

IboxPro API

Руководство по интеграции

V 1.6.38

История изменений

Версия	Дата	Описание
1.0.0	26.05.2015	Исходная версия
1.1.0	09.07.2015	Добавлен API истории платежей. Изменения в структуре библиотеки.
1.1.1	27.07.2015	Добавлен ReaderEvent. WAITING_FOR_CARD_CANCELED для PaymentController. Изменения в TransactionItem, TaxItem
1.2	11.08.2015	Добавлены: ReaderEvent. PAYMENT_CANCELED, ReaderEvent.EJECT_CARD. Удалены: ReaderEvent. WAITING_FOR_CARD_CANCELED, PaymentError.PAYMENT_IN_PROGRESS. Изменена сигнатура некоторых методов PaymentController. В PaymentControllerListener добавлен метод onScheduleCreationFailed. Правки внутренней логики
1.3	10.09.2015	Добавлена возможность отмены/возврата платежа, получение транзакции по ее ID, PaymentResultContext, ScheduleItem. Правки внутренней логики
1.3.1	11.09.2015	Прерывание транзакции. Правки внутренней логики
1.3.2	14.09.2015	Исправления в поведении PaymentController и считывателя карт при отмене/возврате платежа и отмене транзакции. Добавлено свойство PaymentContext.currencyName. Удалено свойство PaymentContext.Type
1.3.3	25.09.2015	Добавлены новые события BAD_SWIPE и LOW_BATTERY считывателя карт. Исправления внутренней логики
1.3.4	30.09.2015	Добавлен метод PaymentControllerListener.onTransactionStarted()
1.3.5	30.05.2016	Добавлен enum Currency, исправление ошибок округления

Версия	Дата	Описание
1.3.7	22.07.2016	Добавлены частичные отмены/возвраты, автоматическая конфигурация считывателей карт. Однофакторная авторизация. Исправления внутренней логики
1.3.8	28.07.2016	Добавлены признаки возможности частичной отмены/возврата транзакции
1.3.9	02.08.2016	Добавлены свойства класса TransactionItem
1.4.0	09.08.2016	Исправления внутренней логики
1.4.1	11.08.2016	Добавлены оплата наличными и отправка данных фискального регистратора
1.4.2	18.08.2016	Добавлен callback для получения уровня заряда считывателя карт в интерфейсе PaymentControllerListener
1.4.3	29.09.2016	Исправления внутренней логики
1.5.0	18.11.2016	Добавлена поддержка новых считывателей карт, добавлена поддержка NFC. Изменения в интерфейсе.
1.5.1	1.12.2016	Исправления логики регулярных платежей, исправление ошибок округления, улучшения работы ридеров QPOS. Добавлена отмена предоплат
1.5.2	11.01.2017	Ридер QPOS_MINI теперь поддерживает соединение по USB и оплату NFC. Улучшения внутренней логики.
1.5.3	16.01.2017	Добавлена поддержка ридера WISEPAD2_PLUS. Добавлен метод PaymentController.printText(), добавлен enum PaymentController.PrintResult.
1.5.3.1	23.01.2017	Добавлены поля CardholderName и TerminalName в TransactionItem
1.5.3.5	16.03.2017	Исправления внутренней логики
1.5.3.8	30.03.2017	Поддержка NFC для ридера QPOS Mini. Улучшения внутренней логики

Версия	Дата	Описание
1.5.3.9	05.04.2017	Исправлена работа вызова PaymentControllerListener.onSelectApplication().
1.5.4.0	12.04.2017	Добавлена поддержка ридеров M17 Изменены названия ридеров Убрана поддержка неиспользуемых ридеров Убрана поддержка автоконфигурации Общие улучшения
1.5.4.1	18.04.2017	Исправление падения для устройств, не оснащенных Bluetooth
1.5.4.3	30.06.2017	Добавлена перегрузка PaymentController.submitFiscal() для отправки фискальных данных по стандарту Ф3-54
1.5.5.3	25.07.2017	Добавлена возможность проведения отмены через EMV/NFC. Добавлен метод auth() для проверки учетных данных и получения информации об учетной записи. Общие улучшения.
1.5.6.1	22.09.2017	Добавлена оплата по ссылке, обновлена процедура платежа, добавлен enum PaymentMethod, удален enum TransactionItem.InputType, добавлена установка банка(acquirer) для платежа, общие улучшенияsss
1.5.6.5	13.10.2017	Исправлено проведение EMV/NFC транзакций для ридера QPOS. Добавлено свойство PaymentController.ClientProductCode. Обновлена сигнатура методов PaymentController.reversePayment(). Добавлены новые свойства для PaymentContext, TransactionItem и TransactionItem.Card
1.5.6.8	31.10.2017	PaymentControllerListener.OnTransactionStarted() теперь вызывается и для отмены/возврата платежа. Общие улучшения
1.5.6.9	01.11.2017	Данные для отображения QR в ExternalPayment теперь могут приходить в количестве более одного кода

Версия	Дата	Описание
1.5.7.0	02.11.2017	Сумма транзакций по ридеру ограничена 1 000 000. Сумма для возврата/отмены не может превышать остаток транзакции. Добавлена ошибка INVALID_AMOUNT
1.5.7.1	06.11.2017	Добавлены транзакции «В обработке» для APIGetHistoryResult
1.5.7.3	09.11.2017	Добавлена поддержка TLSv1.2
1.5.7.6	22.11.2017	Добавлен ИНН пользователя в APIAuthResult
1.5.7.6	10.01.2018	Исправление ошибок
1.5.8.1	15.01.2018	Добавлена поддержка ридера C15
1.5.8.3	02.02.2018	Добавлена поддержка привязанных карт
1.5.8.5	08.02.2018	Добавлена поддержка автоконфигурации
1.5.8.6	13.02.2018	Добавлены поля: PaymentResultContext.AttachedCard LinkedCard.PanMasked
1.5.8.7	16.02.2018	Добавлен метод PaymentController.balanceInquiry(); Добавлено поле TransactionItem.Balance; Исправление ошибок
1.5.8.8	23.02.2018	Добавлена поддержка установки банка для методов PaymentController.addLinkedCard() и PaymentController.balanceInquiry(). Добавлено поле Account.PaymentOptions. Исправлен ошибочный запрос карты ридером при попытке возврата наличных, если ридер еще не был подключен ранее.
1.5.8.9	27.02.2018	Добавлено поле LinkedCard.Balance
1.5.9.0	06.03.2018	Добавлен тип оплаты «Внешний POS-терминал», добавлена поддержка чиповых и NFC регулярных платежей. Исправление ошибок.

Версия	Дата	Описание
1.5.9.1	07.03.2018	Транзакции теперь могут быть отменены в режиме CNP, добавлены поля TransactionItem.CanCancelCNP, TransactionItem.CanCancelCNPpartial. Добавлен callback PaymentControllerListener.onSwitchedToCNP(). Исправлена ошибка отмена/возвраты транзакций, оплаченных способом «Внешний POS-терминал»
1.5.9.2	13.03.2018	Добавлен класс ReversePaymentContext, добавлено поле PaymentContext.ExtID, исправлена ошибка, когда APIResult.getErrorMessage() возвращал строку "null" вместо пустой строки
1.5.9.3	15.03.2018	Добавлено свойство PaymentController.RepeatOnError, исправление ошибок
1.5.9.5	27.03.2018	Добавлен поиск транзакций по RRN, добавлен метод PaymentController.submitEmailNotification(). Добавлены новые поля в Account, TransactionItem и LinkedCard. Исправлена ошибка при частичном возврате для уже частично отмененного платежа.
1.5.9.7	10.04.2018	Добавлен класс FiscalInfo. Улучшена работа ридеров QPOS. Исправление ошибок.
1.5.9.8	12.04.2018	Добавлено очищение контекста операции после успешного выполнения и при вызове метода PaymentController.disable()
1.5.9.9	17.04.2018	Исправлена ошибка отсутствия запроса карты при возврате true из PaymentControllerListener.onCancellationTimeout() при PaymentController.RepeatOnError=true

Версия	Дата	Описание
1.5.10	26.04.2018	Добавлено событие CARD_TIMEOUT. Общие улучшения.
1.5.12	01.06.2018	Общие улучшения
1.5.18	03.07.2018	Добавлена поддержка устройства Urovo. Добавлены методы для прошивки и конфигурации ридера (только Urovo). Добавлены классы Purchase, Tax и TaxContribution. Добавлены поля для печати данных фискального чека в Transaction. Исправлена возможность проведения CNP отмены для не карточных платежей, исправлены пустые поля в FiscalInfo. Общие улучшения
1.5.19	04.07.2018	Изменена процедура работы с ридерами – теперь перед выполнением операции после вызова PaymentController.enable() необходимо дождаться события ReaderEvent.INIT_SUCCESSFULLY. Общие улучшения
1.5.20	04.07.2018	Исправление ошибок
1.5.22	06.07.2018	Добавлена проверка Amount <= Summ(AuxData) в методе PaymentController.startPayment(). Исправление ошибок
1.5.23	09.07.2018	Исправлена ошибка при создании регулярного платежа. Добавлен метод PaymentController.tryGetFiscalInfo()
1.5.24	11.07.2018	Добавлен способ выполнения платежа «Предоплата» (PaymentMethod.PREPAID), улучшена работа ридера P17, исправление ошибок
1.5.25	17.07.2018	Добавлена поддержка устройств АЗУР, добавлена ошибка PaymentError.STANDALONE_FAILED, добавлены поля PaymentContext.Ern и ReversePaymentContext.Ern, исправление ошибок

