくことにより、簡単かつ迅速にお支払い・返金関連の機能を構築することが可能です。尚、クレジットカードの読み取りを行う際は、mPOS SDK がサポートするカードリーダーをご利用ください。

本ガイドでは mPOS SDK をご利用いただくに当たり開発に必要な内容について記載しています。

1-2 対象者

本ガイドは mPOS SDK を加盟店様アプリに組み込むために必要な内容を記載した開発者向けのガイドであり、iOS アプリ開発を行う開発者を対象としています。

1-3 著作権、および問い合わせ先

[お問い合わせ先]
mPOS テクニカルサポート
電子メール：support-mpos@veritrans.jp

第2章　開発ガイドライン

2-1 mPOS SDK 概要

mPOS SDK は DGFT が提供するライブラリです。加盟店様アプリへ mPOS SDK を組み込むことで決済機能などの mPOS が提供する機能を利用いただくことができます。

mPOS SDK を組み込むことで利用することができる機能は次の表（API 概要一覧）の通りです。

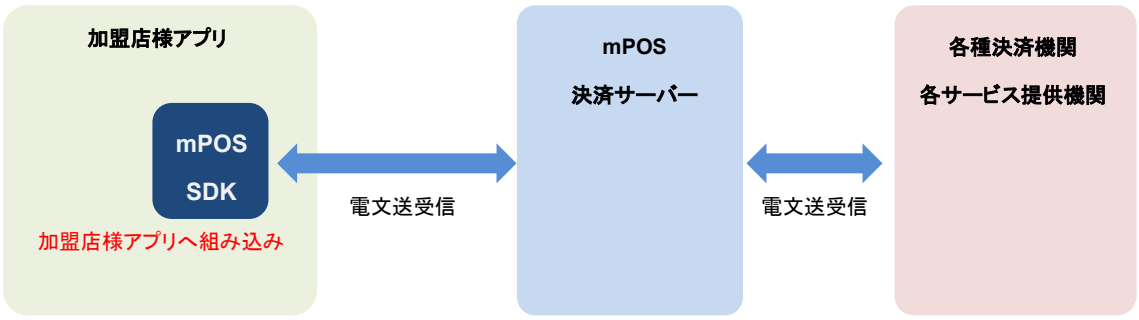

図　1 mPOS SDK におけるアプリ・システム連携フロー

表　1 API 概要一覧

API 名	説明
初期化 API	mPOS SDK を初期化するための API です。他の API を実行する前に1度だけ実行してください。 ※ 初期化 API を実行するタイミングとしては、アプリ起動時に実行していただくことを推奨します。
決済 API	加盟店様アプリから決済に必要な金額等の情報を引き渡し、mPOS SDK を介して決済・マイル積算を行う機能です。
取消 API	売上が成立していない指定した mPOS 取引 ID に紐付く取引に対して取消を行う機能です。 ※ mPOS 取引 ID とは、決済処理時に mPOS サービス内で発番される取引を一意に特定するための ID になります。 ※ 取引が売上未成立の状態で一定時間経過すると mPOS 決済サーバーにて取消処理が実施されます。この取消機能は mPOS 決済サーバーが取消を実施する前に取り消したい場合にご利用ください。
返金 API	指定した mPOS 取引 ID に紐付く取引に対して返金（売上取消）・マイル積算取消を行う機能です。 ※ mPOS 取引 ID とは、決済処理時に mPOS サービス内で発番される取引を一意に特定するた

	めの ID になります。 ※ 仮売上状態の取引を取り消す場合もこの返金 API をご利用ください。 ※ 取引が仮売上の状態で一定時間経過すると mPOS 決済サーバーにて返金処理が実施されます。
取引検索 API	指定した検索条件に合致する取引を検索する機能です。 ただし、この取引検索 API で取得できる取引は、当該ユーザーが実施した取引に限ります。 ※ 同一加盟店に属する別のユーザーが実施した取引は検索することはできません。
取引集計 API	指定した集計条件に合致する売上・返金・マイル積算・マイル積算取消された取引を集計する機能です。ただし、この取引集計 API で集計できる取引は、当該ユーザーが実施した取引に限ります。 ※ 同一加盟店に属する別のユーザーが実施した取引は集計することはできません。 ※ 仮売上状態の取引は売上取引として集計されません。 ※ 仮売上取消状態の取引は返金取引として集計されません。
メール送信 API	レシートや返金受付通知を指定したメールアドレスに送信する機能です。
パスワード変更 API	mPOS サービスに登録されたユーザーのパスワードを変更する機能です。
カードリーダーアプリ 更新 API	ご利用のカードリーダーにインストールされたアプリケーションを更新する機能です。 ※ この機能が対象とするカードリーダーは PIN パッドありカードリーダーに限ります。 ※ ご利用のカードリーダーにインストールされたアプリケーションがこの機能に対応していない場合は、この機能をご利用できません。

2-2 推奨環境

mPOS SDK をご利用いただくに当たり、推奨します動作・開発環境は次の通りです。

尚、mPOS SDK は iOS シミュレーターには対応していません。そのため、ビルドや動作確認を行うためには実機（iOS 端末）が必要となります。

- 動作 OS：iOS 11.0 以上
- 開発言語：Objective-C、または Swift 5.6 - 5.7
- 統合開発環境（IDE）：Xcode 13.4.x - 14.0.x
- カードリーダー：クレジットカード決済、NFC 決済、銀聯決済をご利用いただく場合は mPOS SDK がサポートするカードリーダーが必要となります。尚、銀聯決済を行う場合は PIN パッドありカードリーダーが必要となります。PIN パッドなしカードリーダーについては銀聯決済に対応していません。

2-3 プロジェクト設定について

2-3-1 依存ライブラリ

次のライブラリ（Framework）を加盟店様アプリのプロジェクトに追加してください。

尚、加盟店様アプリプロジェクトに次のライブラリを追加する際は、”Frameworks, Libraries, and Embedded Content”のセクションから追加してください。また、”Embed”の設定としては、”Embed & Sign”を設定してください。

1） mPOS SDK .framework　(mPOS SDK 本体)

2） Alamofire.framework

※Alamofire は Swift で書かれた HTTP ネットワークライブラリです。

（URL: https://github.com/Alamofire/Alamofire）

3） MastercardSonic.framework

※決済時に効果音を出力するために必要となるライブラリです。

2-3-2 ビルド設定

プロジェクトのビルド設定（Build Settings）については、次のように設定してください。

1） Deployment セクションの「iOS Deployment Target」には、”iOS 11.0”以上を設定してください。

2） Architectures セクションの「Architectures」には、”Standard architectures”を設定してください。

3） Linking セクションの「Other Linker Flags」には、”-all _load”と”–ObjC”を設定してください。

4） Search Paths セクションの「Framework Search Path」には、mPOS SDK と Alamofire のライブラリが格納されているフォルダパスを設定してください。

5） Objective-C のプロジェクトの場合、Build Options セクションの「Always Embed Swift Standard Libraries」には、”Yes”を

設定してください。

2-4 Info.plist の設定

加盟店様アプリにある Info.plist に次の内容を設定してください。

1）プライバシー設定

mPOS SDK では、カメラ、マイク、Bluetooth を使用（アクセス）するため、これらを使用する目的を Info.plist ファイルに記述する必要があります。これらの設定しない場合、使用しようとしたときにアプリが異常終了しますので設定してください。また、Info.plist に設定した文言については、カメラ、マイク、Bluetooth の初回利用した時にユーザーに対してアクセス許可を確認するためのダイアログ上に表示されます。

※設定値については「表２ Info.plist 追加項目」を参照してください。

[Info.plist への追加設定項目]

- “Privacy - Bluetooth Always Usage Description” : プリンターやカードリーダーと Bluetooth 接続するために必要となります。
- “Privacy - Bluetooth Peripheral Usage Description” : プリンターやカードリーダーと Bluetooth 接続するために必要となります。
- “Privacy - Camera Usage Description” : カメラで QR コードやバーコードを読取るために必要となります。
- “Privacy - Microphone Usage Description” : PIN パッドなしカードリーダーを使用するために必要となります。

2）カードリーダーデバイスを使用するための設定

カードリーダーデバイスを使用するための設定として Info.plist に次の項目を追加してください。尚、”Supported external accessory protocols”項目については、PIN パッドありカードリーダーをご利用いただく場合にのみ設定してください。

※設定値については「表２ Info.plist 追加項目」を参照してください。

※”Supported external accessory protocols”を設定される場合は、「6-1-1 PIN パッドありカードリーダーをご利用いただく場合」の内容もご確認ください。

[Info.plist への追加設定項目]

- “Required background modes”
- “Supported external accessory protocols” : PIN パッドありカードリーダーをご利用される場合にのみ設定。

3）回転制御に関する設定

mPOS SDK にて画面の回転が発生する場合があるため、回転を許可するために次のように設定してください。

※設定値については「表２ Info.plist 追加項目」を参照してください。

[iPnone の場合の Info.plist への追加設定項目]

・“Supported interface orientations”に対して”Landscape (left home button)”、”Landscape (right home button)”、”Portrait (top home button)”を追加してください。

[iPad の場合の Info.plist への追加設定項目]

・“Supported interface orientations (iPad)” に対して ”Landscape (left home button)” 、”Landscape (right home button)”、”Portrait (top home button)”を追加してください。

表 2 Info.plist 追加項目

Key	Type	Value Type	Value
Privacy - Bluetooth Always Usage Description	String	-	Bluetooth を使用する目的・理由を設定してください。 ※設定例：カードリーダーなどの外部機器と接続するために必要となります。
Privacy - Bluetooth Peripheral Usage Description	String	-	Bluetooth を使用する目的・理由を設定してください。 ※設定例：“カードリーダーなどの外部機器と接続するために必要となります。”
Privacy - Camera Usage Description	String	-	カメラを使用する目的・理由を設定してください。 ※設定例：“QR コードやバーコードを読取るために必要となります。”
Privacy - Microphone Usage Description	String	-	マイクを使用する目的・理由を設定してください。 ※設定例：“カードリーダーと接続するために必要となります。”
Required background modes	Array	String	App communicates with an accessory
Supported external accessory protocols	Array	String	com.castles.protocolCastles
Supported interface orientations	Array	String	Landscape (left home button)
Supported interface orientations (iPad)	Array	String	Landscape (left home button)
		String	Landscape (right home button)

2-5 決済 API 概要

mPOS SDK が提供する決済 API を加盟店様アプリから実行することにより決済処理・マイル積算処理を行います。決済 API 実行後の処理については、mPOS SDK が提供します画面に従い画面操作を行ってください。処理結果については処理終了後に決済 API のレスポンスとして加盟店様アプリに返却します。

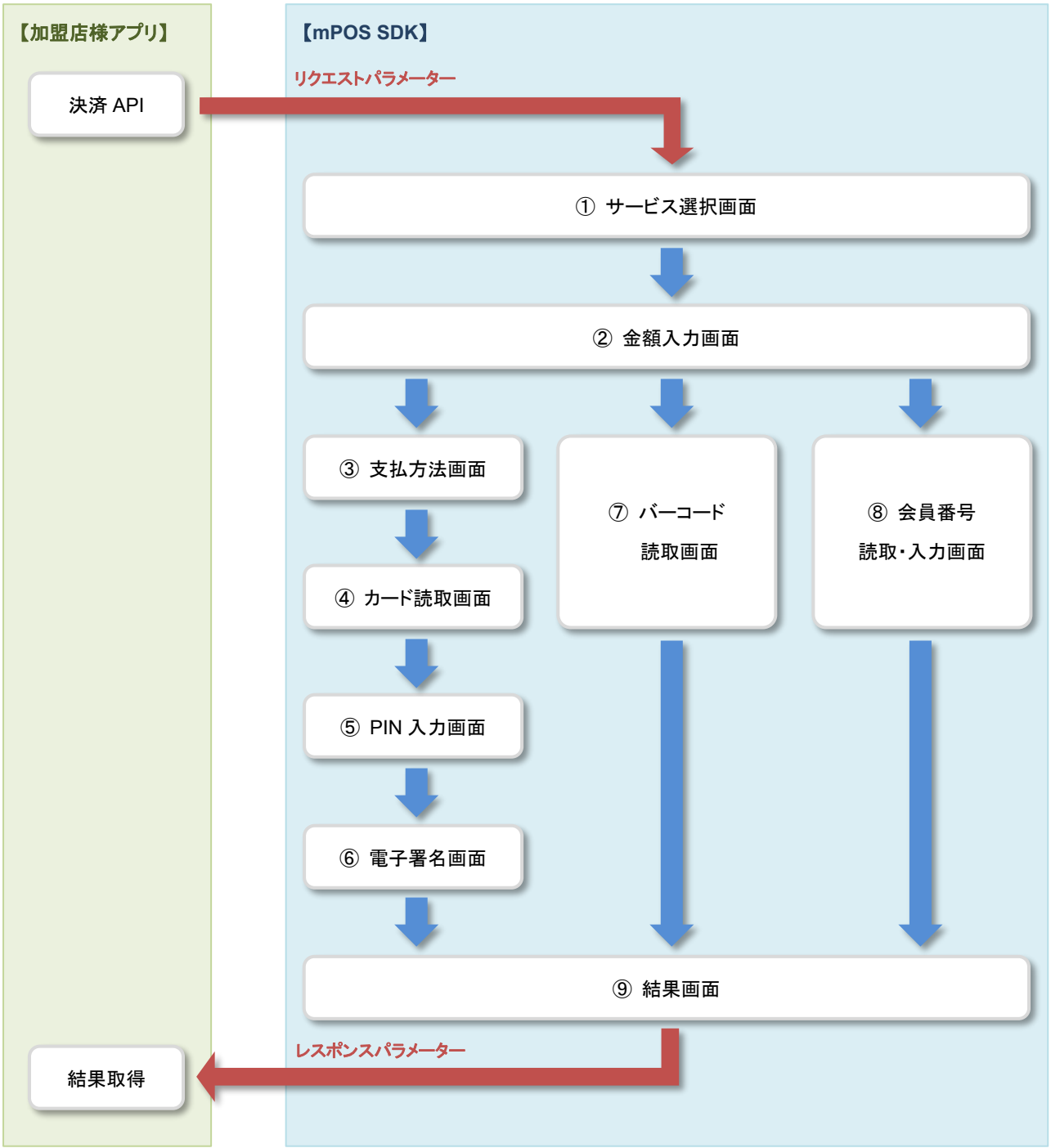

図 2 決済処理フロー

① サービス選択画面

クレジットカード決済など利用するサービスを選択します。利用可能なサービスについては mPOS サービスのご契約状況によって異なり、ご契約状況によっては本画面がスキップされることがあります。また、mPOS SDK のリクエストパラメーターにて決済方法（サービス）を指定した場合は本画面がスキップされます。

② 金額入力画面

取引に必要な「金額」と「説明」を設定します。mPOS SDK のリクエストパラメーターで設定した「金額」と「説明」が画面に表示されますが、この画面にて内容の変更することができます。尚、金額入力画面をスキップするように mPOS SDK のリクエストパラメーターを設定した場合は本画面がスキップされます。

③ 支払方法画面 （クレジットカード決済、NFC 決済）

利用代金の支払方法を選択します。支払方法として「一括払い」、「分割払い」、「ボーナス一括」、「ボーナス月指定」、「リボ払い」を選択することができます。ただし、選択できる支払方法は mPOS サービスのご契約状況によって異なります。また、支払方法が1つしか選択できない場合は本画面がスキップされます。

④ カード読取画面 （クレジットカード決済、NFC 決済、銀聯決済）

消費者様が保持するカード（クレジットカードなど）の読み取りを行う画面です。カードの読み取りについては mPOS SDK がサポートするカードリーダーにて行ってください。

⑤ PIN 入力画面 （クレジットカード決済、NFC 決済、銀聯決済）

消費者様が保持するカードの暗証番号を入力する画面です。暗証番号の入力については mPOS SDK がサポートするカードリーダーにて行ってください。尚、取引を行うに当たり消費者様が提示されたカードの暗証番号の入力が不要の場合は本画面がスキップされます。

⑥ 電子署名画面 （クレジットカード決済、NFC 決済、銀聯決済）

消費者様に署名を行っていただく画面です。尚、署名が不要の場合は本画面がスキップされます。

⑦ バーコード読取画面 （コード決済）

消費者様が提示したバーコードの読み取りを行う画面です。消費者様が提示したバーコードの自動識別を行いコード決済方法の指定なく決済を行うことができます。また、自動識別ではなく加盟店（ユーザー）側でコード決済方法を指定することも可能です。mPOS サービスがサポートするコード決済の種類については「3-11 決済方法コード」をご参照ください。

