**Author:** Evgenia Grishina

**Version:** 118

**Date:** 2015.06.09

**Alfabank RBS**

**Merchant Manual (RU) New**

0

Contents

[1 Read before reading the documentation. 12](#_Toc256000000)

[2 Glossary 13](#_Toc256000001)

[3 Algorithm to connect to the Payment Gateway 15](#_Toc256000002)

[4 General description of the methods of interaction with the Payment Gateway 17](#_Toc256000003)

[4.1 Payment mechanisms (one-phase and two-phase) 17](#_Toc256000004)

[4.2 Schemes of interaction with the Payment Gateway 17](#_Toc256000005)

[4.3 Interfaces of interaction with the Payment Gateway 19](#_Toc256000006)

[4.4 Connection URLs 20](#_Toc256000007)

[5 Description of the interaction schemes 23](#_Toc256000008)

[5.1 One-phase payment with specifying card data on the payment page 23](#_Toc256000009)

[5.1.1 Scenario of a payment for an order 23](#_Toc256000010)

[5.1.2 Cancellation of a payment for an order 26](#_Toc256000011)

[5.1.3 Refund of a payment for an order 27](#_Toc256000012)

[5.1.4 Check for enrolment of a card to 3D-Secure 27](#_Toc256000013)

[5.1.5 Adding additional parameters to an order 27](#_Toc256000014)

[5.1.6 Payment statistics for a certain period 28](#_Toc256000015)

[5.1.7 Adding a card to the list of SSL-cards 28](#_Toc256000016)

[5.2 Two-phase payment with specifying card data on the payment page 29](#_Toc256000017)

[5.2.1 Scenario of a payment for an order 29](#_Toc256000018)

[5.2.2 Cancellation of a payment for an order 32](#_Toc256000019)

[5.2.3 Refund of a payment for an order 33](#_Toc256000020)

[5.2.4 Check for enrolment of a card to 3D-Secure 33](#_Toc256000021)

[5.2.5 Adding additional parameters to an order 34](#_Toc256000022)

[5.2.6 Payment statistics for a certain period 34](#_Toc256000023)

[5.2.7 Adding a card to the list of SSL-cards 34](#_Toc256000024)

[5.3 Payment with the aid of a binding on the payment page 34](#_Toc256000025)

[5.3.1 General description of the autocompletetion functionality on the payment page 34](#_Toc256000026)

[5.3.2 Scenario of a payment for an order 37](#_Toc256000027)

[5.3.3 Getting the bindings list of a customer 40](#_Toc256000028)

[5.3.4 Getting the bindings list of a bank card 40](#_Toc256000029)

[5.3.5 Deactivating/activating an existing binding 40](#_Toc256000030)

[5.3.6 Changing the validity period of a binding 40](#_Toc256000031)

[5.3.7 Adding a card to the list of SSL-cards 41](#_Toc256000032)

[5.4 One-phase auto-payments 42](#_Toc256000033)

[5.4.1 Scenario of executing an initial payment 42](#_Toc256000034)

[5.4.2 Scenario of executing an auto-payment 45](#_Toc256000035)

[5.4.3 Getting the bindings list of a customer 46](#_Toc256000036)

[5.4.4 Deactivating/activating an existing binding 46](#_Toc256000037)

[5.4.5 Changing the validity period of a binding 47](#_Toc256000038)

[5.4.6 Adding a card to the list of SSL-cards 47](#_Toc256000039)

[5.5 Two-phase auto-payments 48](#_Toc256000040)

[5.5.1 Scenario of executing an initial payment 48](#_Toc256000041)

[5.5.2 Scenario of executing an auto-payment 52](#_Toc256000042)

[5.5.3 Getting the bindings list of a customer 53](#_Toc256000043)

[5.5.4 Deactivating/activating an existing binding 53](#_Toc256000044)

[5.5.5 Changing the validity period of a binding 53](#_Toc256000045)

[5.5.6 Adding a card to the list of SSL-cards 53](#_Toc256000046)

[5.6 Using "Alfa-click" to pay for an order 54](#_Toc256000047)

[5.6.1 Short description of the PayByClick system 54](#_Toc256000048)

[5.6.2 Scenario of a payment for an order 54](#_Toc256000049)

[5.6.3 Testing a payment through "Alfa-click" 58](#_Toc256000050)

[5.7 Using UPOP to pay for an order 61](#_Toc256000051)

[5.7.1 Short description of the CUP system 61](#_Toc256000052)

[5.7.2 Scenario of a payment for an order 62](#_Toc256000053)

[5.7.3 Testing a payment through UPOP 66](#_Toc256000054)

[5.7.4 Refunds for orders paid through UPOP 72](#_Toc256000055)

[5.8 Payment using Apple Pay 72](#_Toc256000056)

[5.8.1 Merchant actions necessary to connect to Apple Pay 72](#_Toc256000057)

[5.8.2 Interaction scheme on a payment with Apple Pay 74](#_Toc256000058)

[5.8.3 Executing recurring payments through Apple Pay 76](#_Toc256000059)

[5.8.4 Apple Pay - links to the reference information 76](#_Toc256000060)

[5.9 Payment using Android Pay 79](#_Toc256000061)

[5.9.1 Preliminary actions 79](#_Toc256000062)

[5.9.2 Interaction schemes on a payment with Android Pay 81](#_Toc256000063)

[5.10 Payment using Samsung Pay 83](#_Toc256000064)

[5.10.1 Preliminary actions 83](#_Toc256000065)

[5.10.2 Scheme with the use of the mobile application 83](#_Toc256000066)

[6 Payment page 86](#_Toc256000067)

[6.1 Using a standard payment page 86](#_Toc256000068)

[6.2 Using your own payment page 89](#_Toc256000069)

[6.3 Requirements for the pages of the payment interface 92](#_Toc256000070)

[6.3.1 General requirements for the file containing the payment interface 92](#_Toc256000071)

[6.3.2 Requirements for the payment page 94](#_Toc256000072)

[6.3.3 Requirements for the payment page in case of passing a binding ID in the registration request 104](#_Toc256000073)

[6.3.4 Requirements for the errors page 107](#_Toc256000074)

[6.3.5 Requirements for the final page 108](#_Toc256000075)

[6.3.6 Requirements for the receipt page 111](#_Toc256000076)

[7 Plug-ins and code examples for the integration with the gateway 114](#_Toc256000077)

[8 Request specifications 124](#_Toc256000078)

[8.1 Web-Service interface 124](#_Toc256000079)

[8.1.1 Order registration request 124](#_Toc256000080)

[8.1.2 Registration request for orders with pre-authorization 131](#_Toc256000081)

[8.1.3 Order payment completion request 138](#_Toc256000082)

[8.1.4 Order status request 140](#_Toc256000083)

[8.1.5 Extended order status request 143](#_Toc256000084)

[8.1.6 Order payment cancellation request 151](#_Toc256000085)

[8.1.7 Order payment refund request 153](#_Toc256000086)

[8.1.8 Request for checking a card for 3D-Secure enrolment 155](#_Toc256000087)

[8.1.9 Request for adding additional parameters to an order 157](#_Toc256000088)

[8.1.10 Request for payments statistics for a period 159](#_Toc256000089)

[8.1.11 Request for a payment through an external payment network 166](#_Toc256000090)

[8.1.12 Request for processing a payment by a binding 169](#_Toc256000091)

[8.1.13 Request for a binding deactivation 173](#_Toc256000092)

[8.1.14 Request for a binding activation 174](#_Toc256000093)

[8.1.15 Request for changing the validity period of a binding 176](#_Toc256000094)

[8.1.16 Request for the list of bindings of a customer 178](#_Toc256000095)

[8.1.17 Request for the list of bindings of a bank card 180](#_Toc256000096)

[8.1.18 Request for adding a card to the list of SSL-cards 183](#_Toc256000097)

[8.1.19 Request for a payment through Apple Pay 185](#_Toc256000098)

[8.1.20 Request for executing recurring payments through Apple Pay 194](#_Toc256000099)

[8.1.21 Request for a payment through Android Pay 196](#_Toc256000100)

[8.1.22 Request for a payment through Samsung Pay 201](#_Toc256000101)

[8.2 REST interface 205](#_Toc256000102)

[8.2.1 Order registration request 205](#_Toc256000103)

[8.2.2 Registration request for orders with pre-authorization 211](#_Toc256000104)

[8.2.3 Order payment completion request 217](#_Toc256000105)

[8.2.4 Order status request 219](#_Toc256000106)

[8.2.5 Extended order status request 222](#_Toc256000107)

[8.2.6 Order payment cancellation request 229](#_Toc256000108)

[8.2.7 Order payment refund request 231](#_Toc256000109)

[8.2.8 Request for checking a card for enrolment to 3D-Secure 232](#_Toc256000110)

[8.2.9 Request for adding additional parameters to an order 234](#_Toc256000111)

[8.2.10 Request for payments statistics for a period 235](#_Toc256000112)

[8.2.11 Request for a payment through an external payment system 242](#_Toc256000113)

[8.2.12 Request for executing a payment by a binding 244](#_Toc256000114)

[8.2.13 Request for deactivation of a binding 247](#_Toc256000115)

[8.2.14 Request for activation of a binding 248](#_Toc256000116)

[8.2.15 Request for changing the validity period of a binding 249](#_Toc256000117)

[8.2.16 Request for the list of binding of a customer 251](#_Toc256000118)

[8.2.17 Request for the list of binding of a bank card 252](#_Toc256000119)

[8.2.18 Request for adding a card to the list of SSL-cards 254](#_Toc256000120)

[8.2.19 Request for a payment through Apple Pay 257](#_Toc256000121)

[8.2.20 Request for executing recurring payments through Apple Pay 266](#_Toc256000122)

[8.2.21 Request for a payment through Android Pay 269](#_Toc256000123)

[8.2.22 Request for a payment through Samsung Pay 273](#_Toc256000124)

[9 Test cards 278](#_Toc256000125)

[10 Appendix 1. External fee for payments 283](#_Toc256000126)

[10.1 Supplement to the description of the payment page 283](#_Toc256000127)

[10.2 Testing 283](#_Toc256000128)

[11 Appendix 2. Specification of additional fields for air-commerce payments and hotels booking 286](#_Toc256000129)

[11.1 Additional information passed in air-commerce 286](#_Toc256000130)

[11.2 Additional information passed on booking and paying for an hotel 289](#_Toc256000131)

[12 Appendix 3. Response codes - interpretation of actionCode (responses from the processing system) 292](#_Toc256000132)

* [Read before reading the documentation.](#scroll-bookmark-2)
* [Glossary](#scroll-bookmark-3)
* [Algorithm to connect to the Payment Gateway](#scroll-bookmark-4)
* [General description of the methods of interaction with the Payment Gateway](#scroll-bookmark-5)
  + [Payment mechanisms (one-phase and two-phase)](#scroll-bookmark-6)
  + [Schemes of interaction with the Payment Gateway](#scroll-bookmark-7)
  + [Interfaces of interaction with the Payment Gateway](#scroll-bookmark-8)
  + [Connection URLs](#scroll-bookmark-9)
* [Description of the interaction schemes](#scroll-bookmark-10)
  + [One-phase payment with specifying card data on the payment page](#scroll-bookmark-11)
    - [Scenario of a payment for an order](#scroll-bookmark-12)
    - [Cancellation of a payment for an order](#scroll-bookmark-13)
    - [Refund of a payment for an order](#scroll-bookmark-14)
    - [Check for enrolment of a card to 3D-Secure](#scroll-bookmark-15)
    - [Adding additional parameters to an order](#scroll-bookmark-16)
    - [Payment statistics for a certain period](#scroll-bookmark-17)
    - [Adding a card to the list of SSL-cards](#scroll-bookmark-18)
  + [Two-phase payment with specifying card data on the payment page](#scroll-bookmark-19)
    - [Scenario of a payment for an order](#scroll-bookmark-20)
    - [Cancellation of a payment for an order](#scroll-bookmark-21)
    - [Refund of a payment for an order](#scroll-bookmark-22)
    - [Check for enrolment of a card to 3D-Secure](#scroll-bookmark-23)
    - [Adding additional parameters to an order](#scroll-bookmark-24)
    - [Payment statistics for a certain period](#scroll-bookmark-25)
    - [Adding a card to the list of SSL-cards](#scroll-bookmark-26)
  + [Payment with the aid of a binding on the payment page](#scroll-bookmark-27)
    - [General description of the autocompletetion functionality on the payment page](#scroll-bookmark-28)
    - [Scenario of a payment for an order](#scroll-bookmark-29)
    - [Getting the bindings list of a customer](#scroll-bookmark-30)
    - [Getting the bindings list of a bank card](#scroll-bookmark-31)
    - [Deactivating/activating an existing binding](#scroll-bookmark-32)
    - [Changing the validity period of a binding](#scroll-bookmark-33)
    - [Adding a card to the list of SSL-cards](#scroll-bookmark-34)
  + [One-phase auto-payments](#scroll-bookmark-35)
    - [Scenario of executing an initial payment](#scroll-bookmark-36)
    - [Scenario of executing an auto-payment](#scroll-bookmark-37)
    - [Getting the bindings list of a customer](#scroll-bookmark-38)
    - [Deactivating/activating an existing binding](#scroll-bookmark-39)
    - [Changing the validity period of a binding](#scroll-bookmark-40)
    - [Adding a card to the list of SSL-cards](#scroll-bookmark-41)
  + [Two-phase auto-payments](#scroll-bookmark-42)
    - [Scenario of executing an initial payment](#scroll-bookmark-43)
    - [Scenario of executing an auto-payment](#scroll-bookmark-44)
    - [Getting the bindings list of a customer](#scroll-bookmark-45)
    - [Deactivating/activating an existing binding](#scroll-bookmark-46)
    - [Changing the validity period of a binding](#scroll-bookmark-47)
    - [Adding a card to the list of SSL-cards](#scroll-bookmark-48)
  + [Using "Alfa-click" to pay for an order](#scroll-bookmark-49)
    - [Short description of the PayByClick system](#scroll-bookmark-50)
    - [Scenario of a payment for an order](#scroll-bookmark-51)
      * [Using "Alfa-click" and e-commerce](#scroll-bookmark-52)
      * [Using solely "Alfa-click"](#scroll-bookmark-53)
    - [Testing a payment through "Alfa-click"](#scroll-bookmark-54)
  + [Using UPOP to pay for an order](#scroll-bookmark-55)
    - [Short description of the CUP system](#scroll-bookmark-56)
    - [Scenario of a payment for an order](#scroll-bookmark-57)
      * [Using UPOP and e-commerce](#scroll-bookmark-58)
      * [Using solely UPOP](#scroll-bookmark-59)
    - [Testing a payment through UPOP](#scroll-bookmark-60)
      * [Testing process](#scroll-bookmark-61)
      * [Test China UnionPay cards](#scroll-bookmark-62)
    - [Refunds for orders paid through UPOP](#scroll-bookmark-63)
  + [Payment using Apple Pay](#scroll-bookmark-64)
    - [Merchant actions necessary to connect to Apple Pay](#scroll-bookmark-65)
      * [Actions in the personal area of the Payment Gateway](#scroll-bookmark-66)
      * [Creating Merchant ID](#scroll-bookmark-67)
      * [Creating a certificate for Merchant ID](#scroll-bookmark-68)
    - [Interaction scheme on a payment with Apple Pay](#scroll-bookmark-69)
    - [Executing recurring payments through Apple Pay](#scroll-bookmark-70)
    - [Apple Pay - links to the reference information](#scroll-bookmark-71)
  + [Payment using Android Pay](#scroll-bookmark-72)
    - [Preliminary actions](#scroll-bookmark-73)
    - [Interaction schemes on a payment with Android Pay](#scroll-bookmark-74)
      * [Scheme with the use of the mobile application](#scroll-bookmark-75)
      * [Scheme with the use of the site adapted for mobile applications](#scroll-bookmark-76)
  + [Payment using Samsung Pay](#scroll-bookmark-77)
    - [Preliminary actions](#scroll-bookmark-78)
    - [Scheme with the use of the mobile application](#scroll-bookmark-79)
* [Payment page](#scroll-bookmark-80)
  + [Using a standard payment page](#scroll-bookmark-81)
  + [Using your own payment page](#scroll-bookmark-82)
  + [Requirements for the pages of the payment interface](#scroll-bookmark-83)
    - [General requirements for the file containing the payment interface](#scroll-bookmark-84)
    - [Requirements for the payment page](#scroll-bookmark-85)
      * [Page name](#scroll-bookmark-86)
      * [Page header](#scroll-bookmark-87)
      * [Page body](#scroll-bookmark-88)
        + [Mandatory elements on the payment page](#scroll-bookmark-89)
        + [Placing additional elements on the payment page](#scroll-bookmark-90)

[Displaying payment parameters](#scroll-bookmark-91)

[Displaying additional parameters of an order](#scroll-bookmark-92)

[Notifying a customer about an executed operation](#scroll-bookmark-93)

[Displaying the amount of the fee](#scroll-bookmark-94)

[Check box of saving the card data of a payment](#scroll-bookmark-95)

* + - * + [Card data validation](#scroll-bookmark-96)
    - [Requirements for the payment page in case of passing a binding ID in the registration request](#scroll-bookmark-97)
      * [Page name](#scroll-bookmark-98)
      * [Page header](#scroll-bookmark-99)
      * [Page body](#scroll-bookmark-100)
    - [Requirements for the errors page](#scroll-bookmark-101)
      * [Page name](#scroll-bookmark-102)
      * [Page header](#scroll-bookmark-103)
      * [Page body](#scroll-bookmark-104)
    - [Requirements for the final page](#scroll-bookmark-105)
      * [Page name](#scroll-bookmark-106)
      * [Page header](#scroll-bookmark-107)
      * [Page body](#scroll-bookmark-108)
    - [Requirements for the receipt page](#scroll-bookmark-109)
      * [Page name](#scroll-bookmark-110)
      * [Requirements for creation of a page](#scroll-bookmark-111)
* [Plug-ins and code examples for the integration with the gateway](#scroll-bookmark-112)
* [Request specifications](#scroll-bookmark-113)
  + [Web-Service interface](#scroll-bookmark-114)
    - [Order registration request](#scroll-bookmark-115)
    - [Registration request for orders with pre-authorization](#scroll-bookmark-116)
    - [Order payment completion request](#scroll-bookmark-117)
    - [Order status request](#scroll-bookmark-118)
    - [Extended order status request](#scroll-bookmark-119)
    - [Order payment cancellation request](#scroll-bookmark-120)
    - [Order payment refund request](#scroll-bookmark-121)
    - [Request for checking a card for 3D-Secure enrolment](#scroll-bookmark-122)
    - [Request for adding additional parameters to an order](#scroll-bookmark-123)
    - [Request for payments statistics for a period](#scroll-bookmark-124)
    - [Request for a payment through an external payment network](#scroll-bookmark-125)
    - [Request for processing a payment by a binding](#scroll-bookmark-126)
    - [Request for a binding deactivation](#scroll-bookmark-127)
    - [Request for a binding activation](#scroll-bookmark-128)
    - [Request for changing the validity period of a binding](#scroll-bookmark-129)
    - [Request for the list of bindings of a customer](#scroll-bookmark-130)
    - [Request for the list of bindings of a bank card](#scroll-bookmark-131)
    - [Request for adding a card to the list of SSL-cards](#scroll-bookmark-132)
    - [Request for a payment through Apple Pay](#scroll-bookmark-133)
    - [Request for executing recurring payments through Apple Pay](#scroll-bookmark-134)
    - [Request for a payment through Android Pay](#scroll-bookmark-135)
    - [Request for a payment through Samsung Pay](#scroll-bookmark-136)
  + [REST interface](#scroll-bookmark-137)
    - [Order registration request](#scroll-bookmark-138)
    - [Registration request for orders with pre-authorization](#scroll-bookmark-139)
    - [Order payment completion request](#scroll-bookmark-140)
    - [Order status request](#scroll-bookmark-141)
    - [Extended order status request](#scroll-bookmark-142)
    - [Order payment cancellation request](#scroll-bookmark-143)
    - [Order payment refund request](#scroll-bookmark-144)
    - [Request for checking a card for enrolment to 3D-Secure](#scroll-bookmark-145)
    - [Request for adding additional parameters to an order](#scroll-bookmark-146)
    - [Request for payments statistics for a period](#scroll-bookmark-147)
    - [Request for a payment through an external payment system](#scroll-bookmark-148)
    - [Request for executing a payment by a binding](#scroll-bookmark-149)
    - [Request for deactivation of a binding](#scroll-bookmark-150)
    - [Request for activation of a binding](#scroll-bookmark-151)
    - [Request for changing the validity period of a binding](#scroll-bookmark-152)
    - [Request for the list of binding of a customer](#scroll-bookmark-153)
    - [Request for the list of binding of a bank card](#scroll-bookmark-154)
    - [Request for adding a card to the list of SSL-cards](#scroll-bookmark-155)
    - [Request for a payment through Apple Pay](#scroll-bookmark-156)
    - [Request for executing recurring payments through Apple Pay](#scroll-bookmark-157)
    - [Request for a payment through Android Pay](#scroll-bookmark-158)
    - [Request for a payment through Samsung Pay](#scroll-bookmark-159)
* [Test cards](#scroll-bookmark-160)
* [Appendix 1. External fee for payments](#scroll-bookmark-161)
  + [Supplement to the description of the payment page](#scroll-bookmark-162)
  + [Testing](#scroll-bookmark-163)
* [Appendix 2. Specification of additional fields for air-commerce payments and hotels booking](#scroll-bookmark-164)
  + [Additional information passed in air-commerce](#scroll-bookmark-165)
  + [Additional information passed on booking and paying for an hotel](#scroll-bookmark-166)
* [Appendix 3. Response codes - interpretation of actionCode (responses from the processing system)](#scroll-bookmark-167)

Read before reading the documentation.

|  |
| --- |
| Any use of the functions or functionality of the system outside the scope of this document is at your own risk. |

Glossary

* *3-D Secure* – a technology of Visa IPS that enables additional authorization of a user of funds of the issuing bank.
* *ACS* – Access Control Server, an element of the 3-D Secure infrastructure that enables validation of a payer on the side of the issuing bank.
* *Merchant Plugin Interface (MPI)* – a technological component of 3-D Secure and SecureCode that can be placed on the payment network side or on the side of the merchant.
* *SecureCode* – a technology of MasterCard IPS that enables additional authorization of a user of funds of the issuing bank. Technologically, it is equal to 3-D Secure. In the text below, mentioning 3-D Secure implies also SecureCode.
* *Refund* - a partial or full refund of funds to a buyer's card in case of their refusal to receive the goods or services or in case of returning them. The refund operation is performed after the funds are debited from the buyer's account.
* *Bank card* – a card of an international payment system Visa or MasterCard.
* *Acquiring bank* – a bank that implements and uses the payment gateway.
* *Issuing bank* – a bank that has issued the card of a customer.
* *Two-phase payment* – an operation of paying for goods or services executed through the Internet with the use of bank cards and requiring additional confirmation. Two-phase mechanism of work enables splitting the process into checking whether the card is capable of paying (authorization) and debiting the money from the account (financial confirmation). On the first phase of a two-phase payment, a bank card is checked for its solvency and funds are put on hold on the customer's account.
* *Order* – an elementary entity in the system, it describes an order in a certain online store or its analog. Any order has its amount.
* *Store (merchant)* – a trade or service company selling goods or services through an Internet-site.
* *IPS* – an international payment system (for example, Visa or MasterCard).
* *One-phase payment* – a payment operation for goods or services executed through the Internet with the use of a bank card that does not require additional confirmation.
* *Reversal* – cancellation of a payment operation, removing from hold funds on the buyer's card. This operation is available within a limited time period, the exact duration should be found out in the bank.
* *Payer* – a person paying with their card for the Merchant's services in the Merchant's online store.
* *Payment form* – an HTML-page that is used by the customer to enter the payment details.
* *Payment details* – requisites used by the customer to pay for an order. Usually they represent a card number, expiration date, and CVC.
* *Payment gateway of the acquiring bank*  – an automatic system that allows *a store* to receive payments and to a customer to send payments through the Internet using bank cards.
* *Binding* – a correspondence between the payer and the payment details of a card (the card number, its validity period).

Algorithm to connect to the Payment Gateway

To connect to the payment gateway, an online store needs:

1. To receive logins and passwords to the test server from the Bank staff:

* + 1. A login with the "-api" suffix – to connect to the programming interface (API);
    2. A login with the "-operator" suffix – to work in the personal area through the web-interface.

2. To place the payment page on the test server. The following options are available:

* + - The store can use a standard page <https://web.rbsuat.com/ab/merchants/rbs/payment_ru.html> .
    - The store can use the standard page, having placed on it the store logo or footer or both. The description is in section [5.1 Using the standard payment page](#scroll-bookmark-168).
    - A merchant can create its own page. The detailed description of creation of a page is present in section [5.2. Using own payment page](#scroll-bookmark-169).

3. To implement integration according to this document. To simplify the work performed, plug-ins and code samples can be used for the integration with the gateway. See section [6 Plug-ins and code samples for the integration with the gateway](#scroll-bookmark-170).

4. To check the solution work using the test cards (they are listed in section [8. Test cards](#scroll-bookmark-171)) through:

* + 1. REST interface or Web-Service interface;
    2. the personal area in the administrative console.

For this check it is necessary:

* + 1. to process several orders successfully paid and unsuccessfully paid;
    2. to check that a correct status is displayed on the payment result page and to compare it with the status in the personal area;
    3. to process a payment completion (in case of using a two-phase scheme), cancellation and refund.

5. After testing, it is necessary to notify the bank that the integration is ready to work in the production environment. It is advisable to provide the address of the testing resource through which the Bank staff could process verification payments.

6. After the integration and payment page (in case it had been created by the store) have been successfully checked, under the condition of the contract has been signed, the store is provided with the credentials for connection to the production environment.

*Note:* Further, you can change the payment page on the production server at any time. To do this, it is necessary to download a new page to the test server, check its functionality and to send a request to the support service to change the page on the production server (usually the process takes no more than a day)

7. After receiving logins and passwords for the production server, it is necessary to process verification payments using a real card: to pay an order, cancel and (or) refund it.

General description of the methods of interaction with the Payment Gateway

Paying by a bank card for goods and services through the Internet can be performed with passing the complete card details to the payment gateway using bindings as well as using external payment systems.

Payment mechanisms (one-phase and two-phase)

A store can use the one-phase or two-phase mechanism of accepting a payment:

* *One-phase payment – an operation of* paying for goods or services processed through the Internet with the use of bank cards that does not require additional confirmation (the same request initiates at once putting on hold and debiting funds from a card).
* *Two-phase payment* – an operation of paying for goods or services processed through the Internet with the use of bank cards that does not require additional confirmation. Two-phase mechanism enables splitting the process into checking whether a bank card has enough funds for a payment (authorization) and debiting funds from the card (financial confirmation). The first request initiates a check for the card's solvency and puts an amount on hold on the customer's account; the second request initiates debiting the amount from the card.

Schemes of interaction with the Payment Gateway

Various connection schemes are available for a merchant to integrate with the payment gateway depending on the payment mechanism used (one-phase or two-phase) and on the means of payment. The present document represents the following connection schemas:

* Section [4.1 One-phase payment with specifying card data on the payment page](#scroll-bookmark-172) describes a scheme in which to pay an order a customer enters his or her card details on the payment page, from which the card details are passed to the payment gateway to process the payment.
* Section [4.2 Two-phase payment with specifying card data on the payment page](#scroll-bookmark-173) describes a scheme in which to pay an order a customer enters his or her card details on the payment page, from which the card details are passed to the payment gateway to process the two-phase payment.
* Section [4.3 Payment with the aid of a binding on the payment page](#scroll-bookmark-174) describes a scheme in which a customer card is associated with the customer identifier in the store. This enables offering to authorized users autocompletion of the card data fields on the payment page.
* A scheme described in section [4.4 One-phase auto-payments](#scroll-bookmark-175) assumes that a customer makes an initial payment on the payment page with an agreement to activate the "Auto-payment" service. After this, the store independently tracks the date when it is necessary to process a subsequent payment and initiates a payment without additionally entering the card data (without the participation of the customer).
* A scheme described in section [4.5 Two-phase auto-payments](#scroll-bookmark-176) assumes that a customer makes an initial payment on the payment page with an agreement to activate the "Auto-payment" service. After this, the store independently tracks the date when it is necessary to process a subsequent payment and initiates a payment without additionally entering the card data (without the participation of the customer).
* Section [4.6 Using "Alfa-click" to pay for an order](#scroll-bookmark-177) describes a payment scheme that is available only to customers of the "Alfa-click" Internet bank . With this scheme, entering payment details is performed externally, on the side of the PayByClick system.
* Section [4.7 Using UPOP to pay for an order](#scroll-bookmark-178) describes a payment scheme that is available only to holders of China UnionPay cards. With this scheme, entering payment details is performed externally, on the side of the China UnionPay system.
* Section [4.8 Payment using Apple Pay](#scroll-bookmark-64) describes a payment scheme that is available to owners of Apple mobile devices through the Apple Pay system.

Depending on the interaction scheme that a merchant uses, it is necessary to use a certain set of requests. The descriptions of the above mentioned scenarios specify at what step a request must be used and also contain links to sections with the request specifications.

The available interfaces of interaction with the payment gateway are present in the next section.

Interfaces of interaction with the Payment Gateway

A store to interact with the payment gateway can use one of these interfaces: the WebService interface or REST.

To authorize a call of the store to the payment gateway system, any request from the store must contain the store name and password received on the store registration in the system. The detailed information about calls authorization is provided below.:

* **Implementation of interaction though the interface using Web-Service:**

The description (WSDL) of the service is stored on the test server that is available without restrictions.

The values of the login and password are passed in the format described in the WS-Security specification, the authorization type is userName token. The header with an authorization of this type will look, for example, as follows:

|  |
| --- |
| <wsse:Security xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity- secext-1.0.xsd" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss- %20wssecurity-utility-1.0.xsd">   <wsse:UsernameToken wsu:Id="UsernameToken-87">   <wsse:Username>aa</wsse:Username>   <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-  token-profile-1.0#PasswordText">123456</wsse:Password>   </wsse:UsernameToken>   </wsse:Security> |

If the response to this request contains an errorCode = 0, it means that the request has been processed by the payment gateway without system errors. Meanwhile, errorCode does not display the order status.

To get the order status it is necessary to use the getOrderStatus or getOrderStatusExtended request (it is mandatory to implement one of these methods in the integration of a merchant and the payment gateway).

* **Implementation of interaction though the REST interface**

Interaction is implemented as HTTP-calls with the GET or POST methods to certain URLs. Parameters are passed as parameters of GET or POST requests, their values must be compatible with the URL (that is, URL-encoded).

The result of processing a request is returned as a JSON-object. For example: {"errorCode":"12","errorMessage":"Empty amount"}

The values of the login and password are passed in the following parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection |
| password | AN..30 | yes | Store password received on the connection |

All the text fields must be in the Unicode encoding (UTF-8).

Special characters in a REST request must be escaped in conformance with the URL-code. The table of symbols is stored at the following location: <http://web-developer.name/urlcode/>. For example, the password "qwe?rt%y" must be passed as "qwe%0Frt%25y".

If the response to this request contains an errorCode = 0, it means that the request has been processed by the payment gateway without system errors. Meanwhile, errorCode does not display the order status.

To get the order status, it is necessary to use the getOrderStatus.do or getOrderStatusExtende.do request (it is mandatory to implement one of these methods in the integration of a merchant and the payment gateway).

The connection URLs for the WebService and REST interfaces are present in the following section.

Connection URLs

When registering a merchant, the merchant representative is provided with a login/password couple that must be used in the protocols.

**The description of the test service (WSDL) is stored at the following address:**

https://web.rbsuat.com/ab/webservices/merchant-ws?wsdl

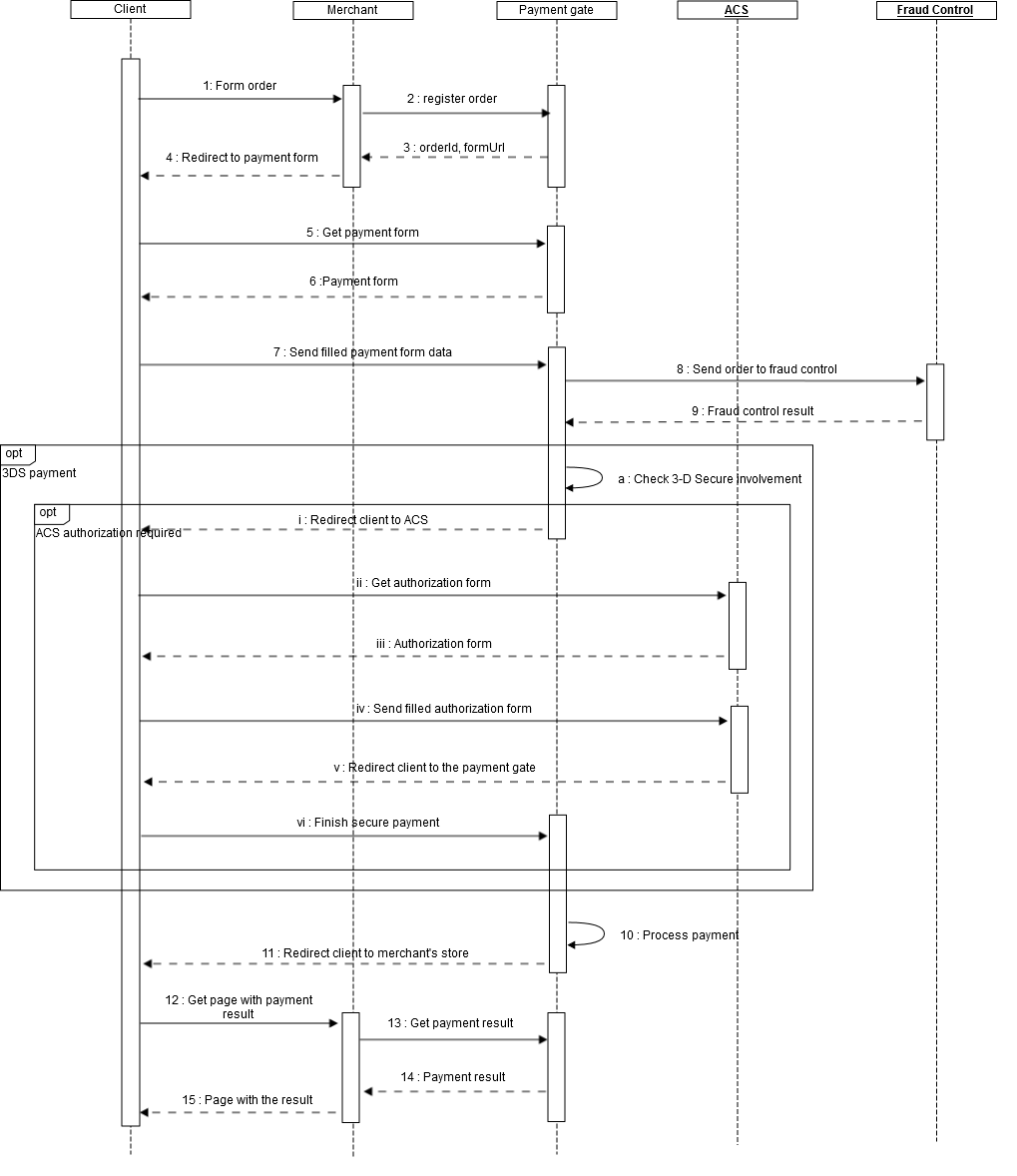
**The URL to access the REST methods:**

|  |  |
| --- | --- |
| Method name | URL-address |
| Order registration | https://web.rbsuat.com/ab/rest/register.do |
| Order registration with pre-authorization | https://web.rbsuat.com/ab/rest/registerPreAuth.do |
| Order payment completion request | https://web.rbsuat.com/ab/rest/deposit.do |
| Order status request | https://web.rbsuat.com/ab/rest/getOrderStatus.do |
| Extended order status request | https://web.rbsuat.com/ab/rest/getOrderStatusExtended.do |
| Order payment cancellation request | https://web.rbsuat.com/ab/rest/reverse.do |
| Order payment refund request | https://web.rbsuat.com/ab/rest/refund.do |
| Request for checking a card for enrolment to 3D-Secure | https://web.rbsuat.com/ab/rest/verifyEnrollment.do |
| Request for adding additional parameters to an order | https://web.rbsuat.com/ab/rest/addParams.do |
| Request for payments statistics for a period | https://web.rbsuat.com/ab/rest/getLastOrdersForMerchants.do |
| Request for a payment through an external payment network | https://web.rbsuat.com/ab/rest/paymentotherway.do |
| Request for adding a card number to the list of SSL-cards | https://web.rbsuat.com/ab/rest/updateSSLCardList.do |
| Request for processing a payment by a binding | https://web.rbsuat.com/ab/rest/paymentOrderBinding.do |
| Request for a binding deactivation | https://web.rbsuat.com/ab/rest/unBindCard.do |
| Request for a binding activation | https://web.rbsuat.com/ab/rest/bindCard.do |
| Request for changing the validity period of a binding | https://web.rbsuat.com/ab/rest/extendBinding.do |
| Request for the list of bindings of a customer | https://web.rbsuat.com/ab/rest/getBindings.do |
| Request for the list of bindings of a certain bank card | https://web.rbsuat.com/ab/rest/getBindingsByCardOrId.do |
| Request for a payment through Apple Pay | https://web.rbsuat.com/ab/applepay/payment.do |
| Request for a recurring payment | https://web.rbsuat.com/ab/recurrentPayment.do |
| Request for a payment through Android Pay | https://web.rbsuat.com/ab/android/payment.do |
| Request for a payment through Samsung Pay | https://web.rbsuat.com/ab/samsung/payment.do |

Description of the interaction schemes

One-phase payment with specifying card data on the payment page

### Scenario of a payment for an order



One-phase scheme of a payment by a card:

1. A customer creates an order at the merchant resource and selects the bank card payment method.
2. After the bank card payment method has been selected, a request for the order registration must be sent to the payment gateway. To register the order such parameters as the amount to be debited, order number in the store system, as well as the customer return URL, are used. The request specification is presented in sections:  
   - [7.1.1 Order registration request" (SOAP)](#scroll-bookmark-180) ,  
   - [7.2.1. Order registration request (REST)](#scroll-bookmark-181) .
3. In the response to the registration request, the payment gateway returns a unique identifier of the order in the payment system (in the orderId parameter) and a URL to which the customer is to be redirected to get the payment form (in the formUrl parameter).
4. The store system must pass to the browser of the redirect URL received from the payment gateway in the formUrl parameter as a response to the order registration request.
5. The browser of the customer opens the received URL.
6. The customer gets the payment form.
7. The customer fills in the form and sends the data to the payment gateway server.
8. The order details are passed to the fraud control system to determine the probability of fraud. The result of applying the rule to the order is adding to the order a fraud probability coefficient (fraud-score). Each rule has its fraud score that represents a number from 0 to 100. (If the total fraud score of an order for all rules applied to the order exceeds 100, such an order is considered fraud and a payment for it will be declined.)
9. The result of the order fraud check is returned to the payment gateway.

If according to the store settings the payment is to be processed through SSL, the following step of the scenario is to be performed (10).

If according to the store settings the payment is must be 3D-Secure, the following actions will be done:

* 1. The payment gateway checks the card for enrolment to 3-D Secure.

If the authorization on ACS is not required for this card, the following step of the scenario is performed (10).

If the authorization on ACS is required, the following actions will be executed:

* + 1. The gateway sends to the customer's browser the redirect URL to the ACS page of the issuing bank.
    2. The customer's browser requires from ACS the customer authorization form (each issuer implements this its own way)
    3. ACS sends to the customer's browser the authorization form.
    4. The customer fills in the authorization form and sends it to ACS.
    5. ACS handles the filled in form and, regardless of the result, passes to the browser the redirect URL to the payment gateway pages. Along with the URL the encrypted parameters of the authorization result are passed.
    6. The customer's browser requires the payment gateway page passing the encrypted authorization result parameters. If the authorization has completed successfully, the next step of the scenario is executed.

1. The payment gateway processes the payment (debiting funds from the account of the customer)
2. After the payment has been processed, the payment gateway passes to the customer's browser the URL for returning to the store page (earlier specified by the store on registering the order, see step 2).
3. The customer's browser requires the payment result page from the store.
4. The store system requires the order payment status from the payment gateway (by the order unique identifier in the payment system that has been received on the order registration in theorderId parameter).  
   The specification of a typical order status request is present in sections:  
   - [7.1.4. Order status request (SOAP)](#scroll-bookmark-182),  
   - [7.2.4. Order status request (REST)](#scroll-bookmark-183).  
   The specification of an extended order status request is present in sections:  
   - ["7.1..5 Extended order status request" (SOAP)](#scroll-bookmark-184),  
   - [7.2.5. Extended order status request (REST)](#scroll-bookmark-185).
5. The payment gateway returns the order payment status.
6. The store system passes to the customer's browser a page with the payment result – a successful payment or unsuccessful.

### Cancellation of a payment for an order

Cancellation of a payment for an order is available to merchants provided that they have the corresponding permissions (upon agreement with the bank). On one-phase payments, cancellation of a payment is available for orders in the "Complete" / "Deposited" status.

Cancellation of a payment for an order is executed in the standard way:

* Through the administrative console (the description is provided in the document "[Instruction on working with the console](https://rbs-develop.paymentgate.ru/wiki/pages/viewpage.action?pageId=12976308)", the section "Working with orders");
* Using API, by means of the REST or SOAP interfaces. The request specification is presented in sections:   
   - [7.1.6. Order payment cancellation request (SOAP)](#scroll-bookmark-186),   
   - [7.2.6. Order payment cancellation request (REST)](#scroll-bookmark-187).

In case of a successful cancellation an order will be turned from the "Complete"/"Deposited" status to "Cancelled"/"Reversed".

### Refund of a payment for an order

A full or partial refund for paid orders (in the "Complete"/"Deposited"status) is to be processed in the standard way:

* Through the administrative console (the description is provided in the document "[Instruction on working with the console](https://rbs-develop.paymentgate.ru/wiki/pages/viewpage.action?pageId=12976308)", the section "Working with orders");
* Using API, by means of the SOAP or REST interface. The request specification is presented in sections:  
   - [7.1.7. Request for a refund of an order payment funds (SOAP)](#scroll-bookmark-188),  
   - [7.2.7. Request for a refund of an order payment funds (REST)](#scroll-bookmark-189).

After a refund request sent in one of the above mentioned ways has been received in RBS, RBS returns the specified amount to the account of the customer.

### Check for enrolment of a card to 3D-Secure

If necessary, the system allows a store to independently check a bank card for enrolment to 3-D Secure. This can be done using API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.8. Request for checking a card for enrolment to 3D-Secure (SOAP)](#scroll-bookmark-190),  
 - [7.2.8. Request for checking a card for enrolment to 3D-Secure (REST)](#scroll-bookmark-191).

### Adding additional parameters to an order

In case of necessity, the system enables adding additional parameters to an order. This can be done using API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.9. Request for adding additional parameters to an order (SOAP)](#scroll-bookmark-192),  
 - [7.2.9. Request for adding additional parameters to an order (REST)](#scroll-bookmark-193).

### Payment statistics for a certain period

The system allows you to form payment statistics for a certain period using API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.10. Request for payment statistics for a period (SOAP)](#scroll-bookmark-194),  
 - [7.2.10. Request for payment statistics for a period (REST)](#scroll-bookmark-195).

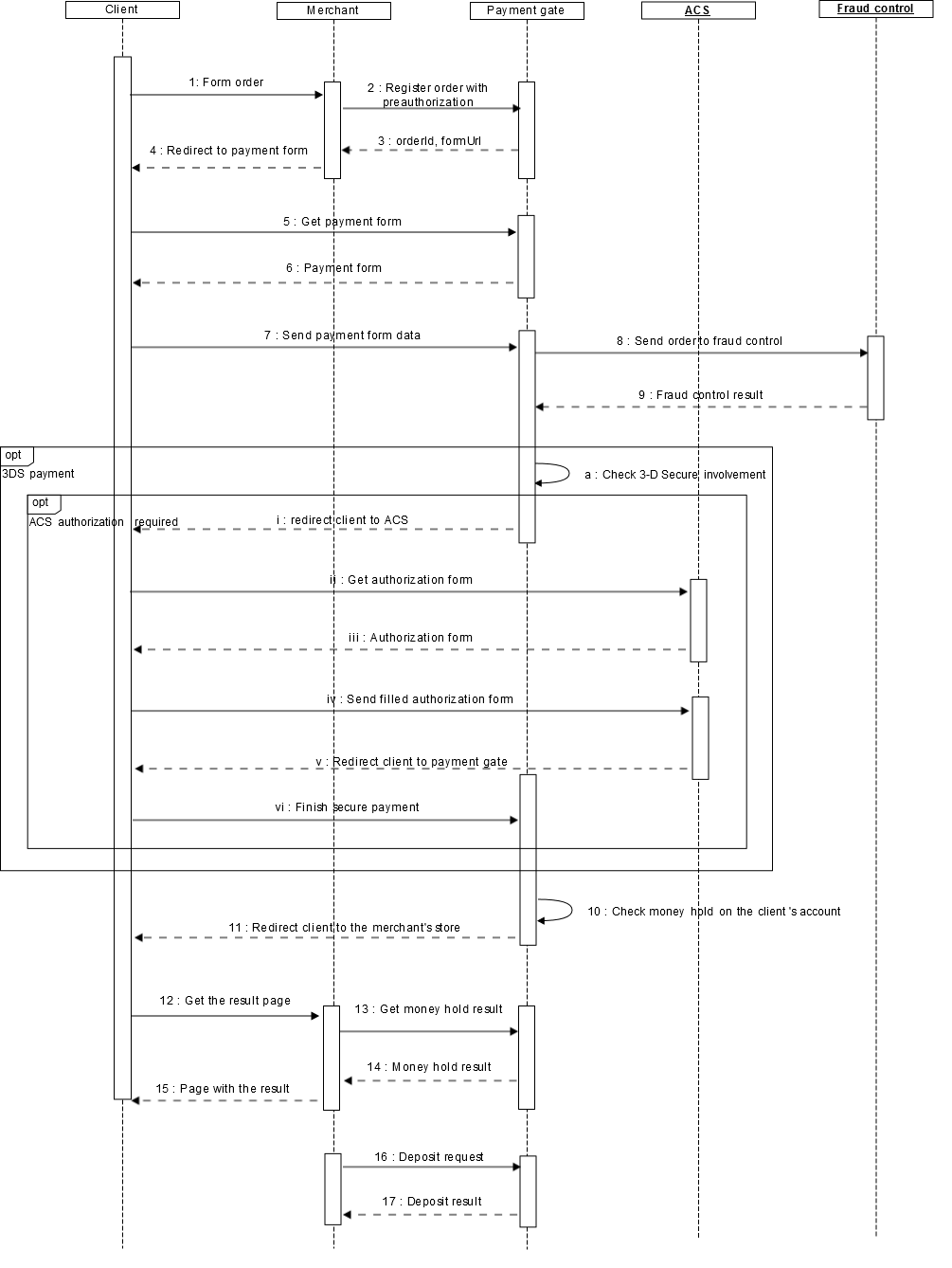
### Adding a card to the list of SSL-cards

Upon agreement with the bank, a merchant can use the method for adding the number of a card that has been used on a payment attempt to the list of SSL-cards:

- [7.1.18. Request for adding a card to the list of SSL-cards (SOAP)](#scroll-bookmark-196),  
 - [7.2.18. Request for adding a card to the list of SSL-cards (REST)](#scroll-bookmark-197).

