

TINKOFF ACQUIRING SDK FOR IOS



Содержание

| | | |
|-----|---|----|
| 1. | ТЕРМИНЫ И СОКРАЩЕНИЯ | 4 |
| 2. | ОСНОВНЫЕ ПОЛОЖЕНИЯ | 5 |
| | Требования и ограничения | 5 |
| | Подготовка к использованию | 5 |
| 3. | ПРОВЕДЕНИЕ ОПЛАТЫ | 10 |
| | Инициализация платежа | 10 |
| | Завершение платежа | 11 |
| | Схема проведения платежа | 12 |
| | Интеграция с онлайн-кассами | 14 |
| 4. | ОПЛАТА ТОВАРА ЧЕРЕЗ APPLE PAY | 15 |
| | Подключение | 15 |
| | Схема взаимодействия | 22 |
| 5. | РЕКУРРЕНТНЫЙ ПЛАТЕЖ С ПРИВЯЗАННОЙ КАРТЫ | 24 |
| 6. | ПРОВЕДЕНИЕ ОПЛАТЫ С ПОМОЩЬЮ СБП | 26 |
| | Включение оплаты платежей через СБП в SDK | 26 |
| | Инициализация платежа | 26 |
| | Завершение платежа | 27 |
| 7. | ПРИЕМ ПЛАТЕЖЕЙ ЧЕРЕЗ СИСТЕМУ БЫСТРЫХ ПЛАТЕЖЕЙ (СБП) | 28 |
| | Включение приема платежей через СБП в SDK | 28 |
| | Сгенерировать QR-код для приема платежей | 28 |
| 8. | СПИСОК СОХРАНЕННЫХ КАРТ | 30 |
| | Сохранение новой карты | 30 |
| | Схема работы | 31 |
| 9. | ТЕСТОВОЕ ПРИЛОЖЕНИЕ | 34 |
| | Структура | 34 |
| | API Методы для работы с сервером | 35 |
| | Дополнительная информация | 57 |
| 10. | Коды ошибок API и возможные исключения | 59 |
| 11. | ПОДДЕРЖКА | 62 |

История изменений

| Версия | Описание | Вносил изменения | Дата |
|--------|---|-------------------|------------|
| 1.0 | Первоначальное составление документа | Никита Кашин | 20.10.2017 |
| 2.0 | Дополнено описание методов, информация актуализирована | Никита Кашин | 29.11.2017 |
| 3.0 | Изменения в структуре документа, корректировка ошибок, минорные изменения | Никита Кашин | 26.12.2017 |
| 4.0 | Обновлены методы оплаты SDK, скриншоты и добавлено описание СПБ | Будников Вячеслав | 24.04.2020 |

1. Термины и сокращения

| Термин | Определение |
|----------------------|--|
| Продавец | Участник, принимающий через интернет в свою пользу платежи по банковским картам за товары и услуги через протокол Merchant API |
| Клиент | Участник, производящий оплату с использованием банковской карты на сайте продавца |
| Терминал | Точка приема платежей продавца (в общем случае привязывается к сайту, на котором осуществляется прием платежей) |
| 3-D Secure | Протокол, который используется как дополнительный уровень безопасности при осуществлении онлайн-платежей с банковских карт. 3-D Secure добавляет ещё один шаг аутентификации при совершении онлайн-платежей |
| PCI DSS | Payment Card Industry Data Security Standard, стандарт безопасности данных индустрии платежных карт. Стандарт утверждён международными платежными системами Visa и MasterCard, American Express, JCB и Discover. Стандарт представляет собой совокупность 12 детализированных требований по обеспечению безопасности данных о держателях платёжных карт, которые передаются, хранятся и обрабатываются в информационных инфраструктурах организаций. |
| Интернет-эквайринг | Технология приёма оплаты в интернете через платёжную страницу банка-эквайера. С помощью интернет-эквайринга покупатели оплачивают покупки на сайтах, в мобильных приложениях, по прямым ссылкам на страницу с платёжной формой банка. |
| Оплата | Операция с использованием карты и с обязательной авторизацией, проводимой банком-эквайером по поручению владельца карты в магазине в целях покупки товаров или услуг, и по которой был сформирован счёт. |
| Рекуррентные платежи | <p>Регулярная оплата с банковской карты без подтверждения владельца карты. Банк-эквайер списывает оплату по графику, заранее оговоренному покупателем и магазином. По правилам интервал между двумя оплатами не превышает одного года.</p> <p>Магазин и покупатель заранее договариваются, какие товары или услуги магазин предоставляет покупателю, пока действует соглашение об оплате регулярными платежами. Соглашением о регулярных платежах служит первичная оплата покупателем стандартным способом — с вводом реквизитов карты и проверкой 3-D Secure.</p> |

2. Основные положения

Acquiring SDK позволяет интегрировать Интернет-Эквайринг в мобильные приложения для платформы iOS.

Возможности SDK:

- Прием платежей (в том числе рекуррентных);
- Сохранение банковских карт клиента;
- Сканирование и распознавание карт с помощью камеры;
- Работа с привязанными картами;
- Поддержка английского;
- Интеграция с онлайн-кассами;
- Оплата с помощью [ApplePay](#);
- Оплата с помощью Системы Быстрых Платежей;
- Настройки окна оплаты.

Требования и ограничения

Для работы Tinkoff Acquiring SDK необходимо:

- Поддержка iOS 11 и выше;

Подготовка к использованию

Подключение

Рекомендуется использовать [Cocoa Pods](#).

Для подключения добавьте в файл Podfile зависимости:

```
pod 'CardIO'  
pod 'ASDKCore'  
pod 'ASDKUI'
```

Если вы не используете Cocoa Pods, необходимо добавить ASDKUI.xcodeproj в проект.

Подготовка к работе

Для начала работы с SDK вам понадобятся:

- **Terminal key** - терминал Продавца;
- **Password** - пароль от терминала;
- **Public key** – публичный ключ. Используется для шифрования данных. Необходим для интеграции вашего приложения с интернет-эквайрингом Тинькофф.

Данные выдаются в личном кабинете после подключения к [Интернет-Эквайрингу](#).

Для получения данных необходимо:

1. Выбрать раздел «**Терминалы**» в дополнительном меню в профиле магазина.

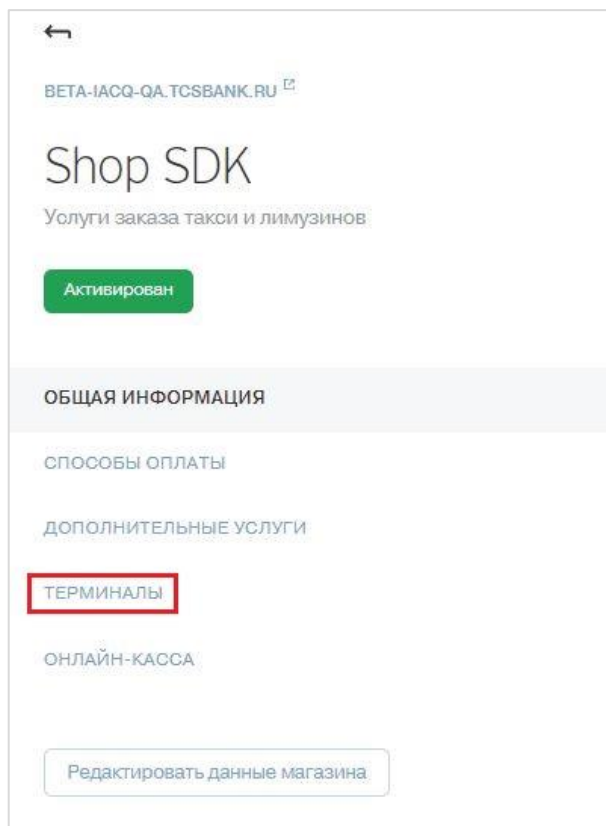


Рисунок 1. Раздел "Терминалы"

Выбрать терминал и нажать кнопку НАСТРОИТЬ.

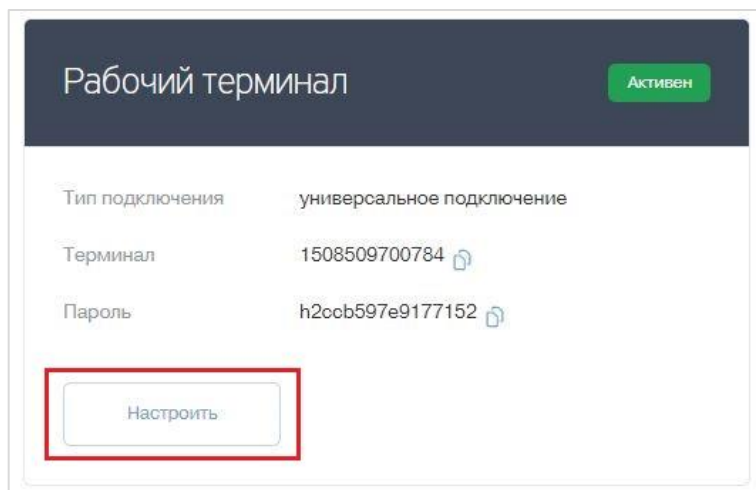
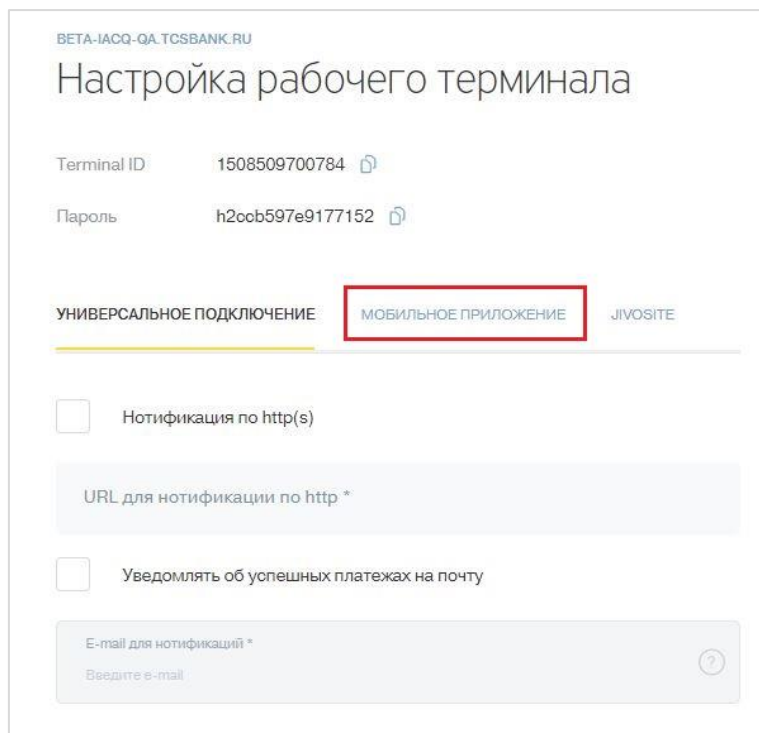


Рисунок 2. Терминал

2. Выбрать вкладку «**Мобильное приложение**».

Примечание. Если в настройках терминала выбрана не вкладка «Мобильное

приложение», а «Универсальное подключение» или «Jivosite», это может привести к сбросу настроек терминала для типа подключения: Мобильный SDK.



BETA-IACQ-QA.TCSBANK.RU

Настройка рабочего терминала

Terminal ID 1508509700784

Пароль h2ccb597e9177152

УНИВЕРСАЛЬНОЕ ПОДКЛЮЧЕНИЕ **МОБИЛЬНОЕ ПРИЛОЖЕНИЕ** JIVOSITE

☐ Нотификация по http(s)

URL для нотификации по http *

☐ Уведомлять об успешных платежах на почту

E-mail для нотификаций *

Введите e-mail ?

Рисунок 3. Настройка терминала

3. На вкладке указаны:

- **Terminal ID** – Terminal Key;
- **Пароль** - Password;
- **Открытый ключ** – Public Key.

Для завершения работы необходимо нажать кнопку СОХРАНИТЬ.

BETA-IACQ-QA.TCSBANK.RU

Настройка рабочего терминала

Terminal ID

1508509700784

Пароль

h2cob597e9177152

УНИВЕРСАЛЬНОЕ ПОДКЛЮЧЕНИЕ

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ

JIVOSITE

☐

Нотификация по http(s)

URL для нотификации по http *

☐

Уведомлять об успешных платежах на почту

Е-mail для нотификаций *

Введите e-mail

Открытый ключ

MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA5y9ka3ZQ
E0feuGtemYv3lqOILok8zHUM7ITr0za6IXTszRSXfUO7jMb+L5C7e2QNFs+
7sIX2OQJ6a+HG8kr+jwJ4tS3cVsWtd9NXpsU40PE4MeNr5RqiNXjcDxA+
L4OsEm/BlyFOEOh2epGyYUd5/iO3OiQFRNicomT2saQYAeqIwuELPs1Xp
Lk9HLx5qPbm8fRrQhjeUD5TLO8b+4yCnObe8vy/BMUwBfq+ieWADljwW
CMp2KTPMGLz48qnaD9kdrYJ0iyHqzb2mkDhdlzkim24A3IWoytJCBrrB2
xM05sm9+OdCI1f7nPNJbl5URHobSwR94IRGT7CJcUjwWDAQAB

Отмена

Сохранить

Рисунок 4. Данные терминала

Начало работы

В начале нужно создать конфигурацию, используем объект **AcquiringSdkConfiguration**, обязательные параметры:

- **credential:** **AcquiringSdkCredential** - учетные данные, полученные в личном кабинете
- **serverEnvironment:** **AcquiringSdkEnvironment** - тестовый или боевой сервер

Дополнительные параметры которые можно установить:

- **logger** - Интерфейс для логирования работы, есть реализация по умолчанию **ASDKApiLoggerDelegate** форматированный вывод данных о работе SDK в консоль

- `language` - Язык платежной формы. На каком языке сервер будет присылать тексты ошибок клиенту.
- `showErrorAlert: Bool` - Показывать ошибки после выполнения запроса системным `UIAlertViewController`

после инициализации SDK им можно пользоваться и вызывать форму для оплаты, и список карт.

Для проведения оплаты в модуле `ASDKUI/AcquiringUISDK` реализованы методы для полного сценария оплаты. Доступны методы:

- `presentPaymentView(paymentData: PaymentInitPaymentData)` показать форму оплаты для выбора источника оплаты (сохраненная карта или реквизиты новой) и оплатить
- `presentPaymentView(paymentData: PaymentInitPaymentData, parentPatmentId: Int64)` оплатить рекуррентный платеж
- `presentPaymentSBP(paymentData: PaymentInitPaymentData)` оплатить используя Систему Быстрых Платежей
- `presentPaymentAcceptanceQR` сгенерировать и показать QR-код для приема платежей
- `paymentApplePay(paymentData: PaymentInitPaymentData)` оплатить используя ApplePay

Для работы со списком сохраненных карты реализованы метод

- `presentCardList` показать список сохраненных карт, и отредактировать список карт - добавить карту, удалить карту

3. Проведение оплаты

Для оплаты нужно сформировать информацию о платеже [PaymentInitData](#).

