ibox

iboxPro

iOS SDK description

Версия 1.6.2 от 26 апреля 2018 г.

Содержание

1 PaymentController 5

1.1 Методы 5

1.1.1 instance 5

1.1.2 destroy 5

1.1.3 setPaymentContext: 5

1.1.4 enable 5

1.1.5 disable 5

1.1.6 retry 5

1.1.7 setDelegate: 5

1.1.8 isReaderConnected 5

1.1.9 setCardApplication: 6

1.1.10 saveBTDevice: 6

1.1.11 setBTDevice: 6

1.1.12 scheduleStepsConfirm 6

1.1.13 pingReaderWithDoneAction: 6

1.1.14 setSingleStepAuthentication: 6

1.1.15 authentication 6

1.1.16 historyWithPage: 6

1.1.17 historyWithTransactionID: 7

1.1.18 adjustWithTrId:Signature:ReceiptEmail:ReceiptPhone: 7

1.1.19 adjustWithScheduleId:Signature:ReceiptEmail:ReceiptPhone: 7

1.1.20 reverseAdjustWithTrId:Signature:ReceiptEmail:ReceiptPhone: 7

1.1.21 paymentStatusWithTrId: 8

1.1.22 paymentStatusWithTrId:DoneAction: 8

1.1.23 setEmail:Password: 8

1.1.24 setReaderType: 8

1.1.25 setRequestTimeOut: 8

1.2 Значения числовых полей и перечисления. 8

1.2.1 Перечисление PaymentControllerErrorType 8

1.2.2 Перечисление PaymentControllerReaderEventType 9

1.2.3 Перечисление PaymentControllerReaderType 9

2 Интерфейс PaymentControllerDelegate 10
2.1 Методы 10
2.1.1 PaymentControllerStartTransaction: 10
2.1.2 PaymentControllerDone: 10
2.1.3 PaymentControllerError:Message: 10
2.1.4 PaymentControllerReaderEvent: 10
2.1.5 PaymentControllerRequestCardApplication: 10
2.1.6 PaymentControllerRequestBTDevice: 10
2.1.7 PaymentControllerScheduleStepsStart 10
2.1.8 PaymentControllerScheduleStepsCreated: 10
3 PaymentContext 11
3.1 Перечисление CurrencyType 11
4 ReversePaymentContext 11
5 RecurrentPaymentContext 11
5.1 Перечисление ScheduleType 12
5.2 Перечисление ScheduleEndType 12
6 TransactionData 12
6.1 Перечисление TransactionDataType 13
7 StepItem 13
8 BTDevice 13
9 APIResult 13
10 APIAuthenticationResult 13
11 APIHistoryResult 13
12 Account 14
13 PaymentOption 14
14 Acquirer 14
15 TransactionItem 14
15.1 Коды состояний: 16
15.2 Коды подсостояний: 16
15.3 Перечисление TransactionInputType: 16
15.4 Перечисление TransactionItemReverseMode 16
15.5 Перечисление TransactionItemDisplayMode 17
16 Card 17
17 DescriptionProduct 17

17.1 Коды состояний:.....17
17.2 DescriptionProductInputType.....18
18 DescriptionProductField.....18
18.1 Коды состояний:.....18
18.2 DescriptionProductFieldType.....18
19 TransactionProduct.....18
20 SignatureView.....19
20.1 Методы.....19
20.1.1 update.....19
20.1.2 updateWithOrientation:(BOOL)update.....19
20.1.3 isEmpty.....19
20.1.4 getByteArray.....19
20.1.5 clear.....19
21 Описание алгоритма совершения платежа.....19

1 PaymentController

1.1 Методы

1.1.1 instance

Возвращает экземляр класса – объект синглтон.

1.1.2 destroy

Уничтожает объект синглтон.

1.1.3 setPaymentContext:

Устанавливает объект содержащий данные платежа.

Параметры:

`paymentContext`	`PaymentContext *`	Объект, который содержит данные необходимые для совершения транзакций.

