PAYTURE

Порядок технической интеграции Apple Pay

Версия 1.1

Дата: 22 мая 2020 г.

+7 495 783 83 93
info@payture.com
payture.com

Изменения документа

Версия	Описание	Дата
1.0	Первая версия документа	31.01.20
1.1	Незначительные изменения	22.05.20

Содержание

Общая информация

Apple Pay позволяет Покупателям оплачивать покупки в одно касание на сайтах, в мобильных приложениях и других каналах оплаты. Apple Pay отображается на платежной странице в виде специальной кнопки и доступен для выбора Покупателем наряду с другими способами оплаты.

Актуальный список устройств, совместимых с Apple Pay, доступен на сайте Apple.

Спецификация API размещена на payture.com/api.

С основными терминами можно ознакомиться на странице справочной информации: https://payture.com/api/#helpful-information_.

Об ошибках, неточностях, а также предложения по улучшению документации, пожалуйста, сообщайте службе поддержки Payture: support@payture.com.

Виды интеграций Apple Pay

Payture предлагает различные варианты подключения Apple Pay.

	Платежи на странице оплаты Payture Интеграция с Apple Pay на стороне Payture	**Платежи на сайте или в приложении Продавца** Необходима прямая интеграция Продавца с Apple Pay
Только разовые платежи	Интерфейс InPay **Описание:** Подключение Apple Pay — раздел 1.1	Интерфейс Payture API **Описание:** Подключение Apple Pay в приложении — раздел 1.2 Подключение Apple Pay на сайте — раздел 1.3 Выполнение платежа — раздел 2
Разовые и рекуррентные платежи	Интерфейс eWallet (на стороне Payture) **Описание:** Подключение Apple Pay — раздел 1.1 Рекуррентные платежи — раздел 3.2	Интерфейс eWallet (на стороне Продавца) **Описание:** Подключение Apple Pay в приложении — раздел 1.2 Подключение Apple Pay на сайте — раздел 1.3 Выполнение платежа — раздел 3.1 Рекуррентные платежи — раздел 3.2

1 Подготовка к платежам — подключение Apple Pay

В зависимости от канала приема платежей порядок подключения Apple Pay различается. Если прием платежей происходит на странице оплаты Payture, от Продавца не требуется дополнительная интеграция с Apple. А для приема платежей с Apple Pay на сайте или в приложении Продавца необходима дополнительная интеграция с Apple для получения криптограммы с платежными данными.

1.1 Apple Pay на странице оплаты Payture

При таком варианте интеграции Покупатель находится на сайте Продавца только до момента ввода данных своей платежной карты или до оплаты с Apple Pay. Для оплаты Покупатель перенаправляется на платежную страницу на стороне Payture. После оплаты Покупатель будет проинформирован о результате и возвращен обратно на сайт Продавца, а Продавцу будут отправлены нотификации с результатом платежа.

Прием платежей на стороне Payture возможен с использованием двух интерфейсов:

- InPay — простой способ приема платежей без сохранения карт;
- eWallet — платежи с возможностью сохранения карт Покупателя и выполнения рекуррентных платежей.

Порядок интеграции в таком случае почти не отличается от стандартного. Продавцу не требуется интегрироваться с Apple Pay, все необходимое уже реализовано на стороне Payture.

Для использования Apple Pay на странице оплаты Payture достаточно:

- включить Apple Pay на шаблоне страницы оплаты (как это сделать см. на вкладке «Оплата с помощью Apple Pay, Google Pay, Samsung Pay»);
- сообщить о необходимости подключения Apple Pay службе поддержки Payture и направить обновленные шаблоны.

Кнопка Apple Pay будет автоматически отображаться на странице оплаты Payture, если:

1) Покупатель использует браузер Safari на устройстве, поддерживающим Apple Pay;
2) у Покупателя есть доступные для оплаты карты в Apple Pay.

Если оплата была выполнена картой из Apple Pay, дополнительное поле нотификации ExternalWallet (при его добавлении в нотификацию) принимает значение «ApplePay».

1.2 Apple Pay в приложении Продавца

Для подключения Apple Pay в приложении Продавцу необходимо самостоятельно выполнить интеграцию с Apple для получения на устройстве Покупателя криптограммы с платежными данными.

Процесс выполнения платежа в приложении с Apple Pay состоит из двух основных этапов:

1. Получение зашифрованных платежных данных из Apple Pay;
2. Проведение платежа в платежном шлюзе Payture.

1.2.1 Порядок подключения

Для внедрения Apple Pay в приложении потребуется:

- Аккаунт разработчика в Apple Developer Program;
- Использование фреймворка PassKit.

В рамках интеграции Продавцу необходимо выполнить следующие шаги:

1) Зарегистрировать **Merchant ID** — идентификатор Продавца в Apple, необходимый для выполнения платежей;
2) Получить **Payment Processing Certificate** — сертификат, необходимый для шифрования платежных данных. Платежные данные передаются от Apple, зашифрованные открытым ключом. Payture для выполнения платежа расшифровывает данные закрытым ключом, который содержится в сертификате. Этот сертификат выдается на 25 месяцев.