Инициализация платежа

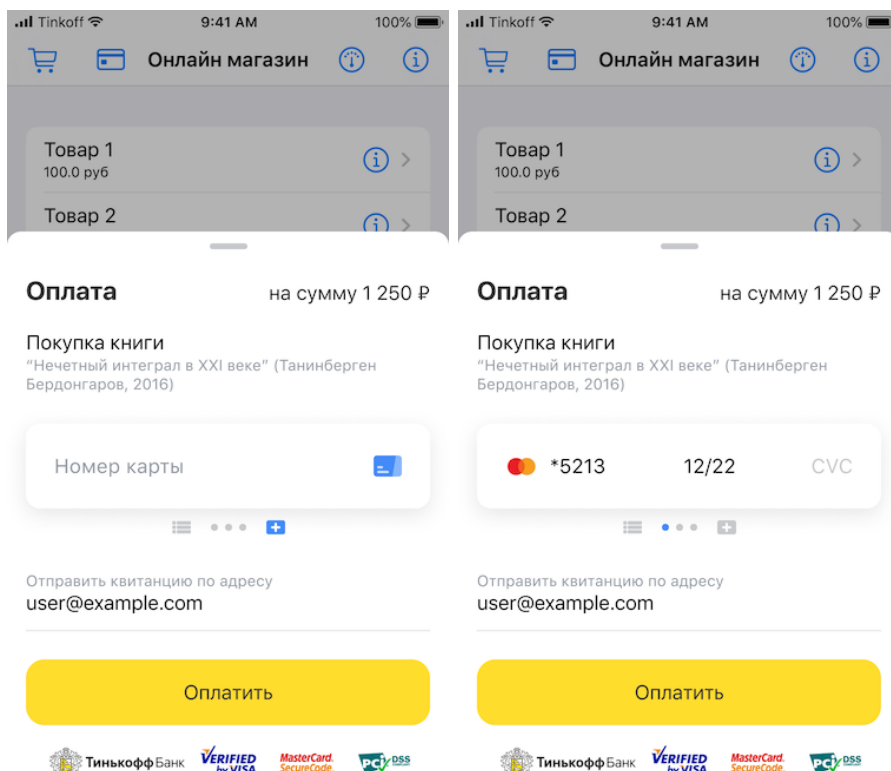
Для проведения оплаты нужно сформировать информацию о платеже [PaymentInitData](#):

- **amount**: [Int64](#). Сумма в копейках. Например, сумма 3руб. 12коп. это число `312`. Параметр должен быть равен сумме всех товаров в чеке.
- **orderId**: [Int64](#). Номер заказа в системе Продавца.
- **customerKey**: [String](#). Идентификатор клиента в системе продавца. Например, для этого идентификатора будут сохраняться список карт.
- **description**: [String](#). Краткое описание покупки.
- **payType**: [PayType](#). Тип проведения платежа
- **savingAsParentPayment**: [Bool](#). Если передается и установлен в `true`, то регистрирует платёж как рекуррентный (родительский).
- **paymentFormData**: [[String](#): [String](#)] `JSON` объект, содержащий дополнительные параметры в виде `[Key: Value]`. `Key` [String](#) – 20 знаков, `Value` [String](#) – 100 знаков. Максимальное количество пар параметров не может превышать 20.
- **receipt**: [Receipt](#) Данные чека
- **shops**: [[Shop](#)] информация о магазинах
- **receipts**: [[Receipt](#)] информация о чеках для онлайн магазинов

Далее вызываем метод [AcquiringUISDK.presentPaymentView](#). Перед началом оплаты открывается форма оплаты товара и запускается метод [Init](#) результатом которого является регистрация и получение информации о платеже для оплаты [PaymentInitResponse](#):

- **success**: [Bool](#) - статус обработки запроса
- **errorCode**: [Int](#) - номер ошибки в случае ошибки, по умолчанию 0
- **errorMessage**: [String?](#) - описание ошибки
- **errorDetails**: [String?](#) - детальное описание ошибки
- **terminalKey**: [String?](#) - идентификатор терминала на котором инициализирован платеж
- **amount**: [Int64](#) - сумма платежа в копейках
- **orderId**: [Int64](#) - номер заказа в системе продавца
- **paymentId**: [Int64](#) - номер для оплаты в системе банка
- **status**: [PaymentStatus](#) - статус платежа, подробнее:
https://oplata.tinkoff.ru/landing/develop/documentation/processing_payment

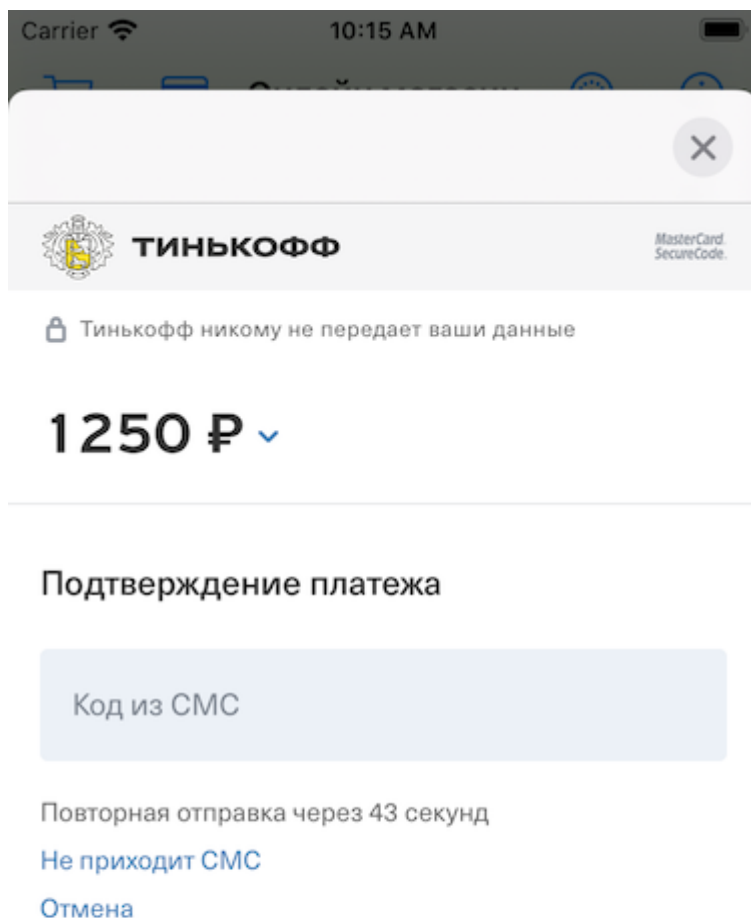
в случае удачной регистрации платежа на форме оплаты показываются список карт, карточка ввода реквизитов новой карты и кнопка **Оплатить**. Пользователь выбирает карту, либо вводит реквизиты новой нажимает и нажимает **Оплатить**.



Завершение платежа

Вызывается метод `FinishAuthorize` с реквизитами карты или `FinishAuthorize` с номером ранее сохраненной карты.

Во время проведения платежа сервер может запросить подтверждение в этом случае пользователю показывается форма 3D Secure в зависимости от выбранного источника оплаты, это решение принимается сервером.



Если процедура 3D Secure пройдена успешно то отображается уведомление о результате оплаты, если нет то отображается уведомление об ошибке.

Результатом `FinishAuthorize` является `Result<PaymentStatusResponse, Error>`

- `Error` - в случае ошибки
- `PaymentStatusResponse` - если платеж прошел успешно

Схема проведения платежа

Полная схема: <https://oplata.tinkoff.ru/landing/images/scheme.svg>

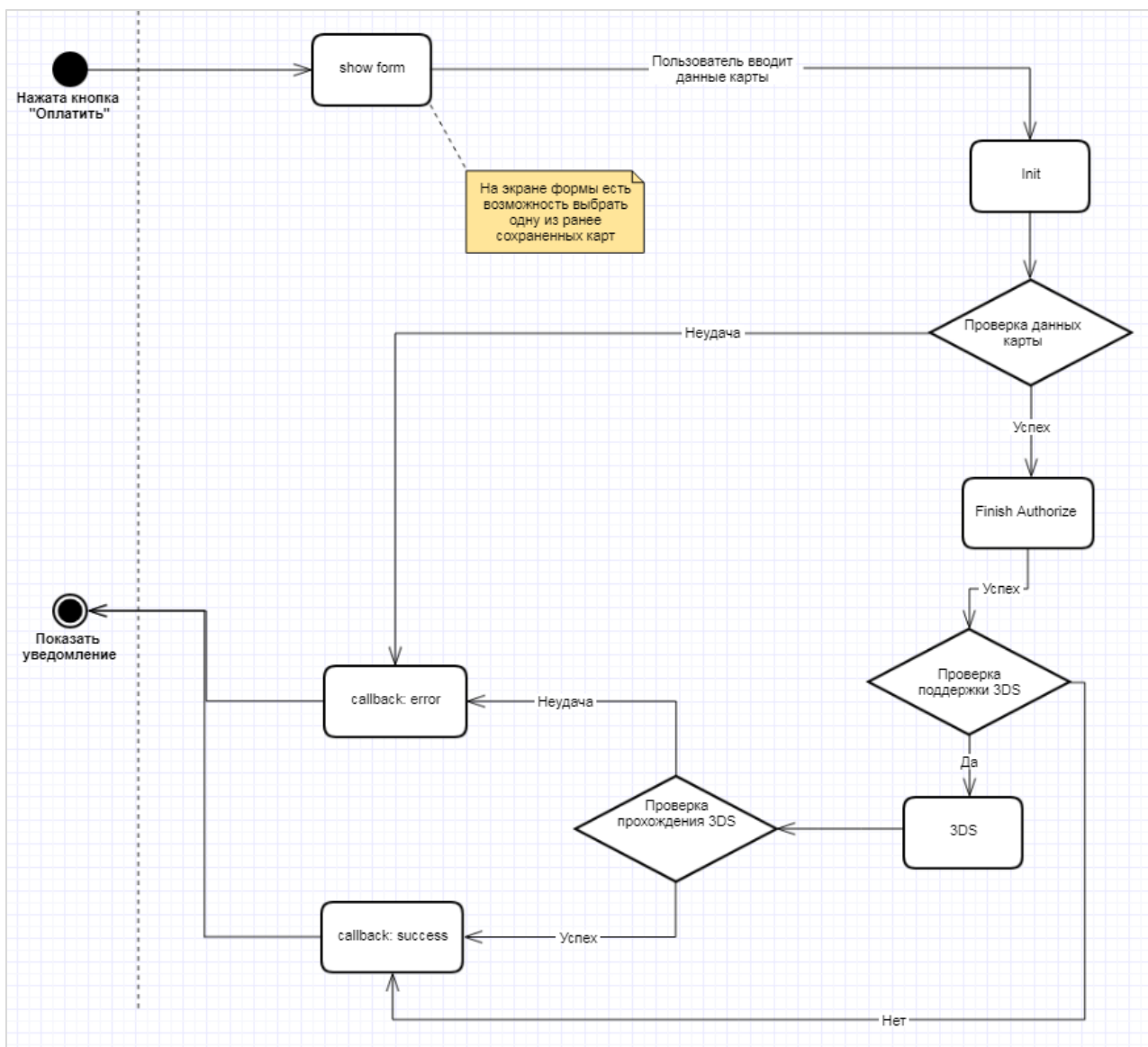


Рисунок 5. Диаграмма состояний платежа

При нажатии кнопки ОПЛАТИТЬ открывается форма ввода данных карты.

4. Пользователь вводит данные.

5. Запускается метод Метод Init.

6. Совершается проверка данных карты:

Если проверка прошла успешно – запускается метод (*) В случае если у терминала включена опция привязки покупателя после успешной оплаты и передается параметр **CustomerKey**, то в передаваемых параметрах **DATA** могут присутствовать параметры команды **AddCustomer**. Если они присутствуют, они автоматически привязываются к покупателю.

Например, если указать:

`"DATA":{"Phone":"+71234567890", "Email":"a@test.com"}`

к покупателю автоматически будут привязаны данные Email и телефон, и они будут возвращаться при вызове метода *GetCustomer*.

Таблица 3. Структура объекта Receipt

| Наименование | Тип | Обязательность | Описание |
|--------------|---------------------|----------------|--|
| Items | Массив объектов | Да | Массив, содержащий в себе информацию о товарах |
| Email | String | Да | Электронная почта |
| Phone | String | Нет | Телефон |
| Taxation | Перечисление (Enum) | Да | Система налогообложения. Перечисление со значениями: - «osp» – общая СН; - «usn_income» – упрощенная СН (доходы); - «usn_income_outcome» – упрощенная СН (доходы минус расходы); - «envd» – единый налог на вмененный доход; - «esn» – единый сельскохозяйственный налог; - «patent» – патентная СН. |

Таблица 4. Структура объекта Items

| Наименование | Тип | Обязательность | Описание |
|--------------|--------|----------------|--|
| Name | String | Да | Наименование товара. Максимальная длина строки – 128 символов |
| Price | Number | Да | Цена в копейках Целочисленное значение не более 10 знаков |
| Quantity | Number | Да | Количество/вес: - целая часть не более 8 знаков; - дробная часть не более 3 знаков |

| Наименование | Тип | Обязательность | Описание |
|--------------|---------------------|----------------|---|
| Amount | Number | Да | Сумма в копейках. Целочисленное значение не более 10 знаков |
| Tax | Перечисление (Enum) | Да | Ставка налога. Перечисление со значениями: - «none» – без НДС; - «vat0» – НДС по ставке 0%; - «vat10» – НДС чека по ставке 10%; - «vat18» – НДС чека по ставке 18%; - «vat110» – НДС чека по расчетной ставке 10/110; - «vat118» – НДС чека по расчетной ставке 18/118 |
| Ean13 | String | Нет | Штрих-код |
| ShopCode | String | Нет | Код магазина. Для параметра ShopCode необходимо использовать значение параметра Submerchant_ID, полученного в ответ при регистрации магазинов через xml. Если xml не используется, передавать поле не |

Пример запроса:

```
{
  "TerminalKey": "TestB",
  "Amount": "140000",
  "OrderId": "21050",
  "Description": "Подарочная карта на 1000 рублей",
  "Token": "2ED30E046136931431B5251B7C9A1EAC68DAB082203BD42676BA14A851359DF4",
  "DATA": { "Phone": "+71234567890", "Email": "a@test.com" },
  "Receipt": {
    "Email": "a@test.ru",
    "Phone": "+79031234567",
    "Taxation": "osn",
    "Items": [
```

```
{
  "Name": "Наименование товара 1",
  "Price": 10000,
  "Quantity": 1.00,
  "Amount": 10000,
  "Tax": "vat10",
  "Ean13": "0123456789",
  "ShopCode": "12345"
},
{
  "Name": "Наименование товара 2",
  "Price": 20000,
  "Quantity": 2.00,
  "Amount": 40000,
  "Tax": "vat18"
},
{
  "Name": "Наименование товара 3",
  "Price": 30000,
  "Quantity": 3.00,
  "Amount": 90000,
  "Tax": "vat10"
}
]
}
```

Таблица 5. Параметры ответа

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| Amount | Number | Да | Сумма в копейках |
| OrderId | String | Да | Номер заказа в системе Продавца |
| Success | bool | Да | Успешность операции |
| Status | String | Да | Статус транзакции |
| PaymentId | Number | Да | Уникальный идентификатор транзакции в системе Банка |

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|--|
| ErrorCode | String | Да | Код ошибки, «0» - если успешно |
| PaymentURL | String | Нет | Ссылка на страницу оплаты. Не передается, если ввод данных карты осуществляется на стороне Продавца. <i>Доступна в течении 24 часов по умолчанию.</i> |
| Message | String | Нет | Краткое описание ошибки |
| Details | String | Нет | Подробное описание ошибки |

Пример ответа:

```
{ "Success":true,"ErrorCode":"0","TerminalKey":"TestB","Status":"NEW","PaymentId":"13660",
  "OrderId":"21050","Amount":"100000",
  "PaymentURL":"https://rest-api-test.tcsbank.ru/rest/Authorize/1b63d14a-4208-44a8-a288-ad1b04008e51" }
```

Статус платежа:

при успешном сценарии: **NEW**;

при неуспешном: **REJECTED**.

- a) Метод FinishAuthorize.
- b) Если нет – операция отменяется, показывается уведомление об ошибке.

7. Совершается проверка на поддержку терминалом 3-D Secure:

- c) Если 3-D Secure есть и проверка прошла успешно – операция успешна, отображается уведомление об успехе операции.
- d) Если 3-D Secure есть и проверка не прошла – отображается уведомление об ошибке.
- e) Если 3-D Secure нет – операция успешна, отображается уведомление об успехе операции.

Интеграция с онлайн-кассами

Примечание. Для перехода со старой версии SDK на актуальную необходимо заменить URL методов на <https://securepay.tinkoff.ru/v2/>.