1.1.4 enable

Включает ридер

1.1.5 disable

Выключает ридер

1.1.6 retry

Возвращает PaymentController в исходное положение, после чего он снова будет ожидать проведения картой или вставки карты. В случае оплаты наличными или предоплаты, будет проведена еще одна попытка совершить транзакцию.

1.1.7 setDelegate:

Устанавливает объект-обработчик событий возможных при совершении транзакций.

Параметры:

`delegate`	`id<PaymentControllerDelegate>`	Объект-обработчик событий возможных при совершении транзакций.

1.1.8 isReaderConnected

Проверяет подключен ли ридер.

Возвращаемое значение:

BOOL

1.1.9 setCardApplication:

Устанавливает с каким приложением карты будет производиться транзакция.

Параметры:

`application`	`int`	Номер приложения в массиве полученом приложением из события `PaymentControllerRequestCardApplication:`

1.1.10 saveBTDevice:

Запоминает Bluetooth ридер, с которым будет производиться транзакция.

Параметры:

`device`	BTDevice *	Объект, который содержит данные о ридере.

1.1.11 setBTDevice:

Устанавливает Bluetooth ридер, с которым будет производиться транзакция.

Параметры:

`device`	BTDevice *	Объект, который содержит данные о ридере.

1.1.12 scheduleStepsConfirm

Подтверждает шаги расписания и дает возможность запустить это расписание.

1.1.13 pingReaderWithDoneAction:

При помощи этого метода можна получить текущее состояние ридера.

`doneAction`	`void (^)(NSDictionary *)`	Блок кода, в котором, в качестве параметра, приходят данные ридера(подключен/нет и состояние заряда батареи).

1.1.14 setSingleStepAuthentication:

Задает признак однофакторной авторизации при проведении EMV транзакции.

`singleStepAuthentication`	BOOL	Признак однофакторной авторизации.

1.1.15 authentication

При помощи этого метода можна получить данные аккаунта пользователя.

Возвращаемое значение:

APIAuthenticationResult

1.1.16 historyWithPage:

При помощи этого метода можна получить историю транзакций пользователя.

page	int	Номер страницы (первая страница - 1).

Возвращаемое значение:

APIHistoryResult

1.1.17 historyWithTransactionID:

При помощи этого метода можна получить транзакций, которая есть в истории пользователя.

transactionID	NSString *	Уникальный идентификатор транзакции.

Возвращаемое значение:

APIHistoryResult

1.1.18 adjustWithTrId:Signature:ReceiptEmail:ReceiptPhone:

Вызывается для отправки дополнительных данных о транзакции.

trId	NSString *	Уникальный идентификатор транзакции полученный в событии PaymentControllerDone:
signature	NSData *	Картинка с подписью плательщика в формате PNG
receiptEmail	NSString *	Email для отправки чека
receiptPhone	NSString *	Телефон для отправки SMS чека

Возвращаемое значение:

APIResult

1.1.19 adjustWithScheduleId:Signature:ReceiptEmail:ReceiptPhone:

Вызывается для отправки дополнительных данных о расписании.

scheduleId	NSString *	Уникальный идентификатор расписания полученный в событии PaymentControllerDone:
signature	NSData *	Картинка с подписью плательщика в формате PNG
receiptEmail	NSString *	Email для отправки чека
receiptPhone	NSString *	Телефон для отправки SMS чека

Возвращаемое значение:

APIResult

1.1.20 reverseAdjustWithTrId:Signature:ReceiptEmail:ReceiptPhone:

Вызывается для отправки дополнительных данных о отмене/возврате платежа.

trId	NSString *	Уникальный идентификатор расписания полученный в событии PaymentControllerDone:
signature	NSData *	Картинка с подписью плательщика в формате PNG
receiptEmail	NSString *	Email для отправки чека
receiptPhone	NSString *	Телефон для отправки SMS чека

Возвращаемое значение:

APIResult

1.1.21 paymentStatusWithTrId:

Вызывается для полученя текущего состояния транзакции.

trId	NSString *	Уникальный идентификатор расписания полученный в событии PaymentControllerDone:

Возвращаемое значение:
APIResult

1.1.22 paymentStatusWithTrId:DoneAction:

При помощи этого метода можна получить текущее состояние транзакции.

trId	NSString *	Уникальный идентификатор расписания полученный в событии PaymentControllerDone:
doneAction	void (^)(APIResult *)	Блок кода, в котором, в качестве параметра, приходит ответ от сервера APIResult *.