   Подробная инструкция по регистрации Merchant ID и созданию Payment Processing Certificate.
3) Направить в службу поддержки Payture:
   - Merchant ID;
   - закрытый ключ сертификата Payment Processing Certificate;
   - пароль закрытого ключа;
4) Получить от службы поддержки Payture параметры тестового доступа;
5) Включить возможность Apple Pay в вашем проекте XCode. Инструкция Apple;
6) Реализовать необходимые для Продавца сценарии выполнения платежей;
7) Провести тестирование интеграции с Apple и платежным шлюзом Payture;
8) Получить от службы поддержки Payture параметры боевого доступа;
9) Выполнить переход в боевое окружение.

1.2.2 Получение paymentData в приложении

Перед выполнением платежа в платежном шлюзе Продавцу необходимо получить платежные данные в Apple Pay: **paymentData** из объекта PKPaymentToken.

Все необходимое взаимодействие с Apple для получения платежных данных реализуется на стороне Продавца. Для этого используется фреймворк **PassKit**. Документация по работе с **PassKit** доступна на сайте Apple.

1.3 Apple Pay на сайте Продавца

Для подключения Apple Pay на сайте Продавца необходимо самостоятельно выполнить интеграцию с Apple для получения на устройстве Покупателя криптограммы с платежными данными.

Процесс выполнения платежа на сайте Продавца состоит из трех основных этапов:

1 Создание платежной сессии Apple Pay и валидация Продавца в Apple Pay;

2 Получение зашифрованных платежных данных из Apple Pay;

3 Проведение платежа в платежном шлюзе Payture.

1.3.1 Порядок подключения

Для внедрения Apple Pay на веб-странице потребуется:

- Аккаунт разработчика в Apple Developer Program;
- Использование HTTPS на странице с Apple Pay и поддержка TLS 1.2 (см. подробнее);
- Соблюдение правил Apple по использованию Apple Pay на сайтах;
- Использование фреймворка Apple Pay JS API.

В рамках интеграции Продавцу необходимо выполнить следующие шаги:

1) Зарегистрировать **Merchant ID** — идентификатор Продавца в Apple, необходимый для выполнения платежей;
2) Получить **Payment Processing Certificate** — сертификат, необходимый для шифрования платежных данных. Платежные данные передаются от Apple, зашифрованные открытым ключом. Payture для выполнения платежа расшифровывает данные закрытым ключом, который содержится в сертификате. Этот сертификат выдается на 25 месяцев.

   Подробная инструкция по регистрации Merchant ID и созданию Payment Processing certificate.
3) Направить в службу поддержки Payture:
   - Merchant ID;
   - закрытый ключ сертификата Payment Processing Certificate;
   - пароль закрытого ключа.
4) Получить от службы поддержки Payture параметры тестового доступа;
5) Зарегистрировать и верифицировать все домены и поддомены, где будет использоваться Apple Pay;
6) Получить **Merchant Identity certificate** — TLS сертификат, используемый для подтверждения данных Продавца и авторизации платежных сессий в Apple. Используется Продавцом, передавать в Payture его не нужно. Срок действия сертификата 25 месяцев;