Для подключения к онлайн-кассам необходимо сформировать данные чека, указав параметры при инициализации платежа,

PaymentInitPaymentData:

- shops
- receipts

Описание параметра receipts см. в Таблица 4. Структура объекта [Receipt](#).

При передаче данных в метод происходит передача данных чека по операции.

4. Оплата товара через Apple Pay

Подключение

Для использования Apple Pay нужно интегрировать мобильный SDK в приложение iOS

Шаг 1. Подключение услуги в личном кабинете.

Для подключения Apple Pay необходимо:

- а) Перейти в личный кабинет.
- б) Выбрать магазин.
- с) Открыть вкладку «Дополнительные услуги».

Нажать кнопку Подключить.

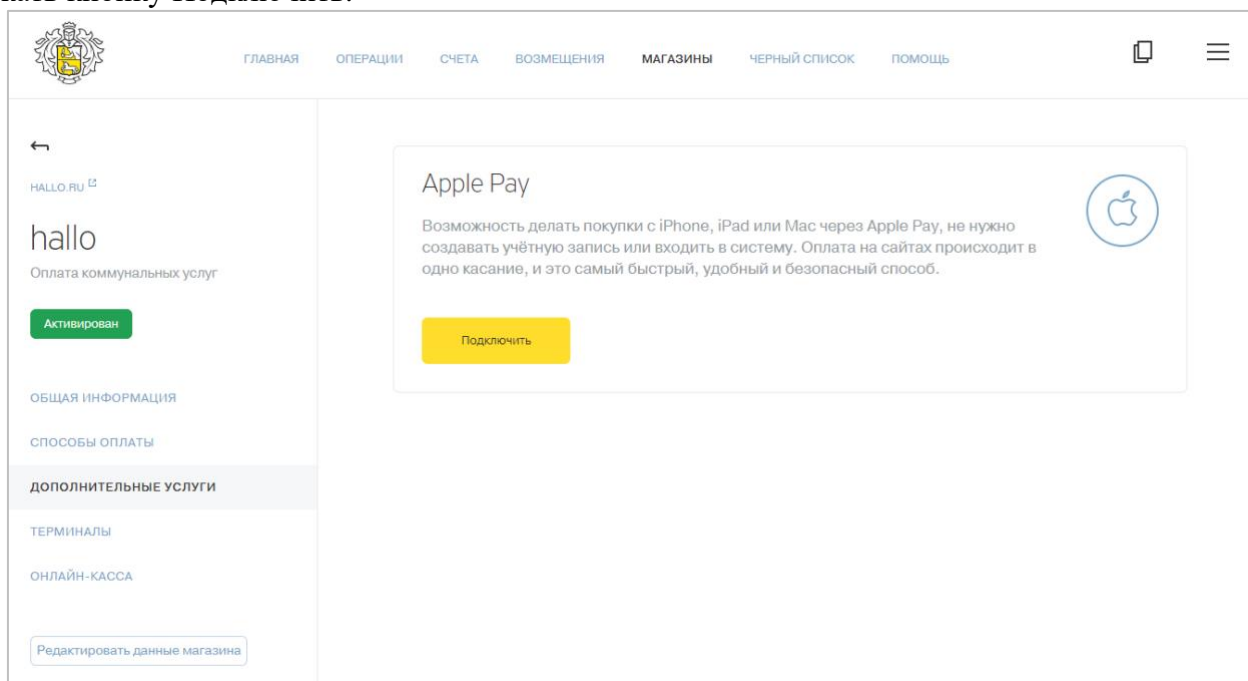


Рисунок 6. Подключение услуги в личном кабинете. Шаг 1

Шаг 2. **Получение сертификата**

Для получения сертификата необходимо скачать CSR-файл (запрос на сертификат) и направить в AppleDeveloper. (<https://developer.apple.com/>)

Шаг 3. Получение сертификата от Apple.

Внимание! Вам потребуется активная учётная запись MerchantID в AppleDeveloper.

Для этого нужно:

1. Создать MerchantID для оплаты.

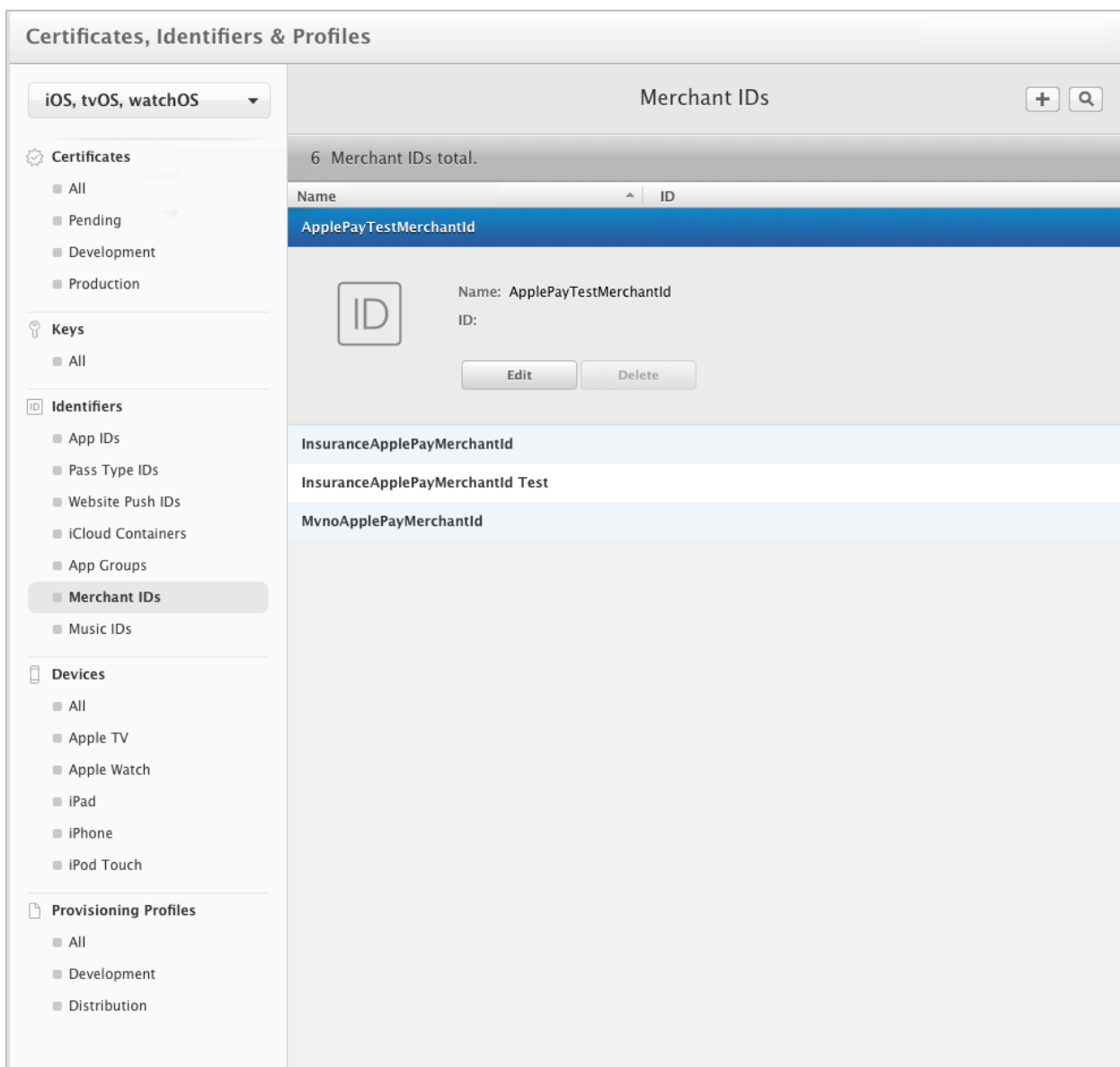


Рисунок 7. Создание MerchantID

2. Подключить MerchantID к AppID.

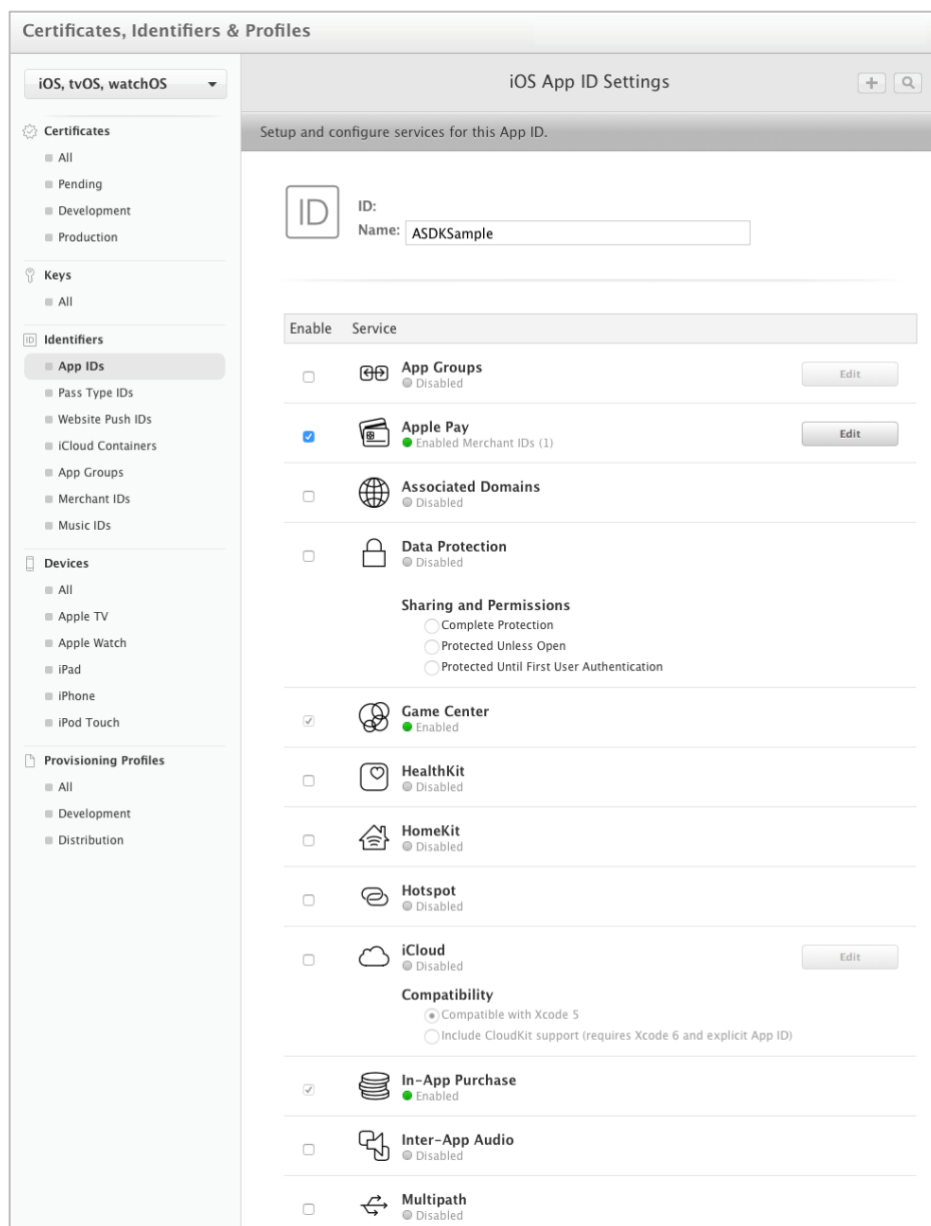


Рисунок 8. Подключение MerchantID

3. Выбрать MerchantID для подключения.

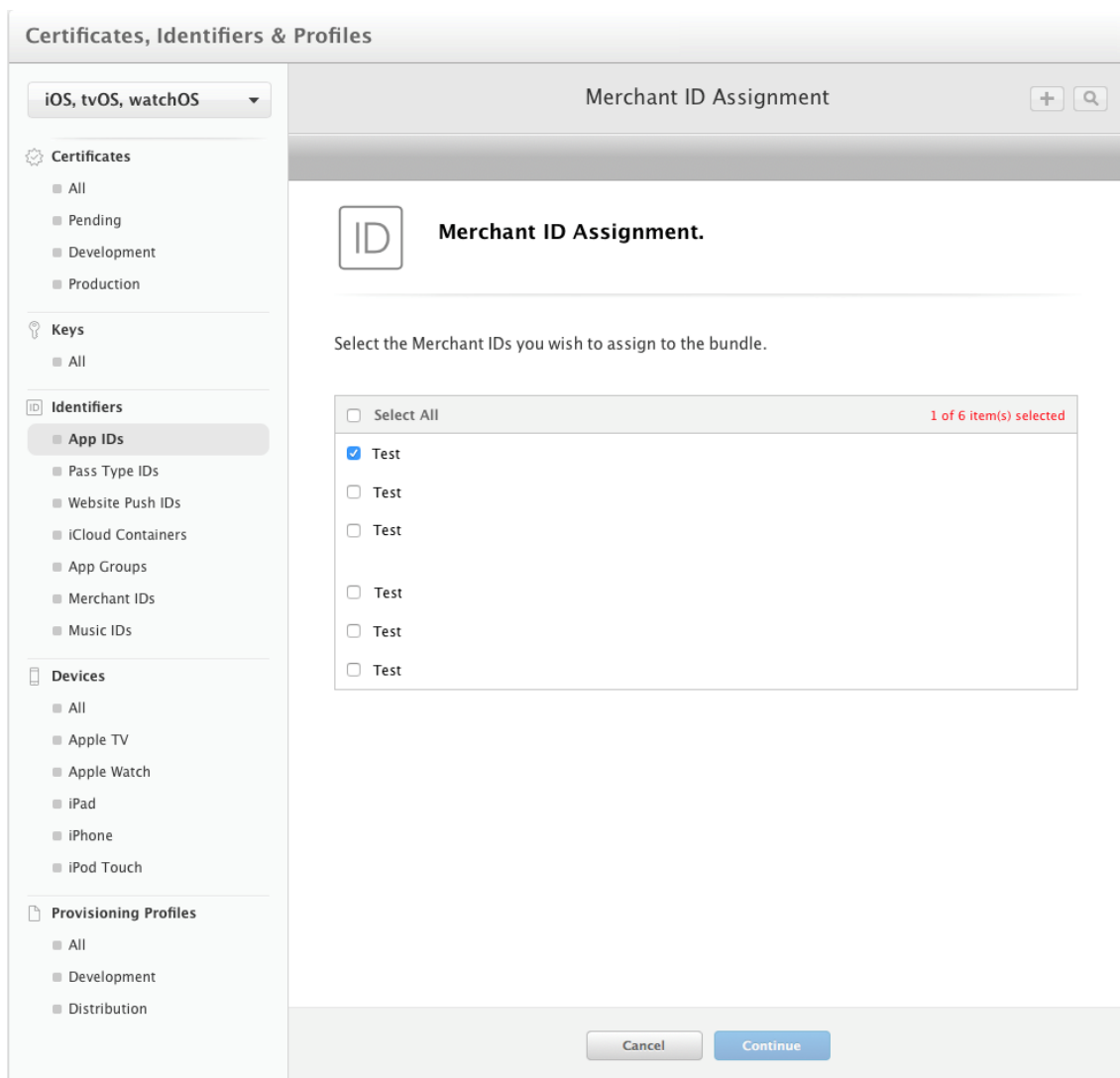


Рисунок 9. Выбор MerchantID

4. Получить сертификат для подписи сборки. Он позволяет работать с токенами Apple Pay