⑧ 会員番号読取・入力画面 （ANA マイル積算、ユナイテッド航空マイル積算、その他マイル積算）

消費者様が提示した会員番号の読み取り、もしくは入力を行う画面です。次の方法で消費者様の会員番号の読み取り、もしくは入力を行います。

1. カードからの読み取り （※ANA マイル積算のみ）

mPOS SDK がサポートするカードリーダーを用いて消費者様が提示したカードから会員番号を読み取ります。

2. バーコードからの読み取り （※ANA マイル積算のみ）

消費者様が提示したバーコードから会員番号を読み取ります。

3. 手入力

消費者様の会員番号を画面上で手入力を行います

⑨ 結果画面

実行した取引の成否を表示する画面です。尚、完了・失敗画面をスキップするように mPOS SDK API のリクエストパラメーターを設定した場合は、本画面はスキップされ加盟店様アプリに戻ります。

2-6 取消 API 概要

mPOS SDK が提供する取消 API を加盟店様アプリから実行することにより取消処理を行います。処理を行う場合は、対象となる取引を特定する ID（mPOS 取引 ID）を指定することにより処理を行うことができます。処理結果については処理終了後に取消 API のレスポンスとして加盟店様アプリに返却します。

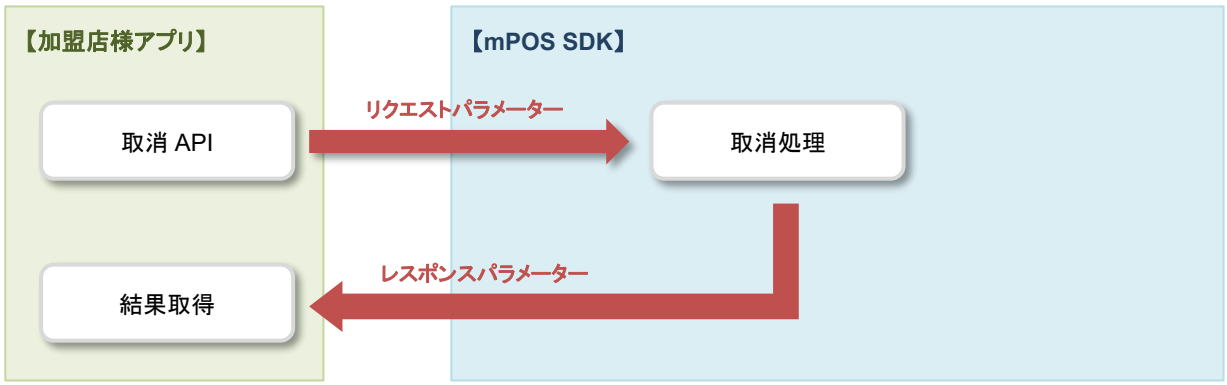

図 3 取消処理フロー

2-7 返金 API 概要

mPOS SDK が提供する返金 API を加盟店様アプリから実行することにより返金（売上取消）処理・マイル積算取消処理を行います。処理を行う場合は、対象となる取引を特定する ID（mPOS 取引 ID）を指定することにより処理を行うことができます。処理結果については処理終了後に返金 API のレスポンスとして加盟店様アプリに返却します。

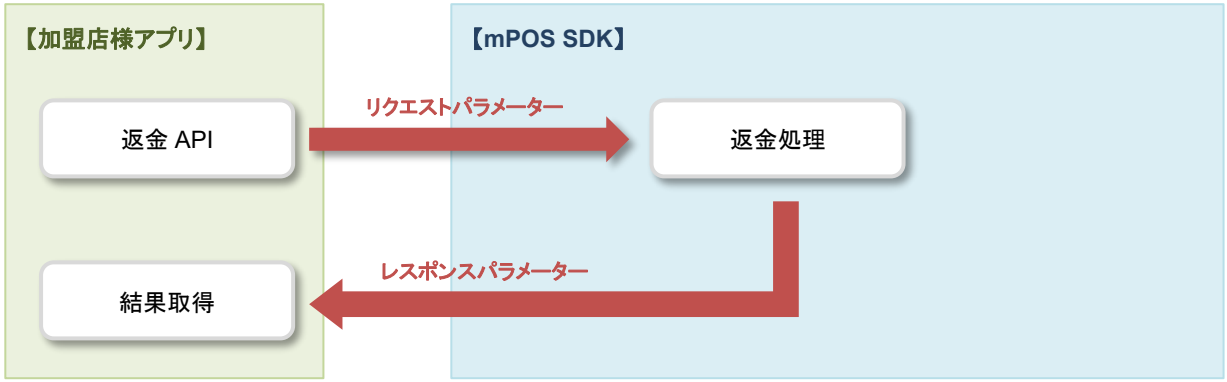

図 4 返金処理フロー

2-8 取引検索 API 概要

mPOS SDK が提供する取引検索 API を加盟店様アプリから実行することにより取引検索処理を行います。

取引検索処理では、検索条件を指定し実行することで、その条件に該当する取引に関する情報を取引検索 API のレスポンスとして取得することができます。尚、mPOS SDK が提供する取引検索 API で検索できる取引は、当該ユーザーが実施した取引に限ります。そのため、同一加盟店に属する他のユーザーが実施した取引は検索することはできません。

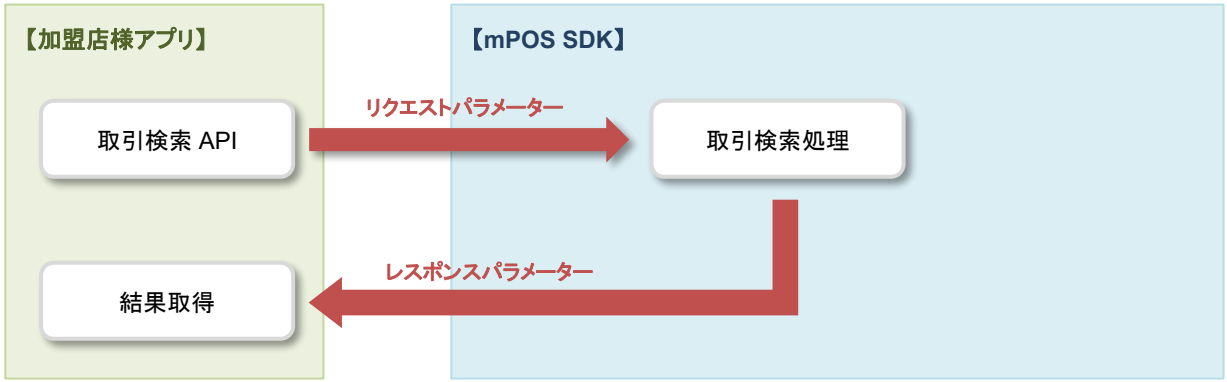

図 5 取引検索処理フロー

2-9 取引集計 API 概要

mPOS SDK が提供する取引集計 API を加盟店様アプリから実行することにより取引集計処理を行います。

取引集計処理では、集計条件を指定し実行することで、その条件に該当する売上・返金・マイル積算・マイル積算取消処理された取引の件数と金額の集計を行い、取引集計 API のレスポンスとして取得することができます。尚、mPOS SDK が提供する取引集計 API で集計できる取引は、当該ユーザーが実施した取引に限ります。そのため、同一加盟店に属する他のユーザーが実施した取引は集計結果に含まれません。

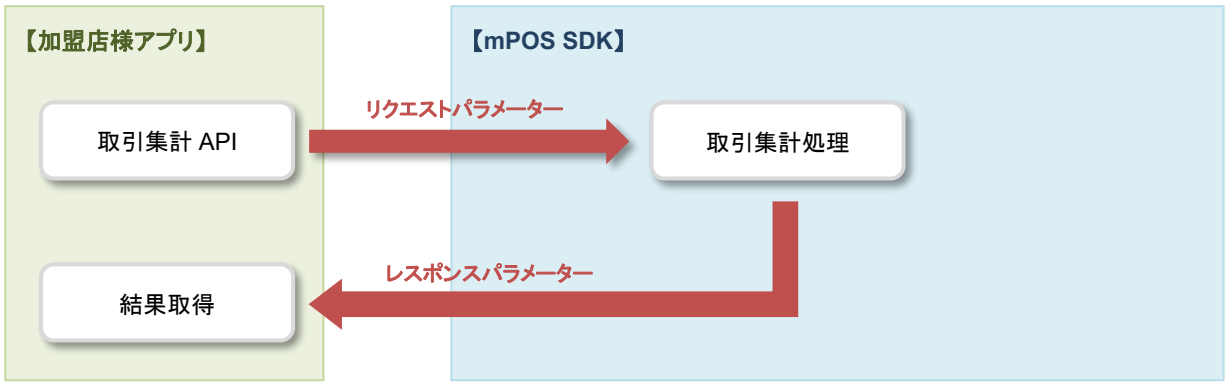

図 6 取引集計処理フロー

2-10 メール送信 API 概要

mPOS SDK が提供するメール送信 API を加盟店様アプリから実行することにより、指定されたメールアドレスに向けてメールを送信します。尚、このメール送信 API では、消費者様向けにレシートメールや返金受付通知メールを送信します。

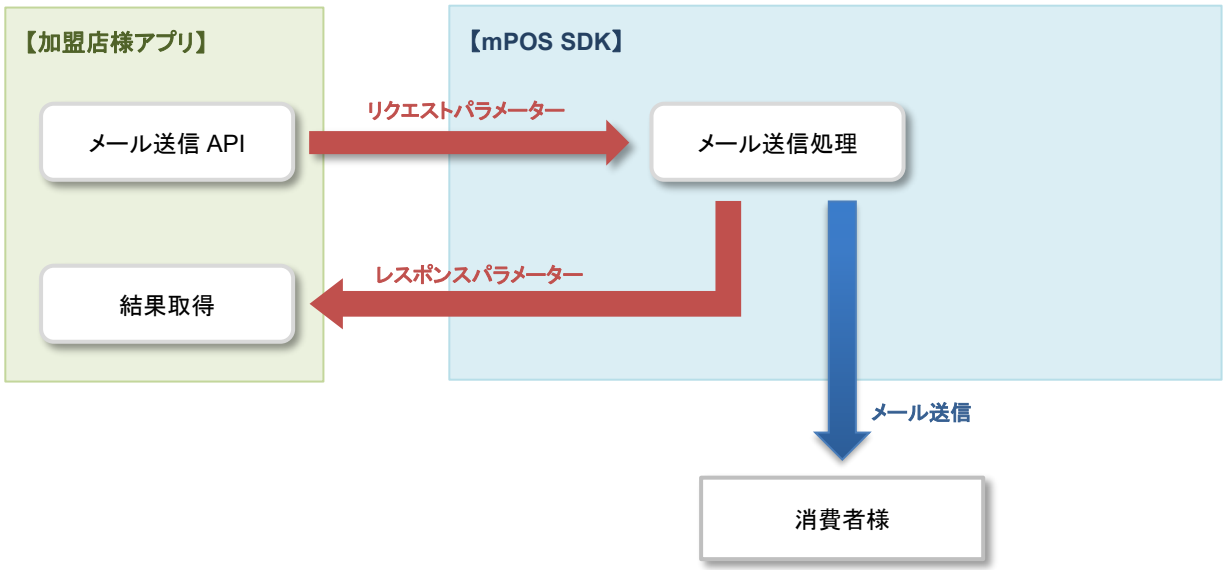

図 7 メール送信処理フロー

2-11 パスワード変更 API 概要

mPOS SDK が提供するパスワード変更 API を加盟店様アプリから実行することにより、ユーザーのパスワードを変更することができます。

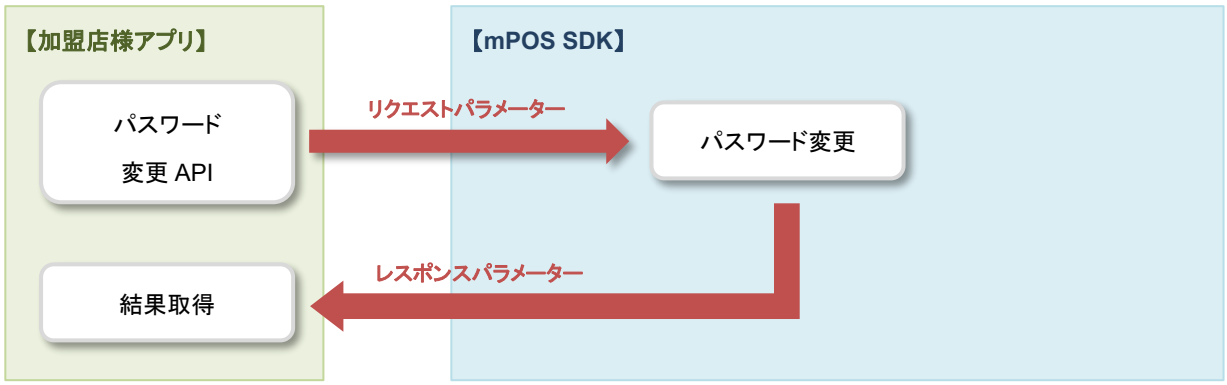

図 8 ユーザーパスワード変更処理フロー

2-12 カードリーダーアプリ更新 API 概要

mPOS SDK が提供するカードリーダーアプリ更新 API を加盟店様アプリから実行することにより、ご利用のカードリーダーにインストールされたアプリケーションを更新することができます。

この機能が対象とするカードリーダーは PIN パッドありカードリーダーに限ります。また、ご利用のカードリーダーにインストールされたアプリケーションがこの機能に対応していない場合は、この機能をご利用できません。

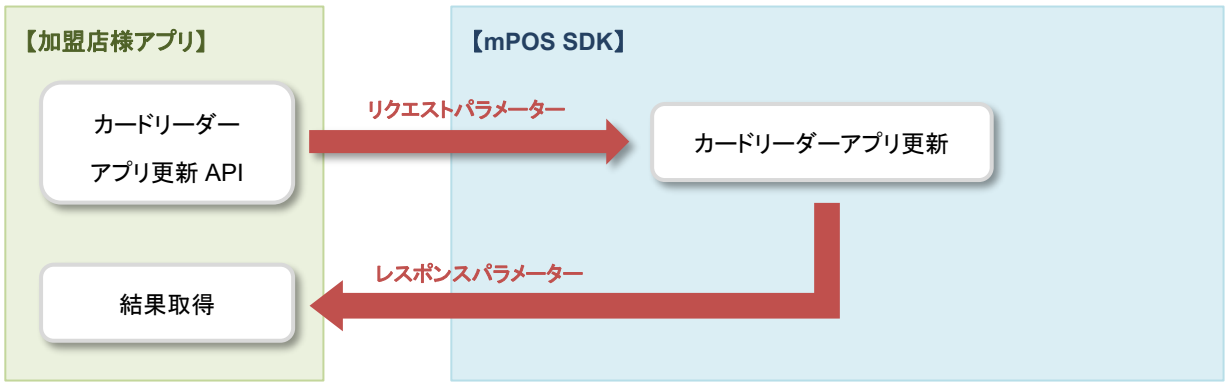

図 9 カードリーダーアプリ更新処理フロー

第3章　インターフェース詳細

本章では各機能にて使用する API について説明します。

3-1 API 共通

3-1-1 API の実行方法について

mPOS SDK が提供する各 API は MposHandler クラスのインスタンスメソッドとして定義されています。そのため、各 API を実行する場合は、MposHandler クラスのインスタンスを生成した後に API を実行してください。

ただし、例外として初期化 API については MposHandler のクラスメソッドとして定義されています。

3-1-2 パラメーターの形式

パラメーターの種類として「共通ヘッダーパラメーター」、「リクエストパラメーター」、「レスポンスパラメーター」が存在します。これらのパラメーターは全て辞書形式（Swift では Dictionary、Objective-C では NSDictionary）として定義するため、各要素はキーと値で構成されます。

尚、各パラメーターで使用する要素のキーは MposRequestParam クラスと MposResponseParam に静的変数として各要素のキーを定義されていますので、その静的変数を要素のキーとして使用してください。

※パラメーターの設定例

■Swift の場合

```
let params: [MposRequestParam: String] = [
    MposRequestParam.userId : “test_user@xxxxxx.co.jp”,
    MposRequestParam.userPassword: “12345678”
]
```

■Objective-C の場合

```
NSDIctionary *headerParams = @{
    MposRequestParam.userId : @”test_user@xxxxxx.co.jp”,
    MposRequestParam.userPassword: @”12345678”
}
```

3-1-3 処理結果（レスポンスパラメーター）の取得方法について

初期化 API を除く各 API の処理結果は、API の引数にクロージャーを設定することで受け取ることができます。API を実行し処理が終了すると、API の引数に設定したクロージャーが処理結果を持って呼び出されます。