Версия	Дата	Описание
1.5.26	30.07.2018	Исправлено не переподключение ридера P17 в некоторых случаях
1.5.27	01.08.2018	Улучшена работа методов PaymentController.onCreate, PaymentController.onSaveInstanceState. Исправлен Account.getAcquirersByMethods(), исправление ошибок отмены и отображения платежей с помощью OUTER_CARD
1.5.28	03.08.2018	Добавлена операция «Сверка», исправление ошибок
1.5.29	07.08.2018	Сигнатуры AZUR и Standlalone заменены на ТТК, исправление настроек ТТК
1.5.30	06.09.2018	Добавлено поле SuppressSignatureWaiting для классов PaymentContext и CancelPaymentContext
1.5.31	11.09.2018	Добавлена поддержка устройств Sunmi. Добавлено событие ReaderEvent.PIN_TIMEOUT. Реализована отправка дополнительных данных при использовании ТТК. Исправление ошибок
1.5.32	24.09.2018	Добавлена поддержка кредит-ваучера(ReversePaymentContext). Исправление ошибок
1.5.33	01.11.2018	Теперь платежи(отмены платежей) не могут быть проведены NFC, если сумма в рублях превышает 10000.
1.5.34	07.11.2018	Обновлен набор библиотек, необходимых для запуска. Обновлен драйвер устройств Sunmi.
1.5.35	15.01.2019	Исправлена работа возвратов для устройств, работающих по протоколу ТТК. Улучшена работа ридеров P17 и SUNMI. Обновлен протокол связи.

Версия	Дата	Описание
1.5.36	23.01.2019	Добавлена расширенная поддержка пользовательских продуктов: PaymentProductItem и PaymentProductItemField. Подробнее о заполнении продуктов – см. описание PaymentContext.prepare() и соответствующих классов. Добавлен метод PaymentContext.prepare(). Добавлена константа для НДС20% в Purchase.
1.6.37	01.02.2019	Добавлена поддержка платежей через стандартный NFC интерфейс – ридер NFC(при использовании ознакомьтесь с обновленным описанием PaymentController.setReaderType). Изменена процедура прошивки ридеров – добавлен метод PaymentController.readerSetConfig(). Добавлен метод PaymentController.setCustomReaderParams(). Добавлен метод PaymentController.readerBeep(). minSdkVersion увеличен до 19.
1.6.38	04.02.2019	Добавлена поддержка тегов для товаров согласно ФФД 1.05 (см. Purchase). Добавлен метод PaymentController.fiscalize().

Содержание

История изменений	2
Обзор	12
Разрешения и зависимости Android	13
Пакет <code>ibox.pro.sdk.external</code>	14
Класс <code>PaymentController</code>	14
Интерфейс <code>PaymentControllerListener</code>	30
Класс <code>PaymentContext</code>	34
Класс <code>RegularPaymentContext</code>	36
Класс <code>ReversePaymentContext</code>	38
Класс <code>PaymentResultContext</code>	39
Класс <code>AbstractEntity</code>	40
Пакет <code>ibox.pro.sdk.external.entities</code>	41
Класс <code>TransactionItem</code>	41
Класс <code>TransactionItem.Card</code>	44
Класс <code>TransactionItem.ExternalPayment</code>	46
Класс <code>TransactionItem.FiscalInfo</code>	47
Класс <code>TransactionItem.Product</code>	48
Класс <code>TransactionItem.ProductField</code>	49
Класс <code>ScheduleItem</code>	50
Класс <code>Account</code>	51
Класс <code>Account.PaymentOption</code>	52
Класс <code>LinkedCard</code>	53
Класс <code>Purchase</code>	54
Класс <code>Tax</code>	56
Класс <code>TaxContribution</code>	57
Класс <code>PaymentProductItem</code>	58
Класс <code>PaymentProductItemField</code>	60
Класс <code>PreparedField</code>	61
Класс <code>APIResult</code>	62
Класс <code>APIGetHistoryResult</code>	63
Класс <code>APIAuthResult</code>	64
Класс <code>APIReadLinkedCardsResult</code>	65
Класс <code>APITryGetPaymentStatusResult</code>	66
Класс <code>SettlementResult</code>	67

Класс APIPrepareResult	68
Пакет ibox.pro.sdk.external.ui	69
Класс SignatureView.....	69
Приложение 1: Печать слипа	70
Приложение 2: Пример чека	71
Приложение 3: Некоторые коды ошибок для ридеров, работающих по протоколу ТТК	72

Обзор

Разрешения и зависимости Android

Минимальная версия Android API – 19. Перед началом работы с библиотекой в файл **AndroidManifest.xml** приложения необходимо добавить следующие строки:

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.MODIFY_AUDIO_SETTINGS" />
<uses-permission android:name="android.permission.ACTION_HEADSET_PLUG" />
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

Для работы на устройствах Urovo необходимо дополнительно добавить строки:

```
<uses-permission android:name="android.permission.SYSTEM_OVERLAY_WINDOW" />
<uses-permission
android:name="android.permission.ACTION_MANAGE_OVERLAY_PERMISSION" />
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />
<uses-permission android:name="android.permission.BROADCAST_STICKY" />
<uses-permission android:name="android.permission.VIBRATE" />
<uses-permission android:name="android.permission.FLASHLIGHT" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
<uses-permission android:name="android.permission.DISABLE_KEYGUARD" />
```

Для работы на устройствах Sunmi необходимо дополнительно добавить строки:

```
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

Для работы через встроенный NFC-интерфейс необходимо дополнительно добавить строки:

```
<uses-permission android:name="android.permission.NFC" />
<uses-feature android:name="android.hardware.nfc" android:required="true" />
```

Пакет `ibox.pro.sdk.external`

Класс `PaymentController`

Данный класс является центральным в библиотеке. Содержит методы для создания транзакций и передачи в них дополнительных параметров а также инкапсулирует работу со считывателями карт. Класс содержит наборы различных параметров в виде **enum**, необходимых для проведения платежа.

Перед проведением транзакций необходимо установить пользовательские Email и пароль, необходимые для аутентификации с помощью метода **setCredentials** и установить тип считывателя карт с помощью метода **setReaderType**.

Перед началом работы со считывателем карт необходимо вызвать метод **enable**, а после завершения работы – метод **disable**. Вызов метода **disable** или отключение считывателя карт прервет обработку текущей транзакции. Перед началом выполнения операции необходимо дождаться события `ReaderEvent.INIT_SUCCESSFULLY`.

Для корректной работы также необходимо вызывать одноименные методы экземпляра класса при вызове в родительской **Activity** следующих методов: **onCreate**, **onDestroy**, **onSaveInstanceState**. Таймаут для извлечения карты при сбое транзакции – 5 секунд.

Для обработки событий считывателя карт и/или процесса выполнения транзакции экземпляру класса может быть передан **PaymentControllerListener**, с помощью метода **setPaymentControllerListener**.

При указании суммы платежа, разрядность десятичной части которой превышает разрядность десятичной части валюты, количество знаков после запятой такой суммы будет урезано **без округления**.

Наборы параметров:

`ReaderType`

Набор поддерживаемых типов считывателей карт

Тип	Описание
P15	Считыватель карт «Chip&Pin», P15
P16PLUS	Считыватель карт «Chip&Pin», P16 Plus
M17	
P17	Считыватель карт «Chip&Pin NFC», P17
UROVO	Устройство Urovo i9000s
TTK	Устройство, работающее по протоколу TTK
SUNMI	Устройство Sunmi P1
NFC	Встроенный NFC-интерфейс

ReaderEvent

Набор возможных событий, которые могут быть переданы считывателем карт

Тип	Описание
CONNECTED	Считыватель карт был подключен
DISCONNECTED	Считыватель карт был отключен
START_INIT	Начало инициализации
INIT_SUCCESSFULLY	Инициализация завершена успешно
INIT_FAILED	Произошла ошибка инициализации
EJECT_CARD_TIMEOUT	Не используется
SWIPE_CARD	Обнаружено проведение магнитной полосой
EMV_TRANSACTION_STARTED	Начата чиповая транзакция
NFC_TRANSACTION_STARTED	Начата NFC транзакция
WAITING_FOR_CARD	Ожидание проведения магнитной полосой или вставки чиповой карты
PAYMENT_CANCELED	Платеж отменен пользователем
EJECT_CARD	Пользователь может извлечь карту (возникает при ошибке проведения транзакции)
BAD_SWIPE	Не удалось считать данные магнитной ленты
LOW_BATTERY	Уровень заряда батареи считывателя карт менее 10%
CARD_TIMEOUT	Превышен таймаут ожидания карты
PIN_TIMEOUT	Превышен таймаут ожидания ввода PIN

PaymentInputType

Набор возможных типов оплаты

Тип	Описание
SWIPE	Оплата с помощью проката карты магнитной лентой
CHIP	Оплата с помощью чипа на карте
NFC	Оплата NFC
PREPAID	Предоплата
CREDIT	Оплата кредитом
CASH	Оплата наличными
OTHER	Оплата по ссылке
OUTER_CARD	Оплата внешним POS-терминалом

PaymentError

Набор возможных ошибок, которые могут возникнуть в процессе выполнения платежа

Тип	Описание
CONNECTION_ERROR	Ошибка соединения с сервером
SERVER_ERROR	Ошибка выполнения транзакции
TRANSACTION_NULL_OR_EMPTY	Ошибка создания транзакции
NO_SUCH_TRANSACTION	Транзакция не была найдена, либо не уникальна
EMV_ERROR	Общая ошибка EMV

Тип	Описание
EMV_TERMINATED	Транзакция прервана
EMV_DECLINED	Транзакция отклонена
EMV_CANCEL	Транзакция отменена
EMV_CARD_ERROR	Ошибка карты
EMV_DEVICE_ERROR	Ошибка ридера
EMV_CARD_NOT_SUPPORTED	Карта не поддерживается
EMV_NOT_ALLOWED	Чиповая транзакция не разрешена
EMV_ZERO_TRAN_EMV	Попытка провести транзакцию на нулевую сумму
NFC_NOT_ALLOWED	NFC транзакция не разрешена
INVALID_AMOUNT	Сумма возврата/отмены превышает остаток транзакции
STANDALONE_FAILED	Ошибка выполнения платежа по ТТК протоколу

RegularRepeatType

Набор возможных типов регулярного платежа

Тип	Описание
Never	Платеж будет выполнен один раз
Weekly	Еженедельный платеж
Monthly	Ежемесячный платеж
Quarterly	Ежеквартальный платеж
Annual	Ежегодный платеж
ArbitraryDays	Платеж будет выполняться в заданные дни

RegularEndType

Набор возможных способов окончания выполнения регулярного платежа

Тип	Описание
BY_QUANTITY	Окончание по количеству повторов
BY_DAY	Окончание в заданный день

ReverseAction

Набор возможных способов отмены платежа

Тип	Описание
CANCEL	Отмена платежа
RETURN	Возврат платежа

Currency

Валюты, которыми можно провести оплату

Тип	Описание
RUB	Российский рубль
VND	Вьетнамский донг

PrintResult

Набор возможных результатов выполнения печати

Тип	Описание
SUCCESS	Печать успешно завершена
NO_PAPER	Нет бумаги
WRONG_CMD	Неправильная команда
OVERHEAT	Перегрев печатающей головки
TIMEOUT	Превышение таймаута ожидания ответа
PRINTER_ERROR	Ошибка принтера

PaymentMethod

Набор возможных способов выполнения платежа

Тип	Описание
CARD	Платежной картой
CREDIT	Кредитом
CASH	Наличные
OTHER	По ссылке
LINKED_CARD	Привязанной картой
OUTER_CARD	Внешним POS-терминалом
PREPAID	Предоплатой

Методы класса:

getInstance

Сигнатура	PaymentController getInstance()
Входные параметры	Нет
Возвращаемое значение	Экземпляр класса
Описание	Метод для получения экземпляра класса

onCreate

Сигнатура	void onCreate(Context context, Bundle savedInstanceState)
Входные параметры	context – контекст приложения savedInstanceState – передается из метода родительской Activity
Возвращаемое значение	Нет
Описание	Должен быть вызван при вызове одноименного метода родительской Activity

onDestroy

Сигнатура	void onDestroy()
Входные параметры	Нет
Возвращаемое значение	Нет

Описание	Должен быть вызван при вызове одноименного метода родительской Activity
----------	---

onSaveInstanceState

Сигнатура	void onSaveInstanceState(Context context, Bundle savedInstanceState)
Входные параметры	savedInstanceState – передается из метода родительской Activity
Возвращаемое значение	Нет
Описание	Должен быть вызван при вызове одноименного метода родительской Activity

enable

Сигнатура	void enable()
Входные параметры	Нет
Возвращаемое значение	Нет
Описание	Начинает работу со считывателем карт

disable

Сигнатура	void disable()
Входные параметры	Нет
Возвращаемое значение	Нет
Описание	Завершает работу со считывателем карт

isConnected

Сигнатура	boolean isConnected()
Входные параметры	Нет
Возвращаемое значение	true , если считыватель карт подключен
Описание	Используется для проверки готовности считывателя карт

getBluetoothDevices

Сигнатура	ArrayList<BluetoothDevice> getBluetoothDevices(Context context)
Входные параметры	context – контекст приложения
Возвращаемое значение	ArrayList сопряженных устройств
Описание	Используется для получения набора доступных для соединения Bluetooth-устройств

setPaymentControllerListener

Сигнатура	void setPaymentControllerListener(PaymentControllerListener listener)
Входные параметры	listener – обработчик событий
Возвращаемое значение	Нет
Описание	Задаёт новый обработчик событий проведения платежа

setCredentials

Сигнатура	void setCredentials(String email, String password)
Входные параметры	email – email пользователя password – пароль пользователя
Возвращаемое значение	Нет
Описание	Задаёт данные пользователя, необходимые для проведения транзакций

auth

Сигнатура	APIAuthResult auth(Context context)
Входные параметры	context – контекст приложения
Возвращаемое значение	APIAuthResult
Описание	Проверяет правильность введенных учетных данных и возвращает информацию об учетной записи

setReaderType

Сигнатура	void setReaderType(Context context, ReaderType readerType, String address, String config) throws IllegalStateException
Входные параметры	context – контекст приложения. При <i>readerType == NFC</i> необходимо передавать activity readerType – тип считывателя карт address – MAC-адрес Bluetooth-считывателя карт. Для подключения USB передавать константу PaymentController.USB_MODE_KEY config – параметры конфигурации считывателя карт
Возвращаемое значение	Нет
Описание	Изменяет тип текущего считывателя карт. При попытке изменить тип считывателя во время проведения платежа будет сгенерировано исключение IllegalStateException

getReaderType

Сигнатура	ReaderType getReaderType()
Входные параметры	Нет
Возвращаемое значение	Текущий тип считывателя карт
Описание	Возвращает текущий тип считывателя карт

startPayment

Сигнатура	void startPayment(Context context, PaymentContext paymentContext) throws PaymentException
Входные параметры	context – контекст приложения paymentContext – данные платежа
Возвращаемое значение	Нет
Описание	Начинает выполнение платежа. При неверных параметрах платежа или при попытке начать новый платеж/отмену платежа до окончания будет сгенерировано исключение PaymentException

reversePayment

Сигнатура	void reversePayment(Context context, String transactionID, ReverseAction action, Double amountToReverse, Currency currency, String receiptPhone, String receiptEmail) throws PaymentException
Входные параметры	context – контекст приложения transactionID – ID транзакции отменяемого платежа action – Тип отмены amountToReverse – сумма, на которую будет выполнена отмена. Для полной отмены передавать null currency – валюта, используемая для отмены/возврата receiptPhone – номер телефона для отправки чека receiptEmail – email для отправки чека
Возвращаемое значение	Нет
Описание	Устаевший. См. reversePayment(Context context, ReversePaymentContext reversePaymentContext)

reversePayment

Сигнатура	void reversePayment(Context context, ReversePaymentContext reversePaymentContext) throws PaymentException
Входные параметры	context – контекст приложения reversePaymentContext – данные для проведения отмены/возврата
Возвращаемое значение	Нет
Описание	Начинает выполнение отмены платежа. При попытке начать новый платеж/отмену платежа до окончания будет сгенерировано исключение PaymentException

adjust

Сигнатура	APIResult adjust(Context context, String transactionID, String receiptPhone, String receiptEmail, byte [] signature)
Входные параметры	context – контекст приложения transactionID – ID транзакции, для которой требуется отправить дополнительные данные receiptPhone – номер телефона для отправки чека receiptEmail – email для отправки чека signature – изображение с подписью плательщика
Возвращаемое значение	Результат отправки данных
Описание	Используется для отправки подписи и чека для транзакции единичного платежа

adjust

Сигнатура	APIResult adjust(Context context, int regularID, byte [] signature)
