Протокол взаимодействия с интернет кассой

Версия 2.7.1 от 28.06.2017

Оглавление

[Изменения в документе 3](#_Toc486261161)

[1. Общее описание 4](#_Toc486261162)

[Пример запроса с подписью 4](#_Toc486261163)

[2. Запросы 7](#_Toc486261164)

[2.1 Создания чека 7](#_Toc486261165)

[2.1.1 Тело запроса 7](#_Toc486261166)

[2.1.1.1 Содержимое документа 7](#_Toc486261167)

[2.1.1.2 Позиция чека 7](#_Toc486261168)

[2.1.1.3 Параметры закрытия чека 7](#_Toc486261169)

[2.1.1.4 Оплата 8](#_Toc486261170)

[2.1.2 Тело ответа с ошибками валидации 9](#_Toc486261171)

[Пример запроса: 9](#_Toc486261172)

[Пример ответа с ошибкой: 9](#_Toc486261173)

[2.2 Состояние чека 10](#_Toc486261174)

[2.2.1 Тело ответа 10](#_Toc486261175)

[Пример ответа: 10](#_Toc486261176)

[3. Swagger 12](#_Toc486261177)

# Изменения в документе

**Версия 0.1.0 от 13.02.2017**

Первая версия.

**Версия 0.2.0 от 17.02.2017**

Изменено содержимое чека и выдаваемые на запрос статуса документа данные.

**Версия 0.3.0 от 21.02.2017**

Изменено время выдаваемое в результате пробития чека на локальное, уточнение по цене товара, добавлен признак отправки в ОФД и данные квитанции.

**Версия 0.4.0 от 23.03.2017**

Добавлены значения параметров система налогообложения и тип оплаты, в запрос регистрации добавлен атрибут группа.

**Версия 0.5.0 от 13.04.2017**

Убраны признак отправки в ОФД и данные квитанции.

**Версия 2.0.0 от 24.05.2017**

Добавлен ИНН в запрос статуса документа. Изменена версию API на v2.

**Версия 2.1.0 от 29.05.2017**

Добавлено разъяснение по поводу группы по умолчанию.

Добавлено упоминание о swagger.

**Версия 2.2.0 от 01.06.2017**

Добавил примеры генерации подписи на php и .net core и уточнение по используемому паддингу.

**Версия 2.3.0 от 07.06.2017**

Добавлено описание 401 ошибки.

Добавлен пример генерации подписи на .net full.

**Версия 2.4.0 от 14.06.2017**

Добавлен пример генерации подписи на php.

**Версия 2.5.0 от 19.06.2017**

Формат сообщений приведен в соответствие с тем, что возвращает API.

Добавлен пример ответа с ошибкой.

**Версия 2.6.0 от 20.06.2017**

Добавлен пример создания подписи на php с использованием openssl.

**Версия 2.7.0 от 26.06.2017**

Изменен пример сообщения с подписью.

**Версия 2.7.1 от 28.06.2017**

Исправлен код ответа на ненайденный документ при проверке статуса в соответствии с поведением системы.

# 1. Общее описание

Взаимодействие с интернет-кассой осуществляется посредством обмена сообщениями с web-сервисом поднимаемым устройством.

Транспорт – https с использованием клиентского и серверного сертификатов для взаимной аутентификации.

Формат сообщений – JSON.

Для защиты клиентских сообщений от изменений используется подпись SHA256-RSA с ключом не менее 2048 Кбит, паддинг Pkcs1.

Подпись передается в http-заголовке X-Signature закодированная с использованием base64 кодировки.

Интернет-касса имеет публичную часть ключа клиента и осуществляет проверку переданной подписи.

При генерации подписи используется сообщение конвертированное в байты с использованием кодировки UTF-8.

### Пример запроса с подписью

POST /api/v2/documents HTTP/1.1

Content-Length: 363

Content-Type: application/json; charset=utf-8

X-Signature: K/YpoNDWcG5FOrq4VZgBKVK7IYvYQnqJ1YLUUe5N3dTnGVzaGV6s3HLh0HcOFrp54eauMdU2O3SPRDJJBkpzhFF/Ydf7FpfAb9Tj1XpRoYt3Q9acciTFferE8gb5niW/yq3bmYJQiSqM8hSIDhoqF4HcaWKHiXPsu/XusgjsS1RqQeC8eH96/m2e35cUcAK980Kv5dCj6ePByC+9Q4h8g1QqSpNQdG0UsayFo2o5yCWMe1fbLE0EseRUUSZ2Z79YS42H6S+psozd/WNTtCkF6p3PvabVmOMBvZaPPtij//51W1cS7bm9civYH/FgjpxVm476F4com+zhutXhe2VHlg==