3-1-4 共通ヘッダーパラメーター

共通ヘッダーパラメーターとは各 API に対して共通に定義されるパラメーターです。ただし、例外として初期化 API については共通ヘッダーパラメーターを必要としません。

共通ヘッダーパラメーターとして定義する内容は次の通りです。

No.	パラメーターキー	必須	書式・制限	説明
1	userId （ユーザーID）	○	文字列	mPOS サービスに接続するユーザーの ID を設定します。
2	userPassword （ユーザーパスワード）	△	文字列	mPOS サービスに接続するユーザーのパスワードを設定します。 ※ “userPassword”もしくは” merchantPassword”のどちらかが設定されている必要があります。 ※ ユーザーパスワードとマーチャントパスワードの両方が設定されている場合は、ユーザーパスワードが優先して使用されます。
3	merchantPassword （マーチャントパスワード）	△	文字列	ユーザーが所属するマーチャントに対して発行されたマーチャントのパスワードを設定します。 ※ “userPassword”もしくは” merchantPassword”のどちらかが設定されている必要があります。 ※ ユーザーパスワードとマーチャントパスワードの両方が設定されている場合は、ユーザーパスワードが優先して使用されます。 ※ マーチャントパスワードについては「6-2 マーチャントパスワードとは」を参照してください。
4	readerType （カードリーダータイプ）	△	半角数字	決済処理時に使用するカードリーダーを設定します。 ※ 決済機能 API に限り必須項目となります。 ※ クレジットカード決済・NFC 決済をご利用されない場合でも決済機能 API をご利用の際は、いずれかの値を設定してください。 “iPOS”： PIN パッドなしカードリーダー “ctMP200”： PIN パッドありカードリーダー
5	locale （言語情報）	△	半角英字	mPOS SDK が提供する画面で適用される表示言語を設定します。未設定の場合は日本語が設定されたものとして扱います。 “ja”： 日本語 “en”： 英語

				“zh”： 中国語 “vi”： ベトナム語 “in”： インドネシア語
6	cardThreeWay （カード決済3面待ち）	△	半角数字	クレジットカード決済と NFC 決済を統合し、クレジットカード決済にて NFC 決済を扱えるようにするかを設定します。 ※ カードリーダータイプとして PIN パッドなしカードリーダーを設定した場合は設定値に関係なく”0”が設定されたものとして扱われます。 ※ 値が未設定の場合は”0”が設定された場合と同じ扱いになります。 “0”： クレジットカード決済と NFC 決済を統合しない。 “1”： クレジットカード決済と NFC 決済を統合する。
7	cameraFacing （カメラ向き）	△	半角数字	mPOS SDK で利用するカメラを設定します。 ※ 値が未設定の場合は”0”が設定された場合と同じ扱いになります。 “0”： メイン（リア）カメラ “1”： イン（フロント）カメラ

※凡例： “〇”：必須、 ”△”：任意、 ”-”：対象外

3-2 初期化 API

この API は mPOS SDK を初期化する API です。mPOS SDK をご利用するに当たり最初に1度だけ実行してください。

尚、初期化 API を実行するタイミングとしては、加盟店様アプリ起動する際に実行していただくことを推奨します。

3-2-1 API 定義

MposHandler のクラスメソッド（initSdk）として定義されています。また、初期化 API には引数、戻り値は存在しません。

■API 定義：Swift の場合

```
class func initSdk()
```

■API 定義：Objective-C の場合

```
+(void) initSdk;
```

3-2-2 リクエストパラメーター

初期化 API にはリクエストパラメーターは存在しません。

3-2-3 レスポンスパラメーター

初期化 API にはレスポンスパラメーターは存在しません。

3-2-4 API 実行例

■Swift の場合

```
import MPOS SDK
    :
// 初期化 API を実行する。
MposHandler.initSdk() -> Void
```

■Objective-C の場合

```
#import <MPOS SDK/MPOS SDK-Swift.h>
    :
// 初期化 API を実行する。
[MposHandler initSdk];
```

3-3 決済 API

この API はクレジットカード決済や NFC 決済などの決済処理を行うための API です。この API を実行すると画面制御が加盟店様アプリから mPOS SDK に一時的に移ります。そのため、加盟店様アプリからは mPOS SDK が提供する画面自体を制御することはできません。処理結果については、決済 API の引数に設定したクロージャーが決済処理終了時に呼び出され、そのクロージャーの引数に処理結果が設定されています。

尚、決済 API の処理が終了した後の画面表示については、決済 API 呼び出し前の状態に戻ります。

3-3-1 API 定義

MposHandler クラスのインスタンスメソッド（startPayment）として定義されています。

API の引数としては次の通りです。

No.	引数名	必須	型	説明
1	headerParams （共通ヘッダーパラメーター）	○	[MposRequestParam: String]	「3-1-4 共通ヘッダーパラメーター」を参照。
2	requestParams （リクエストパラメーター）	○	[MposRequestParam: String]	「3-3-2 リクエストパラメーター」を参照。
3	complete （処理終了通知クロージャー）	△	(([MposResponseParam: AnyObject]) -> Swift.Void)?)	引数に Dictionary が定義されたクロージャー。 API の処理が終了する際に設定したクロージャーが呼び出されます。呼び出されたクロージャーの引数には処理結果（レスポンスパラメーター）が設定されていますので、その引数から処理結果を受け取ってください。 尚、クロージャーを設定しない場合（nil の場合）については、処理結果が返却されないまま API の処理が終了します。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

■API 定義：Swift の場合

```
func startPayment(headerParams: [MposRequestParam: String],
                  requestParams: [MposRequestParam: String],
                      complete: (([MposResponseParam: AnyObject]) -> Swift.Void)?
                 ) -> Void
```

■API 定義：Objective-C の場合

```
- (void)startPayment:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)headerParams
      requestParams:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)requestParams
           complete:(void (^ _Nullable)(NSDictionary<MposResponseParam *, id> * _Nonnull))complete;
```

3-3-2 リクエストパラメーター

パラメーターキーは MposRequestParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	amount （金額）	△	半角数字 最大 8 桁	金額を設定します。 ※ 設定可能な値は 1 以上、99,999,999 以下です。 ※ 設定値にカンマを含めることはできません。 ※ 金額入力画面表示フラグが表示”0”の場合は必須項目となります。
2	orderDescription （説明）	△	文字列 最大 40 文字	取引に対する説明を設定します。 ※ Alipay、WeChat Pay、LINE Pay、d 払い、PayPay 決済をご利用される場合は必須項目となります。
3	inputAmountView （金額入力画面表示フラグ）	△	半角数字 最大 1 桁	支払いを受付ける画面の表示を行うかを設定するためのフラグです。 ※ 値が未設定の場合は非表示”0”が設定された場合と同じ扱いとなります。 非表示：”0”、 表示：”1”
4	completeView （完了画面表示フラグ）	△	半角数字 最大 1 桁	完了・失敗画面の表示を行うかを設定するためのフラグです。 ※ 値が未設定の場合は非表示”0”が設定された場合と同じ扱いとなります。 非表示：”0”、 表示：”1”
5	paymentMethodCode （決済方法コード）	△	半角英数字 3 桁	決済方法（サービス）を設定します。未設定、且つ利用可能な決済方法が複数存在する場合は、決済方法を選択する画面が表示されます。 ※ 決済方法コードとして定義されているコード値は「3-11 決済方法コード」を参照

※凡例：“〇”：必須、 ”△”：任意、 ”-”：対象外

3-3-3 レスポンスパラメーター

レスポンスパラメーターは API の引数に指定した処理終了通知クロージャーが呼び出された際に、そのクロージャーの引数に設定されます。また、パラメーターキーは MposResponseParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	status （処理ステータス）	○	文字列	“success”： 正常終了 “failure”： 異常終了 “cancel”： キャンセル終了（途中離脱）
2	errorCode （エラーコード）	△	文字列	エラー内容を示すコード ※ 処理が失敗した場合に設定されます。
3	errorMessage （エラーメッセージ）	△	文字列	エラーコードメッセージ ※ 処理が失敗した場合に設定されます。
4	resultCode （結果コード）	△	半角数字 最大 10 桁	処理結果を示すコードで、主にエラーコードの補足するコードが設定されます。
5	orderId （mPOS 取引 ID）	△	半角英数字 6 桁	mPOS サービスにて取引を一意に特定する ID
6	orderReferenceId （mPOS 取引照会 ID）	△	文字列 最大 64 文字	mPOS サービスにて取引を照会するための ID
7	paymentMethodCode （決済方法コード）	△	半角英数字 3 桁	決済方法（サービス）を示すコード ※ 定義される値は「3-11 決済方法コード」を参照
8	paymentSource （支払手段）	△	半角英数字 最大 64 桁	決済時に利用された支払いの手段を示す情報 ※ 定義される値は「3-12 支払手段」を参照
9	amount （金額）	△	半角数字 最大 8 桁	取引を行った金額 ※ 金額にはカンマは含みません。
10	discountAmount （割引金額）	△	半角数字 最大 8 桁	クーポン等の利用で割引された金額 ※ 金額にはカンマは含みません。
11	totalAmount （合計金額）	△	半角数字 最大 8 桁	取引の合計金額 ※ 金額にはカンマは含みません。
12	paymentTime （取引日時）	△	半角数字 14 桁	決済処理が行われた最終日時（日本時間） ※ 書式：yyyyMMddHHmmss
13	orderDescription （説明）	△	文字列 最大 40 文字	取引に対する説明
14	cardNumber （カード番号）	△	文字列 最大 19 桁	先頭6桁、末尾4桁以外がマスクされたカード番号
15	cardNumberToken （カード番号トークン）	△	文字列 最大 19 桁	先頭6桁、末尾4桁以外がマスクされたカード番号トークン

16	cardExpiry （カード有効期限）	△	文字列 最大 5 桁	年月部分がマスクされたカードの有効期限
17	cardType （カードタイプ）	△	半角英数字 2 桁	カードの取引種別を示すコード “MS”：磁気ストライプ取引 “IC”：IC カード取引
18	jpo （支払種別）	△	半角英数字 最大 83 桁	一括払いや分割払い等の支払方法を示すコード ※ 定義される値は「3-13 支払種別」を参照
19	authCode （承認番号）	△	半角英数字 最大 7 桁	※ 半角英数字の他に半角スペースを許容します。
20	acquirerCode （仕向け先コード）	△	半角英数字 最大 2 桁	決済要求電文を仕向けた加盟店管理会社を示すコード
21	centerAcquirerId （決済センターアクワイアラ ID）	△	半角英数字 最大 8 桁	決済センターにてアクワイアラを識別するために関連付けられた一意な値
22	centerTradeId （決済センター取引 ID）	△	半角英数字 最大 64 桁	決済センターで取引を特定するための ID
23	centerProcessId （決済センター処理 ID）	△	半角英数字 最大 6 桁	決済センターで処理の照合を行うための ID
24	centerReferenceCode （決済センター照会コード）	△	文字列 最大 32 桁	決済センターで発番される照会用のコード
25	centerTransactionType （決済センター取引タイプ）	△	半角英数字 最大 8 桁	決済センターで管理される取引のタイプ
26	centerTransactionDatetime （決済センター取引日時）	△	文字列 最大 24 桁	決済センターで決済処理された日時 ※ 書式：yyyy-MM-ddTHH:mm:ssZ（ISO 8601 形式） （例）2021-05-01T01:00:00+08:00
27	centerTerminalId （決済センター端末 ID）	△	半角英数字 最大 32 桁	決済センターに登録された端末 ID
28	centerMerchantId （決済センターマーチャント ID）	△	文字列 最大 32 桁	決済センターに登録されている加盟店 ID
29	centerMerchantName （決済センターマーチャント名）	△	文字列 最大 64 桁	決済センターに登録されている加盟店 ID
30	cupRequestTime （銀聯要求送信日時）	△	半角数字 10 桁	銀聯へ送信した際の送信日時 ※ 書式：MMddHHmmss
31	emvAppId （EMV アプリケーション ID）	△	半角英数字 最大 32 桁	IC カードに設定されたアプリケーションを識別する ID
32	emvAtc （EMV アプリケーショントランザクションカウンター）	△	半角英数字 最大 4 桁	IC カード取引時に使用された IC カードに設定されたカウンターの値

33	emvAppLabel （EM アプリケーションラベル）	△	半角英数字 最大 16 桁	IC カードに設定されたアプリケーションの名称
34	panSeqNumber （カードシーケンス番号）	△	半角数字 最大 2 桁	IC カード取引時のカードシーケンス番号
35	cvmType （カード所有者検証方法タイプ）	△	半角英数字 最大 2 桁	カード保有者検証（本人確認）方法を示す値 ※ 定義される値は「3-14 カード所有者検証方法タイプ」を参照
36	cvmr （カード所有者検証結果）	△	半角英数字 最大 6 桁	カード保有者検証（本人認証）の結果
37	tvr （端末検証結果）	△	半角英数字 最大 10 桁	IC カード取引時の端末検証結果
38	tsi （トランザクションステータス情報）	△	半角英数字 最大 4 桁	IC カード取引時のカード認証の記録
39	txnCryptogram （トランザクション暗号文）	△	半角英数字 最大 16 桁	IC カード取引時に作成される暗号文
40	mileageMembershipNo （マイレージ会員番号）	△	半角英数字 最大 10 桁	ANA マイレージクラブお客様番号などの各種マイレージの会員番号
41	usageType （利用タイプ）	△	半角英数字 1 桁	消費者が加盟店を利用した状況を示す値 “0”：初回利用 “1”：複数回利用
42	lastUseDate （前回利用日）	△	半角数字 8 桁	当該取引の決済・積算を行った日時を基準とした前回利用した日 ※ 書式：yyyyMMdd

※凡例：“〇”：必須、 ”△”：任意、 ”-”：対象外

3-3-4 API 実行例

■Swift の場合

```
import MPOS SDK
    :
// 決済 API を実行する。
let handler = MposHandler()
handler.startPayment(headerParamVar,
                  requestParams: requestParamVar,
                  complete: { (response) -> Void in
                      :
                  })
```

■Objective-C の場合

```
#import <MPOS SDK/MPOS SDK-Swift.h>
    :
// 決済 API を実行する。
MposHandler *handler = [[MposHandler alloc] init];
[handler startPayment:[SettingData requestHeader]
       requestParams:requestParam
            complete:^(NSDictionary<MposResponseParam *,id> *param) {
              :
            }];
```

3-4 取消 API

この API は売上が成立していない取引に対して取消処理を行うための API です。取消処理を行う場合は、決済 API を実行することにより得られる mPOS 取引 ID を指定して取消処理を行ってください。

処理結果については、取消 API の引数に設定したクロージャーが取消処理終了時に呼び出され、そのクロージャーの引数に処理結果が設定されています。また、取消 API については、クレジットカード決済、NFC 決済、LINE Pay 決済の売上が成立していない取引に対して実行可能です。

3-4-1 API 定義

MposHandler クラスのインスタンスメソッド（requestCancel）として定義されています。

API の引数としては次の通りです。

No.	引数名	必須	型	説明
1	headerParams （共通ヘッダーパラメーター）	○	[MposRequestParam: String]	「3-1-4 共通ヘッダーパラメーター」を参照。
2	requestParams （リクエストパラメーター）	○	[MposRequestParam: String]	「3-4-2 リクエストパラメーター」を参照。
3	complete （処理終了通知クロージャー）	○	(response: [MposResponseParam: AnyObject]) -> Void	引数に Dictionary が定義されたクロージャー。 API の処理が終了する際に設定したクロージャーが呼び出されます。呼び出されたクロージャーの引数には処理結果（レスポンスパラメーター）が設定されていますので、その引数から処理結果を受け取ってください。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

■API 定義：Swift の場合

```
func requestCancel(headerParams: [MposRequestParam: String],
                   requestParams: [MposRequestParam: String],
                        complete: (response: [MposResponseParam: AnyObject]) -> Void
                   ) -> Void
```

■API 定義：Objective-C の場合

```
- (void)requestCancel:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)headerParams
       requestParams:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)requestParams
            complete:(void (^ _Nonnull)(NSDictionary<MposResponseParam *, id> * _Nonnull))complete;
```

3-4-2 リクエストパラメーター

パラメーターキーは MposRequestParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	orderId （mPOS 取引 ID）	○	半角英数字 6 桁	mPOS サービスにて取引を一意に特定する ID