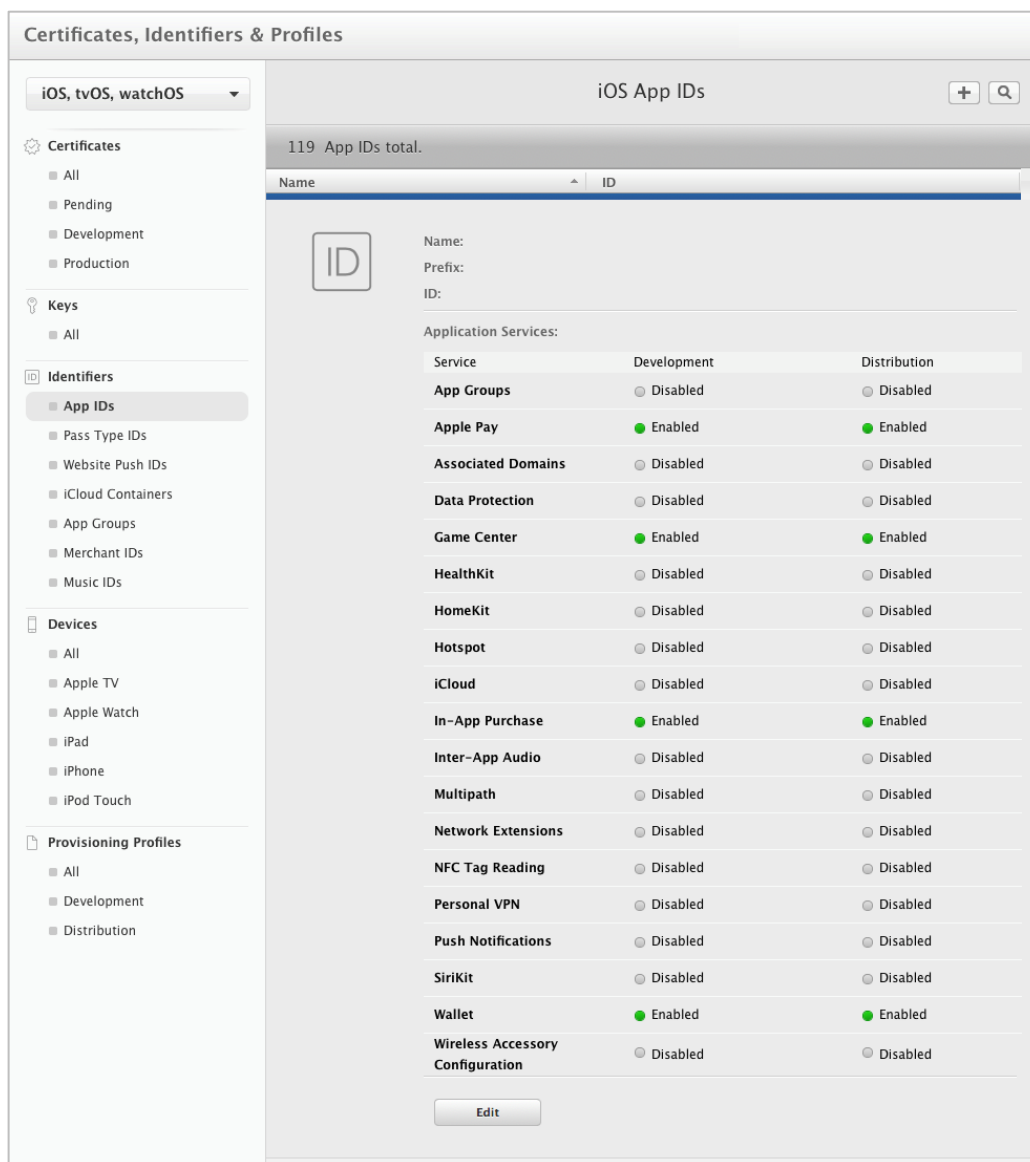


Рисунок 10. Получение сертификата

5. Загрузить сертификат

Certificates, Identifiers & Profiles

iOS, tvOS, watchOS

Certificates

All

Pending

Development

Production

Keys

All

Identifiers

App IDs

Pass Type IDs

Website Push IDs

iCloud Containers

App Groups

Merchant IDs

Music IDs

Devices

All

Apple TV

Apple Watch

iPad

iPhone

iPod Touch

Provisioning Profiles

All

Development

Distribution


iOS Certificates (Production)

+

Q

30 Certificates Total

| Name | Type | Expires |
|---|-----------|--------------|
| merchant.tcsbank.ApplePayTestMerchantId | Apple Pay | Mar 12, 2019 |



Name:

merchant.tcsbank.ApplePayTestMerchantId

Type:

Apple Pay

Expires:

Mar 12, 2019

Created By:

Revoke

Download

| | | |
|--|-------------------------------|--------------|
| merchant.tcsbank.gibdd-fines.ApplePayMerchantId.test | Merchant Identity Certific... | Feb 10, 2020 |
| merchant.tcsbank.gibdd-fines.ApplePayMerchantId.test | Apple Pay | Feb 10, 2020 |
| merchant.tcsbank.insurance.ApplePayMerchantId | Apple Pay | Apr 21, 2019 |
| merchant.tcsbank.insurance.ApplePayMerchantId.test | Apple Pay | Dec 15, 2019 |
| merchant.tcsbank.insurance.ApplePayMerchantId.test | Merchant Identity Certific... | Dec 15, 2019 |
| merchant.tcsbank.mvno.ApplePayMerchantId | Apple Pay | Nov 05, 2019 |

Рисунок 11. Загрузка сертификата

6. Включить Apple Pay в настройках приложения

24

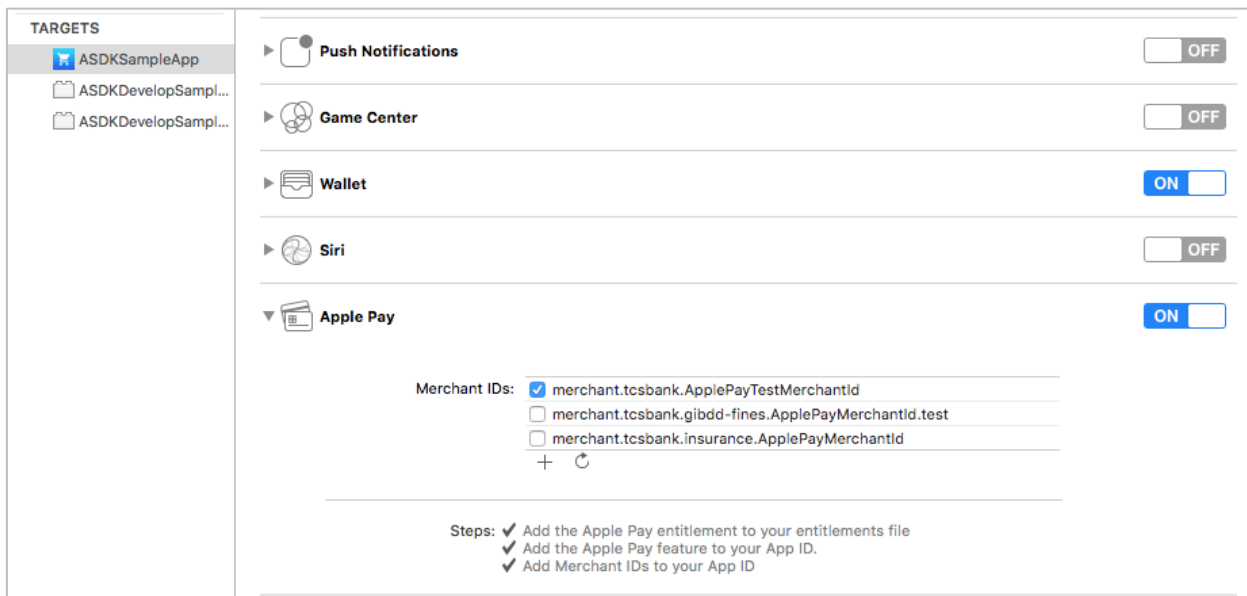


Рисунок 12. Включение оплаты Apple Pay в настройках приложения

Шаг 4. Загрузка подписанного сертификата

Полученный от Apple сертификат apple_pay.cer необходимо загрузить в личном кабинете во вкладке «Дополнительные услуги».

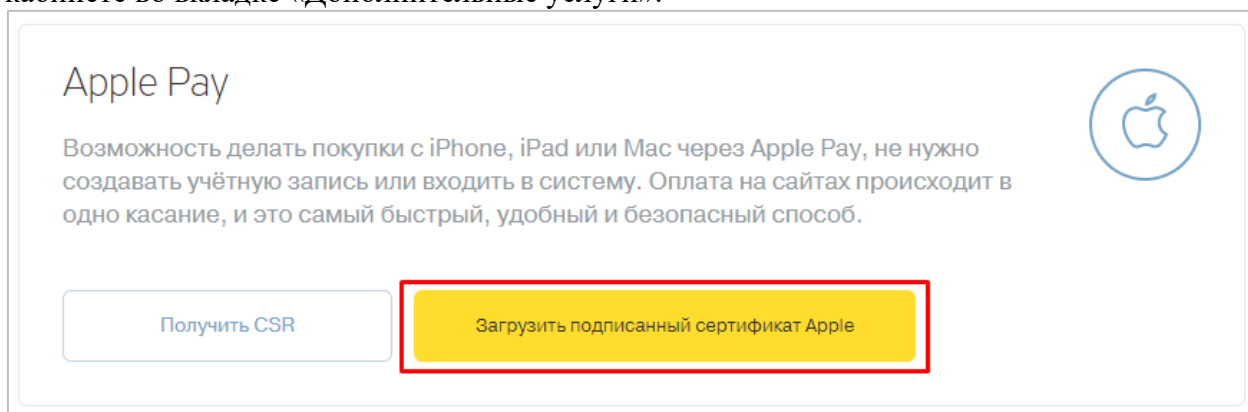


Рисунок 13. Загрузка сертификата

Внимание!

Если после загрузки сертификата вы запросите новый CSR-файл, старый с этого момента будет не действителен, а способ оплаты Apple Pay будет недоступен до загрузки нового подписанного в AppleDeveloper сертификата

Схема взаимодействия

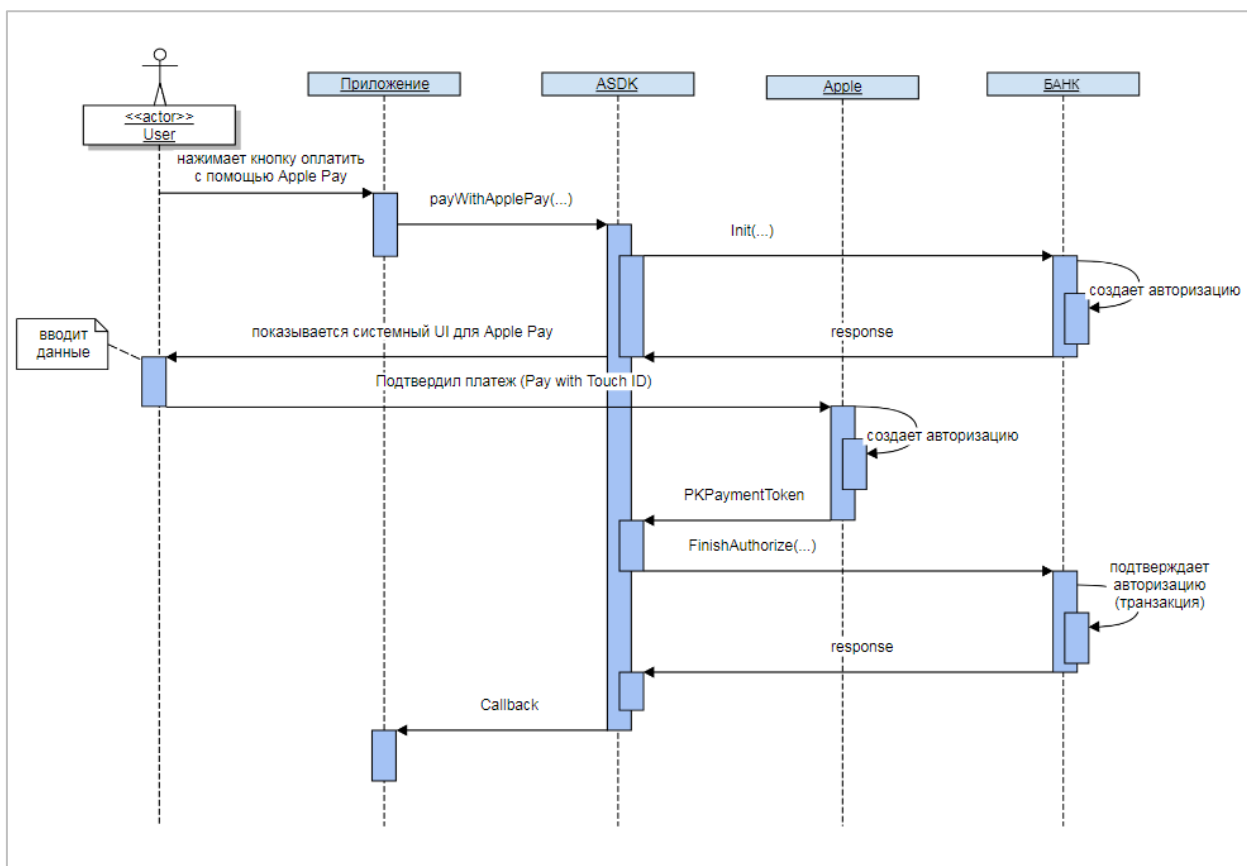


Рисунок 14. Схема взаимодействия

Пользователь нажимает кнопку Оплатить.

1. Вызывается метод `paymentApplePay(paymentData: PaymentInitPaymentData)`
2. Показывается системный UI.
3. Пользователь вводит данные (карта, адрес, контакты) в зависимости от настроек аккаунта Apple Pay.
4. Пользователь подтверждает оплату.
5. Apple Pay создает авторизацию и возвращает **PKPaymentToken**.
6. ASDK вызывает метод `FinishAuthorize` и передает в него закодированное в base64 значение `paymentData` из **PKPaymentToken**.
7. Банк расшифровывает **PKPaymentToken**, подтверждает авторизацию и возвращает ответ.

Результатом `FinishAuthorize` является `Result<PaymentStatusResponse, Error>`

- **Error** - в случае ошибки
- **PaymentStatusResponse** - если платеж прошел успешно

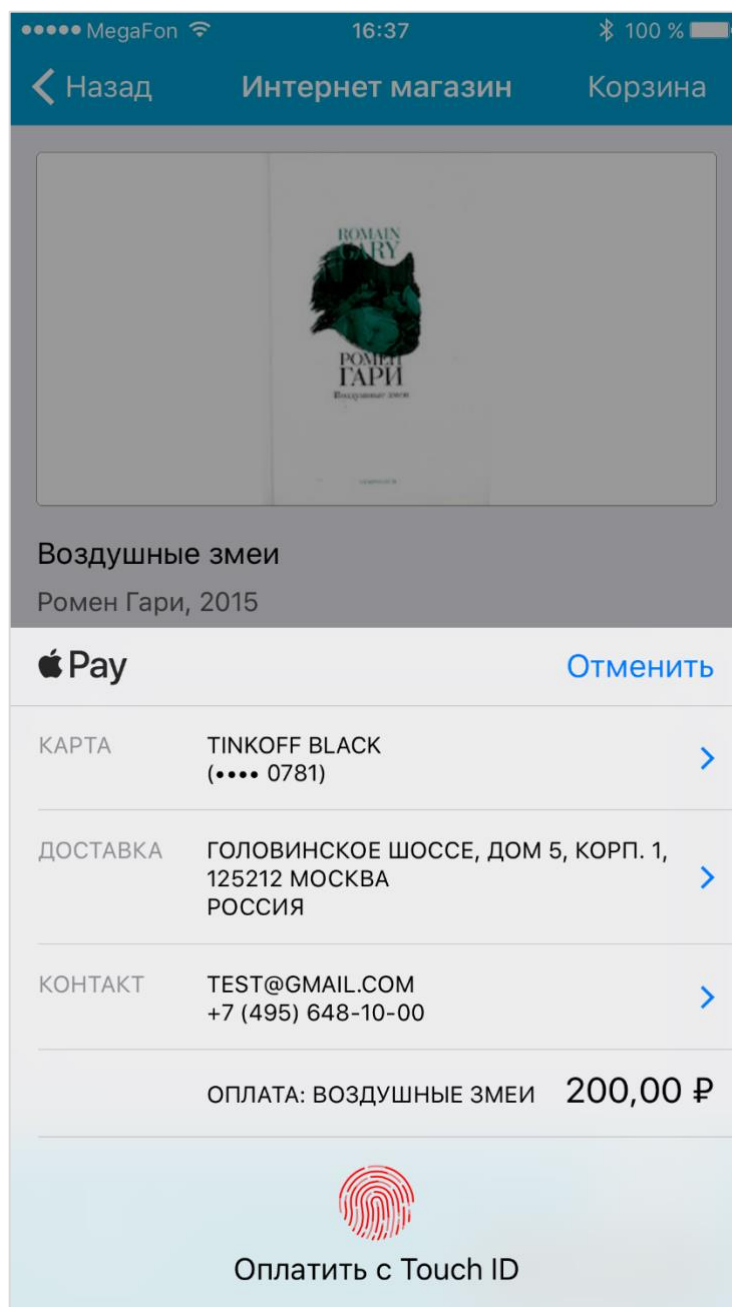


Рисунок 15. Экран оплаты через ApplePay

5. Рекуррентный платеж с привязанной карты

Рекуррентный платеж нужен для дальнейшего списания средств с сохраненной карты без ввода ее реквизитов. Эта возможность используется, например, для осуществления платежей по подписке - регулярные платежи.

