

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

Версия 1.7.4 от 10 сентября 2020 г.

## Содержание

1. PaymentController .....	5
1.1. Методы.....	5
1.1.1. instance.....	5
1.1.2. destroy .....	5
1.1.3. version .....	5
1.1.4. supportedReaders .....	5
1.1.5. readerInfo .....	5
1.1.6. setPaymentContext: .....	5
1.1.7. setClientProductCode: .....	5
1.1.8. enable .....	5
1.1.9. disable.....	5
1.1.10. retry .....	6
1.1.11. setDelegate: .....	6
1.1.12. isReaderConnected.....	6
1.1.13. setCardApplication: .....	6
1.1.14. saveBTDevice: .....	6
1.1.15. setBTDevice: .....	6
1.1.16. scheduleStepsConfirm .....	6
1.1.17. pingReaderWithDoneAction:.....	7
1.1.18. setSingleStepAuthentication: .....	7
1.1.19. authentication .....	7
1.1.20. historyWithPage: .....	7
1.1.21. historyWithTransactionID:.....	7
1.1.22. fiscalInfoWithTrId: .....	7
1.1.23. fiscalizeWithTrId: .....	7
1.1.24. adjustWithTrId:Signature:ReceiptEmail:ReceiptPhone:.....	8
1.1.25. adjustWithScheduleId:Signature:ReceiptEmail:ReceiptPhone: .....	8
1.1.26. reverseAdjustWithTrId:Signature:ReceiptEmail:ReceiptPhone: .....	8
1.1.27. paymentStatusWithTrId: .....	8
1.1.28. paymentStatusWithTrId:DoneAction: .....	8
1.1.29. setEmail:Password:.....	9
1.1.30. search4BTReadersWithType:.....	9
1.1.31. prepareWithProductCode:PrepareData: .....	9

1.1.32.	stopSearch4BTReaders .....	9
1.1.33.	setReaderType: .....	9
1.1.34.	setRequestTimeOut: .....	9
1.2.	Значения числовых полей и перечисления. ....	9
1.2.1.	Перечисление PaymentControllerErrorType .....	9
1.2.2.	Перечисление PaymentControllerReaderEventType.....	10
1.2.3.	Перечисление PaymentControllerReaderType.....	11
2.	Интерфейс PaymentControllerDelegate .....	11
2.1.	Методы.....	11
2.1.1.	PaymentControllerStartTransaction:.....	11
2.1.2.	PaymentControllerDone:.....	11
2.1.3.	PaymentControllerError:Message:.....	11
2.1.4.	PaymentControllerReaderEvent:.....	11
2.1.5.	PaymentControllerRequestCardApplication: .....	11
2.1.6.	PaymentControllerRequestBTDevice: .....	12
2.1.7.	PaymentControllerScheduleStepsStart .....	12
2.1.8.	PaymentControllerScheduleStepsCreated: .....	12
3.	PaymentContext.....	12
3.1.	Перечисление CurrencyType.....	12
4.	ReversePaymentContext .....	12
5.	RecurrentPaymentContext.....	13
5.1.	Перечисление ScheduleType.....	13
5.2.	Перечисление ScheduleEndType .....	13
6.	TransactionData .....	14
6.1.	Перечисление TransactionDataType.....	14
7.	StepItem .....	14
8.	BTDevice .....	14
9.	APIResult .....	15
10.	APIAuthenticationResult.....	15
11.	APIHistoryResult .....	15
12.	APIFiscalizeResult.....	15
13.	APIFiscalInfoResult.....	15
14.	APIPrepareResult .....	15
15.	Account .....	16