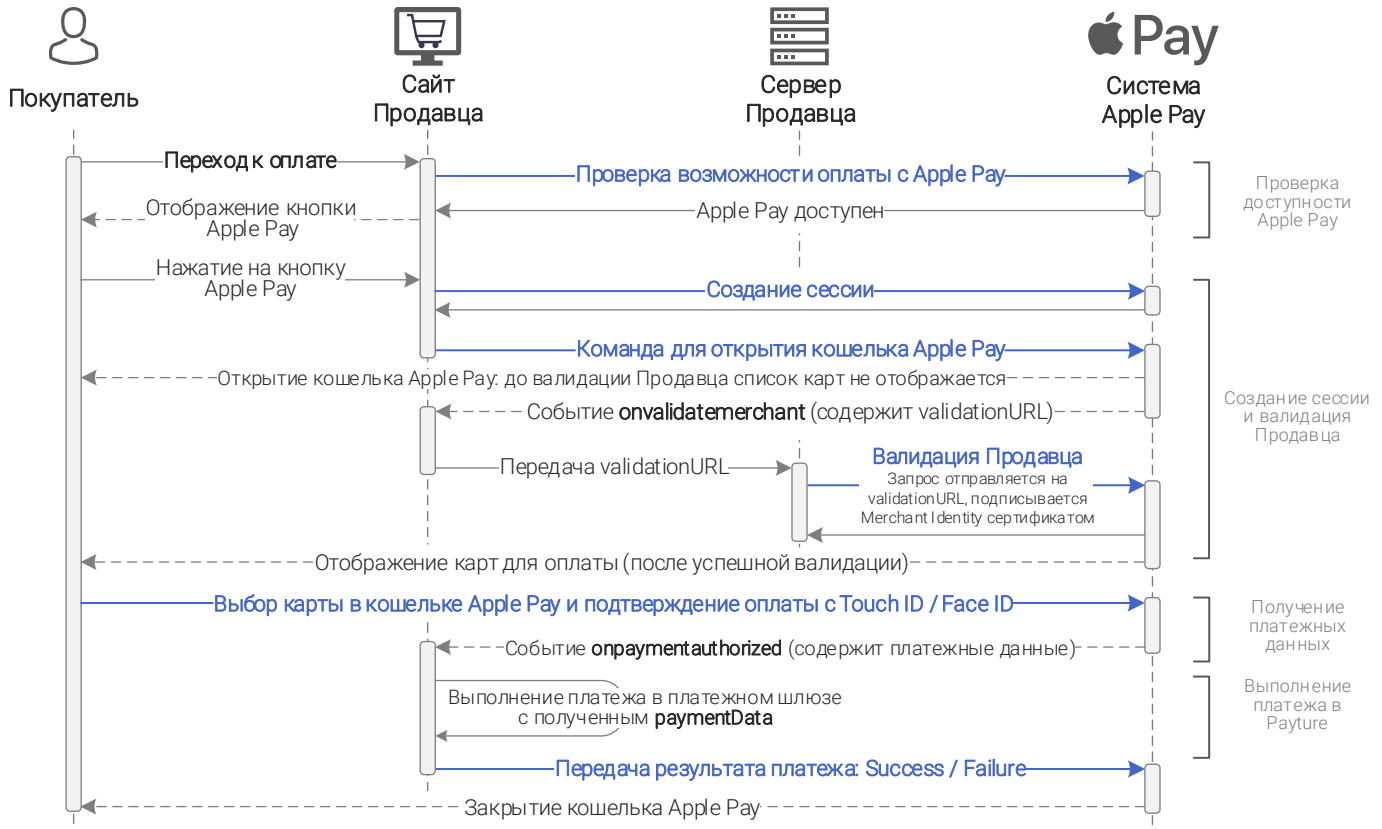
   Подробная инструкция по верификации доменов и регистрации Merchant Identity certificate.
7) Реализовать необходимые для Продавца сценарии выполнения платежей;
8) Провести тестирование интеграции с Apple и платежным шлюзом Payture;
9) Получить от службы поддержки Payture параметры боевого доступа;
10) Выполнить переход в боевое окружение.

1.3.2 Получение paymentData на сайте

Перед выполнением платежа в платежном шлюзе Продавцу необходимо получить платежные данные в Apple Pay: **paymentData** из объекта PaymentToken.

Все необходимое взаимодействие с Apple для получения платежных данных реализуется на стороне Продавца. Для этого используется фреймворк Apple Pay JS API.

Порядок получения paymentData:



Пример реализации

```
if (window.ApplePaySession) {                  // Оплата с Apple Pay поддерживается на текущем устройстве
       var merchantIdentifier = YOUR_MERCHANT_ID;           // Ваш идентификатор мерчанта (Merchant ID)
       var promise = ApplePaySession.canMakePaymentsWithActiveCard(merchantIdentifier);
       promise.then(function (canMakePayments) {
              if (canMakePayments) {      // В Apple Pay у Покупателя есть хотя бы одна карта для оплаты
                     $('#apple-pay-button').css({'display':'inline-flex'});  /* Отображение кнопки Apple
Pay */

                     // Сессию необходимо создавать при клике Покупателя на кнопку Apple Pay

                     document.getElementById("apple-pay-button").onclick = function(evt) {
                        var request = {      // Пример задания параметров объекта ApplePayPaymentRequest
                           countryCode: 'RU',
                           currencyCode: 'RUB',
                           supportedNetworks: ['visa', 'masterCard'],
                           merchantCapabilities: ['supports3DS'],
                           total: {
                            label: payture.com,
                            amount: 42.42
                           },
                        }
                        var session = new ApplePaySession(1, request);     // Создание сессии Apple Pay
```

```
                        /* Далее необходимо реализовать функции для обработки событий, как минимум для
onvalidatemerchant и onpaymentauthorized */
                        /* Функция для обработки события onvalidatemerchant – срабатывает при открытии
окна с Apple Pay и содержит validationURL. При срабатывании этого события необходимом выполнить валидацию
Продавца. Список карт для оплаты будет отображен Покупателю только после успешной валидации */

                        session.onvalidatemerchant = function (event) {
                           var promise = Payture.applePerformValidation(event.validationURL);  /* Функция
для валидации Продавца (подробнее здесь). Пример функции приведен ниже */
                           promise.then(function (merchantSession) {
                              session.completeMerchantValidation(merchantSession);         /* Завершение
валидации */
                           });
                        }

                        /* Функция для обработки события onpaymentauthorized – срабатывает после
подтверждения Покупателем платежа через Touch ID / Face ID. Событие содержит параметр payment. В
платежный шлюз необходимо передать payment->token->paymentData в кодировке Base64 */

                        session.onpaymentauthorized = function (event) {
                           var promise = Payture.sendPaymentToken(event.payment.token.paymentData); /*
Функция для оплаты в платежном шлюзе */
                           promise.then(function (data) {
                              var status;
                              if (data.Success){ // Передача результата платежа (success/failure) в Apple
                                 status = ApplePaySession.STATUS_SUCCESS;
                                 session.completePayment(status);
                              } else {
                                 status = ApplePaySession.STATUS_FAILURE;
                                 session.completePayment(status);
                              }
                           });
                        }

                        /* Открытие окна с кошельком Apple Pay
                        После открытия начинается процесс валидации Продавца «на лету», в этот момент
список карт еще не доступен Покупателю */
                        session.begin();

               } else {  /* В Apple Pay нет доступных карт для оплаты: кнопка оплаты не отображается. Но
также этом месте можно добавить, например, кнопку «Set Up Apple Pay» */
                        $('#apple-pay-button').css({'display':'none'});
               }
        });
} else {                // Apple Pay не поддерживается на устройстве / в браузере, кнопка не отображается
        $('#apple-pay-button').css({'display':'none'});
}
```