1.1.23 setEmail:Password:

Устанавливает email и пароль для входа в систему.

email	NSString *	Email для входа в систему
password	NSString *	Пароль для входа в систему.

1.1.24 setReaderType:

Устанавливает тип используемого ридера.

readerType	PaymentControllerReaderType	Тип ридера.

1.1.25 setRequestTimeOut:

Устанавливает таймаут для запросов на сервер.

timeOut	double	Время для выполнения.

1.2 Значения числовых полей и перечисления.

1.2.1 Перечисление PaymentControllerErrorType

PaymentControllerErrorType_COMMON	Общая ошибка.
PaymentControllerErrorType_CARD_INSERTED_WRONG	Карта вставлена некорректно.
PaymentControllerErrorType_READER_DISCONNECTED	Ридер одключен.
PaymentControllerErrorType_READER_TIMEOUT	Таймаут ридера.
PaymentControllerErrorType_SUBMIT	Общая ошибка при проведении транзакции.
PaymentControllerErrorType_SUBMIT_CASH	Ошибка при проведении наличной операции
PaymentControllerErrorType_SUBMIT_PREPAID	Ошибка при проведении операции предоплаты.
PaymentControllerErrorType_SUBMIT_CREDIT	Ошибка при проведении операции постоплаты.
PaymentControllerErrorType_SUBMIT_LINK	Ошибка при проведении оплаты по ссылке.

PaymentControllerErrorType_SWIPE	Ошибка при проведении транзакции по магнитной полосе.
PaymentControllerErrorType_ONLINE_PROCESS	Ошибка обработки online транзакции.
PaymentControllerErrorType_REVERSE	Ошибка при попытке отменения транзакции.
PaymentControllerErrorType_REVERSE_CASH	Ошибка при попытке отменения наличной транзакции.
PaymentControllerErrorType_REVERSE_PREPAID	Ошибка при попытке отменения транзакции предоплаты.
PaymentControllerErrorType_REVERSE_CREDIT	Ошибка при попытке отменения транзакции постоплаты.
PaymentControllerErrorType_SCHEDULE_STEPS	Ошибка при создания шагов расписания.
PaymentControllerErrorType_EMV_ERROR	Ошибка подтверждения транзакции EMV чипом карты.
PaymentControllerErrorType_EMV_TERMINATED	Ошибка проведения EMV транзакции. Транзакция прервана.
PaymentControllerErrorType_EMV_DECLINED	Ошибка проведения EMV транзакции. Транзакция отклонена.
PaymentControllerErrorType_EMV_CANCEL	Ошибка проведения EMV транзакции. Транзакция отменена.
PaymentControllerErrorType_EMV_CARD_ERROR	Ошибка проведения EMV транзакции. Ошибка карты.
PaymentControllerErrorType_EMV_CARD_BLOCKED	Ошибка проведения EMV транзакции. Карта заблокирована.
PaymentControllerErrorType_EMV_DEVICE_ERROR	Ошибка проведения EMV транзакции. Ошибка ридера.
PaymentControllerErrorType_EMV_CARD_NOT_SUPPORTED	Ошибка проведения EMV транзакции. Карта не поддерживается.
PaymentControllerErrorType_EMV_ZERO_TRAN	Ошибка проведения EMV транзакции. Нулевая сумма транзакции.

1.2.2 Перечисление PaymentControllerReaderEventType

Описывает типы событий ридера.

