

**ibox**  
**iboxPro**  
**iOS SDK description**

Версия 1.5.2 от 11 октября 2016 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 setBTDevice: .....	6
1.1.11 scheduleStepsConfirm.....	6
1.1.12 pingReaderWithDoneAction: .....	6
1.1.13 historyWithPage: .....	6
1.1.14 historyWithTransactionID: .....	6
1.1.15 adjustWithTrId:Signature:ReceiptEmail:ReceiptPhone: .....	6
1.1.16 adjustWithScheduleId:Signature:ReceiptEmail:ReceiptPhone: .....	7
1.1.17 reverseAdjustWithTrId:Signature:ReceiptEmail:ReceiptPhone: .....	7
1.1.18 setEmail:Password:.....	7
1.1.19 setReaderType:.....	7
1.1.20 setRequestTimeOut:.....	7
1.2 Значения числовых полей и перечисления.....	8
1.2.1 Перечисление PaymentControllerErrorType .....	8
1.2.2 Перечисление PaymentControllerReaderEventType .....	8
1.2.3 Перечисление PaymentControllerReaderType .....	9
2 Интерфейс PaymentControllerDelegate .....	9
2.1 Методы .....	9
2.1.1 PaymentControllerStartTransaction: .....	9
2.1.2 PaymentControllerDone:.....	9
2.1.3 PaymentControllerError:Message:.....	9

2.1.4 PaymentControllerReaderEvent: .....	9
2.1.5 PaymentControllerRequestCardApplication: .....	9
2.1.6 PaymentControllerRequestBTDevice: .....	10
2.1.7 PaymentControllerScheduleStepsStart .....	10
2.1.8 PaymentControllerScheduleStepsCreated: .....	10
3 PaymentContext .....	10
3.1 Перечисление CurrencyType .....	10
4 ReversePaymentContext .....	10
5 RecurrentPaymentContext .....	11
5.1 Перечисление ScheduleType .....	11
5.2 Перечисление ScheduleEndType .....	11
6 TransactionData .....	12
6.1 Перечисление TransactionDataType .....	12
7 StepItem .....	12
8 BTDevice .....	12
9 APIResult .....	12
10 APIHistoryResult .....	13
11 TransactionItem .....	13
11.1 Коды состояний: .....	14
11.2 Коды подсостояний: .....	14
11.3 Перечисление TransactionInputType: .....	14
11.4 Перечисление TransactionItemReverseMode .....	15
11.5 Перечисление TransactionItemDisplayMode .....	15
12 DescriptionProduct .....	15
12.1 Коды состояний: .....	15
12.2 DescriptionProductInputType .....	15
13 DescriptionProductField .....	16
13.1 Коды состояний: .....	16
13.2 DescriptionProductFieldType .....	16
14 TransactionProduct .....	16
15 SignatureView .....	17
15.1 Методы .....	17
15.1.1 update .....	17
15.1.2 updateWithOrientation:(BOOL)update .....	17

15.1.3 isEmpty .....	17
15.1.4 getByteArray.....	17
15.1.5 clear .....	17
16 Описание алгоритма совершения платежа.....	17

# 1 PaymentController

## 1.1 Методы

### 1.1.1 instance

Возвращает экземпляр 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 полученном приложением из события PaymentControllerRequestCardApplication:
-------------	-----	---

### 1.1.10 setBTDevice:

Устанавливает Bluetooth ридер, с которым будет производиться транзакция.

Параметры:

device	int	Номер ридера в массиве полученном из события PaymentControllerRequestBTDevice:
--------	-----	--

### 1.1.11 scheduleStepsConfirm

Подтверждает шаги расписания и дает возможность запустить это расписание.

### 1.1.12 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).
------------	--------------------------	---

### 1.1.13 historyWithPage:

При помощи этого метода можна получить историю транзакций пользователя.

page	int	Номер страницы (первая страница - 1).
------	-----	---------------------------------------

Возвращаемое значение:

APIHistoryResult

### 1.1.14 historyWithTransactionID:

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

transactionID	NSString *	Уникальный идентификатор транзакции.
---------------	------------	--------------------------------------

Возвращаемое значение:

APIHistoryResult

### 1.1.15 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

#### 1.1.16 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

#### 1.1.17 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

#### 1.1.18 setEmail:Password:

Устанавливает email и пароль для входа в систему.