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

3-4-3 レスポンスパラメーター

レスポンスパラメーターは API の引数に指定した処理終了通知クロージャーが呼び出された際に、そのクロージャーの引数に設定されます。また、パラメーターキーは MposResponseParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	status （処理ステータス）	○	文字列	“success”： 正常終了 “failure”： 異常終了
2	errorCode （エラーコード）	△	文字列	エラー内容を示すコード ※ 処理が失敗した場合に設定されます。
3	errorMessage （エラーメッセージ）	△	文字列	エラーコードメッセージ ※ 処理が失敗した場合に設定されます。
4	resultCode （結果コード）	△	半角数字 最大 10 桁	処理結果を示すコードで、主にエラーコードの補足するコードが設定される。
5	orderId （mPOS 取引 ID）	△	半角英数字 6 桁	取消処理対象となった取引の mPOS 取引 ID
6	paymentMethodCode （決済方法コード）	△	半角英数字 3 桁	決済方法（サービス）を示すコード ※ 定義される値は「3-11 決済方法コード」を参照
7	cardNumber （カード番号）	△	文字列 最大 19 桁	先頭6桁、末尾4桁以外がマスクされたカード番号
8	cardExpiry （カード有効期限）	△	文字列 最大 5 桁	年月部分がマスクされたカードの有効期限

9	jpo （支払種別）	△	半角英数字 最大 83 桁	一括払いや分割払い等の支払方法を示すコード ※ 定義される値は「3-13 支払種別」を参照
10	authCode （承認番号）	△	半角英数字 最大 7 桁	※ 半角英数字の他に半角スペースを許容します。
11	acquirerCode （仕向け先コード）	△	半角英数字 最大 2 桁	決済時に決済要求電文を仕向けた加盟店管理会社を示すコード

※凡例：“〇”：必須、 ”△”：任意、 ”-”：対象外

3-4-4 API 実行例

■Swift の場合

```
import MPOS SDK
    :
// 取消 API を実行する。
let handler = MposHandler()
handler.requestCancel(headerParamVar,
                      requestParams: requestParamVar,
                           complete: { (response) -> Void in
                               :
                      })
```

■Objective-C の場合

```
#import <MPOS SDK/MPOS SDK-Swift.h>
    :
// 取消 API を実行する。
MposHandler *handler = [[MposHandler alloc] init];
[handler requestCancel:[SettingData requestHeader]
        requestParams:requestParam
             complete:^(NSDictionary<MposResponseParam *, id> *param) {
               :
      }];
```

3-5 返金 API

この API は決済された取引に対して返金（売上取消）処理を行うための API です。返金処理を行う場合は、決済 API を実行することにより得られる mPOS 取引 ID を指定して返金処理を行ってください。

処理結果については、返金 API の引数に設定したクロージャーが返金処理終了時に呼び出され、そのクロージャーの引数に処理結果が設定されています。

3-5-1 API 定義

MposHandler クラスのインスタンスメソッド（requestRefund）として定義されています。

API の引数としては次の通りです。

No.	引数名	必須	型	説明
1	headerParams （共通ヘッダーパラメーター）	○	[MposRequestParam: String]	「3-1-4 共通ヘッダーパラメーター」を参照。
2	requestParams （リクエストパラメーター）	○	[MposRequestParam: String]	「3-5-2 リクエストパラメーター」を参照。
3	complete （処理終了通知クロージャー）	○	(response: [MposResponseParam: AnyObject]) -> Void	引数に Dictionary が定義されたクロージャー。 API の処理が終了する際に設定したクロージャーが呼び出されます。呼び出されたクロージャーの引数には処理結果（レスポンスパラメーター）が設定されていますので、その引数から処理結果を受け取ってください。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

■API 定義：Swift の場合

```
func requestRefund(headerParams: [MposRequestParam: String],
                   requestParams: [MposRequestParam: String],
                       complete: (response: [MposResponseParam: AnyObject]) -> Void
                   ) -> Void
```

■API 定義：Objective-C の場合

```
- (void)requestRefund:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)headerParams
        requestParams:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)requestParams
            complete:(void (^ _Nonnull)(NSDictionary<MposResponseParam *, id> * _Nonnull))complete;
```

3-5-2 リクエストパラメーター

パラメーターキーは MposRequestParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	orderId （mPOS 取引 ID）	○	半角英数字 6 桁	mPOS サービスにて取引を一意に特定する ID

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

3-5-3 レスポンスパラメーター

レスポンスパラメーターは API の引数に指定した処理終了通知クロージャーが呼び出された際に、そのクロージャーの引数に設定されます。また、パラメーターキーは MposResponseParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	status （処理ステータス）	○	文字列	“success”： 正常終了 “failure”： 異常終了
2	errorCode （エラーコード）	△	文字列	エラー内容を示すコード ※ 処理が失敗した場合に設定されます。
3	errorMessage （エラーメッセージ）	△	文字列	エラーコードメッセージ ※ 処理が失敗した場合に設定されます。
4	resultCode （結果コード）	△	半角数字 最大 10 桁	処理結果を示すコードで、主にエラーコードの補足するコードが設定される。
5	orderId （mPOS 取引 ID）	△	半角英数字 6 桁	返金処理対象となった取引の mPOS 取引 ID
6	paymentMethodCode （決済方法コード）	△	半角英数字 3 桁	決済方法（サービス）を示すコード ※ 定義される値は「3-11 決済方法コード」を参照
7	paymentSource （支払手段）	△	半角英数字 最大 64 桁	決済時に利用された支払いの手段を示す情報 ※ 定義される値は「3-12 支払手段」を参照
8	amount （金額）	△	半角数字 最大 8 桁	決済時に指定した金額 ※ 金額にはカンマは含みません。
9	discountAmount （割引金額）	△	半角数字 最大 8 桁	決済時に適用されたクーポン等の利用で割引された金額 ※ 金額にはカンマは含みません。
10	totalAmount （合計金額）	△	半角数字 最大 8 桁	決済時における取引の合計金額 ※ 金額にはカンマは含みません。
11	cardNumber （カード番号）	△	文字列 最大 19 桁	先頭6桁、末尾4桁以外がマスクされたカード番号

12	cardNumberToken （カード番号トークン）	△	文字列 最大 19 桁	先頭6桁、末尾4桁以外がマスクされたカード番号トークン
13	cardExpiry （カード有効期限）	△	文字列 最大 5 桁	年月部分がマスクされたカードの有効期限
14	cardType （カードタイプ）	△	半角英数字 2 桁	決済時に扱ったカードの取引種別を示すコード “MS”：磁気ストライプ取引 “IC”：IC カード取引
15	jpo （支払種別）	△	半角英数字 最大 83 桁	一括払いや分割払い等の支払方法を示すコード ※ 定義される値は「3-13 支払種別」を参照
16	authCode （承認番号）	△	半角英数字 最大 7 桁	※ 半角英数字の他に半角スペースを許容します。
17	acquirerCode （仕向け先コード）	△	半角英数字 最大 2 桁	決済時に決済要求電文を仕向けた加盟店管理会社を示すコード
18	centerAcquirerId （決済センターアクワイアラ ID）	△	半角英数字 最大 8 桁	決済センターにてアクワイアラを識別するために関連付けられた一意な値
19	centerTradeId （決済センター取引 ID）	△	半角英数字 最大 64 桁	決済センターで取引を特定するための ID
20	centerProcessId （決済センター処理 ID）	△	半角英数字 最大 6 桁	決済センターで処理の照合を行うための ID
21	centerReferenceCode （決済センター照会コード）	△	文字列 最大 32 桁	決済センターで発番される照会用のコード
22	centerTransactionType （決済センター取引タイプ）	△	半角英数字 最大 8 桁	決済センターで管理される取引のタイプ
23	centerTransactionDatetime （決済センター取引日時）	△	文字列 最大 24 桁	決済センターで決済処理された日時 ※ 書式：yyyy-MM-ddTHH:mm:ssZ（ISO 8601 形式） （例）2021-05-01T01:00:00+08:00
24	centerRefundDatetime （決済センター返金日時）	△	文字列 最大 24 桁	決済センターで返金処理された日時 ※ 書式：yyyy-MM-ddTHH:mm:ssZ（ISO 8601 形式） （例）2021-05-01T01:00:00+08:00
25	centerTerminalId （決済センター端末 ID）	△	半角英数字 最大 32 桁	決済センターに登録された端末 ID
26	centerMerchantId （決済センターマーチャント ID）	△	文字列 最大 32 桁	決済センターに登録されている加盟店 ID
27	centerMerchantName （決済センターマーチャント名）	△	文字列 最大 64 桁	決済センターに登録されている加盟店 ID
28	cupRequestTime （銀聯要求送信日時）	△	半角数字 10 桁	銀聯へ送信した際の送信日時 ※ 書式：MMddHHmmss

29	emvAppId （EMV アプリケーション ID）	△	半角英数字 最大 32 桁	IC カードに設定されたアプリケーションを識別する ID
30	emvAtc （EMV アプリケーショントランザクションカウンター）	△	半角英数字 最大 4 桁	IC カード取引時に使用された IC カードに設定されたカウンターの値
31	emvAppLabel （EM アプリケーションラベル）	△	半角英数字 最大 16 桁	IC カードに設定されたアプリケーションの名称
32	panSeqNumber （カードシーケンス番号）	△	半角数字 最大 2 桁	IC カード取引時のカードシーケンス番号
33	cvmType （カード所有者検証方法タイプ）	△	半角英数字 最大 2 桁	カード保有者検証（本人確認）方法を示す値 ※ 定義される値は「3-14 カード所有者検証方法タイプ」を参照
34	cvmr （カード所有者検証結果）	△	半角英数字 最大 6 桁	カード保有者検証（本人認証）の結果
35	tvr （端末検証結果）	△	半角英数字 最大 10 桁	IC カード取引時の端末検証結果
36	tsi （トランザクションステータス情報）	△	半角英数字 最大 4 桁	IC カード取引時のカード認証の記録
37	txnCryptogram （トランザクション暗号文）	△	半角英数字 最大 16 桁	IC カード取引時に作成される暗号文
38	mileageMembershipNo （マイレージ会員番号）	△	半角英数字 最大 10 桁	ANA マイレージクラブお客様番号などの各種マイレージの会員番号

※凡例：“〇”：必須、 ”△”：任意、 ”-”：対象外

3-5-4 API 実行例

■Swift の場合

```
import MPOS SDK
    :
// 返金 API を実行する。
let handler = MposHandler()
handler.requestRefund(headerParamVar,
                      requestParams: requestParamVar,
                          complete: { (response) -> Void in
                              :
                      })
```

■Objective-C の場合

```
#import <MPOS SDK/MPOS SDK-Swift.h>

    :

// 返金 API を実行する。

MposHandler *handler = [[MposHandler alloc] init];

[handler requestRefund:[SettingData requestHeader]

        requestParams:requestParam

             complete:^(NSDictionary<MposResponseParam *, id> *param) {

               :

      }];
```

3-6 取引検索 API

この API はユーザーが実施した取引の履歴を取得するための API です。検索条件を指定することにより、指定した条件に合致する取引履歴が取得できます。尚、取引検索 API で検索できる取引は、当該ユーザーが決済した取引に限ります。そのため、同一加盟店に属する他のユーザーが実施した取引は検索することはできません。

処理結果については、この API の引数に設定したクロージャーが処理終了時に呼び出され、そのクロージャーの引数に検索結果が設定されています。

3-6-1 API 定義

MposHandler クラスのインスタンスメソッド（requestSearchTransactionHistory）として定義されています。

API の引数としては次の通りです。

No.	引数名	必須	型	説明
1	headerParams （共通ヘッダーパラメーター）	○	[MposRequestParam: String]	「3-1-4 共通ヘッダーパラメーター」を参照。
2	requestParams （リクエストパラメーター）	○	[MposRequestParam: String]	「3-6-2 リクエストパラメーター」を参照。
3	complete （処理終了通知クロージャー）	○	(response: [MposResponseParam: AnyObject]) -> Void	引数に Dictionary が定義されたクロージャー。 API の処理が終了する際に設定したクロージャーが呼び出されます。呼び出されたクロージャーの引数には処理結果（レスポンスパラメーター）が設定されていますので、その引数から処理結果を受け取ってください。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

■API 定義：Swift の場合

```
func requestSearchTransactionHistory(headerParams: [MposRequestParam: String],
                                     requestParams: [MposRequestParam: String],
                                         complete: (response: [MposResponseParam: AnyObject]) -> Void
                                     ) -> Void
```

■API 定義：Objective-C の場合

```
- (void)requestSearchTransactionHistory:
                        (NSDictionary<MposRequestParam *, NSString *> * _Nonnull)headerParams
           requestParams:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)requestParams
                complete:(void (^ _Nonnull)(NSDictionary<MposResponseParam *, id> * _Nonnull))complete;
```

3-6-2 リクエストパラメーター

パラメーターキーは MposRequestParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	transactionCount （取引件数）	△	半角数字 最大 10 桁	取引の最大取得件数を設定します。未指定の場合は初期値として”1”が設定されたものとします。 ※ 10 桁以内ですが 2,147,483,647 を上限値とします。 また、負の値は指定できません。
2	orderId （mPOS 取引 ID）	△	半角英数字 6 桁	指定した mPOS 取引 ID を持つ取引を検索します。
3	orderReferenceId （mPOS 取引照会 ID）	△	文字列 最大 64 文字	指定した mPOS 取引照会 ID を持つ取引を検索します。 ※半角英数字と”-”（ハイフン）、”_”（アンダースコア）が使用可能です。
4	registeredTimeFrom （取引登録日時 FROM）	△	半角数字 14 桁	指定した日時以降に登録された取引を検索します。 ※ 指定した日時は検索範囲に含みます。 [x >= 取引登録日時 FROM] ※ 書式：yyyyMMddHHmmss
5	registeredTimeTo （取引登録日時 TO）	△	半角数字 14 桁	指定した日時以前に登録された取引を検索します。 ※ 指定した日時は検索範囲に含まれません。 [x < 取引登録日時 TO] ※ 書式：yyyyMMddHHmmss

※凡例：“〇”：必須、 ”△”：任意、 ”-”：対象外

3-6-3 レスポンスパラメーター

レスポンスパラメーターは API の引数に指定した処理終了通知クロージャーが呼び出された際に、そのクロージャーの引数に設定されます。また、パラメーターキーは MposResponseParam に定義されている値を使用してください。

尚、検索結果として取得される取引履歴については、レスポンスパラメーターの”historyList”キーに紐付く値として格納されており、その値は配列形式となります。

No.	パラメーターキー	必須	書式・制限	説明
1	status （処理ステータス）	○	文字列	“success”： 正常終了 “failure”： 異常終了
2	errorCode （エラーコード）	△	文字列	エラー内容を示すコード ※ 処理が失敗した場合に設定されます。
3	errorMessage （エラーメッセージ）	△	文字列	エラーコードメッセージ ※ 処理が失敗した場合に設定されます。
4	resultCode （結果コード）	△	半角数字 最大 10 桁	処理結果を示すコードで、主にエラーコードの補足するコードが設定される。
5	historyList	△		検索条件に該当する取引履歴については、 [Dictionary <MposResponseParam, AnyObject>]型の配列として格納されています。
※該当する取引履歴（繰り返し項目）				
6	orderId （mPOS 取引 ID）	△	半角英数字 6 桁	mPOS サービスにて取引を一意に特定する ID
7	orderReferenceId （mPOS 取引照会 ID）	△	文字列 最大 64 文字	mPOS サービスにて取引を照会するための ID
8	paymentMethodCode （決済方法コード）	△	半角英数字 3 桁	決済方法（サービス）を示すコード ※ 定義される値は「3-11 決済方法コード」を参照
9	paymentSource （支払手段）	△	半角英数字 最大 64 桁	決済時に利用された支払いの手段を示す情報 ※ 定義される値は「3-12 支払手段」を参照
10	orderStatus （取引ステータス）	△	文字列	取引の状態を示す。 “Init”： 決済申込 “Auth”： 与信 “PreCapture”： 仮売上 “Capture”： 売上 “PreRefund”： 仮売上取消 “Refund”： 返金 “Cancel”： 取消
11	status （処理ステータス）	△	文字列	“success”： 正常終了 “failure”： 異常終了