Для совершения регулярного платежа нужно вызвать метод

[AcquiringUISDK.presentPaymentView](#) с параметром [parentPatmentId](#), этот параметр нужно брать у карты:

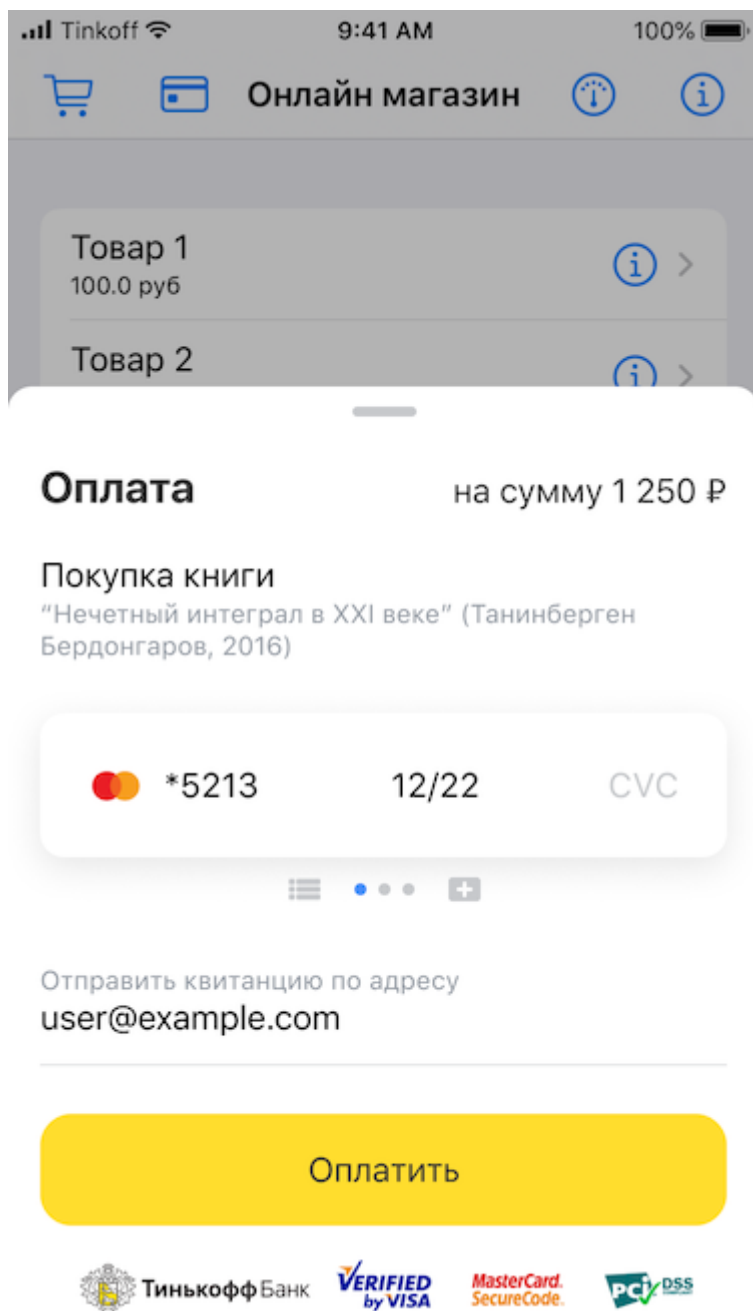
PaymentCard:

- [parentPatmentId](#): [Int64](#) - последний платеж с этой карты, который был зарегистрирован как родительский платеж.

Вместо вызова [FinishAuthorize](#), SDK вызывает [Charge](#).

В ответ на запрос [Charge](#) SDK может получить ответ с [ErrorCode](#) = [104](#), это означает что пользователю необходимо подтвердить платеж через ввод CVC.

В этом случае SDK показывает экран оплаты с реквизитами выбранной для оплаты карты и полем для ввода CVC:



После ввода CVC и нажатия кнопки **ОПЛАТИТЬ** выполняется запрос Init, в DATA, передаются два дополнительных параметра, `recurringType = 12` и `failMapiSessionId = PaymentId` неудачного рекуррента. Продолжаем процесс оплаты, завершаем платеж. Результатом `FinishAuthorize` является `Result<PaymentStatusResponse, Error>`

- `Error` - в случае ошибки
- `PaymentStatusResponse` - если платеж прошел успешно

6. Проведение оплаты с помощью СБП

Для включения в SDK Системы быстрых платежей, необходимо подключить соответствующий тип приема платежей в личном кабинете.

Внимание! Тестирование оплаты через Систему быстрых платежей в настоящее время возможно только на prod окружении и только на prod терминале.

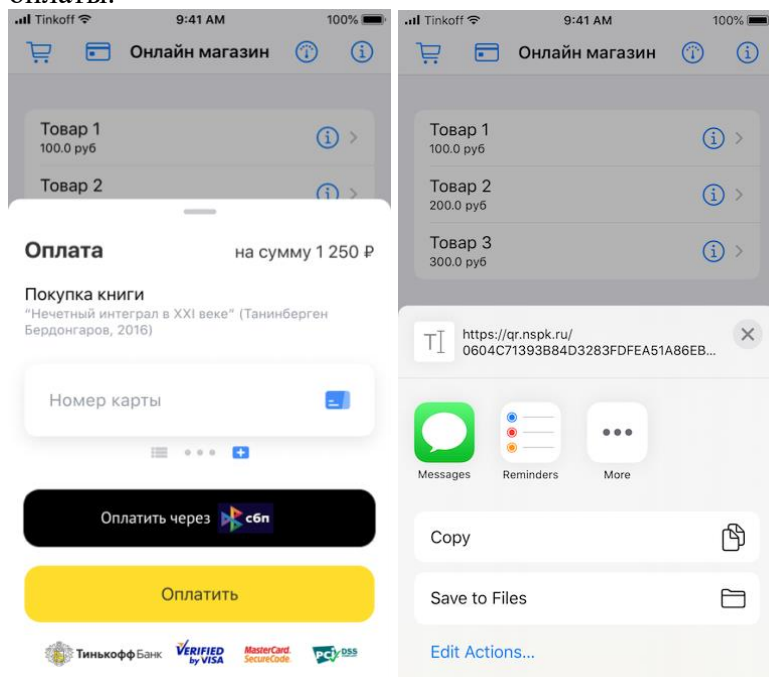
Включение оплаты платежей через СБП в SDK

При конфигурировании параметров экрана оплаты, необходимо передать соответствующий параметр в `AcquiringViewConfiguration.featuresOptions.fpsEnabled = true`. По умолчанию Система быстрых платежей в SDK отключена.

Для оплаты товара карты нужно сформировать информацию о платеже аналогично оплате по реквизитам карты. Далее вызываем метод `AcquiringUISDK.presentPaymentSBP`

Инициализация платежа

Перед началом оплаты открывается форма оплаты товара и запускается метод `Init` результатом которого является регистрация и получение информации о платеже `PaymentInitResponse`, для оплаты, аналогично оплате по реквизитам карты. В случае удачной регистрации платежа вызывается метод `GetQr` который регистрирует QR-код и возвращает информацию о нем в виде специальной ссылки `deepLink`. Далее SDK предлагает выбрать приложение, поддерживающее Систему Быстрых Платежей для оплаты.



Завершение платежа

После оплаты в стороннем/внешнем приложении SDK переходит в режим ожидания, периодически вызывает метод `GetState`, для проверки статуса платежа, ожидая поступления оплаты - изменения статуса платежа.

Результатом `finishAuthorize` является `Result<PaymentStatusResponse, Error>`

- `Error` - в случае ошибки
- `PaymentStatusResponse` - если платеж прошел успешно

7. Прием платежей через Систему Быстрых Платежей (СБП)

Для включения в SDK Системы быстрых платежей, необходимо подключить соответствующий тип приема платежей в личном кабинете.

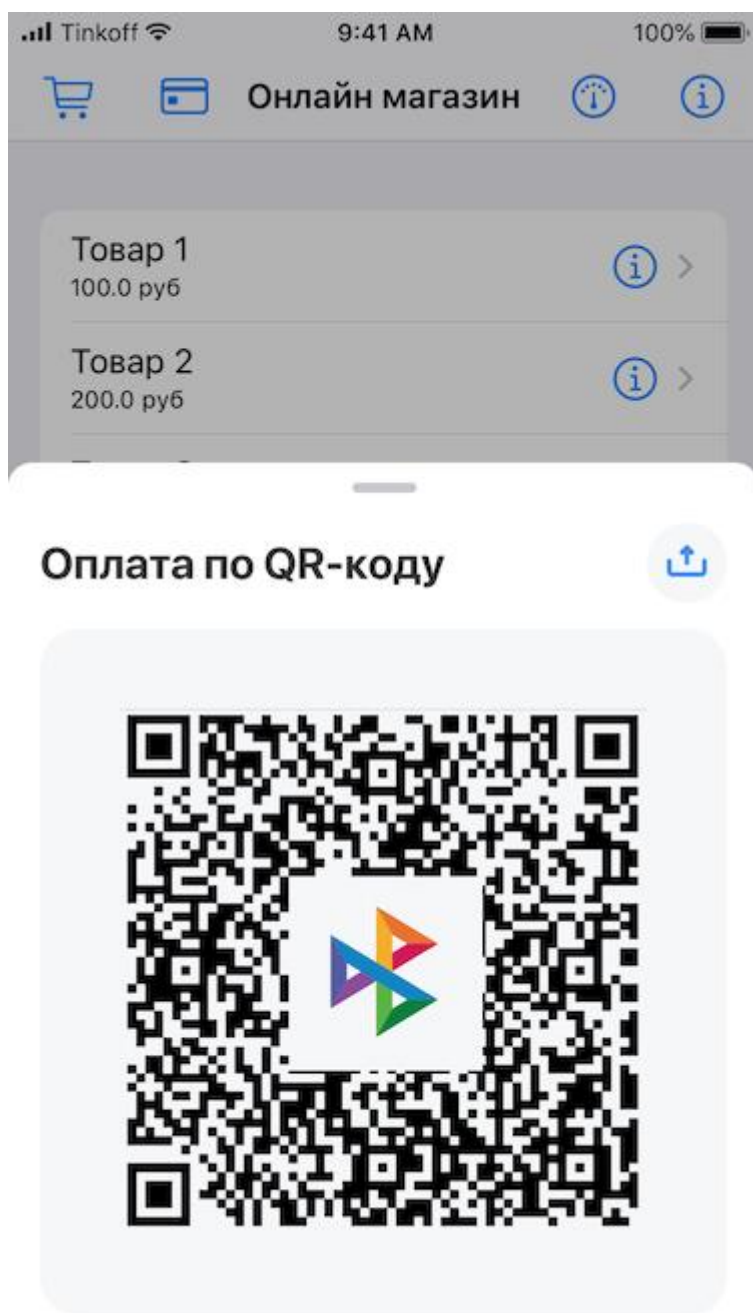
Внимание! Тестирование оплаты через Систему быстрых платежей в настоящее время возможно только на prod окружении и только на prod терминале.

Включение приема платежей через СБП в SDK

При конфигурировании параметров экрана оплаты, необходимо передать соответствующий параметр в `AcquiringViewConfiguration.featuresOptions.fpsEnabled = true`. По умолчанию Система быстрых платежей в SDK отключена.

Сгенерировать QR-код для приема платежей

Метод `AcquiringUISDK.presentPaymentAcceptanceQR` Открывает экран с QR-кодом. Внутри вызывается метод `GetStaticQr` который при первом вызове регистрирует QR-код и возвращает информацию о нем при последующих вызовах возвращает информацию о ранее сгенерированном QR-коде. Перерегистрация статического QR-кода происходит только при смене расчетного счета. Не привязан к конкретному платежу, может быть вызван в любое время без предварительного вызова `Init`.



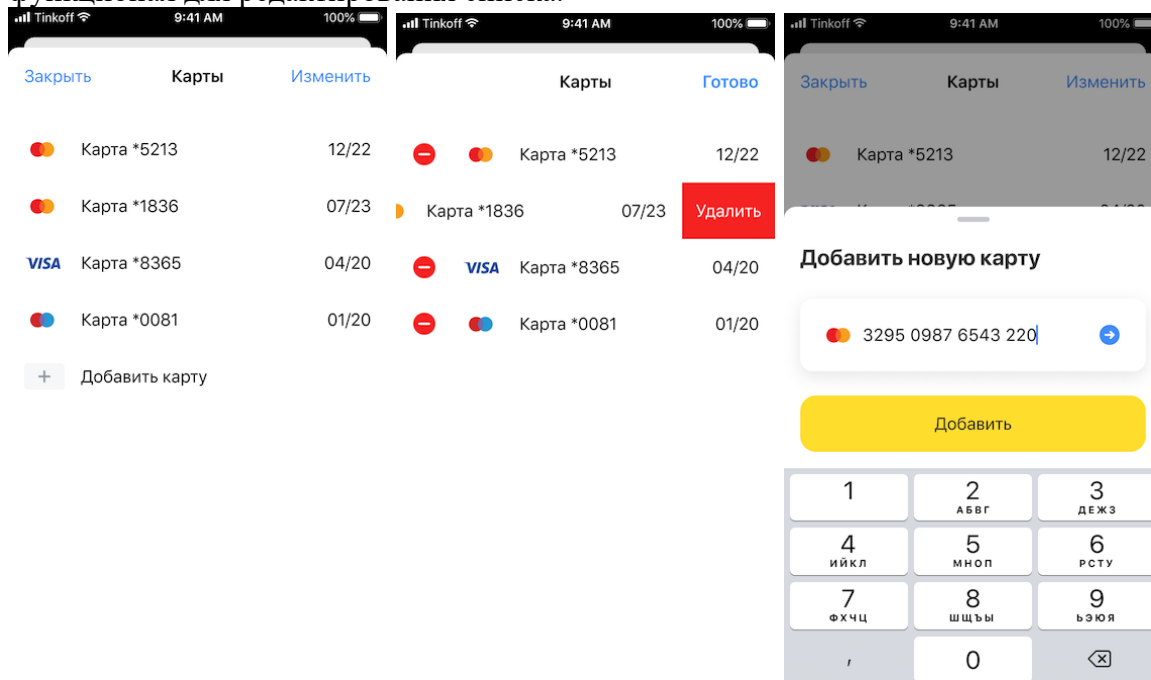
8. Список сохраненных карт

Функционал для работы с привязанными картами реализован в [ASDKCore.CardListDataProvider](#). Методами можно пользоваться в фоне без UI интерфейсов. Доступны методы:

- [update](#) - обновить список карт
- [addCard](#) - добавить карту с реквизитами и типом проверки, результатом будет информация о новой карте [PaymentCard](#) или [Error](#)
- [deactivateCard](#) - де-активировать карту
- [count](#) - количество активных для оплаты карт
- [item](#) [PaymentCard](#) - получить информацию о карте

Для отслеживания изменений в списке карт нужно подписаться на обновления списка карт: [ASDKCore.CardListDataProvider.dataSourceStatusListener](#)

Для отображения списка карт средствами SDK можно использовать [AcquiringUISDK.presentCardList](#). Показывается экран со списком карт, доступен функционал для редактирования списка.