PaymentControllerReaderEventType_INITIALIZATION	Ридер инициализирован
PaymentControllerReaderEventType_CONNECTED	Ридер подключен к разъему.
PaymentControllerReaderEventType_DISCONNECTED	Ридер отключен от разъема.
PaymentControllerReaderEventType_CARD_INSERTED	Карта вставлена корректно.
PaymentControllerReaderEventType_CARD_SWIPED	Карта проведена через считыватель магнитной полосы.
PaymentControllerReaderEventType_EMV_STARTED	Начало EMV транзакции.

1.2.3 Перечисление PaymentControllerReaderType

Описывает доступные типы ридеров.

PaymentControllerReaderType_C15	C15 ридер
PaymentControllerReaderType_P15	P15 ридер
PaymentControllerReaderType_P17	P17 ридер

2 Интерфейс PaymentControllerDelegate

2.1 Методы

2.1.1 PaymentControllerStartTransaction:

Вызывается при начале проведения транзакции.

параметры

`transactionID`	`NSString *`	Уникальный код транзакции.

2.1.2 PaymentControllerDone:

Вызывается при завершении транзакции. Получает объект TransactionData.

2.1.3 PaymentControllerError:Message:

Вызывается при получении ошибки.

параметры

`error`	`PaymentControllerErrorType`	Код ошибки.
`message`	`NSString *`	Сообщение об ошибке в виде текстовой строки.

2.1.4 PaymentControllerReaderEvent:

Вызывается при получении события ридера. Тип события передается в параметре.

2.1.5 PaymentControllerRequestCardApplication:

Вызывается при необходимости выбрать приложение карты. Возвращает массив содержащий названия приложений карты. Выбранный элемент необходимо передать методом setCardApplication:

2.1.6 PaymentControllerRequestBTDevice:

Вызывается при необходимости выбрать Bluetooth ридер. Возвращает массив объектов BTDevice. Выбранный элемент необходимо передать методом setBTDevice:

2.1.7 PaymentControllerScheduleStepsStart

Вызывается при начале выполнения серверного вызова на получения шагов расписания.

2.1.8 PaymentControllerScheduleStepsCreated:

Вызывается после получения шагов расписания от сервера. В качестве параметра – массив объектов StepItem.

3 PaymentContext

Объект, который содержит данные необходимые для совершения транзакций.

Amount	Double	Сума транзакции.
Description	NSString *	Описание транзакции.
ReceiptMail	NSString *	Электронная почта для отправки чека.
ReceiptPhone	NSString *	Телефон для отправки чека.
ProductCode	NSString *	Код продукта.
Acquirer	NSString *	Код эквайера.
ProductData	NSArray *	Данные продукта, каждый элемент массива – объект NSDictionary *, ключем является код поля, а значение либо NSString * либо UIImage *, в зависимости от типа поля.
Purchases	NSArray *	Данные внешнего платежа.
Image	NSData *	Изображение прикрепляемое к транзакции.
Currency	CurrencyType	Тип валюты.

3.1 Перечисление CurrencyType

Тип валюты транзакции.

CurrencyType_RUB	Валюта российский рубль.
CurrencyType_VND	Валюта вьетнамский донг.

4 ReversePaymentContext

Данный класс порожден от PaymentContext, имеет дополнительные свойства для совершения отмены/возврата платежа.

TransactionID	NSString *	Уникальный ключ транзакции.
AmountReverse	double	Сумма отмены транзакции — используется при частичных отменах/возвратах.

5 RecurrentPaymentContext

Данный класс порожден от PaymentContext, имеет дополнительные свойства для создания расписания.

Type	ScheduleType	Тип повторения расписания.
EndType	ScheduleEndType	Тип завершения выполнения расписания.
StartDate	NSString *	Дата начала выполнения расписания. Формат даты – yyyy-MM-dd.
EndDate	NSString *	Дата окончания выполнения расписания(если окончание по дате). Формат даты – yyyy-MM-dd.
Dates	NSArray *	Массив заданных дат расписания(в случае типа расписания – ScheduleType_ArbitraryDates). Формат даты – yyyy-MM-dd.