Пример функции валидации Продавца applePerformValidation()

Запрос валидации должен быть отправлен с сервера Продавца и подписан Merchant Identity Certificate. В ответе сервер получит объект MerchantSession, который необходимо передать в метод completeMerchantValidation.

```
function applePerformValidation(valURL) {
        return new Promise(function (resolve, reject) {
                $.post(API_DOMAIN_NAME + "api/ApplePayValidate", {
                        validationUrl: valURL
                        DomainName: 'apple.payture.com' /* Домен, с которого выполняется оплата – ваш
верифицированный домен */
                }, function (data)
                        resolve(data);
                });
        });
}
```

Пример отправки запроса валидации, подписанного Merchant Identity Certificate, на стороне сервера.

```
private string ApplePayValidateMerchant( string validationUrl, string DomainName )
{
	...
	string data = new { merchantIdentifier = YOUR_MERCHANT_ID, domainName = DomainName, displayName =
"Payture" }.ToJson();
	Process process = new Process();
	process.StartInfo.FileName = "curl//curl.exe";
	process.StartInfo.Arguments = String.Format( "-gv -tlsv1.2 --data {0} --cert
pass//to//Sert.crt:PASSWORD {1}", data, validationUrl );
	process.StartInfo.UseShellExecute = false;
	process.StartInfo.RedirectStandardOutput = true;
	process.StartInfo.RedirectStandardError = true;
	process.StartInfo.CreateNoWindow = true;
	process.Start();
	output = process.StandardOutput.ReadToEnd();
	string err = process.StandardError.ReadToEnd();
	...
}
```

2 Платежи Payture API

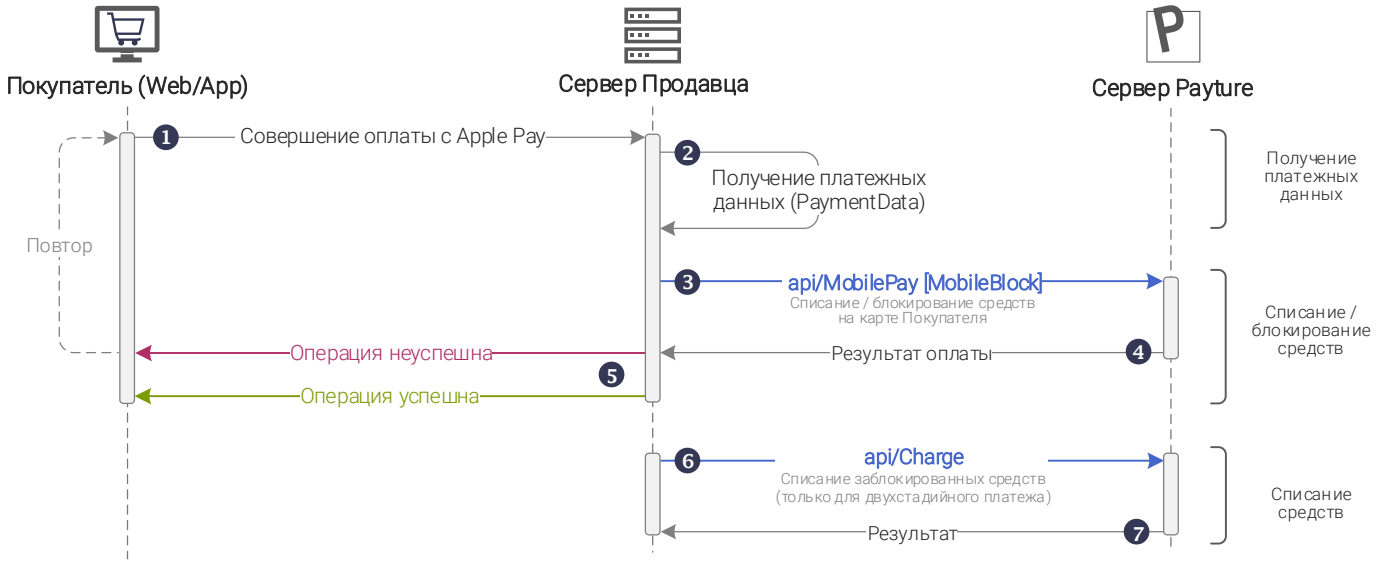
Для выполнения оплаты с Apple Pay Продавцу необходимо передать в Payture платежные данные **paymentData**, полученные на сайте или в приложении Продавца.