Two-phase payment with specifying card data on the payment page

### Scenario of a payment for an order



Two-phase scheme for a payment by a card:

1. A customer creates an order at the merchant resource and selects the bank card payment method.
2. After the bank card method of payment has been selected, the order registration request with pre-authorization is to be sent to the payment gateway. To register the order such parameters as the amount to be debited, order number in the store system, as well as the customer return URL, are used. The request specification is presented in sections:  
   - [7.1.2. Request for registering an order with pre-authorization (SOAP)](#scroll-bookmark-198) ,  
   - [7.2.2. Request for registering an order with pre-authorization (REST)](#scroll-bookmark-199) .
3. In the response to the registration request, the payment gateway returns a unique identifier of the order in the payment system (in the orderId parameter) and a URL to which the customer is to be redirected to get the payment form (in the formUrl parameter).
4. The store system must pass to the browser of the redirect URL received from the payment gateway in the formUrl parameter as a response to the order registration request.
5. The browser of the customer opens the received URL.
6. The customer gets the payment form.
7. The customer fills in the form with the card details and sends the data to the payment gateway server.
8. The order details are passed to the fraud control system to determine the probability of fraud. The result of applying the rule to the order is adding to the order a fraud probability coefficient (fraud-score). Each rule has its fraud score that represents a number from 0 to 100. (If the total fraud score of an order for all rules applied to the order exceeds 100, such an order is considered fraud and a payment for it will be declined.)
9. The result of the order fraud check is returned to the payment gateway.

If according to the store settings the payment is to be processed through SSL, the following step of the scenario is to be performed (10).

If according to the store settings the payment is must be 3D-Secure, the following actions will be done:

* 1. The payment gateway performs a check by the card number to define whether the 3-D Secure technology is to be applies on processing the payment.  
     If authorization on ACS is not required for the given card, the next step of the scenario is to be executed (10).  
     If authorization on ACS is required, the following actions are to be done:  
     1. The gateway sends to the customer's browser the redirect URL to the ACS page of the issuing bank.
     2. The customer's browser requires from ACS the customer authorization form (each issuer implements this its own way)
     3. ACS sends to the customer's browser the authorization form.
     4. The customer fills in the form and sends it to ACS.
     5. ACS handles the filled in form and, regardless of the result, passes to the browser the redirect URL to the payment gateway pages. Along with the URL the encrypted parameters of the authorization result are passed.
     6. The customer's browser requires the payment gateway page passing the encrypted authorization result parameters. If the authorization has completed successfully, the next step of the scenario is executed.

1. The payment gateway performs authorization of the payment (puts on hold funds on the card account of the customer).
2. After the payment has been processed, the payment gateway passes to the customer's browser the URL for returning to the store page (earlier specified by the store on registering the order, see step 2).
3. The customer's browser requires the payment result page from the store.
4. The store system requires the order payment status from the payment gateway (by the order unique identifier in the payment system that has been received on the order registration in theorderId parameter).  
   The specification of a typical order status request is present in sections:  
   - [7.1.4. Order status request (SOAP)](#scroll-bookmark-182),  
   - ["7.2.4 Order status request (REST)](#scroll-bookmark-183).  
   The specification of an extended order status request is present in sections:  
   - [7.1.5. Extended order status request (SOAP)](#scroll-bookmark-184),  
   - [7.2.5. Extended order status request (REST)](#scroll-bookmark-185).
5. The payment gateway returns the order payment status.
6. The store system passes to the customer's browser a page with the payment result – a successful payment or unsuccessful.
7. To debit funds from the customer's account, the store needs to send to the payment gateway the order completion request. The request specification is present in sections:  
    - [7.1.3 Order payment completion request (SOAP)](#scroll-bookmark-200),  
    - [7.2.3. Order payment completion request (REST)](#scroll-bookmark-201).
8. The payment gateway returns the result of processing the request. The order status is not returned. To receive the order status, it is necessary to send to the gateway the corresponding request, as it is described in step 13.

### Cancellation of a payment for an order

Cancellation of a payment for an order is available to merchants provided that they have the corresponding permissions (upon agreement with the bank). On two-phase payments, a payment cancellation can be processed for orders in the "Approved" status.

Cancellation of a payment for an order is executed in the standard way:

* Through the administrative console (the description is provided in the document "[Instruction on working with the console](https://rbs-develop.paymentgate.ru/wiki/pages/viewpage.action?pageId=12976308)", the section "Working with orders");
* Using API, by means of the REST or SOAP interfaces. The request specification is presented in sections:   
   - [7.1.6. Order payment cancellation request (SOAP)](#scroll-bookmark-186),   
   - [7.2.6. Order payment cancellation request (REST)](#scroll-bookmark-187).

In case of a successful operation of an order cancellation, the order will be turned from the "Approved" status to "Reversed".

### Refund of a payment for an order

A full or partial refund for paid orders (orders in the "Deposited" status) is performed in the standard way:

* Through the administrative console (the description is provided in the document "[Instruction on working with the console](https://rbs-develop.paymentgate.ru/wiki/pages/viewpage.action?pageId=12976308)", the section "Working with orders");
* Using API, by means of the REST or SOAP interfaces. The request specification is presented in sections:  
   - [7.1.7. Request for a refund of an order payment funds (SOAP)](#scroll-bookmark-188),  
   - [7.2.7. Request for a refund of an order payment funds (REST)](#scroll-bookmark-189).

After a refund request sent in one of the above mentioned ways has been received in RBS, RBS returns the specified amount to the account of the customer.

### Check for enrolment of a card to 3D-Secure

If necessary, the system allows a store to independently check a bank card for enrolment to 3-D Secure. This can be done using API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.8. Request for checking a card for enrolment to 3D-Secure (SOAP)](#scroll-bookmark-190),  
 - [7.2.8. Request for checking a card for enrolment to 3D-Secure (REST)](#scroll-bookmark-191).

### Adding additional parameters to an order

In case of necessity, the system enables adding additional parameters to an order. This can be done using API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.9. Request for adding additional parameters to an order (SOAP)](#scroll-bookmark-192),  
 - [7.2.9. Request for adding additional parameters to an order (REST)](#scroll-bookmark-193).

### Payment statistics for a certain period

The system allows you to form payment statistics for a certain period using API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.10. Request for payment statistics for a period (SOAP)](#scroll-bookmark-194),  
 - [7.2.10. Request for payment statistics for a period (REST)](#scroll-bookmark-195).

### Adding a card to the list of SSL-cards

Upon agreement with the bank, a merchant can use the method for adding the number of a card that has been used on a payment attempt to the list of SSL-cards:

- [7.1.18. Request for adding a card to the list of SSL-cards (SOAP)](#scroll-bookmark-196),  
 - [7.2.18. Request for adding a card to the list of SSL-cards (REST)](#scroll-bookmark-197).

Payment with the aid of a binding on the payment page

### General description of the autocompletetion functionality on the payment page

The given functionality is used to associate a card number with the customer ID in the store system (for example, with the login).

If after an authorization at the store site and a successful order payment a customer again places an order at that site, on redirecting to the payment page, he or she will be offered autocompletion of the card data, excluding CVC/ CVV.

If for a merchant it is planned to use the bindings functionality, the payment page can contain the form for selection of a binding to pay for an order. The form design must meet the following conditions:

* The form must have an identifier id="formBinding".
* By default the form must be hidden with the use of the CSS property "display: none;".
* The form must contain a drop-down list for the selection of a binding with binding names name="bindingId".
* The drop-down list must contain one option of choice: <option value="" selected="selected">other</option>, upon selection of which the customer makes a usual payment by a card, without using the binding functionality.
* The form must contain a field for entering СVC/CVV with the name name="cvc".
* The form must contain the "Pay" button: <input value="Оплатить" type="button" id="buttonBindingPayment"> with the id="buttonBindingPayment" identifier.
* The CVC/CVV entering field and the "Pay" button must be framed by elements with the class="rbs\_hidden"class. On selecting an option of a payment without the binding functionality, these elements will be hidden by setting the CSS property "display: none;".

Form examples:

|  |
| --- |
| <form action="" id="formBinding" style="display: none;">   <table cellpadding="10">   <tbody>   <tr valign="TOP">   <td valign="top" width="50%" align="right">   <span>Select card:</span>   </td>   <td valign="top">   <select name="bindingId">   <option value="" selected="selected">other</option>   </select>   </td>   </tr>   <tr class="rbs\_hidden">   <td align="right">   <span>Введите CVC2/CVV2/CID код :</span><br>(on the back side of the card)   </td>   <td>   <input name="cvc" maxlength="4" type="password" autocomplete="off" />   </td>   </tr>   <tr class="rbs\_hidden">   <td> </td>   <td valign="top" >   <input value="Pay" type="button" id="buttonBindingPayment">   </td>   </tr>   </tbody>   </table>   </form> |

### Scenario of a payment for an order



1. A customer creates an order at the merchant resource and selects the bank card payment method.
2. After the bank card payment method has been selected, an order registration request is to be sent to the payment gateway with a unique identifier of the customer mandatorily passed in the clientId parameter. For the order registration also such parameters are used as the amount to be debited, order number in the store system, the customer return URL. The request specification is presented in sections:  
   - [7.1.1 Order registration request (SOAP)](#scroll-bookmark-180),  
   - [7.2.1. Order registration request (REST)](#scroll-bookmark-181) .
3. In a response to the order registration, the payment gateway returns a unique identifier on the order in the payment system (in the orderId parameter) and the URL, to which the customer is to be redirected to get a payment form (in the formUrl response parameter).
4. The store system must pass to the browser of the redirect URL received from the payment gateway in the formUrl parameter as a response to the order registration request.
5. The browser of the customer opens the received URL.
6. The customer gets the payment form.
7. If for the given clientId a binding has not been created yet, the customer fills in the received form with the card details and checks the "Remember this card data" box. After that, the customer sends the data to the payment gateway server.

If for a given clientId one or several associated cards exist, they are displayed in the drop-down list of the PAN field. The customer selects the necessary card (there is also a possibility to enter the data of a new card). After that, the customer sends the data to the payment gateway server.

1. The order details are passed to the fraud control system to determine the probability of fraud. The result of applying the rule to the order is adding to the order a fraud probability coefficient (fraud-score). Each rule has its fraud score that represents a number from 0 to 100. (If the total fraud score of an order for all rules applied to the order exceeds 100, such an order is considered fraud and a payment for it will be declined.)
2. The result of the order fraud check is returned to the payment gateway.

If the store settings require processing an SSL-payment, the next step of the scenario is to be executed (10).

If a payment according to the store settings must be 3D-Secure, the following actions will be performed:

* 1. Having received the payment details, the payment gateway checks by the card number whether it is necessary to use the 3-D Secure technology on processing the payment.

If using the 3-D Secure technology is not required, the next step of the scenario is to be executed (10).

If the payment must be 3-D Secure, the following actions will be performed:

* + 1. The gateway sends to the customer's browser the redirect URL to the ACS page of the issuing bank.
    2. The customer's browser requires from ACS the customer authorization form (each issuer implements this its own way)
    3. ACS sends to the customer's browser the authorization form.
    4. The customer fills in the authorization form and sends it to ACS.
    5. ACS handles the filled in form and, regardless of the result, passes to the browser the redirect URL to the payment gateway pages. Along with the URL the encrypted parameters of the authorization result are passed.
    6. The customer's browser requires the payment gateway page passing the encrypted authorization result parameters. If the authorization has completed successfully, the next step of the scenario is executed.

1. The payment gateway processes the payment (debiting funds from the account of the customer)
2. After the payment has been processed, the payment gateway passes to the customer's browser the URL for returning to the store page (earlier specified by the store on registering the order, see step 2).
3. The customer's browser requires the payment result page from the store.
4. The store system requires the order payment status from the payment gateway (by the order unique identifier in the payment system that has been received on the order registration in theorderId parameter).  
   The specification of a typical order status request is present in sections:  
   - [7.1.4. Order status request (SOAP)](#scroll-bookmark-182),  
   - [7.2.4. Order status request (REST)](#scroll-bookmark-183).  
   The specification of an extended order status request is present in sections:  
   - [7.1.5. Extended order status request (SOAP)](#scroll-bookmark-184),  
   - [7.2.5. Extended order status request (REST)](#scroll-bookmark-185).
5. The payment gateway returns the order payment status.
6. The store system passes to the customer's browser a page with the payment result – a successful payment or unsuccessful.

### Getting the bindings list of a customer

If at step 5 of the scenario the customer has entered the details of a new card on the payment page and checked the "Remember this card data" box, in case of a successful payment, a unique identifier of the binding in created on the payment gateway side. A store can get the list of identifiers of the existing binding for a certain clientId by sending the corresponding request to the payment gateway. The request specification is presented in sections:

- [7.1.16. Request for the list of bindings of a customer (SOAP)](#scroll-bookmark-202)

- [7.2.16. Request for the list of bindings of a customer (REST)](#scroll-bookmark-203)

### Getting the bindings list of a bank card

Provided that a store has the corresponding permissions, it can get the list of all bindings that relate to a certain bank card. This can be done by a card number or by a known binding identifier. The specification of the request is present in sections:  
 - [7.1.17. Request for the list of bindings of a bank card (SOAP)](#scroll-bookmark-204),  
 - [7.2.17. Request for the list of bindings of a bank card (REST)](#scroll-bookmark-205).

### Deactivating/activating an existing binding

If necessary, the system allows stores to deactivate existing bindings with the use of API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.13. Request for deactivation of a binding (SOAP)](#scroll-bookmark-206),  
 - [7.2.13. Request for deactivation of a binding (REST)](#scroll-bookmark-207).

Further a deactivated binding can be activated again. To do so, a store needs to send to the payment gateway the corresponding request. The request specification is present in sections:  
 - [7.1.14. Request for activation of a binding (SOAP)](#scroll-bookmark-208),  
 - [7.2.14. Request for activation of a binding (REST)](#scroll-bookmark-209) .

### Changing the validity period of a binding

Changing the validity period of a binding may be required in the case of re-issuing a customer's card. It can be done with the use of API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.15. Request for changing the validity period of a binding (SOAP)](#scroll-bookmark-210),  
 - [7.2.15. Request for changing the validity period of a binding (REST)](#scroll-bookmark-211).

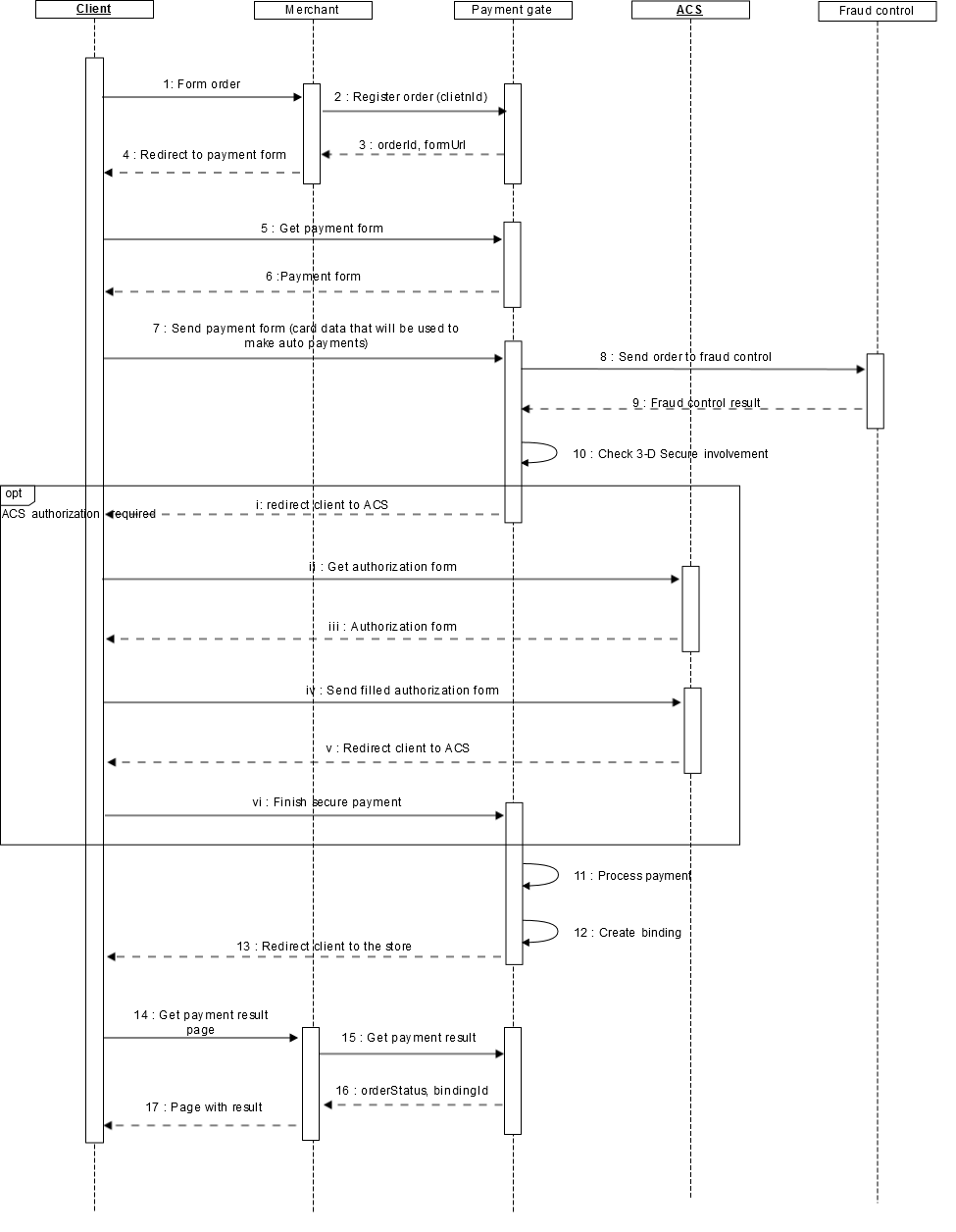
### Adding a card to the list of SSL-cards

Upon agreement with the bank, a store can use the method for adding a card that has been debited (or ought to be debited) to the list of SSL-cards:

- [7.1.18. Request for adding a card to the list of SSL-cards (SOAP)](#scroll-bookmark-196),  
 - [7.2.18. Request for adding a card to the list of SSL-cards (REST)](#scroll-bookmark-197).

One-phase auto-payments

### Scenario of executing an initial payment



1. The customer creates an order on the store resource and selects the bank card payment method.
2. After the bank card payment method has been selected, an order registration request is to be sent to the payment gateway with a unique identifier of the customer mandatorily passed in the clientId parameter. For the order registration also such parameters are used as the amount to be debited, order number in the store system, the customer return URL. The request specification is presented in sections:  
   - [7.1.1 Order registration request (SOAP)](#scroll-bookmark-180),  
   - [7.2.1. Order registration request (REST)](#scroll-bookmark-181) .
3. In a response to the order registration, the payment gateway returns a unique identifier on the order in the payment system (in the orderId parameter) and the URL, to which the customer is to be redirected to get a payment form (in the formUrl response parameter).
4. The store system must pass to the browser of the redirect URL received from the payment gateway in the formUrl parameter as a response to the order registration request.
5. The browser of the customer opens the received URL.
6. The customer gets the payment form.
7. The customer fills in the received form with the card details and sends the data to the payment gateway server.
8. The order details are passed to the fraud control system to determine the probability of fraud. The result of applying the rule to the order is adding to the order a fraud probability coefficient (fraud-score). Each rule has its fraud score that represents a number from 0 to 100. (If the total fraud score of an order for all rules applied to the order exceeds 100, such an order is considered fraud and a payment for it will be declined.)
9. The result of the order fraud check is returned to the payment gateway.
10. The payment gateway checks the card for enrolment to 3-D Secure.

If authorization on ACS is not required, the next step of the scenario is to be executed (11).

If the authorization on ACS is required, the following actions will be executed:

* + 1. The gateway sends to the customer's browser the redirect URL to the ACS page of the issuing bank.
    2. The customer's browser requires the user authorization form from ACS (each issuing bank implements this in its own way).
    3. ACS sends to the customer's browser the authorization form.
    4. The customer fills in the authorization form and sends it to ACS.
    5. ACS handles the filled in form and, regardless of the result, passes to the browser the redirect URL to the payment gateway pages. Along with the URL the encrypted parameters of the authorization result are passed.
    6. The customer's browser requires the payment gateway page passing the encrypted authorization result parameters. If the authorization has completed successfully, the next step of the scenario is executed.

1. The payment gateway processes the payment (debiting funds from the customer's account).
2. In case of a successful payment, a binding is to be created in the payment gateway (the details of the card that has been used for the payment are associated with the customer identifier passed at step 2 in the clientId parameter).
3. After the payment has been processed, the payment gateway passes to the customer's browser the URL for returning to the store page (earlier specified by the store on registering the order, see step 2).
4. The customer's browser requires the payment result page from the store.
5. The store system requests the status of an order from the payment gateway (by the order unique identifier in the store system that has been received at step 3 in the orderId parameter).  
   The specification of a typical order status request is present in sections:  
   - [7.1.4. Order status request (SOAP)](#scroll-bookmark-182),  
   - [7.2.4. Order status request (REST)](#scroll-bookmark-183).  
   The specification of an extended order status request is present in sections:  
   - [7.1.5. Extended order status request (SOAP)](#scroll-bookmark-184),  
   - [7.2.5. Extended order status request (REST)](#scroll-bookmark-185).
6. The payment gateway returns the order payment status (in the orderStatus parameter) along with the unique identifier of the binding (in the bindingId parameter).
7. The store system passes to the customer's browser the page with the payment result: whether the payment is successful or not.

After the initial payment has been successfully processed, the store on its side enable the "Auto-payment" service for the customer (defines the date and amount of debiting for the given customer). Further the store independently tracks the date of a subsequent payment and initiates a payment by the binding identifier.

### Scenario of executing an auto-payment

On the date of a subsequent payment, the store initiates a payment according to the following scenario:

1. An order registration request that mandatory passes the unique identifier in the clientId parameter must be sent to the payment gateway. For the order registration also such parameters are used as the amount to be debited, order number in the store system, the customer return URL. The request specification is presented in sections:  
   - [7.1.1 Order registration request (SOAP)](#scroll-bookmark-180),  
   - [7.2.1. Order registration request (REST)](#scroll-bookmark-181) .
2. In the response to the request, the payment gateway returns the unique identifier of the order in the payment system (as orderId).
3. The store sends the order payment request with the use of a binding, passing the number of the order in the payment system received at the previous step and the binding identifier received after the initial payment has been processed. The request specification is presented in sections:  
   - [7.1.12 Request for processing an order by a binding" (SOAP)](#scroll-bookmark-212),  
   - [7.2.12 Request for processing an order by a binding (REST)](#scroll-bookmark-213).
4. The payment gateway processes the payment (debiting funds from the customer's account).
5. The store system requests from the payment gateway the order status (by the unique identifier of the order in the payment system that was received at step 2 as orderId).  
   The specification of a typical order status request is present in sections:  
   - [7.1.4. Order status request (SOAP)](#scroll-bookmark-182),  
   - [7.2.4. Order status request (REST)](#scroll-bookmark-183).  
   The specification of an extended order status request is present in sections:  
   - [7.1.5. Extended order status request (SOAP)](#scroll-bookmark-184),  
   - [7.2.5. Extended order status request (REST)](#scroll-bookmark-185).
6. The payment gateway returns the payment status (in the orderStatus parameter).

### Getting the bindings list of a customer

The store can get the list of identifiers of existing bindings for a certain clientId by sending the corresponding request to the payment gateway. The request specification is presented in sections:

- [7.1.16. Request for the list of bindings of a customer (SOAP)](#scroll-bookmark-202)

- [7.2.16. Request for the list of bindings of a customer (REST)](#scroll-bookmark-203)

### Deactivating/activating an existing binding

If necessary, the system allows stores to deactivate existing bindings with the use of API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.13. Request for deactivation of a binding (SOAP)](#scroll-bookmark-206),  
 - [7.2.13. Request for deactivation of a binding (REST)](#scroll-bookmark-207).

Further a deactivated binding can be activated again. To do so, a store needs to send to the payment gateway the corresponding request. The request specification is present in sections:  
 - [7.1.14. Request for activation of a binding (SOAP)](#scroll-bookmark-208),  
 - [7.2.14. Request for activation of a binding (REST)](#scroll-bookmark-209) .

### Changing the validity period of a binding

Changing the validity period of a binding may be required in the case of re-issuing a customer's card. It can be done with the use of API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.15. Request for changing the validity period of a binding (SOAP)](#scroll-bookmark-210),  
 - [7.2.15. Request for changing the validity period of a binding (REST)](#scroll-bookmark-211).

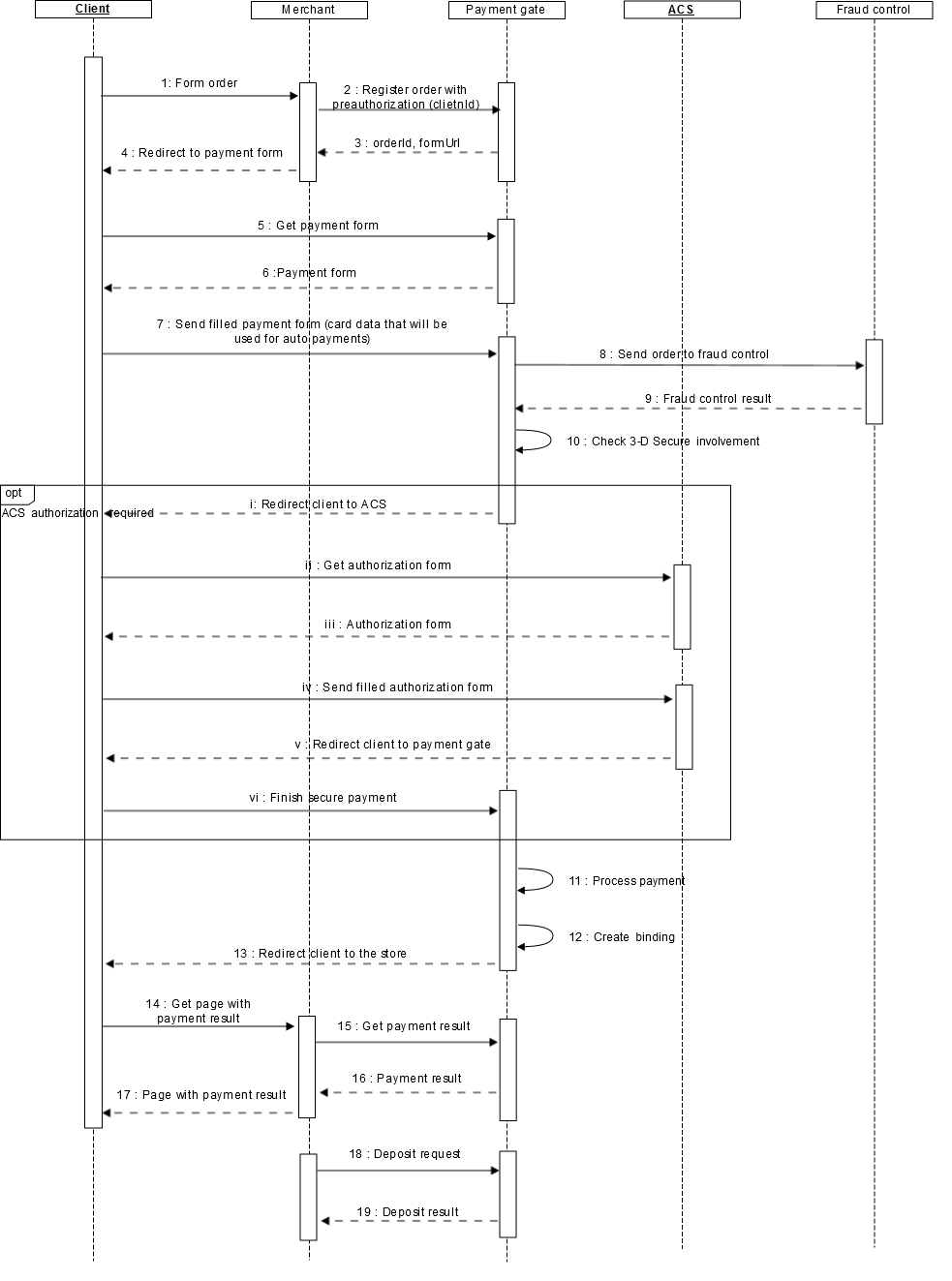
### Adding a card to the list of SSL-cards

Upon agreement with the bank, a merchant can use the method for adding the number of a card that has been used on a payment attempt to the list of SSL-cards:

- [7.1.18. Request for adding a card to the list of SSL-cards (SOAP)](#scroll-bookmark-196),  
 - [7.2.18. Request for adding a card to the list of SSL-cards (REST)](#scroll-bookmark-197).

Two-phase auto-payments

### Scenario of executing an initial payment



1. A customer creates an order at the merchant resource and selects the bank card payment method.
2. After the bank card payment method has been selected, a request for a preliminary order registration must be sent to the payment gateway. The requests must mandatory pass the unique identifier of the customer in the clientId parameter. For the order registration also such parameters are used as the amount to be debited, order number in the store system, the customer return URL. The request specification is presented in sections:  
   - [7.1.2. Request for registering an order with pre-authorization (SOAP)](#scroll-bookmark-198) ,  
   - [7.2.2. Request for registering an order with pre-authorization (REST)](#scroll-bookmark-199) .
3. In a response to the order registration, the payment gateway returns a unique identifier on the order in the payment system (in the orderId parameter) and the URL, to which the customer is to be redirected to get a payment form (in the formUrl response parameter).
4. The store system must pass to the browser of the redirect URL received from the payment gateway in the formUrl parameter as a response to the order registration request.
5. The customer's browser opens the URL.
6. The customer receives the payment form.
7. The customer fills in the form with the card details and sends the data to the payment gateway server.
8. The order details are passed to the fraud control system to determine the probability of fraud. The result of applying the rule to the order is adding to the order a fraud probability coefficient (fraud-score). Each rule has its fraud score that represents a number from 0 to 100. (If the total fraud score of an order for all rules applied to the order exceeds 100, such an order is considered fraud and a payment for it will be declined.)
9. The result of the order fraud check is returned to the payment gateway.
10. Having received the payment details, the payment gateway checks the card for enrolment to 3-D Secure.

If authorization on ACS is not required, the next step of the scenario is to be executed (11).

If authorization on ACS is required, the following actions will take place:

* + 1. The gateway sends to the customer's browser the redirect URL to the ACS page of the issuing bank.
    2. The customer's browser requires from ACS the customer authorization form (each issuer implements this its own way)
    3. ACS sends to the customer's browser the authorization form.
    4. The customer fills in the authorization form and sends it to ACS.
    5. ACS handles the filled in form and, regardless of the result, passes to the browser the redirect URL to the payment gateway pages. Along with the URL the encrypted parameters of the authorization result are passed.
    6. The customer's browser requires the payment gateway page passing the encrypted authorization result parameters. If the authorization has completed successfully, the next step of the scenario is executed.

1. The payment gateway processes the payment (puts on hold funds on the customer's card account).
2. In case of a successful putting on hold the amount on the card, a binding will be created (the details of the card that has been used for the payment will be associated with the customer identifier passes at step 2 as clientId).
3. After the payment has been processed, the payment gateway passes to the customer's browser the URL for returning to the store page (earlier specified by the store on registering the order, see step 2).
4. The customer's browser requires the payment result page from the store.
5. The store system requests the status of an order from the payment gateway (by the order unique identifier in the store system that has been received at step 3 in the orderId parameter).  
   The specification of a typical order status request is present in sections:  
   - [7.1.4. Order status request (SOAP)](#scroll-bookmark-182),  
   - [7.2.4. Order status request (REST)](#scroll-bookmark-183).  
   The specification of an extended order status request is present in sections:  
   - [7.1.5. Extended order status request (SOAP)](#scroll-bookmark-184),  
   - [7.2.5. Extended order status request (REST)](#scroll-bookmark-185).
6. The payment gateway returns the order payment status (in the orderStatus parameter) along with the unique identifier of the binding (in the bindingId parameter).
7. The store system passes to the customer's browser a page with the payment result – a successful payment or unsuccessful.
8. For the amount put on hold to be debited from the customer's account, the store needs to send to the payment gateway the payment completion request. The request specification is present in sections:  
    - [7.1.3 Order payment completion request (SOAP)](#scroll-bookmark-200),  
    - [7.2.3. Order payment completion request (REST)](#scroll-bookmark-201).
9. The payment gateway returns the result of processing the request. The order status is not returned. To receive the order status, it is necessary to send to the payment gateway the corresponding request, as it is described in step 15.

After the initial payment has been successfully processed, the store on its side enable the "Auto-payment" service for the customer (defines the date and amount of debiting for the given customer). Further the store independently tracks the date of a subsequent payment and initiates a payment by the binding identifier.

### Scenario of executing an auto-payment

On the date of a subsequent payment, the store initiates a payment according to the following scenario:

1. An order preliminary registration request that mandatory passes the unique identifier of the customer in the clientId parameter must be sent to the payment gateway For the order registration also such parameters are used as the amount to be debited, order number in the store system, the customer return URL. The request specification is presented in sections:  
   - [7.1.2. Request for registering an order with pre-authorization (SOAP)](#scroll-bookmark-198) ,  
   - [7.2.2. Request for registering an order with pre-authorization (REST)](#scroll-bookmark-199) .
2. In the response to the request, the payment gateway returns the unique identifier of the order in the payment system (as orderId).
3. The store sends the order payment request with the use of a binding, passing the number of the order in the payment system received at the previous step and the binding identifier received after the initial payment has been processed. The request specification is presented in sections:  
   - [7.1.12 Request for processing an order by a binding" (SOAP)](#scroll-bookmark-212),  
   - [7.2.12 Request for processing an order by a binding (REST)](#scroll-bookmark-213).
4. The payment gateway processes the payment (puts on hold funds of the customer's card account) and returns the result of the payment processing. The order status is not returned. To receive the order status, it is necessary to send to the payment gateway the corresponding request.  
   The specification of a typical order status request is present in sections:  
   - [7.1.4. Order status request (SOAP)](#scroll-bookmark-182),  
   - [7.2.4. Order status request (REST)](#scroll-bookmark-183).  
   The specification of an extended order status request is present in sections:  
   - [7.1.5. Extended order status request (SOAP)](#scroll-bookmark-184),  
   - [7.2.5. Extended order status request (REST)](#scroll-bookmark-185).
5. The payment gateway returns the order status (in orderStatus).
6. To debit funds from the customer's account, the store needs to send to the payment gateway the order completion request. The request specification is present in sections:  
    - [7.1.3 Order payment completion request (SOAP)](#scroll-bookmark-200),  
    - [7.2.3. Order payment completion request (REST)](#scroll-bookmark-201).
7. The payment gateway returns the result of processing the request. The order status is not returned. To receive the order status, it is necessary to send to the payment gateway the corresponding request, as it is described in step 4.

### Getting the bindings list of a customer

The store can get the list of identifiers of existing bindings for a certain clientId by sending the corresponding request to the payment gateway. The request specification is presented in sections:

- [7.1.16. Request for the list of bindings of a customer (SOAP)](#scroll-bookmark-202)

- [7.2.16. Request for the list of bindings of a customer (REST)](#scroll-bookmark-203)

### Deactivating/activating an existing binding

If necessary, the system allows stores to deactivate existing bindings with the use of API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.13. Request for deactivation of a binding (SOAP)](#scroll-bookmark-206),  
 - [7.2.13. Request for deactivation of a binding (REST)](#scroll-bookmark-207).

Further a deactivated binding can be activated again. To do so, a store needs to send to the payment gateway the corresponding request. The request specification is present in sections:  
 - [7.1.14. Request for activation of a binding (SOAP)](#scroll-bookmark-208),  
 - [7.2.14. Request for activation of a binding (REST)](#scroll-bookmark-209) .

### Changing the validity period of a binding

Changing the validity period of a binding may be required in the case of re-issuing a customer's card. It can be done with the use of API, by means of the SOAP or REST interface. The request specification is present in sections:  
 - [7.1.15. Request for changing the validity period of a binding (SOAP)](#scroll-bookmark-210),  
 - [7.2.15. Request for changing the validity period of a binding (REST)](#scroll-bookmark-211).

### Adding a card to the list of SSL-cards

Upon agreement with the bank, a merchant can use the method for adding the number of a card that has been used on a payment attempt to the list of SSL-cards:

- [7.1.18. Request for adding a card to the list of SSL-cards (SOAP)](#scroll-bookmark-196),  
 - [7.2.18. Request for adding a card to the list of SSL-cards (REST)](#scroll-bookmark-197).

Using "Alfa-click" to pay for an order

### Short description of the PayByClick system

The PayByClick system is one more means of payment of the payment gateway along with bank card payments. Meanwhile, the interaction scheme between the store and payment gateway does not change.

Payments through PayByClick are dedicated to customers of "Alfa-click".

The integration scheme depends on the way in which the "Alfa-click" means of payment is used:

1. The merchant accepts payments through "Alfa-click" along with using e-commerce. In this case, the button to switch to the PayByClick system is located on the payment page to which the customer is redirected to pay an order. The description of the process is present in section [4.6.2.1. Using "Alfa-click" and e-commerce](#scroll-bookmark-214).
2. The merchant accepts payments processed only through "Alfa-click". In this case, the request of a payment through "Alfa-click" is created to redirect the customer to the PayByClick system. The description of the process is present in section [4.6.2.2. Using solely "Alfa-click"](#scroll-bookmark-215).

The PayByClick system does not provides for a partial debiting of a pre-authorized order (on a two-phase payment), for a partial payment, and partial or full refund (reversal or refund). Payments are available only in roubles. There is no possibility to locate the payment data entering page on the side of a store.

The existing e-invoicing payment registers are used for reconciliation.

### Scenario of a payment for an order

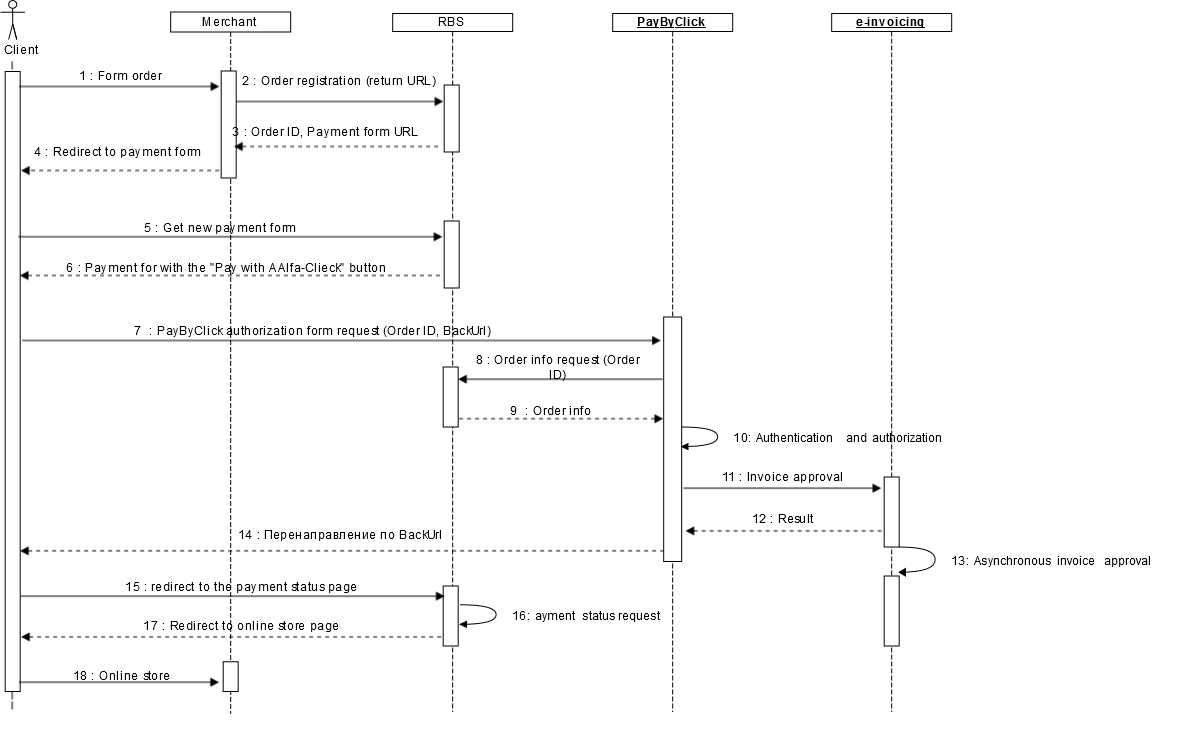
#### Using "Alfa-click" and e-commerce

If a store uses "Alfa-click" along with e-commerce, on creating the payment page, besides the standard requirements described in the "Payment page design" document, a requirement to locate on the page the following button is added:

|  |
| --- |
| <input type="button" class="alfaclick" id="buttonPaymentAlfa" value="Pay through Alfa-click" />" |

Also it is possible for a store to download the standard payment page that already has a button for switching to a payment through Alfa-click.

Below the general process of paying through PayByClick is described (without taking into account negative scenarios):



1. The customer creates an order at the site of the store.

2. After the order has been confirmed by the customer, the store registers the order in RBS. To register the order such parameters as the amount to be debited, order number in the store system, as well as the customer return URL, are used. The request specification is presented in sections:  
- [7.1.1 Order registration request" (SOAP)](#scroll-bookmark-180) ,  
- [7.2.1. Order registration request" (REST)](#scroll-bookmark-181).

3. RBS returns the order ID and the URL to redirect the customer to the payment form. In case of implementing the payment method selection on the side of the payment page, steps 4-6 are to be executed. In case of a selection of the payment method on the side of the store, at this step, the URL is passed to continue at step 7.

4. The store sends to the customer the redirect URL received at step 3.

5. The customer opens the received URL proceeding to the payment form.

6. The payment method selection form is displayed to the customer. On selecting "Pay through Alfa-click", the customer is redirected to PayByClick.

7. The customer's browser requests the authorization form with parameters:

* Order ID (received at step 3);
* The BackURL to return to the order status request page (that has been passed in the registration request at step 2).

8. PayByClick requests the order data by its ID.

9. RBS returns the order data.

10-13. The operations of authenticating the "Alfa-click" customer and authorizing a payment through "Alfa-click" .

14. PayByClick passed to the customer the RBS redirect URL received at step 7 and completes the operation. The payment status is not passed, only the order ID is passed.

15. The customer's browser opens the received BackURL of the payment status request page.

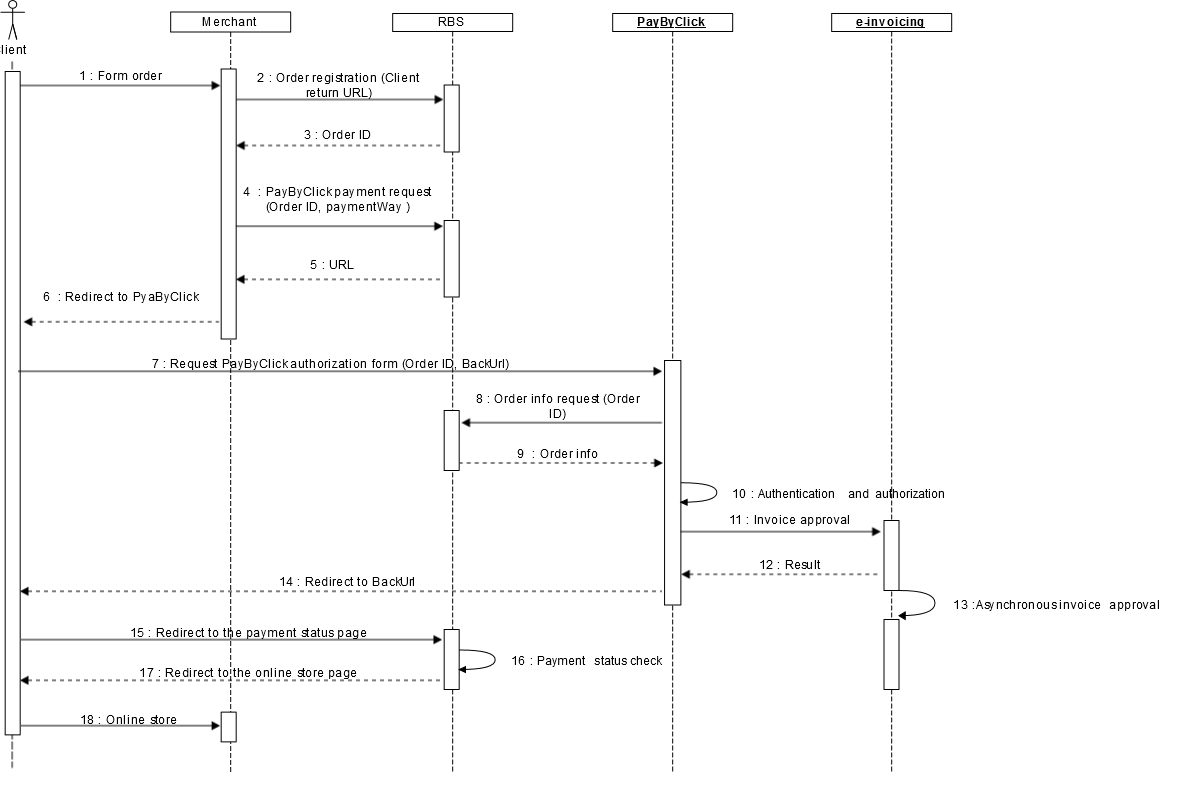
16. The order payment status is checked on the page.

17. After the order status gets the necessary value (DEPOSITED), the customer is redirected to the store page where the order status is displayed.