EndCount	Int	Количество выполнений расписания(если окончание по количеству повторов).
Month	Int	Месяц для выполнения платежа ([1,12] и [1,4] в случае типа расписания ScheduleType_Quarterly)
Day	int	День выполнения платежа(в случае типа ScheduleType_Weekly – [0-7], где 0 - восресенье, в случае ScheduleType_Monthly, ScheduleType_Quaterly и ScheduleType_Annual – [1-32], где 32 – последний день месяца)
Hour	int	Час выполнения платежа расписания.
Minute	Int	Минута выполнения платежа расписания.

5.1 Перечисление ScheduleType

Тип повторения расписания и набор необходимых параметров.

ScheduleType_DelayedOnce	Платеж будет выполнен один раз.	StartDate
ScheduleType_Weekly	Еженедельный платеж.	StartDate, (EndDate or EndCount), Day
ScheduleType_Monthly	Ежемесячный платеж.	StartDate, (EndDate or EndCount), Day
ScheduleType_Quarterly	Ежеквартальный платеж.	StartDate, (EndDate or EndCount), Month, Day
ScheduleType_Annual	Ежегодный платеж.	StartDate, (EndDate or EndCount), Month, Day
ScheduleType_ArbitraryDates	Платеж будет выполнятся в заданные дни.	Dates

5.2 Перечисление ScheduleEndType

Тип окончания выполнения расписания.

ScheduleEndType_Date	Окончание по дате. Формат даты – yyyy-MM-dd.
ScheduleEndType_Count	Окончание по количеству раз.

Параметры Type, EndType являются обязательными для всех типов регулярных платежей.Параметры Hour, Minute являются необязательными для всех типов регулярных платежей.

6 TransactionData

Предоставляет доступ к данным выполненной транзакции.

ID	NSString *	Уникальный идентификатор транзакции.
Amount	double	Сумма транзакции.
Invoice	NSString *	Номер чека транзакции.
Card	Card *	Объект, который содержит данные о карте.ы
RequiredSignature	BOOL	Признак необходимости запросить подпись держателя.
Transaction	TransactionItem *	Объект с детальными данными транзакции.

6.1 Перечисление TransactionDataType

Тип транзакции.

TransactionDataType_Payment	Обычная транзакция.
TransactionDataType_Schedule	Расписание.

В зависимости от типа транзакции некоторые поля объекта TransactionData могут быть не заполненными.

7 StepItem

Предоставляет доступ к данным шага расписания.

amount	double	Сума списания за один шаг.
date	NSString *	Дата выполнения списания.

8 BTDevice

Объект, который содержит информацию о блютузном ридере.

ID	NSString *	Уникальный идентификатор ридера.
name	NSString *	Название ридера.

9 APIResult

Объект содержащий базовый ответ сервера.

valid	BOOL	Признак успешного получения ответа от сервера.
errorCode	int	Код ошибки (0 – нет ошибки).
errorMessage	NSString *	Сообщение об ошибке.

10 APIAuthenticationResult

Объект содержащий ответ сервера на запрос получения данных аккаунта пользователя. Порожден от базового класса APIResult.

account	Account *	Данные аккаунта пользователя.

11 APIHistoryResult

Объект содержащий ответ сервера на запрос получения истории платежей. Порожден от базового класса APIResult.

transactions	NSArray *	Массив объектов TransactionItem.