16.	PaymentOption .....	16
17.	Acquirer.....	16
18.	TransactionItem .....	17
18.1.	Коды состояний:.....	18
18.2.	Коды подсостояний: .....	18
18.3.	Перечисление TransactionInputType: .....	19
18.4.	Перечисление TransactionItemReverseMode .....	19
18.5.	Перечисление TransactionItemDisplayMode .....	19
19.	Card .....	19
20.	FiscalInfo .....	20
20.1.	Перечисление FiscalInfoStatus:.....	20
21.	ExternalPayment .....	20
21.1.	Перечисление ExternalPaymentType: .....	20
22.	QRData .....	20
23.	DescriptionProduct .....	21
23.1.	Коды состояний:.....	21
23.2.	Перечисление DescriptionProductRecurrentMode:.....	21
23.3.	Перечисление DescriptionProductInputType: .....	21
24.	DescriptionProductField .....	22
24.1.	Коды состояний:.....	22
24.2.	DescriptionProductFieldType .....	23
25.	TransactionProduct.....	23
26.	SignatureView .....	23
26.1.	Методы .....	23
26.1.1.	update.....	23
26.1.2.	updateWithOrientation:(BOOL)update .....	23
26.1.3.	isEmpty .....	23
26.1.4.	getBytesArray .....	23
26.1.5.	clear .....	24
27.	Описание алгоритма совершения платежа. ....	24

## 1. PaymentController

### 1.1. Методы

#### 1.1.1. instance

Возвращает экземпляр класса – объект singleton.

#### 1.1.2. destroy

Уничтожает объект singleton.

#### 1.1.3. version

Возвращает текущую версию SDK.

#### 1.1.4. supportedReaders

Возвращает массив типов поддерживаемых ридеров.

#### 1.1.5. readerInfo

Возвращает информацию о последнем подключенном ридере.

#### 1.1.6. setPaymentContext:

Устанавливает объект, содержащий данные платежа.

Параметры:

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

#### 1.1.7. setClientProductCode:

Устанавливает код продукта клиента.

Параметры:

clientProductCode	NSString *	Код продукта клиента.
-------------------	------------	-----------------------

#### 1.1.8. enable

Включает ридер

#### 1.1.9. disable

Выключает ридер

#### 1.1.10. **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.

#### 1.1.11. **setDelegate:**

Устанавливает объект-обработчик событий возможных при совершении транзакций.

Параметры:

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

#### 1.1.12. **isReaderConnected**

Проверяет подключен ли ридер.

Возвращаемое значение:

BOOL

#### 1.1.13. **setCardApplication:**

Устанавливает с каким приложением карты будет производиться транзакция.

Параметры:

Application	int	Номер приложения в массиве, полученном приложением из события PaymentControllerRequestCardApplication:
-------------	-----	--

#### 1.1.14. **saveBTDevice:**

Запоминает Bluetooth ридер, с которым будет производиться транзакция.

Параметры:

Device	BTDevice *	Объект, который содержит данные о ридере.
--------	------------	---

#### 1.1.15. **setBTDevice:**

Устанавливает Bluetooth ридер, с которым будет производиться транзакция.

Параметры:

Device	BTDevice *	Объект, который содержит данные о ридере.
--------	------------	---

#### 1.1.16. **scheduleStepsConfirm**

Подтверждает шаги расписания и дает возможность запустить это расписание.

#### 1.1.17. **pingReaderWithDoneAction:**

Возвращает текущее состояние ридера.

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

#### 1.1.18. **setSingleStepAuthentication:**

Задаёт признак однофакторной авторизации при проведении EMV транзакции.

singleStepAuthentication	BOOL	Признак однофакторной авторизации.
--------------------------	------	------------------------------------

#### 1.1.19. **authentication**

Возвращает данные аккаунта пользователя.

Возвращаемое значение:

APIAuthenticationResult

#### 1.1.20. **historyWithPage:**

Возвращает историю транзакций.

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

Возвращаемое значение:

APIHistoryResult

#### 1.1.21. **historyWithTransactionID:**

Возвращает транзакцию из истории по ID.

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

Возвращаемое значение:

APIHistoryResult

#### 1.1.22. **fiscalInfoWithTrId:**

Возвращает фискальные данные транзакции.

trId	NSString *	Уникальный идентификатор транзакции.
------	------------	--------------------------------------

Возвращаемое значение:

APIFiscalInfoResult

#### 1.1.23. **fiscalizeWithTrId:**

Иницирует фискализацию транзакции на сервере.

trId	NSString *	Уникальный идентификатор транзакции.
------	------------	--------------------------------------

Возвращаемое значение:

APIFiscalizeResult

#### 1.1.24. **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

#### 1.1.25. **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

#### 1.1.26. **reverseAdjustWithTrId:Signature:ReceiptEmail:ReceiptPhone:**

Вызывается для отправки дополнительных данных об 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

#### 1.1.27. **paymentStatusWithTrId:**

Вызывается для получения текущего состояния транзакции.

trId	NSString *	Уникальный идентификатор расписания, полученный в событии PaymentControllerDone:
------	------------	--

Возвращаемое значение:

APIResult

#### 1.1.28. **paymentStatusWithTrId:DoneAction:**

Возвращает 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 *.