18. The customer gets the payment status page.

#### Using solely "Alfa-click"

Below the general process of a payment through the PayByClick system is described (without negative scenarios) for the case when the store accepts payments only through PayByClick:



1. The customer creates an order at the site of the store.

2. After the order has been confirmed by the customer, the store registers the order in RBS. To register the order such parameters as the amount to be debited, order number in the store system, as well as the customer return URL, are used. The request specification is presented in sections:  
- [7.1.1 Order registration request (SOAP)](#scroll-bookmark-180),  
- [7.2.1. Order registration request (REST)](#scroll-bookmark-181) .

3. RBS returns the order ID.

4. The store sends to RBS the request for paying the order through "Alfa-click" passing the order ID received at step 3. To do so, the request for an alternative payment method is used with a mandatory ALFA\_ALFACLICK value in the paymentWay parameter as well as with the order ID. The request specification is presented in sections:

- [7.1.11. Request for a payment through an external payment system (SOAP)](#scroll-bookmark-216),

- ["7.2.11. Request for a payment through an external payment system" (REST)](#scroll-bookmark-217).

5. In the response, the payment gateway sends the URL to redirect the customer to the PayByClick system.

6. The redirect link to the PayByClick system is passed to the customer's browser.

7. The customer's browser requests the authorization form with parameters:

* Order ID (received at step 3);
* The BackURL to return to the order status request page (that has been passed in the registration request at step 2).

8. PayByClick requests the order data by its ID.

9. RBS returns the order data.

10-13. The operations of authenticating the "Alfa-click" customer and authorizing a payment through "Alfa-click" .

14. PayByClick passed to the customer the RBS redirect URL received at step 7 and completes the operation. The payment status is not passed, only the order ID is passed.

15. The customer's browser opens the received BackURL of the payment status request page.

16. The order payment status is checked on the page.

17. After the order status gets the necessary value (DEPOSITED), the customer is redirected to the store page where the order status is displayed.

18. The customer gets the payment status page.

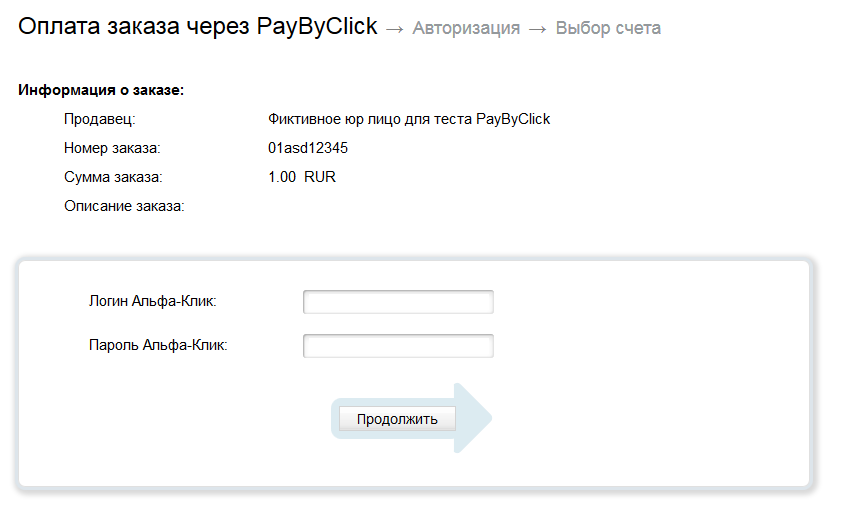
### Testing a payment through "Alfa-click"

1. Register an order in the payment gateway. The registration can be done with REST/SOAP.

2. Redirecting to the PayByClick system takes place:

* After the "Pay through Alfa-click" button has been pressed on the payment page, if the customer has been redirected there after registering the order,
* After the request for confirmation of a payment through "Alfa-click" (REST / SOAP) has been sent.

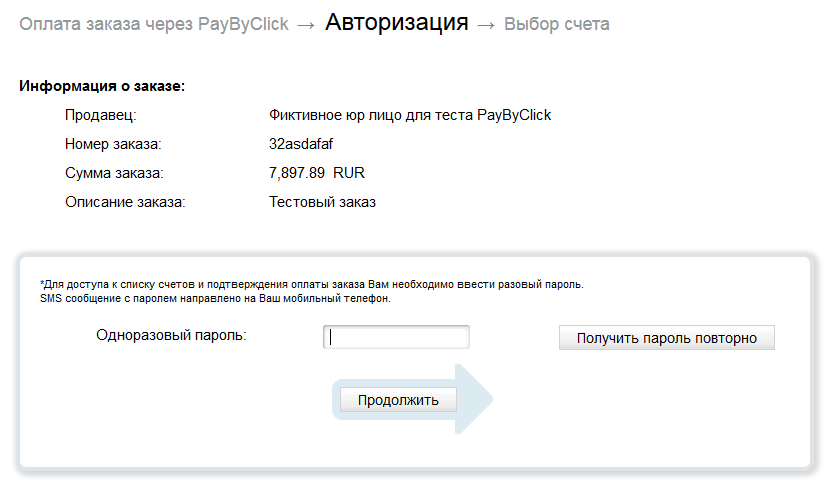
3. The page for a payment through "Alfa-click" will open at the following address <http://217.12.96.193/PayByClick/login.xhtml?faces-redirect=true>:



4. Enter the login and password for "Alfa-click" and click "Next". Test credentials:

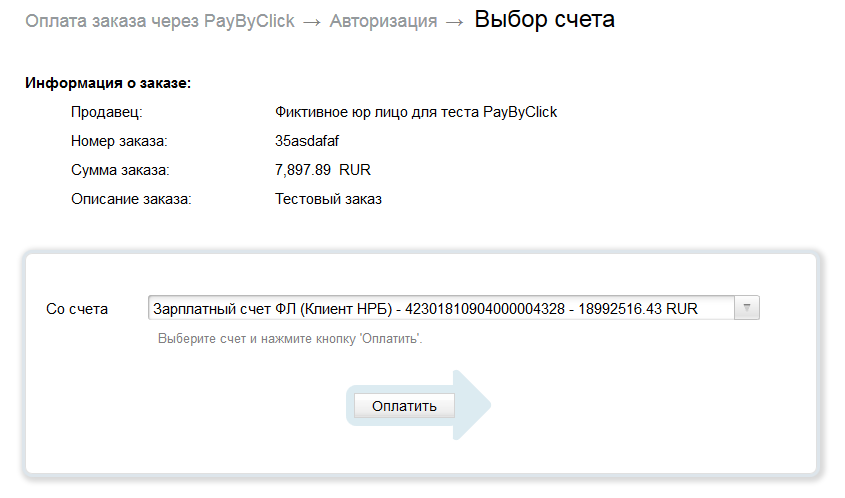
* "Alfa-click" login: 6819507
* "Alfa-click" password: 000000

5. The "Authorization" page will open:



6. Enter a one-time password and click "Next". The test one-time password: 0000.

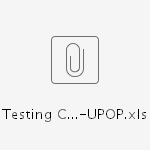
7. The page of selection of an account to be debited will open:



8. Select an account to be debited from the drop-down list and click "Pay". Redirection to the store page specified on the registration will occur.

9. After that, the payment is considered to be formally complete. Upon returning to the store site after the payment in the "Alfa-click" system has been processed, the payment status is not passed. Because of that, to clarify the payment status, the store needs to request RBS using the standard order status request (getOrderStatus) and to wait for the order to turn to the DEPOSITED status (funds are debited).

Using UPOP to pay for an order

The most recent information from UPOP are here 

### Short description of the CUP system

The UPOP tool is the payment means of the payment gateway that enables processing payments through China UnionPay (CUP). Meanwhile, the interaction scheme between the store and payment gateway does not change.

Payments using UPOP are available for holders of China Union Pay cards.

The integration scheme depends on the way in which the UPOP payment tool is used.

* A merchant accepts UPOP payments along with the use of e-commerce. In this case the button for switching to the CUP system is located on the payment page to which the customer is redirected to pay for the order. The description of the payment process is present in section [4.7.2.1. Using UPOP and e-commerce](#scroll-bookmark-218).
* The merchant accepts only payments through UPOP. In this case, to redirect the customer to the CUP system, the request for a payment through UPOP is created. The description of the payment process is present in section [4.7.2.2. Using solely UPOP](#scroll-bookmark-219).

The CUP system does not supports two-phase payments.

### Scenario of a payment for an order

#### Using UPOP and e-commerce

If a store uses "UPOP" along with e-commerce, on creating the payment page, besides the standard requirement described in the "Designing the payment page" document, a requirement to locate on the page the button element is added:

|  |
| --- |
| <input type="button" class="alfaclick" id="buttonPaymentUpop" value="Pay through UPOP" />" |

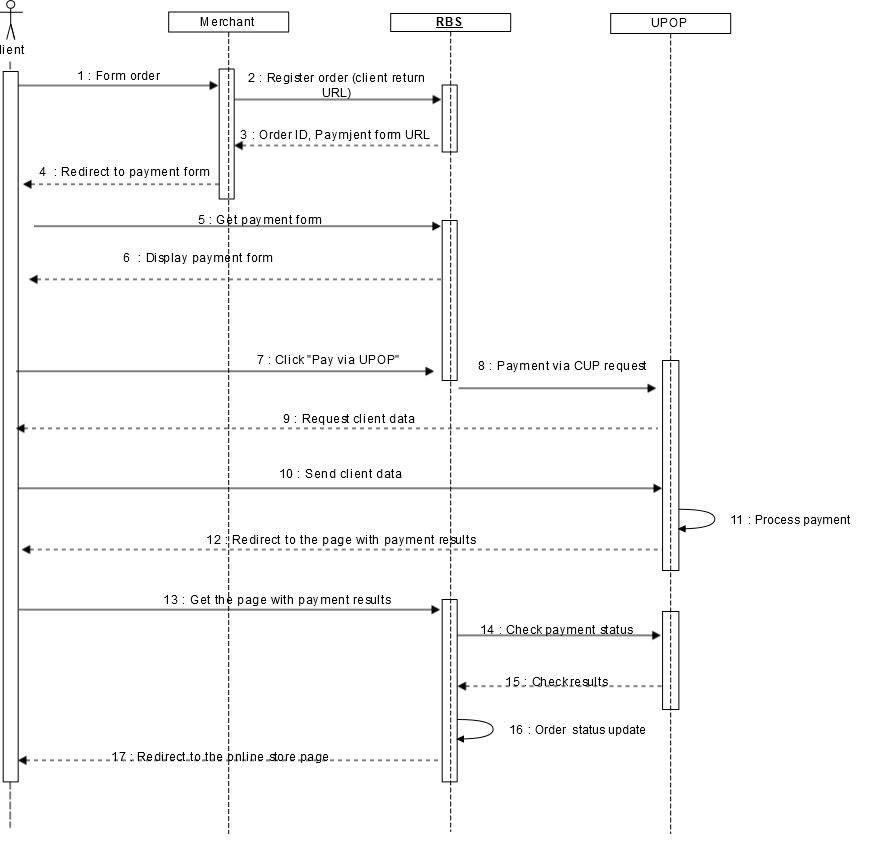
Upon pressing the button the request for processing a payment through UPOP is executed. The description of the request is present in sections:

- [7.1.11. Request for a payment through an external payment system (SOAP)](#scroll-bookmark-216),

- [7.2.11. Request for a payment through an external payment system (REST)](#scroll-bookmark-217).

Also there is a possibility to download for a store the standard payment page that already contains the button for switching to UPOP.

Below the general process of paying through the CUP system is described (without taking into account negative scenarios):



1. The customer creates an order and confirms it;

2. After the order has been confirmed by the customer, the store registers the order in RBS. To register an order, such parameters are used as the amount to be debited, order number in the store system (an alfa-numeric value from 8 to 32 symbols in length), as well as the customer redirect URL. The request specification is presented in sections:

- [7.1.1 Order registration request" (SOAP)](#scroll-bookmark-180), (in case of paying through UPOP, the order number in the store system must be an alfa-numeric value from 8 to 32 symbols in length);  
- [7.2.1. Order registration request" (REST)](#scroll-bookmark-181), (in case of paying through UPOP, the order number in the store system must be an alfa-numeric value from 8 to 32 symbols in length).

3. RBS returns the order ID and the URL to redirect the customer to the payment form.

4. The store sends to the customer the redirect URL received at step 3.

5. The customer opens the received URL proceeding to the payment form.

6. The customer is redirected to the payment page that belongs to the bank. On the payment page, the available method of payments through UPOP is displayed (the "Pay through UPOP" button).

7. The customer selects a payment through UPOP (presses the button).

8. The RBS system requests the order payment in the UPOP payment gateway.

9. The UPOP system requests from the customer the payment data.

10. The customer sends his or her data.

11. The UPOP system processes the payment.

12-13. The customer is redirected to the page with the payment result.

14. The RBS system calls the UPOP system to the check the payment status.

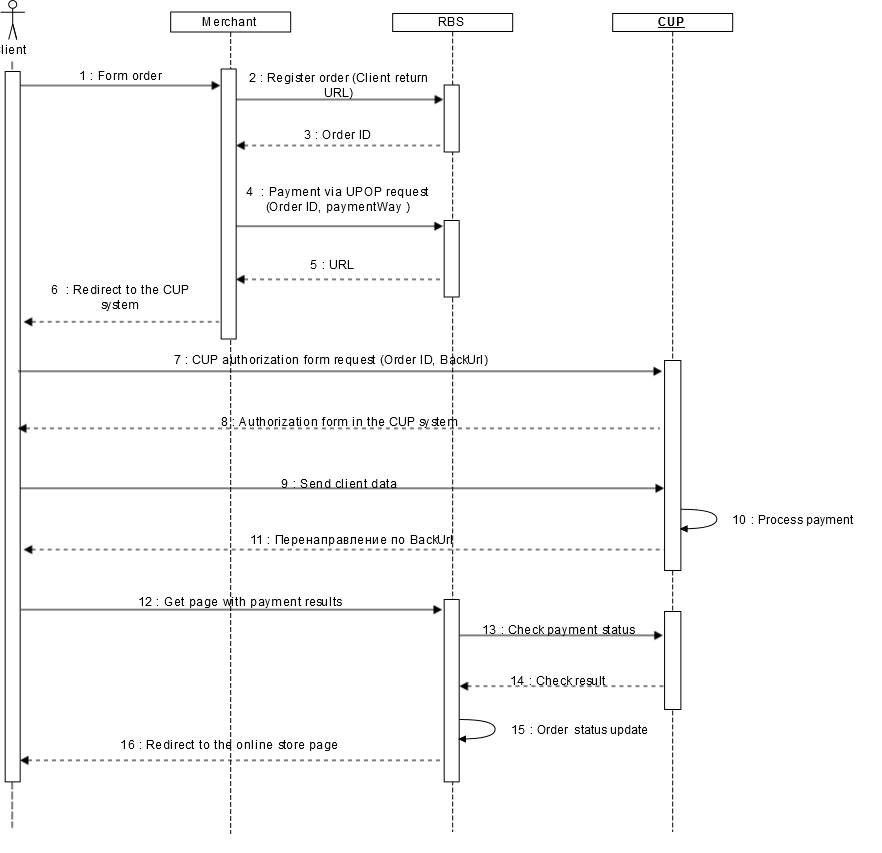
15. The UPOP system checks the result of the payment and returns it to the RBS system.

16. The order status in the RBS system is renewed.

17. The result is displayed to the customer.

#### Using solely UPOP

Below the general process of paying through the CUP system (without negative scenarios) for the case when the store accepts only payments through CUP:



1. The customer creates an order at the site of the store.

2. After the order has been confirmed by the customer, the store registers the order in RBS. To register an order, such parameters are used as the amount to be debited, order number in the store system (an alfa-numeric value from 8 to 32 symbols in length), as well as the customer redirect URL. The request specification is presented in sections:

- [7.1.1 Order registration request" (SOAP)](#scroll-bookmark-180), (in case of paying through UPOP, the order number in the store system must be an alfa-numeric value from 8 to 32 symbols in length);  
- [7.2.1. Order registration request" (REST)](#scroll-bookmark-181), (in case of paying through UPOP, the order number in the store system must be an alfa-numeric value from 8 to 32 symbols in length).

3. RBS returns the order ID.

4. The store sends to RBS the request for an order payment through UPOP passing the order ID received at step 3. To do this, the request for an alternative payment method is used with the UPOP value mandatorily passed in the paymentWay parameter, as well as the order ID. The request specification is presented in sections:

- [7.1.11. Request for a payment through an external payment system (SOAP)](#scroll-bookmark-216),

- [7.2.11. Request for a payment through an external payment system (REST)](#scroll-bookmark-217).

5. In the response, the payment gateway sends the URL to redirect the customer to the CUP system.

6. The redirect URL to the CUP system is passed to the customer's browser.

7. The customer's browser requests the authorization form with parameters:

* Order ID (received at step 3);
* The BackURL to return to the order status request page (that has been passed in the registration request at step 2).

8. The customer receives the form for authorization in the CUP system.

9. The customer fills in and sends the form to CUP.

10. The UPOP system processes the payment.

11-12. The customer is redirected to the page with the payment result.

13. The RBS system calls the UPOP system to the check the payment status.

14. The UPOP system checks the result of the payment and returns it to the RBS system.

15. The order status is renewed in the RBS system.

16. The result is displayed to the customer.

### Testing a payment through UPOP

#### Testing process

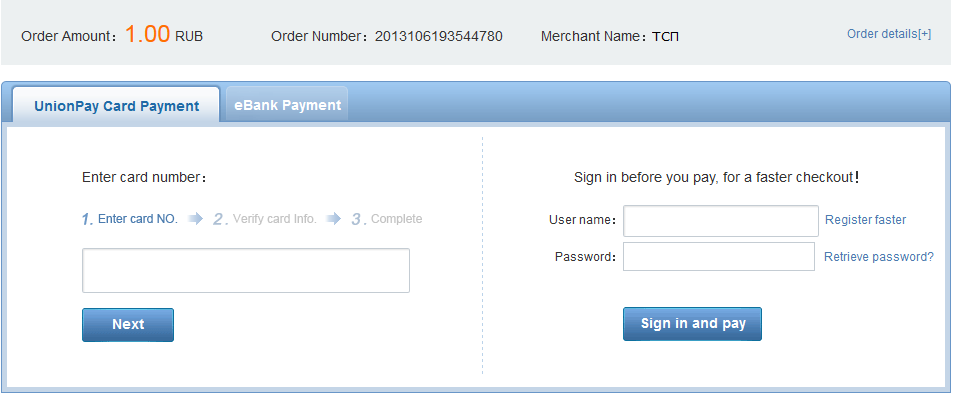
To test processing a payment through UPOP:

1. Register an order in the payment gateway. The order registration can be performed using REST / SOAP (the order number in the store system must be an alfa-numeric value from 8 to 32 symbols in length).

2. Redirecting to the CUP system occurs:

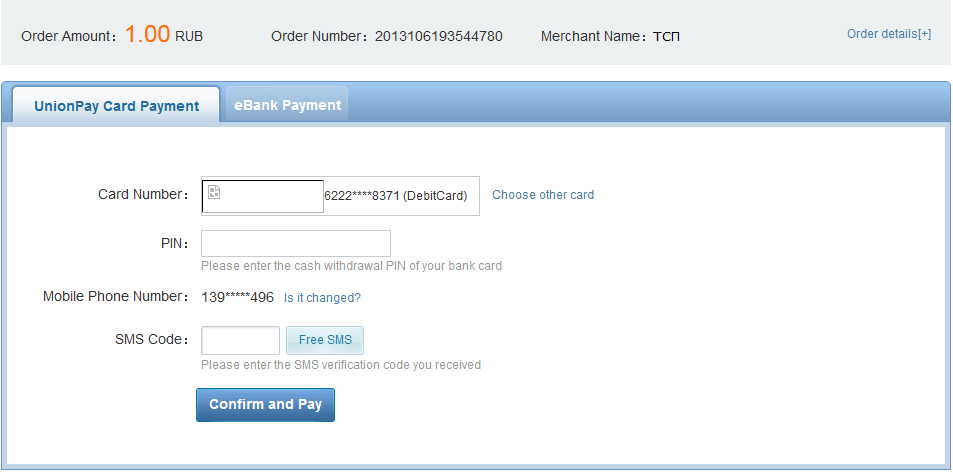
* + After the "Pay through UPOP" button on the payment page has been pressed, if the customer was redirected to it after registering an order.
  + After sending the request for a payment through UPOP (REST / SOAP).

3. The page of authorization in the CUP system will open at the address <http://202.101.25.184/beta/index.action?transNumber=201311062352028710592>:



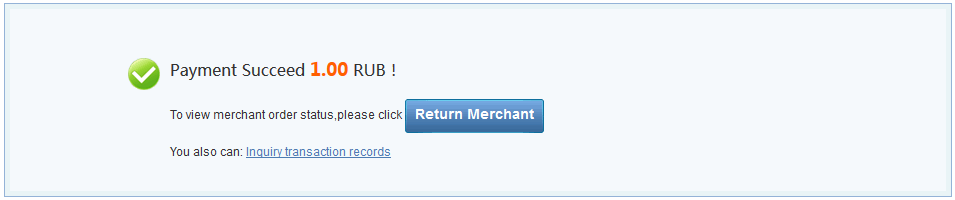
4. Enter the card number and click "Next". The test card credentials are present in the next section.

5. The confirmation page will open:



6. Enter the PIN-code and the confirmation SMS-code.

7. Click "Confirm and Pay". The page with the payment result will open:



Upon pressing the Return Merchant button redirection takes place back to the store page that was specified on the order registration in the returnUrl parameter (if the registration was performed by REST/ SOAP) or in the return address parameter (on the registration through the form).

8. After that, the payment is considered to be formally complete. On returning to the store site after a payment in the CUP system, the payment status is not passed. Because of this, the store needs to request the RBS system by the standard order status request (getOrderStatus) and to wait till the order status turns to DEPOSITED (the funds are debited).

#### Test China UnionPay cards

The cards that are present in the section are designed only to test processing payments through UPOP:

**Debit cards:**

|  |  |
| --- | --- |
| Card number | 6222 9888 1234 0000 |
| Mobile phone number | 13552535506 |
| PIN | 123456 |
| SMS Code on PC | 111111 |
| SMS Code on Mobile | 123456 |

|  |  |
| --- | --- |
| Card number | 6216261000000000018 |
| Mobile phone number | 13552535506 |
| PIN | 123456 |
| SMS Code on PC | 111111 |
| SMS Code on Mobile | 123456 |

|  |  |
| --- | --- |
| Card number | 6222988812340000 |
| Mobile phone number | 13552535506 |
| PIN | 123456 |
| Name | 互联网 |
| Personal ID | 341126197709218366 |
| OTP | 111111 |

|  |  |
| --- | --- |
| Card number | 6221558812340000 |
| Mobile phone number | 13552535506 |
| cvn2 | 123 |
| Name | 互联网 |
| Personal ID | 341126197709218366 |
| Expiry Date | 1711 |
| OPT | 111111 |

|  |  |
| --- | --- |
| Card number | 6216261000000000018 |
| Mobile phone number | 13552535506 |
| PIN | 123456 |
| Name | 全渠道 |
| Personal ID | 341126197709218366 |

|  |  |
| --- | --- |
| Card number | 5200831111111113 |
| Mobile phone number | 13552535506 |
| cvn2 | 123 |
| Name | 全渠道 |
| Personal ID | 341126197709218366 |
| Expiry Date | 1911 |

**Credit cards:**

|  |  |
| --- | --- |
| Card number | 6221 5588 1234 0000 |
| Mobile phone number | 13552535506 |
| CVN2 | 123 |
| Expiration Date | month 11 year 17 |
| SMS Code on PC | 111111 |
| SMS Code on Mobile | 123456 |

|  |  |
| --- | --- |
| Card number | 6226388000000095 |
| Mobile phone number | 18100000000 |
| CVN2 | 248 |
| Expiration Date | month 12 year 1 |
| SMS Code on PC | 111111 |
| SMS Code on Mobile | 123456 |

**The card issued outside China:**

|  |  |
| --- | --- |
| Card number | 4938 8112 3456 2006 |
| Mobile phone number | 11112222 |
| CVN2 | 123 |
| Expiration Date | month 11 year22 |
| SMS Code on PC | 111111 |

### Refunds for orders paid through UPOP

A full or a partial refund for orders paid using UPOP is executed by the standard tools:

* Through the administrative console (the description is provided in the document "[Instruction on working with the console](https://rbs-develop.paymentgate.ru/wiki/pages/viewpage.action?pageId=12976308)", the section "Working with orders");
* Using API, by means of the REST or SOAP interfaces. The request specification is presented in sections:  
   - [7.1.7. Request for a refund of an order payment funds (SOAP)](#scroll-bookmark-188),  
   - [7.2.7. Request for a refund of an order payment funds (REST)](#scroll-bookmark-189).

After RBS receives a refund request sent in one of the above mentioned ways, RBS sends a refund request to the UPOP system. In case of a successful response, the specified amount is refunded to the customer's account.

Payment using Apple Pay

|  |
| --- |
| Currently, payments with the use of mobile applications are supported. Also a merchant can place on its site a special button that enables accepting payments through the Apple Pay system. The description of preparing the site of a merchant to accepting Apple Pay payment is outside of the scope of this document. |

### Merchant actions necessary to connect to Apple Pay

#### Actions in the personal area of the Payment Gateway

Before you start accepting payments through Apple Pay, perform the following actions.

1. In the personal area, generate a key pair and upload the public key certificate signature request.

|  |
| --- |
| The procedure is described in the instruction for the console administrator. |

#### Creating Merchant ID

To create your Merchant ID, perform the following actions.

|  |
| --- |
| To be able to complete this procedure, you need to have an Apple Developer account. |

1. In the personal area of Apple **Member Center**, go to **Certificates, Identifiers & Profiles** .
2. On the page that opens in the **Identifiers** section on the left, select **Merchant IDs**.
3. On the page that opens, click **+** (**Add**) in the right upper corner.
4. In the **Merchant ID Description** and **Identifier** fields, enter the description of your Apple merchant identifier and the identifier itself correspondingly.

|  |
| --- |
| The identifier should begin with the word **merchant**. The address of your main site should be specified in the reverse order. For example, for the site **sale.test.ru** the identifier will look as **merchant.ru.test.sale**. |

1. Click **Continue**.
2. On the page that opens, check the data that you have entered and click **Register**.
3. On the page that opens, click **Done**.

#### Creating a certificate for Merchant ID

To create a certificate for your Merchant ID, perform the following actions.

1. In the personal area of Apple **Member Center**, go to **Certificates, Identifiers & Profiles** .
2. On the page that opens in the **Identifiers** section on the left, select **Merchant IDs**.
3. Select your Merchant ID from the list and click **Edit**.
4. Click **Create Certificate** and after this click **Continue**.
5. Click **Choose File** and specify the path to the file with the certificate signature request downloaded from the personal area of the payment gateway.

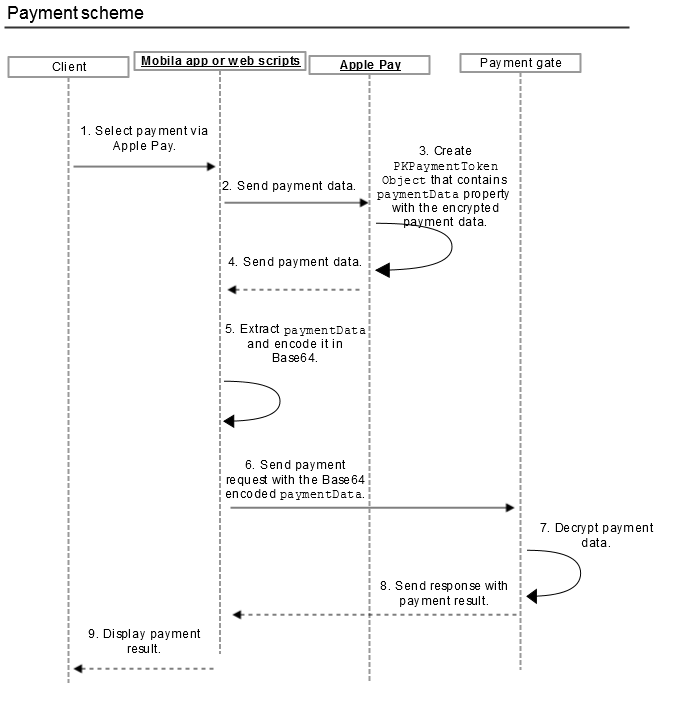
|  |
| --- |
| The procedure for creation of a file with the certificate signature request is presented in the «Administrator's instruction on working with the console» document . |

1. Click **Generate**.
2. Click **Download**, to download the created certificate to your computer.
3. After the certificate has been downloaded, click **Done**.

If you have completed the above mentioned actions, you can start developing the interaction of your mobile application or site with the payment gateway.

### Interaction scheme on a payment with Apple Pay

On a payment with Apple Pay, the interaction occurs according to the following scheme.



The scheme description is given below.

1. The user selects the payment method with the use of Apple Pay.
2. The payment data is sent to the Apple Pay system for processing.
3. To process the payment data in Apple Pay, the PKPaymentToken Object object is created that contains the paymentData property (here and further for more details, see [Apple Pay documentation](https://developer.apple.com/library/content/documentation/PassKit/Reference/PaymentTokenJSON/PaymentTokenJSON.html#//apple_ref/doc/uid/TP40014929)).
4. Apply Pay send a response to the merchant (a mobile application or a site).
5. The merchant extracts from the received PKPaymentToken Object object the paymentData property and encodes its content in Base64.
6. The merchant creates a payment request containing the paymentData property received for the response of Apple Pay an encoded in Base64, and sends in to the payment gateway for processing - see sections [Request for a payment through Apple Pay (Web-Service)](#scroll-bookmark-221) and [Request for a payment through Apple Pay (REST)](#scroll-bookmark-222).
7. The payment gateway processed the request.
8. The payment gateway returns the response with the result.
9. The mobile application or site displays to the customer the result.

### Executing recurring payments through Apple Pay

To initiate recurring payments, it is necessary to create a corresponding binding. To do so, it is necessary to create a payment processing request and specify in it clientId:

* [Request for processing a payment through Apple Pay (WS);](#scroll-bookmark-223)
* [Request for processing a payment through Apple Pay (REST)](#scroll-bookmark-222).

For the subsequent recurring payments, the recurrentPayment request is used:

* [Request for processing recurring payments through Apple Pay (WS);](#scroll-bookmark-224)
* [Request for processing recurring payments through Apple Pay (REST)](#scroll-bookmark-225).

|  |
| --- |
| When using the recurrentPayment request, the procedure for encrypting and decrypting payment data is not used. |

### Apple Pay - links to the reference information

The table below contains the links to the reference information on Apple Pay.

|  |  |
| --- | --- |
| Link | Description |
| <https://www.apple.com/apple-pay/> | A section of the **apple.com** site containing general information on Apple Pay. |
| <https://developer.apple.com/apple-pay/> | A section of the **apple.com** site dedicated to developers and containing links and reference information concerning Apple Pay. |
| <https://developer.apple.com/support/apple-pay-sandbox/> | A section of the **apple.com** site containing information on testing. |
| <https://developer.apple.com/apple-pay/Getting-Started-with-Apple-Pay.pdf> | A document in the PDF format containing general information on Apple Pay and links to the reference information. |
| <https://developer.apple.com/apple-pay/Apple-Pay-Identity-Guidelines.pdf> | A document in the PDF format containing recommendations on designing sites and mobile applications in the Apple style. |
| <https://developer.apple.com/library/ios/ApplePay_Guide/> | A section of the **apple.com** site containing a programming reference. |
| <https://developer.apple.com/app-store/review/guidelines/#apple-pay> | A section of the App Store reference dedicated to Apple Pay. |
| <https://developer.apple.com/library/ios/documentation/UserExperience/Reference/PassKit_Framework/index.html#//apple_ref/doc/uid/TP40012158> | API reference (application programming interface). |
| <https://developer.apple.com/library/ios/documentation/PassKit/Reference/PaymentTokenJSON/PaymentTokenJSON.html#//apple_ref/doc/uid/TP40014929> | The description of the PKPaymentToken Object structure. |
| <https://devforums.apple.com/community/ios/connected/applepay/> | The development environment login page. |

Payment using Android Pay

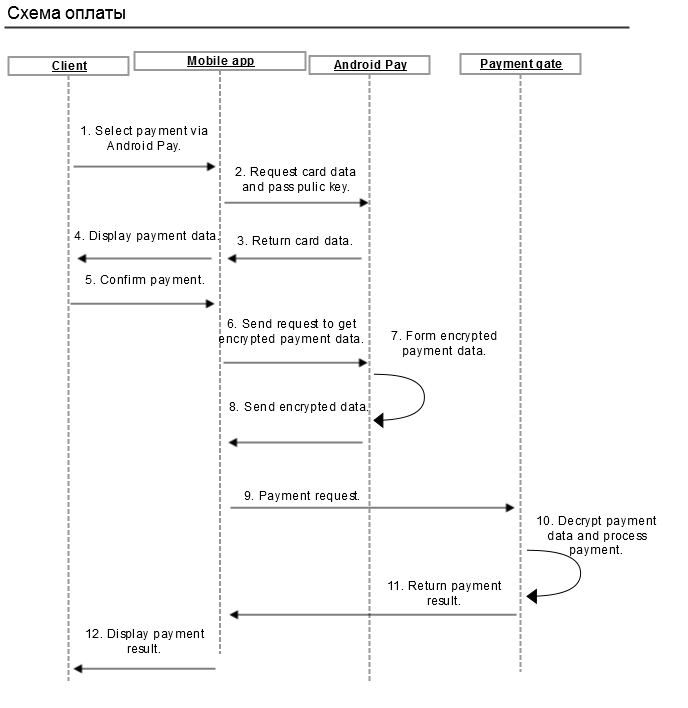
* [Preliminary actions](#scroll-bookmark-226)
* [Interaction schemes on a payment with Android Pay](#scroll-bookmark-227)
  + [Scheme with the use of the mobile application](#scroll-bookmark-228)
  + [Scheme with the use of the site adapted for mobile applications](#scroll-bookmark-229)

### Preliminary actions

Before starting accepting payments through Android Pay, the merchant needs to create a key pair in the personal area of the payment gateway. Also on enabling Android Pay, merchants need to fill in the form at this link:<https://androidpay.developers.google.com/signup>.

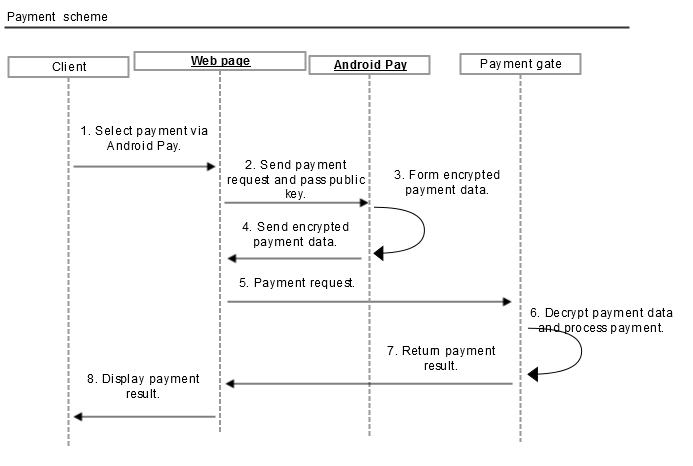
### Interaction schemes on a payment with Android Pay

#### Scheme with the use of the mobile application



1. A payer selects the Android Pay payment method.
2. The application requests the information on masked card data from Android Pay.
3. Android Pay returns the masked card data to the application.
4. The application displays to the payer the masked data of the card added in Android Pay.
5. The payer confirms the payment with the card added in Android Pay.
6. The application requests the encrypted card data from Android Pay
7. Android Pay encrypts the data using the public key.
8. Android Pay returns to the merchant the encrypted payment data.
9. The application send to the payment gateway the request for an Android Pay payment specifying the token received from Android:
   * [a payment request - the REST interface](#scroll-bookmark-230);
   * [a payment request - the WS](#scroll-bookmark-231) interface.
10. The payment gateway decrypts the received token and processes the payment.
11. The payment gateway returns the payment result to the application.
12. The application displays the result of the purchase to the payer.

#### Scheme with the use of the site adapted for mobile applications



1. A payer selects the Android Pay payment method.
2. The application requests the encrypted card data from Android Pay
3. Android Pay encrypts the data using the public key.
4. Android Pay returns to the merchant the encrypted payment data.
5. The application send to the payment gateway the request for an Android Pay payment specifying the token received from Android:
   * [a payment request - the REST interface](#scroll-bookmark-230);
   * [a payment request - the WS](#scroll-bookmark-231) interface.
6. The payment gateway decrypts the received token.
7. The payment gateway processes the payment.
8. The payment gateway returns the payment result to the application.
9. The application displays the result of the purchase to the payer.

Payment using Samsung Pay

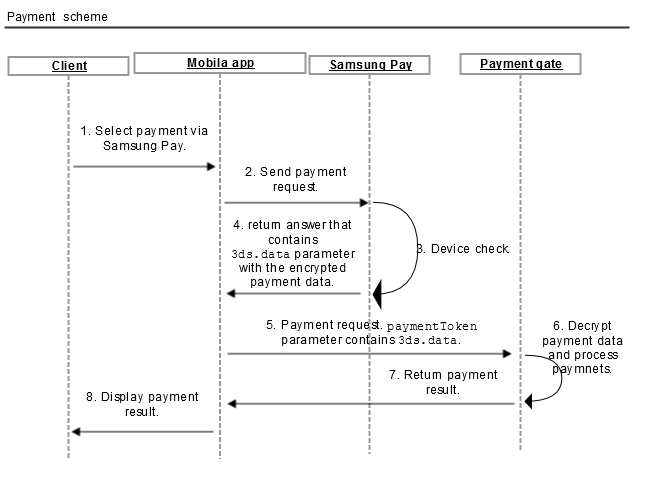
|  |
| --- |
| The functionality is currently in the testing phase. |

### Preliminary actions

Before the merchant starts accepting payments through Samsung Pay, it must register on the Samsung partner portal. After that, in the personal area of the payment gateway, the merchant must generate a key pair, export the certificate signature request, and upload it to the partner portal Samsung.

### Scheme with the use of the mobile application

Below is the interaction scheme for processing a payment using the mobile application.



1. A payer selects the Samsung Pay payment method.
2. The application sends the payment details to Samsung.
3. Samsung checks the application.
4. Samsung sends to the application a response containing, among other things, the 3ds.data parameter with the encrypted payment data.
5. The store send the payment request to the payment gateway, and the paymentToken parameter contains the contents of 3ds.data received from Samsung:
   * [samsungPay request, the REST interface](#scroll-bookmark-232);
   * [samsungPay request, the WS interface](#scroll-bookmark-233).
6. The payment gateway decrypts the contents of paymentToken and processed the payment.
7. The payment gateway sends the payment result to the application.
8. The application displays to the payer the result of the payment.

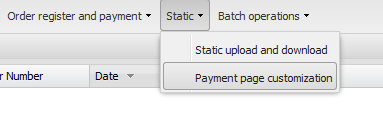
Payment page

Using a standard payment page

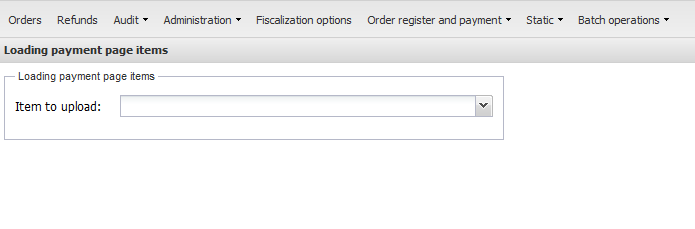
A store can use the standard payment page. To do so, the appropriate permission must be set in the store settings.

The system allows stores to load some elements (a logo and footer) to display them on the standard payment page. To load elements of the payment page:

1. In the administrative console, in the "Static" menu select "Payment page customization"



1. The page designed for loading elements of the payment page will open:

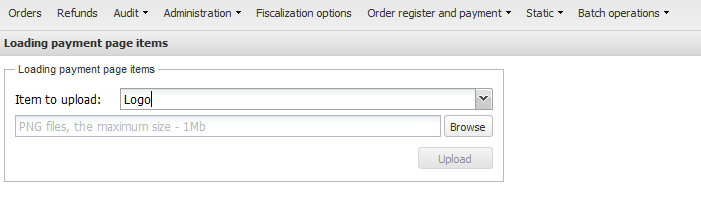


If the appropriate permission is present, the "Merchant" field is available to the user to select a merchant to the payment page of which it is planned to load the login or footer.

To upload a logo, follow these steps:

|  |
| --- |
| For the logo to be displayed without distortion, its width and height must be 245 x 64 px. |

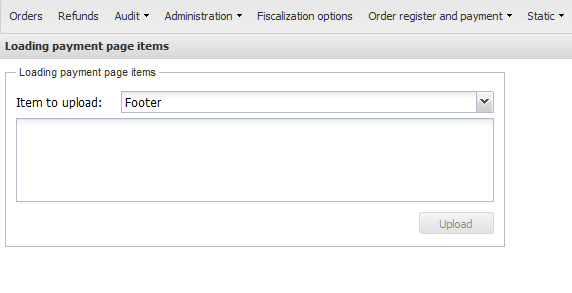
1. If the **Merchant** field is active, specify the merchant onto the payment page of which the logo is to be uploaded. Enter several first symbols of the merchant login characters or name, then press ENTER on the keyboard. A list of merchant with names beginning with the entered characters will be displayed. Select the necessary merchant.
2. From the drop-down **Element to load** list, select the **Logo** value.  
   The line for the selection of a file with a logo will be displayed below and the button to load it (see the picture below).



1. Click **Browse** and specify the path to the file on your computer. The file with the logo must be in the PNG format, the file size must not exceed 1 MB.
2. After the file has been selected, click **Upload**.
3. In case of a successful load, an information message will appear on the screen **The element successfully loaded**.
4. Click **ОК**.

After that, the uploaded logo will be displayed on the merchant's payment page (see [the picture below](#scroll-bookmark-234)). To change the logo, upload a new logo by repeating the steps above.  
To download a footer, follow these steps:

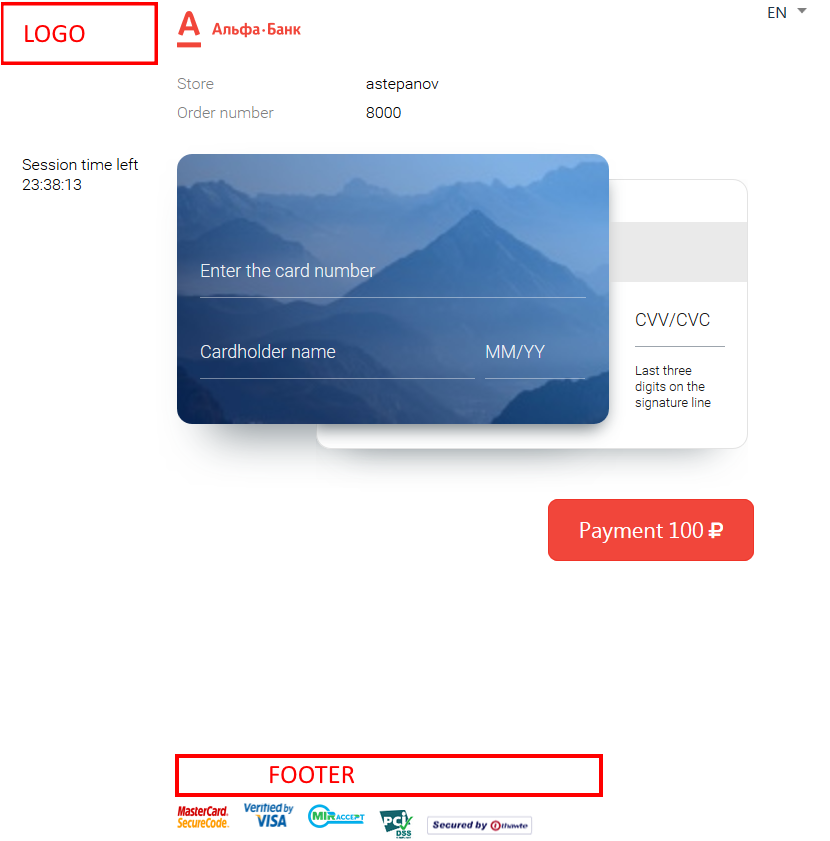
1. If the Merchantfield is active, specify the merchant onto the payment page of which the footer is to be uploaded. Enter several first symbols of the merchant login characters or name, then press ENTER on the keyboard. A list of merchant with names beginning with the entered characters will be displayed. Select the necessary merchant.
2. From the drop-down **Element to load** list, select the **Footer** value.  
   The line for entering the text will be displayed below and the button to load it (see the picture below).



1. Using the keyboard, enter the text of the footer and click **Upload**.

After that, the entered text will be displayed at the bottom of the merchant's payment page (see [the picture below](#scroll-bookmark-234)). To change the text of the footer, repeat the steps above.

The picture below shows an example of the standard payment page with a logo and footer loaded.



Using your own payment page

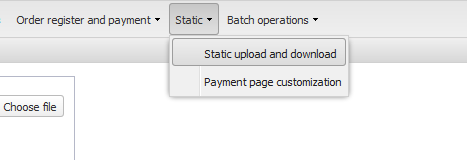
If a store plans to use its own payment interface pages, it is necessary:

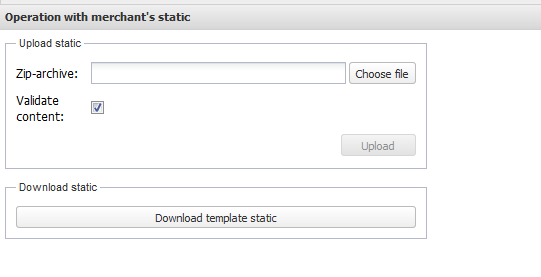
1. To create an archive with payment interface pages in accordance with the requirements described in this document.
2. To upload the archive with the pages of the payment interface through the administrative console.

Creating payment interface pages

Requirements for creating pages are listed below in section [5.3. Requirements for creation of a payment page](#scroll-bookmark-235).

It is possible to use the standard pages of the payment interface as the base sample. To download an archive with the standard payment page:

1. In the administrative console in the "Statics" menu, select "Uploading and downloading statics".  
   
2. The Operation with merchant's static page will open

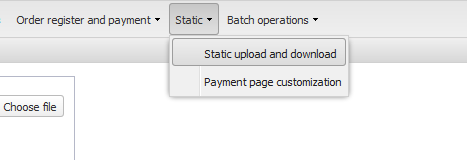


1. Click "Download template static". The archive with the standard pages of the payment interface will be saved on your computer.