12 Account

Предоставляет доступ к информации об учетной записи.

singleStepAuthMode	BOOL	Признак доступности одношаговой авторизации.
name	NSString *	Имя агента.
branchName	NSString *	Название филиала.
branchAdress	NSString *	Адрес филиала.
branchPhone	NSString *	Телефон филиала.
clientName	NSString *	Название компании.
clientLegalName	NSString *	Юридическое название компании.
clientLegalAddress	NSString *	Юридический адрес компании.
clientRealAddress	NSString *	Фактический адрес компании.
clientPhone	NSString *	Телефон компании.
clientWeb	NSString *	Сайт компании.
bankName	NSString *	Название банка.
terminalName	NSString *	Номер терминала.
paymentOptions	NSArray *	Допустимые способы проведения транзакции.
acquirerWithCode:	Acquirer *	Объект эквайер с соответсвующим кодом.

13 PaymentOption

Предоставляет доступ к данным объекта способа проведения транзакции.

inputType	TransactionInputType	Способ совершения транзакции.
acquirer	Acquirer *	Объект эквайер.

14 Acquirer

Предоставляет доступ к данным эквайера.

inputType	TransactionInputType	Способ совершения транзакции.
ID	NSString *	Уникальный идентификатор эквайера.
name	NSString *	Имя эквайера.
imageUrl	NSString *	Путь к картинке эквайера.

15 TransactionItem

Предоставляет доступ к данным транзакции.

ID	NSString *	Уникальный идентификатор транзакции.
date	NSString *	Дата и время проведения транзакции.
currencyID	NSString *	Уникальный идентификатор валюты.

amountFormat	NSString *	Формат суммы с знаком валюты.
amountFormatWithoutCurrency	NSString *	Формат суммы без знака валюты.
currencySign	NSString *	Знак валюты.
descriptionOfTransaction	NSString *	Описание транзакции.
stateDisplay	NSString *	Описание состояния транзакции.
stateline1	NSString *	Более детальное описание состояния транзакции (первая строка).
stateLine2	NSString *	Более детальное описание состояния транзакции (вторая строка).
invoice	NSString *	Номер чека транзакции.
signatureURL	NSString *	URL подписи приложеной к транзакции.
photoURL	NSString *	URL картинки приложеной к транзакции.
scheduleID	NSString *	Уникальный идентификатор очереди.
scheduleStepID	NSString *	ID описания для рекуррентного платежа.
approvalCode	NSString *	Код подтверждения.
operation	NSString *	Название операции.
cardholderName	NSString *	Владелец платежной транзакции.
terminalName	NSString *	Терминал.
amount	double	Сумма транзакции.
amountNetto	double	Сумма транзакции без комиссии.
feeTotal	double	Сумма комиссии.
latitude	double	Географическая широта места выполнения транзакции.
longitude	double	Географическая долгота места выполнения транзакци.
hasSignature	BOOL	Признак наличия приложенной подписи.
hasPhoto	BOOL	Признак наличия приложенной картинки.
hasGPSData	BOOL	Признак наличия приложенных данных геолокации.
canCancel	BOOL	Признак возможности проведения отмены платежа.
canReturn	BOOL	Признак возможности проведения возврата платежа.
withOrder	BOOL	Признак наличия заказа из каталога.
withCustomFields	BOOL	Признак наличия пользовательского продукта.
cashPayment	BOOL	Признак проведения платежа наличными.
productsCount	int	Количество продуктов из каталога в транзакции.
currencyDecimalsCount	int	Количество знаков после десятичного разделителя в сумме.
state	int	Код состояния транзакции.
subState	int	Код подсостояния транзакции.
inputType	TransactionInputType	Способ совершения транзакции.

reverseMode	TransactionItemReverseMode	Тип отмены транзакции который можно совершить.
displayMode	TransactionItemDisplayMode	Состояние транзакции для отображения.
products	NSArray *	Массив продуктов из каталога, каждый элемент TransactionProduct.
customFields	NSArray *	Массив полей пользовательского продукта, каждый элемент DescriptionProductField.
customFieldsProduct	DescriptionProduct *	Пользовательский продукт DescriptionProduct.
Card	Card *	Предоставляет доступ к данным карты.

15.1 Коды состояний:

0	Нет.
100	Транзакция создана.
200	Проверена возможность списания средств.
400	Транзакция выполнена.
500	Транзакция отменена.