{"Id": "34658763458763458","INN": "3123011520","Content": {"Type": 1,"Positions": [{"Quantity": 1.000,"Price": 123.45,"Tax": 6,"Text": "Булка"},{"Quantity": 2.000,"Price": 4.45,"Tax": 4,"Text": "Спички"}],"CheckClose": {"Payments": [{"Type": 1,"Amount": 123.45},{"Type": 2,"Amount": 8.90000}],"TaxationSystem": 1},"CustomerContact": "foo@example.com"}}

Ключ RSA:

<RSAKeyValue>

  <Modulus>t8nC/Eth8UabQbXu8pdro3v7NqUanV8Y+g92YgT7z1xqkBLRHXZ1guml3PxrqjNX9AvOmu8R+qaKOyHfJW0PcRDLzCoIUcHNAwpDO/E5j6WAaLIv7gAjTtyr9kJB9rfJaparViJNZu3RSUYGTvVznOmXMf7LTOTMR6HP/5H1TP5n1g4+BbLmC9EhjUf2eNFqwZBqPtzybBb6jaHBRaJ0XdE3lh2OeE9/OF0BtLwiYPDKsVTxIekbNf7l/DREy+YbUOxQLceeHXrvbYLiGWecP0a7CqHGj9ZNY1oJThK3AwrSd4yHa9Wnx/GaZUNtWud1BaP9g3sVX+sRV9xtnI96dw==</Modulus>

  <Exponent>AQAB</Exponent>

  <P>3WSb72a1erb6jcLkyZA2Y21VNIipGz+ta1RP+iacs3xnktFsxgTYgqWyt6SWZ2rStp0u4vb/IAHyKhgJPNTUSi2u0G44MOsRxMC/FWTF8zdyrDF4BjPBM4j84nAmE/FQYv5F8ldDkakc96zEPiTk5Fka3MPeN8mMk6/OA59JdF0=</P>

  <Q>1IRVid5SsDrOwJQAEKkdT436XEb0sVWe9AcU8JyaCEEMj0NPzownNbIrebPofMYdDHikopQpr2XqxZYDbb7AneoHkhEV26TfpPVbN4wBJFXih3lAP2n5hqhgqHGp5Wq2Lu7jUS376Ruw3bhwW+MiWpXv1xhMTZ8AtDfnZFFNvOM=</Q>

  <DP>Fo5KiNCJCtCbpFfH4XVM5UJdXPXTbNBHBdlYMJ9AddTl5IJrt50ExgLFu4oMPMsYXryS61LI2WT5XCqIvmbcnhYbambgWLOKYuZUUYSr2kS67So5FUCunWaGhTdx2bRLQVqwm6kiXDPDnMRAViiCHXWqk/VsrXheVymhLqNK440=</DP>

  <DQ>mowSWMzhfV+G8+2tjnAt7KjnpSvEzyHhEr4DsGdybQZBR/4/j4nFCfukOkFnlTXN8j/aGpF9Lx0C+uX5YFoUYcLL9qGOL8lbCu+TgnXCbtY2gybeXj+HQzI3+MeQMlLEYqU/ks3KIOAOY2+55ljrpszbOqVk+B3luSnekMm/qtk=</DQ>

  <InverseQ>aP5e5F1j6s82Pm7dCpH3mRZWnfZIKqoNQIq2BO8vA9/WrdFI2C27uNhxCp2ZDMulRdBZcoeHcwJjnyDzg4I4gBZ2nSKkVdlN1REoTjLBBdlHi8XKiXzxvpItc2wjNC2AKHaJqj/dnh3bbTAQD1iUAxPmmLJYYkhfZ2i1IrTVxZE=</InverseQ>

  <D>PUfM+Aq6kZSVWAetsL3EajKAxOuwQCDhVx+ovW4j+DQ8Y+WiTEyfShNV9qVD0PBltz3omch1GjpFhQn6OaRvraeIDH9HXttb3FOjr2zzYG4yrrYbPSRWoYj63ZWiIP2O7zdl0caGQHezfNcYa2N0NTG99DGc3/q6EnhlvjWQsSbiEjmxcPx8fmV1i4DoflMQ383nsixAFapgrROUAtCgMvhWn1kSeoojKd+e4eKZxa/SNYulsBJWNFkmo1CZH4YTqlPM+IwYeDUOnOUGNxGurRZ3qQdWs2N2ZQhnrvlh+zpzurD2hwAz6gQXP7mxxMR1xHtAD8XQ+w4OiJK6VWjoIQ==</D>

</RSAKeyValue>

Пример создания подписи в php с использованием https://github.com/phpseclib/phpseclib

function computeSignature($data)

{

    $rsa = new RSA();

    $rsa->setPrivateKey(($this->privateKey);