12	resultCode （結果コード）	△	半角数字 最大 10 桁	取引に対する処理結果を示すコード
13	amount （金額）	△	半角数字 最大 8 桁	決済時に指定した金額 金額にはカンマは含みません。
14	discountAmount （割引金額）	△	半角数字 最大 8 桁	決済時に適用されたクーポン等の利用で割引された金額 金額にはカンマは含みません。
15	totalAmount （合計金額）	△	半角数字 最大 8 桁	決済時における取引の合計金額 金額にはカンマは含みません。
16	registeredTime （取引登録日時）	△	半角数字 14 桁	取引が登録された日時（日本時間） ※ 書式：yyyyMMddHHmmss
17	paymentTime （取引日時）	△	半角数字 14 桁	決済処理が行われた最終日時（日本時間） ※ 書式：yyyyMMddHHmmss
18	orderDescription （説明）	△	文字列 最大 40 文字	取引に対する説明
19	cardNumber （カード番号）	△	文字列 最大 19 桁	先頭6桁、末尾4桁以外がマスクされたカード番号
20	cardNumberToken （カード番号トークン）	△	文字列 最大 19 桁	先頭6桁、末尾4桁以外がマスクされたカード番号トークン
21	cardExpiry （カード有効期限）	△	文字列 最大 5 桁	年月部分がマスクされたカードの有効期限
22	cardType （カードタイプ）	△	半角英数字 2 桁	決済時に扱ったカードの取引種別を示すコード “MS”：磁気ストライプ取引 “IC”：IC カード取引
23	jpo （支払種別）	△	半角英数字 最大 83 桁	一括払いや分割払い等の支払方法を示すコード ※ 定義される値は「3-13 支払種別」を参照
24	authCode （承認番号）	△	半角英数字 最大 7 桁	※ 半角英数字の他に半角スペースを許容します。
25	acquirerCode （仕向け先コード）	△	半角英数字 最大 2 桁	決済時に決済要求電文を仕向けた加盟店管理会社を示すコード
26	centerAcquirerId （決済センターアクワイアラ ID）	△	半角英数字 最大 8 桁	決済センターにてアクワイアラを識別するために関連付けられた一意な値
27	centerTradeId （決済センター取引 ID）	△	半角英数字 最大 64 桁	決済センターで取引を特定するための ID
28	centerProcessId （決済センター処理 ID）	△	半角英数字 最大 6 桁	決済センターで処理の照合を行うための ID
29	centerReferenceCode （決済センター照会コード）	△	文字列 最大 32 桁	決済センターで発番される照会用のコード

30	centerTransactionType （決済センター取引タイプ）	△	半角英数字 最大 8 桁	決済センターで管理される取引のタイプ
31	centerTransactionDatetime （決済センター取引日時）	△	文字列 最大 24 桁	決済センターで決済処理された日時 ※ 書式：yyyy-MM-ddTHH:mm:ssZ（ISO 8601 形式） （例）2021-05-01T01:00:00+08:00
32	centerRefundDatetime （決済センター返金日時）	△	文字列 最大 24 桁	決済センターで返金処理された日時 ※ 書式：yyyy-MM-ddTHH:mm:ssZ（ISO 8601 形式） （例）2021-05-01T01:00:00+08:00
33	centerTerminalId （決済センター端末 ID）	△	半角英数字 最大 32 桁	決済センターに登録された端末 ID
34	centerMerchantId （決済センターマーチャント ID）	△	文字列 最大 32 桁	決済センターに登録されている加盟店 ID
35	centerMerchantName （決済センターマーチャント名）	△	文字列 最大 64 桁	決済センターに登録されている加盟店 ID
36	cupRequestTime （銀聯要求送信日時）	△	半角数字 10 桁	銀聯へ送信した際の送信日時 ※ 書式：MMddHHmmss
37	emvAppId （EMV アプリケーション ID）	△	半角英数字 最大 32 桁	IC カードに設定されたアプリケーションを識別する ID
38	emvAtc （EMV アプリケーショントランザクションカウンター）	△	半角英数字 最大 4 桁	IC カード取引時に使用された IC カードに設定されたカウンターの値
39	emvAppLabel （EM アプリケーションラベル）	△	半角英数字 最大 16 桁	IC カードに設定されたアプリケーションの名称
40	panSeqNumber （カードシーケンス番号）	△	半角数字 最大 2 桁	IC カード取引時のカードシーケンス番号
41	cvmType （カード所有者検証方法タイプ）	△	半角英数字 最大 2 桁	カード保有者検証（本人確認）方法を示す値 ※ 定義される値は「3-14 カード所有者検証方法タイプ」を参照
42	cvmr （カード所有者検証結果）	△	半角英数字 最大 6 桁	カード保有者検証（本人認証）の結果
43	tvr （端末検証結果）	△	半角英数字 最大 10 桁	IC カード取引時の端末検証結果
44	tsi （トランザクションステータス情報）	△	半角英数字 最大 4 桁	IC カード取引時のカード認証の記録
45	txnCryptogram （トランザクション暗号文）	△	半角英数字 最大 16 桁	IC カード取引時に作成される暗号文

46	mileageMembershipNo （マイレージ会員番号）	△	半角英数字 最大 10 桁	ANA マイレージクラブお客様番号などの各種マイレージの会員番号
47	usageType （利用タイプ）	△	半角英数字 1 桁	消費者が加盟店を利用した状況を示す値 “0”：初回利用 “1”：複数回利用
48	lastUseDate （前回利用日）	△	半角数字 8 桁	当該取引の決済・積算を行った日時を基準とした前回利用した日 ※ 書式：yyyyMMdd

※凡例：“〇”：必須、 ”△”：任意、 ”-”：対象外

3-6-4 API 実行例

■Swift の場合

```
import MPOS SDK
    :
// 取引検索 API を実行する。
let handler = MposHandler()
handler.requestSearchTransactionHistory(headerParamVar,
                        requestParams: requestParamVar,
                             complete: { (response) -> Void in
                                :
                             })
```

■Objective-C の場合

```
#import <MPOS SDK/MPOS SDK-Swift.h>
    :
// 取引検索 API を実行する。
MposHandler *handler = [[MposHandler alloc] init];
[handler requestSearchTransactionHistory:headerParamVar
                          requestParams:requestParamVar
                               complete:^(NSDictionary<MposResponseParam *,id> *param) {
                                 :
                               }];
```

3-7 取引集計 API

この API はユーザーが実施した取引を集計する API です。集計条件を指定することにより、指定した条件に合致する売上・返金・マイル積算・マイル積算取消処理された取引の集計を行います。尚、取引集計 API で集計できる取引は当該ユーザーが実施した取引に限ります。そのため、同一加盟店に属する他のユーザーが決済した取引は集計することはできません。

処理結果については、この API の引数に設定したクロージャーが処理終了時に呼び出され、そのクロージャーの引数に集計結果が設定されています。

3-7-1 API 定義

MposHandler クラスのインスタンスメソッド（requestReport）として定義されています。

API の引数としては次の通りです。

No.	引数名	必須	型	説明
1	headerParams （共通ヘッダーパラメーター）	○	[MposRequestParam: String]	「3-1-4 共通ヘッダーパラメーター」を参照。
2	requestParams （リクエストパラメーター）	○	[MposRequestParam: String]	「3-7-2 リクエストパラメーター」を参照。
3	complete （処理終了通知クロージャー）	○	(response: [MposResponseParam: AnyObject]) -> Void	引数に Dictionary が定義されたクロージャー。 API の処理が終了する際に設定したクロージャーが呼び出されます。呼び出されたクロージャーの引数には処理結果（レスポンスパラメーター）が設定されていますので、その引数から処理結果を受け取ってください。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

■API 定義：Swift の場合

```
func requestReport(headerParams: [MposRequestParam: String],
                   requestParams: [MposRequestParam: String],
                       complete: (response: [MposResponseParam: AnyObject]) -> Void
                     ) -> Void
```

■API 定義：Objective-C の場合

```
- (void)requestReport:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)headerParams
       requestParams:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)requestParams
            complete:(void (^ _Nonnull)(NSDictionary<MposResponseParam *, id> * _Nonnull))complete;
```

3-7-2 リクエストパラメーター

パラメーターキーは MposRequestParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	reportTimeFrom （集計日時 FROM）	○	半角英数字 10 桁	指定した日時以降に処理された取引を集計します。 ※ 指定した日時は検索範囲に含みます。 [x >= 集計日時 FROM] ※ 書式：yyyyMMddHH
2	reportTimeTo （集計日時 TO）	○	半角英数字 10 桁	指定した日時以前に処理された取引を集計します。 ※ 指定した日時は検索範囲に含みます。 [x <= 集計日時 TO] ※ 書式：yyyyMMddHH ※ 書式上は分秒を指定できませんが、59 分 59 秒までが集計範囲に含まれます。
3	paymentMethodCode （決済方法コード）	△	文字列	集計対象とする決済方法（サービス）をカンマ区切りで設定します。 ※ 決済方法（サービス）コードとして定義される値は「3-11 決済方法コード」を参照
4	orderStatus （取引ステータス）	△	文字列	集計対象とする取引ステータスをカンマ区切りで設定します。 [取引ステータス] “Capture”： 売上 “Refund”： 返金 “Accrual”： マイル積算 “AccrualCancel”： マイル積算取消

※凡例： “○”：必須、 ”△”：任意、 ”-”：対象外

3-7-3 レスポンスパラメーター

レスポンスパラメーターは API の引数に指定した処理終了通知クロージャーが呼び出された際に、そのクロージャーの引数に設定されます。また、パラメーターキーは MposResponseParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	status （処理ステータス）	○	文字列	“success”： 正常終了 “failure”： 異常終了
2	errorCode （エラーコード）	△	文字列	エラー内容を示すコード ※ 処理が失敗した場合に設定されます。
3	errorMessage （エラーメッセージ）	△	文字列	エラーコードメッセージ ※ 処理が失敗した場合に設定されます。
4	resultCode （結果コード）	△	半角数字 最大 10 桁	処理結果を示すコードで、主にエラーコードの補足するコードが設定される。
5	reportTimeFrom （集計日時 FROM）	△	半角英数字 10 桁	指定した集計日時 FROM ※ 書式：yyyyMMddHH
6	reportTimeTo （集計日時 TO）	△	半角英数字 10 桁	指定した集計日時 TO ※ 書式：yyyyMMddHH
7	paymentMethodCode （決済方法コード）	△	文字列 ※リスト形式	指定した決済方法コード。 ※ 決済方法コードはリスト形式で設定されます。
8	orderStatus （取引ステータス）	△	文字列 ※リスト形式	指定した取引ステータス。 ※ 取引ステータスはリスト形式で設定されます。
9	salesReport （決済集計）	△	-	決済系取引の集計結果が設定されます。 ※ 決済集計として設定される各内容ついては No.9-X を参照
※salesReport（決済集計）に含まれる内容				
9-1	captureTotalAmount （売上額合計）	△	半角数字	集計条件に合致する決済系取引の売上額合計
9-2	refundTotalAmount （返金額合計）	△	半角数字	集計条件に合致する決済系取引の返金額合計
9-3	differenceTotalAmount （差引額合計）	△	半角数字	売上額合計から返金額合計を引いた額
9-4	captureCount （売上件数）	△	半角数字	集計条件に合致する決済系取引の売上件数
9-5	refundCount （返金件数）	△	半角数字	集計条件に合致する決済系取引の返金件数
9-6	differenceCount （差引件数）	△	半角数字	売上件数から返金件数を引いた件数

10	mileageReport （マイル積算集計）	△	-	マイル系取引の集計結果が設定されます。 ※ マイル積算集計として設定される各内容ついては No.10-X を参照
※mileageReport（マイル積算集計）に含まれる内容				
10-1	accrualTotalAmount （積算額合計）	△	半角数字	集計条件に合致するマイル系取引の積算額合計
10-2	cancelTotalAmount （積算取消額合計）	△	半角数字	集計条件に合致するマイル系取引の積算取消額合計
10-3	differenceTotalAmount （積算差引額合計）	△	半角数字	積算額合計から積算取消額合計を引いた額
10-4	accrualCount （積算件数）	△	半角数字	集計条件に合致するマイル系取引の積算件数
10-5	cancelCount （積算取消件数）	△	半角数字	集計条件に合致するマイル系取引の積算取消件数
10-6	differenceCount （積算差引件数）	△	半角数字	積算件数から積算取消件数を引いた件数

※凡例：“〇”：必須、 ”△”：任意、 ”-”：対象外

3-7-4 API 実行例

■Swift の場合

```
import MPOS SDK
    :
// 取引集計 API を実行する。
let handler = MposHandler()
handler.requestReport(headerParamVar,
                      requestParams: requestParamVar,
                           complete: { (response) -> Void in
                              :
                      })
```

■Objective-C の場合

```
#import <MPOS SDK/MPOS SDK-Swift.h>

    :

// 取引集計 API を実行する。

MposHandler *handler = [[MposHandler alloc] init];

[handler requestReport:headerParamVar

        requestParams:requestParamVar

             complete:^(NSDictionary<MposResponseParam *,id> * param) {

                :

       }];
```

3-8 メール送信 API

この API は指定したメールアドレス宛てに取引に関するメールを送信するための API です。この API で取り扱うメールは、消費者様向けのレシートメール、消費者様向けの返金受付通知メールになります。

処理結果については、この API の引数に設定したクロージャーが処理終了時に呼び出され、そのクロージャーの引数に処理結果が設定されています。

3-8-1 API 定義

MposHandler クラスのインスタンスメソッド（requestSendReceipt）として定義されています。

API の引数としては次の通りです。

No.	引数名	必須	型	説明
1	headerParams （共通ヘッダーパラメーター）	○	[MposRequestParam: String]	「3-1-4 共通ヘッダーパラメーター」を参照。
2	requestParams （リクエストパラメーター）	○	[MposRequestParam: String]	「3-8-2 リクエストパラメーター」を参照。
3	complete （処理終了通知クロージャー）	○	(response: [MposResponseParam: AnyObject]) -> Void	引数に Dictionary が定義されたクロージャー。 API の処理が終了する際に設定したクロージャーが呼び出されます。呼び出されたクロージャーの引数には処理結果（レスポンスパラメーター）が設定されていますので、その引数から処理結果を受け取ってください。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

■API 定義：Swift の場合

```
func requestSendReceipt(headerParams: [MposRequestParam: String],
                        requestParams: [MposRequestParam: String],
                             complete: (response: [MposResponseParam: AnyObject]) -> Void
                    ) -> Void
```

■API 定義：Objective-C の場合

```
- (void)requestSendReceipt:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)headerParams
            requestParams:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)requestParams
                 complete:(void (^ _Nonnull)(NSDictionary<MposResponseParam *, id> * _Nonnull))complete;
```

3-8-2 リクエストパラメーター

パラメーターキーは MposRequestParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	orderId （mPOS 取引 ID）	○	半角英数字 6 桁	メールを送信 mPOS 取引 ID
2	email （メールアドレス）	○	半角英数字 最大 256 桁	メールを送信する宛先を設定します。 ※ 半角英数字の他に次の記号を許容します。 “.”（ドット）、”-”（ハイフン）、”_”（アンダースコア）、”@”（アットマーク）
3	mailType （メールタイプ）	△	半角数字 1 桁	“1”：レシート再送信 “2”：返金受付通知再送信 ※ 未設定の場合は「レシート再送信」となります。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

3-8-3 レスポンスパラメーター

レスポンスパラメーターは API の引数に指定した処理終了通知クロージャーが呼び出された際に、そのクロージャーの引数に設定されます。また、パラメーターキーは MposResponseParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	status （処理ステータス）	○	文字列	“success”： 正常終了 “failure”： 異常終了
2	errorCode （エラーコード）	△	文字列	エラー内容を示すコード ※ 処理が失敗した場合に設定されます。
3	errorMessage （エラーメッセージ）	△	文字列	エラーコードメッセージ ※ 処理が失敗した場合に設定されます。
4	resultCode （結果コード）	△	半角数字 最大 10 桁	処理結果を示すコードで、主にエラーコードの補足するコードが設定される。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

3-8-4 API 実行例

■Swift の場合

```
import MPOS SDK

    :

// メール送信 API を実行する。

let handler = MposHandler()

handler.requestSendReceipt(headerParamVar,

                           requestParams: requestParamVar,

                                complete: { (response) -> Void in

                                   :

                       })
```

■Objective-C の場合

```
#import <MPOS SDK/MPOS SDK-Swift.h>

    :

// メール送信 API を実行する。

MposHandler *handler = [[MposHandler alloc] init];

[handler requestSendReceipt:headerParamVar

             requestParams:requestParamVar

                  complete:^(NSDictionary<MposResponseParam *,id> *param) {

                     :

                  }];
```

3-9 パスワード変更 API

この API はユーザーのパスワードを変更するための API です。パスワードを変更するためには、変更前と変更後のパスワードを指定して実行してください。

処理結果については、この API の引数に設定したクロージャーが処理終了時に呼び出され、そのクロージャーの引数に処理結果が設定されています。

3-9-1 API 定義

MposHandler クラスのインスタンスメソッド（requestChangePassword）として定義されています。

API の引数としては次の通りです。

No.	引数名	必須	型	説明
1	headerParams （共通ヘッダーパラメーター）	○	[MposRequestParam: String]	「3-1-4 共通ヘッダーパラメーター」を参照。
2	requestParams （リクエストパラメーター）	○	[MposRequestParam: String]	「3-9-2 リクエストパラメーター」を参照。
3	complete （処理終了通知クロージャー）	○	(response: [MposResponseParam: AnyObject]) -> Void	引数に Dictionary が定義されたクロージャー。 API の処理が終了する際に設定したクロージャーが呼び出されます。呼び出されたクロージャーの引数には処理結果（レスポンスパラメーター）が設定されていますので、その引数から処理結果を受け取ってください。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