Сохранение новой карты

Для привязки новой карты нужно воспользоваться методом [ASDKCore.CardListDataProvider](#).

Указать полные реквизиты карты и тип проверки, подробнее о типе проверки в таблице:

Таблица 1. Типы проверки

| Тип проверки | Описание |
|---------------------------|---|
| ASDKCardCheckType_NO | Сохранить карту без проверок. Rebill ID для рекуррентных платежей не возвращается |
| ASDKCardCheckType_3DS | При сохранении карты выполнить проверку 3DS и выполнить списание, а затем отмену на 1 р. В этом случае RebillID будет только для 3DS карт. Карты, не поддерживающие 3DS, привязаны не будут |
| ASDKCardCheckType_HOLD | При сохранении сделать списание и затем отмену на 1 руб. RebillID для рекуррентных платежей возвращается в ответе |
| ASDKCardCheckType_3DSHOLD | При привязке карты выполняем проверку, поддерживает карта 3DS или нет. Если карта поддерживает 3DS, далее выполняем списание и затем отмену на 1 руб |

Схема работы

Сценарий

1. Инициализируем привязку карты для клиента указываем параметры:

- *customerKey*;
- *checkType*;

Пользователь вводит номер карты и нажимает кнопку добавить.

2. Выполняется Метод AddCard для получения RequestKey.
3. Выполняется Метод AttachCard для привязки карты.
4. Если в запросе Метод AddCard параметр CheckType=:
 - a) NO – сохранить карту без проверок. Rebill ID для рекуррентных платежей не возвращается.
 - b) HOLD – при сохранении сделать списание и затем отмену на 1 руб. RebillID для рекуррентных платежей возвращается в ответе.
 - c) 3DS – при сохранении карты выполнить проверку 3DS. Если карта поддерживает 3DS, выполняется списание и последующая отмена на 1 руб. В этом случае RebillID будет только для 3DS карт.
Карты, не поддерживающие 3DS, привязаны не будут.
 - d) 3DSHOLD – при привязке карты выполнить проверку поддержки картой 3DS. Если карта поддерживает 3DS, выполняется списание и последующая отмена на 1 руб. Если карта не поддерживает 3DS, выполняется списание и последующая отмена на произвольную сумму от 100 до 199 копеек. Клиент будет перенаправлен на страницу для ввода списанной суммы, где должен корректно указать случайную сумму. В случае успешного подтверждения случайной суммы карта будет привязана и возвращен Rebill ID.

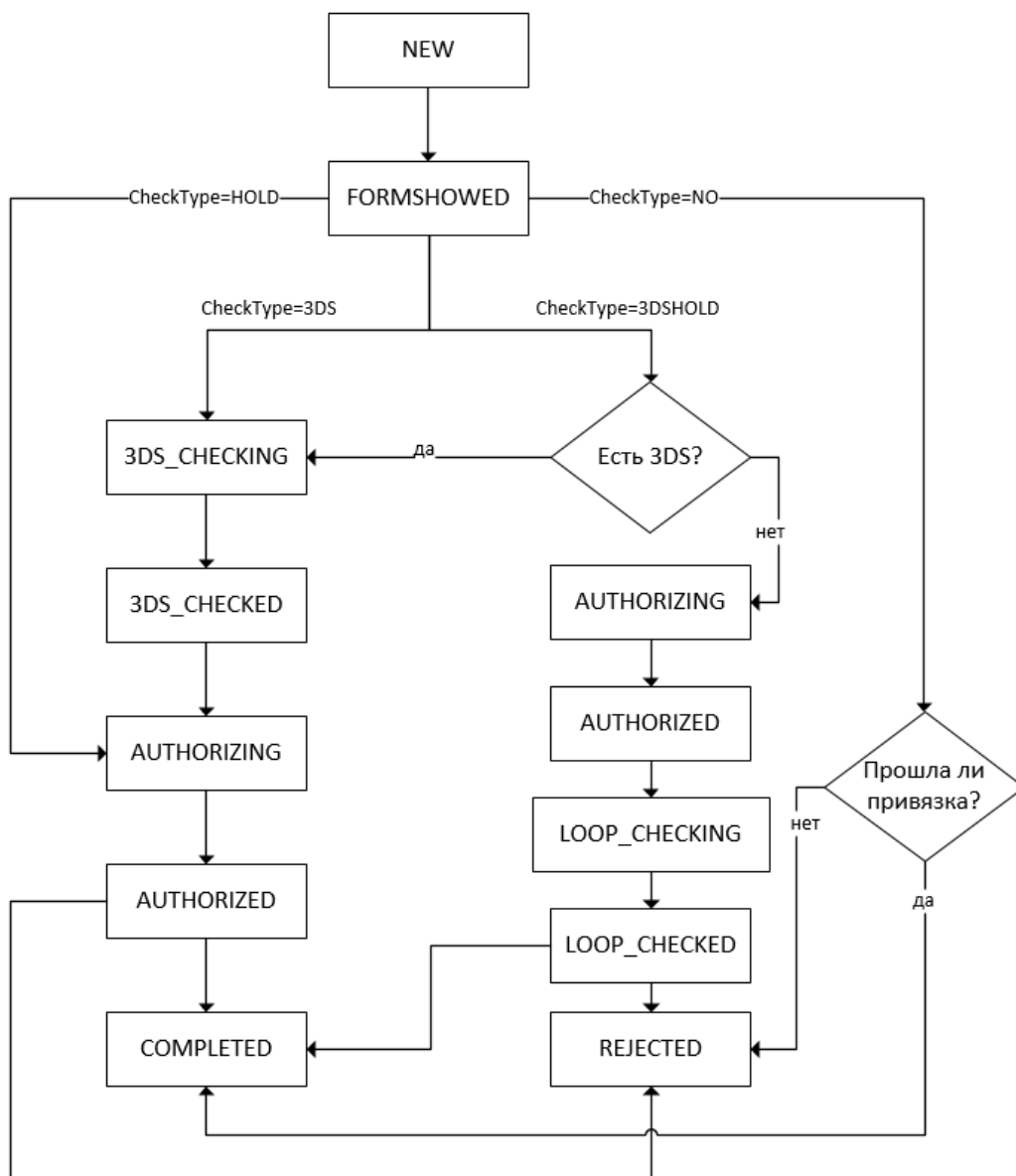


Рисунок 16. Статусная схема привязки карт

Описание статусов:

- NEW — новая сессия;
- FORMSHOWED — показ формы привязки карты;
- 3DS_CHECKING — отправка пользователя на проверку 3DS
- 3DS_CHECKED — пользователь успешно прошел проверку 3DS;
- LOOP_CHECKING — пользователь отправлен на проверку блокирования случайной суммы;
- LOOP_CHECKED — пользователь успешно прошел проверку блокирования случайной суммы;
- AUTHORIZING — блокировка 1 рубля;

- AUTHORIZED — успешно заблокировали и разблокировали 1 рубль;
- COMPLETED — привязка успешно завершена;
- REJECTED — привязка отклонена.

9. Тестовое приложение

Структура

SDK состоит из следующих модулей:

- ASDKCore
- ASDKUI
- Sample

ASDKCore

Является базовым модулем для работы с Tinkoff Acquiring API.

Он содержит модель данных и позволяет осуществлять запросы к API с разбором ответов из JSON в имеющуюся модель. Запросы осуществляются синхронно. Модуль можно использовать отдельно от остальной SDK для реализации десктопных или веб-приложений.

Класс позволяет конфигурировать SDK и осуществлять взаимодействие с Тинькофф Эквайринг API. Методы, осуществляющие обращение к API, возвращают результат в случае успешного выполнения запроса или бросают исключение `AcquiringSdkException`.

Основной класс модуля - `AcquiringSdk` - предоставляет фасад для взаимодействия с Tinkoff Acquiring API. Для работы необходимы ключи и пароль продавца (см. Для подключения добавьте в файл Podfile зависимости:

```
pod 'CardIO'
pod 'ASDKCore'
pod 'ASDKUI'
```

Если вы не используете Cocoa Pods, необходимо добавить `ASDKUI.xcodeproj` в проект.

Подготовка к работе).

ASDKUI

Содержит интерфейс необходимый для приема платежей через мобильное приложение.

Основной класс - ***AcquiringUISDK*** - экран с формой оплаты, который позволяет:

- просматривать информацию о платеже;
- вводить или сканировать реквизиты карты для оплаты;
- проходить 3-DS подтверждение;
- управлять списком ранее сохраненных карт.

Все строки, представленные в интерфейсе, имеют две локализации: на русском и на английском в зависимости от языка, выбранного в системе.

Методы для оплаты:

- `presentCardList` показать список сохраненных карт
- `presentPaymentView(paymentData: PaymentInitPaymentData)` показать форму оплаты для выбора источника оплаты (сохраненная карта или реквизиты новой) и оплатить
- `presentPaymentView(paymentData: PaymentInitPaymentData, parentPatmentId: Int64)` оплатить рекуррентный платеж
- `presentPaymentSBP(paymentData: PaymentInitPaymentData)` оплатить используя Систему Быстрых Платежей
- `presentPaymentAcceptanceQR` сгенерировать и показать QR-код для приема платежей
- `paymentApplePay(paymentData: PaymentInitPaymentData)` оплатить используя ApplePay

Sample

Содержит пример интеграции Tinkoff Acquiring SDK в мобильное приложение по продаже товаров. Основные классы и файлы:

`ASDKStageTestData` содержат **Terminal key**, **Пароль**, **Public key**

Модули SDK подключены как зависимости CocoaPods

Приложение состоит из следующих экранов

- **Основной экран** – экран со списком книг в продаже;
- **Детали товара** – экран с подробной информацией о товаре и кнопками оплаты;
- **Корзина** – экран с набранными товарами;
- **Настройки** – Настройки для работы с SDK;
- **О программе** – информационный экран: лого, версия SDK, и проч.

API Методы для работы с сервером

Метод Init

Описание: Инициализирует новый платеж.

URL: <https://securepay.tinkoff.ru/v2/Init>

Метод: GET или POST

Таблица 2. Параметры запроса

| Параметр | Тип | Обязательный | Описание |
|-------------|--------|--------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| Amount | Number | Да | Сумма в копейках |
| OrderId | String | Да | Номер заказа в системе Продавца |
| IP | String | Нет | IP-адрес клиента |
| Description | String | Нет | Краткое описание |

| Параметр | Тип | Обязательный | Описание |
|-----------------|----------|--------------|---|
| Currency | Number | Нет | Код валюты ISO 4217 (например, 643). Если передан Currency, и он разрешен для Продавца, то транзакция будет инициирована в переданной валюте. Иначе будет использована валюта по умолчанию для данного терминала |
| Token | String | Да | Подпись запроса |
| PayForm | String | Нет | Название шаблона формы оплаты продавца |
| CustomerKey | String | Нет | Идентификатор покупателя в системе Продавца. Если передается и Банком разрешена автоматическая привязка карт к терминалу, то для данного покупателя будет осуществлена привязка карты. Тогда в нотификации на AUTHORIZED будет передан параметр CardId , подробнее см. метод <i>GetGardList</i> |
| Language | String | Нет | Язык платёжной формы. ru - форма оплаты на русском языке; en - форма оплаты на английском языке. По умолчанию (если параметр не передан) - форма оплаты на русском языке. |
| Recurrent | String | Нет | Если передается и установлен в Y, то регистрирует платеж как рекуррентный. В этом случае после оплаты в нотификации на AUTHORIZED будет передан параметр RebillId для использования в методе <i>Charge</i> |
| RedirectDueDate | Datetime | Нет | Срок жизни ссылки. В случае, если текущая дата превышает дату переданную в данном параметре, ссылка для оплаты становится недоступной и платёж выполнить нельзя. Формат даты: YYYY-MM-DDTHH24:MI:SS+GMT Пример даты: 2016-08-31T12:28:00+03:00 |

| Параметр | Тип | Обязательный | Описание |
|----------|--------|--------------|---|
| DATA* | Object | Нет | JSON объект содержащий дополнительные параметры в виде «ключ»:»значение». Данные параметры будут переданы на страницу оплаты (в случае ее кастомизации). Максимальная длина для каждого передаваемого параметра: Ключ – 20 знаков, Значение – 100 знаков. Максимальное количество пар «ключ-значение» не может превышать 20(*) |
| Receipt | Object | Нет | JSON объект с данными чека |

(*) В случае если у терминала включена опция привязки покупателя после успешной оплаты и передается параметр **CustomerKey**, то в передаваемых параметрах **DATA** могут присутствовать параметры команды **AddCustomer**. Если они присутствуют, они автоматически привязываются к покупателю.

Например, если указать:

```
"DATA":{"Phone":"+71234567890", "Email":"a@test.com"}
```

к покупателю автоматически будут привязаны данные Email и телефон, и они будут возвращаться при вызове метода *GetCustomer*.

Таблица 3. Структура объекта Receipt

| Наименование | Тип | Обязательность | Описание |
|--------------|---------------------|----------------|---|
| Items | Массив объектов | Да | Массив, содержащий в себе информацию о товарах |
| Email | String | Да | Электронная почта |
| Phone | String | Нет | Телефон |
| Taxation | Перечисление (Enum) | Да | Система налогообложения. Перечисление со значениями: - «osn» – общая СН; - «usn_income» – упрощенная СН (доходы); - «usn_income_outcome» – упрощенная СН (доходы минус расходы); |

| | | | |
|--|--|--|---|
| | | | <ul style="list-style-type: none"> - «envd» – единый налог на вмененный доход; - «esn» – единый сельскохозяйственный налог; - «patent» – патентная СН. |
|--|--|--|---|

Таблица 4. Структура объекта Items

| Наименование | Тип | Обязательность | Описание |
|--------------|---------------------|----------------|---|
| Name | String | Да | Наименование товара. Максимальная длина строки – 128 символов |
| Price | Number | Да | Цена в копейках Целочисленное значение не более 10 знаков |
| Quantity | Number | Да | Количество/вес: - целая часть не более 8 знаков; - дробная часть не более 3 знаков |
| Amount | Number | Да | Сумма в копейках. Целочисленное значение не более 10 знаков |
| Tax | Перечисление (Enum) | Да | Ставка налога. Перечисление со значениями: - «none» – без НДС; - «vat0» – НДС по ставке 0%; - «vat10» – НДС чека по ставке 10%; - «vat18» – НДС чека по ставке 18%; - «vat110» – НДС чека по расчетной ставке 10/110; - «vat118» – НДС чека по расчетной ставке 18/118 |
| Ean13 | String | Нет | Штрих-код |
| ShopCode | String | Нет | Код магазина. Для параметра |

| Наименование | Тип | Обязательность | Описание |
|--------------|-----|----------------|---|
| | | | ShopCode необходимо использовать значение параметра Submerchant_ID, полученного в ответ при регистрации магазинов через xml. Если xml не используется, передавать поле не |

Пример запроса:

```
{
  "TerminalKey": "TestB",
  "Amount": "140000",
  "OrderId": "21050",
  "Description": "Подарочная карта на 1000 рублей",
  "Token": "2ED30E046136931431B5251B7C9A1EAC68DAB082203BD42676BA14A851359DF4",
  "DATA": {
    "Phone": "+71234567890",
    "Email": "a@test.com"
  },
  "Receipt": {
    "Email": "a@test.ru",
    "Phone": "+79031234567",
    "Taxation": "osn",
    "Items": [
      {
        "Name": "Наименование товара 1",
        "Price": 10000,
        "Quantity": 1.00,
        "Amount": 10000,
        "Tax": "vat10",
        "Ean13": "0123456789",
        "ShopCode": "12345"
      },
      {
        "Name": "Наименование товара 2",
        "Price": 20000,
        "Quantity": 2.00,
        "Amount": 40000,
        "Tax": "vat18"
      },
      {
        "Name": "Наименование товара 3",
        "Price": 30000,
        "Quantity": 3.00,
        "Amount": 90000,
        "Tax": "vat10"
      }
    ]
  }
}
```

```

    }
  ]
}
}

```

Таблица 5. Параметры ответа

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|--|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| Amount | Number | Да | Сумма в копейках |
| OrderId | String | Да | Номер заказа в системе Продавца |
| Success | bool | Да | Успешность операции |
| Status | String | Да | Статус транзакции |
| PaymentId | Number | Да | Уникальный идентификатор транзакции в системе Банка |
| ErrorCode | String | Да | Код ошибки, «0» - если успешно |
| PaymentURL | String | Нет | Ссылка на страницу оплаты. Не передается, если ввод данных карты осуществляется на стороне Продавца. <i>Доступна в течении 24 часов по умолчанию.</i> |
| Message | String | Нет | Краткое описание ошибки |
| Details | String | Нет | Подробное описание ошибки |

Пример ответа:

```

{"Success":true,"ErrorCode":"0","TerminalKey":"TestB","Status":"NEW","PaymentId":"13660",
"OrderId":"21050","Amount":"100000",
"PaymentURL":"https://rest-api-test.tcsbank.ru/rest/Authorize/1b63d14a-4208-44a8-a288-ad1b04008e51"}

```

Статус платежа:

при успешном сценарии: **NEW**;

при неуспешном: **REJECTED**.