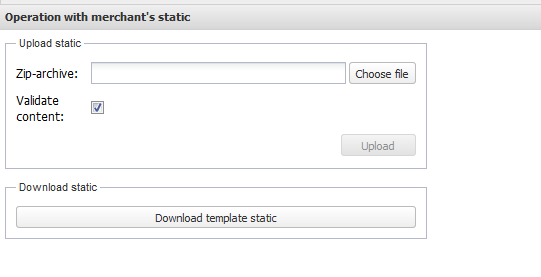
Uploading the archive with the payment interface pages

After the archive with the payment gateway pages is prepared, it must be uploaded to the payment gateway. To upload the archive:

1. In the administrative console in the "Static" menu, select "Uploading and downloading statics":



1. The "Operation with merchant's static" page will open:



1. In the "Upload static" section, click "Choose file" and specify the path to your archive with the payment interface pages.
2. The file name will be displayed in the "Zip-archive" field.
3. To start uploading the selected archive, click the "Upload".
4. In case of a successful operation, the information message "Archive successfully loaded" will be displayed. Click ОК.

In case of an unsuccessful operation, an error message will be displayed. The following reasons for errors are possible:

* + The uploaded file is not a ZIP archive or is empty.
  + The archive cannot be read.
  + The archive does not contain the required elements - a payment page and error pages. Or the necessary elements are located not in the root of the archive, but in a folder that is packed into the archive. It is necessary to archive the items themselves, not the folder that contains them.

After uploading the archive with the payment page on the test server, the user performs testing of the payment page:

* + using the REST interface \ the interface on web-services.
  + using the order registration form.
  + using the personal area and the console.

After testing the payment page, it is necessary to contact the bank to check the payment page. If the check is successful, the bank employees will transfer your payment page to the production server.

Requirements for the pages of the payment interface

### General requirements for the file containing the payment interface

The file with payment pages must be a zip-format archive, where the pages and folders with JS-scripts, CSS-styles and pictures are located in the root. An example of an archive with payment pages can be provided by the bank.

It is recommended to use the 7-zip archiver for archiving.

The page must be an XHTML page with a mandatory DTD declaration:

|  |
| --- |
| <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"> |

/wiki/s/en_GB/6217/82a642d727bee456b0a91262638763d5620b1a86.1/_/images/icons/emoticons/warning.png **ATTENTION** It is mandatory to use the declared XHTML standard, otherwise browsers may not work.

It is forbidden to use absolute addresses to connect to any resources (images, scripts, styles). All addresses must be relative to the locations of the page and folders with all the necessary resources (http:host/images/test.jpg - forbidden, images/test.jpg - allowed).

The archive with payment pages <t0 /> must mandatory contain payment interface pages<t1 /> designed for displaying on computer monitors. The names of these pages should be the following (in lowercase):

* payment\_<locale>.html – the payment page;
* errors\_<locale>.html – the errors page.  
  Where:
  + <locale> – the language of the page in the ISO 639-1 encoding. For example, ru for Russian or en for English.

For mobile devices, special pages can be used that are composed taking into account the need for compact and high-quality displaying on the screens of mobile devices. These pages are also put in the archive. The names of these pages must mandatorily be in lowercase and conform with the following templates:

* mobile\_payment\_<locale>.html – the payment page for mobile devices;
* mobile\_errors\_<locale>.html – the errors page for mobile devices.  
  Where
  + <locale> – the language of the page in the ISO 639-1 encoding. For example, ru for Russian or en for English.

Moreover, arbitrary prefixes can be added to the names of page files (according to the template below). This will enable redirecting the customer to the appropriate page of the payment interface by passing in the registration request the necessary prefix (in the pageViewparameter). The names of these pages must mandatorily be in lowercase and conform with the following templates:

* <prefix>\_payment\_<locale>.html – the payment page;
* <prefix>\_errors\_<locale>.html – the errors page.  
  Where:
  + <prefix> – an arbitrary value (up to 20 characters that can include letters, numbers and the "\_" symbol), indicating what payment interface this page belongs to. It is necessary to pass this value in the pageView parameter of the order registration request so that the pages of this interface are loaded for the customer.
  + <locale> – the page language in the ISO 639-1 encoding . For example, ru for Russian or en for English.

The payment page language:

* Is defined by the language parameter that is passed on registering an order
* Also, the localization.js script allows you to switch the language of the payment page when the customer is redirected to it. To do this, the <t0 /> language <t1 /> parameter with the language specified in the ISO 639-1 encoding must be passed to the URL of the payment page.. The examples of URLs of payment pages in Russian and English correspondingly:

-- https://server/application\_context/payment.html?mdOrder=0ec5a6ee-0a1e-4e71-ab34-0cf818c1ad6f&language=ru

-- https://server/application\_context/payment.html?mdOrder=0ec5a6ee-0a1e-4e71-ab34-0cf818c1ad6f&language=en

Which pages are to be loaded to the customer's browser (regular, mobile or with a certain prefix in the name) is determined by the special parameter, pageView, when registering an order The description of pageView is presented in the sections "Order registration request" and "Request for order registration with pre-authorization"

The archive with payment interface pages can also contain the following pages:

* A payment page that is loaded in case of passing a binding ID in the registration request.
* The final page (if a store has the appropriate permission, the customer is redirected to the final page after a payment attempt).
* The receipt page (the customer receives a receipt in the pdf-format after the corresponding request on the final page).

When creating the layout of the pages of the payment interface, it is necessary to use the Unicode (UTF-8) encoding.

### Requirements for the payment page

The page must contain a number of necessary objects, as well as a number of fields with specific names for entering payment information.

#### Page name

Name of a general page – payment\_<locale>.html,

Name of a mobile page – mobile\_payment\_<locale>.html,

Name of a page with an arbitrary prefix – <prefix>\_payment\_<locale>.html

Where:

* <prefix> – a part of a page name that can be passed in the <t3 /> pageView <t4 /> parameter of the order registration request for redirecting the customer to the corresponding page of the payment interface.
* <locale> – the page language in the ISO 639-1 encoding . For example, ru for Russian or en for English.

#### Page header

The following scripts must be included in the page header:

The standard set:

|  |
| --- |
| <script type="text/javascript" src="../../js/jquery-1.8.1.min.js"></script>   <script type="text/javascript" src="../../js/jquery.timers-1.2.js"></script>   <script type="text/javascript" src="../../js/jquery.url.js"></script>   <script type="text/javascript" src="../../js/jquery.payment\_new.js"></script>   <script>   $(document).payment({   });   </script> |

The extended set:

|  |
| --- |
| <script type="text/javascript" src="../../js/jquery-1.8.1.min.js"></script>   <script type="text/javascript" src="../../js/jquery.timers-1.2.js"></script>   <script type="text/javascript" src="../../js/jquery.url.js"></script>   <script type="text/javascript" src="../../js/jquery.payment\_new.js"></script>   <script>   $(document).payment({   language: "ru",   messageAjaxError: "The service is temporarily unavailable. Try again later.",   messageTimeRemaining: "The session will end in #MIN#:#SEC#",   visualValidationEnabled: true,  agreementCheckboxEnabled: true,   bindingCheckboxEnabled: true,  getFeeEnabled:true  });   </script> |

The fields of a script of the extended type must be filled in the following way:

* language – the value of the language name that matches the one selected for the page header
* messageAjaxError – the message about an internal Ajax error (that occurs, for example, when there is no access to the system)
* messageTimeRemaining – a message of the session counter. It must include the "# MIN #" and "# SEC #" keywords, which in real time will be replaced by minutes and seconds indicating the time remaining before the end of the session.
* visualValidationEnabled – the indicator of switching on/off visual confirmation of the validation results. If true, then a field with a correct value is colored in green and a field with an incorrect value — in red. If false (or not specified), visual validation will not work — the field colors will not change.
* agreementCheckboxEnabled – a flag for switching on/off the support of the store offer checkbox. If the variable value is true, then validation is switched on for the selection of the check-box that assumes the user's agreement to pay.
* bindingCheckboxEnabled – a flag or switching on/off the support of remembering a card.
* getFeeEnabled – a flag or switching on/off displaying the fee in a payment.

#### Page body

##### Mandatory elements on the payment page

All blocks and elements described in this paragraph must be placed in the body of the page, unless other is specified.

* The block containing the unique order number:

|  |
| --- |
| <div id="orderNumber"></div> |

* The block containing the order amount:

|  |
| --- |
| <div id="amount"></div> |

* The block containing the order description:

|  |
| --- |
| <div id="description"></div> |

The page must contain the payment form:

* All the hidden fields specified below are mandatory. The language field value must contain a two-letter code of the page locale.

|  |
| --- |
| <form name="PaymentForm" action="#" method="post" id="formPayment">   <input type="hidden" id="expiry" >   <input type="hidden" id="mdOrder" >   <input type="hidden" id="location" value="/../" >   <input type="hidden" id="language" value="<locale>" >   </form> |

The form must also contain fields for entering information for processing the payment:

* The field for entering the credit card number:

|  |
| --- |
| <input name="$PAN" id="iPAN" maxlength="19" type="text" autocomplete="off" /> |

* The selector of the month and the selector of the credit card expiration year (it is filled in automatically when the page is loaded):

|  |
| --- |
| <select name="MM" id="month">   <option value="01" selected> 1 - January</option>   <option value="02"> 2 - February</option>   <option value="03"> 3 - March</option>   <option value="04"> 4 - Aprile</option>   <option value="05"> 5 - May</option>   <option value="06"> 6 - June</option>   <option value="07"> 7 - July</option>   <option value="08"> 8 - August</option>   <option value="09"> 9 - September</option>   <option value="10">10 - October</option>   <option value="11">11 - November</option>   <option value="12">12 - December</option>   </select> /   <select name="YYYY" id="year">   <option value='2012' selected>2012</option>   <option value='2013'>2013</option>   <option value='2014'>2014</option>   <option value='2015'>2015</option>   <option value='2016'>2016</option>   <option value='2017'>2017</option>   <option value='2018'>2018</option>   <option value='2019'>2019</option>   <option value='2020'>2020</option>   <option value='2021'>2021</option>   <option value='2022'>2022</option>   </select> |

* The field for entering the cardholder name:

|  |
| --- |
| <input name="TEXT" id="iTEXT" maxlength="90" type="text" autocomplete="off" /> |

* The field for entering the cvc/cvv/cid -code:

|  |
| --- |
| <input name="$CVC" id="iCVC" maxlength="3" type="password" autocomplete="off" /> |

* The button for the payment confirmation:

|  |
| --- |
| <input value="Pay" type="button" id="buttonPayment"> |

The following code must be placed below the payment form:

|  |
| --- |
| <form id="acs" method="post" action="">   <input type="hidden" id="MD" name="MD"/>   <input type="hidden" id="PaReq" name="PaReq"/>   <input type="hidden" id="TermUrl" name="TermUrl"/>   </form> |

The following objects must also be placed on the payment page:

* The block where errors are displayed (for example, incorrect card data):

|  |
| --- |
| <div id="errorBlock" style="color:red;"></div> |

* The block where the message is displayed about the time remaining till the end of the payment session:

|  |
| --- |
| <div id="numberCountdown"></div> |

* The block where the information message is displayed on switching from the payment page to the final page:

|  |
| --- |
| <div id="infoBlock"></div> |

* The block where the indicator of the progress of a request to the server is displayed (on confirming the payment and a subsequent call to the server):

|  |
| --- |
| <div id="indicator" style="display:none;"><img src="../../img/ajax-loader.gif" height="19" width="220" alt="indicator"></div> |

If all the requirements are met, on paying for an order, on the payment page you will see:

- the order amount;

- the order number in the store system;

- the order description (displayed only in the description field is filled in).

##### Placing additional elements on the payment page

###### Displaying payment parameters

To display other parameters on the payment page, use the following blocks:

|  |
| --- |
| <tr valign="top" id="Parameter name"> <td valign="top"></td> </tr> |

where The parameter name can have the following values:

* + amount - the amount of order registration in the minor units;
  + amountFormatted - a formatted registration amount (with separators of digit groups and a decimal point);
  + approvedAmount - the amount put on hold;
  + approvedAmountFormatted - a formatted hold amount (with separators of digit groups and a decimal point);
  + currency - three-digit numeric currency code;
  + currencyName - three-digit alphabetic currency code;
  + date - the date of completion of the transaction payment, dd.MM.yyyy HH: mm: ss;
  + depositedAmount - сthe amount to be debited;
  + depositedAmountFormatted - the formatted amount to be debited (with separators of digit groups and a decimal point);
  + depositFlag - a sign of a two-stage transaction;
  + ipCountryCode - the country code of the payer;
  + mdorder - the order number in the payment system;
  + mdOrder - нorder number in the payment system;
  + merchantFullName - the full name of the merchant;
  + merchantLogin - the merchant login;
  + orderDescription - the order description;
  + orderNumber - the order number in the merchant system;
  + panCountryCode - the country code of the card of the payer;
  + paymentState - the transaction status (started / payment\_approved / payment\_declined / payment\_void/payment\_deposited / refunded);
  + paymentWay - the payment method;
  + processingId - the merchant identifier in the processing system;
  + terminalId - the terminal.

###### Displaying additional parameters of an order

To add any additional order parameters, use queriedParams. To do this, add to the payment page headers theparamNamesparameter. The parameters to be displayed are listed in square brackets separated by commas.

|  |
| --- |
| $(document).payment({  paramNames:[parameter\_name1, parameter\_name2, parameter\_name3]  }); |

To display a parameter on the page, add an element with the ID equal to the parameter name. If a parameter will pass a link, add an A element with the same ID.

|  |
| --- |
| <div id=” parameter\_name1”></div>  <a href=”#” id=”parameter\_name2”></div> |

**Example:**

To display the return to the store link on the payment page, it is necessary:

1. To add the payment page:

* + Script:

|  |
| --- |
| <script> $(document).payment( { paramNames:['backUrl'] } ); </script> |

* + An element of the link type with the same ID as specified above in paramNames:

|  |
| --- |
| <a href="#" id="backUrl">Return to store</a> |

2. To pass backUrlas an additional parameter on registering an order (for a REST-request in jsonParams; for a SOAP-request in the params tag).

###### Notifying a customer about an executed operation

If the functionality is used of notifying the customer about the payment / cancellation / return / rejection of the order, add the following block:

|  |
| --- |
| <input type="text" id="email"/> |

the field for entering the email address of the customer.

###### Displaying the amount of the fee

If the merchant has the permission to work with fees, it is necessary to place the following elements on the payment page:

The block that contains the fee amount:

|  |
| --- |
| <div id="feeAmount"></div> |

The block with the information about the offer conditions of the store.. The payment will not be processed if the user has not checked the agreement box. The header of the agreementCheckboxEnabled page must contain true:

|  |
| --- |
| <label><input type="checkbox" class="checkbox" id="iAgree"> I have read and accept the conditions<br><a href="#">of the offer</a></label> |

###### Check box of saving the card data of a payment

If the permission to work with binding is set in the settings of the merchant, the payment page must contain a checkbox, which allows the customer to indicate the need to remember the card data of this payment. The header of the bindingCheckboxEnabled page must contain true:

|  |
| --- |
| <label class="ch-wrap"><input type="checkbox" class="checkbox" id="createBinding">Remember the data of this card</label> |

##### Card data validation

To display the process of passing the validation of the card data to the customer, the following code must be placed on the payment page:

1. .valid  
   { border: 1px solid #088A08; background-color: #CEF6CE; }  
   .invalid  
   { border: 1px solid #8A0808; background-color: #F6CECE; }
2. visualValidationEnabled:true,
3. id="iPAN", id="iCVC", id="iTEXT", id="month", id="year

### Requirements for the payment page in case of passing a binding ID in the registration request

If the binding ID has been passed in the order registration request, the payer is redirected to the payment page where only CVC is required.

#### Page name

The page name is payment\_binding\_<locale>.html,

Where:

* <locale> – the language of the page in the ISO 639-1 encoding. For example, ru for Russian or en for English.

#### Page header

The following scripts must be included in the page header:

|  |
| --- |
| <script type="text/javascript" src="../../js/jquery-1.4.2.min.js"></script>   <script type="text/javascript" src="../../js/jquery.timers-1.2.js"></script>   <script type="text/javascript" src="../../js/jquery.url.js"></script>   <script type="text/javascript" src="../../js/jquery.payment\_binding.js"></script>   <script type="text/javascript">   $(document).payment\_binding({   visualValidationEnabled:true   });   $(document).ready(function () {   $(document).payment\_binding("validate");   $(document).payment\_binding("showError", "");   });   </script> |

#### Page body

Optional fields:

* The block containing the unique order number:

|  |
| --- |
| <div id="orderNumber"></div> |

* The block containing the order amount:

|  |
| --- |
| <div id="amount"></div> |

* The block containing the order description:

|  |
| --- |
| <div id="description"></div> |

* The block that contains the masked card number:

|  |
| --- |
| <div id="maskedPan"></div> |

The page must contain the payment form:

|  |
| --- |
| <form action="" method="post" id="bindingPaymentForm"> |

The form must also contain fields for entering information for processing the payment:

* The field for entering the cvc/cvv/cid -code:

|  |
| --- |
| <input name="$CVC" id="cvc" maxlength="3" type="password" autocomplete="off" /> |

* The button for the payment confirmation:

|  |
| --- |
| <input value="Pay" type="button" id="sendPayment"> |

The following code must be placed below the payment form:

|  |
| --- |
| <form id="acs" method="post" action="">   <input type="hidden" id="md" name="MD"/>   <input type="hidden" id="paReq" name="PaReq"/>   <input type="hidden" id="termUrl" name="TermUrl"/>   </form> |

The payment page must also contain the following objects:

* The block that displays errors:

|  |
| --- |
| <div id="errorBlock" style="color:red;"></div> |

* The block where the message is displayed about the time remaining till the end of the payment session:

|  |
| --- |
| <div id="countdown"></div> |

* The block where the information message is displayed on switching from the payment page to the final page:

|  |
| --- |
| <div id="infoBlock"></div> |

* The block where the indicator of the progress of processing the request to the server is displayed (when confirming a payment and a subsequent call to the server):

|  |
| --- |
| <div id="indicator" style="display:none;"><img src="../../img/ajax-loader.gif" height="19" width="220" alt="indicator"></div> |

### Requirements for the errors page

The page must contain a number of necessary objects.

#### Page name

The name of a general page – errors\_<locale>.html,

The name of a mobile device page – mobile\_errors\_<locale>.html

The name of al page with an arbitrary prefix – <prefix>\_errors\_<locale>.html

Where:

* <prefix> – the part of the page name that can be passed in the pageView parameter of the order registration request to redirect the customer to the corresponding page of the payment interface.
* <locale> – the page language in the ISO 639-1 encoding . For example, ru for Russian or en for English.

#### Page header

The page header must contain the following scripts:

|  |
| --- |
| <script type="text/javascript" src="../../js/jquery-1.4.2.min.js"></script>   <script type="text/javascript" src="../../js/jquery.timers-1.2.js"></script>   <script type="text/javascript" src="../../js/jquery.url.js"></script>   <script type="text/javascript" src="../../js/error\_page.js"></script> |

|  |
| --- |
| <script type="text/javascript">   var lang = "<locale>";   </script> |

where <locale> - a two-letter code of a locale the page (for example, ru - Russian, en - English, ISO 639-1).

#### Page body

All the blocks and elements described below must mandatory be placed in the body of the page.

Form:

|  |
| --- |
| <form name="errorForm" action="#" method="post" id="errorForm">   <input type="hidden" id="language" value="<locale>" type="hidden">   </form> |

where <locale> - a two-letter code of a locale the page (for example, ru - Russian, en - English, ISO 639-1).

The form must also contain a block where errors are displayed (for example, about the end of a session or a message about an already processed payment):

|  |
| --- |
| <div id="errorBlock" style="color:red;"></div> |

### Requirements for the final page

If the merchant has the appropriate permission, the customer will be redirected to the final page after an attempt to pay.

#### Page name

The page name is finish.html.

#### Page header

The following scripts must be included in the page header:

|  |
| --- |
| <script type="text/javascript" src="../../js/jquery-1.9.1.min.js"></script>  <script type="text/javascript" src="../../js/jquery-ui-1.9.2.custom.min.js"></script>  <script type="text/javascript" src="../../js/select.js"></script>  <script type="text/javascript" src="../../js/maskedinput.js"></script>  <script type="text/javascript" src="../../js/jquery.finish.main.js"></script>  <script type="text/javascript" src="../../js/jquery.finishpayment.js"></script>  <script type="text/javascript" src="../../js/jquery.url.js"></script>  <script type="text/javascript" src="../../js/localization.js"></script>   <script type="text/javascript">  $(document).paymentFinished();  $( document ).ready(function() {  var logo = $(".user-logo");  function isLogoImg(logodiv){  var img = new Image();  img = logodiv.find(".bg");  img.error(function(){  logodiv.remove();  });  img.load(function(){  logodiv.find(".title-logo").remove();  });  }  isLogoImg(logo);  });  localizePage();  </script> |

#### Page body

The page used elements of the span type:

|  |
| --- |
| <span langLbl="<LOCALIZE\_PROPERTY>"> |

For the values of<LOCALIZE\_PROPERTY> see localization.js

Also the page used elements of the divtype:

|  |
| --- |
| <div id="<PROPERTY>"></div> |

, where <PROPERTY> – the order parameter. The possible values are:

* status – the order status;
* paymentDate – the payment date;
* orderNumber – the order number in the store system;
* approvalCode – the authorization code in IPS;
* terminalId – the terminal identifier;
* refNum – the reference number;
* amount – the order amount;
* formattedAmount – the formatted registration amount (with separators of digit groups and a decimal point);
* feeAmount – the fee amount;
* formattedFeeAmount – the formatted fee amount (with separators of digit groups and a decimal point);
* currency – the payment currency;
* panMasked – the masked number of the card ;
* expiry – the card expiration date;
* cardholderName – the cardholder name;
* orderDescription – the order description;
* merchantShortName – the short name of the merchant;
* merchantFullName – the full name of the merchant;
* merchantUrl – the address of the merchant site;
* actionCodeDescription – the description of the response code of the processing system;
* orderParams.<ADDITIONAL\_PARAM> – is used to display additional transaction parameters, where <ADDITIONAL\_PARAM> – the name of an additional parameter.

The final page can contain the following elements:

The element of returning to the store from the final page:

|  |
| --- |
| <a class="back-btn" href="#"><span langLbl="backToShop">Back to the store</span></a> |

The element of the generation of a PDF file :

|  |
| --- |
| <a class="pdf" href="#"><span langLbl="saveToPDF">Save as PDF</span></a> |

The slider element:

|  |
| --- |
| <div class="load-wrapper">  <div class="bg"></div>  <div class="clock"><span class="hour"></span><span class="minute"></span></div> </div> |

### Requirements for the receipt page

The customer receives this page as a PDF-file, when he or she presses "Save as PDF" on the final page.

#### Page name

The name of the successful payment page - finish\_template\_success\_<locale>.html.

The name of the declined payment page – finish\_template\_error\_<locale>.html.

Where:

* <locale> – the language of the page in the ISO 639-1 encoding. For example, ru for Russian or en for English.

#### Requirements for creation of a page

It is forbidden to use JavaScript when creating this page.

To display an order parameter on the page, specify the parameter name as follows:

|  |
| --- |
| ${<PROPERTY>} |

, where <PROPERTY> – the name of an order parameter. The possible values are:

* status – the order status;
* paymentDate – the payment date;
* orderNumber – the order number in the store system;
* approvalCode – the authorization code in IPS;
* terminalId – the terminal identifier;
* refNum – the reference number;
* amount – the order amount;
* formattedAmount – the formatted registration amount (with separators of digit groups and a decimal point);
* feeAmount – the fee amount;
* formattedFeeAmount – the formatted fee amount (with separators of digit groups and a decimal point);
* currency – the payment currency;
* panMasked – the masked number of the card ;
* expiry – the card expiration date;
* cardholderName – the cardholder name;
* orderDescription – the order description;
* merchantShortName – the short name of the merchant;
* merchantFullName – the full name of the merchant;
* merchantUrl – the address of the merchant site;
* actionCodeDescription – the description of the response code of the processing system;
* orderParams.<ADDITIONAL\_PARAM> – is used to display additional transaction parameters, where <ADDITIONAL\_PARAM> – the name of an additional parameter.

Plug-ins and code examples for the integration with the gateway

Plug-ins are provided on demand for the following CMSs:

1. Bitrix
2. CS-Cart
3. Drupal
4. Joomla
5. UMI
6. Wordpress

The following samples of the PHP code can be used to simplify the integration:

**A sample of the PHP code to integrate with the gateway through WS:**

|  |
| --- |
| <?php   /\*\*  \* THE PAYMENT GATEWAY CONNECTION DATA  \*  \* USERNAME Store login received on the connection.  \* PASSWORD Store password received on the connection.  \* WSDL The address of the web-service description.  \* RETURN\_URL The address to which to redirect the user   \* in case of a successful payment.  \*/ define('USERNAME', 'USERNAME'); define('PASSWORD', 'PASSWORD'); define('WSDL', 'https://server/payment/webservices/merchant-ws?wsdl'); define('RETURN\_URL', 'http://your.site/ws.php');   /\*\*  \* THE CLASS FOR INTERACTION WITH THE PAYMENT GATEWAY  \* The class is inherited from the standard class, SoapClient.  \*/ class Gateway extends SoapClient {    /\*\*  \* AUTHORIZATION IN THE PAYMENT GATEWAY  \* Generating a SOAP-header for WS\_Security.  \*  \* RESPONSE  \* SoapHeader SOAP-header for authorization  \*/  private function generateWSSecurityHeader() {  $xml = '  <wsse:Security SOAP-ENV:mustUnderstand="1" xmlns:wsse="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">  <wsse:UsernameToken>  <wsse:Username>' . USERNAME . '</wsse:Username>  <wsse:Password Type="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0#PasswordText">' . PASSWORD . '</wsse:Password>  <wsse:Nonce EncodingType="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0#Base64Binary">' . sha1(mt\_rand()) . '</wsse:Nonce>  </wsse:UsernameToken>  </wsse:Security>';    return new SoapHeader('http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd', 'Security', new SoapVar($xml, XSD\_ANYXML), true);  }   /\*\*  \* CALLING A METHOD OF THE PAYMENT GATEWAY  \* SoapClient::\_\_call() function override.  \*  \* PARAMETERS  \* method A method from API.  \* data Data array.  \*   \* RESPONSE  \* response Response.  \*/   public function \_\_call($method, $data) {  $this->\_\_setSoapHeaders($this->generateWSSecurityHeader()); // Creating a header for authorization  return parent::\_\_call($method, $data); // Returning the result of the SoapClient::\_\_call() method  } }  /\*\*  \* DISPLAYING THE FORM ON THE SCREEN  \*/ if ($\_SERVER['REQUEST\_METHOD'] == 'GET' && !isset($\_GET['orderId'])) {  echo '  <form method="post" action="/ws.php">  <label>Order number</label><br />  <input type="text" name="orderNumber" /><br />  <label>Amount</label><br />  <input type="text" name="amount" /><br />  <button type="submit">Submit</button>  </form>  '; }  /\*\*  \* PROCESSING THE DATA FROM THE FORM  \*/ else if ($\_SERVER['REQUEST\_METHOD'] == 'POST') {  $client = new Gateway(WSDL);  $data = array('orderParams' => array(  'returnUrl' => RETURN\_URL,  'merchantOrderNumber' => urlencode($\_POST['orderNumber']),  'amount' => urlencode($\_POST['amount'])  ));    /\*\*  \* REGISTERING A ONE-PHASE PAYMENT IN THE PAYMENT GATEWAY  \* registerOrder  \*   \* PARAMETERS  \* merchantOrderNumber Unique identifier of the order in the store.  \* amount Order amount.  \* returnUrl Address to which the user is to be redirected in case of a successful payment.  \*  \* RESPONSE  \* In case of an error:  \* errorCode Error code. The list of available values is presented in the table below.  \* errorMessage Error description.  \*  \* In case of a successful registration:  \* orderId Identifier of the order in the payment system. It is unique within the system.  \* formUrl URL of the payment form to which the customer's browser is to be redirected.  \*  \* Error code Description  \* 0 The request has been processed without system errors.  \* 1 A request with this number has been already registered in the system;  \* Wrong order number.  \* 3 Unknown (forbidden) currency.  \* 4 A required request parameter is missing.  \* 5 An error of the request parameter value.  \* 7 System error.  \*/  $response = $client->\_\_call('registerOrder', $data);   /\*\*  \* REGISTERING A TWO-PHASE PAYMENT IN THE PAYMENT GATEWAY  \* registerOrderPreAuth  \*  \* The parameters and the response are the same as in the previous method.  \* It is necessary to call either registerOrder or registerOrderPreAuth.  \*/ // $response = $client->\_\_call('registerOrderPreAuth', $data);    if ($response->errorCode != 0) { // In case of an error, displaying it  echo 'Error #' . $response->errorCode . ': ' . $response->errorMessage;  } else { // In case of success, redirecting the customer to the payment page  header('Location: ' . $response->formUrl);  die();  }   }  /\*\*  \* PROCESSING THE DATA AFTER COMPLETING THE PAYMENT FORM  \*/ else if ($\_SERVER['REQUEST\_METHOD'] == 'GET' && isset($\_GET['orderId'])){  $client = new Gateway(WSDL);  $data = array('orderParams' => array('orderId' => $\_GET['orderId']));    /\*\*  \* ORDER STATUS REQUEST  \* getOrderStatus  \*  \* PARAMETERS  \* orderId Identifier of the order in the payment system. It is unique within the system.  \*   \* RESPONSE  \* ErrorCode Error code. The list of available values is presented in the table below.  \* OrderStatus The status of the order in the payment system is defined by the value of this parameter.   \* The list of available values is presented in the table below. It is missing if the order has not been found.  \*  \* Error code Description  \* 0 The request has been processed without system errors.  \* 2 The order has been declined because of an error in the payment details.  \* 5 Access denied;  \* The user must change the password;  \* The order number is not specified.  \* 6 An unknown order id.  \* 7 System error.  \*  \* Order status Description  \* 0 The order has been registered, but not paid.  \* 1 The-preauthorized amount is put on hold (for two-phase payments).  \* 2 The full authorization of the order amount has been processed.  \* 3 Authorization has been cancelled.  \* 4 A refund operation has been executed for the transaction.  \* 5 Authorization through the ACS of the issuing bank has been initiated.  \* 6 Authorization declined.  \*/  $response = $client->\_\_call('getOrderStatus', $data);    // Displaying an error code and the order status  echo '  <b>Error code:</b> ' . $response->errorCode . '<br />  <b>Order status:</b> ' . $response->orderStatus . '<br />  '; }  ?> |

**An example of the PHP code**  **for the integration with the gateway using REST:**

|  |
| --- |
| <?php  /\*\*  \* THE PAYMENT GATEWAY CONNECTION DATA  \*  \* USERNAME Store login received on the connection.  \* PASSWORD Store password received on the connection.  \* GATEWAY\_URL The payment gateway address.  \* RETURN\_URL The address to which to redirect the user   \* in case of a successful payment.  \*/ define('USERNAME', 'USERNAME'); define('PASSWORD', 'PASSWORD'); define('GATEWAY\_URL', 'https://server/payment/rest/'); define('RETURN\_URL', 'http://your.site/rest.php');  /\*\*  \* THE FUNCTION FOR THE INTERACTION WITH THE PAYMENT GATEWAY  \*   \* To send POST-requests to the payment gateway a standard library with URL is used  \* .  \*  \* PARAMETERS  \* method A method from API.  \* data Data array.  \*  \* RESPONSE  \* response Response.  \*/ function gateway($method, $data) {  $curl = curl\_init(); // Initiating the request  curl\_setopt\_array($curl, array(  CURLOPT\_URL => GATEWAY\_URL.$method, // The full address of the method  CURLOPT\_RETURNTRANSFER => true, // To return a response  CURLOPT\_POST => true, // POST method  CURLOPT\_POSTFIELDS => http\_build\_query($data) // The data in the request  ));  $response = curl\_exec($curl); // Executing the request    $response = json\_decode($response, true); // Decoding from JSON to an array  curl\_close($curl); // Closing the connection  return $response; // Returning the response }  /\*\*  \* DISPLAYING THE FORM ON THE SCREEN  \*/ if ($\_SERVER['REQUEST\_METHOD'] == 'GET' && !isset($\_GET['orderId'])) {  echo '  <form method="post" action="/rest.php">  <label>Order number</label><br />  <input type="text" name="orderNumber" /><br />  <label>Amount</label><br />  <input type="text" name="amount" /><br />  <button type="submit">Submit</button>  </form>  '; }  /\*\*  \* PROCESSING THE DATA FROM THE FORM  \*/ else if ($\_SERVER['REQUEST\_METHOD'] == 'POST') {  $data = array(  'userName' => USERNAME,  'password' => PASSWORD,  'orderNumber' => urlencode($\_POST['orderNumber']),   'amount' => urlencode($\_POST['amount']),  'returnUrl' => RETURN\_URL  );   /\*\*  \* THE REQUEST FOR REGISTERING A ONE-PHASE PAYMENT IN THE PAYMENT GATEWAY  \* register.do  \*   \* PARAMETERS  \* userName Store login.  \* password Store password.  \* orderNumber Unique identifier of the order in the store.  \* amount Order amount in the minor denomination of the currency.  \* returnUrl Address to which the user is to be redirected in case of a successful payment.  \*  \* RESPONSE  \* In case of an error:  \* errorCode Error code. The list of available values is presented in the table below.  \* errorMessage Error description.  \*  \* In case of a successful registration:  \* orderId Identifier of the order in the payment system. It is unique within the system.  \* formUrl URL of the payment form to which the customer's browser is to be redirected.  \*  \* Error code Description  \* 0 The request has been processed without system errors.  \* 1 An order with the same identifier has already been registered in the system.  \* 3 Unknown (forbidden) currency.  \* 4 A required request parameter is missing.  \* 5 An error of the request parameter value.  \* 7 System error.  \*/  $response = gateway('register.do', $data);    /\*\*  \* THE REQUEST FOR REGISTERING A TWO-PHASE PAYMENT IN THE PAYMENT GATEWAY  \* registerPreAuth.do  \*  \* The parameters and the response are the same as in the previous method.  \* It is necessary to call either register.do or registerPreAuth.do.  \*/ // $response = gateway('registerPreAuth.do', $data);    if (isset($response['errorCode'])) { // In case of an error, displaying it  echo 'Error #' . $response['errorCode'] . ': ' . $response['errorMessage'];  } else { // In case of success, redirecting the customer to the payment page  header('Location: ' . $response['formUrl']);  die();  } }  /\*\*  \* PROCESSING THE DATA AFTER COMPLETING THE PAYMENT FORM  \*/ else if ($\_SERVER['REQUEST\_METHOD'] == 'GET' && isset($\_GET['orderId'])){  $data = array(  'userName' => USERNAME,  'password' => PASSWORD,  'orderId' => $\_GET['orderId']  );    /\*\*  \* ORDER STATUS REQUEST  \* getOrderStatus.do  \*  \* PARAMETERS  \* userName Store login.  \* password Store password.  \* orderId Identifier of the order in the payment system. It is unique within the system.  \*   \* RESPONSE  \* ErrorCode Error code. The list of available values is presented in the table below.  \* OrderStatus The status of the order in the payment system is defined by the value of this parameter.   \* The list of available values is presented in the table below. It is missing if the order has not been found.  \*  \* Error code Description  \* 0 The request has been processed without system errors.  \* 2 The order has been declined because of an error in the payment details.  \* 5 Access denied;  \* The user must change the password;  \* The order number is not specified.  \* 6 An unknown order id.  \* 7 System error.  \*  \* Order status Description  \* 0 The order has been registered, but not paid.  \* 1 The-preauthorized amount is put on hold (for two-phase payments).  \* 2 The full authorization of the order amount has been processed.  \* 3 Authorization has been cancelled.  \* 4 A refund operation has been executed for the transaction.  \* 5 Authorization through the ACS of the issuing bank has been initiated.  \* 6 Authorization declined.  \*/  $response = gateway('getOrderStatus.do', $data);    // Displaying an error code and the order status  echo '  <b>Error code:</b> ' . $response['ErrorCode'] . '<br />  <b>Order status:</b> ' . $response['OrderStatus'] . '<br />  '; }  ?> |

Request specifications

Web-Service interface

### Order registration request

The order registration request is called registerOrder, it is described in WSDL of the service.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| merchantOrderNumber | ANS..32 | yes | Identifier of an order in the store system, it is unique for each store within the system |
| description | ANS..512 | no | Description of the order in any format |
| amount | N..20 | yes | Amount of the payment in the minor denomination of the currency |
| currency | N3 | no | ISO 4217 code of the payment currency. If the code is not specified, the default value is 810 (Russian roubles). |
| language | A2 | no | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used. |
| pageView | ANS..20 | no | By the value of this parameter, it is defined what pages of the payment interface are to be loaded for the customer. The possible values are:   * DESKTOP – to load pages the layout of which is designed to be displayed on displays of PCs (a search for pages with the payment\_<locale>.html and errors\_<locale>.htmlnames will be executed in the archive of payment interface pages); * MOBILE – to load pages the layout of which is designed to be displayed on displays of mobile devices (a search for pages with the mobile\_payment\_<locale>.html and mobile\_errors\_<locale>.html names will be carried out in the archive of payment interface pages); * If a store has created payment interface pages with arbitrary prefixes added to the names of page files, pas the value of the necessary prefix in the pageView parameter to load corresponding pages. For example on passing the iphone value, a search will be carried out in the archive of payment interface pages for pages with the iphone\_payment\_<locale>.html and iphone\_error\_<locale>.html names.   Where:  locale – the language of the page in ISO 639-1. For example, ru for Russian or en for English.  If a parameter is missing or does not match the format, it is considered that by default pageView=DESKTOP. |
| sessionTimeoutSecs | N...9 | no | Order lifetime in seconds.  If the parameter is not specified, the value specified in the merchant settings or the default value (1200 seconds = 20 minutes) will be used.  If the expirationDateparameter is present in the request, the value of the sessionTimeoutSecs parameter is ignored. |
| bindingId | AN..255 | no | Identifier of the binding created earlier. It can be used only if the store has the permission to work with bindings. If this parameter is passed in the given request, it means that: 1. The given order can be paid only using a binding; 2. The payer will be redirected to the payment page on which only entering CVC is required. |
| expirationDate | ANS | no | The date and time of the order lifetime expiration. The format used: yyyy-MM-dd'T'HH:mm:ss.  If this parameter is not passed in the request, sessionTimeoutSecs is used to define the date and time of the order lifetime expiration. |
| returnUrl | AN..512 | yes | Address to which the user is to be redirected in case of a successful payment. The address must be specified in full including the protocol used (for example, **https://test.ru** instead of **test.ru**). Otherwise, the user will be redirected to the address of the following type: **http://<payment\_gateway\_address>/<merchant\_address>**. |
| failUrl | AN..512 | no | The address to which the user iis to be redirected in case of an unsuccessful payment. The address must be specified in full including the protocol used (for example, **https://test.ru** instead of **test.ru**). Otherwise, the user will be redirected to the address of the following type: **http://<payment\_gateway\_address>/<merchant\_address>**. |
| params |  | no | Tag containing attributes to pass additional parameters of the merchant.  The fields for additional information and its subsequent storage. To pass N parameters, a request must contain N params tags, where the name attribute contains the name of a parameter and the value attribute contain its value:   |  |  |  |  | | --- | --- | --- | --- | | Name | Type | Mandatory | Description | | name | AN..20 | yes | Name of the additional parameter | | value | AN..1024 | yes | Value of the additional parameter |   These fields can be passed to the Bank processing system to be subsequently displayed in the registers.\*  Enabling the functionality is possible upon agreement with the Bank during the integration period.   |  | | --- | | On payments for housing and utilities services it is necessary to pass the following additional parameters:   * payment\_doc\_id - the identifier of the payment document; * order\_num - the order number. |   If sending notifications to customers is set up for a merchant, the email address of a customer must be passed in this tag in the email parameter. |
| clientId | AN..255 | no | Identifier of the customer in the store system. This parameter is used for the binding functionality. It can be present if creating bindings is allowed for the store.   |  | | --- | | Specifying this parameter when processing payments with the use of bindings is mandatory. Otherwise, a payment will be unsuccessful. | |
| merchantLogin | AN..255 | no | To register an order on behalf of a child merchant, specify the merchant login in this parameter. |
| features | ANS..255 | no | Container for the feature parameter, the available values for the parameter are:  AUTO\_PAYMENT - if the order registration request initiates executing auto-payments.  VERIFY - If this parameter is specified after the order registration request, the cardholder is to be verified without debiting funds from the cardholder account. Thus it is possible to pass a zero amount in the request. This verification allow the merchant to ensure that a card is used by the cardholder and to debit this card in the future without verifying authentication data (CVC, 3D-Secure) on processing subsequent payments.   |  | | --- | | **The details of passing the VERIFY value**   * Even if the payment amount is to be passed in the request, it will not be debited from the account. * After the order has been successfully registered, it is passed to the REVERSED (cancelled) status. |   Example:   |  | | --- | | <features>  <feature>AUTO\_PAYMENT</feature> </features> | |

**\*** By default, the following fields are passed to the Bank processing system:

* + - orderNumber – the order number in the store system;
    - description – the order description (no more than 99 symbols, it is forbidden to use %, +, an end of line \r and line break \n).

If the additional parameter merchantOrderId is passed in an order, its value is to be passed to the processing system of the Bank as an order number (instead of the value in the orderNumber field).

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | no | Identifier of the order in the payment system. It is unique within the system. The identifier is missing if the order registration failed due to an error detailed in errorCode. |
| formUrl | AN..512 | no | URL of the payment form to which the customer's browser is to be redirected. The URL is not returned if the registration of the order fails due to an error detailed in errorCode. |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Error description in the language passed in the language parameter in the request. |

**Error codes (the** errorCode **field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | Unknown order ID |
| 1 | An order with this identifier has already been processed |
| 3 | Unknown currency |
| 4 | Amount is missing |
| 4 | Order number cannot be empty |
| 4 | Return URL cannot empty |
| 5 | Wrong value for one the parameters |
| 5 | Access denied |
| 5 | The user must change the password |
| 7 | System error |
| 13 | Using both values, Features FORCE\_TDS/FORCE\_SSL and AUTO\_PAYMENT, is not allowed |
| 13 | The merchant does not have the permission to process auto-payments |
| 13 | The merchant does not have the permission to process verification payments |
| 14 | Features are specified incorrectly |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:registerOrder>   <order merchantOrderNumber="78ds901234567890" description=" " amount="15000" currency=" " language=" " pageView="MOBILE" sessionTimeoutSecs=" " bindingId=" " expirationDate="2014-09-08T14:14:14">   <returnUrl>https://web.rbsuat.com/ab/finish.html</returnUrl>   <params name="param1" value="valueParam1"/>  <params name="param2" value="valueParam2"/>   <clientId>666</clientId>   <merchantLogin> </merchantLogin>  <features>  <feature>AUTO\_PAYMENT</feature>  </features>  </order>   </mer:registerOrder>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:registerOrderResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return orderId="05fcbc62-7ee6-4f1a-b3d5-6ca41a982283" errorCode="0" errorMessage="Success">   <formUrl> https://web.rbsuat.com/ab/mobile\_payment\_ru.html?mdOrder=05fcbc62-7ee6-4f1a-b3d5-6ca41a982283 </formUrl>   </return>   </ns1:registerOrderResponse>   </soap:Body>   </soap:Envelope> |

### Registration request for orders with pre-authorization

The order pre-authorization request is called registerOrderPreAuth. It is used to register an order in the case of a two-phase payment scheme.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| merchantOrderNumber | ANS..32 | yes | Identifier of an order in the store system, it is unique for each store within the system |
| description | ANS..512 | no | Description of the order in any format |
| amount | N..20 | yes | Payment amount in the minor denomination (cents or kopeks) |
| currency | N3 | no | ISO 4217 code of the payment currency. If this parameter is not specified, it is considered to be equal to the default currency code. |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, the default language specified in the store settings is to be used. |
| pageView | ANS..20 | no | By the value of this parameter, it is defined what pages of the payment interface are to be loaded for the customer. The possible values are:   * DESKTOP – to load pages the layout of which is designed to be displayed on displays of PCs (a search for pages with the payment\_<locale>.html and errors\_<locale>.htmlnames will be executed in the archive of payment interface pages); * MOBILE – to load pages the layout of which is designed to be displayed on displays of mobile devices (a search for pages with the mobile\_payment\_<locale>.html and mobile\_errors\_<locale>.html names will be carried out in the archive of payment interface pages); * If a store has created payment interface pages with arbitrary prefixes added to the names of page files, pas the value of the necessary prefix in the pageView parameter to load corresponding pages. For example on passing the iphone value, a search will be carried out in the archive of payment interface pages for pages with the iphone\_payment\_<locale>.html and iphone\_error\_<locale>.html names.   Where:  locale – the language of the page in ISO 639-1. For example, ru for Russian or en for English.  If a parameter is missing or does not match the format, it is considered that by default pageView=DESKTOP. |
| sessionTimeoutSecs | N...9 | no | Order lifetime in seconds.  If the parameter is not specified, the value specified in the merchant settings or the default value (1200 seconds = 20 minutes) will be used.  If the expirationDateparameter is present in the request, the value of the sessionTimeoutSecs parameter is ignored. |
| bindingId | AN..255 | no | Identifier of the binding created earlier. It can be used only if the store has the permission to work with bindings. If this parameter is passed in the given request, it means that:  1. The given order can be paid only using a binding;  2. The payer will be redirected to the payment page on which only entering CVC is required. |
| expirationDate | ANS | no | The date and time of the order lifetime expiration. The format used: yyyy-MM-dd'T'HH:mm:ss.  If this parameter is not passed in the request, sessionTimeoutSecs is used to define the date and time of the order lifetime expiration. |
| returnUrl | AN..512 | yes | The address to which the user is to be redirected in case of a successful payment. The address must be specified in full including the protocol used (for example, **https://test.ru** instead of **test.ru**). Otherwise, the user will be redirected to the address of the following type: **http://<payment\_gateway\_address>/<merchant\_address>**. |
| failUrl | AN..512 | no | The address to which the user is to be redirected in case of an unsuccessful payment. The address must be specified in full including the protocol used (for example, **https://test.ru** instead of **test.ru**). Otherwise, the user will be redirected to the address of the following type **http://<payment\_gateway\_address>/<merchant\_address>**. |
| params |  | no | Tag containing attributes to pass additional parameters of the merchant.  The fields for additional information and its subsequent storage. To pass N parameters, a request must contain N params tags, where the name attribute contains the name of a parameter and the value attribute contain its value:   |  |  |  |  | | --- | --- | --- | --- | | Name | Type | Mandatory | Description | | name | AN..20 | yes | Name of the additional parameter | | value | AN..1024 | yes | Value of the additional parameter |   These fields can be passed to the Bank processing system to be subsequently displayed in the registers.\*  Enabling this functionality is possible upon agreement with the Bank during the integration period. If sending notifications to customers is set up for the merchant, the customer's email address must be passed in this tag in the email parameter. |
| clientId | AN..255 | no | Identifier of the customer in the store system. This parameter is used for the binding functionality. It can be present if creating bindings is allowed for the store.   |  | | --- | | Specifying this parameter when processing payments with the use of bindings is mandatory. Otherwise, a payment will be unsuccessful. | |
| merchantLogin | AN..255 | no | To register an order on behalf of a child merchant, specify the merchant login in this parameter. |
| features | ANS..255 | no | Container for the feature parameter, the available values for the parameter are:  AUTO\_PAYMENT - if the order registration request initiates executing auto-payments.  VERIFY - If this parameter is specified after the order registration request, the cardholder is to be verified without debiting funds from the cardholder account. Thus it is possible to pass a zero amount in the request. This verification allow the merchant to ensure that a card is used by the cardholder and to debit this card in the future without verifying authentication data (CVC, 3D-Secure) on processing subsequent payments.   |  | | --- | | **The details of passing the VERIFY value**   * Even if the payment amount is to be passed in the request, it will not be debited from the account. * After the order has been successfully registered, it is passed to the REVERSED (cancelled) status. |   Example:   |  | | --- | | <features>  <feature>AUTO\_PAYMENT</feature> </features> | |

**\*** By default, the following fields are passed to the Bank processing system:

* + - orderNumber – the order number in the store system;
    - description – the order description (no more than 99 symbols, it is forbidden to use %, +, an end of line \r and line break \n).