■API 定義：Swift の場合

```
func requestChangePassword(headerParams: [MposRequestParam: String],
                           requestParams: [MposRequestParam: String],
                                complete: (response: [MposResponseParam: AnyObject]) -> Void
                          ) -> Void
```

■API 定義：Objective-C の場合

```
- (void)requestChangePassword:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)headerParams
        requestParams:(NSDictionary<MposRequestParam *, NSString *> * _Nonnull)requestParams
            complete:(void (^ _Nonnull)(NSDictionary<MposResponseParam *, id> * _Nonnull))complete;
```

3-9-2 リクエストパラメーター

パラメーターキーは MposRequestParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	oldPasswd （旧パスワード）	○	半角英数字 8～20 桁	変更前のパスワードを設定します。
2	newPasswd （新パスワード）	○	半角英数字 8～20 桁	変更後のパスワードを設定します。 新しく設定するパスワードは、必ず半角英文字（小文字）、半角英文字（大文字）、半角数字の3種類の文字種で構成されたパスワードを設定してください。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

3-9-3 レスポンスパラメーター

レスポンスパラメーターは API の引数に指定した処理終了通知クロージャーが呼び出された際に、そのクロージャーの引数に設定されます。また、パラメーターキーは MposResponseParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	status （処理ステータス）	○	文字列	“success”： 正常終了 “failure”： 異常終了
2	errorCode （エラーコード）	△	文字列	エラー内容を示すコード ※ 処理が失敗した場合に設定されます。
3	errorMessage （エラーメッセージ）	△	文字列	エラーコードメッセージ ※ 処理が失敗した場合に設定されます。
4	resultCode （結果コード）	△	半角数字 最大 10 桁	処理結果を示すコードで、主にエラーコードの補足するコードが設定される。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

3-9-4 API 実行例

■Swift の場合

```
import MPOS SDK

    :

// パスワード変更 API を実行する。

let handler = MposHandler()

handler.requestChangePassword(headerParamVar,

                      requestParams: requestParamVar,

                           complete: { (response) -> Void in

                              :

                      })
```

■Objective-C の場合

```
#import <MPOS SDK/MPOS SDK-Swift.h>

    :

// パスワード変更 API を実行する。

MposHandler *handler = [[MposHandler alloc] init];

[handler requestChangePassword:headerParamVar

                requestParams:requestParamVar

                     complete:^(NSDictionary<MposResponseParam *,id> *param) {

                        :

                     }];
```

3-10 カードリーダーアプリ更新 API

この API はカードリーダーにインストールされたアプリケーションの更新要求を行う API です。カードリーダーアプリに対して更新要求を行う際は、対象のカードリーダーに接続して実施してください。尚、この API が対象とするカードリーダーは PIN パッドありカードリーダーに限ります。また、ご利用のカードリーダーにインストールされたアプリケーションがこの機能に対応していない場合は、この機能をご利用できません。

処理結果については、この API の引数に設定したクロージャーが処理終了時に呼び出され、そのクロージャーの引数に処理結果が設定されています。尚、この API は更新要求を行う API であるため、処理結果として成功を示している場合であっても、それは更新要求が受理されたことを示すものであり、カードリーダーにインストールされたアプリケーションが更新されたことを示すものではありません。

3-10-1 API 定義

MposHandler クラスのインスタンスメソッド（requestReaderAppUpdateBT）として定義されています。

API の引数としては次の通りです。

No.	引数名	必須	型	説明
1	headerParams （共通ヘッダーパラメーター）	○	[MposRequestParam: String]	「3-1-4 共通ヘッダーパラメーター」を参照。
2	requestParams （リクエストパラメーター）	○	[MposRequestParam: String]	「3-10-2 リクエストパラメーター」を参照。
3	complete （処理終了通知クロージャー）	○	(response: [MposResponseParam: AnyObject]) -> Void	引数に Dictionary が定義されたクロージャー。 API の処理が終了する際に設定したクロージャーが呼び出されます。呼び出されたクロージャーの引数には処理結果（レスポンスパラメーター）が設定されていますので、その引数から処理結果を受け取ってください。

※凡例：“○”：必須、 ”△”：任意、 ”-”：対象外

■API 定義：Swift の場合

```
func requestReaderAppUpdateBT(headerParams: [MposRequestParam: String],
                              requestParams: [MposRequestParam: String],
                                  complete: (response: [MposResponseParam: AnyObject]) -> Void
                          ) -> Void
```

■API 定義：Objective-C の場合

```
- (void)requestReaderAppUpdateBT:(NSDictionary<MposRequestParam *, NSString *> * __nonnull)headerParams
        requestParams:(NSDictionary<MposRequestParam *, NSString *> * __nonnull)requestParams
            complete:(void (^ __nonnull)(NSDictionary<MposResponseParam *, id> * __nonnull))complete;
```

3-10-2 リクエストパラメーター

パラメーターキーは MposRequestParam に定義されている値を使用してください。

No.	パラメーターキー	必須	書式・制限	説明
1	progressDialog （進捗状況ダイアログフラグ）	△	半角数字 1 桁	進捗状況ダイアログの表示有無を設定します。 ※ 値が未設定の場合は非表示”0”が設定された場合と同じ扱いとなります。 非表示： ”0”、 表示： ”1”
2	btConnectTimeout （接続タイムアウト時間）	△	半角数字 最大 3 桁	対象のカードリーダーとの Bluetooth 接続が確立されるまでのタイムアウト時間（秒）を設定します。 ※ 進捗状況ダイアログを表示しない場合に設定値が有効となります。 ※ 値を指定しなかった場合はデフォルト値（60 秒）が適用されます。

※凡例： “〇”：必須、 ”△”：任意、 ”-”：対象外

3-10-3 レスポンスパラメーター

レスポンスパラメーターは API の引数に指定した処理終了通知クロージャーが呼び出された際に、そのクロージャーの引数に設定されます。また、パラメーターキーは MposResponseParam に定義されている値を使用して下さい。

No.	パラメーターキー	必須	書式・制限	説明
1	status （処理ステータス）	〇	文字列	“success”： 正常終了 “failure”： 異常終了
2	errorCode （エラーコード）	△	文字列	エラー内容を示すコード ※ 処理が失敗した場合に設定されます。
3	errorMessage （エラーメッセージ）	△	文字列	エラーコードメッセージ ※ 処理が失敗した場合に設定されます。
4	resultCode （結果コード）	△	半角数字 最大 10 桁	処理結果を示すコードで、主にエラーコードの補足するコードが設定される。

※凡例： “〇”：必須、 ”△”：任意、 ”-”：対象外

3-10-4 API 実行例

■Swift の場合

```
import MPOS SDK

    :

// カードリーダーアプリ更新 API を実行する。

let handler = MposHandler()

handler.requestReaderAppUpdateBT(headerParamVar,

                      requestParams: requestParamVar,

                           complete: { (response) -> Void in

                              :

                      })
```

■Objective-C の場合

```
#import <MPOS SDK/MPOS SDK-Swift.h>

    :

// カードリーダーアプリ更新 API を実行する。

MposHandler *handler = [[MposHandler alloc] init];

[handler requestReaderAppUpdateBT:headerParamVar

                  requestParams:requestParamVar

                       complete:^(NSDictionary<MposResponseParam *,id> * param) {

                          :

                    }];
```

3-11 決済方法コード

決済方法（サービス）として定義する値とその名称は次の通りです。

値	分類	名称
001	カード決済系	クレジットカード決済
002	カード決済系	銀聯決済
010	カード決済系	NFC 決済
020	コード決済系	Alipay 決済
021	コード決済系	Alipay+決済
030	コード決済系	WeChat Pay 決済
040	コード決済系	LINE Pay 決済
050	コード決済系	d 払い決済
060	コード決済系	PayPay 決済
070	コード決済系	au Pay 決済
080	コード決済系	メルペイ決済
090	コード決済系	楽天ペイ決済
100	コード決済系	J-Coin Pay 決済
110	コード決済系	UnionPay QR コード決済
120	コード決済系	Smart Code 決済
Z01	コード決済系	コード決済　※コード決済を特定できない場合の汎用的なコードです。
A01	マイル積算系	ANA マイル積算
A02	マイル積算系	ユナイテッド航空マイル積算
AZZ	マイル積算系	その他マイル積算

3-12 支払手段

支払手段は消費者が支払いを行なった手段を示した情報です。この支払手段はコード決済を実施した場合に限り取得することができます。ただし、Smart Code 決済を行なった場合については決済処理を実施した後日に取得可能となります。また、テストアカウントを利用して Alipay+決済、Smart Code 決済を行なった場合は支払手段が無作為に決定されます。

支払手段として定義する値は次の通りです。

値	名称
ALIPAY_CN	Alipay
ALIPAY_HK	AlipayHK
WECHATPAY	WeChat Pay
LINEPAY	LINE Pay
DPAYMENT	d 払い
AUPAY	au PAY
PAYPAY	PayPay
RPAY	楽天ペイ
MERPAY	メルペイ
JCOINPAY	J-Coin Pay
UNIONPAY_QR	UnionPay QR コード
GINKOUPAY	銀行 Pay
KPLUS	K PLUS
EPOSPAY	EPOS Pay
PRING	pring
ATONE	atone
AFTEE	aftee
PREMOCODE	プレモコード払い
MYJCBPAY	MyJCB Pay
ANAPAY	ANA Pay
FAMIPAY	FamiPay
LALAPAY	ララ Pay
PAIDY	Paidy
KAKAOPAY	KakaoPay
TNG	TnG e-wallet
EZLINK	Ez-Link eWallet
TRUEMONEY	TrueMoney
DANA	DANA

値	名称
GCASH	GCash

3-13 支払種別

支払種別は一括払いや分割払い等の支払方法を定義した値です。支払種別として定義するは次の通りです。

値	内容
10	一括払い
21	ボーナス一括払い
23Fmm	ボーナス払い（ボーナス月指定）
61AmmCxx	分割払い
80	リボルビング払い

※ “Amm”は省略される場合があります。

※ “Amm”の”mm”部分は支払開始月を設定します。　（例）1月の場合：A01

※ “Cxx”の”xx”部分は分割回数を設定します。　（例）3回の場合：C03

※ “Fmm”の”mm”はボーナス月を設定します。　（例）4月の場合：F04

3-14 カード所有者検証方法タイプ

カード所有者検証（本人確認）方法として定義する値は次の通りです。

値	内容
00	不明
01	検証失敗
02	検証不要
10	PIN（暗証番号）での検証
11	署名での検証
12	PIN（暗証番号）と署名の両方での検証
13	消費者端末上での検証

3-15 エラーコード

コード値	説明
101	エラーが発生しました。別のカードをご利用ください。
102	限度額オーバーのためカードが使用できない状態です。
103	磁気決済はご利用できません。IC カードスロットにカードを挿入するか、別のカードをご利用ください。
104	IC 決済はご利用できません。カードをスライドするか、別のカードをご利用ください。
105	エラーが発生しました。別の支払い方法をご利用ください。
106	暗証番号誤りのため取引に失敗しました。再度お試しください。
107	NFC 決済はご利用できません。別のカードをご利用ください。
108	取引に失敗しました。クレジットカード決済を指定し、IC 決済対応のカードで再度お試しください。
109	暗証番号を一定回数以上間違えました。カード発行会社にお問い合わせください。
110	消費者事由（残高不足など）により取引に失敗しました。
111	消費者事由（限度額超過など）により処理に失敗しました。
112	消費者様がアプリに登録されているクレジットカードをご確認ください。
113	もう一度、QR コード又はバーコードの読取りを行ってください。
114	会員番号に誤りがあります。会員番号をご確認ください。
115	会員番号の読取りに失敗しました。他のカードをご利用いただくか、別の読取り手段をご利用ください。
116	会員番号の読取りに失敗しました。もう一度お試しいただくか、別の読取り手段をご利用ください。
117	指定した支払方法では決済できません。
118	消費者事由により処理に失敗しました。
119	消費者事由（回数制限など）により処理に失敗しました。
120	消費者様が入力したパスワードが不正です。
121	消費者様が決済中にキャンセルしました。
122	使用できないクーポンです。クーポンをご確認ください。
123	期限切れのプロモーションです。プロモーションの内容をご確認ください。
124	使用できないクーポンです。クーポンをご確認ください。
201	エラーが発生しました。利用可能なカードブランドをご利用ください。
202	決済センターが混雑しています。後ほどお試しください。
203	エラーが発生しました。もう一度お試しください。
204	エラーが発生しました。もう一度お試しください。
205	決済センター側が決済申込を受け付けることができない時間帯であるため、数分後に再度お試しください。
206	マイルの積算に失敗しました。
207	上限額を超える金額は指定できません。
208	金額が大きすぎるため受付できません。
209	決済処理に失敗しました。もう一度お試しください。

コード値	説明
210	決済センターがメンテナンス中です。
211	取引に失敗しました。加盟店様の契約状況等をご確認ください。
301	エラーが発生しました。別のカードをご利用ください。
401	取引の有効期限が切れました。取引をやり直してください。
402	エラーが発生しました。もう一度お試しください。
403	エラーが発生しました。もう一度お試しください。
404	決済待ち時間を超過したため取引が失敗しました。
501	返金処理に失敗しました。
502	対象の取引は既に振込の対象となっています。そのため返金できません。
503	対象の取引は既に振込が完了しています。そのため返金できません。
504	返金の期限が切れました。このお取引を返金できません。
505	返金処理を実行する権限がありません。
507	マイルの積算取消に失敗しました。
508	このマイルの積算は既に取消されています。
509	積算取消の期限が切れました。指定した取引を取消しできません。
510	指定した取引は既に返金（売上取消）されています。
601	取消に失敗しました。
602	取消処理を実行する権限がありません。
603	指定した取引は既に取消されています。
604	取消の実行期限が切れました。指定した取引を取消しできません。
901	エラーが発生しました。もう一度お試しください。
902	指定した取引が存在しません。
903	指定した取引は現在処理中のため失敗しました。
904	指定した取引は実行可能なステータスではありません。
905	ご利用のサービスは未契約です。
906	指定した処理・要求は拒否されました。
991	認証に失敗しました。
997	エラーが発生しました。もう一度お試しください。
998	エラーが発生しました。もう一度お試しください。
999	エラーが発生しました。もう一度お試しください。
A01	ユーザーID もしくはパスワードに誤りがあります。
A02	ユーザーパスワードの有効期限が切れました。パスワードの変更を行ってください。
A03	アカウントがロックされました。パスワードの再設定を行ってください。
B01	新パスワードの書式に誤りがあります。
B02	旧パスワードに誤りがあります。

コード値	説明
B03	過去に使用したパスワードは設定できません。
C01	有効な金額を入力してください。
C02	有効な説明を入力してください。
G01	有効な金額を入力してください。
G02	有効な説明を入力してください。
H01	有効な金額を入力してください。
H02	有効な説明を入力してください。
F01	正しいメールアドレスを入力してください。
1003	指定した mPOS 取引 ID に紐付く取引が存在しません。
1006	ご利用のスマートフォン/タブレットが圏外にあるなどの理由で通信が行えません。
1007	利用可能な決済方法がありません。
1008	指定した決済方法に対応していないカードリーダーです。指定したカードリーダーをご確認ください。
1101	指定した mPOS 取引 ID に紐付く取引の状態が返金できない状態です。
3003	マイクの設定をオンにしてください。
3004	カメラの設定をオンにしてください。
3201	ユーザーID は必須です。
3202	ユーザーパスワードかマーチャントパスワードのどちらかが必須です。
3203	カードリーダーのパラメーターが不正です。
3204	言語情報のパラメーターが不正です。
3205	カード決済3面待ちのパラメーターが不正です。
3300	必須であるリクエストパラメーターが設定されていません。
3301	金額に誤りがあります。
3302	説明に誤りがあります。
3303	金額入力画面表示フラグに誤りがあります。
3304	完了画面表示フラグに誤りがあります。
3305	指定した決済方法は利用できません。
3306	進捗状況ダイアログフラグに誤りがあります。
3307	カードリーダーアプリ更新タイムアウト値に誤りがあります。
3801	カードリーダーに接続できませんでした。
3802	カードリーダーアプリ更新指示に失敗しました。
3803	ご利用のカードリーダーはアプリ更新に対応していません。
3804	ご利用のカードリーダーはアプリ更新に対応していません。
3999	予期せぬエラー
5001	カードリーダー内で処理に失敗しました。もう一度お試しください。
5002	カードリーダー内で処理に失敗しました。別のカードをご利用ください。