    $rsa->setPrivateKeyFormat(RSA::PRIVATE\_FORMAT\_XML);

    $rsa->setHash('sha256');

    $rsa->setMGFHash('sha256');

    $rsa->setSignatureMode(RSA::SIGNATURE\_PKCS1);

    return base64\_encode($rsa->sign($data));

}

Пример создания подписи на php с использованием openssl. Для данного способа нужен ключ в PEM формате, сконвертировать xml в PEM можно, например, онлайн-конвертером <https://superdry.apphb.com/tools/online-rsa-key-converter>:

function computeSignature($data)

{

    $key = <<<KEYDATA

    -----BEGIN RSA PRIVATE KEY-----

    MIIEvQIBADANBgkqhkiG9w0BAQEFAASCBKcwggSjAgEAAoIBAQC3ycL8S2HxRptB

    te7yl2uje/s2pRqdXxj6D3ZiBPvPXGqQEtEddnWC6aXc/GuqM1f0C86a7xH6poo7

    Id8lbQ9xEMvMKghRwc0DCkM78TmPpYBosi/uACNO3Kv2QkH2t8lqlqtWIk1m7dFJ

    RgZO9XOc6Zcx/stM5MxHoc//kfVM/mfWDj4FsuYL0SGNR/Z40WrBkGo+3PJsFvqN

    ocFFonRd0TeWHY54T384XQG0vCJg8MqxVPEh6Rs1/uX8NETL5htQ7FAtx54deu9t

    guIZZ5w/RrsKocaP1k1jWglOErcDCtJ3jIdr1afH8ZplQ21a53UFo/2DexVf6xFX

    3G2cj3p3AgMBAAECggEAPUfM+Aq6kZSVWAetsL3EajKAxOuwQCDhVx+ovW4j+DQ8

    Y+WiTEyfShNV9qVD0PBltz3omch1GjpFhQn6OaRvraeIDH9HXttb3FOjr2zzYG4y

    rrYbPSRWoYj63ZWiIP2O7zdl0caGQHezfNcYa2N0NTG99DGc3/q6EnhlvjWQsSbi

    EjmxcPx8fmV1i4DoflMQ383nsixAFapgrROUAtCgMvhWn1kSeoojKd+e4eKZxa/S

    NYulsBJWNFkmo1CZH4YTqlPM+IwYeDUOnOUGNxGurRZ3qQdWs2N2ZQhnrvlh+zpz

    urD2hwAz6gQXP7mxxMR1xHtAD8XQ+w4OiJK6VWjoIQKBgQDdZJvvZrV6tvqNwuTJ

    kDZjbVU0iKkbP61rVE/6JpyzfGeS0WzGBNiCpbK3pJZnatK2nS7i9v8gAfIqGAk8

    1NRKLa7Qbjgw6xHEwL8VZMXzN3KsMXgGM8EziPzicCYT8VBi/kXyV0ORqRz3rMQ+

    JOTkWRrcw943yYyTr84Dn0l0XQKBgQDUhFWJ3lKwOs7AlAAQqR1PjfpcRvSxVZ70

    BxTwnJoIQQyPQ0/OjCc1sit5s+h8xh0MeKSilCmvZerFlgNtvsCd6geSERXbpN+k

    9Vs3jAEkVeKHeUA/afmGqGCocanlarYu7uNRLfvpG7DduHBb4yJale/XGExNnwC0

    N+dkUU284wKBgBaOSojQiQrQm6RXx+F1TOVCXVz102zQRwXZWDCfQHXU5eSCa7ed

    BMYCxbuKDDzLGF68kutSyNlk+VwqiL5m3J4WG2pm4FizimLmVFGEq9pEuu0qORVA

    rp1mhoU3cdm0S0FasJupIlwzw5zEQFYogh11qpP1bK14XlcpoS6jSuONAoGBAJqM

    EljM4X1fhvPtrY5wLeyo56UrxM8h4RK+A7Bncm0GQUf+P4+JxQn7pDpBZ5U1zfI/

    2hqRfS8dAvrl+WBaFGHCy/ahji/JWwrvk4J1wm7WNoMm3l4/h0MyN/jHkDJSxGKl

    P5LNyiDgDmNvueZY66bM2zqlZPgd5bkp3pDJv6rZAoGAaP5e5F1j6s82Pm7dCpH3

    mRZWnfZIKqoNQIq2BO8vA9/WrdFI2C27uNhxCp2ZDMulRdBZcoeHcwJjnyDzg4I4

    gBZ2nSKkVdlN1REoTjLBBdlHi8XKiXzxvpItc2wjNC2AKHaJqj/dnh3bbTAQD1iU

    AxPmmLJYYkhfZ2i1IrTVxZE =

    -----END RSA PRIVATE KEY-----

    KEYDATA;