#### 1.1.29. setEmail:Password:

Устанавливает email и пароль для входа в систему.

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

#### 1.1.30. search4BTReadersWithType:

Начать поиск bluetooth ридеров соответствующего типа.

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

#### 1.1.31. prepareWithProductCode:PrepareData:

Загрузка данных заказа для продукта, имеющего свойство preparable.

productCode	NSString *	Код продукта.
prepareData	NSDictionary *	Данные полей продукта необходимые для загрузки заказа. Каждая пара ключ-значение в словаре – это код поля и его текстовое значение.

#### 1.1.32. stopSearch4BTReaders

Прервать поиск блютуз ридеров.

#### 1.1.33. setReaderType:

Устанавливает тип используемого ридера.

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

#### 1.1.34. 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_ZERO_AMOUNT	Ошибка проведения транзакции. Нулевая сумма транзакции.
PaymentControllerErrorType_CARD_INSERTED_WRONG	Карта вставлена некорректно.
PaymentControllerErrorType_READER_DISCONNECTED	Ридер отключен.
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_OUTER_CARD	Ошибка при проведении транзакции внешней картой.
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 отмены транзакции.
PaymentControllerErrorType_REVERSE_CASH	Ошибка при попытке отмены наличной транзакции.
PaymentControllerErrorType_REVERSE_PREPAID	Ошибка при попытке отмены транзакции предоплаты.
PaymentControllerErrorType_REVERSE_CREDIT	Ошибка при попытке отмены транзакции постоплаты.
PaymentControllerErrorType_REVERSE_OUTER_CARD	Ошибка при попытке отмены транзакции внешней картой.
PaymentControllerErrorType_REVERSE_LINK	Ошибка при попытке отмены транзакции по ссылке.
PaymentControllerErrorType_REVERSE_CNP	Ошибка при попытке отмены транзакции без использования карты.
PaymentControllerErrorType_SCHEDULE_STEPS	Ошибка при создании 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.

### 1.2.2. Перечисление PaymentControllerReaderEventType

Описывает типы событий ридера.

PaymentControllerReaderEventType_INITIALIZED	Ридер инициализирован.
PaymentControllerReaderEventType_CONNECTED	Ридер подключен.
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.

#### 2.1.5. PaymentControllerRequestCardApplication:

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

#### 2.1.6. PaymentControllerRequestBTDevice:

Вызывается при необходимости выбрать Bluetooth ридер. Возвращает массив объектов BTDevice. Выбранный ридер 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	Сумма транзакции.
AmountCash	double	Принято наличными.
Description	NSString *	Описание транзакции.
ExtID	NSString *	Внешний ID платежа.
ReceiptMail	NSString *	Электронная почта для отправки чека.
ReceiptPhone	NSString *	Телефон для отправки чека.
ProductCode	NSString *	Код продукта.
Acquirer	NSString *	Код эквайера.
AuxData	NSString *	Данные внешнего платежа.
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.
Tags	NSArray *	Теги платежа.
Image	NSData *	Изображение, прикрепляемое к транзакции.
Currency	CurrencyType	Тип валюты.
InputType	TransactionInputType	Способ совершения транзакции.

#### 3.1. Перечисление CurrencyType

Тип валюты транзакции.

CurrencyType_RUB	Российский рубль.
CurrencyType_VND	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.
Card	Card *	Объект, который содержит данные о карте.
RequiredSignature	BOOL	Признак необходимости запросить подпись держателя.
Transaction	TransactionItem *	Объект с данными транзакции.
Type	TransactionDataType	Тип транзакции.

### 6.1. Перечисление TransactionDataType

Тип транзакции.

TransactionDataType_Payment	Обычная транзакция.
TransactionDataType_Schedule	Расписание.
TransactionDataType_REVERSE	Отмена.

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.

uuid	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.
products	NSArray *	Массив с продуктами пользователя, каждый элемент - DescriptionProduct.