コード値	説明
5003	カードリーダーが応答しません。カードリーダーとの接続をご確認ください。
5004	取引に失敗しました。クレジットカード決済を指定し、IC 決済対応のカードで再度お試しください。
5101	ユーザー操作によって処理がキャンセルされました。
5102	取引が拒否されました。もう一度お試しください。
5103	取引がキャンセルされました。
5104	エラーが発生しました。もう一度お試しください。
5105	エラーが発生しました。もう一度お試しください。
5106	IC アプリケーションの選択に失敗しました。
5107	エラーが発生しました。もう一度お試しください。
5108	エラーが発生しました。もう一度お試しください。
5109	エラーが発生しました。もう一度お試しください。
5110	IC カードがブロックされているため IC 決済をご利用できません。カード発行会社にお問い合わせください。
5111	エラーが発生しました。もう一度お試しください。
5112	エラーが発生しました。もう一度お試しください。
5113	エラーが発生しました。もう一度お試しください。
5114	エラーが発生しました。もう一度お試しください。
5115	エラーが発生しました。もう一度お試しください。
5116	エラーが発生しました。もう一度お試しください。
5117	エラーが発生しました。もう一度お試しください。
5118	エラーが発生しました。もう一度お試しください。
5199	エラーが発生しました。もう一度お試しください。
5201	IC 決済はご利用できません。カードをスライドするか、別のカードをご利用ください。
5202	IC 決済未対応の IC カードが挿入されました。別のカードをご利用ください。
5203	取引がキャンセルされました。
5204	非接触（NFC）対応のカードではありません。
9001	ユーザー操作によって処理がキャンセルされました。
9004	取引が拒否されました。もう一度お試しください。
9005	取引が拒否されました。もう一度お試しください。
9006	IC カードがブロックされているため IC 決済をご利用できません。カード発行会社にお問い合わせください。
9008	カメラの起動に失敗しました。
9009	結果を得られませんでした。別途照会を行い、結果をご確認ください。
9010	エラーが発生しました。もう一度お試しください。

第4章　サンプルプログラムの利用

加盟店様アプリに mPOS SDK を組み込むための実装例としては、サンプルプログラムをご参照ください。

4-1 サンプルプログラム動作要件

サンプルプログラムの動作要件としては次の通りです。

■Swift 版サンプルプログラム

- 動作 OS：iOS 11.0 以上
- 開発言語：Swift 5.6 - 5.7
- 統合開発環境（IDE）：Xcode 13.4.x - 14.0.x

■Objective-C 版サンプルプログラム

- 動作 OS：iOS 11.0 以上
- 開発言語：Objective-C
- 統合開発環境（IDE）：Xcode 13.4.x - 14.0.x

4-2 参照ライブラリについて

サンプルプログラムは次のライブラリに依存しています。

- mPOS SDK.framework
- Alamofire.framework
- MastercardSonic.framework

4-3 サンプルプログラム機能概要

サンプルプログラムの機能としては次の通りです。

■Swift 版サンプルプログラム

機能名	対象ソースファイル	備考
Payment	PaymentViewController.swift	決済処理を行うサンプルプログラムです。
Cancel Refund	RefundViewController.swift	取消・返金を行うサンプルプログラムです。
Search	SearchViewController.swift	取引履歴を取得するサンプルプログラムです。 取得結果はテーブル形式の別画面で表示されます。
Report	ReportViewController.swift	取引集計処理を行うサンプルプログラムです。
Mail	MailViewController.swift	指定したメールアドレス宛てにメール送信要求を行うサンプルプログラムです。
Password	PasswordViewController.swift	パスワード変更処理を行うサンプルプログラムです。
ReaderAppUpdateBT	ReaderAppUpdateBTViewController.swift	カードリーダーアプリ更新処理を行うサンプルです。
Setting	SettingViewController.swift	共通ヘッダーパラメーターの内容を設定するサンプルプログラムです。

■Objective-C 版サンプルプログラム

機能名	対象ソースファイル	備考
Payment	PaymentViewController.h PaymentViewController.m	決済処理を行うサンプルプログラムです。
Cancel Refund	RefundViewController.h RefundViewController.m	取消・返金を行うサンプルプログラムです。
Search	SearchViewController.h SearchViewController .m	取引履歴を取得するサンプルプログラムです。 取得結果はテーブル形式の別画面で表示されます。
Report	ReportViewController.h ReportViewController.m	取引集計処理を行うサンプルプログラムです。
Mail	MailViewController.h MailViewController.m	指定したメールアドレス宛てにメール送信要求を行うサンプルプログラムです。
Password	PasswordViewController.h PasswordViewController.m	パスワード変更処理を行うサンプルプログラムです。
ReaderAppUpdateBT	ReaderAppUpdateBTViewController.h ReaderAppUpdateBTViewController.m	カードリーダーアプリ更新処理を行うサンプルです。
Setting	SettingViewController.h SettingViewController.m	共通ヘッダーパラメーターの内容を設定するサンプルプログラムです。

第5章 テストパターン

mPOS ではテストアカウントを利用することにより、決済事業者まで通信を行わず mPOS サービス内での折り返しテストを行うことができます。テストアカウントを使用したときのテストパターンは下記の通りになります。

尚、テストアカウントの利用につきましては弊社サポートまでお問い合わせください。

5-1 カード決済系のテストパターン

カード決済系（クレジットカード決済、NFC 決済、銀聯決済）のテストを行う場合のテストパターンは以下の通りです。

5-1-1 クレジットカード決済処理のテストパターン

5-1-1-1. IC チップを用いた決済処理を行う場合

IC チップを用いたテスト決済処理を行う場合、金額の下 3 桁の値が”990”から”999”である場合は決済処理が失敗します。それ以外の金額を設定された場合は決済処理が成功します。

尚、ご利用の IC チップ搭載のカードによっては署名が求められる場合があります。その場合につきましては「5-1-4 署名処理のテストパターン」も決済処理結果に関係しますため、そちらも併せてご確認ください。

5-1-1-2. IC チップ搭載のカードにある磁気ストライプを用いてテストを行う場合

IC チップ搭載のカードにて磁気ストライプを用いたクレジットカード決済処理を行うと決済が拒否される場合がありますが、テスト用のユーザーID を使用した場合に限り、千の位が”2”となる金額を指定していただくことで、IC チップ搭載のカードでも磁気ストライプによる決済処理を行うことが可能です。開発やテストを行うに当たり IC チップ非搭載のカードをご準備できない場合などにご利用ください。

尚、磁気ストライプを用いた決済処理のテストパターンについては「5-1-1-3 磁気ストライプを用いた決済処理を行う場合」をご確認ください。

5-1-1-3. 磁気ストライプを用いた決済処理を行う場合

磁気ストライプを用いたテスト決済処理を行う場合、mPOS サービスの決済処理内ではテストで使用した実カードのブランドとは別に金額の下１桁の値から処理で使用するカードブランドが決定されます。この時、その決定されたカードブランドが mPOS サービスとして決済利用可能な状態であれば、決済処理が成功します。決済利用できない状態である場合については決済処理が失敗します。また、金額の下 1 桁が下記の表に存在しない場合についても決済処理が失敗します。

尚、「5-1-4 署名処理のテストパターン」も決済処理結果に関係しますため、そちらも併せてご確認ください。

条件	カードブランド
金額の下1桁が”0”	VISA
金額の下1桁が”1”	MASTER
金額の下1桁が”2”	JCB
金額の下1桁が”3”	AMEX
金額の下1桁が”4”	Diners

5-1-1-4. 新韓カード使用の場合

新韓カードを使用してテスト決済処理を行う場合、金額の下 3 桁の値が”989”～”999”である場合は決済処理が失敗します。それ以外の金額を設定された場合は決済処理が成功します。尚、新韓カード取引のテスト決済を行う場合については、実際に新韓カードが必要となります。新韓カードの手配については加盟店様にてご準備ください。

尚、「5-1-4 署名処理のテストパターン」も決済処理結果に関係しますため、そちらも併せてご確認ください。

5-1-2 NFC 決済処理のテストパターン

NFC 決済でのテスト決済処理を行う場合、金額の下 3 桁の値が”990”から”999”である場合は決済処理が失敗します。それ以外の金額を設定された場合は決済処理が成功します。

尚、ご利用の IC チップ搭載のカードによっては署名が求められる場合があります。その場合につきましては「5-1-4 署名処理のテストパターン」も決済処理結果に関係しますため、そちらも併せてご確認ください。

5-1-3 銀聯決済処理のテストパターン

銀聯決済でのテスト決済処理を行う場合、金額の下 3 桁の値が”989”～”999”である場合は決済処理が失敗します。それ以外の金額を設定された場合は決済処理が成功します。また、テスト決済を行う場合は、銀聯カードをご用意していただく必要はなく、銀聯以外のブランドのクレジットカードでもテスト決済処理を実施することが可能です。

尚、「5-1-4 署名処理のテストパターン」も決済処理結果に関係しますため、そちらも併せてご確認ください。

5-1-4 署名処理のテストパターン

金額の下 2 桁が”10”である場合は、署名後の処理が失敗します。それ以外の金額を設定された場合は、署名後の処理が成功します。

5-1-5 取消処理のテストパターン

決済処理時に設定した金額の下 2 桁が”20”である場合は、処理が失敗します。それ以外の金額を設定された場合は処理が成功します。

5-1-6 返金処理のテストパターン

決済処理時に設定した金額の下 2 桁が”20”である場合は、処理が失敗します。それ以外の金額を設定された場合は処理が成功します。

5-2 コード決済系のテストパターン

コード決済系のテストを行う場合のテストパターンは以下の通りです。

5-2-1 Alipay 決済

5-2-1-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “5”, “6”, “7”, “9”のいずれかの場合	処理成功
金額の下1桁が”1”, “2”, “3”, “4”, “8”のいずれかの場合	処理失敗

5-2-1-1. 返金処理のテストパターン

決済処理時に設定した金額の下 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-2-2 Alipay+決済

5-2-2-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “1”, “5”, “9”のいずれかの場合	処理成功
金額の下1桁が”2”, “3”, “4”, “6”, “7”, “8”のいずれかの場合	処理失敗

5-2-2-2. 返金処理のテストパターン

決済処理時に設定した金額の下 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-2-3 WeChat Pay 決済

5-2-3-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “5”, “6”, “7”, “9”のいずれかの場合	処理成功
金額の下1桁が”1”, “2”, “3”, “4”, “8”のいずれかの場合	処理失敗

5-2-3-2. 返金処理のテストパターン

決済処理時に設定した金額の下 1 桁もしくは 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 1 桁が”6”の場合	処理失敗
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-2-4 LINE Pay 決済

5-2-4-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “4”, “5”, “6”, “7”, “9”のいずれかの場合	処理成功
金額の下1桁が”1”, “2”, “3”, “8”のいずれかの場合	処理失敗

5-2-4-2. 返金処理のテストパターン

決済処理時に設定した金額の下 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-2-5 d 払い決済

5-2-5-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “1”, “2”, “3”, “4”, “7”, “9”のいずれかの場合	処理成功
金額の下1桁が”5”, “6”, “8”のいずれかの場合	処理失敗

5-2-5-2. 返金処理のテストパターン

決済処理時に設定した金額の下 1 桁もしくは 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 1 桁が”3”の場合	処理失敗
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-2-6 PayPay 決済

5-2-6-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “2”, “6”, “7”, “9”のいずれかの場合	処理成功
金額の下1桁が”1”, “3”, “4”, “5”, “8”のいずれかの場合	処理失敗

5-2-6-2. 返金処理のテストパターン

決済処理時に設定した金額の下 1 桁もしくは 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 1 桁が”6”, “9”の場合	処理失敗
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-2-7 au PAY 決済

5-2-7-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “1”, “2”, “3”, “4”, “7”, “9”のいずれかの場合	処理成功
金額の下1桁が”5”, “6”, “8”のいずれかの場合	処理失敗

5-2-7-2. 返金処理のテストパターン

決済処理時に設定した金額の下 1 桁もしくは 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 1 桁が”3”の場合	処理失敗
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-2-8 メルペイ決済

5-2-8-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “2”, “6”, “7”, “9”のいずれかの場合	処理成功
金額の下1桁が”1”, “3”, “4”, “5”, “8”のいずれかの場合	処理失敗

5-2-8-2. 返金処理のテストパターン

決済処理時に設定した金額の下 1 桁もしくは 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 1 桁が”6”, “9”の場合	処理失敗
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-2-9 楽天ペイ決済

5-2-9-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “7”, “9”のいずれかの場合	処理成功
金額の下1桁が”1”, “2”, “3”, “4”, “5”, “6”, “8”のいずれかの場合	処理失敗

5-2-9-2. 返金処理のテストパターン

決済処理時に設定した金額の下 1 桁もしくは 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 1 桁が”7”, “9”の場合	処理失敗
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-2-10 J-Coin Pay 決済

5-2-10-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “1”, “9”のいずれかの場合	処理成功
金額の下1桁が”2”, “3”, “4”, “5”, “6”, “7”, “8”のいずれかの場合	処理失敗

5-2-10-2. 返金処理のテストパターン

決済処理時に設定した金額の下 1 桁もしくは 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 1 桁が”1”, “9”の場合	処理失敗
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-2-11 UnionPay QR コード決済

5-2-11-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “1”, “4”, “5”, “9”のいずれかの場合	処理成功
金額の下1桁が”2”, “3”, “6”, “7”, “8”のいずれかの場合	処理失敗

5-2-11-2. 返金処理のテストパターン

決済処理時に設定した金額の下 1 桁もしくは 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 1 桁が”4”, “5”の場合	処理失敗
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-2-12 Smart Code 決済

5-2-12-1. 決済処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “1”, “9”のいずれかの場合	処理成功
金額の下1桁が”2”, “3”, “4”, “5”, “6”, “7”, “8”のいずれかの場合	処理失敗

5-2-12-2. 返金処理のテストパターン

決済処理時に設定した金額の下 1 桁もしくは 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 1 桁が”1”, “9”の場合	処理失敗
金額の下 2 桁が”20”の場合	処理失敗
上記以外の場合	処理成功

5-3 マイル積算系のテストパターン

マイル積算系（ANA マイル積算、ユナイテッド航空マイル積算、その他マイル積算）のテストを行う場合のテストパターンは以下の通りです。

5-3-1 マイル積算処理のテストパターン

金額の下 1 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下1桁が”0”, “3”, “4”, “5”, “6”, “7”, “8”, “9”のいずれかの場合	処理成功
金額の下1桁が”1”, “2”のいずれかの場合	処理失敗

5-3-2 マイル積算取消処理のテストパターン

マイル積算処理時に設定した金額の下 2 桁の値により処理の成否が決定されます。成否の条件については次の表の通りです。

条件	成否
金額の下 2 桁が”20”, “30”, “40”のいずれかの場合	処理失敗
上記以外の場合	処理成功

第6章 補足事項

6-1 App Store に公開される場合の注意事項

mPOS SDK を組み込んだ加盟店様アプリを App Store に公開されることを検討されている場合は、次の点にご注意ください。

6-1-1 PIN パッドありカードリーダーをご利用いただく場合

加盟店様が PIN パッドありカードリーダーをご利用いただく形で mPOS SDK を組み込んだアプリの開発を行い、App Store に申請される場合は、そのアプリが PIN パッドありカードリーダーに対応したアプリケーションとして事前に Apple から MFi ライセンスプログラム認定を受ける必要があります。

この MFi ライセンスプラグラムの認定を受けるための申請作業についてはカードリーダーメーカーが行います。そのため、MFi ライセンスプログラムの申請を依頼される場合につきましては、mPOS テクニカルサポート宛に加盟店様が開発されますアプリの情報も併せてご連携ください。mPOS テクニカルサポートはその申請の依頼を受けましたら、カードリーダーメーカーに対して加盟店様アプリの情報を連携し、MFi ライセンスプログラムの申請の依頼を行います。MFi ライセンスプログラム申請に必要な情報としては次の通りです。

【MFi ライセンスプログラム申請に必要な情報】

（1）アプリケーション名

（2）アプリケーションのバージョン番号

（3）アプリケーションの Bundle Identifier

（4）申請するアプリは新規アプリか既存アプリのどちらであるか。（新規アプリ or 既存アプリ）

（5）アプリケーションの説明

加盟店様アプリが MFi ライセンスプログラムとして認定されましたら、弊社より加盟店様に対して通知致します。その際、PIN パッドありカードリーダーの Production Plan ID(PPID)番号を併せて通知致します。MFi ライセンスプログラムの認定後、加盟店様アプリを App Store に申請される際は、App Review Information の Review Notes (Optional)欄に通知されました PPID 番号を記入し申請を行ってください。（記入例: MFI PPID ******-****)