If the additional parameter merchantOrderId is passed in an order, its value is to be passed to the processing system of the Bank as an order number (instead of the value in the orderNumber field).

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | no | Identifier of the order in the payment system. It is unique within the system. The identifier is missing if the order registration failed due to an error detailed in errorCode. |
| formUrl | AN..512 | no | URL of the payment form to which the customer's browser is to be redirected. The URL is not returned if the registration of the order fails due to an error detailed in errorCode. |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Error description in the language passed in the language parameter in the request. |

**Error codes (the** errorCode **field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | Unknown order ID |
| 1 | An order with this identifier has already been processed |
| 3 | Unknown currency |
| 4 | Amount is missing |
| 4 | Order number cannot be empty |
| 4 | Return URL cannot empty |
| 5 | Payments using pre-authorization are forbidden |
| 5 | Incorrect value for one of the parameters |
| 5 | Access denied |
| 5 | The user must change the password |
| 7 | System error |
| 13 | Using both values, Features FORCE\_TDS/FORCE\_SSL and AUTO\_PAYMENT, is not allowed |
| 13 | The merchant does not have the permission to process auto-payments |
| 13 | The merchant does not have the permission to process verification payments |
| 14 | Features are specified incorrectly |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:registerOrderPreAuth>   <order merchantOrderNumber="asuaakdfadsfasdfasdd5" description=" " amount="10000" currency=" " language="ru" pageView="DESKTOP" bindingId=" " sessionTimeoutSecs=" ">   <returnUrl>https://web.rbsuat.com/ab/finish.html</returnUrl>   <params name="param1" value="paramValue1"/>   <params name="param2" value="paramValue2"/>   <clientId>7777</clientId>   <merchantLogin> </merchantLogin>  <features>  <feature>AUTO\_PAYMENT</feature>  </features>  </order>   </mer:registerOrderPreAuth>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:registerOrderPreAuthResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return orderId="5e5dc6bd-dee3-4c96-849a-09f3f575f4b6" errorCode="0" errorMessage="Success">   <formUrl> https://web.rbsuat.com/ab/payment\_ru.html?mdOrder=5e5dc6bd-dee3-4c96-849a-09f3f575f4b6 </formUrl>   </return>   </ns1:registerOrderPreAuthResponse>   </soap:Body>   </soap:Envelope> |

### Order payment completion request

To debit an earlier pre-authorized order, the depositOrder request is used. This operation can be executed provided that you have the corresponding permissions in the system.  
  
**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| depositAmount | N..5 | yes | The amount to be debited in the order currency. The amount can be less or equal to the pre-authorization amount. The amount cannot be less that one rouble. |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, it is considered that the language is Russian. An error message will be returned in this language. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Error description in the language passed in the Language parameter of the request. |

**ATTENTION:** If not to specify the amount parameter, the operation will be completed for the whole pre-authorization amount.  
  
  
Error codes (the errorCode fields):

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | The deposit amount exceeds the amount put on hold |
| 5 | The deposit amount must be more than zero and not less than one rouble |
| 5 | Access denied |
| 5 | The user must change the password |
| 6 | Unknown order ID |
| 6 | [orderId] is not specified |
| 7 | Payment must be in the correct status |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:depositOrder>   <order language="ru" orderId="4302d369-a5e8-4432-a5e5-42acfab52c86" depositAmount="30000">   <!- Zero or more repetitions: ->   <params name="?" value="?"/>   </order>   </mer:depositOrder>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:depositOrderResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return errorCode="6" errorMessage="Unknown order id"/>   </ns1:depositOrderResponse>   </soap:Body>   </soap:Envelope> |

### Order status request

To request the status of a registered order, the getOrderStatus request is used.

The order status must be defined by the value of the orderStatus parameter.  
The authCode field is obsolete.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, it is considered that the language is Russian. An error message will be returned in this language. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderStatus | N2 | no | The status of the order in the payment system is defined by the value of this parameter. The list of available values is presented in the table below. It is missing if the order has not been found. |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Error description in the language passed in the Language parameter of the request. |
| orderNumber | AN..32 | yes | Order identifier in the store system |
| pan | N..19 | no | Masked number of the card that has been used for the payment. This parameter is to be specified only after the order has been paid. |
| expiration | N6 | no | Card expiration . This parameter is to be specified only after the order has been paid. |
| cardholderName | A..64 | no | Name of the cardholder. This parameter is to be specified only after the order has been paid. |
| amount | N..20 | yes | Payment amount in the minor denomination (cents or kopeks) |
| currency | N3 | no | ISO 4217 code of the payment currency. If the code is not specified, the default value is 810 (Russian roubles). |
| approvalCode | AN6 | no | Code of the IPS authorization. The length of the field is fixed (six symbols), it can contain digits and Latin letters. |
| authCode | N3 | no | This field is obsolete. Its value is always equal to"2" regardless the order status and the authorization code of the processing system. |
| ip | NS..15 | no | IP address of the user who has paid the order |
| date | ANS | yes | Order registration date |
| orderDesctiption | AN..512 | yes | Order description passed on its registration |
| actionCodeDesctiption | AN..512 | yes | Explanation of the response code in the language passed in the Language parameter of the request. |
| clientId | AN..255 | no | Identifier of the customer in the store system passed on the order registration. This parameter is present only if a store has the permission to create bindings. |
| bindingId | AN..255 | no | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. |

**The** orderStatus **field can have the following values:**

|  |  |
| --- | --- |
| Status number | Description |
| 0 | The order has been registered but not paid |
| 1 | The pre-authorized amount has been put on hold (for two-phase payments) |
| 2 | Full authorization of the order amount has been performed |
| 3 | Authorization is cancelled |
| 4 | A refund operation has been processed for the transaction |
| 5 | Authorization through ACS of the issuing bank has been initiated |
| 6 | Authorization is declined |

**Error codes (the**  errorCode **field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 2 | Payment is declined |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | [orderId] is not specified |
| 6 | Unknown order ID |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:getOrderStatus>   <order orderId="b1221b79-5703-42c9-a4b1-ed0d0f36493e" language="ru"/>   </mer:getOrderStatus>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:getOrderStatusResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return orderStatus="2" errorCode="0" orderNumber="456789012345678" pan="411111\*\*1111" expiration="201512" cardholderName="Ivan" amount="15000" currency="810" approvalCode="123456" uthCode="2" ip="212.5.125.194" date="2013-09-03T16:48:52.459+04:00" orderDescription=" " actionCodeDescription="The request has been successfully processed" clientId="666" bindingId="df0b1801-a754-4576-b174-c2485d98bc9b"/>   </ns1:getOrderStatusResponse>   </soap:Body>   </soap:Envelope> |

### Extended order status request

To request the status of a registered order, the getOrderStatusExtended request is used.  
  
**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | yes\* | Order identifier in the payment system. It is unique within the system. |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, it is considered that the language is Russian. An error message will be returned in this language. |
| merchantOrderNumber | ANS..32 | yes\* | Order identifier in the store system. |

\* It is necessary to pass either the orderId ormerchantOrderNumber parameter in the request. If both parameters are passed, orderId has a higher priority.

Several sets of response parameters exist. It depends of the version of getOrderStatusExtended specified in the merchant settings, what sets of parameters are to be returned.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Mandatory | Description | Version of getOrderStatusExtended |
| orderNumber | AN..32 | yes | Order identifier in the store system. | All versions. |
| orderStatus | N2 | no | The order status in the payment system is defined by the value of this parameter. The available values are: given in the list below. It is missing if the order has not been found.   * **0** - the order is registered, but not paid; * **1** - the pre-authorized amount is put on hold (for two-phase payments); * **2** - a full authorization of the order amount has been performed; * **3** - the authorization was cancelled; * **4** - a refund operation has been processed for the transaction; * **5** - an authorization through ACS of the issuing bank has been initiated; * **6** - the authorization was declined | All versions. |
| actionCode | N3 | yes | Response code. | All versions. |
| actionCodeDescription | AN..512 | yes | Explanation of the response code in the language passed in the Language parameter of the request. | All versions. |
| errorCode | N3 | no | Error code. The following values are available.   * **0** - the request has been processed without system errors.; * **1** - [orderid] or [ordernumber] is expected; * **5** - access denied; * **5** - the user must change the password; * **6** - order is not found; * **7** - a system error. | All versions. |
| errorMessage | AN..512 | no | Error description in the language passed in the Language parameter of the request. | All versions. |
| amount | N..20 | yes | Payment amount in the minor denomination (cents or kopeks) | All versions. |
| currency | N3 | no | ISO 4217 code of the payment currency. If the code is not specified, the default value is 810 (Russian roubles). | All versions. |
| date | ANS | yes | Order registration date. | All versions. |
| orderDescription | AN..512 | no | Order description passed on its registration | All versions. |
| ip | AN..20 | yes | IP-address of the buyer. | All versions. |
| *The* merchantOrderParams *element is present in a response if an order contains additional parameters of the merchant. Each additional parameter of an order is present in a separate* merchantOrderParams *element.* | | | | |
| name | AN..20 | no | Name of the additional parameter | All versions. |
| value | AN..1024 | no | Value of the additional parameter | All versions. |
| *The* cardAuthInfo *element contains a structure that consists of a list of the* secureAuthInfo *element and the following parameters:* | | | | |
| maskedPan | N..19 | no | Masked number of the card that has been used for the payment. This parameter is to be specified only after the order has been paid. | All versions. |
| expiration | N6 | no | Card expiration . This parameter is to be specified only after the order has been paid. | All versions. |
| cardholderName | A..64 | no | Name of the cardholder. This parameter is to be specified only after the order has been paid. | All versions. |
| approvalCode | AN6 | no | Payment authorization code. The field has a fixed length (six symbols), it can contain digits and Latin letters. The parameter is specified only after the order has been paid. | All versions. |
| chargeback | A..5 | no | The parameter defines whether the funds have been forcibly returned to the buyer by the bank. The available values are:   * **true** (the funds have been refunded); * **false** (the funds have not been refunded). | 06 and higher. |
| paymentSystem | N..10 | yes | The payment system name. The available values are:   * **VISA**; * **MASTERCARD**; * **AMEX**; * **JCB**; * **CUP**; * **MIR**. | 08 and higher. |
| product | AN..255 | yes | Additional details on corporate cards. These details are filled in by the technical support service through the administrative console. If such details are missing, an empty value is returned. | 08 and higher. |
| paymentWay | AS..14 | yes | The method of order completion (a payment with entering card data, a payment using a binding, etc.). | 09 and higher. |
| *The* secureAuthInfo *element (the element consists of the* eci *and* threeDSInfo *elements that are the lists of the* cavv *and* xid *parameters):* | | | | |
| eci | N..4 | no | Electronic Commerce Indicator. The indicator is specified only after an order has been paid and in case the corresponding permission is present. | All versions. |
| cavv | ANS..200 | no | The value for the cardholder and card authentication check. The indicator is specified only after an order has been paid and in case the corresponding permission is present. | All versions. |
| xid | ANS..80 | no | Electronic Commerce Indicator of the transaction. The indicator is specified only after an order has been paid and in case the corresponding permission is present. | All versions. |
| *The* bindingInfo *element consists of these parameters:* | | | | |
| clientId | AN..255 | no | Identifier of the customer in the store system passed on the order registration. This parameter is present only if a store has the permission to create bindings. | All versions. |
| bindingId | AN..255 | no | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. | All versions. |
| authDateTime | ANS | no | The authorization date and time. | 02 and higher. |
| authRefNum | AN..24 | no | Reference number. | 02 and higher. |
| terminalId | AN..10 | no | Terminal ID. | 02 and higher. |
| *The* paymentAmountInfo *element consists of these parameters:* | | | | |
| approvedAmount | N..20 | no | The amount put on hold on the card (is used only for two-phase payments). | 03 and higher. |
| depositedAmount | N..20 | no | The amount confirmed for debiting from the card. | 03 and higher. |
| refundedAmount | N..20 | no | The refund amount. | 03 and higher. |
| paymentState | A..10 | no | The order status. | 03 and higher. |
| feeAmount | N..20 | no | The fee amount. | 11 and higher. |
| *The* bankInfo *element consists of these parameters:* | | | | |
| bankName | AN..200 | no | Name of the issuing bank. | 03 and higher. |
| bankCountryCode | AN..4 | no | Country codes of the issuing bank. | 03 and higher. |
| bankCountryName | AN..160 | no | Name of the country of the issuing bank passed in the language parameter of the request or in the language of the user who has called the method if the language has not been specified in the request. | 03 and higher. |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:getOrderStatusExtended>   <order orderId="942e8534-ac73-4e3c-96c6-f6cc448018f7" language="ru">   <!--Optional:-->   <merchantOrderNumber> </merchantOrderNumber>   </order>   </mer:getOrderStatusExtended>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:getOrderStatusExtendedResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return orderNumber="00000123" orderStatus="2" actionCode="0" actionCodeDescription="" amount="10000" currency="810" date="2016-12-20T18:48:49.590+03:00" ip="10.7.5.21" errorCode="0" errorMessage="Success">  <attributes name="mdOrder" value="08c87ec5-399d-49d0-85ed-85c94bb26675"/>  <cardAuthInfo maskedPan="555555\*\*5599" expiration="201912" cardholderName="asd asd" approvalCode="123456" paymentSystem="MASTERCARD" product="DEBIT">  <secureAuthInfo>  <eci>1</eci>  </secureAuthInfo>  </cardAuthInfo>  <authDateTime>2016-12-20T18:52:16.276+03:00</authDateTime>  <terminalId>000001</terminalId>  <authRefNum>111111111111</authRefNum>  <paymentAmountInfo paymentState="DEPOSITED" approvedAmount="10000" depositedAmount="10000" refundedAmount="0"/>  <bankInfo bankName="Unated State Bank of Atlanta" bankCountryCode="US" bankCountryName="The Unated States of America"/>  <chargeback>false</chargeback>  <paymentWay>CARD</paymentWay>  </return>  </ns1:getOrderStatusExtendedResponse>  </soap:Body> </soap:Envelope> |

### Order payment cancellation request

To cancel an order, the reverseOrder request is used. The cancellation function is available within a limited time period after a payment has been processed. The exact duration should be found out from the bank.

The cancellation operation can be executed only once. If it ends with an error, a repetitive cancellation operation will not be processed.

This functionality is available for a merchant upon agreement with the Bank. To execute a cancellation operation, a user needs to have the corresponding permissions.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, it is considered that the language is Russian. An error message will be returned in this language. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Error description in the language passed in the Language parameter of the request. |

Error codes (the errorCode fields):

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | [orderId] is not specified |
| 6 | Unknown order ID |
| 7 | Invalid operation for the current status of the order |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:reverseOrder>   <order language="ru" orderId="f88a2bbf-2021-4ccc-8783-8a13068a89f9">   <!- Zero or more repetitions: ->   <params name=" " value=" "/>   </order>   </mer:reverseOrder>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:reverseOrderResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return errorCode="7" errorMessage="Invalid operation for the current order status"/>   </ns1:reverseOrderResponse>   </soap:Body>   </soap:Envelope> |

### Order payment refund request

To refund funds, the refundOrder request is used.

Upon this request, the funds for the specified order are to be returned to the payer. The request will end with an error if the funds have not been debited for this order. The system permits returning funds more than once, but for a total amount not exceeding the initial debit amount.

|  |
| --- |
| When processing a refund for a payment for housing and utilities services, only a full refund is available. |

To process a refund operation, it is necessary to have the corresponding permissions in the system.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| refundAmount | N..5 | yes | The refund amount in the order currency. The amount can be equal to or less than the order remaining amount. |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, it is considered that the language is Russian. An error message will be returned in this language. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Error description in the language passed in the Language parameter of the request. |

Error codes (the errorCode fields):

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | [orderId] is not specified |
| 5 | Incorrect amount |
| 6 | Unknown order ID |
| 7 | Payment must be in the correct status |
| 7 | The refund amount exceeds the debited amount |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:refundOrder>   <order language="ru" orderId="4302d369-a5e8-4432-a5e5-42acfab52c86" refundAmount="20000">   <!--Zero or more repetitions:-->   <params name=" " value=" "/>   </order>   </mer:refundOrder>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:refundOrderResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return errorCode="7" errorMessage="The refund amount exceeds the debited amount/>   </ns1:refundOrderResponse>   </soap:Body>   </soap:Envelope> |

### Request for checking a card for 3D-Secure enrolment

The verifyEnrollment request is used to check a card for enrolment to 3D-Secure.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| pan | N12...19 | yes | Card number |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Error description. |
| isEnrolled | A1 | no | The flag signifying enrolment of the card to 3D-Secure. The available values are: Y, N, U. |
| emitterName | AN..160 | no | Name of the issuing bank. |
| emitterCountryCode | AN..4 | no | Country codes of the issuing bank. |

**Error codes (the ErrorCode field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | The card number is not specified |
| 1 | The card number must be a number containing 13 to 19 digits |
| 5 | Access denied |
| 5 | The user must change the password |
| 6 | No information is found for the specified card number |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:verifyEnrollment>   <pan>4111111111111111</pan>   </mer:verifyEnrollment>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:verifyEnrollmentResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return isEnrolled="Y" emitterName="TEST CARD" emitterCountryCode="RU" errorCode="0"/>   </ns1:verifyEnrollmentResponse>   </soap:Body>   </soap:Envelope> |

### Request for adding additional parameters to an order

The addParams method is used to add additional parameters to an order.

If an additional parameter already exists in the order, the value most recently passed is saved to the order when adding a parameter with the same name.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| *The additional parameters block - params:* |  |  |  |
| name | AN..20 | yes | Name of the additional parameter |
| value | AN..1024 | yes | Value of the additional parameter |

**Response parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | yes | Error code. |
| errorMessage | AN..512 | no | Error description. It is missing when a request is processed unsuccessfully. |

**Error codes (the ErrorCode field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | Access denied |
| 5 | The user must change the password |
| 6 | orderId is not specified |
| 6 | Unknown order ID |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">  <soapenv:Header/>  <soapenv:Body>  <mer:addParams>  <request orderId="02988563-0cc4-4109-84ed-15781f0d718e">  <!--Zero or more repetitions:-->  <params name="Param1" value="Param1"/>  <params name="Param2" value="Param2"/>  </request>  </mer:addParams>  </soapenv:Body> </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:addParamsResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return errorCode="0"/>  </ns1:addParamsResponse>  </soap:Body> </soap:Envelope> |

### Request for payments statistics for a period

The getLastOrdersForMerchants method allows you to get the payment statistics for a particular period.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, it is considered that the language is Russian. An error message will be returned in this language. |
| page | N | no | When processing the request, a list is formed that is broken down into pages (with the number of records on each page equal to size). A page with the number that was specified in the page parameter is returned in the response. The pages numbering starts from 0. If the parameter is not specified, a page with the number 0 is returned. |
| size | N..3 | yes | Number of elements on a page (the maximum value = 200). |
| c | ANS | yes | The date and time of the beginning of the period for which the orders to be collected, in the format YYYY-MM-DDTHH:mm:ss. |
| по | ANS | yes | Date and time of the ending of the period for which the orders to be collected, in the format YYYY-MM-DDTHH:mm:ss. |
| transactionStates | A..9 | yes | In this block, it is necessary to list the required order statuses. Only orders in one of the specified statuses are included into the report. The available values are: CREATED, APPROVED, DEPOSITED, DECLINED, REVERSED, REFUNDED. |
| merchants | ANS | yes | List of logins of the merchant whose transactions are included into the report.  Leave this block empty to get a list of reports on all available merchants (child merchants and merchants specified in the user's settings). |
| searchByCreatedDate | boolean | no | The possible values are:   * true – a search for orders that have the creation date that falls into the specified period. * false – a search for orders that have the payment date that falls into the specified period (thus, orders in the CREATED or DECLINED status cannot be present in the report).   The default value is false. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N..2 | yes | Error code. The description of the available codes is presented below in the Error codes (the errorCode field) table |
| errorMessage | AN..512 | no | Error description. It is present only an error is present (errorCode is not equal to 0). |
| orderStatuses |  |  | Blocks that contain the information on the orders included into the report. See the orderStatuses block parameters table below. |
| totalCount | N | yes | Total number of elements in the report (on all pages). |
| page | N | yes | Number of the current page (it is equal to the page number passed in the request). |
| pageSize | N..3 | yes | Maximum number of records on a page (it is equal to the page number passed in the request). |

Parameters of the orderStatuses block:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderNumber | AN..32 | yes | Order identifier in the store system. |
| orderStatus | N..2 | yes | Status of the order in the payment system. The available values are presented in the orderStatus field table below. |
| actionCode | N..3 | yes | Response code. |
| actionCodeDescription | AN..512 | yes | Meaning of the response code. |
| amount | N..20 | yes | Payment amount in the minor denomination of the currency. |
| currency | N3 | yes | ISO 4217 code of the payment currency. If it is not specified, it is considered to be equal to the default currency value. |
| date | ANS | yes | Order registration date. |
| orderDescription | AN..512 | no | Order description passed on its registration |
| ip | AN..20 | no | IP-address of the buyer. It is specified only after a payment. |
| errorCode | N..2 | yes | Error code. |
| merchantOrderParams |  | no | Tag containing attributes that contain additional merchant parameters. See the merchantOrderParams block parameters table below. |
| attributes |  | yes | Attributes of the order in the payment system (order number). See the attributes block parameters table below. |
| cardAuthInfo |  | no | A tag containing the payment attributes. See the cardAuthInfo block parameters table below. |
| bindingInfo |  | no | Tag containing the information on the binding with which the payment is performed. See the bindingInfo block parameters table below. |
| authDateTime | ANS | no | Authorization date and time |
| terminalId | AN..10 | no | Terminal ID |
| authRefNum | AN..24 | no | Reference number |
| paymentAmountInfo |  | no | Tag containing the information on the confirmation amount, debit amount, and refund amount. See the paymentAmountInfo block parameters table below. |
| bankInfo |  | no | Tag containing the information on the issuing bank. See the bankInfo block parameters table below. |

merchantOrderParams block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| name | AN..20 | yes | Name of the additional merchant parameter |
| value | AN..1024 | yes | Value of the additional merchant parameter |

attributes block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| name | A7 | yes | Attribute name is "mdOrder". |
| value | ANS36 | yes | Attribute value is the order number in the payment system (is unique within the system). |

cardAuthInfo block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| maskedPan | N..19 | no | Masked number of the card that has been used for the payment. |
| expiration | N6 | no | Card expiration . |
| cardholderName | A..64 | no | Name of the cardholder. |
| approvalCode | AN6 | no | Payment authorization code. The field has a fixed length (six symbols), it can contain digits and Latin letters. |

bindingInfo block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| clientId | AN..255 | no | Identifier of the customer in the store system. |
| bindingId | AN..255 | no | Identifier of the binding used for the payment. |

paymentAmountInfo block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| paymentState | N..9 | no | Payment status |
| approvedAmount | N..20 | no | Amount confirmed to be debited. |
| depositedAmount | N..20 | no | Amount confirmed debited from the card. |
| refundedAmount | N..20 | no | The refund amount. |

bankInfo block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| bankName | AN..200 | no | Name of the issuing bank. |
| bankCountryCode | AN..4 | no | Code of the issuing bank country |
| bankCountryName | AN..160 | no | Name of the country of the issuing bank passed in the language parameter of the request or in the language of the user who has called the method if the language has not been specified in the request. |

The orderStatus field can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 0 | The order has been registered but not paid |
| 1 | The pre-authorized amount has been put on hold (for two-phase payments) |
| 2 | Full authorization of the order amount has been performed |
| 3 | Authorization is cancelled |
| 4 | A refund operation has been processed for the transaction |
| 5 | Authorization through ACS of the issuing bank has been initiated |
| 6 | Authorization is declined |

**Error codes (the** errorCode **field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | One of the mandatory fields is not filled in |
| 5 | Incorrect format of the transactionStates parameter |
| 7 | System error |
| 10 | Value of the size parameter exceeds the maximum allowed value |
| 10 | Insufficient permissions to view transactions for the specified merchant |

**Request example:**

|  |
| --- |
| <soapenv:Envelope  xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"  xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:getLastOrdersForMerchants>   <request language="ru">   <page>0</page>   <size>200</size>   <from>2014-10-10T00:00:00</from>   <to>2014-11-11T00:00:00</to>   <transactionStates>   <transactionStates>DEPOSITED</transactionStates>   <transactionStates>REVERSED</transactionStates>   </transactionStates>   <merchants>   <merchants>SevenEightNine</merchants>   </merchants>   <searchByCreatedDate>false</searchByCreatedDate>  </request>   </mer:getLastOrdersForMerchants>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:getLastOrdersForMerchantsResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return errorCode="0">   <orderStatuses orderNumber="58drs0Pes459Hdsddd0567a0" orderStatus="2" actionCode="0" actionCodeDescription="The request has been successfully processed" amount="250000" currency="810" date="2014-10-28T12:40:49.233+04:00" orderDescription="Opisanie" ip="212.5.125.194" errorCode="0">   <merchantOrderParams name="registr1" value="registr1"/>   <attributes name="mdOrder" value="f1a3365b-542c-4c8d-b34c-e9a7ee8dbc9c"/>   <cardAuthInfo maskedPan="411111\*\*1111" expiration="201512" cardholderName="Ivan" approvalCode="123456"/>   <bindingInfo clientId="666" bindingId="1eabfb8e-b90e-4dc8-bef6-14bd392b1cec"/>   <authDateTime>2014-10-28T12:41:01.207+04:00</authDateTime>   <terminalId>111113</terminalId>   <authRefNum>111111111111</authRefNum>   <paymentAmountInfo paymentState="DEPOSITED" approvedAmount="250000" depositedAmount="250000" refundedAmount="0"/>  <bankInfo bankName="TEST CARD" bankCountryCode="RU" bankCountryName="Russia"/>   </orderStatuses>   <orderStatuses orderNumber="57drs0Pes459Hdsddd0567a0" orderStatus="2" actionCode="0" actionCodeDescription="Request processed successfully" amount="250000" currency="810" date="2014-10-28T12:34:37.286+04:00" orderDescription="Opisanie" ip="212.5.125.194" errorCode="0">   <merchantOrderParams name="registr1" value="registr1"/>   <attributes name="mdOrder" value="09489184-bc5e-44a7-b6c4-3ca1feb8ef69"/>   <cardAuthInfo maskedPan="411111\*\*1111" expiration="201512" cardholderName="Ivan" approvalCode="123456"/>   <bindingInfo clientId="666" bindingId="1eabfb8e-b90e-4dc8-bef6-14bd392b1cec"/>   <authDateTime>2014-10-28T12:34:56.046+04:00</authDateTime>   <terminalId>111113</terminalId>   <authRefNum>111111111111</authRefNum>   <paymentAmountInfo paymentState="DEPOSITED" approvedAmount="250000" depositedAmount="250000" refundedAmount="0"/>  <bankInfo bankName="TEST CARD" bankCountryCode="RU" bankCountryName="Russia"/>   </orderStatuses>   <totalCount>2</totalCount>   <page>0</page>   <pageSize>200</pageSize>   </return>   </ns1:getLastOrdersForMerchantsResponse>   </soap:Body>   </soap:Envelope> |

### Request for a payment through an external payment network

To pay the order through an external processing system the paymentOrderOtherWay request with special parameters is used.

This operation is available provided that the merchant has the corresponding permissions in the system.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | yes | Order number received on its registration |
| paymentWay | ANS..\* | yes | Payment method is passed in this parameter. The possible values are:   * ALFA\_ALFACLICK – for a payment with "Alfa-click" (through the PayByClik system). * UPOP – for a payment through the UPOP system, for the holders of the China Union Pay cards. |
| ip | AS..15 | no | IP-address of the payer |
| language | A2 | no | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N1 | yes | Error code |
| errorMessage | ANS..\* | (on an error) | Error message |
| redirect | ANS..\* | no | Return address after the payment |

Error codes (the errorCode fields):

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | [orderId] is not specified |
| 1 | [paymentWay] is not set |
| 2 | Order is not found |
| 5 | Session timeout |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | paymentWay is incorrect |
| 5 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:paymentOrderOtherWay>   <order language="ru" orderId="8232a33f-c44f-48ec-b52f-0d63a88c50ae" paymentWay="ALFA\_ALFACLICK" ip=" "/>   </mer:paymentOrderOtherWay>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:paymentOrderOtherWayResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return redirect="http://217.12.96.193/PayByClick/login.xhtml?orderId=4e6b383e-809d-4e7e-9477-956e9c828934&amp;backUrl=http%3A%2F%2Fya.ru%3ForderId%3D4e6b383e-809d-4e7e-9477-956e9c828934" errorCode="0"/>  </ns1:paymentOrderOtherWayResponse>  </soap:Body> </soap:Envelope> |

### Request for processing a payment by a binding

To process a payment with a binding, the paymentOrderBinding request is used.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| mdOrder | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| bindingId | ANS36 | yes | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. |
| params |  | no | Additional tag containing attributes to pass additional merchant parameters.\* |
| language | A2 | no | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used |
| ip | NS..15 | yes | IP-address of the payer |
| cvc | N4 | no | CVV code. This parameter is mandatory, if "Can process payments without confirmation of CVC" is not selected for a merchant. |
| email | ANS..\* | no | Email-address of the payer |

\* Additional information fields for the subsequent storage. These fields can be passed to the Bank processing system to subsequently be displayed in registers.\*\*

Enabling the functionality is possible upon agreement with the Bank during the integration period. To pass N parameters, a request must contain N Params tags, where the name attribute contains the parameter name and value contains its value:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| name | AN..20 | yes | Name of the additional parameter |
| value | AN..1024 | yes | Value of the additional parameter |

\*\* *By default, orderNumber and its description (not more than 99 symbols, it is prohibited to use %, +, end of line \r and line break \n) are passed to the Bank processing system.*

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| redirect | ANS..\* | no | On a successful response. In case of an SSL-payment, the redirection URL after a payment. In case of a 3D-Secure payment, the URL to return to ACS. |
| info | ANS..\* | no | On a successful response. Result of a payment attempt. The available values are presented below:   * Your payment has been processed, redirecting.. * Payment declined. Check the entered data, ensure that there are enough funds on the card. Redirecting... * The payment cannot be processed. Redirecting... * Payment declined. Contact the merchant. Redirecting... * Payment declined. Contact the bank that issued the card. Redirecting... * Operation is impossible. Cardholder authentication completed unsuccessfully. Redirecting... * No connection to the bank. Try again later. Redirecting... * Data entering timeout expiration. Redirecting... * No response from the bank. Try again later. Redirecting... |
| errorCode | N1 | yes | Expiration code. |
| errorMessage | ANS..\* | no | On a response with an error. Error message. |
| acsUrl | ANS..\* | no | On a successful response in case of a 3D-Secure payment. URL-address of the redirection to ACS |
| paReq | ANS..\* | no | On a successful response in case of a 3D-Secure payment. Payment Authentication Request. |

Error codes (the errorCode field):

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | It is necessary to specify the CVC2/CVV2 code because the merchant does not have a permission to process a payment without CVC |
| 1 | Incorrect format of CVC |
| 1 | [mdOrder] is not specified |
| 1 | [bindingId] is not specified |
| 1 | The email address does not meet the template format |
| 1 | Incorrect language |
| 2 | Order is not found |
| 2 | The binding is not found |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | The maximum number of payments attempted run out or the session timeout expired |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:paymentOrderBinding>   <order mdOrder="9213bc5f-5d5b-43d6-a408-b6b93cdde992" bindingId="ca91a4ab-b6d4-495d-b606-8fb0114e679e" language="ru" ip="127.0.0.1" cvc="123" email=" ">   <!-Zero or more repetitions:->   <params name=" " value=" "/>   </order>   </mer:paymentOrderBinding>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:paymentOrderBindingResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return errorCode="0" info="Your payment has been processed, redirecting..." redirect="http://example.ru?orderId=9213bc5f-5d5b-43d6-a408-b6b93cdde992"/"/>   </ns1:paymentOrderBindingResponse>   </soap:Body>   </soap:Envelope> |

### Request for a binding deactivation

The unBindCard request is used to disable an existing active binding.

**Request parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| bindingId | ANS36 | yes | Identifier of a binding created on an order payment or used to pay for an order. It is present only if a store has the permission to create bindings. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N1 | yes | Completion code |
| errorMessage | ANS..\* | (on an error) | Error message |

**Error codes (the errorCode fields):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 2 | Incorrect binding status (when attempting to deactivate an inactive binding) |
| 2 | The binding is not found |
| 5 | Access denied |
| 5 | The user must change the password |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:unBindCard>   <bindingId>fd3afc57-c6d0-4e08-aaef-1b7cfeb093dc</bindingId>   </mer:unBindCard>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:unBindCardResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return errorCode="0" errorMessage="Success"/>   </ns1:unBindCardResponse>   </soap:Body>   </soap:Envelope> |

### Request for a binding activation

To activate a binding that has not been deactivated earlier, the bindCard request is used.

**Request parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| bindingId | ANS36 | yes | Identifier of a binding created on an order payment or used to pay for an order. It is present only if a store has the permission to create bindings. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N1 | yes | Completion code |
| errorMessage | ANS..\* | (on an error) | Error message |

**Error codes (the errorCode fields):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 2 | Incorrect binding status (when attempting to deactivate an inactive binding) |
| 2 | The binding is not found |
| 5 | Access denied |
| 5 | The user must change the password |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:bindCard>   <bindingId>fd3afc57-c6d0-4e08-aaef-1b7cfeb093dc</bindingId>   </mer:bindCard>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:bindCardResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return errorCode="5" errorMessage="The user must change the password"/>   </ns1:bindCardResponse>   </soap:Body>   </soap:Envelope> |

### Request for changing the validity period of a binding

Use the extendBinding method to change the validity period of a binding.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| bindingId | ANS36 | yes | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. |
| newExpiry | N6 | yes | A new date (the year and month) of the expiration of the validity period, in the format YYYYMM |
| language | А2 | no | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N1 | yes | Completion code |
| errorMessage | ANS..\* | (on an error) | Error message |

**Error codes (the errorCode fields):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | One (or several) of the mandatory parameters is not specified or is specified incorrectly |
| 2 | The binding is not found |
| 5 | Access denied |
| 5 | The user must change the password |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">  <soapenv:Header/>  <soapenv:Body>  <mer:extendBinding>  <request language=" " bindingId="1eabfb8e-b90e-4dc8-bef6-14bd392b1cec" newExpiry="201807"/>  </mer:extendBinding>  </soapenv:Body> </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:extendBindingResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return errorCode="0" errorMessage="Success"/>  </ns1:extendBindingResponse>  </soap:Body> </soap:Envelope> |

### Request for the list of bindings of a customer

To get a list of bindings by the identifier of a customer, the getBindings request is used.

**Request parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| clientId | AN..255 | yes | Identifier of the customer in the store system passed on the order registration. This parameter is present only if a store has the permission to create bindings. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N1 | yes | Completion code |
| errorMessage | ANS..\* | (on an error) | Error message |
| *Binding element (it consists of bindingId, maskedPan and expiryDate):* |  |  |  |
| bindingId | AN..255 | no | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. |
| maskedPan | N..19 | no | Masked number of the card that has been used for the payment. This parameter is to be specified only after the order has been paid. |
| expiryDate | N6 | no | Card expiration . This parameter is to be specified only after the order has been paid. |

Error codes (the errorCode fields):

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | [clientId] is not specified |
| 2 | The information is not found |
| 5 | Access denied |
| 5 | The user must change the password |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">   <soapenv:Header/>   <soapenv:Body>   <mer:getBindings>   <request clientId="client"/>   </mer:getBindings>   </soapenv:Body>   </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">   <soap:Body>   <ns1:getBindingsResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">   <return errorCode="0" errorMessage="Success">   <bindings>   <binding bindingId="fd3afc57-c6d0-4e08-aaef-1b7cfeb093dc" maskedPan="400000\*\*0002" expiryDate="201512"/>   </bindings>   </return>   </ns1:getBindingsResponse>   </soap:Body>   </soap:Envelope> |

### Request for the list of bindings of a bank card

Provided that a store has the corresponding permissions, it can get the list of all bindings that relate to a certain bank card. This can be done by a card number or by a known binding identifier.

The getBindingsByCardOrId method is used to get the list of bindings for a bank card.

All bindings that are available to the merchant according to the merchant settings are returned in the response.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| pan | N..19 | no | Card number. The parameter is mandatory, unless bindingId is specified.  A search by the full number of a card is available to stores only provided that they have the corresponding permission. |
| bindingId | AN..255 | no | Binding identifier. The parameter is mandatory, unless pan is specified.  If the request contains pan, the bindingId value is ignored. |
| showExpired | boolean | no | This parameter defines the necessity to display bindings with expired card validity periods. The available values are: true, false. By default the parameter has the false value. |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, the default language specified in the merchant settings is used. An error message will be returned in this language. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N1 | yes | Expiration code. |
| errorMessage | ANS..\* | yes | Description of the completion code. |
| *Bindings element (can consist of several binding elements)* |  |  |  |
| *Binding element (consists of bindingId, maskedPan, expiryDate and clientId):* |  |  |  |
| bindingId | AN..255 | no | Binding identifier. |
| maskedPan | N..19 | no | Masked number of the card that has been used for the payment. |
| expiryDate | N6 | no | The expiration date of the card validity period, in the format YYYYMM. |
| clientId | AN..255 | no | Number (identifier) of a customer in the merchant system. |

**Error codes (the** errorCode **field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | Neither a card number, nor a binding identifier is specified. |
| 2 | The information is not found. |
| 5 | Access denied. |
| 5 | The user must change the password. |
| 7 | System error. |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">  <soapenv:Header/>  <soapenv:Body>  <mer:getBindingsByCardOrId>  <request pan="4111111111111111" bindingId=" " showExpired="true" language="ru"/>  </mer:getBindingsByCardOrId>  </soapenv:Body> </soapenv:Envelope> |

**Response parameters:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:getBindingsByCardOrIdResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return errorCode="0" errorMessage="Success">  <bindings>  <binding bindingId="9f43e86d-9744-42a5-8646-c4bb7cf8799a" maskedPan="411111\*\*1111" expiryDate="201912" clientId="12345"/>  <binding bindingId="51c0750b-1a23-424b-8989-5e8f669124b4" maskedPan="411111\*\*1111" expiryDate="201512" clientId="123456"/>  <binding bindingId="6a8c0738-cc88-4200-acf6-afc264d66cb0" maskedPan="411111\*\*1111" expiryDate="201912" clientId="666"/>  <binding bindingId="97a70989-c1fb-49f7-8a42-27c19dc160dw" maskedPan="411111\*\*1111" expiryDate="201512" clientId="666"/>  </bindings>  </return>  </ns1:getBindingsByCardOrIdResponse>  </soap:Body> </soap:Envelope> |

### Request for adding a card to the list of SSL-cards

To add the number of a card that was used on an attempt to pay an order to the list of SSL-cards, the updateSSLCardList request is used.

The method is available only upon agreement with the bank. The user must be assigned the permissions to work with fraud lists.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Description of the error in the default language of the user. |

**Error codes (the** errorCode **field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | Access denied |
| 2 | The user must change the password |
| 3 | Order number is not specified |
| 4 | Payment is absent from the system or an incorrect mdorder is specified |
| 6 | The card number is already present in the list |
| 7 | System error |

**Request example:**

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">  <soapenv:Header/>  <soapenv:Body>  <mer:updateSSLCardList>  <mdorder>e40927ef-6fa2-43f3-bf97-f1f2bc61d4fd</mdorder>  </mer:updateSSLCardList>  </soapenv:Body> </soapenv:Envelope> |

**Response example:**

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:updateSSLCardListResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return errorCode="0" errorMessage="Request processing took place without system errors"/>  </ns1:updateSSLCardListResponse>  </soap:Body> </soap:Envelope> |

### Request for a payment through Apple Pay

The applePay request is used to process a payment through Apple Pay.

|  |
| --- |
| Use the standard requests to the payment gateway for the operations of cancellation, refund and payment completion. |

An example of the request is given below.