Входные параметры	context – контекст приложения regularID – ID транзакции, для которой требуется отправить дополнительные данные signature – изображение с подписью плательщика
Возвращаемое значение	Результат отправки данных
Описание	Используется для отправки подписи и чека для транзакции регулярного платежа

adjustReverse

Сигнатура	APIResult adjustReverse(Context String transactionID, String receiptPhone, String receiptEmail, byte [] signature)
Входные параметры	context – контекст приложения transactionID – ID транзакции, для которой требуется отправить дополнительные данные receiptPhone – номер телефона для отправки чека receiptEmail – email для отправки чека signature – изображение с подписью плательщика
Возвращаемое значение	Результат отправки данных
Описание	Используется для отправки подписи и чека для транзакции отмены платежа

isPaymentInProgress

Сигнатура	boolean isPaymentInProgress()
Входные параметры	Нет
Возвращаемое значение	true , если выполнение платежа не завершено
Описание	Используется для проверки состояния контроллера

getHistory

Сигнатура	APIGetHistoryResult getHistory(Context context, int page)
Входные параметры	context – контекст приложения page – номер страницы
Возвращаемое значение	Объект APIGetHistoryResult, содержащий набор транзакций
Описание	Позволяет получить историю транзакций в постраничном виде

getTransactionByID

Сигнатура	APIGetHistoryResult getTransactionByID(Context context, String transactionID)
Входные параметры	context – контекст приложения transactionID – ID запрашиваемой транзакции
Возвращаемое значение	Объект APIGetHistoryResult, содержащий запрашиваемую транзакцию
Описание	Позволяет получить данные транзакции по ее ID

getTransactionByRRN

Сигнатура	APIGetHistoryResult getTransactionByRRN(Context context, String rrn)
Входные параметры	context – контекст приложения rrn – Retrieval Reference Number
Возвращаемое значение	Объект APIGetHistoryResult, содержащий запрашиваемую транзакцию
Описание	Позволяет получить данные транзакции по ее RRN

getTransactionByInvoice

Сигнатура	APIGetHistoryResult getTransactionByInvoice(Context context, String invoice)
Входные параметры	context – контекст приложения invoice – номер чека
Возвращаемое значение	Объект APIGetHistoryResult, содержащий запрашиваемую транзакцию
Описание	Позволяет получить данные транзакции по номеру чека

setSingleStepEMV

Сигнатура	void setSingleStepEMV(boolean singleStepEMV)
Входные параметры	singleStepEMV – признак однофакторной авторизации
Возвращаемое значение	Нет
Описание	Позволяет проводить платежи с однофакторной авторизацией

getSingleStepEMV

Сигнатура	boolean isSingleStepEMV()
Входные параметры	Нет
Возвращаемое значение	Признак режима однофакторной авторизации
Описание	Возвращает признак режима однофакторной авторизации

submitFiscal

Сигнатура	APIResult submitFiscal(Context context, String transactionID, int printerID, int docID, int CVC, int shift)
Входные параметры	context – контекст приложения transactionID – ID транзакции, для которой требуется отправить фискальные данные printerID – ID фискального регистратора docID – сквозной номер документа CVC – КПК документа shift – номер операционной смены
Возвращаемое значение	Результат отправки данных
Описание	Выполняет отправку данных фискализации

submitFiscal

Сигнатура	APIResult submitFiscal(Context context, String transactionID, String printerID, int 2300lea, int docSN, String fdn, String fdm, String fs, Date fdt)
Входные параметры	context – контекст приложения transactionID – ID транзакции, для которой требуется отправить фискальные данные printerID – ID фискального регистратора shift – номер операционной смены docID – сквозной номер документа fdn – фискальный номер документа fdm – фискальный признак документа fs – фискальное хранилище fdt – дата и время проведения фискальной операции
Возвращаемое значение	Результат отправки данных
Описание	Выполняет отправку данных фискализации согласно стандарту Ф3-54

printText

Сигнатура	PrintResult printText(String text, Layout.Alignment alignment) throws IllegalStateException
Входные параметры	text – Текст для печати alignment – Выравнивание текста
Возвращаемое значение	Результат печати

Описание	Команда работает только для ридера WISEPAD2_PLUS, иначе будет сгенерировано исключение <code>IllegalStateException</code>
----------	---

setClientProductCode

Сигнатура	<code>void setClientProductCode(String clientProductCode)</code>
Входные параметры	Код клиентского приложения
Возвращаемое значение	Нет
Описание	Устанавливает код клиентского приложения

getClientProductCode

Сигнатура	<code>String getClientProductCode()</code>
Входные параметры	Нет
Возвращаемое значение	Установленный код клиентского приложения
Описание	Возвращает устанавливает код клиентского приложения

addLinkedCard

Сигнатура	<code>void addLinkedCard(Context context, PaymentController.Currency currency, String acquirerCode)</code> throws <code>PaymentException</code>
Входные параметры	context – контекст приложения currency – валюта карты acquirerCode – код банка
Возвращаемое значение	нет
Описание	Начинает выполнение процедуры привязки карты. При попытке привязать карту до окончания выполняющегося платежа(отмены) будет сгенерировано исключение <code>PaymentException</code> . Асинхронный, см. PaymentControllerListener

addLinkedCard

Сигнатура	<code>void addLinkedCard(Context context, PaymentController.Currency currency)</code> throws <code>PaymentException</code>
Входные параметры	
Возвращаемое значение	
Описание	Устаревший. См. <code>addLinkedCard (Context context, PaymentController.Currency currency, null)</code>

removeLinkedCard

Сигнатура	<code>APIResult removeLinkedCard(Context context, int cardID)</code>
Входные параметры	context – контекст приложения

	cardID – ID удаляемой карты
Возвращаемое значение	Результат операции
Описание	Удаляет привязанную карту для текущего аккаунта. Синхронный

getLinkedCards

Сигнатура	APIReadLinkedCardsResult getLinkedCards(Context context)
Входные параметры	context – контекст приложения
Возвращаемое значение	Объект APIReadLinkedCardsResult, содержащий набор привязанных карт
Описание	Запрашивает набор привязанных карт для текущего аккаунта. Синхронный

startAutoConfig

Сигнатура	String startAutoConfig() throws IllegalStateException
Входные параметры	Нет
Возвращаемое значение	Строка конфигурации считывателя карт
Описание	Позволяет получить конфигурацию считывателя карт для дальнейшей работы с ним. При попытке вызвать во время проведения платежа или изменении конфигурации будет сгенерировано исключение IllegalStateException

balanceInquiry

Сигнатура	void balanceInquiry(Context context, PaymentController.Currency currency, String acquirerCode) throws PaymentException
Входные параметры	context – контекст приложения currency – валюта карты acquirerCode – код банка
Возвращаемое значение	нет
Описание	Начинает выполнение процедуры запроса баланса карты. При попытке проверить баланс до окончания выполняющегося платежа(отмены) будет сгенерировано исключение PaymentException. Асинхронный, см. PaymentControllerListener

balanceInquiry

Сигнатура	void balanceInquiry(Context context, PaymentController.Currency currency) throws PaymentException
Входные параметры	context – контекст приложения currency – валюта карты

Возвращаемое значение	Нет
Описание	Устаревший. См. <code>balanceInquiry(Context context, PaymentController.Currency currency, null)</code>

setRepeatOnError

Сигнатура	<code>void setRepeatOnError(boolean repeatOnError)</code>
Входные параметры	<code>repeatOnError</code> – признак необходимости повторного запроса карты при ошибке проведения транзакции
Возвращаемое значение	Нет
Описание	Позволяет включить/выключить режим повторного запроса карты считывателем карт при ошибке проведения транзакции

getRepeatOnError

Сигнатура	<code>boolean getRepeatOnError ()</code>
Входные параметры	Нет
Возвращаемое значение	Признак необходимости повторного запроса карты при ошибке проведения транзакции
Описание	Возвращает признак необходимости повторного запроса карты при ошибке проведения транзакции

submitEmailNotification

Сигнатура	<code>APIResult submitEmailNotification(Context context, String email, String subj, String body, Map<String, byte[]> images)</code>
Входные параметры	<code>context</code> – контекст приложения <code>email</code> – адрес для отправки письма <code>subj</code> – тема письма <code>body</code> – html содержимое письма <code>images</code> – набор изображений
Возвращаемое значение	Результат отправки запроса
Описание	Позволяет отправить уведомление на email пользователя. Изображения в письме должны размещаться по своему коду указанном в атрибуте <code>src</code> как параметр <code>cid:код</code> изображения (пример <code></code>). Пары код – изображения указываются в <code>images</code>

readerInjectKeys

Сигнатура	<code>void readerInjectKeys(String host) throws IllegalStateException</code>
Входные параметры	<code>host</code> – url хоста для прошивки ключей
Возвращаемое значение	Нет

Описание	Начинает процедуру прошивки ключей ридера. Доступно только для устройств Urovo. При попытке вызвать во время проведения платежа или изменении конфигурации будет сгенерировано исключение <code>IllegalStateException</code>
----------	--

readerConfigEmv

Сигнатура	<code>void readerConfigEmv(Hastable<String, Object> data) throws IllegalStateException</code>
Входные параметры	<code>data</code> – параметры конфигурации
Возвращаемое значение	Нет
Описание	Устаревший, используйте <code>readerSetConfig</code> . Начинает процедуру изменения EMV-конфигурации. Доступно только для устройств Urovo. При попытке вызвать во время проведения платежа или изменении конфигурации будет сгенерировано исключение <code>IllegalStateException</code>

readerConfigCapk

Сигнатура	<code>void readerConfigCapk(Hastable<String, Object> data) throws IllegalStateException</code>
Входные параметры	<code>data</code> – параметры конфигурации
Возвращаемое значение	Нет
Описание	Устаревший, используйте <code>readerSetConfig</code> . Начинает процедуру изменения CAPK-конфигурации. Доступно только для устройств Urovo. При попытке вызвать во время проведения платежа или изменении конфигурации будет сгенерировано исключение <code>IllegalStateException</code>

getKeysHosts

Сигнатура	<code>Map<String, String> getKeysHosts(Context context, String pin)</code>
Входные параметры	<code>context</code> – контекст приложения <code>pin</code> – ключ доступа
Возвращаемое значение	Перечень пар вида <Название хоста, url хоста>
Описание	Позволяет получить перечень хостов для прошивки ридера

tryGetFiscalInfo

Сигнатура	<code>APITryGetPaymentStatusResult tryGetFiscalInfo(Context context, String transactionID)</code>