Порядок проведения операций возврата, отмены блокировки и получения статуса соответствует стандартному для Payture API.

Спецификация **Payture API** доступна на https://payture.com/api/#payture-api_.

2.1 Выполнение платежа

При оплате с Apple Pay взаимодействие с платежным шлюзом выполняется по следующей схеме:



	Одностадийный платеж	Двухстадийный платеж
1	Покупатель формирует заказ на сайте / в приложении Продавца и переходит к оплате.	
2	Перед проведением оплаты в платежном шлюзе Продавцу необходимо получить платежные данные в системе Apple Pay. Порядок получения платежных данных для приложения указан в разделе 1.2.2, для сайта — в разделе 1.3.2.	
3	Для одностадийного списания необходимо передать в запросе api/MobilePay платежные данные **paymentData** из объекта paymentToken, закодированные в Base64. **Примечание**. Списанные средства могут быть полностью или частично возвращены Покупателю командой api/Refund.	Для блокировки средств необходимо передать в запросе api/MobileBlock платежные данные **paymentData** из объекта paymentToken, закодированные в Base64. **Примечание**. Заблокированные средства могут быть списаны (шаг 6) или разблокированы командой api/Unblock.
4	Платежный шлюз обрабатывает запрос и возвращает ответ с результатом оплаты. **Примечание**. 3-D Secure аутентификация для платежей с Apple Pay не требуется.	
5	На сайте / в приложении Продавца для Покупателя отображаются результаты операции. При неуспешной попытке Продавец может повторить платеж. В таком случае необходимо передавать новый номер заказа **OrderId** в платежный шлюз.	

	Одностадийный платеж	Двухстадийный платеж
6		Для списания заблокированных средств Продавец должен отправить запрос api/Charge, используя номер заказа **OrderId** из запроса на блокировку средств. Сумма списания **Amount** не должна превышать заблокированную. Если сумма списания меньше заблокированной, то оставшиеся средства будут разблокированы на карте Покупателя. **Примечание 1**. Запрос на списание должен быть осуществлен в течение 7 дней после блокировки. **Примечание 2**. Списание (или отмена блокировки) может выполняться автоматически через заданный промежуток времени после блокировки по согласованию со службой поддержки Payture.
7	—	Платежный шлюз обрабатывает запрос и возвращает ответ с результатом. После успешного списания платеж перейдет в статус Charged. **Примечание**. Списанные средства могут быть полностью или частично возвращены на карту Покупателя командой api/Refund.

3 Платежи eWallet

Для выполнения оплаты с Apple Pay Продавцу необходимо передать в Payture платежные данные **paymentData**, полученные на сайте или в приложении Продавца.

Порядок проведения операций возврата, отмены блокировки и получения статуса соответствует стандартному для Payture eWallet.

Спецификация **Payture eWallet** доступна на payture.com/api/#ewallet_.

3.1 Выполнение платежа

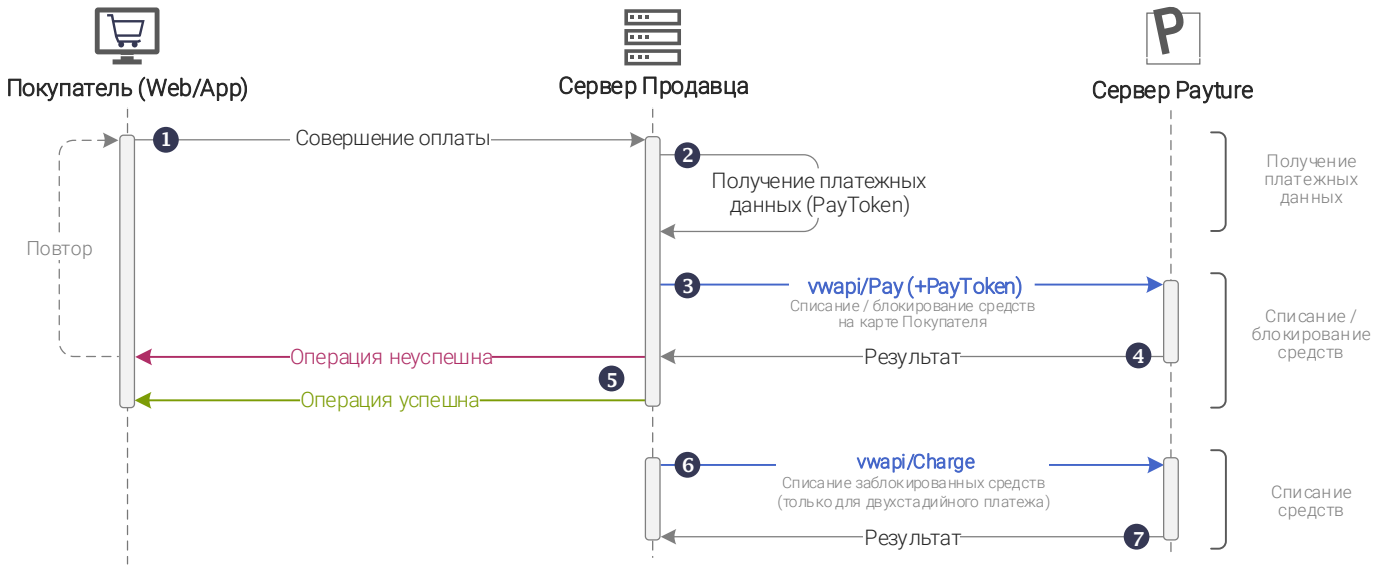
Через интерфейс eWallet ввод платежных данных Покупателя может происходить как на стороне Продавца (на сайте или в приложении), так и на странице защищенного шлюза Payture.