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">  <soapenv:Header/>  <soapenv:Body>  <mer:applePay>  <arg0>  <merchant>OurBestMerchantLogin</merchant>  <orderNumber>UAF-203974-DE</orderNumber>  <description>Test description</description>  <paymentToken></paymentToken>  <language>RU</language>  <additionalParameters>  <entry>  <key>firstParamName</key>  <value>firstParamValue</value>  </entry>  </additionalParameters>  <preAuth>true</preAuth>  <ip>127.0.0.1</ip>  </arg0>  </mer:applePay>  </soapenv:Body> </soapenv:Envelope> |

The description of the request is given in the table below.

|  |  |  |
| --- | --- | --- |
| Parameter | Mandatory | Description |
| merchant | No | Merchant login in the payment gateway system. |
| orderNumber | No | Unique identifier of the order on the merchant side. |
| description | No | Order description. |
| paymentToken | Yes | The paymentToken parameter must contain a Base64 encoded value of the paymentData property that was received in PKPaymentToken Object from the Apple Pay system (see [the documentation for Apple Pay](https://developer.apple.com/library/ios/documentation/PassKit/Reference/PaymentTokenJSON/PaymentTokenJSON.html)). Thus, to send a payment request to the payment gateway, the merchant must:   1. Receive from the Apple Pay system the PKPaymentToken Object object containing the paymentData property; 2. Extract the value of the paymentData property and encode it in Base64; 3. Include the encoded value of the paymentData property as the value of the paymentToken parameter in the payment request that the merchant sends to the payment gateway.   Thus, the merchant receives from Apple PKPaymentToken Object that looks as follows:   |  | | --- | | {"paymentData":{"data":"vj5Uvux7Im8DD8YhSOsJvw5lWmfl2HMUnTNWJhVfTehvFffRhDo54mfpjxMt9vJdp6DwD7fgcNHDxBvnj56qYG4DpOxg1fTSdXgPFrezprZHCrRxPhN\/aQQEThe2pQ0c7hgzzZlA6TpkIR\/Xtk6CTcEbD1W6znFVdvMgX8G96Gg4OAGl8GaTXdSU3wlMQL5E63CLQzPi1xHVErWl1OOn6hYQuREUDGc7mAjmqMyLwX+p6mOwJZ6ZFO\/b9HkgFi428rqtOH08AfqkfaIWwIIAz2w3xEoZrDXbgFpNBnN7F2oretCU1\/dFvQJjDYbMorKQ8+GJbWtlsVb+Ksy0U91eoUetDcyMpB9zc139STYVoC8yp6Yk6Mn3icCLY0ZBujq7\/404kMGpnHgkNVqFc\/4SN0U2XQ5rrb14DM8M69w=","signature":"\/0komJPnwPE6OCAhEwggINMEUGCCsGAQUFBwEBBDkwNzA1BggrBgEFBQcwAYYpaHR0cDovL29jc3AuYXBwbGUuY29tL29jc3AwNC1hcHBsZWFpY2EzMDEwHQYDVR0OBBYEFJRX22\/VdIGGiYl2L35XhQfnm1gkMAwGA1UdEwEB\/wQCMAAwHwYDVR0jBBgwFoAUI\/JJxE+T5O8n5sT2KGw\/orv9LkswggEdBgNVHSAEggEUMIIBEDCCAQwGCSqGSIb3Y2QFATCB\/jCBwwYIKwYBBQUHAgIwgbYMgbNSZWxpYW5jZSBvbiB0aGlzIGNlcnRpZmljYXRlIGJ5IGFueSBwYXJ0eSBhc3N1bWVzIGFjY2VwdGFuY2Ugb2YgdGhlIHRoZW4gYXBwbGljYWJsZSBzdGFuZGFyZCB0ZXJtcyBhbmQgY29uZGl0aW9ucyBvZiB1c2UsIGNlcnRpZmljYXRlIHBvbGljeSBhbmQgY2VydGlmaWNhdGlvbiBwcmFjdGljZSBzdGF0ZW1lbnRzLjA2BggrBgEFBQcCARYqaHR0cDovL3d3dy5hcHBsZS5jb20vY2VydGlmaWNhdGVhdXRob3JpdHkvMDQGA1UdHwQtMCswKaAnoCWGI2h0dHA6Ly9jcmwuYXBwbGUuY29tL2FwcGxlYWljYTMuY3JsMA4GA1UdDwEB\/wQEAwIHgDAPBgkqhkiG92NkBh0EAgUAMAoGCCqGSM49BAMCA0gAMEUCIHKKnw+Soyq5mXQr1V62c0BXKpaHodYu9TWXEPUWPpbpAiEAkTecfW6+W5l0r0ADfzTCPq2YtbS39w01XIayqBNy8bEwggLuMIICdaADAgECAghJbS+\/OpjalzAKBggqhkjOPQQDAjBnMRswGQYDVQQDDBJBcHBsZSBSb290IENBIC0gRzMxJjAkBgNVBAsMHUFwcGxlIENlcnRpZmljYXRpb24gQXV0aG9yaXR5MRMwEQYDVQQKDApBcHBsZSBJbmMuMQswCQYDVQQGEwJVUzAeFw0xNDA1MDYyMzQ2MzBaFw0yOTA1MDYyMzQ2MzBaMHoxLjAsBgNVBAMMJUFwcGxlIEFwcGxpY2F0aW9uIEludGVncmF0aW9uIENBIC0gRzMxJjAkBgNVBAsMHUFwcGxlIENlcnRpZmljYXRpb24gQXV0aG9yaXR5MRMwEQYDVQQKDApBcHBsZSBJbmMuMQswCQYDVQQGEwJVUzBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABPAXEYQZ12SF1RpeJYEHduiAou\/ee65N4I38S5PhM1bVZls1riLQl3YNIk57ugj9dhfOiMt2u2ZwvsjoKYT\/VEWjgfcwgfQwRgYIKwYBBQUHAQEEOjA4MDYGCCsGAQUFBzABhipodHRwOi8vb2NzcC5hcHBsZS5jb20vb2NzcDA0LWFwcGxlcm9vdGNhZzMwHQYDVR0OBBYEFCPyScRPk+TvJ+bE9ihsP6K7\/S5LMA8GA1UdEwEB\/wQFMAMBAf8wHwYDVR0jBBgwFoAUu7DeoVgziJqkipnevr3rr9rLJKswNwYDVR0fBDAwLjAsoCqgKIYmaHR0cDovL2NybC5hcHBsZS5jb20vYXBwbGVyb290Y2FnMy5jcmwwDgYDVR0PAQH\/BAQDAgEGMBAGCiqGSIb3Y2QGAg4EAgUAMAoGCCqGSM49BAMCA2cAMGQCMDrPcoNRFpmxhvs1w1bKYr\/0F+3ZD3VNoo6+8ZyBXkK3ifiY95tZn5jVQQ2PnenC\/gIwMi3VRCGwowV3bF3zODuQZ\/0XfCwhbZZPxnJpghJvVPh6fRuZy5sJiSFhBpkPCZIdAAAxggGMMIIBiAIBATCBhjB6MS4wLAYDVQQDDCVBcHBsZSBBcHBsaWNhdGlvbiBJbnRlZ3JhdGlvbiBDQSAtIEczMSYwJAYDVQQLDB1BcHBsZSBDZXJ0aWZpY2F0aW9uIEF1dGhvcml0eTETMBEGA1UECgwKQXBwbGUgSW5jLjELMAkGA1UEBhMCVVMCCCRD8qgGnfV3MA0GCWCGSAFlAwQCAQUAoIGVMBgGCSqGSIb3DQEJAzELBgkqhkiG9w0BBwEwHAYJKoZIhvcNAQkFMQ8XDTE3MDMxNzEwMzgzOVowKgYJKoZIhvcNAQk0MR0wGzANBglghkgBZQMEAgEFAKEKBggqhkjOPQQDAjAvBgkqhkiG9w0BCQQxIgQgvL+q07\/reM0N\/5b0hwWT7TJReVTdS9QX5SPhiqeie+cwCgYIKoZIzj0EAwIERzBFAiEAttC68Xyzs6I0+tAKmg6x+0UrqmkQN\/V5c8RMMIEJHooCIHIgUHbAt2p5WrFHQKrAVL4c7nohRplZWVbVu6wbBeCgAAAAAAAA","header":{"publicKeyHash":"fpvAnSDwQFX4NX4pghdjpNwUFhoTH\/DDGhew94uJaRA=","ephemeralPublicKey":"MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAErKZUfqvhlieGAOaCKeTB\/oDEo29fS1jWSKemNDh3fIqmbfs86nL4BGtRsWRxWcMnHN6GFOQm1MEj4m7ZHxe78g==","transactionId":"38e4c267ef1de62a343d0eccada3f7e19f6b22ffc7ede899c039865432ba6aa2"},"version":"EC\_v1"},"transactionIdentifier":"38E4C267EF1DE62A343D0ECCADA3F7E19F6B22FFC7EDE899C039865432BA6AA2","paymentMethod":{"network":"Visa","type":"debit","displayName":"Visa 5223"}} |   The value of the paymentData property (from the example above) that is to be encoded in Base64 and to be passed in the payment request to the payment gateway looks as follows:   |  | | --- | | {"data":"vj5Uvux7Im8DD8YhSOsJvw5lWmfl2HMUnTNWJhVfTehvFffRhDo54mfpjxMt9vJdp6DwD7fgcNHDxBvnj56qYG4DpOxg1fTSdXgPFrezprZHCrRxPhN\/aQQEThe2pQ0c7hgzzZlA6TpkIR\/Xtk6CTcEbD1W6znFVdvMgX8G96Gg4OAGl8GaTXdSU3wlMQL5E63CLQzPi1xHVErWl1OOn6hYQuREUDGc7mAjmqMyLwX+p6mOwJZ6ZFO\/b9HkgFi428rqtOH08AfqkfaIWwIIAz2w3xEoZrDXbgFpNBnN7F2oretCU1\/dFvQJjDYbMorKQ8+GJbWtlsVb+Ksy0U91eoUetDcyMpB9zc139STYVoC8yp6Yk6Mn3icCLY0ZBujq7\/404kMGpnHgkNVqFc\/4SN0U2XQ5rrb14DM8M69w=","signature":"\/0komJPnwPE6OCAhEwggINMEUGCCsGAQUFBwEBBDkwNzA1BggrBgEFBQcwAYYpaHR0cDovL29jc3AuYXBwbGUuY29tL29jc3AwNC1hcHBsZWFpY2EzMDEwHQYDVR0OBBYEFJRX22\/VdIGGiYl2L35XhQfnm1gkMAwGA1UdEwEB\/wQCMAAwHwYDVR0jBBgwFoAUI\/JJxE+T5O8n5sT2KGw\/orv9LkswggEdBgNVHSAEggEUMIIBEDCCAQwGCSqGSIb3Y2QFATCB\/jCBwwYIKwYBBQUHAgIwgbYMgbNSZWxpYW5jZSBvbiB0aGlzIGNlcnRpZmljYXRlIGJ5IGFueSBwYXJ0eSBhc3N1bWVzIGFjY2VwdGFuY2Ugb2YgdGhlIHRoZW4gYXBwbGljYWJsZSBzdGFuZGFyZCB0ZXJtcyBhbmQgY29uZGl0aW9ucyBvZiB1c2UsIGNlcnRpZmljYXRlIHBvbGljeSBhbmQgY2VydGlmaWNhdGlvbiBwcmFjdGljZSBzdGF0ZW1lbnRzLjA2BggrBgEFBQcCARYqaHR0cDovL3d3dy5hcHBsZS5jb20vY2VydGlmaWNhdGVhdXRob3JpdHkvMDQGA1UdHwQtMCswKaAnoCWGI2h0dHA6Ly9jcmwuYXBwbGUuY29tL2FwcGxlYWljYTMuY3JsMA4GA1UdDwEB\/wQEAwIHgDAPBgkqhkiG92NkBh0EAgUAMAoGCCqGSM49BAMCA0gAMEUCIHKKnw+Soyq5mXQr1V62c0BXKpaHodYu9TWXEPUWPpbpAiEAkTecfW6+W5l0r0ADfzTCPq2YtbS39w01XIayqBNy8bEwggLuMIICdaADAgECAghJbS+\/OpjalzAKBggqhkjOPQQDAjBnMRswGQYDVQQDDBJBcHBsZSBSb290IENBIC0gRzMxJjAkBgNVBAsMHUFwcGxlIENlcnRpZmljYXRpb24gQXV0aG9yaXR5MRMwEQYDVQQKDApBcHBsZSBJbmMuMQswCQYDVQQGEwJVUzAeFw0xNDA1MDYyMzQ2MzBaFw0yOTA1MDYyMzQ2MzBaMHoxLjAsBgNVBAMMJUFwcGxlIEFwcGxpY2F0aW9uIEludGVncmF0aW9uIENBIC0gRzMxJjAkBgNVBAsMHUFwcGxlIENlcnRpZmljYXRpb24gQXV0aG9yaXR5MRMwEQYDVQQKDApBcHBsZSBJbmMuMQswCQYDVQQGEwJVUzBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABPAXEYQZ12SF1RpeJYEHduiAou\/ee65N4I38S5PhM1bVZls1riLQl3YNIk57ugj9dhfOiMt2u2ZwvsjoKYT\/VEWjgfcwgfQwRgYIKwYBBQUHAQEEOjA4MDYGCCsGAQUFBzABhipodHRwOi8vb2NzcC5hcHBsZS5jb20vb2NzcDA0LWFwcGxlcm9vdGNhZzMwHQYDVR0OBBYEFCPyScRPk+TvJ+bE9ihsP6K7\/S5LMA8GA1UdEwEB\/wQFMAMBAf8wHwYDVR0jBBgwFoAUu7DeoVgziJqkipnevr3rr9rLJKswNwYDVR0fBDAwLjAsoCqgKIYmaHR0cDovL2NybC5hcHBsZS5jb20vYXBwbGVyb290Y2FnMy5jcmwwDgYDVR0PAQH\/BAQDAgEGMBAGCiqGSIb3Y2QGAg4EAgUAMAoGCCqGSM49BAMCA2cAMGQCMDrPcoNRFpmxhvs1w1bKYr\/0F+3ZD3VNoo6+8ZyBXkK3ifiY95tZn5jVQQ2PnenC\/gIwMi3VRCGwowV3bF3zODuQZ\/0XfCwhbZZPxnJpghJvVPh6fRuZy5sJiSFhBpkPCZIdAAAxggGMMIIBiAIBATCBhjB6MS4wLAYDVQQDDCVBcHBsZSBBcHBsaWNhdGlvbiBJbnRlZ3JhdGlvbiBDQSAtIEczMSYwJAYDVQQLDB1BcHBsZSBDZXJ0aWZpY2F0aW9uIEF1dGhvcml0eTETMBEGA1UECgwKQXBwbGUgSW5jLjELMAkGA1UEBhMCVVMCCCRD8qgGnfV3MA0GCWCGSAFlAwQCAQUAoIGVMBgGCSqGSIb3DQEJAzELBgkqhkiG9w0BBwEwHAYJKoZIhvcNAQkFMQ8XDTE3MDMxNzEwMzgzOVowKgYJKoZIhvcNAQk0MR0wGzANBglghkgBZQMEAgEFAKEKBggqhkjOPQQDAjAvBgkqhkiG9w0BCQQxIgQgvL+q07\/reM0N\/5b0hwWT7TJReVTdS9QX5SPhiqeie+cwCgYIKoZIzj0EAwIERzBFAiEAttC68Xyzs6I0+tAKmg6x+0UrqmkQN\/V5c8RMMIEJHooCIHIgUHbAt2p5WrFHQKrAVL4c7nohRplZWVbVu6wbBeCgAAAAAAAA","header":{"publicKeyHash":"fpvAnSDwQFX4NX4pghdjpNwUFhoTH\/DDGhew94uJaRA=","ephemeralPublicKey":"MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAErKZUfqvhlieGAOaCKeTB\/oDEo29fS1jWSKemNDh3fIqmbfs86nL4BGtRsWRxWcMnHN6GFOQm1MEj4m7ZHxe78g==","transactionId":"38e4c267ef1de62a343d0eccada3f7e19f6b22ffc7ede899c039865432ba6aa2"},"version":"EC\_v1"} | |
| language | No | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used. |
| additionalParameters | No | Additional parameters of the order that are stored in the merchant personal area for the subsequent viewing. For each additional parameter, use the nested parameter entry that, in its turn, contains the following nested parameters:   * key - the name of a parameter; * value - the parameter value.   Below an example of a part of the request containing several nested parameters is presented.   |  | | --- | | <entry>  <key>parameter\_1</key>  <value>value\_1</value> </entry> <entry>  <key>parameter\_2</key>  <value>value\_2</value> </entry> | |
| preAuth | Yes | Parameter that defines the necessity of a pre-authorization (putting the amount on hold on the customer's account until its debiting). The following values are available:   * true - the parameter is enabled, a payment is processed with a pre-authorization (the amount on the customer's account is put on hold until the debiting); * false - the parameter is disabled (the amount is debited immediately).   If the parameter is not specified in the request, the amount is debited immediately. |
| clientId | No | Customer identifier for which a binding for recurring payments is to be created. Specify it only if the payment is dummy and is intended for subsequent recurring payments. |
| ip | No | IP-address of the payer. |

Below an example is presented of a response after a successful payment.

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:applePayResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return>  <success>true</success>  <data>  <orderId>12312312123</orderId>  </data>  <orderStatus errorCode="0"/>  </return>  </ns1:applePayResponse>  </soap:Body> </soap:Envelope> |

Below an example is presented of a response after an unsuccessful payment.

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:applePayResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return>  <success>false</success>  <error>  <code>10</code>  <description>The private key is missing</description>  </error>  <orderStatus errorCode="0"/>  </return>  </ns1:applePayResponse>  </soap:Body> </soap:Envelope> |

The description of the response parameters is presented in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Nested parameter | Mandatory parameter | Description |
| success | Not relevant | Yes | Designates a successful payment. The following values are available:   * true - the payment is processed successfully; * false - the payment failed. |
| data (this parameter is returned only if the payment is processed successfully) | orderId | Yes | Order number in the payment system unique for the merchant. |
| error  (this parameter is returned only if the payment failed) | code | Yes | Error code. |
| description | Yes | A detailed technical explanation of the error - the contents of this parameter is not to be displayed to the customer. |

Error codes that may be returned as a result of a failed payment are presented in the table below. The information on the parameters used is also presented in the [Apple documentation](https://developer.apple.com/library/content/documentation/PassKit/Reference/PaymentTokenJSON/PaymentTokenJSON.html).

|  |  |
| --- | --- |
| Error code | Message |
| 4 | Incorrect value of the [paymentToken.signature] parameter, the check failed |
| 10 | Incorrect value of the [merchant] parameter |
| Incorrect value of the [orderNumber] parameter |
| Incorrect value of the [paymentToken] parameter |
| Incorrect value of the [paymentToken.version] parameter |
| Incorrect value of the [paymentToken.header] parameter |
| Incorrect value of the [paymentToken.signature] parameter |
| Incorrect value of the [paymentToken.header.transactionId] parameter |
| Incorrect value of the [paymentToken.header.wrappedKey] parameter |
| Incorrect value of the [paymentToken.header.publicKeyHash] parameter |
| Authorization is invalid |

### Request for executing recurring payments through Apple Pay

**A request for processing a recurring payment through Apple Pay**

The recurrentPayment request is used to register the order (see [Connection URLs](#scroll-bookmark-9)).

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">  <soapenv:Header/>  <soapenv:Body>  <mer:recurrentPayment>  <arg0>  <orderNumber>123123</orderNumber>  <bindingId>binding\_id</bindingId>  <amount>123</amount>  <description>description</description>  <additionalParameters>  <entry>  <key>firstParamName</key>  <value>firstParamName</value>  </entry>  </additionalParameters>  </arg0>  </mer:recurrentPayment>  </soapenv:Body> </soapenv:Envelope> |

The description of the parameters is presented in the table below.

|  |  |  |
| --- | --- | --- |
| Parameter | Mandatory | Description |
| orderNumber | Yes | Order number. |
| bindingId | Yes | Binding identifier. |
| amount | Yes | Payment amount. |
| description | No | Order description. |
| additionalParameters | No | Additional parameters of the order that are stored in the merchant personal area for the subsequent viewing. For each additional parameter, use the nested parameter entry that, in its turn, contains the following nested parameters:   * key - the name of a parameter; * value - the parameter value.   Below an example of a part of the request containing several nested parameters is presented.   |  | | --- | | <entry>  <key>parameter\_1</key>  <value>value\_1</value> </entry> <entry>  <key>parameter\_2</key>  <value>value\_2</value> </entry> | |

Below an example of a success response to the request is presented.

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:recurrentPaymentResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return>  <success>true</success>  <data>  <orderId>134561314642</orderId>  </data>  <orderStatus errorCode="0"/>  </return>  </ns1:recurrentPaymentResponse>  </soap:Body> </soap:Envelope> |

Below an example of a failure response to the request is presented.

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:recurrentPaymentResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return>  <success>false</success>  <error>  <code>10</code>  <description>An order with this number is already registered.</description>  </error>  <orderStatus errorCode="0"/>  </return>  </ns1:recurrentPaymentResponse>  </soap:Body> </soap:Envelope> |

The description of the response parameters is presented in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Nested parameter | Mandatory parameter | Description |
| success | Not relevant | Yes | Designates a successful payment. The following values are available:   * true - the payment is processed successfully; * false - the payment failed. |
| data (this parameter is returned only if the payment is processed successfully) | orderId | Yes | Order number in the payment system unique for the merchant. |
| error  (this parameter is returned only if the payment failed) | code | Yes | Error code. |
| description | Yes | A detailed technical explanation of the error - the contents of this parameter is not to be displayed to the customer. |

### Request for a payment through Android Pay

The androidPay request is used for payments through Android Pay.

|  |
| --- |
| Use the standard requests to the payment gateway for the operations of cancellation, refund and payment completion. |

Below an example of a payment request is given.

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">  <soapenv:Header/>  <soapenv:Body>  <mer:androidPay>  <arg0>  <merchant>OurBestMerchantLogin</merchant>  <orderNumber>UAF-203974-DE</orderNumber>  <language>RU</language>  <preAuth>true</preAuth>  <paymentToken>\x7B\x0A\x20\x20\xE2\x80\x9C\x65\x6E\x63\x72\x79\x70\x74\x65\x64\x4D\x65\x73\x73\x61\x67\x65\xE2\x80\x9D\x3A\x20\xE2\x80\x9C\x5A\x57\x35\x6A\x63\x6E\x6C\x77\x64\x47\x56\x6B\x54\x57\x56\x7A\x63\x32\x46\x6E\x5A\x51\x3D\x3D\xE2\x80\x9D\x2C\x0A\x20\x20\xE2\x80\x9C\x65\x70\x68\x65\x6D\x65\x72\x61\x6C\x50\x75\x62\x6C\x69\x63\x4B\x65\x79\xE2\x80\x9D\x3A\x20\xE2\x80\x9C\x5A\x58\x42\x6F\x5A\x57\x31\x6C\x63\x6D\x46\x73\x55\x48\x56\x69\x62\x47\x6C\x6A\x53\x32\x56\x35\xE2\x80\x9D\x2C\x0A\x20\x20\x22\x74\x61\x67\x22\x3A\x20\xE2\x80\x9D\x63\x32\x6C\x6E\x62\x6D\x46\x30\x64\x58\x4A\x6C\xE2\x80\x9D\x0A\x7D</paymentToken>  <ip>127.0.0.1</ip>  <amount>230000</amount>  <currencyCode>810</currencyCode>  </arg0>  </mer:androidPay>  </soapenv:Body> </soapenv:Envelope> |

|  |  |  |
| --- | --- | --- |
| Parameter | Mandatory | Description |
| merchant | No | Merchant login in the payment gateway system. |
| orderNumber | Yes | Unique identifier of the order on the merchant side. |
| paymentToken | Yes | JSON-object that contains a token received from Android Pay. This parameter must be in the UTF-8 encoding. |
| ip | Yes | IP-address of the payer. |
| amount | Yes | Payment amount in the minimum denomination of the currency (for example, in kopeks). |
| description | No | Order description. |
| language | No | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used. |
| additionalParameters | No | Additional parameters of the order that are stored in the merchant personal area for the subsequent viewing. For each additional parameter, use the nested parameter entry that, in its turn, contains the following nested parameters:   * key - the name of a parameter; * value - the parameter value.   Below an example of a part of the request containing several nested parameters is presented.   |  | | --- | | <entry>  <key>parameter\_1</key>  <value>value\_1</value> </entry> <entry>  <key>parameter\_2</key>  <value>value\_2</value> </entry> | |
| preAuth | No | Parameter that defines the necessity of a pre-authorization (putting the amount on hold on the customer's account until its debiting). The following values are available:   * true - the parameter is enabled, a payment is processed with a pre-authorization (the amount on the customer's account is put on hold until the debiting); * false - the parameter is disabled (the amount is debited immediately).   If the parameter is not specified in the request, the amount is debited immediately. |
| clientId | No | Customer identifier for which a binding for recurring payments is to be created. Specify it only if the payment is dummy and is intended for subsequent recurring payments. |
| currencyCode | No | Numeric ISO 4217 code of the payment currency. If this parameter is not specified, it is considered to be equal to the default currency code. |

The description of the response parameters is presented in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Nested parameter | Mandatory parameter | Description |
| success | Not relevant | Yes | Designates a successful payment. The following values are available:   * **true** - the payment is processed successfully; * **false** - the payment failed. |
| data (this parameter is returned only if the payment is processed successfully) | orderId | Yes | Order number in the payment system unique for the merchant. |
| error  (this parameter is returned only if the payment failed) | code | Yes | Error code. |
| description | Yes | A detailed technical explanation of the error - the contents of this parameter is not to be displayed to the customer. |
| message | Yes | Comprehensive error description - it is intended for displaying to the user. |

Below an example is presented of a response after a successful payment.

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:androidPayResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return>  <success>true</success>  <data>  <orderId>12312312123</orderId>  </data>  <orderStatus errorCode="0"/>  </return>  </ns1:androidPayResponse>  </soap:Body> </soap:Envelope> |

An example of a response after a failed payment is given below.

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:androidPayResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return>  <success>false</success>  <error>  <code>10</code>  <description>Incorrect value of the [orderNumber] parameter</description>  </error>  <orderStatus errorCode="0"/>  </return>  </ns1:androidPayResponse>  </soap:Body> </soap:Envelope> |

The possible error codes are presented in the table below.

|  |  |
| --- | --- |
| Error code | Error message |
| 0 | The request has been processed without system errors. |
| 1 | The funds on the card are not sufficient |
| 5 | Access denied |
| The user must change the password |
| 7 | System error |
| 10 | Incorrect value of the [paymentToken] parameter |
| Incorrect value of the [orderNumber] parameter |
| Incorrect value of the [merchant] parameter |
| Incorrect value of the [ip] parameter |
| Encryption of the data passed in unsuccessful |

### Request for a payment through Samsung Pay

For a payment through Samsung Pay, the samsungPay request is used.

|  |
| --- |
| Use the standard requests to the payment gateway for the operations of cancellation, refund and payment completion. |

Below an example of a payment request is given.

|  |
| --- |
| <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:mer="http://engine.paymentgate.ru/webservices/merchant">  <soapenv:Header/>  <soapenv:Body>  <mer:samsungPay>  <arg0>  <merchant>OurBestMerchantLogin</merchant>  <orderNumber>UAF-203974-DE</orderNumber>  <description>Test description</description>  <paymentToken></paymentToken>  <language>RU</language>  <additionalParameters>  <entry>  <key>firstParamName</key>  <value>firstParamValue</value>  </entry>  </additionalParameters>  <preAuth>true</preAuth>  <ip>127.0.0.1</ip>  </arg0>  </mer:samsungPay>  </soapenv:Body> </soapenv:Envelope> |

The description of the request parameters is given in the table below.

|  |  |  |
| --- | --- | --- |
| merchant | No | Merchant login in the payment gateway system. |
| orderNumber | Yes | Unique identifier of the order on the merchant side. |
| paymentToken | Yes | The contents of the 3ds.data parameter from the response received from Samsung Pay. |
| ip | Yes | IP-address of the payer. |
| amount | Yes | Payment amount in the minimum denomination of the currency (for example, in kopeks). |
| description | No | Order description. |
| language | No | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used. |
| additionalParameters | No | Additional parameters of the order that are stored in the merchant personal area for the subsequent viewing. For each additional parameter, use the nested parameter entry that, in its turn, contains the following nested parameters:   * key - the name of a parameter; * value - the parameter value.   Below an example of a part of the request containing several nested parameters is presented.   |  | | --- | | <entry> <key>parameter\_1</key> <value>value\_1</value></entry><entry> <key>parameter\_2</key> <value>value\_2</value></entry> | |
| preAuth | No | Parameter that defines the necessity of a pre-authorization (putting the amount on hold on the customer's account until its debiting). The following values are available:   * true - the parameter is enabled, a payment is processed with a pre-authorization (the amount on the customer's account is put on hold until the debiting); * false - the parameter is disabled (the amount is debited immediately).   If the parameter is not specified in the request, the amount is debited immediately. |
| clientId | No | Customer identifier for which a binding for recurring payments is to be created. Specify it only if the payment is dummy and is intended for subsequent recurring payments. |
| currencyCode | No | Numeric ISO 4217 code of the payment currency. If this parameter is not specified, it is considered to be equal to the default currency code. |

Below an example is presented of a response after a successful payment.

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:samsungPayResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return>  <success>true</success>  <data>  <orderId>12312312123</orderId>  </data>  <orderStatus errorCode="0"/>  </return>  </ns1:samsungPayResponse>  </soap:Body> </soap:Envelope> |

Below an example is presented of a response after an unsuccessful payment.

|  |
| --- |
| <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">  <soap:Body>  <ns1:samsungPayResponse xmlns:ns1="http://engine.paymentgate.ru/webservices/merchant">  <return>  <success>false</success>  <error>  <code>1</code>  <description>The funds on the card are not sufficient</description>  </error>  <orderStatus errorCode="0"/>  </return>  </ns1:samsungPayResponse>  </soap:Body> </soap:Envelope> |

The description of the response parameters is presented in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Nested parameter | Description | Mandatory |
| success | Not relevant | Yes | Designates a successful payment. The following values are available:   * **true** - the payment is processed successfully; * **false** - the payment failed. |
| data (this parameter is returned only if the payment is processed successfully) | orderId | Yes | Order number in the payment system unique for the merchant. |
| error  (this parameter is returned only if the payment failed) | code | Yes | Error code. |
| description | Yes | A detailed technical explanation of the error - the contents of this parameter is not to be displayed to the customer. |
| message | Yes | Comprehensive error description - it is intended for displaying to the user. |

The description of the possible error codes is given in the table below.

|  |  |
| --- | --- |
| Code | Description |
| 0 | The request has been processed without system errors. |
| 1 | The funds on the card are not sufficient |
| 5 | Access denied |
| The user must change the password |
| 7 | System error |
| 10 | Incorrect value of the [paymentToken] parameter |
| Incorrect value of the [orderNumber] parameter |
| Incorrect value of the [merchant] parameter |
| Incorrect value of the [ip] parameter |
| Incorrect value of the [paymentToken.header.alg] parameter |
| Incorrect value of the [paymentToken.header.enc] parameter |
| Incorrect value of the [paymentToken.header.typ] parameter |
| Incorrect value of the [paymentToken.header.channelSecurityContext] parameter |
| Incorrect value of the [paymentToken.header.kid] parameter |
| Encryption of the data passed in unsuccessful |

REST interface

### Order registration request

The register.do request is used to register an order (see the Connection URLs section).

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection |
| password | AN..30 | yes | Store password received on the connection |
| orderNumber | AN..32 | yes | Identifier of an order in the store system, it is unique for each store within the system |
| amount | N..20 | yes | Payment amount in the minor denomination (cents or kopeks) |
| currency | N3 | no | ISO 4217 code of the payment currency. If the code is not specified, the default value is 810 (Russian roubles). |
| returnUrl | AN..512 | yes | The address to which the user is to be redirected in case of a successful payment. The address must be specified in full including the protocol used (for example, **https://test.ru** instead of **test.ru**). Otherwise, the user will be redirected to the address of the following type: **http://<payment\_gateway\_address>/<merchant\_address>**. |
| failUrl | AN..512 | no | The address to which the user is to be redirected in case of a failed payment. The address must be specified in full including the protocol used (for example, **https://test.ru** instead of **test.ru**). Otherwise, the user will be redirected to the address of the following type: **http://<payment\_gateway\_address>/<merchant\_address>**. |
| description | ANS..512 | no | Description of the order in any format |
| language | A2 | no | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used. |
| pageView | ANS..20 | no | By the value of this parameter, it is defined what pages of the payment interface are to be loaded for the customer. The possible values are:   * DESKTOP – to load pages the layout of which is designed to be displayed on displays of PCs (a search for pages with the payment\_<locale>.html and errors\_<locale>.htmlnames will be executed in the archive of payment interface pages); * MOBILE – to load pages the layout of which is designed to be displayed on displays of mobile devices (a search for pages with the mobile\_payment\_<locale>.html and mobile\_errors\_<locale>.html names will be carried out in the archive of payment interface pages); * If a store has created payment interface pages with arbitrary prefixes added to the names of page files, pas the value of the necessary prefix in the pageView parameter to load corresponding pages. For example on passing the iphone value, a search will be carried out in the archive of payment interface pages for pages with the iphone\_payment\_<locale>.html and iphone\_error\_<locale>.html names.   Where:  locale – the language of the page in ISO 639-1. For example, ru for Russian or en for English.  If a parameter is missing or does not match the format, it is considered that by default pageView=DESKTOP. |
| clientId | AN..255 | no | Identifier of the customer in the store system. This parameter is used for the binding functionality. It can be present if creating bindings is allowed for the store.   |  | | --- | | Specifying this parameter when processing payments with the use of bindings is mandatory. Otherwise, a payment will be unsuccessful. | |
| merchantLogin | AN..255 | no | To register an order on behalf of a child merchant, specify the merchant login in this parameter. |
| jsonParams | AN..1024 | no | Block for passing merchant parameters. The additional information fields for the subsequent storage are passed in the following way:  {"<name1>":"<value1>",...,"<nameN>":"<valueN>"},  These fields can be passed to the Bank processing system to be subsequently displayed in the registers.\*  Enabling the functionality is possible upon agreement with the Bank during the integration period.  If sending notifications to a customer is configured for a merchant, the customer's email address must be passed in this block in the email parameter.   |  | | --- | | On payments for housing and utilities services it is necessary to pass the following additional parameters:   * payment\_doc\_id - the identifier of the payment document; * order\_num - the order number. | |
| sessionTimeoutSecs | N...9 | no | Order lifetime in seconds.  If the parameter is not specified, the value specified in the merchant settings or the default value (1200 seconds = 20 minutes) will be used.  If the expirationDate parameter is present in the request, the value of the sessionTimeoutSecs parameter is ignored. |
| expirationDate | ANS | no | The date and time of the order lifetime expiration. The format used is: yyyy-MM-ddTHH:mm:ss.  If this parameter is not passed in the request, sessionTimeoutSecs is used to define the time of the order lifetime expiration. |
| bindingId | AN..255 | no | Identifier of the binding created earlier. It can be used only if the store has the permission to work with bindings. If this parameter is passed in the given request, it means that: 1. The given order can be paid only using a binding; 2. The payer will be redirected to the payment page on which only entering CVC is required. |
| features | ANS..255 | no | It is possible to use the following values.  AUTO\_PAYMENT - if the order registration request initiates executing auto-payments.  VERIFY - If this parameter is specified after the order registration request, the cardholder is to be verified without debiting funds from the cardholder account. Thus it is possible to pass a zero amount in the request. This verification allow the merchant to ensure that a card is used by the cardholder and to debit this card in the future without verifying authentication data (CVC, 3D-Secure) on processing subsequent payments.   |  | | --- | | **The details of passing the VERIFY value**   * Even if the payment amount is to be passed in the request, it will not be debited from the account. * After the order has been successfully registered, it is passed to the REVERSED (cancelled) status. | |

**\*** By default, the following fields are passed to the Bank processing system:

* + orderNumber – the order number in the store system;
  + description – the order description (no more than 99 symbols, it is forbidden to use %, +, an end of line \r and line break \n).

If the additional parameter merchantOrderId is passed in an order, its value is to be passed to the processing system of the Bank as an order number (instead of the value in the orderNumber field).

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | No | Identifier of the order in the payment system. It is unique within the system. The parameter is missing if the order registration failed due to an error detailed in errorCode. |
| formUrl | AN..512 | No | URL of the payment form to which the customer's browser is to be redirected. The parameter is not returned if the order registration failed due to an error detailed in errorCode. |
| errorCode | N3 | No | Error code. |
| errorMessage | AN..512 | no | Error description in the language passed in the language parameter in the request. |

**Error codes (the** errorCode **field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | An order with this identifier has already been processed |
| 1 | An order with this number has been registered, but has not been paid |
| 1 | Unknown order ID |
| 3 | Unknown currency |
| 4 | Order number cannot be empty |
| 4 | Merchant name cannot be empty |
| 4 | Amount is missing |
| 4 | Return URL cannot empty |
| 4 | Password cannot be empty |
| 5 | Merchant login is incorrect |
| 5 | Incorrect amount |
| 5 | Language parameter is incorrect |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | Access denied |
| 5 | [jsonParams] is incorrect |
| 7 | System error |
| 13 | Using both values, Features FORCE\_TDS/FORCE\_SSL and AUTO\_PAYMENT, is not allowed |
| 13 | The merchant does not have the permission to process auto-payments |
| 13 | The merchant does not have the permission to process verification payments |
| 14 | Features are specified incorrectly |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab//rest/register.do?amount=100&currency=810&language=ru&orderNumber=87654321&password=password&returnUrl=https://web.rbsuat.com/ab/finish.html&userName=userName&{jsonParams="param1":"value1","param2":"value2"}&pageView=DESKTOP&expirationDate=2014-09-08T14:14:14&merchantLogin=merch\_child&features=AUTO\_PAYMENT |

**An example of the POST request:**

|  |
| --- |
| amount=100&currency=810&language=ru&orderNumber=87654321&returnUrl=https://web.rbsuat.com/ab/finish.html&pageView=DESKTOP&{jsonParams="param1":"value1","param2":"value2"}&expirationDate=2014-09-08T14:14:14&merchantLogin=merch\_child&features=AUTO\_PAYMENT |

**Response example:**

|  |
| --- |
| {"orderId":"70906e55-7114-41d6-8332-4609dc6590f4","formUrl":"https://web.rbsuat.com/ab/merchants/test/payment\_ru.html?mdOrder=70906e55-7114-41d6-8332-4609dc6590f4"} |

### Registration request for orders with pre-authorization

The registerPreAuth.do request is used to register an order with pre-authorization (see the Connection URLs section).

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection |
| password | AN..30 | yes | Store password received on the connection |
| orderNumber | AN..32 | yes | Identifier of an order in the store system, it is unique for each store within the system |
| amount | N..20 | yes | Payment amount in the minor denomination (cents or kopeks) |
| currency | N3 | no | ISO 4217 code of the payment currency. If this parameter is not specified, it is considered to be equal to the default currency code. |
| returnUrl | AN..512 | yes | The address to which the user is to be redirected in case of a successful payment. The address must be specified in full including the protocol used (for example, **https://test.ru** instead of **test.ru**). Otherwise, the user will be redirected to the address of the following type: **http://<payment\_gateway\_address>/<merchant\_address>**. |
| failUrl | AN..512 | no | The address to which the user is to be redirected in case of a failed payment. The address must be specified in full including the protocol used (for example, **https://test.ru** instead of **test.ru**). Otherwise, the user will be redirected to the address of the following type: **http://<payment\_gateway\_address>/<merchant\_address>**. |
| description | ANS..512 | no | Description of the order in any format |
| language | A2 | no | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used |
| pageView | ANS..20 | no | By the value of this parameter, it is defined what pages of the payment interface are to be loaded for the customer. The possible values are:   * DESKTOP – to load pages the layout of which is designed to be displayed on displays of PCs (a search for pages with the payment\_<locale>.html and errors\_<locale>.htmlnames will be executed in the archive of payment interface pages); * MOBILE – to load pages the layout of which is designed to be displayed on displays of mobile devices (a search for pages with the mobile\_payment\_<locale>.html and mobile\_errors\_<locale>.html names will be carried out in the archive of payment interface pages); * If a store has created payment interface pages with arbitrary prefixes added to the names of page files, pas the value of the necessary prefix in the pageView parameter to load corresponding pages. For example on passing the iphone value, a search will be carried out in the archive of payment interface pages for pages with the iphone\_payment\_<locale>.html and iphone\_error\_<locale>.html names.   Where:  locale – the language of the page in ISO 639-1. For example, ru for Russian or en for English.  If a parameter is missing or does not match the format, it is considered that by default pageView=DESKTOP. |
| clientId | AN..255 | no | Identifier of the customer in the store system. This parameter is used for the binding functionality. It can be present if creating bindings is allowed for the store.   |  | | --- | | Specifying this parameter when processing payments with the use of bindings is mandatory. Otherwise, a payment will be unsuccessful. | |
| merchantLogin | AN..255 | no | To register an order on behalf of a child merchant, specify the merchant login in this parameter. |
| jsonParams | AN..1024 | no | Block for passing merchant parameters. The additional information fields for the subsequent storage are passed in the following way:  {"<name1>":"<value1>",...,"<nameN>":"<valueN>"},  These fields can be passed to the Bank processing system to be subsequently displayed in the registers.\*  Enabling the functionality is possible upon agreement with the Bank during the integration period.  If sending notifications to a customer is configured for a merchant, the customer's email address must be passed in this block in the email parameter. |
| sessionTimeoutSecs | N...9 | no | Order lifetime in seconds.  If the parameter is not specified, the value specified in the merchant settings or the default value (1200 seconds = 20 minutes) will be used.  If the expirationDateparameter is present in the request, the value of the sessionTimeoutSecs parameter is ignored. |
| expirationDate | ANS | no | The date and time of the order lifetime expiration. The format used is: yyyy-MM-ddTHH:mm:ss.  If this parameter is not passed in the request, sessionTimeoutSecs is used to define the time of the order lifetime expiration. |
| bindingId | AN..255 | no | Identifier of the binding created earlier. It can be used only if the store has the permission to work with bindings. If this parameter is passed in the given request, it means that: 1. The given order can be paid only using a binding; 2. The payer will be redirected to the payment page on which only entering CVC is required. |
| features | ANS..255 | no | It is possible to use the following values.  AUTO\_PAYMENT - if the order registration request initiates executing auto-payments.  VERIFY - If this parameter is specified after the order registration request, the cardholder is to be verified without debiting funds from the cardholder account. Thus it is possible to pass a zero amount in the request. This verification allow the merchant to ensure that a card is used by the cardholder and to debit this card in the future without verifying authentication data (CVC, 3D-Secure) on processing subsequent payments.   |  | | --- | | **The details of passing the VERIFY value**   * Even if the payment amount is to be passed in the request, it will not be debited from the account. * After the order has been successfully registered, it is passed to the REVERSED (cancelled) status. | |

**\*** By default, the following fields are passed to the Bank processing system:

* + - orderNumber – the order number in the store system;
    - description – the order description (no more than 99 symbols, it is forbidden to use %, +, an end of line \r and line break \n).

If the additional parameter merchantOrderId is passed in an order, its value is to be passed to the processing system of the Bank as an order number (instead of the value in the orderNumber field).

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderId | ANS36 | No | Identifier of the order in the payment system. It is unique within the system. The parameter is missing if the order registration failed due to an error detailed in errorCode. |
| formUrl | AN..512 | No | URL of the payment form to which the customer's browser is to be redirected. The parameter is not returned if the order registration failed due to an error detailed in errorCode. |
| errorCode | N3 | No | Error code. |
| errorMessage | AN..512 | no | Error description in the language passed in the language parameter in the request. |

**Error codes (the** errorCode **field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | An order with this identifier has already been processed |
| 1 | An order with this number has been registered, but has not been paid |
| 1 | Unknown order ID |
| 3 | Unknown currency |
| 4 | Order number cannot be empty |
| 4 | Merchant name cannot be empty |
| 4 | Amount is missing |
| 4 | Return URL cannot empty |
| 4 | Password cannot be empty |
| 5 | Incorrect amount |
| 5 | Language parameter is incorrect |
| 5 | Merchant login is incorrect |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | Access denied |
| 5 | [jsonParams] is incorrect |
| 7 | System error |
| 13 | Using both values, Features FORCE\_TDS/FORCE\_SSL and AUTO\_PAYMENT, is not allowed |
| 13 | The merchant does not have the permission to process auto-payments |
| 13 | The merchant does not have the permission to process verification payments |
| 14 | Features are specified incorrectly |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/registerPreAuth.do?amount=100¤cy=810&language=ru&orderNumber=87654321&password=password&returnUrl=https://web.rbsuat.com/ab/finish.html&userName=userName&{jsonParams="param1":"value1","param2":"value2"}&pageView=MOBILE&merchantLogin=merch\_child&features=AUTO\_PAYMENT |

**An example of the POST request:**

|  |
| --- |
| amount=100&currency=810&language=ru&orderNumber=87654321&returnUrl=https://web.rbsuat.com/ab/finish.html&pageView=MOBILE&{jsonParams="param1":"value1","param2":"value2"}&merchantLogin=merch\_child&features=AUTO\_PAYMENT |

**Response example:**

|  |
| --- |
| {"orderId":"61351fbd-ac25-484f-b930-4d0ce4101ab7","formUrl":"https://web.rbsuat.com/ab/merchants/test/mobile\_payment\_ru.html?mdOrder=61351fbd-ac25-484f-b930-4d0ce4101ab7"} |

### Order payment completion request

The deposit.do request is used to complete an earlier pre-authorized order.

This operation can be executed provided that you have the corresponding permissions in the system.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection |
| password | AN..30 | yes | Store password received on the connection |
| orderId | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| amount | N..20 | yes | Payment amount in the minor denomination (cents or kopeks) |

|  |
| --- |
| If the amount parameter is set to zero, the completion is done for the whole pre-authorized amount. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | No | Error code. |
| errorMessage | AN..512 | No | Description of the error in the language specified in the language parameter. |

**Error codes (the** errorCode **field):**  
Classification of error codes:

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | Incorrect value of a request parameter |
| 6 | Unregistered OrderId |
| 7 | System error |

Meaning of the error codes:

|  |  |
| --- | --- |
| Value | Description |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | Incorrect amount |
| 5 | The deposit amount must be equal to zero or not less than one rouble |
| 6 | Unknown order ID |
| 7 | Payment must be in the correct status |
| 7 | System error |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/deposit.do?amount=100&currency=810&language=ru&orderId=e5b59d3d-746b-4828-9da4-06f126e01b68&password=password&userName=userName |

**An example of the POST request:**

|  |
| --- |
| amount=100&currency=810&language=ru&orderId=e5b59d3d-746b-4828-9da4-06f126e01b68 |

**Response example:**

|  |
| --- |
| {"errorCode":0} |

### Order status request

The getOrderStatus.do is used to get the current order status.

The order status is defined by the value of the OrderStatus parameter.  
The authCode field is obsolete.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection |
| password | AN..30 | yes | Store password received on the connection |
| orderId | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, it is considered that the language is Russian. An error message will be returned in this language. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| OrderStatus | N2 | No | The status of the order in the payment system is defined by the value of this parameter. The list of available values is presented in the table below. It is missing if the order has not been found. |
| ErrorCode | N3 | No | Error code. |
| ErrorMessage | AN..512 | No | Error description in the language passed in the Language parameter of the request. |
| OrderNumber | AN..32 | Yes | Order identifier in the store system |
| Pan | N..19 | no | Masked number of the card that has been used for the payment. This parameter is to be specified only after the order has been paid. |
| expiration | N6 | no | Card expiration . This parameter is to be specified only after the order has been paid. |
| cardholderName | A..64 | no | Name of the cardholder. This parameter is to be specified only after the order has been paid. |
| Amount | N..20 | yes | Payment amount in the minor denomination (cents or kopeks) |
| currency | N3 | no | ISO 4217 code of the payment currency. If the code is not specified, the default value is 810 (Russian roubles). |
| approvalCode | AN6 | no | IPS authorization code. This field has a fixed length (six symbols) and can contain digits and Latin letters. |
| authCode | N3 | no | This field is obsolete. Its value is always equal to"2" regardless the order status and the authorization code of the processing system. |
| Ip | AN..20 | no | IP address of the user who has paid the order |
| *BindingInfo element:* |  |  |  |
| clientId | AN..255 | no | Identifier of the customer in the store system passed on the order registration. This parameter is present only if a store has the permission to create bindings. |
| bindingId | AN..255 | no | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. |

**The** OrderStatus  **field can have the following values:**

|  |  |
| --- | --- |
| Status number | Description |
| 0 | The order has been registered but not paid |
| 1 | The pre-authorized amount has been put on hold (for two-phase payments) |
| 2 | Full authorization of the order amount has been performed |
| 3 | Authorization is cancelled |
| 4 | A refund operation has been processed for the transaction |
| 5 | Authorization through ACS of the issuing bank has been initiated |
| 6 | Authorization is declined |

**Error codes (the** ErrorCode **field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 2 | Order is declined because of an error in the payment details |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | [orderId] is not specified |
| 6 | Unknown order ID |
| 7 | System error |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/getOrderStatus.do?orderId=b8d70aa7-bfb3-4f94-b7bb-aec7273e1fce&language=ru&userName=userName&password=password |

**An example of the POST request:**

|  |
| --- |
| orderId=b8d70aa7-bfb3-4f94-b7bb-aec7273e1fce&language=ru |

**Response example:**

|  |
| --- |
| {"expiration":"201512","cardholderName":"tr tr","depositAmount":789789,"currency":"810","approvalCode":"123456","authCode":2,"clientId":"666","bindingId":"07a90a5d-cc60-4d1b-a9e6-ffd15974a74f","ErrorCode":"0","ErrorMessage":"Успешно","OrderStatus":2,"OrderNumber":"23asdafaf","Pan":"411111\*\*1111","Amount":789789} |

### Extended order status request

The getOrderStatusExtended.do request is used to request the status of a registered order.

**Request parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection |
| password | AN..30 | yes | Store password received on the connection |
| orderId | ANS36 | yes\* | Identifier of the order in the payment system. It is unique within the system. |
| orderNumber | AN..32 | yes\* | Order identifier in the store system. |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, it is considered that the language is Russian. An error message will be returned in this language. |

\* Either orderId or orderNumber must be present in the request. If both parameters are present in the request, orderId is prioritized.