15.2 Коды подсостояний:

101	Транзакция успешно создана.
201	Успешно проверена возможность списания средств.
202	Списание средств невозможно.
401	Платеж успешно прошел эквайринг (промежуточное состояние).
402	Платеж не прошел эквайринг (промежуточное состояние).
403	В системе эквайринга необходимо выполнить возврат платежа.
404	Платеж возвращен в системе эквайринга.
411	Платеж успешно выполнен.
412	Платеж не выполнен (служебное состояние).
502	Платеж успешно отменен.
503	Необходимо отменить платеж (промежуточное состояние, когда клиент пытается отменить платеж).
504	Платеж успешно возвращен.
505	Необходимо вернуть средства покупателю (промежуточное состояние, когда клиент пытается вернуть товар).

15.3 Перечисление TransactionInputType:

TransactionInputType_SWIPE	Оплата с помощью проката карты магнитной лентой.
TransactionInputType_EMV	Оплата с помощью чипа на карте.
TransactionInputType_NFC	Оплата с бесконтакного чипа на карте.
TransactionInputType_CASH	Оплата наличными.
TransactionInputType_PREPAID	Предоплата.
TransactionInputType_CREDIT	Постоплата.
TransactionInputType_LINK	Оплата по ссылке.

15.4 Перечисление TransactionItemReverseMode

TransactionInputReverseMode_NONE	Это состояние говорит о том, что время отпущеное на отменение транзакции истеко.
TransactionInputReverseMode_RETURN	Возможен возврат.
TransactionInputReverseMode_RETURN_PARTIAL	Возможен частичный возврат.
TransactionInputReverseMode_CANCEL	Возможна отмена
TransactionInputReverseMode_CANCEL_PARTIAL	Возможна частичная отмена.

15.5 Перечисление TransactionItemDisplayMode

TransactionInputDisplayMode_DECLINED	Отклоненная транзакция (в примере отображатся красным цветом).
TransactionInputDisplayMode_SUCCESS	Транзакция успешна (в примере отображатся черным цветом).
TransactionInputDisplayMode_REVERSE	Транзакция отмена (в примере отображатся серым цветом).
TransactionInputDisplayMode_REVERSED	Отмененная транзакция (в примере отображатся серым цветом и зачеркнутая).

16 Card

Предоставляет доступ к данным карты.

`iin`	NSString *	Тип карты или «cash» (в случае оплаты наличными).
`binID`	NSString *	Внутрений идентификатор банка.
`expiration`	NSString *	Срок действия карты.
`panMasked`	NSString *	Первые и последние 4 цифры номера карты, разделенные символом «*».
`panEnding`	NSString *	Последние 4 цифры номера карты.

17 DescriptionProduct

Предоставляет доступ к данным пользовательского продукта.

`ID`	`int`	Уникальный идентификатор продукта.
`state`	`DescriptionProductState`	Состояние продукта.
`inputType`	`DescriptionProductInputType`	Допустимый тип оплаты.
`fieldCount`	`int`	Количество полей данного продукта.
`code`	`NSString *`	Код продукта.
`title`	`NSString *`	Название продукта.
`fields`	`NSarray *`	Массив с полями продукта, каждый элемент DescriptionProductField.

17.1 Коды состояний:

DescriptionProductState_Disabled	Продукт отключен. Использование не разрешено.
DescriptionProductState_Enabled	Продукт включен.

17.2 DescriptionProductInputType

Типы оплаты продукта.

DescriptionProductInputType_Payment	Разрешено проводить обычные платежи.
DescriptionProductInputType_Recurrent	Разрешено проводить регулярные платежи.
DescriptionProductInputType_None	Продукт не разрешен к использованию.
DescriptionProductInputType_All	Разрешено проводить все виды платежей.

18 DescriptionProductField

Предоставляет доступ к данным поля пользовательского продукта.