3.1.1 На стороне Payture

См. раздел 1.1 Подключение Apple Pay на веб-странице Payture.

3.1.2 На стороне Продавца

При оплате с Apple Pay взаимодействие с платежным шлюзом выполняется по следующей схеме:



	Одностадийный платеж	Двухстадийный платеж
1	Покупатель формирует заказ на сайте / в приложении Продавца и переходит к оплате.	
2	Перед проведением оплаты в платежном шлюзе Продавцу необходимо получить платежные данные в системе Apple Pay. Процесс получения платежных данных для приложения описан в разделе 1.2.2, для сайта — в разделе 1.3.2.	
3	Для одностадийного списания необходимо передать в параметре PayToken запроса vwapi/Pay* платежные данные **paymentData** из объекта paymentToken, закодированные в Base64. Для одностадийного платежа необходимо указать **SessionType=Pay**, либо не передавать	Для блокировки средств необходимо передать в параметре PayToken запроса vwapi/Pay* платежные данные **paymentData** из объекта paymentToken, закодированные в Base64. Для двухстадийного платежа необходимо указать **SessionType=Block**.

* Набор параметров в запросе vwapi/Pay для платежей с Apple Pay отличается от стандартного. Спецификацию для оплаты через Apple Pay см. в разделе Сервисы -> Apple Pay, Google Pay, Samsung Pay -> API -> Payture eWallet.

	Одностадийный платеж	Двухстадийный платеж
4	Платежный шлюз обрабатывает запрос и возвращает ответ с результатом оплаты. После успешного списания платеж перейдет в статус Charged. **Примечание**. Списанные средства могут быть полностью или частично возвращены Покупателю командой vwapi/Refund.	Платежный шлюз обрабатывает запрос и возвращает ответ с результатом оплаты. После успешной блокировки платеж перейдет в статус Authorized. **Примечание**. Заблокированные средства могут быть списаны командой vwapi/Charge (шаг 6) или разблокированы командой vwapi/Unblock.
5	На сайте / в приложении Продавца для Покупателя отображаются результаты операции. После неуспешной попытки Продавец может повторить платеж. В таком случае необходимо передавать новый номер заказа **OrderId** в платежный шлюз.	
6	–	Для списания заблокированных средств Продавец должен отправить запрос vwapi/Charge, используя номер заказа **OrderId** из запроса на блокировку средств. Сумма списания **Amount** не должна превышать заблокированную. Если сумма списания меньше заблокированной, то оставшиеся средства будут разблокированы на карте Покупателя. **Примечание 1**. Запрос на списание должен быть осуществлен в течение 7 дней после блокировки. **Примечание 2**. Списание (или отмена блокировки) может выполняться автоматически через заданный промежуток времени после блокировки по согласованию со службой поддержки Payture.
7	–	Платежный шлюз обрабатывает запрос и возвращает ответ с результатом. После успешного списания платеж перейдет в статус Charged. **Примечание**. Списанные средства могут быть полностью или частично возвращены на карту Покупателя командой vwapi/Refund.