## 11. APIHistoryResult

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

transactions	NSArray *	Массив объектов TransactionItem.
inProcessTransactions	NSArray *	Массив транзакций со статусом «В обработке».

## 12. APIFiscalizeResult

Объект содержащий ответ сервера на запрос фискализации транзакции. Порожден от базового класса APIResult.

transaction	TransactionItem *	Объект с данными транзакции.
-------------	-------------------	------------------------------

## 13. APIFiscalInfoResult

Объект содержащий ответ сервера на запрос получения данных о фискализации. Порожден от базового класса APIResult.

transaction	TransactionItem *	Объект с данными транзакции.
-------------	-------------------	------------------------------

## 14. APIPrepareResult

Объект содержащий ответ сервера на запрос о подготовке данны продукта. Порожден от базового класса APIResult.

fieldsData	NSArray *	Данные продукта, каждый элемент массива – объект NSDictionary *, ключем является код поля, а значение - NSString *.
------------	-----------	---

## 15. Account

Предоставляет доступ к информации об учетной записи.

singleStepAuthMode	BOOL	Признак доступности одношаговой авторизации.
pkString	NSString *	ID аккаунта.
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.
email	NSString *	Email аккаунта.
paymentOptions	NSArray *	Допустимые способы проведения транзакции.
acquirerWithCode:	Acquirer *	Объект эквайер с соответствующим кодом.

## 16. PaymentOption

Предоставляет доступ к данным объекта способа проведения транзакции.

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

## 17. Acquirer

Предоставляет доступ к данным эквайера.

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



## 18. 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 со 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, приложенной к транзакции.
photoURL	NSString *	URL картинки, приложенной 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 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.
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.
externalPayment	ExternalPayment *	Данные оплаты по ссылке в представлении ExternalPayment.
fiscalInfo	fiscalInfo	Фискальные данные платежа в представлении FiscalInfo

### 18.1. Коды состояний:

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

### 18.2. 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).
-----	--

### 18.3. Перечисление TransactionInputType:

TransactionInputType_SWIPE	Оплата с помощью проката карты магнитной полосой.
TransactionInputType_EMV	Оплата с помощью чипа на карте.
TransactionInputType_NFC	Оплата с помощью бесконтактного чипа на карте.
TransactionInputType_OUTER_CARD	Оплата внешним POS-терминалом.
TransactionInputType_CASH	Оплата наличными.
TransactionInputType_PREPAID	Предоплата.
TransactionInputType_CREDIT	Постоплата.
TransactionInputType_LINK	Оплата по ссылке.

### 18.4. Перечисление TransactionItemReverseMode

TransactionInputReverseMode_NONE	Отмена/возврат недоступны.
TransactionInputReverseMode_RETURN	Возможен возврат.
TransactionInputReverseMode_RETURN_PARTIAL	Возможен частичный возврат.
TransactionInputReverseMode_CANCEL	Возможна отмена.
TransactionInputReverseMode_CANCEL_PARTIAL	Возможна частичная отмена.
TransactionInputReverseMode_CANCEL_CNP	Транзакция может быть отменена в режиме CNP.
TransactionInputReverseMode_CANCEL_CNP_PARTIAL	Транзакция может быть частично отменена в режиме CNP.

### 18.5. Перечисление TransactionItemDisplayMode

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

## 19. Card

Предоставляет доступ к данным карты.

iin	NSString *	Тип карты или «cash» (в случае оплаты наличными).
binID	NSString *	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.

## 20. FiscalInfo

Предоставляет доступ к фискальным данным платежа.

status	FiscalInfoStatus	Статус фискализации.
printerSerialNumber	NSString *	ЗН ККТ
printerRegisterNumber	NSString *	РН ККТ
printerShift	NSString *	Номер смены
printerCryptographicVerificationCode	NSString *	КПК
printerDocSerialNumber	NSString *	Номер чека
documentNumber	NSString *	ФД
documentMark	NSString *	ФП
storageNumber	NSString *	ФН
dateTime	NSString *	Дата и время фискализации.
message	NSString *	Сообщение.
error	Int	Код ошибки фискализации.

### 20.1. Перечисление FiscalInfoStatus:

Статус фискализации.