Входные параметры	<code>context</code> – контекст приложения <code>transactionID</code> – ID транзакции

Возвращаемое значение	Результат запроса, содержащий в себе TransactionItem с обновленными данными
Описание	Позволяет выполнить запрос фискального состояния и фискальных данных транзакции с таймаутом запроса 60 секунд

tryGetPaymentStatus

Сигнатура	APITryGetPaymentStatusResult tryGetPaymentStatus(Context context, String transactionID)
Входные параметры	context – контекст приложения transactionID – ID транзакции
Возвращаемое значение	Результат запроса, содержащий в себе TransactionItem с обновленными данными
Описание	Позволяет выполнить запрос состояния оплаты транзакции с таймаутом запроса 60 секунд. Используется для платежей «по ссылке».

settlement

Сигнатура	SettlementResult settlement()
Входные параметры	Нет
Возвращаемое значение	Результат операции
Описание	Выполняет сверку по протоколу ТТК

prepare

Сигнатура	APIPrepareResult prepare(Context context, String productCode, Map<String, String> fields)
Входные параметры	context – контекст приложения productCode – код продукта, для которого требуется заполнить поля fields – значения preprable-полей продукта, по которым должны заполниться остальные поля
Возвращаемое значение	Результат операции
Описание	Выполняет заполнение полей для пользовательского продукта, имеющего свойство preprable, согласно данным, указанных в его preprable-полях. Рекомендуется заполнять значениями только preprable-полей.

readerSetConfig

Сигнатура	void readerSetConfig(String config) throws IllegalStateException
Входные параметры	config – строка конфигурации
Возвращаемое значение	Нет

Описание	Начинает процедуру конфигурации ридера. При попытке вызвать во время проведения платежа или изменении конфигурации будет сгенерировано исключение <code>IllegalStateException</code>
----------	--

setCustomReaderParams

Сигнатура	<code>void setCustomReaderParams(Hashtable<String, Object> data) throws IllegalStateException</code>
Входные параметры	data – параметры работы ридера
Возвращаемое значение	Нет
Описание	Устанавливает параметры работы ридера. При попытке вызвать во время проведения платежа или изменении конфигурации будет сгенерировано исключение <code>IllegalStateException</code>

readerBeep

Сигнатура	<code>void readerBeep(int count)</code>
Входные параметры	count – количество сигналов
Возвращаемое значение	Нет
Описание	Иницирует заданное количество звуковых сигналов от ридера, если он подключен. Поддерживается только для P17.

fiscalize

Сигнатура	<code>APITryGetPaymentStatusResult fiscalize(Context context, String transactionID)</code>
Входные параметры	context – контекст приложения transactionID – ID транзакции
Возвращаемое значение	Результат запроса, содержащий в себе TransactionItem с обновленными данными
Описание	Иницирует повторный запрос серверной фискализации с таймаутом запроса 60 секунд.

Интерфейс PaymentControllerListener

Callback-интерфейс для класса **PaymentController**.

Методы интерфейса:

onTransactionStarted

Сигнатура	void onTransactionStarted(String transactionID)
Входные параметры	transactionID – ID выполняемой транзакции для простого платежа. ID отменяемой транзакции для возврата/отмены платежа
Возвращаемое значение	Нет
Описание	Метод будет вызван перед выполнением транзакции

onFinished

Сигнатура	void onFinished(PaymentResultContext result)
Входные параметры	result – данные о проведенной транзакции
Возвращаемое значение	Нет
Описание	Метод будет вызван при успешном проведении платежа, успешной привязке карты, или успешной отмене платежа

onError

Сигнатура	void onError(PaymentError error, String errorMessage)
Входные параметры	error – тип ошибки errorMessage – сообщение об ошибке. Используется только когда error == SERVER_ERROR
Возвращаемое значение	Нет
Описание	Метод будет вызван при возникновении ошибки во время попытки проведения транзакции

onSelectApplication

Сигнатура	int onSelectApplication(List<String> apps)
Входные параметры	apps – список названий приложений
Возвращаемое значение	Порядковый номер выбранного приложения (начиная с 0)
Описание	Метод будет вызван при выполнении чиповой транзакции, если чиповая карта содержит более 1 приложения. Вызов метода происходит не в родительском потоке.

onConfirmSchedule

Сигнатура	boolean onConfirmSchedule(List<Map.Entry<Date, Double>> steps, double totalAmount)
Входные параметры	steps – список шагов выполнения расписания, состоящий из пар типа <Дата списания, Сумма к списанию> totalAmount – итоговая сумма по всем дням
Возвращаемое значение	Признак того, что плательщик подтверждает правильность расписания
Описание	Метод будет вызван при создании регулярного платежа. Вызов метода происходит не в родительском потоке.

onScheduleCreationFailed

Сигнатура	boolean onScheduleCreationFailed(PaymentError error, String errorMessage)
Входные параметры	error – тип ошибки errorMessage – сообщение об ошибке. Используется только когда error == SERVER_ERROR
Возвращаемое значение	true , если необходимо повторить попытку создания расписания
Описание	Метод будет вызван в случае возникновения ошибки при создании расписания регулярного платежа

onCancellationTimeout

Сигнатура	boolean onCancellationTimeout()
Входные параметры	Нет
Возвращаемое значение	true , для выполнения возврата платежа
Описание	Метод будет вызван в случае попытки выполнения отмены платежа по истечению доступного для отмены таймута

onPinRequest

Сигнатура	void onPinRequest()
Входные параметры	Нет
Возвращаемое значение	Нет
Описание	Метод будет вызван при запросе PIN-кода карты считывателем карт

onPinEntered

Сигнатура	void onPinEntered()
Входные параметры	Нет
Возвращаемое значение	Нет
Описание	Метод будет вызван после ввода PIN-кода карты

onBatteryState

Сигнатура	void onBatteryState(double percent)
Входные параметры	percent – уровень заряда считывателя карт, в процентах
Возвращаемое значение	Нет
Описание	Метод будет вызван после инициализации ридера

onSelectInputType

Сигнатура	PaymentController.PaymentInputType onSelectInputType(List<PaymentController.PaymentInputType> allowedInputTypes)
Входные параметры	allowedInputTypes – перечень доступных типов ввода для проведения отмены/возврата
Возвращаемое значение	Выбранный тип ввода
Описание	Метод будет вызван при проведении отмены/возврата, если необходимо выбрать способ проведения транзакции

onAutoconfigUpdate

Сигнатура	void onAutoConfigUpdate(double percent)
Входные параметры	percent – индикатор прогресса, в процентах
Возвращаемое значение	Нет
Описание	Метод будет вызываться при обновлении прогресса во время выполнения автоконфигурации считывателя карт

onAutoconfigFinished

Сигнатура	void onAutoConfigFinished(boolean success, String config, boolean isDefault)
Входные параметры	success – true , если автоконфигурация выполнена успешно config – строка конфигурации isDefault – true , если были использованы настройки по-умолчанию
Возвращаемое значение	Нет
Описание	Метод будет вызываться при завершении автоконфигурации считывателя карт

onSwitchedToCNP

Сигнатура	void onSwitchedToCNP()
Входные параметры	Нет
Возвращаемое значение	Нет
Описание	Метод будет вызываться при выполнении отмены транзакции в режиме CNP

onInjectFinished

Сигнатура	void onInjectFinished(boolean success)
Входные параметры	success – true , если прошивка выполнена успешно
Возвращаемое значение	Нет
Описание	Метод будет вызван при завершении прошивки ключей считывателя карт

onEmvConfigFinished

Сигнатура	void onEmvConfigFinished(boolean success)
Входные параметры	success – true , если EMV конфигурация выполнена успешно
Возвращаемое значение	Нет
Описание	Метод будет вызван при завершении emv конфигурации считывателя карт

onCapkConfigFinished

Сигнатура	void onCapkConfigFinished(boolean success)
Входные параметры	success – true , если Capk конфигурация выполнена успешно
Возвращаемое значение	Нет
Описание	Устаревший, используйте onReaderConfigFinished. Метод будет вызван при завершении capk конфигурации считывателя карт

onBarcodeScanned

Сигнатура	void onBarcodeScanned(String barcode)
Входные параметры	barcode – считанный штрих-код
Возвращаемое значение	Нет
Описание	Устаревший, используйте onReaderConfigFinished. Метод будет вызван, если встроенный сканнер считал штрих-код

onReaderConfigFinished

Сигнатура	void onReaderConfigFinished(boolean success)
Входные параметры	success – true , если конфигурация выполнена успешно
Возвращаемое значение	Нет
Описание	Метод будет вызван при завершении конфигурации считывателя карт

Класс PaymentContext

JavaBean контейнер данных, необходимых для выполнения разового платежа. Обратите внимание, что разница Summ(AuxData) – Amount суммы перечня товаров в поле AuxData и суммы платежа Amount будет учтена как предоплата. При Amount > Summ(AuxData) будет сгенерировано исключение **PaymentException**. При проведении транзакции по протоколу ТТК поле AuxData игнорируется.

Свойства класса:

Название	Описание
amount	Сумма платежа
currency	Валюта платежа
description	Описание платежа
transactionID	Не используется
image	Изображение, прикрепленное к платежу
currency	Валюта платежа
cash	Признак оплаты наличными(устаревший)
credit	Признак оплаты кредитом(устаревший)
nfc	Признак проведения транзакции NFC
method	Способ оплаты
acquirerCode	Код банка
receiptEmail	Email для отправки чека
receiptPhone	Телефон для отправки чека
linkedCardID	ID привязанной карты
extID	ID клиентского приложения
amountCashGot	Принято наличными
auxData	Перечень товаров в установленном формате
ern	Уникальный в пределах смены номер документа (используется при операции по протоколу ТТК)
suppressSignatureWaiting	Признак подавления ожидания подписи при формировании и отправке чека покупателю. Устанавливается true в случае, если до начала платежа известно, что подпись не будет отправлена.

Методы класса:

reset

Сигнатура	reset()
Входные параметры	Нет
Возвращаемое значение	Нет
Описание	Выполняет очистку полей объекта

putPurchase

Сигнатура	boolean putPurchase(Purchase purchase)
Входные параметры	purchase – данные о товаре
Возвращаемое значение	true , если товар был успешно добавлен
Описание	Используйте этот метод для добавления данных о товаре в стандартном формате Purchase

Класс RegularPaymentContext

Расширение класса **PaymentContext**, дополнительно содержащее свойства, необходимые для создания регулярного платежа. Для выполнения платежа в последний день месяца свойство **dayOfWeek** должно иметь значение, равное константе **LAST_DAY_OF_MONTH**. При создании регулярного платежа с продуктом, для которого применимы только управляемые регулярные платежи (`RecurrentMode == MANAGED`), необходимо установить поле **Managed true**. Остальные поля должны быть заполнены параметрами, необходимыми для старта платежа. В явном виде:

```
context.setManaged(true);
context.setRepeatType(PaymentController.RegularRepeatType.Never);
context.setEndType(PaymentController.RegularEndType.BY_QUANTITY);
context.setDay(0);
context.setRepeatCount(1);
context.setStartDate(new Date());
context.setEndDate(new Date());
context.setArbitraryDays(null);
context.setDayOfWeek(0);
context.setHour(0);
context.setMinute(0);
context.setMonth(0);
```

Свойства класса:

Название	Описание
repeatType	Тип регулярного платежа
endType	Способ завершения выполнения регулярного платежа
startDate	Дата начала выполнения регулярного платежа
endDate	Дата окончания выполнения регулярного платежа (если окончание по дате)
repeatCount	Количество выполнений регулярного платежа (если окончание по количеству повторов)
arbitraryDays	Дни, заданные для выполнения платежа (если тип платежа – по заданным датам)
month	Месяц для выполнения платежа ([1,12] и [1,4] при repeatType == Quarterly)
day	День для выполнения платежа ([1,31])
dayOfWeek	День недели для выполнения платежа ([0,7], 0 – Воскресенье)
hour	Час выполнения платежа
minute	Минута выполнения платежа
managed	Признак того, что настройки регулярного платежа будут определены хостом

Набор необходимых заполненных свойств зависит от типа платежа:

Тип платежа	Набор свойств
Never	startDate
Weekly	startDate, (endDate или repeatCount)
Monthly	startDate, (endDate или repeatCount), day
Quarterly	startDate, (endDate или repeatCount), month, day
Annual	startDate, (endDate или repeatCount), month, day
ArbitraryDays	arbitraryDays

Параметры repeatType, endType, receiptEmail, receiptPhone являются обязательными для всех типов регулярных платежей.

Параметры hour, minute являются необязательными для всех типов регулярных платежей.

Класс ReversePaymentContext

JavaBean контейнер данных, необходимых для выполнения отмены/возврата платежа. Обратите внимание, что проверка корректности содержимого AuxData остается за конечным пользователем. Разница Summ(AuxData) - Amount суммы перечня товаров в поле AuxData и суммы отмены(возврата) Amount будет учтена как предоплата. Для использования кредит-ваучера поле transactionID должно иметь значение **null**. Кредит-ваучер может быть проведен только при помощи платежной карты.

Свойства класса:

Название	Описание
transactionID	ID транзакции отменяемого платежа
action	Тип отмены
currency	валюта, используемая для отмены/возврата
returnAmount	сумма, на которую будет выполнена отмена. Для полной отмены установить null
auxData	Перечень товаров в установленном формате
extID	ID клиентского приложения
receiptEmail	Email для отправки чека
receiptPhone	Телефон для отправки чека
ern	Уникальный в пределах смены номер документа (для операций по протоколу ТТК)
suppressSignatureWaiting	Признак подавления ожидания подписи при формировании и отправке чека покупателю. Устанавливается true в случае, если до начала платежа известно, что подпись не будет отправлена.
acquirerCode	Код банка (только для кредит-ваучера)
nfc	Признак кредит-ваучера NFC

Методы класса:

reset

Сигнатура	reset()
Входные параметры	Нет
Возвращаемое значение	Нет
Описание	Выполняет очистку полей объекта

putPurchase

Сигнатура	boolean putPurchase(Purchase purchase)
Входные параметры	purchase – данные о товаре
Возвращаемое значение	true , если товар был успешно добавлен
Описание	Используйте этот метод для добавления данных о товаре в стандартном формате Purchase

Класс PaymentResultContext

JavaBean контейнер данных, полученных при успешном проведении платежа или отмене платежа.

Свойства класса:

Название	Описание
transactionItem	Данные о транзакции платежа/отмены платежа в представлении TransactionItem
scheduleItem	Данные о транзакции регулярного платежа в представлении ScheduleItem
requiresSignature	признак необходимости отправки подписи плательщика после оплаты
terminalName	Терминал
emvData	Набор данных EMV(чиповой) транзакции в представлении HashMap<String, String>
attachedCard	Данные о карте, привязанной при вызове <code>PaymentController.addLinkedCard()</code>

Класс AbstractEntity

Абстрактный класс-обертка для массива данных в представлении JSON. Реализует интерфейс **Serializable**.

Методы класса:

getJSON

Сигнатура	JSONObject getJSON()
Входные параметры	Нет
Возвращаемое значение	JSON представление набора данных
Описание	Возвращает JSON представление набора данных

Пакет `ibox.pro.sdk.external.entities`

Класс `TransactionItem`

Дочерний класс **AbstractEntity**. Является объектным представлением транзакции. Содержит набор свойств, определяющих ее. Имеет вложенные классы.

Свойства класса:

Название	Описание
ID	ID транзакции
Date	Время и дата выполнения транзакции, согласно GMT устройства
Description	Описание транзакции
Invoice	Номер чека
ApprovalCode	Код подтверждения
ScheduleID	ID регулярного платежа
ScheduleStepID	ID списания для рекуррентного платежа
Amount	Сумма транзакции
AmountEff	Баланс транзакции
InputType	Способ оплаты
Operation	Название операции
Latitude	Географическая широта места выполнения транзакции
Longitude	Географическая долгота места выполнения транзакции
HasPhoto	Признак наличия приложенного изображения
PhotoUrl	URL приложенного изображения
HasSignature	Признак наличия приложенной подписи
SignatureUrl	URL приложенной подписи
StateDisplay	Описание состояния транзакции
Card	Данные карты, которая была использована для оплаты, в представлении TransactionItem.Card
CanCancel	Признак возможности проведения отмены платежа
CanReturn	Признак возможности проведения возврата платежа
CanCancelPartial	Признак возможности проведения частичной отмены платежа
CanReturnPartial	Признак возможности проведения частичного возврата платежа
DisplayMode	Тип отображения транзакции в представлении DisplayMode
SubstateDisplay	Описание подсостояния транзакции
CardholderName	Владелец платежной карты
TerminalName	Терминал

Название	Описание
ExternalPayment	Данные оплаты по ссылке в представлении TransactionItem .ExternalPayment
CancelReturnTypes	Список типов ввода, доступных для проведения отмены/возврата
ReceiptEmail	Email для отправки чека
ReceiptPhone	Телефон для отправки чека
Balance	Баланс карты
CanCancelCNP	Транзакция может быть отменена в режиме CNP
CanCancelCNPpartial	Транзакция может быть частично отменена в режиме CNP
RRN	Retrieval Reference Number
SignatureRequired	Признак необходимости отправки подписи
FiscalInfo	Фискальные данные платежа в представлении TransactionItem .FiscalInfo
AmountCashGot	Принято наличными
AuxData	Перечень товаров в установленном формате
TaxMode	Режим начисления налогов, примененный при фискализации
TaxContributions	Налоговые начисления в представлении TaxContribution , без учета товаров в AuxData
Taxes	Перечень налогов в представлении Tax , примененных в транзакции, без учета товаров в AuxData
TaxSystemName	Система налогообложения, примененная при фискализации
Products	Привязанные к платежу пользовательские продукты

Наборы параметров:

DisplayMode

Тип отображения транзакции

Тип	Описание
DECLINED	Отклоненная транзакция
SUCCESS	Успешная транзакция
REVERSE	Транзакция отмены/возврата
REVERSED	Платеж отменен/выполнен возврат
NONFINANCIAL	

TaxMode

Режим начисления налогов

Тип	Описание
FOR_EACH	Начисление налогов по позициям
FOR_TOTAL	Начисление налогов на весь чек

Методы класса:`isNotCanceled`

Сигнатура	Boolean isNotCanceled()
Входные параметры	Нет
Возвращаемое значение	Признак того, что для платежа выполнена отмена или возврат
Описание	Возвращает признак отмены/возврата транзакции

Класс TransactionItem.Card

Вложенный класс **TransactionItem**, дочерний класс **AbstractEntity**. Содержит данные о платежной карте.

Свойства класса:

Название	Описание
Iin	Тип карты или "cash"(в случае оплаты наличными)
Bin	Внутренний идентификатор банка
Exp	Срок действия карты
PanMasked	Первые и последние 4 цифры номера карты, разделенные символом "*"
PanEnding	Последние 4 цифры номера карты
BankName	Название банка
BankCountryID	

Методы класса:

isCash

Сигнатура	isCash()
Входные параметры	Нет
Возвращаемое значение	true , если оплата была проведена наличными
Описание	Возвращает признак того, что оплата была проведена наличными

isPrepaid

Сигнатура	isPrepaid()
Входные параметры	Нет
Возвращаемое значение	true , если операция выполнена по предоплате
Описание	Возвращает признак того, что оплата была проведена по предоплате

isCredit

Сигнатура	isCredit()
Входные параметры	Нет
Возвращаемое значение	true , если операция выполнена в кредит
Описание	Возвращает признак того, что оплата была проведена в кредит

isOuter

Сигнатура	isOuter()
Входные параметры	Нет
Возвращаемое значение	true , если операция выполнена через внешний POS-терминал
Описание	Возвращает признак того, что оплата была проведена через внешний POS-терминал

Класс TransactionItem.ExternalPayment

Вложенный класс **ExternalPayment**, дочерний класс **AbstractEntity**. Содержит данные о платеже по ссылке.

Свойства класса:

Название	Описание
Type	Способ отображения данных
Link	Ссылка для оплаты
QR	Набор пар вида <Название, Содержимое> для отображения в виде QR-кодов

Наборы параметров:

Type

Способ отображения платежа

Тип	Описание
LINK	Отображать ссылку
QR	Отображать QR-код

Класс TransactionItem.FiscalInfo

Дочерний класс **AbstractEntity**. Содержит фискальные данные платежа.

Свойства класса:

Название	Описание
CVC	КПК
FiscalDocumentNumber	ФД
FiscalStorageNumber	ФН
FiscalMark	ФП
FiscalDeviceID	ЗН ККТ
FiscalDeviceRegNumber	РН ККТ
FiscalDocSN	Номер чека
FiscalPrinterShift	Номер смены
FiscalDatetime	Дата и время фискализации
FiscalStatus	Статус фискализации

Наборы параметров:

FiscalStatus

Статус фискализации

Тип	Описание
NONE	Не установлен
CREATED	Фискализируется
SUCCESS	Фискализована
FAILURE	Ошибка фискализации

Класс TransactionItem.Product

Дочерний класс **AbstractEntity**. Содержит данные привязанного к транзакции пользовательского продукта.

Свойства класса:

Название	Описание
Description	Описание продукта в представлении PaymentProductItem
Fields	Значения полей продукта

Класс TransactionItem.ProductField

Дочерний класс **AbstractEntity**. Содержит данные полей привязанного к транзакции пользовательского продукта.

Свойства класса:

Название	Описание
Description	Описание поля в представлении PaymentProductItemField
TextValue	Значение текстового поля
ImageUrl	Ссылка на изображение для поля с изображением

Класс ScheduleItem

Дочерний класс **AbstractEntity**. Является объектным представлением данных о регулярном платеже

Свойства класса:

Название	Описание
ID	ID регулярного платежа
Card	Данные карты, которая была использована для оплаты, в представлении TransactionItem.Card

Класс Account

Дочерний класс **AbstractEntity**. Содержит информацию об учетной записи

Свойства класса:

Название	Описание
SingleStepAuth	Признак доступности одношаговой авторизации
Name	Имя агента
BranchName	Название филиала
BranchAddress	Адрес филиала
BranchPhone	Телефон филиала
ClientName	Название компании
ClientLegalName	Юридическое название компании
ClientLegalAddress	Юридический адрес компании
ClientRealAddress	Фактический адрес компании
ClientPhone	Телефон компании
ClientWeb	Сайт компании
BankName	Название банка
TerminalName	Номер терминала
AcquirersByMethods	Сгруппированные по доступным методам ввода пары типа <Код банка, Название банка>
ID	ID аккаунта
PaymentOptions	Набор разрешенных типов оплаты
LinkedCards	Набор привязанных карт
Email	Email аккаунта

Класс Account.PaymentOption

Вложенный класс **Account**, дочерний класс **AbstractEntity**. Содержит информацию о типе оплаты, поддерживаемой банком.

Свойства класса:

Название	Описание
InputType	Тип оплаты
Code	Код банка
AcquirerName	Название банка

Класс LinkedCard

Дочерний класс **AbstractEntity**. Содержит данные о привязанной платежной карте.

Свойства класса:

Название	Описание
ID	ID привязанной карты