Метод FinishAuthorize

Описание: Подтверждает инициированный платеж передачей карточных данных.

При использовании одностадийного проведения осуществляет списание денежных средств с карты покупателя.

При двухстадийном проведении осуществляет блокировку указанной суммы на карте покупателя.

URL: <https://securepay.tinkoff.ru/v2/FinishAuthorize>

Метод: POST

Таблица 6. Параметры запроса

| Наименование | Тип | Обязательный? | Описание |
|--------------|---------|---------------|--|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| PaymentId | Number | Да | Уникальный идентификатор транзакции в системе Банка, полученный в ответе на вызов метода Init |
| CardData(*) | String | Да | Зашифрованные данные карты в формате: "PAN=%pan%;ExpDate=%month%%year%;CVV=%secure_code%" при оплате по полным реквизитам; "CardId=%id%;CVV=%secure_code%" при оплате с сохраненной карты |
| DATA | Object | Нет | JSON объект содержащий дополнительные параметры в виде «ключ»:»значение». Данные параметры будут переданы на страницу оплаты (в случае ее кастомизации). Максимальная длина для каждого передаваемого параметра: Ключ – 20 знаков, Значение – 100 знаков. Максимальное количество пар «ключ-значение» не может превышать 20 |
| IP | String | Нет | IP-адрес клиента |
| Phone | String | Нет | Телефон клиента |
| SendEmail | boolean | Нет | true – отправлять клиенту информацию на почту об оплате; false – не отправлять |
| InfoEmail | String | Нет | Email для отправки информации об |

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|-----------------|
| | | | оплате |
| Token | String | Да | Подпись запроса |

Продавец собирает поле CardData в виде списка «ключ=значение» с разделителем «;» и зашифровывает его выданным при регистрации терминала открытым ключом (X509 RSA 2048).

Таблица 7. Параметры CardData

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| PAN | Number | Да | Номер карты |
| ExpDate | Number | Да | Месяц и год срока действия карты в формате ММYY |
| CardHolder | String | Нет | Имя и фамилия держателя карты (как на карте) |
| CVV | String | Да | Код защиты (с обратной стороны карты) |

Пример значения элемента формы CardData:

“PAN=43000000000000777;ExpDate=0519;CardHolder=IVAN PETROV;CVV=111”

Пример запроса:

```
{
  "TerminalKey": "TestB"
  "PaymentId": "10063"
  "CardData": "b3tSIUYwsf3Erdv5ReB7WpWK3/NBWLiwDiSLjQG0cBxA0Mgs7ALd7edi0RbV
lORsyGZEUJSIRynQ9zLMyHYzWP3z2sQYGA vzOqufoVPe2AozhW3pZV+dN5s7oGcpXd39
NDC0Ma/Zw6oa3dJR0Zh8QYjv/sG0zUllMjXl5aHgTpxk37q6OxUakxuG7euhvSN71JqxHsNE
uoJELAQlq7U+3tuh9AjTuiBpmEH99maK9e7gnVXgZd1Nk8vachs97xj9cL/023qYMk7CMjld
BfG4VOsYVqcHsKfbbJJ8CZXIJgmXhCYns1hmRD/kf3OhEZr038LghC7Iio0yxHYMhZyJoQ=
="
  "Token": "871199b37f207f0c4f721a37cdcc71dfcea880b4a4b85e3cf852c5dc1e99a8d6"
}
```

Формат ответа: JSON

Таблица 8. Параметры ответа

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|--|
| OrderId | String | Да | Номер заказа в системе Продавца |
| Amount | Number | Да | Сумма в копейках |
| RebillId | String | Нет | Идентификатор рекуррентного платежа |
| CardId | String | Нет | Идентификатор карты в системе Банка. Передается только для сохраненной карты |
| Success | bool | Да | Успешность операции (true/false) |
| Status(*) | String | Да | Статус транзакции |
| PaymentId | Number | Да | Уникальный идентификатор транзакции в системе Банка |
| ErrorCode | String | Да | Код ошибки, «0» - если успешно |
| Message | String | Нет | Краткое описание ошибки |
| Details | String | Нет | Подробное описание ошибки |

Пример ответа:

```
{ "Success":true,"ErrorCode":"0","TerminalKey":"TestB","Status":"CONFIRMED","PaymentId":
:"10063","OrderId":"21050","Amount":100000 }
```

(*)Статус платежа:

при успешном сценарии и одностадийном проведении платежа: **CONFIRMED**

при успешном сценарии и двухстадийном проведении платежа: **AUTHORIZED**

при неуспешном: **REJECTED**

при необходимости прохождения проверки 3-D Secure: **3DS_CHECKING**

Метод Charge

Описание: Осуществляет рекуррентный (повторный) платеж — безакцептное списание денежных средств со счета банковской карты Покупателя. Для возможности его использования Покупатель должен совершить хотя бы один платеж в пользу Продавца, который должен быть указан как рекуррентный (см. параметр **Recurrent** в методе *Init*), фактически являющийся первичным. По завершении оплаты такого платежа в нотификации на **AUTHORIZED** будет передан параметр **RebillId**. В дальнейшем при совершении рекуррентного платежа Продавец должен инициировать его, вызвав метод *Init*, а затем, не осуществляя переадресации на PaymentURL, вызвать метод *Charge* для оплаты по тем же самым реквизитам передав

параметр **RebillId**, полученный при совершении первичного платежа. Независимо от установленного типа проведения платежа, метод **Charge** всегда работает по типу одностадийного проведения. Это значит, что во время выполнения метода **Charge** на Notification URL будет отправлен синхронный запрос (подробнее см. Нотификация продавца об операциях), на который требуется корректный ответ.

Другими словами, для использования рекуррентных платежей необходима следующая последовательность действий:

1. Совершить родительский платеж путем вызова **Init** с указанием дополнительного параметра **Recurrent=Y**.
2. Переадресовать Покупателя на **PaymentUrl**.
3. После оплаты покупателем в нотификации на **AUTHORIZED** будет передан параметр **RebillId**, который необходимо сохранить.
4. Спустя некоторое время для совершения рекуррентного платежа необходимо вызвать метод **Init** со стандартным набором параметров (параметр **Recurrent** здесь не нужен).
5. Получить в ответ на **Init** параметр **PaymentId**, при этом переадресацию пользователя на **PaymentUrl** производить не надо.
6. Вызвать метод **Charge** с параметром **RebillId** полученным в п.3 и параметром **PaymentId** полученным в п.5.

URL: <https://securepay.tinkoff.ru/v2/Charge>

Метод: POST

Таблица 9. Параметры запроса

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| PaymentId | Number | Да | Уникальный идентификатор транзакции в системе Банка, полученный в ответе на вызов метода Init |
| IP | String | Нет | IP-адрес клиента |
| RebillId | Number | Да | Идентификатор рекуррентного платежа (см. параметр Recurrent в методе Init) |
| Token | String | Да | Подпись запроса. |
| SendEmail | bool | Нет | true – если покупатель хочет получать уведомления на почту |
| InfoEmail | String | Нет | Адрес почты покупателя |

Пример запроса:

```
{
  "TerminalKey": "TestB",
```



```
"PaymentId":"10063",
"RebillId":"145919",
"Token":"871199b37f207f0c4f721a37cdcc71dfcea880b4a4b85e3cf852c5dc1e99a8d6"
}
```

Формат ответа: JSON

Таблица 10. Параметры ответа

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| OrderId | String | Да | Номер заказа в системе Продавца |
| Success | bool | Да | Успешность операции (true/false) |
| Status | String | Да | Статус транзакции |
| PaymentId | Number | Да | Уникальный идентификатор транзакции в системе Банка |
| ErrorCode | String | Да | Код ошибки, «0» - если успешно |
| Amount | Number | Да | Сумма списания в копейках |
| Message | String | Нет | Краткое описание ошибки |
| Details | String | Нет | Подробное описание ошибки |
| CardId | String | Нет | Идентификатор карты в системе Банка |
| RebillId | String | Нет | Идентификатор рекуррентного платежа |

Пример ответа:

```
{ "Success":true,"ErrorCode":"0","TerminalKey":"TestB","Status":"CONFIRMED" }
```

Метод GetState

Описание: Возвращает статус платежа

URL: <https://securepay.tinkoff.ru/v2/GetState>

Метод: POST

Таблица 11. Параметры запроса

| Наименование | Тип | Обязательный? | Описание |
|--------------|-----|---------------|----------|
|--------------|-----|---------------|----------|

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| PaymentId | Number | Да | Уникальный идентификатор транзакции в системе Банка |
| IP | String | Нет | IP-адрес клиента |
| Token | String | Да | Подпись запроса |

Пример запроса:

```
{
  "TerminalKey": "TestB",
  "PaymentId": "10063",
  "Token": "871199b37f207f0c4f721a37cdcc71dfcea880b4a4b85e3cf852c5dc1e99a8d6"
}
```

Формат ответа: JSON

Таблица 12. Параметры ответа

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| OrderId | String | Да | Номер заказа в системе Продавца |
| Success | bool | Да | Успешность операции (true/false) |
| Status | String | Да | Статус транзакции |
| Amount | Number | Нет | Сумма отмены в копейках |
| PaymentId | Number | Да | Уникальный идентификатор транзакции в системе Банка |
| ErrorCode | String | Да | Код ошибки, «0» - если успешно |
| Message | String | Нет | Краткое описание ошибки |
| Details | String | Нет | Подробное описание ошибки |

Пример ответа:

```
{ "TerminalKey": "TestB", "OrderId": "21057", "Success": true, "Status": "NEW", "PaymentId": "10063", "ErrorCode": "0" }
```

Таблица 13. Возможные статусы транзакции

| Статус | Промежуточный? | Значение |
|---------------------|----------------|---|
| AUTHORIZED | Нет | Средства заблокированы, но не списаны |
| 3DS_CHECKING | Нет | Покупатель начал аутентификацию по протоколу 3-D Secure |
| CONFIRMED | Нет | Денежные средства списаны |

Метод AddCard

Описание: Иницирует привязку карты к покупателю.

В случае успешной привязки переадресует клиента на Success Add Card URL, в противном случае на Fail Add Card URL. Можно использовать форму банка, возможно заменить на кастомную форму.

URL: <https://securepay.tinkoff.ru/v2/AddCard>

Метод: POST

Таблица 14. Параметры запроса

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| CustomerKey | String | Да | Идентификатор покупателя в системе Продавца |
| CheckType | String | Нет | Возможные значения: NO – сохранить карту без проверок. Rebill ID для рекуррентных платежей не возвращается. HOLD – при сохранении сделать списание и затем отмену на 1 руб. RebillID для рекуррентных платежей возвращается в ответе. 3DS – при сохранении карты выполнить проверку 3DS. Если карта поддерживает 3DS, выполняется списание и последующая отмена на 1 руб. В этом случае RebillID будет только для 3DS карт. Карты, не поддерживающие 3DS, привязаны |

| | | | |
|-------------|--------|-----|--|
| | | | <p>не будут.</p> <p>3DSHOLD – при привязке карты выполнить проверку поддержки картой 3DS. Если карта поддерживает 3DS, выполняется списание и последующая отмена на 1 руб.</p> <p>Если карта не поддерживает 3DS, выполняется списание и последующая отмена на произвольную сумму от 100 до 199 копеек. Клиент будет перенаправлен на страницу для ввода списанной суммы, где должен корректно указать случайную сумму. В случае успешного подтверждения случайной суммы карта будет привязана и возвращен Rebill ID</p> |
| Description | String | Нет | Описание/название карты |
| PayForm | String | Нет | Название шаблона формы привязки |
| IP | String | Нет | IP-адрес запроса |
| Token | String | Да | Подпись запроса |

Пример запроса:

```
{
  "TerminalKey": "1201253242594",
  "CustomerKey": "Test-112",
  "CheckType": "3DS",
  "Token": "2ED30E046136931431B5251B7C9A1EAC68DAB082203BD42676BA14A851359D
F4"
}
```

Формат ответа: JSON
Таблица 15. Параметры ответа

| Имя | Тип | Обязательный? | Описание |
|-------------|--------|---------------|--|
| TerminalKey | String | Да | Платежный ключ, выдается Продавцу при заведении терминала |
| CustomerKey | String | Да | Идентификатор покупателя в системе Продавца |
| RequestKey | String | Да | Идентификатор запроса на привязку карты |
| PaymentURL | String | Нет | Ссылка на страницу привязки карты. На данную страницу необходимо переадресовать клиента для привязки карты |
| Success | bool | Да | Успешность операции |
| ErrorCode | String | Да | Код ошибки, «0» если успешно |
| Message | String | Нет | Краткое описание ошибки |
| Details | String | Нет | Подробное описание ошибки |

Пример ответа:

```
{ "Success":true,"ErrorCode":"0","TerminalKey":"TestB","CustomerKey":"Customer1",
"RebillId":"123456", "PaymentURL":"https://securepay.tinkoff.ru/e2c/f36d8e7f-4bc6-4250-9f64-7fe986d3dc62" }
```

Метод AttachCard

Описание: Завершает привязку карты к покупателю. Метод вызывается автоматически после метода AddCard.

В случае успешной привязки переадресует клиента на Success Add Card URL в противном случае на Fail Add Card URL.