Several sets of response parameters exist. It depends of the version of getOrderStatusExtended specified in the merchant settings, what sets of parameters are to be returned.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Mandatory | Description | Version of getOrderStatusExtended |
| orderNumber | AN..32 | yes | Order identifier in the store system. | All versions. |
| orderStatus | N2 | no | The order status in the payment system is defined by the value of this parameter. The available values are: given in the list below. It is missing if the order has not been found.   * **0** - the order is registered, but not paid; * **1** - the pre-authorized amount is put on hold (for two-phase payments); * **2** - a full authorization of the order amount has been performed; * **3** - the authorization was cancelled; * **4** - a refund operation has been processed for the transaction; * **5** - an authorization through ACS of the issuing bank has been initiated; * **6** - the authorization was declined | All versions. |
| actionCode | N3 | yes | Response code. | All versions. |
| actionCodeDescription | AN..512 | yes | Explanation of the response code in the language passed in the Language parameter of the request. | All versions. |
| errorCode | N3 | no | Error code. The following values are available.   * **0** - the request has been processed without system errors.; * **1** - [orderid] or [ordernumber] is expected; * **5** - access denied; * **5** - the user must change the password; * **6** - order is not found; * **7** - a system error. | All versions. |
| errorMessage | AN..512 | no | Error description in the language passed in the Language parameter of the request. | All versions. |
| amount | N..20 | yes | Payment amount in the minor denomination (cents or kopeks) | All versions. |
| currency | N3 | no | ISO 4217 code of the payment currency. If the code is not specified, the default value is 810 (Russian roubles). | All versions. |
| date | ANS | yes | Order registration date. | All versions. |
| orderDescription | AN..512 | no | Order description passed on its registration | All versions. |
| ip | AN..20 | yes | IP-address of the buyer. | All versions. |
| *The* merchantOrderParams *element is present in a response if an order contains additional parameters of the merchant. Each additional parameter of an order is present in a separate* merchantOrderParams *element.* | | | | |
| name | AN..20 | no | Name of the additional parameter | All versions. |
| value | AN..1024 | no | Value of the additional parameter | All versions. |
| *The* cardAuthInfo *element contains a structure that consists of a list of the* secureAuthInfo *element and the following parameters:* | | | | |
| maskedPan | N..19 | no | Masked number of the card that has been used for the payment. This parameter is to be specified only after the order has been paid. | All versions. |
| expiration | N6 | no | Card expiration . This parameter is to be specified only after the order has been paid. | All versions. |
| cardholderName | A..64 | no | Name of the cardholder. This parameter is to be specified only after the order has been paid. | All versions. |
| approvalCode | AN6 | no | Payment authorization code. The field has a fixed length (six symbols), it can contain digits and Latin letters. The parameter is specified only after the order has been paid. | All versions. |
| chargeback | A..5 | no | The parameter defines whether the funds have been forcibly returned to the buyer by the bank. The available values are:   * **true** (the funds have been refunded); * **false** (the funds have not been refunded). | 06 and higher. |
| paymentSystem | N..10 | yes | The payment system name. The available values are:   * **VISA**; * **MASTERCARD**; * **AMEX**; * **JCB**; * **CUP**; * **MIR**. | 08 and higher. |
| product | AN..255 | yes | Additional details on corporate cards. These details are filled in by the technical support service through the administrative console. If such details are missing, an empty value is returned. | 08 and higher. |
| paymentWay | AS..14 | yes | The method of order completion (a payment with entering card data, a payment using a binding, etc.). | 09 and higher. |
| *The* secureAuthInfo *element (the element consists of the* eci *and* threeDSInfo *elements that are the lists of the* cavv *and* xid *parameters):* | | | | |
| eci | N..4 | no | Electronic Commerce Indicator. The indicator is specified only after an order has been paid and in case the corresponding permission is present. | All versions. |
| cavv | ANS..200 | no | The value for the cardholder and card authentication check. The indicator is specified only after an order has been paid and in case the corresponding permission is present. | All versions. |
| xid | ANS..80 | no | Electronic Commerce Indicator of the transaction. The indicator is specified only after an order has been paid and in case the corresponding permission is present. | All versions. |
| *The* bindingInfo *element consists of these parameters:* | | | | |
| clientId | AN..255 | no | Identifier of the customer in the store system passed on the order registration. This parameter is present only if a store has the permission to create bindings. | All versions. |
| bindingId | AN..255 | no | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. | All versions. |
| authDateTime | ANS | no | The authorization date and time. | 02 and higher. |
| authRefNum | AN..24 | no | Reference number. | 02 and higher. |
| terminalId | AN..10 | no | Terminal ID. | 02 and higher. |
| *The* paymentAmountInfo *element consists of these parameters:* | | | | |
| approvedAmount | N..20 | no | The amount put on hold on the card (is used only for two-phase payments). | 03 and higher. |
| depositedAmount | N..20 | no | The amount confirmed for debiting from the card. | 03 and higher. |
| refundedAmount | N..20 | no | The refund amount. | 03 and higher. |
| paymentState | A..10 | no | The order status. | 03 and higher. |
| feeAmount | N..20 | no | The fee amount. | 11 and higher. |
| *The* bankInfo *element consists of these parameters:* | | | | |
| bankName | AN..200 | no | Name of the issuing bank. | 03 and higher. |
| bankCountryCode | AN..4 | no | Country codes of the issuing bank. | 03 and higher. |
| bankCountryName | AN..160 | no | Name of the country of the issuing bank passed in the language parameter of the request or in the language of the user who has called the method if the language has not been specified in the request. | 03 and higher. |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/getOrderStatusExtended.do?userName=userName&password=password&orderId=b9054496-c65a-4975-9418-1051d101f1b9&language=ru&merchantOrderNumber=0784sse49d0s134567890 |

**An example of the POST request:**

|  |
| --- |
| orderId=b9054496-c65a-4975-9418-1051d101f1b9&language=ru&merchantOrderNumber=0784sse49d0s134567890 |

**Response example:**

|  |
| --- |
| {"errorCode":"0","errorMessage":"Success","orderNumber":"0784sse49d0s134567890","orderStatus":6,"actionCode":-2007,"actionCodeDescription":"Время сессии истекло","amount":33000,"currency":"810","date":1383819429914,"orderDescription":" ","merchantOrderParams":[{"name":"email","value":"yap"}],"attributes":[{"name":"mdOrder","value":"b9054496-c65a-4975-9418-1051d101f1b9"}],"cardAuthInfo":{"expiration":"201912","cardholderName":"Ivan","secureAuthInfo":{"eci":6,"threeDSInfo":{"xid":"MDAwMDAwMDEzODM4MTk0MzAzMjM="}},"pan":"411111\*\*1111"},"terminalId":"333333"} |

### Order payment cancellation request

The reverse.do request is used to cancel a payment for an order. The cancellation function is available within a limited time period after a payment has been processed. The exact duration should be found out from the bank.

The cancellation operation can be executed only once. If it ends with an error, a repetitive cancellation operation will not be processed.

This functionality is available for a merchant upon agreement with the Bank. To execute a cancellation operation, a user needs to have the corresponding permissions.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection |
| password | AN..30 | yes | Store password received on the connection |
| orderId | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| language | A2 | no | Language in the ISO 639-1 encoding. The error description is returned in this language. If the parameter is missing, the default language specified in the merchant settings is used. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | No | Error code. |
| errorMessage | AN..512 | No | Description of the error in the language specified in the language parameter. |

**Error codes (the ErrorCode field):**  
  
Classification of error codes:

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | Incorrect value of a request parameter |
| 6 | Unregistered OrderId |
| 7 | System error |

Meaning of the error codes:

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | [orderId] is not specified |
| 6 | Unknown order ID |
| 7 | Operation is impossible for the current payment status |
| 7 | Reversal is impossible because of incorrect internal parameter values. Check the amount put on hold and the deposit amount |
| 7 | System error |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/reverse.do?language=ru&orderId=9231a838-ac68-4a3e-bddb-d9781433d852&password=password&userName=userName |

**An example of the POST request:**

language=ru&orderId=9231a838-ac68-4a3e-bddb-d9781433d852

**Response example:**

|  |
| --- |
| {"errorCode":"0","errorMessage":"Success"} |

### Order payment refund request

The refund.do request is used to return funds.

Upon this request, the funds for the specified order are to be returned to the payer. The request will end with an error if the funds have not been debited for this order. The system permits returning funds more than once, but for a total amount not exceeding the initial debit amount.

|  |
| --- |
| When processing a refund for a payment for housing and utilities services, only a full refund is available. |

To process a refund operation, it is necessary to have the corresponding permissions in the system.

**Request parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection |
| password | AN..30 | yes | Store password received on the connection |
| orderId | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| amount | N..20 | yes | Payment amount in the minor denomination (cents or kopeks) |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | No | Error code. |
| errorMessage | AN..512 | No | Description of the error in the language specified in the language parameter. |

**Error codes (the** errorCode **field):**

Classification of error codes:

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | Incorrect value of a request parameter |
| 6 | Unregistered OrderId |
| 7 | System error |

Meaning of the error codes:

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | [orderId] is not specified |
| 6 | Unknown order ID |
| 7 | Payment must be in the correct status |
| 7 | Incorrect deposit amount (less than one rouble) |
| 7 | System error |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/refund.do?amount=500&currency=810&language=ru&orderId=5e97e3fd-1d20-4b4b-a542-f5995f5e8208&password=password&userName=userName |

**An example of the POST request:**

|  |
| --- |
| amount=500&currency=810&language=ru&orderId=5e97e3fd-1d20-4b4b-a542-f5995f5e8208 |

**Response example:**

|  |
| --- |
| {"errorCode":0} |

### Request for checking a card for enrolment to 3D-Secure

The verifyEnrollment.do request is used to check a card for enrolment to 3D-Secure.

**Request parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | User login (API) |
| password | AN..30 | yes | User password (API) |
| pan | N12...19 | yes | Card number |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Error description. |
| enrolled | A1 | no | The flag signifying enrolment of the card to 3D-Secure. The available values are: Y, N, U. |
| emitterName | AN..160 | no | Name of the issuing bank. |
| emitterCountryCode | AN..4 | no | Country codes of the issuing bank. |

**Error codes (the errorCode fields):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | The card number is not specified |
| 1 | The card number must be a number containing 13 to 19 digits |
| 5 | The user must change the password. |
| 5 | Access denied |
| 6 | Information is not found for the card number specified. |
| 7 | System error. |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/verifyEnrollment.do?userName=userName&password=password&pan=4111111111111111 |

**An example of the POST request:**

|  |
| --- |
| pan=4111111111111111 |

**Response example:**

|  |
| --- |
| {"errorCode":"0","errorMessage":"Success","emitterName":"TEST CARD","emitterCountryCode":"RU","enrolled":"Y"} |

### Request for adding additional parameters to an order

The addParams.do method is used to add to an order additional parameters.

If an additional parameter already exists in the order, the value most recently passed is saved to the order when adding a parameter with the same name.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection |
| password | AN..30 | yes | Store password received on the connection |
| orderId | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| language | A2 | no | Language in the ISO 639-1 encoding. The error description is returned in this language. If the parameter is missing, the default language specified in the merchant settings is used. |
| params | AN..1024 | yes | Fields to pass additional parameters of the following type: {"param":"value","param2":"value2"}. |

**Response parameters**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | yes | Error code. |
| errorMessage | AN..512 | no | Error description. It is missing when a request is processed unsuccessfully. |

**Error codes (the ErrorCode field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | Access denied |
| 5 | The user must change the password |
| 6 | orderId is not specified |
| 6 | Unknown order ID |
| 7 | System error |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/addParams.do?language=ru&orderId=769b8dad-2318-4c01-bfc4-94532522fa68&password=password&userName=userName&params={"addParams1":"value1","addParams2":"value2"} |

**An example of the POST request:**

|  |
| --- |
| language=ru&orderId=769b8dad-2318-4c01-bfc4-94532522fa68&params={"addParams1":"value1","addParams2":"value2"} |

**Response example:**

|  |
| --- |
| {"errorCode":0} |

### Request for payments statistics for a period

The getLastOrdersForMerchants.do method allows you to get the payment statistics for a specified period.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection. |
| password | AN..30 | yes | Store password received on the connection. |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, it is considered that the language is Russian. An error message will be returned in this language. |
| page | N | no | When processing the request, a list is formed that is broken down into pages (with the number of records on each page equal to size). A page with the number that was specified in the page parameter is returned in the response. The pages numbering starts from 0. If the parameter is not specified, a page with the number 0 is returned. |
| size | N..3 | yes | Number of elements on a page (the maximum value = 200). |
| c | ANS | yes | The date and time when the period of collecting orders starts, in the format YYYYMMDDHHmmss. |
| по | ANS | yes | The date and time when the period of collecting orders ends, in the format YYYYMMDDHHmmss. |
| transactionStates | A..9 | yes | In this block, it is necessary to list the required order statuses. Only orders in one of the specified statuses are included into the report. If there are several values, they are separated by commas. The available values are: CREATED, APPROVED, DEPOSITED, DECLINED, REVERSED, REFUNDED. |
| merchants | ANS | yes | List of logins of the merchant whose transactions are included into the report. If there are several values, they are separated by commas.  Leave this field empty to get the list of reports on all the available merchants (child merchants and merchants specified in the settings of the user). |
| searchByCreatedDate | boolean | no | The possible values are:   * true – a search for orders that have the creation date that falls into the specified period. * false – a search for orders that have the payment date that falls into the specified period (thus, orders in the CREATED or DECLINED status cannot be present in the report).   The default value is false . |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N..2 | yes | Error code. The description of the available codes is presented below in the Error codes (the errorCode field) table |
| errorMessage | AN..512 | no | Error description. It is present only an error is present (errorCode is not equal to 0). |
| orderStatuses |  |  | Block that contains the information on the orders included into the report. See the orderStatuses block parameters table below. |
| totalCount | N | yes | Total number of elements in the report (on all pages). |
| page | N | yes | Number of the current page (it is equal to the page number passed in the request). |
| pageSize | N..3 | yes | Maximum number of records on a page (it is equal to the page number passed in the request). |

Parameters of the orderStatuses block:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| orderNumber | AN..32 | yes | Order identifier in the store system. |
| orderStatus | N..2 | yes | Status of the order in the payment system. The available values are given in the orderStatus field: table. |
| actionCode | N..3 | yes | Response code. |
| actionCodeDescription | AN..512 | yes | Meaning of the response code. |
| amount | N..20 | yes | Payment amount in the minor denomination of the currency. |
| currency | N3 | yes | ISO 4217 code of the payment currency. If it is not specified, it is considered to be equal to the default currency value. |
| date | ANS | yes | Order registration date. |
| orderDescription | AN..512 | no | Order description passed on its registration |
| ip | AN..20 | no | IP-address of the buyer. It is specified only after a payment. |
| errorCode | N..2 | yes | Error code. |
| merchantOrderParams |  | no | Tag containing attributes that contain additional merchant parameters. See the merchantOrderParams block parameters table below. |
| attributes |  | yes | Attributes of the order in the payment system (order number). See the attributes block parameters table below. |
| cardAuthInfo |  | no | A tag containing the payment attributes. See the cardAuthInfo block parameters table below. |
| bindingInfo |  | no | Tag containing the information on the binding with which the payment is performed. See the bindingInfo block parameters table below. |
| authDateTime | ANS | no | Authorization date and time |
| terminalId | AN..10 | no | Terminal ID |
| authRefNum | AN..24 | no | Reference number |
| paymentAmountInfo |  | no | Tag containing the information on the confirmation amount, debit amount, and refund amount. See the paymentAmountInfo block parameters table below. |
| bankInfo |  | no | Tag containing the information on the issuing bank. See the bankInfo block parameters table below. |

merchantOrderParams block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| name | AN..20 | yes | Name of the additional merchant parameter |
| value | AN..1024 | yes | Value of the additional merchant parameter |

attributes block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| name | A7 | yes | Attribute name is "mdOrder". |
| value | ANS36 | yes | Attribute value is the order number in the payment system (is unique within the system). |

cardAuthInfo block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| pan | N..19 | no | Masked number of the card that has been used for the payment. |
| expiration | N6 | no | Card expiration . |
| cardholderName | A..64 | no | Name of the cardholder. |
| approvalCode | AN6 | no | Payment authorization code. The field has a fixed length (six symbols), it can contain digits and Latin letters. |

bindingInfo block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| clientId | AN..255 | no | Identifier of the customer in the store system. |
| bindingId | AN..255 | no | Identifier of the binding used for the payment. |

paymentAmountInfo block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| paymentState | N..9 | no | Payment status |
| approvedAmount | N..20 | no | Amount confirmed to be debited. |
| depositedAmount | N..20 | no | Amount confirmed debited from the card. |
| refundedAmount | N..20 | no | The refund amount. |

bankInfo block parameters:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| bankName | AN..200 | no | Name of the issuing bank. |
| bankCountryCode | AN..4 | no | Code of the issuing bank country |
| bankCountryName | AN..160 | no | Name of the country of the issuing bank passed in the language parameter of the request or in the language of the user who has called the method if the language has not been specified in the request. |

The orderStatus field can have the following values:

|  |  |
| --- | --- |
| Value | Description |
| 0 | The order has been registered but not paid |
| 1 | The pre-authorized amount has been put on hold (for two-phase payments) |
| 2 | Full authorization of the order amount has been performed |
| 3 | Authorization is cancelled |
| 4 | A refund operation has been processed for the transaction |
| 5 | Authorization through ACS of the issuing bank has been initiated |
| 6 | Authorization is declined |

Error codes (the errorCode field):

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 5 | One of the mandatory fields is not filled in |
| 5 | Incorrect format of the transactionStates parameter |
| 5 | Access denied |
| 7 | System error |
| 10 | Value of the size parameter exceeds the maximum allowed value |
| 10 | Insufficient permissions to view transactions for the specified merchant |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/getLastOrdersForMerchants.do?userName=userName&password=password&language=ru&page=0&size=100&from=20141009160000&to=20141111000000&transactionStates=DEPOSITED,REVERSED&merchants=SevenEightNine&searchByCreatedDate=false |

**An example of the POST request:**

|  |
| --- |
| language=ru&page=0&size=100&from=20141009160000&to=20141111000000&transactionStates=DEPOSITED,REVERSED&merchants=SevenEightNine&searchByCreatedDate=false |

**Response example:**

|  |
| --- |
| {"errorCode":0,    "orderStatuses":[   {"errorCode":"0","orderNumber":"58drs0Pes459Hdsddd0567a0","orderStatus":2,"actionCode":0,"actionCodeDescription":"The request has been successfully processed","amount":250000,"currency":"810","date":1414485649233,"orderDescription":"Opisanie","ip":"212.5.125.194","merchantOrderParams":[{"name":"registr1","value":"registr1"}],"attributes":[{"name":"mdOrder","value":"f1a3365b-542c-4c8d-b34c-e9a7ee8dbc9c"}],"cardAuthInfo":{"expiration":"201512","cardholderName":"Ivan","approvalCode":"123456","pan":"411111\*\*1111"},"bindingInfo":{"clientId":"666","bindingId":"1eabfb8e-b90e-4dc8-bef6-14bd392b1cec"},"authDateTime":1414485661207,"terminalId":"111113","authRefNum":"111111111111","paymentAmountInfo":{"paymentState":"DEPOSITED","approvedAmount":250000,"depositedAmount":250000,"refundedAmount":0},"bankInfo":{"bankName":"TEST CARD","bankCountryCode":"RU","bankCountryName":"Russia"}},    {"errorCode":"0","orderNumber":"57drs0Pes459Hdsddd0567a0","orderStatus":2,"actionCode":0,"actionCodeDescription":"Запрос успешно обработан","amount":250000,"currency":"810","date":1414485277286,"orderDescription":"Opisanie","ip":"212.5.125.194","merchantOrderParams":[{"name":"registr1","value":"registr1"}],"attributes":[{"name":"mdOrder","value":"09489184-bc5e-44a7-b6c4-3ca1feb8ef69"}],"cardAuthInfo":{"expiration":"201512","cardholderName":"Ivan","approvalCode":"123456","pan":"411111\*\*1111"},"bindingInfo":{"clientId":"666","bindingId":"1eabfb8e-b90e-4dc8-bef6-14bd392b1cec"},"authDateTime":1414485296046,"terminalId":"111113","authRefNum":"111111111111","paymentAmountInfo":{"paymentState":"DEPOSITED","approvedAmount":250000,"depositedAmount":250000,"refundedAmount":0},"bankInfo":{"bankName":"TEST CARD","bankCountryCode":"RU","bankCountryName":"Russia"}}],    "totalCount":2,"page":0,"pageSize":100} |

### Request for a payment through an external payment system

The paymentotherway.do request with specific parameters is used to pay for an order through an external payment system. Only the POST request is available.

This operation is available provided that the merchant has the corresponding permissions in the system.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection |
| password | AN..30 | yes | Store password received on the connection |
| MDORDER | ANS36 | yes | Order number received on its registration |
| paymentWay | ANS..\* | yes | Payment method is passed in this parameter. The possible values are:   * ALFA\_ALFACLICK – for a payment with "Alfa-click" (through the PayByClik system). * UPOP – for a payment through the UPOP system, for the holders of the China Union Pay cards. |
| language | A2 | no | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N1 | yes | Error code |
| error | ANS..\* | (on an error) | Error message |
| info | ANS..\* | no | Result of a payment attempt. The available values are presented below:   * Your payment has been processed, redirecting.. * Payment declined. Check the entered data and that there are enough funds on the card and repeat the operation. Redirecting... * Sorry, the payment cannot be processed. Redirecting... * Payment declined. Contact the merchant. Redirecting... * Payment declined. Contact the bank that issued the card. Redirecting... * Operation is impossible. Cardholder authentication completed unsuccessfully. Redirecting... * No connection to the bank. Try again later. Redirecting... * The data entering period has expired. Redirecting... * No response from the bank. Try again later. Redirecting... |
| redirect | ANS..\* | no | Return address after the payment |

Error codes (the errorCode fields):

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | Payment method is not specified or an incorrect value is entered |
| 2 | Order is not found |
| 5 | Session timeout |
| 5 | Access denied |
| 5 | The user must change the password |
| 5 | System error |

**An example of the POST request:**

|  |
| --- |
| language=ru&MDORDER=c96a734c-e2c9-429c-8fda-aaa0030c8a92&paymentWay=ALFA\_ALFACLICK |

**Response example:**

|  |
| --- |
| {"redirect":"http://testjmb.alfabank.ru/PayByClick/login.jsp?orderId=b37da970-e2b8-4729-a196-b4c2ab5bb401&backUrl=+","info":"Your  order is processed, redirecting...","errorCode":0} |

### Request for executing a payment by a binding

The paymentOrderBinding.do method is used to process a payment using a binding (see the Connection URLs section).

**Request parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection. |
| password | AN..30 | yes | Store password received on the connection. |
| mdOrder | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |
| bindingId | AN..255 | yes | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. |
| language | A2 | no | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used. |
| ip | NS..15 | yes | IP-address of the payer. |
| cvc | N..3 | no | CVC code. This parameter is mandatory, if "Can process payments without confirmation of CVC" is not selected for a merchant. |
| email | ANS..\* | no | Payer's email address. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| redirect | ANS..\* | no | On a success response in the case of an SSL-payment. URL to which a customer is redirected after a payment. |
| info | ANS..\* | no | On a successful response. Result of a payment attempt. The available values are presented below:   * Your payment has been processed, redirecting.. * Payment declined. Check the entered data, ensure that there are enough funds on the card. Redirecting... * The payment cannot be processed. Redirecting... * Payment declined. Contact the merchant. Redirecting... * Payment declined. Contact the bank that issued the card. Redirecting... * Operation is impossible. Cardholder authentication completed unsuccessfully. Redirecting... * No connection to the bank. Try again later. Redirecting... * Data entering timeout expiration. Redirecting... * No response from the bank. Try again later. Redirecting... |
| errorCode | N1 | yes | Error code. |
| errorMessage | AN..\* | no | On a response with an error. Error message. |
| error | AN..\* | no | On a response with an error. Error message. |
| acsUrl | ANS..\* | no | On a successful response in case of a 3D-Secure payment. URL to redirect to ACS. |
| paReq | ANS..\* | no | On a successful response in case of a 3D-Secure payment. Payment Authentication Request. |
| termUrl | ANS..\* | no | On a successful response in case of a 3D-Secure payment. URL to which to return from ACS. |

Error codes (the ErrorCode field):

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | It is necessary to specify CVC2/CVV2 because the merchant does not have a permission to process payments without CVC |
| 1 | Incorrect format of CVC |
| 1 | Incorrect language |
| 2 | The binding is not found |
| 2 | Order with this number is not found |
| 5 | Access denied |
| 5 | The user who is calling the service must change the password |
| 7 | System error |

**An example of the POST request:**

|  |
| --- |
| mdOrder=eb49300c-95b7-4dcd-9739-eee6c61f2ac4&bindingId=308042e8-2b28-484a-811e-f786c9776c3b&cvc=123 |

**An example of a success response for an SSL-payment:**

|  |
| --- |
| {"redirect":"http://ya.ru?orderId=eb49300c-95b7-4dcd-9739-eee6c61f2ac4","info":"Your payment has been processed, redirecting...","errorCode":0} |

**An example of a success response for a 3D-Secure payment:**

|  |
| --- |
| {"info":"Your payment hes been processed, forwarding...","acsUrl":"https://web.rbsuat.com/ab/acs/auth/start.do","paReq":"eJxVUdtugkAQ/RXCOy7LRdQMa2ixKU28pGrfyTICqSzKpcW/765AbR8mOWcyOWfmDCy74qx9YVXn\npfB1OjF1DQUvk1ykvn48vBgzfcngkFWI4R55WyGDNdZ1nKKWJ74+TVz05tPE8NyZbThOfDJmFjcN\ni55Mz+MJzu25zmAXvOOVwWDEpM/EAjJSqVjxLBYNg5hfn6INcyxvappABgoFVlHIPCA9ABEXyPb4\nhWKVp1mzyQUCuTeBl61oqhubOjaQkUBbnVnWNJcFId5sPuFlAUT1gDy8d61CtdTo8oStw+C7r5W5\nCVNZx9v6ENmyfCBqApK4QWaZ1KXUcjVqLVx7Ycu77n2IC2XOqDqjh3BRDsGj/5eDDLeS2Y+bjwyw\nu5QC5YRU/sVAHts+v6rceCODyfbb7m3bfmzD22dnlycaFHF+DGl0y6hK8z6kFHMZity7l1QEiJIh\nw6PI8GOJ/v3+BweMtyE=","termUrl":"https://web.rbsuat.com/:443/ab/rest/finish3ds.do","errorCode":0} |

**An example of a response containing an error:**

|  |
| --- |
| {"error":"Access denied","errorCode":5,"errorMessage":"Access denied"} |

### Request for deactivation of a binding

The unBindCard.do request is used to deactivate an existing binding (see the Connection URLs section).

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection. |
| password | AN..30 | yes | Store password received on the connection. |
| bindingId | AN..255 | yes | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Error description. |

**Error codes (the errorCode fields):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 2 | Incorrect binding status (when attempting to deactivate an inactive binding) |
| 2 | The binding is not found |
| 5 | Access denied |
| 5 | The user must change the password |
| 7 | System error |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/unBindCard.do?userName=userName&password=password&bindingId=fd3afc57-c6d0-4e08-aaef-1b7cfeb093dc |

**An example of the POST request:**

|  |
| --- |
| bindingId=fd3afc57-c6d0-4e08-aaef-1b7cfeb093dc |

**Response example:**

|  |
| --- |
| {"errorCode":"2","errorMessage":"Binging isn't active"} |

### Request for activation of a binding

The bindCard.do request is used to activate a binding that has been deactivated earlier (see the Connection URLs section).

**Request parameters**:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection. |
| password | AN..30 | yes | Store password received on the connection. |
| bindingId | AN..255 | yes | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Error description. |

**Error codes (the errorCode fields):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 2 | Incorrect status of the binding (on an attempt to activate an active binding) |
| 2 | The binding is not found |
| 5 | Access denied |
| 5 | The user must change the password |
| 7 | System error |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/bindCard.do?userName=userName&password=password&bindingId=fd3afc57-c6d0-4e08-aaef-1b7cfeb093dc |

**An example of the POST request:**

|  |
| --- |
| bindingId=fd3afc57-c6d0-4e08-aaef-1b7cfeb093dc |

**Response example:**

|  |
| --- |
| {"errorCode":"2","errorMessage":"Binding is active"} |

### Request for changing the validity period of a binding

The extendBinding.do request is used to extend the validity period of a binding (see the Connection URLs section).

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection. |
| password | AN..30 | yes | Store password received on the connection. |
| bindingId | ANS36 | yes | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. |
| newExpiry | N6 | yes | New date (the year and the month) of the end of the validity period of a binding, in the format YYYYMM. |
| language | A2 | no | Language in the ISO 639-1 encoding. If this parameter is not specified, the default language specified in the store settings is to be used. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N1 | yes | Completion code |
| errorMessage | ANS..\* | (on an error) | Error message |

**Error codes (the errorCode fields):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | One (or several) of the mandatory parameters is not specified or is specified incorrectly |
| 2 | The binding is not found |
| 5 | Access denied |
| 5 | The user must change the password |
| 7 | System error |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/extendBinding.do?userName=userName&password=password&bindingId=1eabfb8e-b90e-4dc8-bef6-14bd392b1cec&newExpiry=201612&language=ru |

**An example of the POST request:**

|  |
| --- |
| bindingId=1eabfb8e-b90e-4dc8-bef6-14bd392b1cec&newExpiry=201612&language=ru |

**Response example:**

|  |
| --- |
| {"errorCode":"0","errorMessage":"Success"} |

### Request for the list of binding of a customer

The getBindings.do request is used to get the list of bindings by a customer identifier.

**Request parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection. |
| password | AN..30 | yes | Store password received on the connection. |
| clientId | AN..255 | yes | Identifier of the customer in the store system passed on the order registration. This parameter is present only if a store has the permission to create bindings. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N1 | yes | Completion code |
| errorMessage | ANS..\* | (on an error) | Error message |
| *Binding element (it consists of bindingId, maskedPan and expiryDate):* |  |  |  |
| bindingId | AN..255 | no | Identifier of a binding created on an order payment or used to pay for an order. This parameter is present only if a store has the permission to create bindings. |
| maskedPan | N..19 | no | Masked number of the card that has been used for the payment. This parameter is to be specified only after the order has been paid. |
| expiryDate | N6 | no | Card expiration . This parameter is to be specified only after the order has been paid. |

Error codes (the errorCode fields):

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | [clientId] is not specified |
| 2 | The information is not found |
| 5 | Access denied |
| 5 | The user must change the password |
| 7 | System error |

**An example of the GET request:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/getBindings.do?userName=userName&password=password&clientId=client |

**An example of the POST request:**

|  |
| --- |
| clientId=client |

**Response example:**

|  |
| --- |
| {"bindings":[{"bindingId":"fd3afc57-c6d0-4e08-aaef-1b7cfeb093dc","maskedPan":"4000 00\*\* \*\*\*\* \*\*02","expiryDate":"201512"}],"errorCode":"0","errorMessage":"Success"} |

### Request for the list of binding of a bank card

Provided that a store has the corresponding permissions, it can get the list of all bindings that relate to a certain bank card. This can be done by a card number or by a known binding identifier.

The getBindingsByCardOrId.do method is used to get the list of bindings of a bank card.

All bindings that are available to the merchant according to the merchant settings are returned in the response.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| userName | AN..30 | yes | Store login received on the connection |
| password | AN..30 | yes | Store password received on the connection |
| pan | N..19 | no | Card number. The parameter is mandatory, unless bindingId is specified.  A search by the full number of a card is available to stores only provided that they have the corresponding permission. |
| bindingId | AN..255 | no | Binding identifier. The parameter is mandatory, unless pan is specified.  If the request contains pan, the bindingId value is ignored. |
| showExpired | boolean | no | This parameter defines the necessity to display bindings with expired card validity periods. The available values are: true, false. By default the parameter has the false value. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N1 | yes | Expiration code. |
| errorMessage | ANS..\* | yes | Description of the completion code. |
| *Bindings element (contains blocks that consist of the bindingId, maskedPan, expiryDate and clientId parameters):* |  |  |  |
| bindingId | AN..255 | no | Binding identifier. |
| maskedPan | N..19 | no | Masked number of the card that has been used for the payment. |
| expiryDate | N6 | no | The expiration date of the card validity period, in the format YYYYMM. |
| clientId | AN..255 | no | Number (identifier) of a customer in the merchant system. |

**Error codes (the** errorCode **field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | Neither a card number, nor a binding identifier is specified. |
| 2 | The information is not found. |
| 5 | Access denied. |
| 5 | The user must change the password. |
| 7 | System error. |

**Request example:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/getBindingsByCardOrId.do?userName=login&password=password&pan=4111111111111111 |

**Response parameters:**

|  |
| --- |
| {"errorCode":"0","errorMessage":"Success","bindings":[{"bindingId":"0b8edeb2-8380-4092-bf7e-1e1a78f2b15e","maskedPan":"411111\*\*1111","expiryDate":"201912","clientId":"12"},{"bindingId":"6a8c0738-cc88-4200-acf6-afc264d66cb0","maskedPan":"411111\*\*1111","expiryDate":"201912","clientId":"666"},{"bindingId":"97a70989-c1fb-49f7-8a42-27c19dc160dw","maskedPan":"411111\*\*1111","expiryDate":"201512","clientId":"666"}]} |

### Request for adding a card to the list of SSL-cards

The updateSSLCardList.do request is used to add the number of a card that was used when attempting to pay for an order to the list of SSL-cards.

The method is available only upon agreement with the bank. The user must be assigned the permissions to work with fraud lists.

**Request examples:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| mdorder | ANS36 | yes | Identifier of the order in the payment system. It is unique within the system. |

**Response parameters:**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Mandatory | Description |
| errorCode | N3 | no | Error code. |
| errorMessage | AN..512 | no | Description of the error in the default language of the user. |

**Error codes (the** errorCode **field):**

|  |  |
| --- | --- |
| Value | Description |
| 0 | The request has been processed without system errors. |
| 1 | Access denied |
| 2 | The user must change the password |
| 3 | Order number is not specified |
| 4 | Payment is absent from the system or an incorrect mdorder is specified |
| 6 | Card number is already present in the list |
| 7 | System error |

**Request example:**

|  |
| --- |
| https://web.rbsuat.com/ab/rest/updateSSLCardList.do?mdorder=a4d7d5e5-441d-401c-b25e-bd5d9fffe227&userName=987&password=111111 |

**Response example:**

|  |
| --- |
| {"errorCode":"0","errorMessage":"The request has been processed without system errors"} |

When registering a merchant, the merchant representative is provided with a login/password couple that must be used in the protocols.

**The description of the test service (WSDL) is stored at the following address:**

https://web.rbsuat.com/ab/webservices/merchant-ws?wsdl

**The URL to access the REST methods:**

|  |  |
| --- | --- |
| Method name | URL-address |
| Order registration | https://web.rbsuat.com/ab/rest/register.do |
| Order registration with pre-authorization | https://web.rbsuat.com/ab/rest/registerPreAuth.do |
| Order payment completion request | https://web.rbsuat.com/ab/rest/deposit.do |
| Order status request | https://web.rbsuat.com/ab/rest/getOrderStatus.do |
| Extended order status request | https://web.rbsuat.com/ab/rest/getOrderStatusExtended.do |
| Order payment cancellation request | https://web.rbsuat.com/ab/rest/reverse.do |
| Order payment refund request | https://web.rbsuat.com/ab/rest/refund.do |
| Request for checking a card for enrolment to 3D-Secure | https://web.rbsuat.com/ab/rest/verifyEnrollment.do |
| Request for adding additional parameters to an order | https://web.rbsuat.com/ab/rest/addParams.do |
| Request for payments statistics for a period | https://web.rbsuat.com/ab/rest/getLastOrdersForMerchants.do |
| Request for a payment through an external payment network | https://web.rbsuat.com/ab/rest/paymentotherway.do |
| Request for adding a card number to the list of SSL-cards | https://web.rbsuat.com/ab/rest/updateSSLCardList.do |
| Request for processing a payment by a binding | https://web.rbsuat.com/ab/rest/paymentOrderBinding.do |
| Request for a binding deactivation | https://web.rbsuat.com/ab/rest/unBindCard.do |
| Request for a binding activation | https://web.rbsuat.com/ab/rest/bindCard.do |
| Request for changing the validity period of a binding | https://web.rbsuat.com/ab/rest/extendBinding.do |
| Request for the list of bindings of a customer | https://web.rbsuat.com/ab/rest/getBindings.do |
| Request for the list of bindings of a certain bank card | https://web.rbsuat.com/ab/rest/getBindingsByCardOrId.do |
| Request for a payment through Apple Pay | https://web.rbsuat.com/ab/applepay/payment.do |
| Request for a recurring payment | https://web.rbsuat.com/ab/recurrentPayment.do |
| Request for a payment through Android Pay | https://web.rbsuat.com/ab/android/payment.do |
| Request for a payment through Samsung Pay | https://web.rbsuat.com/ab/samsung/payment.do |

### Request for a payment through Apple Pay

**Request to the payment gateway for a payment using Apple Pay**

The payment.do request is used to register an order (see [Connection URLs](#scroll-bookmark-9)).

|  |
| --- |
| Use the standard requests to the payment gateway for the operations of cancellation, refund and payment completion. |

**An example of the POST request is given below**

|  |
| --- |
| {"merchant":"merchant\_name","orderNumber":"applepay123456794","description":"descritpion\_text", "paymentToken":"","language":"ru","additionalParameters":{},"preAuth":"true"} |

|  |
| --- |
| It is necessary to add to the request a header with the definition of the type of contents - Content-Type: application/json to ensure that the request is processed correctly. |

The description of the parameters is given in the table below.

|  |  |  |
| --- | --- | --- |
| Parameter | Mandatory parameter | Description |
| merchant | Yes | Merchant login in the payment gateway system. |
| orderNumber | Yes | Unique identifier of the order on the merchant side. |
| description | No | Order description. |
| language | No | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used. |
| additionalParameters | No | Additional parameters of the order that are stored in the merchant personal area for the subsequent viewing. Additional parameters must be specified in the following format.   |  | | --- | | "parameter\_name": "parameter\_value" |   Each new pair of a parameter name and its value must be separated by a comma. |
| clientId | No | Customer identifier for which a binding for recurring payments is to be created. Specify it only if the payment is dummy and is intended for subsequent recurring payments. |
| preAuth | No | Parameter that defines the necessity of a pre-authorization (putting the amount on hold on the customer's account until its debiting). The following values are available:   * true - the parameter is enabled, a payment is processed with a pre-authorization (the amount on the customer's account is put on hold until the debiting); * false - the parameter is disabled (the amount is debited immediately).   If the parameter is not specified in the request, the amount is debited immediately. |
| paymentToken | Yes | The paymentToken parameter must contain a Base64 encoded value of the paymentData property that was received in PKPaymentToken Object from the Apple Pay system (see [the documentation for Apple Pay](https://developer.apple.com/library/ios/documentation/PassKit/Reference/PaymentTokenJSON/PaymentTokenJSON.html)). Thus, to send a payment request to the payment gateway, the merchant must:   1. Receive from the Apple Pay system the PKPaymentToken Object object containing the paymentData property; 2. Extract the value of the paymentData property and encode it in Base64; 3. Include the encoded value of the paymentData property as the value of the paymentToken parameter in the payment request that the merchant sends to the payment gateway.   Thus, the merchant receives from Apple PKPaymentToken Object that looks as follows:   |  | | --- | | {"paymentData":{"data":"vj5Uvux7Im8DD8YhSOsJvw5lWmfl2HMUnTNWJhVfTehvFffRhDo54mfpjxMt9vJdp6DwD7fgcNHDxBvnj56qYG4DpOxg1fTSdXgPFrezprZHCrRxPhN\/aQQEThe2pQ0c7hgzzZlA6TpkIR\/Xtk6CTcEbD1W6znFVdvMgX8G96Gg4OAGl8GaTXdSU3wlMQL5E63CLQzPi1xHVErWl1OOn6hYQuREUDGc7mAjmqMyLwX+p6mOwJZ6ZFO\/b9HkgFi428rqtOH08AfqkfaIWwIIAz2w3xEoZrDXbgFpNBnN7F2oretCU1\/dFvQJjDYbMorKQ8+GJbWtlsVb+Ksy0U91eoUetDcyMpB9zc139STYVoC8yp6Yk6Mn3icCLY0ZBujq7\/404kMGpnHgkNVqFc\/4SN0U2XQ5rrb14DM8M69w=","signature":"\/0komJPnwPE6OCAhEwggINMEUGCCsGAQUFBwEBBDkwNzA1BggrBgEFBQcwAYYpaHR0cDovL29jc3AuYXBwbGUuY29tL29jc3AwNC1hcHBsZWFpY2EzMDEwHQYDVR0OBBYEFJRX22\/VdIGGiYl2L35XhQfnm1gkMAwGA1UdEwEB\/wQCMAAwHwYDVR0jBBgwFoAUI\/JJxE+T5O8n5sT2KGw\/orv9LkswggEdBgNVHSAEggEUMIIBEDCCAQwGCSqGSIb3Y2QFATCB\/jCBwwYIKwYBBQUHAgIwgbYMgbNSZWxpYW5jZSBvbiB0aGlzIGNlcnRpZmljYXRlIGJ5IGFueSBwYXJ0eSBhc3N1bWVzIGFjY2VwdGFuY2Ugb2YgdGhlIHRoZW4gYXBwbGljYWJsZSBzdGFuZGFyZCB0ZXJtcyBhbmQgY29uZGl0aW9ucyBvZiB1c2UsIGNlcnRpZmljYXRlIHBvbGljeSBhbmQgY2VydGlmaWNhdGlvbiBwcmFjdGljZSBzdGF0ZW1lbnRzLjA2BggrBgEFBQcCARYqaHR0cDovL3d3dy5hcHBsZS5jb20vY2VydGlmaWNhdGVhdXRob3JpdHkvMDQGA1UdHwQtMCswKaAnoCWGI2h0dHA6Ly9jcmwuYXBwbGUuY29tL2FwcGxlYWljYTMuY3JsMA4GA1UdDwEB\/wQEAwIHgDAPBgkqhkiG92NkBh0EAgUAMAoGCCqGSM49BAMCA0gAMEUCIHKKnw+Soyq5mXQr1V62c0BXKpaHodYu9TWXEPUWPpbpAiEAkTecfW6+W5l0r0ADfzTCPq2YtbS39w01XIayqBNy8bEwggLuMIICdaADAgECAghJbS+\/OpjalzAKBggqhkjOPQQDAjBnMRswGQYDVQQDDBJBcHBsZSBSb290IENBIC0gRzMxJjAkBgNVBAsMHUFwcGxlIENlcnRpZmljYXRpb24gQXV0aG9yaXR5MRMwEQYDVQQKDApBcHBsZSBJbmMuMQswCQYDVQQGEwJVUzAeFw0xNDA1MDYyMzQ2MzBaFw0yOTA1MDYyMzQ2MzBaMHoxLjAsBgNVBAMMJUFwcGxlIEFwcGxpY2F0aW9uIEludGVncmF0aW9uIENBIC0gRzMxJjAkBgNVBAsMHUFwcGxlIENlcnRpZmljYXRpb24gQXV0aG9yaXR5MRMwEQYDVQQKDApBcHBsZSBJbmMuMQswCQYDVQQGEwJVUzBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABPAXEYQZ12SF1RpeJYEHduiAou\/ee65N4I38S5PhM1bVZls1riLQl3YNIk57ugj9dhfOiMt2u2ZwvsjoKYT\/VEWjgfcwgfQwRgYIKwYBBQUHAQEEOjA4MDYGCCsGAQUFBzABhipodHRwOi8vb2NzcC5hcHBsZS5jb20vb2NzcDA0LWFwcGxlcm9vdGNhZzMwHQYDVR0OBBYEFCPyScRPk+TvJ+bE9ihsP6K7\/S5LMA8GA1UdEwEB\/wQFMAMBAf8wHwYDVR0jBBgwFoAUu7DeoVgziJqkipnevr3rr9rLJKswNwYDVR0fBDAwLjAsoCqgKIYmaHR0cDovL2NybC5hcHBsZS5jb20vYXBwbGVyb290Y2FnMy5jcmwwDgYDVR0PAQH\/BAQDAgEGMBAGCiqGSIb3Y2QGAg4EAgUAMAoGCCqGSM49BAMCA2cAMGQCMDrPcoNRFpmxhvs1w1bKYr\/0F+3ZD3VNoo6+8ZyBXkK3ifiY95tZn5jVQQ2PnenC\/gIwMi3VRCGwowV3bF3zODuQZ\/0XfCwhbZZPxnJpghJvVPh6fRuZy5sJiSFhBpkPCZIdAAAxggGMMIIBiAIBATCBhjB6MS4wLAYDVQQDDCVBcHBsZSBBcHBsaWNhdGlvbiBJbnRlZ3JhdGlvbiBDQSAtIEczMSYwJAYDVQQLDB1BcHBsZSBDZXJ0aWZpY2F0aW9uIEF1dGhvcml0eTETMBEGA1UECgwKQXBwbGUgSW5jLjELMAkGA1UEBhMCVVMCCCRD8qgGnfV3MA0GCWCGSAFlAwQCAQUAoIGVMBgGCSqGSIb3DQEJAzELBgkqhkiG9w0BBwEwHAYJKoZIhvcNAQkFMQ8XDTE3MDMxNzEwMzgzOVowKgYJKoZIhvcNAQk0MR0wGzANBglghkgBZQMEAgEFAKEKBggqhkjOPQQDAjAvBgkqhkiG9w0BCQQxIgQgvL+q07\/reM0N\/5b0hwWT7TJReVTdS9QX5SPhiqeie+cwCgYIKoZIzj0EAwIERzBFAiEAttC68Xyzs6I0+tAKmg6x+0UrqmkQN\/V5c8RMMIEJHooCIHIgUHbAt2p5WrFHQKrAVL4c7nohRplZWVbVu6wbBeCgAAAAAAAA","header":{"publicKeyHash":"fpvAnSDwQFX4NX4pghdjpNwUFhoTH\/DDGhew94uJaRA=","ephemeralPublicKey":"MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAErKZUfqvhlieGAOaCKeTB\/oDEo29fS1jWSKemNDh3fIqmbfs86nL4BGtRsWRxWcMnHN6GFOQm1MEj4m7ZHxe78g==","transactionId":"38e4c267ef1de62a343d0eccada3f7e19f6b22ffc7ede899c039865432ba6aa2"},"version":"EC\_v1"},"transactionIdentifier":"38E4C267EF1DE62A343D0ECCADA3F7E19F6B22FFC7EDE899C039865432BA6AA2","paymentMethod":{"network":"Visa","type":"debit","displayName":"Visa 5223"}} |   The value of the paymentData property (from the example above) that is to be encoded in Base64 and to be passed in the payment request to the payment gateway looks as follows:   |  | | --- | | {"data":"vj5Uvux7Im8DD8YhSOsJvw5lWmfl2HMUnTNWJhVfTehvFffRhDo54mfpjxMt9vJdp6DwD7fgcNHDxBvnj56qYG4DpOxg1fTSdXgPFrezprZHCrRxPhN\/aQQEThe2pQ0c7hgzzZlA6TpkIR\/Xtk6CTcEbD1W6znFVdvMgX8G96Gg4OAGl8GaTXdSU3wlMQL5E63CLQzPi1xHVErWl1OOn6hYQuREUDGc7mAjmqMyLwX+p6mOwJZ6ZFO\/b9HkgFi428rqtOH08AfqkfaIWwIIAz2w3xEoZrDXbgFpNBnN7F2oretCU1\/dFvQJjDYbMorKQ8+GJbWtlsVb+Ksy0U91eoUetDcyMpB9zc139STYVoC8yp6Yk6Mn3icCLY0ZBujq7\/404kMGpnHgkNVqFc\/4SN0U2XQ5rrb14DM8M69w=","signature":"\/0komJPnwPE6OCAhEwggINMEUGCCsGAQUFBwEBBDkwNzA1BggrBgEFBQcwAYYpaHR0cDovL29jc3AuYXBwbGUuY29tL29jc3AwNC1hcHBsZWFpY2EzMDEwHQYDVR0OBBYEFJRX22\/VdIGGiYl2L35XhQfnm1gkMAwGA1UdEwEB\/wQCMAAwHwYDVR0jBBgwFoAUI\/JJxE+T5O8n5sT2KGw\/orv9LkswggEdBgNVHSAEggEUMIIBEDCCAQwGCSqGSIb3Y2QFATCB\/jCBwwYIKwYBBQUHAgIwgbYMgbNSZWxpYW5jZSBvbiB0aGlzIGNlcnRpZmljYXRlIGJ5IGFueSBwYXJ0eSBhc3N1bWVzIGFjY2VwdGFuY2Ugb2YgdGhlIHRoZW4gYXBwbGljYWJsZSBzdGFuZGFyZCB0ZXJtcyBhbmQgY29uZGl0aW9ucyBvZiB1c2UsIGNlcnRpZmljYXRlIHBvbGljeSBhbmQgY2VydGlmaWNhdGlvbiBwcmFjdGljZSBzdGF0ZW1lbnRzLjA2BggrBgEFBQcCARYqaHR0cDovL3d3dy5hcHBsZS5jb20vY2VydGlmaWNhdGVhdXRob3JpdHkvMDQGA1UdHwQtMCswKaAnoCWGI2h0dHA6Ly9jcmwuYXBwbGUuY29tL2FwcGxlYWljYTMuY3JsMA4GA1UdDwEB\/wQEAwIHgDAPBgkqhkiG92NkBh0EAgUAMAoGCCqGSM49BAMCA0gAMEUCIHKKnw+Soyq5mXQr1V62c0BXKpaHodYu9TWXEPUWPpbpAiEAkTecfW6+W5l0r0ADfzTCPq2YtbS39w01XIayqBNy8bEwggLuMIICdaADAgECAghJbS+\/OpjalzAKBggqhkjOPQQDAjBnMRswGQYDVQQDDBJBcHBsZSBSb290IENBIC0gRzMxJjAkBgNVBAsMHUFwcGxlIENlcnRpZmljYXRpb24gQXV0aG9yaXR5MRMwEQYDVQQKDApBcHBsZSBJbmMuMQswCQYDVQQGEwJVUzAeFw0xNDA1MDYyMzQ2MzBaFw0yOTA1MDYyMzQ2MzBaMHoxLjAsBgNVBAMMJUFwcGxlIEFwcGxpY2F0aW9uIEludGVncmF0aW9uIENBIC0gRzMxJjAkBgNVBAsMHUFwcGxlIENlcnRpZmljYXRpb24gQXV0aG9yaXR5MRMwEQYDVQQKDApBcHBsZSBJbmMuMQswCQYDVQQGEwJVUzBZMBMGByqGSM49AgEGCCqGSM49AwEHA0IABPAXEYQZ12SF1RpeJYEHduiAou\/ee65N4I38S5PhM1bVZls1riLQl3YNIk57ugj9dhfOiMt2u2ZwvsjoKYT\/VEWjgfcwgfQwRgYIKwYBBQUHAQEEOjA4MDYGCCsGAQUFBzABhipodHRwOi8vb2NzcC5hcHBsZS5jb20vb2NzcDA0LWFwcGxlcm9vdGNhZzMwHQYDVR0OBBYEFCPyScRPk+TvJ+bE9ihsP6K7\/S5LMA8GA1UdEwEB\/wQFMAMBAf8wHwYDVR0jBBgwFoAUu7DeoVgziJqkipnevr3rr9rLJKswNwYDVR0fBDAwLjAsoCqgKIYmaHR0cDovL2NybC5hcHBsZS5jb20vYXBwbGVyb290Y2FnMy5jcmwwDgYDVR0PAQH\/BAQDAgEGMBAGCiqGSIb3Y2QGAg4EAgUAMAoGCCqGSM49BAMCA2cAMGQCMDrPcoNRFpmxhvs1w1bKYr\/0F+3ZD3VNoo6+8ZyBXkK3ifiY95tZn5jVQQ2PnenC\/gIwMi3VRCGwowV3bF3zODuQZ\/0XfCwhbZZPxnJpghJvVPh6fRuZy5sJiSFhBpkPCZIdAAAxggGMMIIBiAIBATCBhjB6MS4wLAYDVQQDDCVBcHBsZSBBcHBsaWNhdGlvbiBJbnRlZ3JhdGlvbiBDQSAtIEczMSYwJAYDVQQLDB1BcHBsZSBDZXJ0aWZpY2F0aW9uIEF1dGhvcml0eTETMBEGA1UECgwKQXBwbGUgSW5jLjELMAkGA1UEBhMCVVMCCCRD8qgGnfV3MA0GCWCGSAFlAwQCAQUAoIGVMBgGCSqGSIb3DQEJAzELBgkqhkiG9w0BBwEwHAYJKoZIhvcNAQkFMQ8XDTE3MDMxNzEwMzgzOVowKgYJKoZIhvcNAQk0MR0wGzANBglghkgBZQMEAgEFAKEKBggqhkjOPQQDAjAvBgkqhkiG9w0BCQQxIgQgvL+q07\/reM0N\/5b0hwWT7TJReVTdS9QX5SPhiqeie+cwCgYIKoZIzj0EAwIERzBFAiEAttC68Xyzs6I0+tAKmg6x+0UrqmkQN\/V5c8RMMIEJHooCIHIgUHbAt2p5WrFHQKrAVL4c7nohRplZWVbVu6wbBeCgAAAAAAAA","header":{"publicKeyHash":"fpvAnSDwQFX4NX4pghdjpNwUFhoTH\/DDGhew94uJaRA=","ephemeralPublicKey":"MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAErKZUfqvhlieGAOaCKeTB\/oDEo29fS1jWSKemNDh3fIqmbfs86nL4BGtRsWRxWcMnHN6GFOQm1MEj4m7ZHxe78g==","transactionId":"38e4c267ef1de62a343d0eccada3f7e19f6b22ffc7ede899c039865432ba6aa2"},"version":"EC\_v1"} | |