ID	int	Уникальный идентификатор поля.
parentID	int	Уникальный идентификатор продукта к которому относится данное поле.
state	DescriptionProductFieldState	Состояние поля.
type	DescriptionProductFieldType	Тип поля.
required	BOOL	Признак того, что поле обязательное к заполнению.
textMultiline	BOOL	Признак многострочности поля.
code	NSString *	Код поля.
title	NSString *	Название поля.
textMask	NSString *	Регулярное выражение для ввода текста.
textRegExp	NSString *	Регулярное выражение для проверки введенного текста.
defaultValue	NSString *	Значение по умолчанию.
value	NSString *	Текущее значение поля.

18.1 Коды состояний:

DescriptionProductFieldState_Disabled	Поле отключено. Использование не разрешено.
DescriptionProductFieldState_Enabled	Поле включено.

18.2 DescriptionProductFieldType

DescriptionProductFieldType_None	Ошибка.
DescriptionProductFieldType_Text	Текстовое поле.
DescriptionProductFieldType_Image	Поле с изображением.

19 TransactionProduct

Предоставляет доступ к данным продукта каталога.

ID	NSString *	Уникальный идентификатор продукта.
Name	NSString *	Название продукта.
PriceName	NSString *	Название текущей цены продукта.
CategoryName	NSString *	Название категории к которой относится данн продукт.
ImageURLTN	NSString *	URL картинки продукта.
AmountFormat	NSString *	Формат суммы продукта.
CurrencySign	NSString *	Знак валюты.
Price	double	Общая сумма, учитывая количество.
UnitPrice	double	Сумма одного продукта.
Count	int	Количество продуктов.
HasImage	BOOL	Признак наличия картинки продукта.

20 SignatureView

Наследуется от GLKView. Предназначен для ввода подписи с экрана смартфона или планшета.

20.1 Методы

20.1.1 update

Обновить отображение объекта

20.1.2 updateWithOrientation:(BOOL)update

Обновить отображение с учетом ориентации экрана (если значение параметра «true») либо без.

20.1.3 isEmpty

Возвращает признак наличия введенного изображения.

20.1.4 getByteArray

Возвращает введенное изображение в формате PNG в NSData *

20.1.5 clear

Очистить поле ввода подписи.

21 Описание алгоритма совершения платежа.

1. Устанавливаем email(login) и пароль для входа в систему методом setEmail:Password:
2. Устанавливаем используемый тип ридера методом setReaderType:

3. Создаем и заполняем данными объект PaymentContext или RecurrentPaymentContext, в зависимости от типа предпологаемой транзакции.
4. Передаем объекту-синглтону PaymentController объект, который будет обрабатывать события, которые произойдут при совершении платежа, при помощи метода – [[PaymentController instance] setDelegate:]
5. Включаем ридер вызовом [[PaymentController instance] enable]
6. Ожидаем вызова события PaymentControllerDone:

Кроме события PaymentControllerDone: при обработке платежа также возможны вызовы событий

PaymentControllerRequestCardApplication: в случае если на карте присутствуют несколько ICC приложений

PaymentControllerRequestBTDevice: в случае если в зоне действия устройства находятся несколько Bluetooth ридеров.

PaymentControllerScheduleStepsStart: в случае если вы создаете расписание и начал выполненятся серверный вызова для получения шагов расписания.

PaymentControllerScheduleStepsCreated: в случае если вы создаете расписание и сервер ответил на запрос шагов, в качестве параметра пришел массив с объектами StepItem.

7. В случае если в событие PaymentControllerDone: передается `TransactionData со свойством` RequiredSignature установленным в истину, необходимо запросить у плательщика подпись. Это можно сделать с помощью объекта SignatureView.
8. Email или телефон плательщика, а также, при необходимости, картинка подписи передаются методом adjustWithTrId:Signature:ReceiptEmail:ReceiptPhone: или adjustWithScheduleId:Signature:ReceiptEmail:ReceiptPhone:, если это расписание.