3.2 Рекуррентные платежи

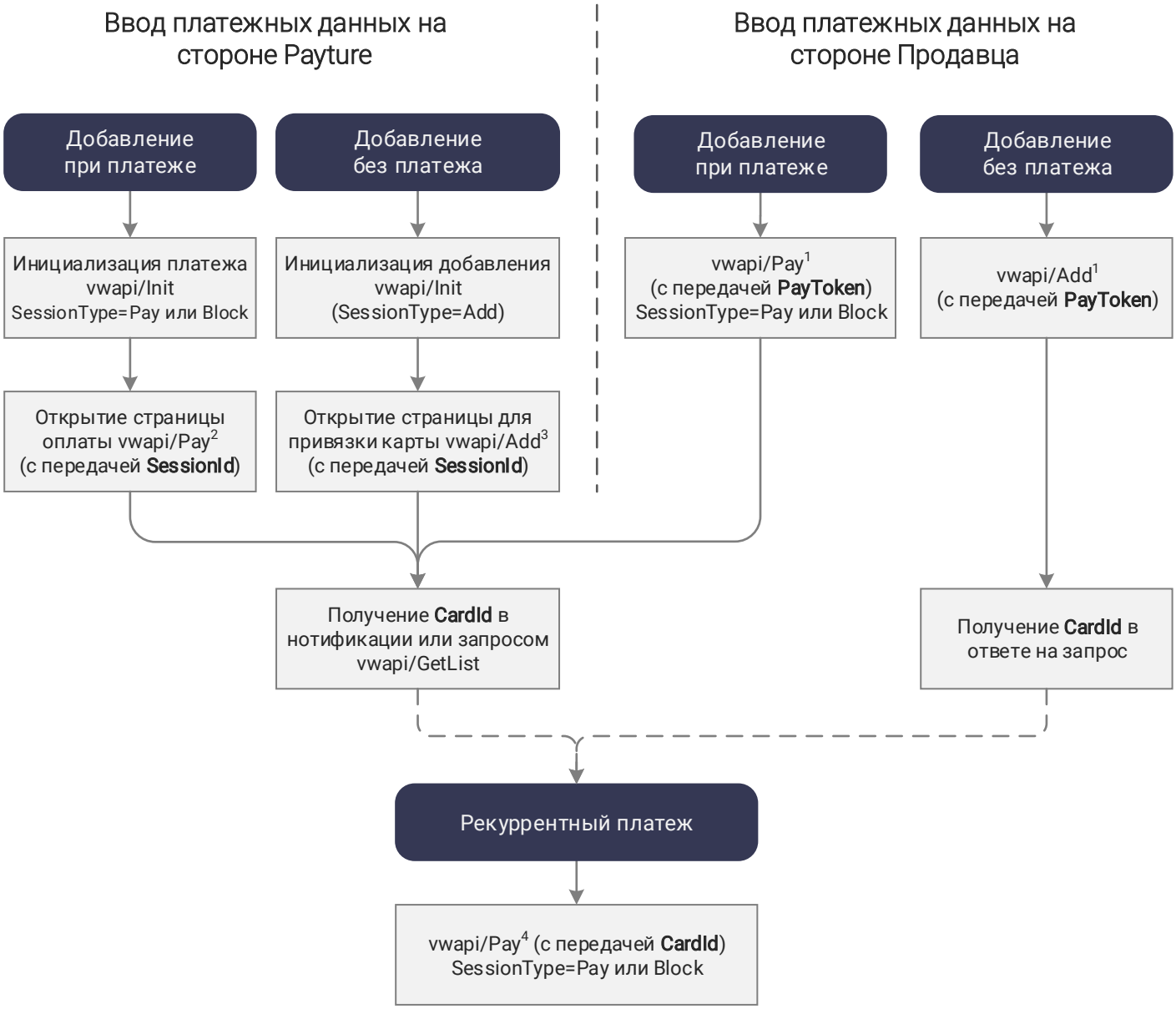
Рекуррентный платеж может быть выполнен только после привязки карты к Покупателю.

> Для выполнения рекуррентных платежей необходимо получение согласия Покупателя при первом платеже или добавлении карты.

Для выполнения платежа необходимо передать в платежный шлюз идентификатор ранее добавленной карты **CardId**. Другие реквизиты, в том числе код CVC2/CVV2, не требуются для рекуррентных платежей.

3.2.1 Привязка карты

Схемы привязки (регистрации) карты к Покупателю различаются в зависимости от вида интеграции. Карта может быть привязана автоматически при платеже или отдельным запросом на добавление карты.



[1] Спецификация запросов vwapi/Pay, vwapi/Add с передачей **PayToken** размещена в разделе в разделе Сервисы -> Apple Pay, Google Pay, Samsung Pay -> API -> Payture eWallet.

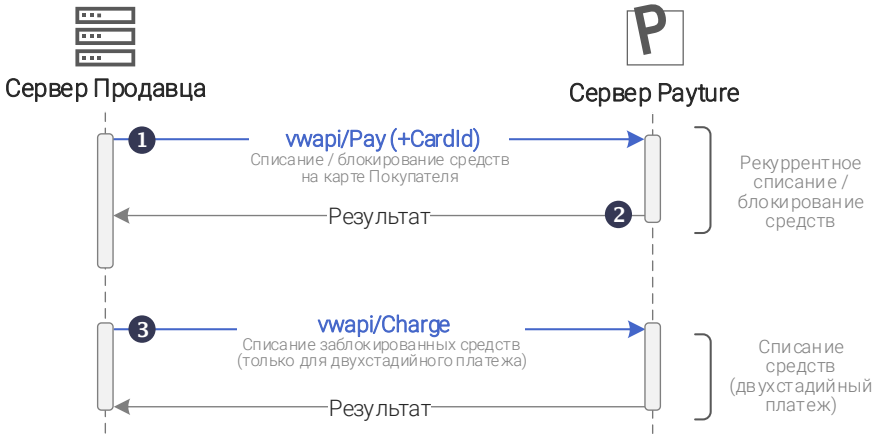
[2] Спецификация запроса vwapi/Pay с передачей **SessionId** размещена в разделе Платежи -> eWallet -> Карты -> Add -> На стороне Payture.