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

#### 1.1.19 setReaderType:

Устанавливает тип используемого ридера.

readerType	PaymentControllerReaderType	Тип ридера.
------------	-----------------------------	-------------

#### 1.1.20 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_SUBMIT	Общая ошибка при проведении транзакции.
PaymentControllerErrorType_SUBMIT_CASH	Ошибка при проведении наличной операции
PaymentControllerErrorType_SUBMIT_PREPAID	Ошибка при проведении операции предоплаты.
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_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_Inititalize	Ридер инициализирован
PaymentControllerReaderEventType_Connect	Ридер подключен к разъему.
PaymentControllerReaderEventType_Disconnect	Ридер отключен от разъема.
PaymentControllerReaderEventType_CardInserted	Карта вставлена корректно.
PaymentControllerReaderEventType_SwipeCard	Карта проведена через считыватель магнитной полосы.
PaymentControllerReaderEventType_StartEMV	Начало EMV транзакции.



### 1.2.3 Перечисление PaymentControllerReaderType

Описывает доступные типы ридеров.

PaymentControllerReaderType_ChipAndSign	Chip&Sign ридер
PaymentControllerReaderType_ChipAndSignBT	Chip&Sign BT ридер
PaymentControllerReaderType_Chipper2X	Chipper2X ридер
PaymentControllerReaderType_ChipAndPIN	Chip&PIN ридер
PaymentControllerReaderType_ChipAndPIN2	Chip&PIN2 ридер
PaymentControllerReaderType_QPOS	QPOS ридер
PaymentControllerReaderType_QPOSMini	QPOS Mini ридер
PaymentControllerReaderType_QPOSAudio	QPOS Audio 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.
ProductData	NSArray *	Данные продукта, каждый элемент массива – объект NSDictionary *, ключем является код поля, а значение либо NSString * либо UIImage *, в зависимости от типа поля.
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 - воскресенье, в случае ScheduleType_Monthly, ScheduleType_Quarterly 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 выполняться 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.
CardNumber	NSString *	Маскированный номер карты.
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 APIHistoryResult

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

transactions	NSArray *	Массив объектов TransactionItem.
--------------	-----------	----------------------------------

## 11 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.
cardNumber	NSString *	Маскированный номер карты.
cardType	NSString *	Тип карты или "cash" (в случае оплаты наличными).
status	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 приложенной к транзакции.
photoURL	NSString *	URL картинки приложенной к транзакции.
scheduleID	NSString *	Уникальный идентификатор очереди.
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 транзакции.
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	TransactionItemReverse Mode	Тип отмены транзакции который можно совершить.
displayMode	TransactionItemDisplay Mode	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.

### 11.1 Коды состояний:

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

### 11.2 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).

### 11.3 Перечисление TransactionInputType:

TransactionInputType_Swipe	Оплата с помощью проката карты магнитной лентой.
TransactionInputType_Chip	Оплата с помощью чипа на карте.

TransactionInputType_Cash	Оплата наличными.
---------------------------	-------------------

## 11.4 Перечисление TransactionItemReverseMode

TransactionInputReverseMode_None	Это состояние говорит о том, что время отпущенное на отмену транзакции истекло.
TransactionInputReverseMode_Return	Возможен возврат.
TransactionInputReverseMode_ReturnPartial	Возможен частичный возврат.
TransactionInputReverseMode_Cancel	Возможна отмена
TransactionInputReverseMode_CancelPartial	Возможна частичная отмена.

## 11.5 Перечисление TransactionItemDisplayMode

TransactionInputDisplayMode_Declined	Отклоненная транзакция (в примере отображаются красным цветом).
TransactionInputDisplayMode_Success	Транзакция успешна (в примере отображаются черным цветом).
TransactionInputDisplayMode_Reverse	Транзакция отмена (в примере отображаются серым цветом).
TransactionInputDisplayMode_Reversed	Отмененная транзакция (в примере отображаются серым цветом и зачеркнутая).

## 12 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.

### 12.1 Коды состояний:

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

### 12.2 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.
---------------------------------	--

## 13 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.

### 13.1 Коды состояний:

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

### 13.2 DescriptionProductFieldType

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

## 14 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 данный 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.

## 15 SignatureView

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

### 15.1 Методы

#### 15.1.1 update

Обновить отображение объекта

#### 15.1.2 updateWithOrientation:(BOOL)update

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

#### 15.1.3 isEmpty

Возвращает признак наличия введенного изображения.

#### 15.1.4 getByteArray

Возвращает введенное изображение в формате PNG в NSData \*

#### 15.1.5 clear

Очистить поле ввода подписи.

## 16 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 предполагаемой 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 выполняться серверный вызов 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.