URL: <https://securepay.tinkoff.ru/v2/AttachCard>

Метод: POST

Таблица 16. Параметры запроса

| Имя | Тип | Обязательный? | Описание |
|-------------|--------|---------------|---|
| TerminalKey | String | Да | Платежный ключ, выдается Продавцу при заведении терминала |
| RequestKey | String | Да | Идентификатор запроса на привязку карты |
| CardData | String | Да | Зашифрованные данные карты в формате: "PAN=%pan%;ExpDate=%month%%year%;CVV=%secure_code%" |
| DATA | Object | Нет | Ключ = значение дополнительных параметров через “ ”, например, Email = a@test.ru Phone = +71234567890. Если ключи или значения содержат в себе спец символы, то получившееся значение должно быть закодировано функцией urlencode. Максимальная длина для каждого передаваемого параметра: Ключ – 20 знаков, Значение – 100 знаков. Максимальное количество пар «ключ-значение» не может превышать 20 |
| Token | String | Да | Подпись запроса |

Пример запроса:

```
{
  "TerminalKey":"testRegress",
  "CardData":"U5jDbwqOVx+2vDApXe/rfACMt+rfWXzPdJ8ZXxNFVliZaLZrOW72bGe9cKZdI
DnekW0nqm88YxRD4jyfa5Ru0kY5cQValU+juS1u1zpamSDtaGFeb8sRZfhj72yGw+io+qHGS
BeorcfgoKStyKGuBPWfG4d0PLHuyBE6QgZyIAM1XfdmNIV0UAxOnkTGDsskLpIt3AWhw
2e8KOar0vwbGCTDjznDB1/DLgOW014Aj/bXyLJoG1BkOrPBm9JURs+f+uyFae0hkRicNKNg
XoN5pJTSQxOEauOi6ylsVJB3WK5MNYXtj6x4GlxcMtk/LD9kvHcjTeojcAlDzRZ87GdWeY
8wgg==",
  "RequestKey":"13021e10-a3ed-4f14-bcd1-823b5ac37390",
  "Token":"7241ac8307f349afb7bb9dda760717721bbb45950b97c67289f23d8c69cc7b96",
```

```
"DATA":{
  "Email":"a@test.com"
}
```

Формат ответа: JSON

Таблица 17. Параметры ответа

| Имя | Тип | Обязательный? | Описание |
|-------------|--------|---------------|---|
| TerminalKey | String | Да | Платежный ключ, выдается Продавцу при заведении терминала |
| CustomerKey | String | Да | Идентификатор покупателя в системе Продавца |
| RequestKey | String | Да | Идентификатор запроса на привязку карты |
| RebillId | String | Нет | Идентификатор рекуррентного платежа |
| CardId | String | Да | Идентификатор карты в системе Банка |
| Status | String | Да | Статус привязки карты |
| Success | bool | Да | Успешность операции |
| ErrorCode | String | Да | Код ошибки, «0» если успешно |
| Message | String | Нет | Краткое описание ошибки |
| Details | String | Нет | Подробное описание ошибки |

Пример ответа:

```
{
  "Success": true,
  "ErrorCode": "0",
  "TerminalKey": "testRegress",
  "Status": "3DS_CHECKING",
  "CustomerKey": "testRegress5",
  "RequestKey": "8de92934-26c9-474c-a4ce-424f2021d24d"
  "CardId": "5555"
}
```

Метод GetCardList

Описание: Возвращает список привязанных карт у покупателя

URL: <https://securepay.tinkoff.ru/v2/GetCardList>

Метод: POST

Таблица 18. Параметры запроса

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| CustomerKey | String | Да | Идентификатор покупателя в системе Продавца |
| IP | String | Нет | IP-адрес запроса |
| Token | String | Да | Подпись запроса |

Пример запроса:

```
{
  "TerminalKey": "TestB"
  "CustomerKey": "Customer1"
  "Token": "871199b37f207f0c4f721a37cdcc71dfcea880b4a4b85e3cf852c5dc1e99a8d6"
}
```

Формат ответа: Массив JSON

Таблица 19. Параметры ответа

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|--|
| Pan | String | Да | Номер карты 411111*****1111 |
| CardId | String | Да | Идентификатор карты в системе Банка |
| Status | String | Да | Статус карты: А – активная, I – не активная. |
| RebillId | Number | Да | Идентификатор рекуррентного платежа (см. параметр Recurrent в методе Init) |
| ExpDate | String | Нет | Срок действия карты |

Пример ответа:

```
[{"CardId": "4750", "Pan": "543211*****4773", "Status": "A", "RebillId": "145919"},
 {"CardId": "5100", "Pan": "411111*****1111", "Status": "I", "RebillId": "145917"}]
```

Метод RemoveCard

Описание: Удаляет привязанную карту у покупателя

URL: <https://securepay.tinkoff.ru/v2/RemoveCard>

Метод: POST

Таблица 20. Параметры запроса

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| CardId | Number | Да | Идентификатор карты в системе Банка |
| CustomerKey | String | Да | Идентификатор покупателя в системе Продавца |
| IP | String | Нет | IP-адрес запроса |
| Token | String | Да | Подпись запроса |

Пример запроса:

```
{
  "TerminalKey": "TestB"
  "CardId": "4750"
  "CustomerKey": "Customer1"
  "Token": "871199b37f207f0c4f721a37cdcc71dfcea880b4a4b85e3cf852c5dc1e99a8d6">
}
```

Формат ответа: JSON

Таблица 21. Параметры ответа

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| CardId | Number | Да | Идентификатор карты в системе Банка |
| CustomerKey | String | Да | Идентификатор покупателя в системе Продавца |
| Success | bool | Да | Успешность операции |
| Status | String | Да | Статус карты: D – удалена |
| ErrorCode | String | Да | Код ошибки, «0» - успешно |
| Message | String | Нет | Краткое описание ошибки |
| Details | String | Нет | Подробное описание ошибки |

Пример ответа:

```
{ "cardId": "4750",
  "Status": "D",
  "Success": true,
  "ErrorCode": "0",
  "TerminalKey": "TestB",
  "CustomerKey": "Customer1" }
```

Методы для оплаты по QR
Метод GetQr

Описание: регистрирует QR и возвращает информацию о нем. Должен быть вызван после вызова метода Init.

URL: <https://securepay.tinkoff.ru/v2/GetQr>

Метод: POST

Запрос
Параметры запроса

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|--|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| PaymentId | Number | Да | Уникальный идентификатор транзакции в системе Банка |
| DataType | String | Нет | Тип возвращаемых данных PAYLOAD – В ответе возвращается только Payload (по-умолчанию) IMAGE – В ответе возвращается SVG изображение QR |
| Token | String | Да | Подпись запроса |

Пример запроса:

```
{
  "TerminalKey": "TinkoffBankTest",
  "PaymentId": "10063",
  "Token": "871199b37f207f0c4f721a37cdcc71dfcea880b4a4b85e3cf852c5dc1e99a8d6"
}
```

Ответ

Формат ответа: JSON

Параметры ответа

| Наименование | Тип | Обязательный? | Описание |
|--------------|-----|---------------|----------|
|--------------|-----|---------------|----------|

| | | | |
|-------------|--------|-----|--|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| OrderId | String | Да | Номер заказа в системе Продавца |
| Success | bool | Да | Успешность операции (true/false) |
| Data | String | Да | В зависимости от параметра DataType в запросе это: Payload - информация, которая должна быть закодирована в QR или SVG изображение QR в котором уже закодирован Payload |
| PaymentId | Number | Да | Уникальный идентификатор транзакции в системе Банка |
| ErrorCode | String | Да | Код ошибки, «0» - если успешно |
| Message | String | Нет | Краткое описание ошибки |
| Details | String | Нет | Подробное описание ошибки |

Пример ответа:

```
{
  "TerminalKey": "TinkoffBankTest",
  "OrderId": "21057",
  "Success": true,
  "Data": "https://qr.nspk.ru/AS1000670LSS7DN18SJQDNP4B05KLJL2?type=01&bank=100000000001&sur",
  "PaymentId": "10063",
  "ErrorCode": "0"
}
```

Метод GetStaticQr

Описание: При первом вызове регистрирует QR и возвращает информацию о нем при последующих вызовах возвращает информацию о ранее сгенерированном QR.

Перерегистрация статического QR происходит только при смене расчетного счета. Не привязан к конкретному платежу, может быть вызван в любое время без предварительного вызова Init.

URL: <https://securepay.tinkoff.ru/v2/GetStaticQr>

Метод: POST

Запрос

Параметры запроса

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|--|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| Data | String | Нет | Тип возвращаемых данных PAYLOAD – В ответе возвращается только Payload (по-умолчанию) IMAGE – В ответе возвращается SVG изображение QR |

Пример запроса:

```
{
  "TerminalKey": "TinkoffBankTest"
}
```

Ответ

Формат ответа: JSON

Параметры ответа

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|--|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| Success | bool | Да | Успешность операции (true/false) |
| Data | String | Да | В зависимости от параметра DataType в запросе это: Payload - информация, которая должна быть закодирована в QR или SVG изображение QR в котором уже закодирован Payload |
| ErrorCode | String | Да | Код ошибки, «0» - если успешно |
| Message | String | Нет | Краткое описание ошибки |
| Details | String | Нет | Подробное описание ошибки |

Пример ответа:

```
{
  "TerminalKey": "TinkoffBankTest",
  "Success": true,
  "Data": "https://qr.nspk.ru/AS1000670LSS7DN18SJQDNP4B05KLJL2?type=01&bank=10000000001&sur",
  "ErrorCode": "0"
}
```

}

Метод QrMembersList

Описание: Список участников куда может быть осуществлен возврат платежа совершенного по QR.

URL: <https://securepay.tinkoff.ru/v2/QrMembersList>

Метод: POST

Запрос Параметры запроса

| Наименование | Тип | Обязательный? | Описание |
|--------------|--------|---------------|---|
| TerminalKey | String | Да | Идентификатор терминала, выдается Продавцу Банком |
| PaymentId | Number | Да | Уникальный идентификатор транзакции в системе Банка |
| Token | String | Да | Подпись запроса |

Пример запроса:

```
{
  "TerminalKey": "TinkoffBankTest",
  "PaymentId": "10063",
  "Token": "871199b37f207f0c4f721a37cdcc71dfcea880b4a4b85e3cf852c5dc1e99a8d6"
}
```

Ответ

Формат ответа: JSON

Параметры ответа

| Наименование | Тип | Обязательный? | Описание |
|--------------|-----------------|---------------|---|
| Members | Array of Member | Нет | Массив списка участников. Возвращается только если возврат возможен |
| OrderId | String | Да | Номер заказа в системе Продавца |
| Success | bool | Да | Успешность операции (true/false) |
| ErrorCode | String | Да | Код ошибки, «0» - если успешно |
| Message | String | Нет | Краткое описание ошибки |

Объект Member

| Наименование | Тип | Обязательный? | Описание |
|--------------|---------|---------------|--|
| MemberId | String | Да | Идентификатор участника |
| MemberName | String | Да | Наименование участника |
| IsPayee | Boolean | Да | true - если данный участник был получателем указанного платежа, false - в противном случае |

Пример ответа:

```
{
  "Members": [
    {
      "MemberId": "100000000001",
      "MemberName": "АО \"Тинькофф Банк\"",
      "IsPayee": true
    },
    {
      "MemberId": "100000000002",
      "MemberName": "ПАО \"Сбербанк\"",
      "IsPayee": false
    }
  ],
  "Success": true,
  "ErrorCode": "0",
  "Message": "OK"
}
```

Дополнительная информация

ASDKCode.AcquiringUtils

Содержит набор методов для работы с хешами и криптографией.

- sha256 - принимает на строку для вычисления хеша;
- encryptRsa - принимает на вход данные и шифрует их алгоритмом RSA/ECB/PKCS1Padding;
- encodeBase64 - принимает на вход данные и энкодит их в base64.

Алгоритм формирования подписи запроса (Token)

Необходимо:

1. Собрать все параметры запроса Ключ-Значение, кроме параметра Token. Например, [["TerminalKey", "TestB"], ["PaymentId", "20150"]].
2. Добавить туда пару Password-Значение. [["TerminalKey", "TestB"], ["PaymentId", "20150"], ["Password", "123456789"]].

3. Отсортировать по ключам.
[["Password","123456789"],["PaymentId","20150"],["TerminalKey","TestB"]].
4. Конкатенировать значения.
12345678920150TestB.
5. Вычислить SHA-256 от пункта 4.

Алгоритм шифрования карточных данных

1. Введенные пользователем номер карты, expiry date и secure code приводятся к виду:
"PAN=%pan%;ExpDate=%month%%year%;CVV=%secure_code%".
2. Выполняется шифрование строк с шага 1 алгоритмом RSA/ECB/PKCS1Padding с использованием publicKey в качестве ключа.
3. Полученная криптограмма на шаге 2 энкодится алгоритмом Base64.

10. Коды ошибок API и возможные исключения

Ошибки должны пробрасываться классом AcquiringSdk как исключения

AcquiringApiException.

API может возвращать следующие ошибки:

Таблица 22. Ошибки валидации

| Код ошибки | Описание |
|------------|--|
| 0 | Нет ошибки |
| 1 | Параметры не сопоставлены |
| 3 | Внутренняя ошибка системы интернет эквайринга |
| 4 | Запрашиваемое состояние транзакции является неверным |
| 5 | Неверный запрос |
| 6 | Неверный статус карты |
| 7 | Неверный статус покупателя |
| 8 | Неверный статус транзакции |
| 9 | Переадресовываемый url пуст |
| 10 | Метод Charge заблокирован для данного терминала |
| 11 | Невозможно выполнить платеж |
| 50 | Ошибка отправки нотификации |
| 51 | Ошибка отправки Email |
| 52 | Ошибка отправки Sms |
| 53 | Обратитесь к продавцу |
| 54 | Метод вызван повторно |
| 201 | Поле {0} должно быть больше или равно {value} |
| 202 | Терминал заблокирован |
| 203 | Параметры запроса не должны быть пустыми |
| 204 | Неверный токен. Проверьте пару TerminalKey/SecretKey |
| 205 | Неверный токен. Проверьте пару TerminalKey/SecretKey |
| 206 | Email не может быть пустым |
| 207 | Параметр {0} превышает максимально допустимый размер |
| 208 | Наименование ключа из параметра DATA превышает максимально допустимый размер |
| 209 | Значение ключа из параметра DATA превышает максимально допустимый размер |
| 210 | Поле {0} должно быть формата "{regex}" |

| Код ошибки | Описание |
|------------|--|
| 211 | Неверный формат IP |
| 212, 213 | Поле {0} должно быть формата "{regex}" |
| 214 | Поле {0} числовое значение должно укладываться в формат (<{integer} цифр>.<{fraction} цифр>) |
| 215 | Поле {0} должно быть формата "{regex}" |
| 216 | Поле {0} должно быть формата "{regex}" |
| 217 | Поле {0} должно быть больше или равно {value} |
| 218 | Значение {0} не является числовым |
| 219 | Неверный срок действия карты |
| 220 | Поле {0} должно быть формата "{regex}" |
| 221 | Значение {0} не является числовым |
| 222 | Поле {0} должно быть больше или равно {value} |
| 223 | Поле {0} должно быть больше или равно {value} |
| 224, 225 | Неверный формат Email |
| 226-230 | Поле {0} должно быть формата "{regex}" |
| 231 | Не найден идентификатор карты |
| 233-239 | Поле {0} должно быть формата "{regex}" |
| 240, 241 | Поле {0} должно быть больше или равно {value} |
| 242 | Поле {0} должно быть формата "{regex}" |
| 243 | Ошибка шифрования карточных данных |
| 244 | Ошибка сопоставления карточных данных |
| 245-250 | Параметр {0} не сопоставлен |
| 251 | Неверная сумма. Сумма должна быть больше или равна {0} копеек |
| 252 | Срок действия карты истек |
| 253 | Валюта {0} не разрешена для данного терминала |

Таблица 23. Ошибки оплаты

| Код ошибки | Описание |
|------------|--|
| 99 | Воспользуйтесь другой картой, банк выпустивший карту отклонил операцию |
| 101 | Не пройдена идентификация 3DS |
| 1006 | Проверьте реквизиты или воспользуйтесь другой картой |
| 1012 | Воспользуйтесь другой картой |
| 1013 | Повторите попытку позже |
| 1014 | Неверно введены реквизиты карты. Проверьте корректность введенных |

| Код ошибки | Описание |
|------------|--|
| | данных |
| 1030 | Повторите попытку позже |
| 1033 | Проверьте реквизиты или воспользуйтесь другой картой |
| 1034-1043 | Воспользуйтесь другой картой, банк выпустивший карту отклонил операцию |
| 1051 | Недостаточно средств на карте |
| 1054 | Проверьте реквизиты или воспользуйтесь другой картой |
| 1057, 1065 | Воспользуйтесь другой картой, банк выпустивший карту отклонил операцию |
| 1082 | Проверьте реквизиты или воспользуйтесь другой картой |
| 1089 | Воспользуйтесь другой картой, банк выпустивший карту отклонил операцию |
| 1091 | Воспользуйтесь другой картой |
| 1096 | Повторите попытку позже |
| 9999 | Внутренняя ошибка системы |

11. Поддержка

- Баги и feature-реквесты можно направлять в раздел [issues](#);
- Подробное описание методов: [API методов](#).