[3] Спецификация запроса vwapi/Add с передачей **SessionId** размещена в разделе Платежи -> eWallet -> Pay — на стороне Payture.

[4] Спецификация запроса vwapi/Pay с передачей **CardId** размещена в разделе Платежи -> eWallet -> Pay — рекуррентные платежи.

3.2.2 Выполнение рекуррентного платежа

Рекуррентный платеж выполняется без участия Покупателя по ранее привязанной карте.



	Одностадийный платеж	Двухстадийный платеж
1	Продавец отправляет запрос vwapi/Pay*. В запросе передается идентификатор ранее зарегистрированной карты **CardId**. Для одностадийного платежа необходимо указать **SessionType=Pay**, либо не передавать.	Продавец отправляет запрос vwapi/Pay*. В запросе передается идентификатор ранее зарегистрированной карты **CardId**. Для двухстадийного платежа необходимо указать **SessionType=Block**.
	* Спецификация запроса vwapi/Pay с передачей **CardId** размещена в разделе Платежи -> eWallet -> Pay — рекуррентные платежи.	
2	Платежный шлюз обрабатывает запрос и возвращает ответ с результатом оплаты. После успешного списания платеж перейдет в статус Charged. **Примечание**. Списанные средства могут быть полностью или частично возвращены Покупателю командой vwapi/Refund.	Платежный шлюз обрабатывает запрос и возвращает ответ с результатом оплаты. После успешной блокировки платеж перейдет в статус Authorized. **Примечание**. Заблокированные средства могут быть списаны командой vwapi/Charge (шаг 4) или разблокированы командой vwapi/Unblock.
3	—	Для списания заблокированных средств Продавец должен отправить запрос vwapi/Charge, используя номер заказа **OrderId** из запроса на блокировку средств. Сумма списания **Amount** не должна превышать заблокированную. Если сумма списания меньше заблокированной, то оставшиеся средства будут разблокированы на карте Покупателя. После успешного списания платеж перейдет в статус Charged. **Примечание 1**. Запрос на списание должен быть осуществлен в течение 7 дней после блокировки. **Примечание 2**. Списание (или отмена блокировки) может выполняться автоматически через заданный промежуток времени после блокировки по согласованию со службой поддержки Payture.

4 Чеки по 54-ФЗ вместе с платежом

Отправка чеков выполняется при поддержке одного из партнеров Payture. Payture реализует техническую возможность передачи чека через сервис онлайн-касс в ФНС.

Описание порядка передачи чеков и структуры чека: payture.com/api#kassy-fz54_cheque-format-with-payment_

> При необходимости Продавец может передавать чеки отдельно от платежа. Подробнее см. «Передача чека без платежа»: payture.com/api#kassy-fz54_cheque-format-with-payment_

Внимание! Продавцу важно правильно формировать чек и соблюдать все ограничения параметров чека для успешной передачи чека в ФНС. При ошибке передачи чека в сервис онлайн-касс, ошибка не будет передана в платежном запросе, так как чек отправляется асинхронно.

Результаты передачи чека Продавец может получить в рамках нотификаций сервиса чеков или запроса статуса чека apicheque/Status.

5 Тестирование интеграции с платежным шлюзом

На тестовой среде все взаимодействие происходит с платежным шлюзом Payture, который эмулирует поведение банка-эквайера. Аналитика и информация о платежах доступна в тестовом личном кабинете Payture.

Для операций с Apple Pay используйте любой платежный токен Apple Pay. На тестовой среде платежные данные, полученные после расшифрования, будут изменены платежным шлюзом на данные тестовой карты 4111 1111 1111 1112. Результат для этой карты — успешное выполнение операции.

При необходимости определения другого поведения для платежного токена, например получение ошибки, обратитесь в службу поддержки.

Тестирование отправки чеков

Формирование чеков в тестовой среде выполняется с использованием тестовых онлайн-касс. Полученный чек идентичен боевому, но не является фискальным документом и не отправляется в ОФД и ФНС.

Приложение 1. Документация Apple

Вся необходимая документация для интеграции с Apple Pay доступна на сайте Apple:

Об Apple Pay:
https://developer.apple.com/apple-pay/planning/

Правила по использованию Apple Pay на сайтах:
https://developer.apple.com/apple-pay/acceptable-use-guidelines-for-websites/

Описание фреймворка Apple Pay JS для интеграции на сайте:
https://developer.apple.com/documentation/apple_pay_on_the_web/apple_pay_js_api

Конфигурирование и регистрация ApplePay для сайта:
https://help.apple.com/developer-account/#/dev1731126fb

Описание фреймворка PassKit для интеграции в приложении:
https://developer.apple.com/documentation/passkit/apple_pay

Конфигурирование и регистрация ApplePay для приложения:
https://help.apple.com/developer-account/#/devb2e62b839 —

Описание структуры Payment Token:
https://developer.apple.com/library/archive/documentation/PassKit/Reference/PaymentTokenJSON/PaymentTokenJSON.html

О тестовой среде Apple Pay:
https://developer.apple.com/apple-pay/sandbox-testing/