FiscalInfoStatus_NONE	Не установлен
FiscalInfoStatus_CREATED	Фискализируется.
FiscalInfoStatus_SUCCESS	Фискализирована.
FiscalInfoStatus_FAILURE	Ошибка фискализации.

## 21. ExternalPayment

Предоставляет доступ к данным о платеже по ссылке.

type	ExternalPaymentType	Способ отображения данных.
data	NSArray *	Массив объектов типа QRData.

### 21.1. Перечисление ExternalPaymentType:

Способ отображения платежа.

ExternalPaymentType_LINK	Ссылка.
ExternalPaymentType_QR	QR код.

## 22. QRData

Объект содержащий данные ссылки или QR кода.

title	NSString *	Название.
value	NSString *	Содержимое.

## 23. DescriptionProduct

Предоставляет доступ к данным пользовательского продукта.

ID	Int	Уникальный идентификатор продукта.
state	DescriptionProductState	Состояние продукта.
inputType	DescriptionProductInputType	Допустимый тип оплаты.
recurrentMode	DescriptionProductRecurrentMode	Настройка регулярного платежа.
fieldsCount	Int	Количество полей данного продукта.
code	NSString *	Код продукта.
title	NSString *	Название продукта.
fields	NSArray *	Массив с полями продукта, каждый элемент DescriptionProductField.
preparableFields	NSArray *	Массив с полями продукта, каждый элемент DescriptionProductField.
preparableField	DescriptionProductField *	Preparable поле продукта.
firstTextField	DescriptionProductField *	Первое текстовое поле продукта.
preparable	BOOL	Признак того, что заполнение полей продукта должно выполняться результатом вызова метода prepareWithProductCode:PrepareData:
preparableOptional	BOOL	Признак того, что значения полей могут быть заполнены без вызова метода prepareWithProductCode:PrepareData:
preparableEditable	BOOL	Признак того, что разрешено редактирование загруженных значений полей.
visible	BOOL	Признак того, что продукт должен отображаться в пользовательском приложении.

### 23.1. Коды состояний:

DescriptionProductState_DISABLED	Продукт отключен. Использование не разрешено.
DescriptionProductState_ENABLED	Продукт включен.

### 23.2. Перечисление DescriptionProductRecurrentMode:

Настройка регулярного платежа для продукта.

DescriptionProductRecurrentMode_NONE	Не определено.
DescriptionProductRecurrentMode_DEFAULT	График и настройки регулярного платежа передаются при создании платежа.
DescriptionProductRecurrentMode_MANAGED	График и настройки регулярного платежа устанавливаются сервером и не передаются при платеже.

### 23.3. Перечисление 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.

## 24. 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.
preparable	BOOL	Значение поля используется при вызове метода prepareWithProductCode:PrepareData:
textMultiline	BOOL	Признак многострочности поля.
receiptMail	BOOL	Признак того, что значение поля должно использоваться как email для отправки чека.
receiptPhone	BOOL	Признак того, что значение поля должно использоваться как телефон для отправки чека.
userVisible	BOOL	Признак того, что поле доступно для пользователя.
isNumeric	BOOL	Признак того, что значение поля является десятичным числом.
printing	BOOL	Признак того, что поле должно быть напечатано в чеке.
isFiscalTag	BOOL	Признак фискального тега.
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.

### 24.1. Коды состояний:

DescriptionProductFieldState_DISABLED	Поле отключено. Использование не разрешено.
DescriptionProductFieldState_ENABLED	Поле включено.

## 24.2. DescriptionProductFieldType

Тип поля.

DescriptionProductFieldType_NONE	Ошибка.
DescriptionProductFieldType_TEXT	Текстовое поле.
DescriptionProductFieldType_IMAGE	Поле с изображением.

## 25. 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.

## 26. SignatureView

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

### 26.1. Методы

#### 26.1.1. update

Обновить отображение объекта

#### 26.1.2. updateWithOrientation:(BOOL)update

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

#### 26.1.3. isEmpty

Возвращает признак наличия введенного изображения.

#### 26.1.4. getByteArray

Возвращает введенное изображение в формате PNG в NSData \*

#### 26.1.5. clear

Очистить поле ввода подписи.

## 27. Описание алгоритма совершения платежа.

1. Устанавливаем email(login) и секретный ключ 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.