尚、MFi ライセンスプログラム申請の結果が得られるのに目安として2～3週間程掛かることが想定されます。そのため、加盟店様アプリを App Store に申請される際は MFi ライセンスプログラム申請に掛かる日数も考慮に予定をご調整ください。

6-2 マーチャントパスワードとは

マーチャントパスワードとは加盟店に対して与えられた認証のためのパスワードです。通常はユーザー毎に管理するパスワードを用いて mPOS サービスに対する認証を行いますが、ユーザーが所属している加盟店のマーチャントパスワードを用いることでも mPOS サービスに対する認証を受けることができます。

尚、マーチャントパスワードはユーザーパスワードと異なりパスワードの有効期限がありません。

6-3 レシート印字要件

レシートを発行する場合、決済事業者のレギュレーションとしてレシート印字要件が存在する場合があります。そのため、レシートを発行される場合は次の点をご注意ください。

6-3-1 カード決済（クレジットカード決済・NFC 決済）

クレジットカード決済・NFC 決済にて磁気ストライプ取引（磁気ストライプを用いた決済）、または IC カード取引（IC チップを用いた決済）を行なった場合は次の項目を印字してください。

パラメーター名	磁気	IC	備考
cardType（カードタイプ）	□	○	
emvAppId（EMV アプリケーション ID）	-	○	
emvAtc（EMV アプリケーショントランザクションカウンター）	-	○	
emvAppLabel（EMV アプリケーションラベル）	-	○	
panSeqNumber（カードシーケンス番号）	-	○	

※凡例：“○”：必須、 ”※”：条件付き必須、 ”□”：推奨、 ”-”：対象外

6-3-2 銀聯決済

銀聯決済にて磁気ストライプ取引（磁気ストライプを用いた決済）、または IC カード取引（IC チップを用いた決済）を行なった場合は次の項目を印字してください。

パラメーター名	磁気	IC	備考
cardType（カードタイプ）	□	○	
authCode（承認番号）	□	○	
centerProcessId（決済センター処理 ID）	□	○	
cupRequestTime（銀聯要求送信日時）	□	○	
emvAppId（EMV アプリケーション ID）	-	○	
emvAtc（EMV アプリケーショントランザクションカウンター）	-	○	
emvAppLabel（EMV アプリケーションラベル）	-	○	
panSeqNumber（カードシーケンス番号）	-	○	
tvr（端末検証結果）	-	○	
tsi（トランザクションステータス情報）	-	○	
txnCryptogram（トランザクション暗号文）	-	○	

※凡例：“○”：必須、 ”※”：条件付き必須、 ”□”：推奨、 ”-”：対象外

6-3-3 UnionPay QR コード決済

UnionPay QR コード決済にて決済・返金処理を行なった場合は次の項目を印字してください。

パラメーター名	決済時	返金時	備考
amount（金額）	○	○	
discountAmount（割引金額）	※	※	割引適用時は必須
totalAmount（割引金額）	○	○	
cardNumberToken（カード番号トークン）	○	○	
centerTransactionDatetime（決済センター取引日時）	○	○	
centerRefundDatetime（決済センター返金日時）	-	○	
centerReferenceCode（決済センター照会コード）	※	※	値が存在する場合は必須
centerTransactionType（決済センター取引タイプ）	○	○	
centerAcquirerId（決済センターアクワイアラ ID）	○	○	
centerTerminalId（決済センター端末 ID）	○	○	
centerProcessId（決済センター処理 ID）	○	○	
centerMerchantId（決済センターマーチャント ID）	○	○	
centerMerchantName（決済センターマーチャント名）	○	○	

※凡例：“○”：必須、 ”※”：条件付き必須、 ”□”：推奨、 ”-”：対象外

6-4 自動取消・返金について

取引が与信もしくは仮売上の状態で一定時間経過すると mPOS 決済サーバーにて自動で取消処理、もしくは返金処理が実施され、対象の取引が取消・返金されます。

第7章 改訂履歴

2016/04　Ver1.0.0 リリース

2016/05　Ver1.0.1 リリース

・次の章を対象にレスポンスパラメーターキーが定義されたクラス名の誤りを修正。

「3-3-3 レスポンスパラメーター」「3-4-3 レスポンスパラメーター」「3-5-3 レスポンスパラメーター」

「3-6-3 レスポンスパラメーター」「3-7-3 レスポンスパラメーター」

・「4-3 サンプルプログラム機能概要」：メール送信 API サンプルの対象ソースファイル名の誤りを修正。

2017/05　Ver1.1.0 リリース

・「2-2 推奨環境」

- 開発言語のバージョン、動作 OS のバージョン、統合開発環境のバージョンを更新。

・「2-3 プロジェクト設定について」

- プライバシー設定についての記述を追記。
- mPOS を利用するに当たり ATS(App Transport Security)を無効化する必要がなくなったため、その設定についての記述を削除。

・「3-3-1 API 定義」

- “complete”に対する型の誤りを修正。

・「3-3-3 レスポンスパラメーター」

- “JPO”に対する説明を修正。

・「3-4-3 レスポンスパラメーター」

- 次のパラメーターを追加。

“paymentMethodCode”, “jpo”, “cardNumber”, “authCode”, “ acquirerCode”

・「3-5-3 レスポンスパラメーター」

- “JPO”に対する説明を修正。

・「3-8 エラーコード」：次のエラーコードを追加。

- 110, 111, 205, 404, 1007, 1008, 5203, 5204

・「4-1 サンプルプログラム動作要件」

- 開発言語のバージョン、動作推奨 OS のバージョン、統合開発環境のバージョンを更新。

・「第 5 章 テストパターン」

- 次の項を追記。それに伴い第 5 章の章立てを更新。

*「5-1 クレジットカード決済処理を行う場合のテストパターン」

*「5-2 NFC 決済処理を行う場合のテストパターン」

2017/10　Ver1.1.1 リリース

・第1章の章名を更新。

・「2-2 推奨環境」：推奨環境を更新。

- 開発言語：Swift 3.2、統合開発環境：Xcode 9.0 系に更新。

・「2-3-2 ビルド設定」

- Objective-C のプロジェクトの場合の設定内容として5）を追記。

・「2-3-3 CommonCrypto ライブラリの利用について」を新規追加。

・「3-8 エラーコード」：次のエラーコードを追加。

- 1101

・「6-3 伝票・レシート出力される場合について」の記載内容を更新。

2018/05 Ver1.1.2 リリース

・「2-2 推奨環境」：推奨環境を更新。

- 開発言語のバージョン、統合開発環境のバージョンを更新。

・「2-3-1 依存ライブラリ」

- Alamofire.framework のバージョンを更新。

・「2-5 決済 API 概要」

- 画面名と画面に対する概要を修正。

・「3-3-2 リクエストパラメーター」

- 次のパラメーターを追加。

“paymentMethodCode”

・「3-8 エラーコード」：次のエラーコードを追加。

- 3305

・「4-1 サンプルプログラム動作要件」

- 開発言語のバージョン、動作推奨 OS のバージョン、統合開発環境のバージョンを更新。

2018/10 Ver1.2.0 リリース

・WeChat Pay に関する記述を追加。

・「2-2 推奨環境」

- 統合開発環境のバージョンを更新。

・「3-8 エラーコード」：次のエラーコードを追加。

- 112, 113

・「4-1 サンプルプログラム動作要件」

- 統合開発環境のバージョンを更新。

2018/12 Ver1.3.0 リリース

・取引集計 API に関する記述を追加。

・カードリーダーアプリ更新 API に関する記述を追加。

・位置情報サービス利用廃止に伴い、位置情報サービスに関する記述を削除。

・金額（amount）の最大桁数を 8 桁に修正。

・「2-2 推奨環境」

- 統合開発環境のバージョンを更新。

・「3-10 エラーコード」：次のエラーコードを追加・削除。

- 追加：207, 208, 998, 1003　削除：3001, 3002, 9003

・「4-1 サンプルプログラム動作要件」

- 統合開発環境のバージョンを更新。

2019/04　Ver1.4.0 リリース

・LINE Pay、d 払い、PayPay、ANA マイル積算に関する記述を追加。

・”description（説明）”の最大文字数を 40 文字に修正。

・”aosa（有効オフライン支出額）”に関する記述を削除。

・CommonCrypto ライブラリの設定が不要となったことに伴い、CommonCrypto に関する記述を削除。

・「2-2 推奨環境」

- 統合開発環境のバージョンを更新。

・「3-5-3 レスポンスパラメーター」

- “orderStatus（取引ステータス）”に対する説明を修正。

・「3-10 エラーコード」

- メッセージ欄を削除。

- 次のエラーコードを追加。

* 追加：114, 115, 116, 117, 206, 209, 210, 211, 507, 508, 509

・「4-1 サンプルプログラム動作要件」

- 統合開発環境のバージョンを更新。

・「第 5 章 テストパターン」

- WeChat Pay 決済に関するテストパターンを更新。

- LINE Pay 決済に関するテストパターンを追加。

- d 払い決済に関するテストパターンを追加。

- PayPay 決済に関するテストパターンを追加。

- ANA マイル積算に関するテストパターンを追加。

・「6-3-3 NFC 決済の場合」の節を削除。

2019/09　Ver1.5.0 リリース

・取消 API に関する記述を追加。

・「2-2 推奨環境」

- 統合開発環境のバージョンを更新。

・「2-4 Info.plist の設定」

- プライバシー設定として”Privacy – Bluetooth Always Usage Description”に関する記述を追加。

- 画面の回転に関する記述を追加。

・「3-1-4 共通ヘッダーパラメーター」

- “cardThreeWay（カード決済3面待ち）”を追加。

・「3-3-2 リクエストパラメーター」

- “amount（金額）”を任意項目に修正。

・「3-3-3 レスポンスパラメーター」

- “amcNumber（AMC 番号）”のパラメーター名の誤りを修正。

・「3-5-3 レスポンスパラメーター」

- “amcNumber（AMC 番号）”のパラメーター名の誤りを修正。

・「3-6-3 レスポンスパラメーター」

- “amcNumber（AMC 番号）”のパラメーター名の誤りを修正。

・「3-11 エラーコード」

- 次のエラーコードを追加。

  * 追加：601, 602, 603, 604, 902, 903, 904, 905, 3205

・「4-1 サンプルプログラム動作要件」

- 統合開発環境のバージョンを更新。

・「第 5 章 テストパターン」

- 取消処理のテストパターンを追加。

2020/06 Ver1.6.0 リリース

・銀聯決済に関する記述を追加。

・「2-2 推奨環境」

- 動作 OS を iOS 10.0 以上に更新。
- 開発言語・統合開発環境のバージョンを更新。

・「3-1-4 共通ヘッダーパラメーター」

- 次のパラメーターを追加。

  “cameraFacing”

・「3-3-3 レスポンスパラメーター」

- 次のパラメーターを追加。

  “orderReferenceId”, “cvmr”, “tvr”, “tsi”, “txnCryptogram”, “centerProcessId”, “cupRequestTime”

・「3-5-3 レスポンスパラメーター」

- 次のパラメーターを追加。

  “cardType”, “emvAppId”, “emvAtc”, “emvAppLabel”, “panSeqNumber”, “cvmr”, “tvr”, “tsi”,

  “txnCryptogram”, “centerTradeId”, “centerProcessId”, “cupRequestTime”

・「3-6-3 レスポンスパラメーター」

- 次のパラメーターを追加。

  “orderReferenceId”, “cvmr”, “tvr”, “tsi”, “txnCryptogram”, “centerProcessId”, “cupRequestTime”

・「3-11 エラーコード」

- 次のエラーコードを追加。

  * 追加：510, 906, 9008, 9009, 9010

・「4-1 サンプルプログラム動作要件」

- 動作 OS を iOS 10.0 以上に更新。
- 開発言語・統合開発環境のバージョンを更新。

・「第 5 章 テストパターン」

- “5-1-4 新韓カード使用の場合”を追加。
- “5-3 銀聯決済処理のテストパターン”を追加。

・「6-3 伝票・レシート出力される場合について」

- 銀聯決済の取引に対する記述を追加。

・「6-4 自動取消・返金について」を追加。

2021/06 Ver1.7.0 リリース

・商号変更に伴う内容の更新。

・コード決済、ユナイテッド航空マイル積算、その他マイル積算に関する記述を追加。

・「2-2 推奨環境」

- 動作 OS を iOS 11.0 以上に更新。
- 開発言語・統合開発環境のバージョンを更新。

・「2-3-1 依存ライブラリ」

- 依存ライブラリの設定方法に関する記述を更新。

・「3-3-3 レスポンスパラメーター」

- 次のパラメーターを追加。

  “paymentSource”, “discountAmount”, “totalAmount”, “cardNumberToken”, “cardExpiry”, “centerAcquirerId”, “centerReferenceCode”, “centerTransactionType”, “centerTransactionDatetime”, “centerTerminalId”, “centerMerchantId”, “centerMerchantName”, “cvmType”, “mileageMembershipNo”, “usageType”, “lastUseDate”

- 次のパラメーターを廃止。

  “settledAmount”, “settledScale”, “settledCurrency”, “amcNumber”

・「3-4-3 レスポンスパラメーター」

- 次のパラメーターを追加。

  “cardExpiry”

・「3-5-3 レスポンスパラメーター」

- 次のパラメーターを追加。

  “paymentSource”, “amount” , “discountAmount”, “totalAmount”, “cardNumberToken”, “cardExpiry”, “centerAcquirerId”, “centerReferenceCode”, “centerTransactionType”, “centerTransactionDatetime”, “centerRefundDatetime”, “centerTerminalId”, “centerMerchantId”, “centerMerchantName”, “cvmType”, “mileageMembershipNo”

- 次のパラメーターを廃止。

  “amcNumber”

・「3-6-3 レスポンスパラメーター」

- 次のパラメーターを追加。

  “paymentSource”, “discountAmount”, “totalAmount”, “cardNumberToken”, “cardExpiry”, “centerAcquirerId”, “centerReferenceCode”, “centerTransactionType”, “centerTransactionDatetime”, “centerRefundDatetime”, “centerTerminalId”, “centerMerchantId”, “centerMerchantName”, “cvmType”, “mileageMembershipNo”, “usageType”, “lastUseDate”

- 次のパラメーターを廃止。

  “settledAmount”, “settledScale”, “settledCurrency”, “amcNumber”

・「3-11 決済方法コード」を追加。

・「3-12 支払手段」を追加。

・「3-13 支払種別」を追加。

・「3-14 カード所有者検証方法タイプ」を追加。

・「3-15 エラーコード」

- 次のエラーコードを追加。

  * 追加：118, 119, 120, 121, 997

・「4-1 サンプルプログラム動作要件」

- 動作 OS を iOS 11.0 以上に更新。

- 開発言語・統合開発環境のバージョンを更新。

・「第 5 章 テストパターン」

- 章立てを見直し。

- Alipay, WeChat Pay, LINE Pay, d 払い, PayPay のテストパターンを更新。

- Alipay+, au PAY, メルペイ, 楽天ペイ, J-Coin Pay, UnionPay QR コードのテストパターンを追加。

・「6-3 レシート印字要件」

- タイトルを「伝票・レシート出力される場合について」から変更。

- 章立てを見直し。

- UnionPay QR コード決済に関する記述を追加。

2021/10 Ver1.8.0 リリース

・「2-2 推奨環境」

- 開発言語・統合開発環境のバージョンを更新。

・「3-11 決済方法コード」

- “Smart Code 決済”を追加。

・「3-12 支払手段」

- 次の値を追加。

  GINKOUPAY, KPLUS, EPOSPAY, PRING, ATONE, AFTEE, PREMOCODE, MYJCBPAY, ANAPAY, FAMIPAY, LALAPAY, PAIDY

・「3-15 エラーコード」

- 次のエラーコードを追加。

  * 追加：122, 123, 124

・「4-1 サンプルプログラム動作要件」

- 開発言語・統合開発環境のバージョンを更新。

・「第 5 章 テストパターン」

- Smart Code 決済に関するテストパターンを追加。

2022/11 Ver1.9.0 リリース

・「2-2 推奨環境」

- 開発言語・統合開発環境のバージョンを更新。

・「2-3-1 依存ライブラリ」

- 依存ライブラリとして MastercardSonic.framework を追加。

・「4-1 サンプルプログラム動作要件」

- 開発言語・統合開発環境のバージョンを更新。

・「4-2 参照ライブラリについて」

- 依存ライブラリとして MastercardSonic.framework を追加。