State	Состояние
isDeleted	Признак того, что карта более не привязана
Alias	Имя карты для отображения
PanEnding	Последние 4 цифры номера карты
PanMasked	Первые и последние 4 цифры номера карты, разделенные символом “*”
Balance	Баланс карты, используется только для операции привязки карты
Bin	Внутренний идентификатор банка

Наборы параметров:

State

Состояние карты

Тип	Описание
DISABLED	Карта недоступна для работы
ENABLED	Операции по карте разрешены

Класс Purchase

Дочерний класс **AbstractEntity**. Данные товара в стандартном формате. Внимание! При наличии в продукте тегов ФФД соответствующие им значения свойств, указанные через поля "Title", "Price", "Quantity", "TaxCode" будут проигнорированы. Т.е. запрос, к примеру, свойства "Title" будет возвращать значение тега 1030, если такой будет указан. Иначе будет возвращено значение свойства "Title". Описание формата ФФД 1.05 см. Приложение № 2 к приказу ФНС России от 21 марта 2017 г. № ММВ-7-20/229

Константы:

Название	Описание
TaxCode.VAT_NA	Код налоговой ставки «Без НДС»
TaxCode.VAT_0	Код налоговой ставки «НДС 0%»
TaxCode.VAT_10	Код налоговой ставки «НДС 10%»
TaxCode.VAT_18	Код налоговой ставки «НДС 18%»
TaxCode.VAT_20	Код налоговой ставки «НДС 20%»

Свойства класса:

Название	Описание
Title	Название товара(тег 1030)
Price	Цена за единицу товара(тег 1079)
Quantity	Количество(тег 1023)
TaxCode	Перечень кодов применяемых налоговых ставок(тег 1199)
Tags	Представление товара в форме перечня <тег, значение> согласно ФФД 1.05

Методы класса:

Build

Сигнатура	static Purchase Build(String title, double price, double quantity, List<String> taxCodes)
Входные параметры	Значения свойств экземпляра
Возвращаемое значение	Объект Purchase
Описание	Позволяет создать объект Purchase, используя стандартный формат товара.

Build

Сигнатура	static Purchase Build(Map<Integer, Object> tags) throws IllegalArgumentException
Входные параметры	tags – Представление товара в форме перечня <тег, значение> согласно ФФД 1.05
Возвращаемое значение	Объект Purchase

Описание	<p>Позволяет создать объект Purchase, используя описание товара посредством ФФД 1.05. Поддерживаемые классы в качестве ключей:</p> <ul style="list-style-type: none"> • Integer (теги типа <i>Целое</i> и <i>Флаги</i>) • Double (только тег 1079 и тег 1023) • String – все остальные <p>Значения типа <i>Структура</i>, <i>Массив</i> следует передавать как hex-строку. При указании в параметрах иного класса будет сгенерировано исключение <code>IllegalArgumentException</code>.</p>
----------	--

Класс Tax

Дочерний класс **AbstractEntity**. Содержит данные о налоговой ставке

Свойства класса:

Название	Описание
Code	Код ставки
Name	Название
Rate	Значение ставки

Класс TaxContribution

Дочерний класс **AbstractEntity**. Содержит данные о налоговом начислении по ставке

Свойства класса:

Название	Описание
Code	Код ставки
Name	Название
Rate	Значение ставки
Total	Сумма начислений

Класс PaymentProductItem

Дочерний класс **AbstractEntity**. Содержит данные о пользовательском продукте. Внимание!

При создании регулярных платежей для продуктов с RecurrentMode == MANAGED дополнительно ознакомьтесь с описанием класса **RegularPaymentContext**.

Свойства класса:

Название	Описание
Code	Уникальный код продукта
State	Состояние продукта
Apply	Доступные типы платежа
Title	Название продукта
TitleReceipt	Название продукта для печати в чеке
Fields	Поля продукта
Preparable	Признак того, что заполнение полей продукта должно выполняться результатом вызова метода PaymentController.prepare().
PreparableEditable	Признак того, что разрешено редактирование загруженных значений полей
PreparableOptional	Признак того, что значения полей могут быть заполнены без вызова метода PaymentController.prepare().
RecurrentMode	Настройка регулярного платежа

Наборы параметров:

State

Состояние продукта

Тип	Описание
DISABLED	Продукт отключе
ENABLED	Продукт доступен

PaymentType

Доступные типы платежа для продукта

Тип	Описание
PAYMENT	Только обычный платеж
RECURRENT	Только регулярный платеж
BOTH	Оба типа платежа
NONE	Платежи недоступны

RecurrentMode

Настройка регулярного платежа для продукта

Тип	Описание
UNDEFINED	Не определено
SIMPLE	График и настройки регулярного платежа передаются при создании платежа
MANAGED	График и настройки регулярного платежа устанавливаются сервером и не передаются при платеже

Класс PaymentProductItemField

Дочерний класс **AbstractEntity**. Содержит данные о поле пользовательского продукта. При формировании платежа с продуктом рекомендуется передавать в том числе и недоступные для пользователя (!UserVisible) поля со значениями по умолчанию.

Свойства класса:

Название	Описание
Type	Тип поля
Code	Уникальный код поля
Required	Обязательное поле
Preparable	Значение поля используется при вызове метода PaymentController.prepare().
Title	Название поля
TitleReceipt	Название поля для печати в чеке
DefaultValue	Значение по умолчанию
TextMask	Маска ввода(регулярное выражение)
TextRegExp	Регулярное выражение для проверки правильности ввода
Multiline	Признак многострочности поля
Numeric	Признак того, что значение поля является десятичным числом
ReceiptEmail	Признак того, что значение поля должно использоваться как email для отправки чека
ReceiptPhone	Признак того, что значение поля должно использоваться как телефон для отправки чека
UserVisible	Признак того, что поле доступно для пользователя
PrintInReceipt	Признак того, что поле должно быть напечатано в чеке

Наборы параметров:

Type

Тип поля

Тип	Описание
TEXT	Текстовое поле
IMAGE	Поле с изображением

Класс PreparedField

Дочерний класс **AbstractEntity**. Обертка для пар код/значение, получаемых в результате `PaymentController.prepare()`

Свойства класса:

Название	Описание
Code	Код поля
Value	Значение поля

Методы класса:

`isPaymentAmount`

Сигнатура	<code>boolean isPaymentAmount()</code>
Входные параметры	Нет
Возвращаемое значение	Признак того, что значение поля должно быть использовано как сумма платежа
Описание	Возвращает признак того, что значение поля должно быть использовано как сумма платежа

Класс APIResult

Дочерний класс **AbstractEntity**. Является примитивной сущностью, содержащую ответ от сервера

Методы класса:

getErrorCode

Сигнатура	int getErrorCode()
Входные параметры	Нет
Возвращаемое значение	Код ошибки
Описание	Возвращает код ошибки. 0 – если ответ не содержит сообщений об ошибке, -1 – если ответ от сервера не получен, или формат ответа неправильный

getErrorMessage

Сигнатура	String getErrorMessage()
Входные параметры	Нет
Возвращаемое значение	Сообщение об ошибке
Описание	Возвращает сообщение об ошибке

isValid

Сигнатура	boolean isValid()
Входные параметры	Нет
Возвращаемое значение	Признак того, что ответ не содержит сообщений об ошибке и его формат правильный
Описание	Возвращает признак того, что ответ не содержит сообщений об ошибке и его формат правильный

Класс APIGetHistoryResult

Дочерний класс **APIResult**. Содержит набор транзакций, полученных в ответ на запрос истории.

Методы класса:

getTransactions

Сигнатура	ArrayList<TransactionItem> getTransactions()
Входные параметры	Нет
Возвращаемое значение	ArrayList транзакций
Описание	Возвращает набор транзакций, содержащихся в ответе.

getInProgressTransactions

Сигнатура	ArrayList<TransactionItem> getInProgressTransactions()
Входные параметры	Нет
Возвращаемое значение	ArrayList транзакций
Описание	Возвращает набор транзакций, со статусом «В обработке»

Класс APIAuthResult

Дочерний класс **APIResult**.

Методы класса:

getAccount

Сигнатура	Account getAccount()
Входные параметры	Нет
Возвращаемое значение	Объект Account
Описание	Возвращает информацию об учетной записи.

getTaxID

Сигнатура	String getTaxID()
Входные параметры	Нет
Возвращаемое значение	ИНН пользователя
Описание	Возвращает ИНН пользователя

getProducts

Сигнатура	String getProducts()
Входные параметры	Нет
Возвращаемое значение	Набор пользовательских продуктов
Описание	Возвращает набор пользовательских продуктов

Класс **APIReadLinkedCardsResult**

Дочерний класс **APIResult**.

Методы класса:

`getLinkedCards`

Сигнатура	<code>ArrayList<LinkedCard> getLinkedCards()</code>
Входные параметры	Нет
Возвращаемое значение	<code>ArrayList</code> привязанных карт
Описание	Возвращает набор привязанных карт, содержащихся в ответе

Класс `APITryGetPaymentStatusResult`

Дочерний класс `APIResult`.

Методы класса:

`getTransaction`

Сигнатура	<code>TransactionItem getTransaction()</code>
Входные параметры	Нет
Возвращаемое значение	<code>TransactionItem</code> с обновленными данными
Описание	Возвращает транакцию с обновленными данными

Класс SettlementResult

Результат операции «Сверка»

Свойства класса:

Название	Описание
Success	Признак успеха операции
ErrorMessage	Сообщение об ошибке

Класс APIPrepareResult

Дочерний класс **APIResult**.

Методы класса:

getFields

Сигнатура	List<PreparedField> getFields()
Входные параметры	Нет
Возвращаемое значение	Перечень полученных значений для полей пользовательского продукта
Описание	Возвращает перечень полученных значений для полей пользовательского продукта

Пакет `ibox.pro.sdk.external.ui`

Класс `SignatureView`

Является View, предоставляющим возможность выполнения подписи клиента с помощью передвижений пальца или стилуса по экрану.

Свойства класса:

Название	Описание
<code>color</code>	Цвет кисти

Методы класса:

`erase`

Сигнатура	<code>erase()</code>
Входные параметры	Нет
Возвращаемое значение	Нет
Описание	Очищает поле для подписи

`getBitmap`

Сигнатура	<code>Bitmap getBitmap()</code>
Входные параметры	Нет
Возвращаемое значение	Bitmap представление подписи
Описание	Возвращает Bitmap представление подписи

`getBitmapByteArray`

Сигнатура	<code>byte [] getBitampByteArray()</code>
Входные параметры	Нет
Возвращаемое значение	<code>byte []</code> представление подписи
Описание	Возвращает <code>byte []</code> представление подписи

Приложение 1: Печать слипа

Данные для слипа приходят в событии onFinished

Реквизиты клиента можно получить при помощи метода `PaymentController.auth()`.

Поля слипа:

Название	Описание
Банк	<code>Account.getBankName()</code>
Название компании	<code>Account.getClientName()</code>
Название юридического лица	<code>Account.getClientLegalName()</code>
Телефон компании	<code>Account.getClientPhone()</code>
WEB-сайт компании	<code>Account.getClientWeb()</code>
Дата и время операции	<code>paymentResultContext.getTransactionItem().getDate()</code>
Номер терминала	<code>paymentResultContext.getTransactionItem().getTerminalName()</code>
Номер чека	<code>paymentResultContext.getTransactionItem().getInvoice()</code>
Код подтверждения	<code>result.TransactionItem.AcquirerApprovalCode</code>
Номер и тип карты	<code>paymentResultContext.getTransactionItem().getCard().getlin()</code> , <code>paymentResultContext.getTransactionItem().getCard().getPanMasked()</code>
EMV тэги транзакции	<code>paymentResultContext.getEmvData()</code> , печатаются в виде ключ-значение
Тип операции	<code>paymentResultContext.getTransactionItem().getOperation()</code>
Сумма операции	<code>paymentResultContext.getTransactionItem().getAmount()</code>
Комиссия	0.00 р.
Статус	Успешно
Подпись клиента	Место для подписи в случае, если <code>paymentResultContext.isRequiresSignature() == true</code> , в ином случае «Подтверждено вводом PIN»

Пример слипа:

ВТБ 24
Тестовый клиент
ООО "Тестовый клиент"
+7 916 111 2233
www.testclient.com
Дата и время операции: 21.03.2017 15:47:34
Терминал: II040001
Чек: RM7ZEDMAAE7L
Код подтверждения: SIMULATION
Карта: mastercard **** 5631
AID: A0000000041010
TSI: 6800
TVR: 8020008000
Операция: Purchase
Итого: 33 р
Комиссия: 0.00 р
Статус: Успешно
Подтверждено вводом PIN кода.s

Приложение 2: Пример чека

Позиция 1 – Без НДС

Позиция 2 – НДС 10%

Позиция 3 – НДС 18%