**The examples and the response description**

**A successful payment**

|  |
| --- |
| { "success":true, "data": {  "orderId": "12312312123"  } } |

**A failed payment**

|  |
| --- |
| {  "error": {  "code": 1,  "description": "Processing Error",  "message": "The funds on the card are not sufficient"  },  "success": false } |

The description of the response parameters is given in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Nested parameter | Mandatory parameter | Description |
| success | Not relevant | Yes | Designates a successful payment. The following values are available:   * **true** - the payment is processed successfully; * **false** - the payment failed. |
| data (this parameter is returned only if the payment is processed successfully) | orderId | Yes | Order number in the payment system unique for the merchant. |
| error  (this parameter is returned only if the payment failed) | code | Yes | Error code. |
| description | Yes | A detailed technical explanation of the error - the contents of this parameter is not to be displayed to the customer. |
| message | Yes | Comprehensive error description - it is intended for displaying to the user. |

### Request for executing recurring payments through Apple Pay

**A request for processing a recurring payment through Apple Pay**

The recurrentPayment.do request is used to register an order (see [Connection URLs](#scroll-bookmark-9)).

**An example of the POST request is given below**

|  |
| --- |
| {  "userName": "userName",  "password": "password",  "orderNumber": "UAF-203974-DE-12",  "language": "RU",  "bindingId": "binding\_id",  "amount": 12300,  "currency": "810",  "description" : "Test description",  "additionalParameters": {  "firstParamName": "firstParamValue",  "secondParamName": "secondParamValue"  }  } |

The description of the parameters is presented in the table below.

|  |  |  |
| --- | --- | --- |
| Parameter | Mandatory | Description |
| userName | Yes | Name of the user who has the access to the payment gateway API. |
| password | Yes | Password of the user who has the access to the payment gateway API. |
| orderNumber | Yes | Order number. |
| language | No | Two-letter language code. |
| bindingId | Yes | Binding identifier. |
| amount | Yes | Order amount in the minor denomination of the currency (for example, in kopeks). |
| currency | No | Numberic ISO 4217 code of the currency. |
| description | No | Order description. |
| additionalParameters | No | "parameter 1 name": "parameter 1 value", "parameter 2 name": "parameter 2 value" |

**The examples and description of the response**

Below an example of a successful payment is given.

|  |
| --- |
| {"success":true,"data":{"orderId":"f7beebe4-7c9a-43cf-8e26-67ab741f9b9e"},"orderStatus":{"errorCode":"0","orderNumber":"UAF-203974-DE-12","orderStatus":2,"actionCode":0,"actionCodeDescription":"","amount":12300,"currency":"810","date":1491333938243,"orderDescription":"Test description","merchantOrderParams":[{"name":"firstParamName","value":"firstParamValue"},{"name":"secondParamName","value":"secondParamValue"}],"attributes":[],"cardAuthInfo":{"expiration":"201912","cardholderName":"sdf sdf","approvalCode":"123456","paymentSystem":"VISA","pan":"411111\*\*1111"},"authDateTime":1491333939454,"terminalId":"11111","authRefNum":"111111111111","paymentAmountInfo":{"paymentState":"DEPOSITED","approvedAmount":12300,"depositedAmount":12300,"refundedAmount":0},"bankInfo":{"bankCountryName":"<Unknown>"},"chargeback":false,"operations":[{"amount":12300,"cardHolder":"sdf sdf","authCode":"123456"}]}} |

Below an example of a failed payment is given.

|  |
| --- |
| {  "error": {  "code": "10",  "description": "An order with this number is already registered.",  "message": "An order with this number is already registered."  },  "success": false } |

The description of the response parameters is given in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Nested parameter | Mandatory parameter | Description |
| success | Not relevant | Yes | Designates a successful payment. The following values are available:   * **true** - the payment is processed successfully; * **false** - the payment failed. |
| data (this parameter is returned only if the payment is processed successfully) | orderId | Yes | Order number in the payment system unique for the merchant. |
| error  (this parameter is returned only if the payment failed) | code | Yes | Error code. |
| description | Yes | A detailed technical explanation of the error - the contents of this parameter is not to be displayed to the customer. |
| message | Yes | Comprehensive error description - it is intended for displaying to the user. |

**Error codes**

|  |  |
| --- | --- |
| Error code | Message |
| 0 | Processing executed without system errors. |
| 1 | Incorrect payment details. |
| 1 | Impossible to use the binding for recurring payments. |
| 1 | Incorrect payment parameters. |
| 1 | Wrong order number. |
| 4 | Incorrect binding identifier. |
| 5 | Using binding is disabled. |
| 5 | Incorrect amount. |
| 5 | Incorrect login. |
| 5 | Incorrect password. |
| 5 | Authentication error. |
| 5 | Authentication error. |
| 10 | An order with this number has already been registered. |

### Request for a payment through Android Pay

**Request to the payment gateway for a payment using Android Pay**

The payment.do request is used to register an order (see [Connection URLs](#scroll-bookmark-9)).

|  |
| --- |
| Use the standard requests to the payment gateway for the operations of cancellation, refund and payment completion. |

Below an example of a payment request is given.

|  |
| --- |
| It is necessary to add to the request a header with the definition of the type of contents - Content-Type: application/json to ensure that the request is processed correctly. |

|  |
| --- |
| {  "merchant": "OurBestMerchantLogin",  "orderNumber": "UAF-203974-DE",  "language": "RU",  "preAuth": true,  "description" : "Test description",  "additionalParameters":   {  "firstParamName": "firstParamValue",  "secondParamName": "secondParamValue"  },  "paymentToken": "\x7B\x0A\x20\x20\xE2\x80\x9C\x65\x6E\x63\x72\x79\x70\x74\x65\x64\x4D\x65\x73\x73\x61\x67\x65\xE2\x80\x9D\x3A\x20\xE2\x80\x9C\x5A\x57\x35\x6A\x63\x6E\x6C\x77\x64\x47\x56\x6B\x54\x57\x56\x7A\x63\x32\x46\x6E\x5A\x51\x3D\x3D\xE2\x80\x9D\x2C\x0A\x20\x20\xE2\x80\x9C\x65\x70\x68\x65\x6D\x65\x72\x61\x6C\x50\x75\x62\x6C\x69\x63\x4B\x65\x79\xE2\x80\x9D\x3A\x20\xE2\x80\x9C\x5A\x58\x42\x6F\x5A\x57\x31\x6C\x63\x6D\x46\x73\x55\x48\x56\x69\x62\x47\x6C\x6A\x53\x32\x56\x35\xE2\x80\x9D\x2C\x0A\x20\x20\x22\x74\x61\x67\x22\x3A\x20\xE2\x80\x9D\x63\x32\x6C\x6E\x62\x6D\x46\x30\x64\x58\x4A\x6C\xE2\x80\x9D\x0A\x7D",  "ip" : "127.0.0.1",  "amount" : "230000",  "currencyCode" : 810 } |

The description of the parameters is presented in the table below.

|  |  |  |
| --- | --- | --- |
| Parameter | Mandatory | Description |
| merchant | Yes | Merchant login in the payment gateway system. |
| orderNumber | Yes | Unique identifier of the order on the merchant side. |
| description | No | Order description. |
| language | No | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used. |
| additionalParameters | No | Additional parameters of the order that are stored in the merchant personal area for the subsequent viewing. Additional parameters must be specified in the following format.   |  | | --- | | "parameter\_name": "parameter\_value" |   Each new pair of a parameter name and its value must be separated by a comma. |
| preAuth | No | Parameter that defines the necessity of a pre-authorization (putting the amount on hold on the customer's account until its debiting). The following values are available:   * true - the parameter is enabled, a payment is processed with a pre-authorization (the amount on the customer's account is put on hold until the debiting); * false - the parameter is disabled (the amount is debited immediately).   If the parameter is not specified in the request, the amount is debited immediately. |
| clientId | No | Customer identifier for which a binding for recurring payments is to be created. Specify it only if the payment is dummy and is intended for subsequent recurring payments. |
| paymentToken | Yes | JSON-object that contains a token received from Android Pay. This parameter must be in the UTF-8 encoding. |
| ip | Yes | IP-address of the payer. |
| amount | Yes | Payment amount in the minimum denomination of the currency (for example, in kopeks). |
| currencyCode | No | Numeric ISO 4217 code of the payment currency. If this parameter is not specified, it is considered to be equal to the default currency code. |

**The examples and the response description**

**A successful payment**

|  |
| --- |
| { "success":true, "data": {  "orderId": "12312312123"  } |

**A failed payment**

|  |
| --- |
| {  "error": {  "code": 1,  "description": "Processing Error",  "message": "The funds on the card are not sufficient"  },  "success": false } |

The description of the response parameters is given in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Nested parameter | Mandatory parameter | Description |
| success | Not relevant | Yes | Designates a successful payment. The following values are available:   * **true** - the payment is processed successfully; * **false** - the payment failed. |
| data (this parameter is returned only if the payment is processed successfully) | orderId | Yes | Order number in the payment system unique for the merchant. |
| error  (this parameter is returned only if the payment failed) | code | Yes | Error code. |
| description | Yes | A detailed technical explanation of the error - the contents of this parameter is not to be displayed to the customer. |
| message | Yes | Comprehensive error description - it is intended for displaying to the user. |

The description of the possible error codes is given in the table below.

|  |  |
| --- | --- |
| Error code | Error message |
| 0 | The request has been processed without system errors. |
| 1 | The funds on the card are not sufficient |
| 5 | Access denied |
| The user must change the password |
| 7 | System error |
| 10 | Incorrect value of the [paymentToken] parameter |
| Incorrect value of the [orderNumber] parameter |
| Incorrect value of the [merchant] parameter |
| Incorrect value of the [ip] parameter |
| Encryption of the data passed in unsuccessful |

### Request for a payment through Samsung Pay

The payment.do request is used to register an order in Samsung Pay.

|  |
| --- |
| Use the standard requests to the payment gateway for the operations of cancellation, refund and payment completion. |

Below an example of a request for a payment through Samsung Pay is given.

|  |
| --- |
| It is necessary to add to the request a header with the definition of the type of contents - Content-Type: application/json to ensure that the request is processed correctly. |

|  |
| --- |
| {  "merchant": "OurBestMerchantLogin",  "orderNumber": "UAF-203974-DE",  "language": "RU",  "preAuth": true,  "description" : "Test description",  "additionalParameters":  {  "firstParamName": "firstParamValue",  "secondParamName": "secondParamValue"  },  "paymentToken": "",  "ip" : "127.0.0.1" } |

The description of the parameters is given in the table below.

|  |  |  |
| --- | --- | --- |
| Name | Mandatory | Description |
| merchant | Yes | Merchant login in the payment gateway system. |
| orderNumber | Yes | Unique identifier of the order on the merchant side. |
| description | No | Order description. |
| language | No | Language in the ISO 639-1 encoding. If the language is not specified, the default language specified in the store settings is used. |
| additionalParameters | No | Additional parameters of the order that are stored in the merchant personal area for the subsequent viewing. Additional parameters must be specified in the following format.   |  | | --- | | "parameter\_name": "parameter\_value" |   Each new pair of a parameter name and its value must be separated by a comma. |
| preAuth | No | Parameter that defines the necessity of a pre-authorization (putting the amount on hold on the customer's account until its debiting). The following values are available:   * true - the parameter is enabled, a payment is processed with a pre-authorization (the amount on the customer's account is put on hold until the debiting); * false - the parameter is disabled (the amount is debited immediately).   If the parameter is not specified in the request, the amount is debited immediately. |
| clientId | No | Customer identifier for which a binding for recurring payments is to be created. Specify it only if the payment is dummy and is intended for subsequent recurring payments. |
| paymentToken | Yes | The contents of the 3ds.data parameter from the response received from Samsung Pay. |
| ip | Yes | IP-address of the payer. |
| currencyCode | No | Numeric ISO 4217 code of the payment currency. If this parameter is not specified, it is considered to be equal to the default currency code. |

**The examples and the response description**

**A successful payment**

|  |
| --- |
| { "success":true, "data": {  "orderId": "12312312123"  } } |

**A failed payment**

|  |
| --- |
| {  "error": {  "code": 1,  "description": "Processing Error",  "message": "The funds on the card are not sufficient"  },  "success": false } |

The description of the response parameters is presented in the table below.

|  |  |  |  |
| --- | --- | --- | --- |
| Parameter | Nested parameter | Mandatory | Description |
| success | Not relevant | Yes | Designates a successful payment. The following values are available:   * **true** - the payment is processed successfully; * **false** - the payment failed. |
| data (this parameter is returned only if the payment is processed successfully) | orderId | Yes | Order number in the payment system unique for the merchant. |
| error  (this parameter is returned only if the payment failed) | code | Yes | Error code. |
| description | Yes | A detailed technical explanation of the error - the contents of this parameter is not to be displayed to the customer. |
| message | Yes | Comprehensive error description - it is intended for displaying to the user. |

The description of the possible error codes is given in the table below.

|  |  |
| --- | --- |
| Code | Description |
| 0 | The request has been processed without system errors. |
| 1 | The funds on the card are not sufficient |
| 5 | Access denied |
| The user must change the password |
| 7 | System error |
| 10 | Incorrect value of the [paymentToken] parameter |
| Incorrect value of the [orderNumber] parameter |
| Incorrect value of the [merchant] parameter |
| Incorrect value of the [ip] parameter |
| Incorrect value of the [paymentToken.header.alg] parameter |
| Incorrect value of the [paymentToken.header.enc] parameter |
| Incorrect value of the [paymentToken.header.typ] parameter |
| Incorrect value of the [paymentToken.header.channelSecurityContext] parameter |
| Incorrect value of the [paymentToken.header.kid] parameter |
| Encryption of the data passed in unsuccessful |

Test cards

Specify at least two words in Latin alphabet letters as Cardholder name.

For all cards enrolled to 3-D Secure (**veres=y, pares=y или a**) the ACS password is: 12345678.

|  |
| --- |
| The test environment is not designed for the load-testing: if necessary to conduct such a testing, contact the Bank. |

Test cards:

|  |
| --- |
| **pan:** **4111 1111 1111 1111** **exp date:** **2019/12** **cvv2:** **123** **3dsecure: veres=y, pares=y** |

|  |
| --- |
| **pan:** **5100 0000 0000 0008** **exp date:** **2017/12** **cvv2:** **123** **3dsecure: veres=y, pares=y** |

|  |
| --- |
| **pan: 6011 0000 0000 0004** **exp date:** **2019/12** **cvv2:** **123** **3dsecure: veres=y, pares=y** |

|  |
| --- |
| **pan:** **6390 0200 0000 000003** **exp date:** **2019/12** **cvv2:** **123(optional parameter)** **3dsecure: veres=y, pares=a** |

|  |
| --- |
| **pan:** **5555 5555 5555 5599** **exp date:** **2019/12** **cvv2:** **123** **3dsecure: veres=n** |

|  |
| --- |
| **pan:** **4444 0000 0000 1111** **exp date:** **2019/12** **cvv2:** **123** **3dsecure: veres=n** |

|  |
| --- |
| **pan:** **2200 0000 0000 0004** **exp date:** **2019/12** **cvc:** **123 3dsecure: veres=n** |

|  |
| --- |
| **pan:** **2200 0000 0000 0012** **exp date:** **2019/12** **cvc:** **123 3dsecure: veres=y pares=n** |

|  |
| --- |
| **pan:** **2200 0000 0000 0020** **exp date:** **2019/12** **cvc:** **123 3dsecure: veres=u** |

|  |
| --- |
| **pan:** **2200 0000 0000 0038** **exp date:** **2019/12** **cvc:** **123 3dsecure: veres=y pares=u** |

|  |
| --- |
| **pan: 2200 0000 0000 0046 exp date: 2019/12  cvc: 123 3dsecure: veres=y pares=a** |

|  |
| --- |
| **pan: 2200 0000 0000 0053 exp date: 2019/12  cvc: 123 3dsecure: veres=y pares=y** |

|  |
| --- |
| **pan: 2200 0000 0000 0053**  **exp date: 2019/12**  **cvv2: 123** **3dsecure: VeRes=Y, PaRes=Y** |

|  |
| --- |
| **pan: 2200 0000 0000 0046**  **exp date: 2019/12**  **cvv2: 123** **3dsecure: VeRes=Y, PaRes=A** |

|  |
| --- |
| **pan: 2200 0000 0000 0012** **exp date: 2019/12**  **cvv2: 123** **3dsecure: VeRes=Y, PaRes=N** |

|  |
| --- |
| **pan: 2200 0000 0000 0038** **exp date: 2019/12**  **cvv2: 123** **3dsecure: VeRes=Y, PaRes=U** |

|  |
| --- |
| **pan: 2200 0000 0000 0020** **exp date: 2019/12**  **cvv2:** 123 **3dsecure: VeRes=U** |

|  |
| --- |
| **pan: 2200 0000 0000 0004** **exp date: 2019/12**  **cvv2: 123** **3dsecure: VeRes=N** |

Cards that return errors /  
Cards returning errors:

|  |
| --- |
| **pan:** **5555 5555 5555 5557** **exp date:** **2019/12** **cvv2:** **123** **3dsecure: veres=y, pares=u** |

|  |
| --- |
| **pan:** **4444 3333 2222 1111** **exp date:** **2019/12** **cvv2:** **123** **3dsecure: veres=y, pares=u** |

Declined. PaRes status is U (-2011)

|  |
| --- |
| **pan:** **4000 0000 0000 0002** **exp date:** **2019/12** **cvv2:** **123** **3dsecure: veres=u** |

|  |
| --- |
| **pan:** **5555 5555 4444 4442** **exp date:** **2019/12** **cvv2:** **123** **3dsecure: veres=u** |

Declined. VeRes status is U (-2016)

|  |
| --- |
| **pan:** **4444 4444 4444 4422** **exp date:** **2019/12** **cvv2:** **123** |

Invalid message format (913)

|  |
| --- |
| **pan:** **4444 4444 4444 4455** **exp date:** **2019/12** **cvv2:** **123** |

Card limitations exceeded (902)

|  |
| --- |
| **pan:** **4444 4444 4444 3333** **exp date:** **2019/12** **cvv2:** **123** |

Limit exceeded (123)

|  |
| --- |
| **pan:** **4444 4444 4444 6666** **exp date:** **2019/12** **cvv2:** **123** |

BLOCKED\_BY\_LIMIT (-20010)

|  |
| --- |
| **pan:** **4444 4444 1111 1111** **exp date:** **2019/12** **cvv2:** **123** |

Transaction rejected by the network (5)

|  |
| --- |
| **pan:** **4444 4444 9999 9999** **exp date:** **2019/12** **cvv2:** **123** |

TDSEC\_COMM\_ERROR (151017)

|  |
| --- |
| **pan:** **5432 5432 5432 5430** **exp date:** **2018/08** **cvv2:** **521** |

INSUFFICIENT\_FUNDS (116)

Appendix 1. External fee for payments

The possibility for a merchant to use this functionality must be agreed with the Bank.

Supplement to the description of the payment page

1. It is necessary to enable the check for the presense of a fee in the payment script settings (in the page header):

|  |
| --- |
| getFeeEnabled: true |

2. The payment script executes a request to the payment gateway to define whether a fee is to be charged. In case of a confirming response (true), the fee amount is displayed on the payment page.

To enable displaying the fee, the payment page body must contain the following block:

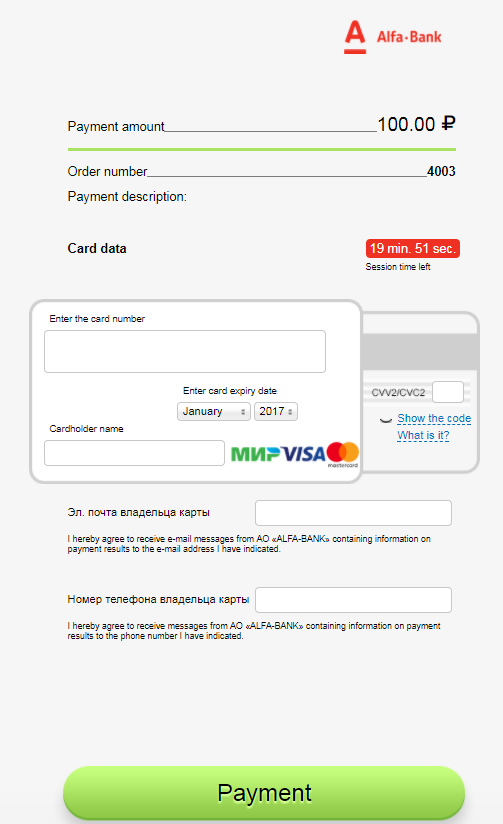
|  |
| --- |
| <div id="feeBlock" class="row" style="display: none;"> <div class="price"> <span class="size24"> <div id="feeAmount" style="float: left; margin-right: 8px;">0.00</div> <img src="images/ruble.gif" id="fee-ruble-sign" class="ruble-img" style="display: none; float: right;"/> </span> </div> <div class="name"><span>Payment fee</span></div> </div> |

Testing

1. Register an order in the payment gateway. The registration can be done with REST/SOAP.

2. Go to the payment page. If working with fees is supported for the merchant, the payment page will contain:

* Fee amount;
* "I am aware of the terms of the offer and accept them" flag. To familiarize yourself with the offer, click the corresponding link. You will be redirected to the corresponding page specified in the merchant settings.



3. To process a payment, enter the card data and tick the"I am aware of the terms of the offer and accept them" flag. Click Payment.

The fee amount will be passed in the additional parameter of the payment request, payment\_fee. In the console, the fee amount is displayed in the payment currency:

* On the Orders page, the fee amount is displayed in the Additional Parameters column;
* On the Order Details page, the fee amount is displayed in the Additional Parameters section.

Appendix 2. Specification of additional fields for air-commerce payments and hotels booking

To improve fraud prevention in e-commerce services, it is necessary to pass additional information when registering payments in the payment gateway. It is necessary to provide the details on passengers and flight parameters when purchasing airline tickets and on guests when booking hotels.

Additional information passed in air-commerce

The table containing the names of additional fields and their descriptions is given below.

|  |  |  |  |
| --- | --- | --- | --- |
| №№ | Mandatory | Field name | Field description |
| 1 | Yes | TICKET | Ticket number/booking number |
| 2 | No | LANGUAGE | Language and the home country of the person booking a ticket |
| 3 | Yes | USER\_FIRST\_NAME | Name of the customer who books a ticket |
| 4 | Yes | USER\_LAST\_NAME | Surname of the customer who books a ticket |
| 5 | No | USER\_MOBILE\_PHONE | Mobile phone number of the customer who books a ticket |
| 6 | No | USER\_HOME\_PHONE | Landline phone number of the customer who books a ticket |
| 7 | Yes | USER\_EMAIL | Email address of the customer who books a ticket |
| 8 | Yes | S{N}-BDATE | Departure date and time for segment N |
| 9 | Yes | S{N}-EDATE | Arrival date and time for segment N |
| 10 | Yes | S{N}-BLOCATIONCOUN | Departure country name for segment N specified in the Russian or English language |
| 11 | Yes | S{N}-ELOCATIONCOUN | Arrival country name for segment N specified in the Russian or English language |
| 12 | Yes | S{N}-BLOCATIONCODE | Departure airport code for segment N specified in the Russian or English language |
| 13 | Yes | S{N}-ELOCATIONCODE | Arrival airport code for segment N specified in the Russian or English language |
| 14 | Yes | S{N}-BLOCATIONCITY | Departure city name for segment N specified in the Russian or English language |
| 15 | Yes | S{N}-ELOCATIONCITY | Arrival city name for segment N specified in the Russian or English language |
| 16 | Yes | S{N}-FLIGHTNUMBER | Flight number for segment N |
| 17 | Yes | P{M}-FIRSTNAME | Passenger M name |
| 18 | Yes | P{M}-LASTNAME | Passenger M surname |

**Additional clarifications:**

* S{N} – a reference to a flight segment number. A segment here means a flight from one airport to another without stops. Parameter {N} is a number from 1 to 99. This means that a full segment name may be S1 to S99.
* P{M} – specifies the passenger number. The {M} parameter is a number from 1 to 99. This means that a full parameter name may be P1 to P99.

**Example of filling in the parameters:**

|  |  |  |
| --- | --- | --- |
| №№ | Field name | Field value |
| 1 | TICKET | 5WY8FZ |
| 2 | LANGUAGE | RU |
| 3 | USER\_FIRST\_NAME | Maxim |
| 4 | USER\_LAST\_NAME | Zhukoveckiy |
| 5 | USER\_MOBILE\_PHONE | +7 9055457319 |
| 6 | USER\_HOME\_PHONE | null |
| 7 | USER\_EMAIL | 11@11.ru |
| 8 | S{N}-BDATE | Tuesday , January 22, 2013 6:40:00 AM |
| 9 | S{N}-EDATE | Tuesday , January 22, 2013 7:25:00 AM |
| 10 | S{N}-BLOCATIONCOUN | Russian Federation |
| 11 | S{N}-ELOCATIONCOUN | Russian Federation |
| 12 | S{N}-BLOCATIONCODE | PEE |
| 13 | S{N}-ELOCATIONCODE | LED |
| 14 | S{N}-BLOCATIONCITY | Perm |
| 15 | S{N}-ELOCATIONCITY | Saint-Petersburg |
| 16 | S{N}-FLIGHTNUMBER | 712 |
| 17 | P{M}-FIRSTNAME | Maxim |
| 18 | P{M}-LASTNAME | Zhukoveckiy |

Additional information passed on booking and paying for an hotel

The table containing the names of additional fields and their descriptions is given below.

|  |  |  |  |
| --- | --- | --- | --- |
| №№ | Mandatory | Field name | Description |
| 1 | No | HLANGUAGE | Language of the nationality of the customer who books a ticket |
| 2 | Yes | HUSER\_FIRST\_NAME | Name of the customer who books a ticket |
| 3 | Yes | HUSER\_LAST\_NAME | Surname of the customer who books a ticket |
| 4 | No | HUSER\_MOBILE\_PHONE | Mobile phone number of the customer who books a ticket |
| 5 | No | HUSER\_HOME\_PHONE | Landline phone number of the customer who books a ticket |
| 6 | Yes | HUSER\_EMAIL | Email address of the customer who books a ticket |
| 7 | Yes | H{N}­-BDATE | Check in date for hotel N |
| 8 | Yes | H{N}­-EDATE | Checkout date for hotel N |
| 9 | Yes | H{N}-­LOCATIONCOUN | Name of the country of hotel N specified in the Russian or English language |
| 10 | Yes | H{N}­-SLOCATIONCOUN | Name of the city of hotel N specified in the Russian or English language |
| 11 | Yes | H{N}­-NAME | Name of hotel N specified in the Russian or English language |
| 12 | Yes | H{N}-­GCOUNT | Number of guests in hotel N. It is necessary to specify the number of adults and the number of children separated by a colon (:) |
| 13 | Yes | H{M}­-GFIRSTNAME | Name of guest M |
| 14 | Yes | H{M}-­GLASTNAME | Surname of guest M |

**Additional clarifications:**

* H{N} – specifies the number of the hotel in the booking list. The booking list implies that a customer can book several hotels and move between them over time. Parameter {N} is a number from 1 to 99. That is, the full parameter name may be H1 to H99.
* H{N}­-GCOUNT – the total number of guests that specifies the number of adults and children. For example, if a booking in an hotel is for two adults and one child, it is necessary to specify "2:1", if a booking in an hotel is for two adults without children, it is necessary to specify "2:0".  
  Checking in a child without adults is also possible (for example, language schools or internships). Т. That is, if a booking in an hotel is for two children without adults, it is necessary to specify "0:2".
* H{M} – specifies the number of a guest. Parameter {M} is a number from 1 to 99. That is, the full parameter name

may be H1 to H99.

It is possible to send the details only on one guest and the total number of guests as usually hotels require detailed information on only one person and on the total number of guests to sojourn. In this case it is possible to pass information only about the selected guest and about the total amount of guests.

**Example of filling in the parameters:**

|  |  |  |
| --- | --- | --- |
| №№ | Field name | Field value |
| 1 | HLANGUAGE | RU |
| 2 | HUSER\_FIRST\_NAME | Maxim |
| 3 | HUSER\_LAST\_NAME | Zhukoveckiy |
| 4 | HUSER\_MOBILE\_PHONE | +7 9055457319 |
| 5 | HUSER\_HOME\_PHONE | null |
| 6 | HUSER\_EMAIL | 11@11. r u |
| 7 | H{N}­-BDATE | Tuesday, Januar y 22, 2013 |
| 8 | H{N}-­EDATE | Wednesday, Januar y 23, 2013 |
| 9 | H{N}­-LOCATIONCOUN | Russian Federation |
| 10 | H{N}­-SLOCATIONCOUN | Saint-Petersburg |
| 11 | H{N}­-NAME | Akyan St­-Petersburg |
| 12 | H{N}­-GCOUNT | 2:0 |
| 13 | H{M}­-GFIRSTNAME | Maxim |
| 14 | H{M}­-GLASTNAME | Zhukoveckiy |

Appendix 3. Response codes - interpretation of actionCode (responses from the processing system)

Response code is a numeric code that designates the result of an operation performed by the user in the payment gateway system. The following codes are defined in the system.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Action code | error\_id | error\_message | Description | The messages recommended for a customer are: |
| -20010 | -20010 | Blocked by the limit | Transaction is rejected because the payment amount exceeds the limit specified by the Issuing bank | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| -9000 | -9000 | Started | State of transaction start | *On getting this actionCode, no messages are displayed to the customer.* |
| -3003 | -3003 | Unknown | Unknown | Payment error. Try again later. |
| -2102 | -2102 | Reject by a passenger name | Reject by a passenger name | Payment error. Contact the representative of the merchant for clarification of the reasons. |
| -2101 | -2101 | Reject by email | Reject by email | Payment error. Contact the representative of the merchant for clarification of the reasons. |
| -2020 | -2020 | Incorrect ECI is received | Invalid ECI. This code means that ECI received in PaRes is not valid for the IPS. The rule applies only to MasterCard (01,02) and Visa (05,06), where the values in the brackets are the valid values for IPS. | Payment error. Try again later. |
| -2019 | -2019 | Declined by iReq in PARes | PARes from the issuing bank contains iReq, which caused the payment rejection | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| -2018 | -2018 | Declined. DS connection timeout | There is no access to Directory server Visa or MasterCard or a connection error occurred after card involvement request (VeReq). This is an error of interaction between payment gate and IPS servers due to technical problems on the side of IPS servers. | Payment error. Try again later. |
| -2017 | -2017 | Rejected. PARes status is not "Y" | Rejected. PARes status is not "Y" | Payment error. Contact the representative of the merchant for clarification of the reasons. |
| -2016 | -2016 | VERes status is "U" | Issuing bank could not determine if the card is 3D-Secure. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| -2015 | -2015 | Declined by iReq in VERes | VERes from DS contains iReq, which caused the payment rejection. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| -2013 | -2013 | No attempts left | All payment attempts were used. | Payment error. Try again later. |
| -2012 | -2012 | Operation is not supported | This operation is not supported. | Payment error. Try again later. |
| -2011 | -2011 | PARes status is "U" | Issuing bank was not able to perfor the authorization of a 3D-Secure card. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| -2010 | -2010 | TDS\_XID\_MISMATCH | Mismatching of XID. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| -2008 | -2008 | INVALID\_WALLET | Incorrect wallet. | *This actionCode is not used.* |
| -2007 | 2007 | The session time out has expired | The period for entering card data has expired. The default timeout value is 20 minutes; the session timeout can be specified on registering an order; if a merchant has the "Non-standard session duration" privilege, the value specified in the merchant settings is used. | Payment error. Try again later. |
| -2006 | 2006 | 3D-Secure authorization failed | Means that the issuing bank rejected the authentication (3D-Secure authorization has not been performed). | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| -2005 | 2005 | 3D-Secure signature check failed | Means that RBS could not check issuing bank sign, i.e. PARes was readable, but the sign was wrong. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| -2003 | -2003 | BLOCKED\_BY\_PORT | Blocked by port. | *This actionCode is not used.* |
| -2002 | 2002 | Blocked by the amount | Transaction was rejected because the payment amount exceeded the configured limits.   Note: The limits mean the limits of the issuing bank, or the daily turnover of the store, or the store limits for operations with the same card, or the store limits for the same operation. | Payment error. Contact the representative of the merchant for clarification of the reasons. |
| -2001 | 2001 | Blocked by IP | Transaction is rejected because the IP-address of the customer is in the black list. | Payment error. Contact the representative of the merchant for clarification of the reasons. |
| -2000 | 2000 | BLOCKED\_BY\_PAN | Transaction is rejected because the card number is in the black list. | Payment error. Contact the representative of the merchant for clarification of the reasons. |
| -102 | -102 | The payment is declined by the payment agent | The payment is declined by the payment agent. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| -100 | -100 | No payment attempted | There were no payment attempts. | *On getting this actionCode, no messages are displayed to the customer.* |
| -1 | -1 | The processing system is unavailable | The period of waiting for a response from the processing system has expired. | Payment error. Try again later. |
| 0 | 0 | Request is processed successfully | Payment has been processed successfully. | Successful transaction. |
| 1 | 1 | Declined. The identity check is required. | The identity confirmation is necessary to successfully complete the transaction. In case of an internet transaction (the present case) confirmation is impossible, so transaction is considered to be declined. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 5 | 5 | Processing the transaction is rejected by the network. | Processing the transaction is rejected by the network. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 15 | 15 | IPS did not define the card issuer | IPS cannot identify the issuing bank. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 53 | 53 | Invalid account | The card does not exist in the processing systems. | Payment declined. Contact the merchant. |
| 100 | 100 | Card limits exceeded. | Card limits exceeded (the issuing bank has declined a transaction with the card). | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 101 | 101 | Incorrect card expiration date. | Incorrect card expiration date. | Check the card data entered. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 103 | 103 | Contact the issuing bank. | There is no connection with the Issuing bank. Sales outlet needs to contact the Issuing bank. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 104 | 104 | Card account limits reached | This is an attempt to perform a transaction for an account that has usage restrictions. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 106 | 106 | The maximum number of attempts to enter PIN is exceeded. The card might be temporarily blocked. | The maximum number of attempts to enter PIN is exceeded. The card might be temporarily blocked. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 109 | 109 | Incorrect merchant identifier or terminal identifier | Merchant or terminal identifier is incorrect or ACC is blocked at the processing system level. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 111 | 111 | Incorrect card number | Card number is incorrect. | Check your card balance and verify the card details entered. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 116 | 116 | Insufficient funds. | Transaction amount exceeds the available balance of the selected account. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 120 | 120 | Transaction was rejected by the issuing bank. | The operation is declined: the transaction is not allowed by the Issuing bank. Response code of the IPS is 57. Address the issuing bank to clarify the reasons for rejection. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 121 | 121 | Available limit exceeded. | This is an attempt to perform a transaction for an amount exceeding the day limit set by the issuing bank. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 123 | 123 | Available limit exceeded. | Available limit of the number of transactions was exceeded: the customer has executed the maximum number of transactions within the limit cycle and tries to execute another one. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 125 | 125 | Card number is incorrect. | Card number is incorrect. This error may have several meanings: an attempt to perform a refund for an amount exceeding the amount put on hold; an attempt to refund a zero amount. For AmEx this code means that an incorrect card expiration date was specified | Check your card balance and verify the card details entered. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 208 | 208 | Card is lost | Card is lost. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 209 | 209 | Card limits exceeded. | Card limitations exceeded. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 400 | 400 | Reversal is processed. | Reversal is processed. | Payment has been reversed. |
| 902 | 902 | Card limits exceeded. | Card limitations (the Cardholder is attempting to perform a transaction that is forbidden for him). | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 903 | 903 | Available limit exceeded. | An attempt is made to perform a transaction for an amount exceeding the Issuing bank limit. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 904 | 904 | Incorrect format of the message | The message format is incorrect from the point of view of the issuing bank. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 907 | 907 | No connection to the bank. | No connection with the issuing bank. Authorization in the stand-in mode is not allowed for this card number (this mode means that the Issuing bank is unable to connect to the IPS, and, therefore, the transaction can be either processed offline with the further uploading to the back office or can be declined). | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 909 | 909 | Impossible to process the operation. | It is impossible to process the operation (a common error in the system functioning has occured). It is fixed by the payment network or the issuing bank. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 910 | 910 | No connection to the bank. | Issuing bank is not available. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 913 | 913 | Incorrect format of the message. | The message format is incorrect in terms of IPS. | Payment error. Try again later. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 914 | 914 | Original transaction is not found | Transaction is not found (when sending a completion, reversal or refund request). | *When getting this actionCode, no messages are displayed to the customer*. |
| 999 | 999 | Suspicion of fraud. | The beginning of the transaction authorization is missed. Declined by fraud. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 1001 | 1001 | Empty | Empty (is specified at the moment of transaction authorization, when card details are not entered yet). | *On getting this actionCode, no messages are displayed to the customer.* |
| 1004 | 1004 | Authorization phase 1 | Authorization phase 1. | *On getting this actionCode, no messages are displayed to the customer.* |
| 1005 | 1005 | Authorization phase 2 | Authorization phase 2. | *On getting this actionCode, no messages are displayed to the customer.* |
| 2001 | 2001 | FRAUD | Fraud (in terms of IPS). | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 2002 | 2002 | Incorrect operation | Incorrect operation. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 2003 | 2003 | SSL is forbidden | SSL (not 3D-Secure/SecureCode) transactions are forbidden for the Merchant. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 2004 | 2004 | SSL without CVC is forbidden | Payment through SSL without CVC2 is forbidden. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 2005 | 2005 | Order does not comply with the 3D-Secure rule | The payment does not meet the conditions of the validation rule for 3D-Secure. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 2006 | 2006 | One-phase payments are forbidden | One-phase payments are forbidden. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 2007 | 2007 | Order has already been paid | The order is payed. | Order has already been paid. |
| 2008 | 2008 | Transaction has not been completed yet | The transaction is not completed. | *On getting this actionCode, no messages are displayed to the customer.* |
| 2009 | 2009 | Refund amount exceeds the payment amount | Refund amount exceeds deposited amount. | Refund amount exceeds deposited amount. |
| 2014 | 2014 | Error of 3DS rule execution. | Error of 3DS rule execution. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 2015 | 2015 | Error in executing the rule of terminal selection | Error in the selection of a terminal rule (the rule is incorrect). | Payment error. Contact a representative of the merchant for more details. |
| 2016 | 2016 | 3D-Secure is forbidden | 3-D Secure payment is necessary, but the merchant does not have permission for 3-D Secure payment. | Payment error. Contact a representative of the merchant for more details. |
| 2023 | 2023 | Thread limit is exceeded | The queue of requests to be processed exceeded the allowed limit. | Payment error. Try again later. |
| 4005 | 4005 | Declined by the merchant | The order was declined by the merchant. | Declined by merchant. |
| 9001 | 9001 | Internal RBS error | RBS internal error. | Payment error. Try again later. |
| 71015 | 1015 | Entered data is incorrect | Entered card details are incorrect. | Check your card balance and verify the card details entered. If this error occurs repeatedly, contact your bank for more details. You can find the telephone number of the bank on the back side of your bank card. |
| 151017 | 1017 | Decline. 3-D Secure comm error | 3-D Secure - communication error. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 151018 | 018 | Decline. Processing timeout | Processing timeout. Sending is failed. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 151019 | 1019 | Decline. Processing timeout | Processing timeout. Sending is success, response from the bank was not received. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |
| 341014 | 1014 | Decline. General Error | RBS general error. | Payment error. Try again later. If this error occurs repeatedly, contact a representative of the merchant for more details. |