```
-----ЧЕК-----
ПРИХОД
КАССИР          Тестовый агент с принтером
ДАТА И ВРЕМЯ      01.02.18 12:00:11
НОМЕР ЧЕКА        ABCDEFGHIJKL
КОД ПОДТВЕРЖДЕНИЯ 123456ABCD78
-----
Позиция 1          2,000x111,26=222,52
Позиция 2          1,000x14,00=14,00
Позиция 3          2,990x43,22=129,23
-----
ИТОГ                365,75
-----
НАЛИЧНЫМИ          365,75
ЭЛЕКТРОННЫМИ        0,00
ПРЕДВАРИТЕЛЬНАЯ ОПЛАТА (АВАНС)  0,00
ПОСЛЕДУЮЩАЯ ОПЛАТА (КРЕДИТ)    0,00
СДАЧА                0,00
Сумма без НДС        222,52
Сумма по НДС 0%      0,00
Сумма НДС 10%        1,27
Сумма НДС 18%        19,71
-----ФИСК. БЛОК-----
14.03.18 14:58
ИНН                  123567890
СНО                  ОЧН
САЙТ ФНС             http://nalog.ru
СМЕНА №              188
ЧЕК №                1
ЗН ККТ               000000000011111
РН ККТ               000000002222222
ФН                   999999999999999
ФД                   0000003333
ФПД                  4444444444
-----КОНЕЦ ФИСК. БЛОКА-----
-----КОНЕЦ ЧЕКА-----
```

Приложение 3: Некоторые коды ошибок для ридеров, работающих по протоколу ТТК

Код ошибки	Описание
B4	Неверный номер ERN
BB	Требуется синхронизация журнала
FE	Неверный формат сообщения, отсутствуют обязательные поля
JF	Требуется сверка итогов
NF	Оригинальная транзакция не найдена по номеру банковского чека
UN	Выполнение операции невозможно из-за ограничений функциональности
UP	Требуется обновление ПО