    // sha256 + Pkcs1 https://tools.ietf.org/html/rfc3447#page-43

    $data = pack('H\*', '3031300d060960864801650304020105000420') . hash('sha256', $data, true);

    $pk  = openssl\_get\_privatekey($key);

    openssl\_private\_encrypt($data, $res, $pk);

    return base64\_encode($res);

}

Пример отправки данных на php:

$sign = $this->computeSignature(json\_encode($data, JSON\_UNESCAPED\_UNICODE));

$headers = [

    'Accept: application/json',

    'Content-Type: application/json',

    'X-Signature: '.$sign

];

$ch = new Curl();

$ch->init();

$ch->setOptions($x=[

    CURLOPT\_HEADER => false,

    CURLOPT\_SSL\_VERIFYPEER => false,

    CURLOPT\_SSLCERT => $this->client\_crt(),

    CURLOPT\_SSLKEY => $this->client\_key(),

    CURLOPT\_SSLCERTPASSWD => '1234',

    CURLOPT\_HTTPHEADER => $headers,

]);

$result = $ch->post(

    'https://apip.orangedata.ru:2443/api/v2/documents/',

    json\_encode($data, JSON\_UNESCAPED\_UNICODE));

Пример создания подписи на C# с использованием стандартной библиотеки классов .net core:

private string ComputeSignature(string document)

{

    var data = Encoding.UTF8.GetBytes(document);

    using (var rsa = RSA.Create())

    {

        rsa.FromXmlString(privateKey);

        return Convert.ToBase64String(rsa.SignData(data, HashAlgorithmName.SHA256, RSASignaturePadding.Pkcs1));

    }

}

Пример создания подписи на C# с использованием стандартной библиотеки классов .net framework full:

private string ComputeSignature(string document)

{

    var data = Encoding.UTF8.GetBytes(document);

    using (var rsa = new RSACryptoServiceProvider())

    {

        rsa.FromXmlString(privateKey);

        return Convert.ToBase64String(rsa.SignData(data, "SHA256"));

    }

}

# 2. Запросы

## 2.1 Создания чека

Создание чека – асинхронный запрос, после его выполнения чек оказывается в очереди на обработку. Помимо данных чека клиент передает уникальный идентификатор документа использую который в последствии, он сможет запросить состояние чека(п.2.2). Плюс, данный идентификатор обеспечивает идемпотентность запросов. Данный идентификатор должен быть уникальным в пределах организации.

ИНН служит для определения устройства с помощью которого может быть проведен чек.

Group в запросе может быть null или не указан, в таком случае чек отправится в группу по умолчанию.

Группой по умолчанию считается группа с названием Main.

Запрос: **POST** **/api/v2/documents/**

### 2.1.1 Тело запроса

|  |  |  |
| --- | --- | --- |
| id | Идентификатор документа | Строка от 1 до 32 символов |
| Inn | ИНН организации, для которой пробивается чек | Строка 12 символов |
| group | Группа устройств, с помощью которых будет пробит чек | Строка от 1 до 32 символов или null |
| content | Содержимое документа | Структура п.2.1.1.1 |

### 2.1.1.1 Содержимое документа

|  |  |  |
| --- | --- | --- |
| type | Тип документа:   1. Приход 2. Возврат прихода 3. Расход 4. Возврат расхода | Число |
| positions | Список позиций в чеке | Массив структур п.2.1.1.2 |
| checkClose | Параметры закрытия чека | Структура п.2.1.1.3 |
| customerContact | Телефон или e-mail покупателя | Строка до 64 символов, формат +{Ц} или {С}@{C} |

### 2.1.1.2 Позиция чека

|  |  |  |
| --- | --- | --- |
| quantity | Количество товара | Десятичное число с точностью до 3 символов после точки |
| price | Цена товара с учетом всех скидок и наценок | Десятичное число с точностью до 2 символов после точки |
| tax | Ставка НДС:  1 – ставка НДС 18%  2 – ставка НДС 10%  3 – ставка НДС расч. 18/118  4 – ставка НДС расч. 10/110  5 – ставка НДС 0%  6 – НДС не облагается | Число от 1 до 6 |
| text | Текст позиции | Строка до 128 символов |

### 2.1.1.3 Параметры закрытия чека

|  |  |  |
| --- | --- | --- |
| payments | Оплаты | Массив структур п.2.1.1.4 |
| taxationSystem | Система налогообложения:  0 – Общая, ОСН  1 – Упрощенная доход, УСН доход  2 – Упрощенная доход минус расход, УСН доход - расход  3 – Единый налог на вмененный доход, ЕНВД  4 – Единый сельскохозяйственный налог, ЕСН  5 – Патентная система налогообложения, Патент | Число от 0 до 5 |

### 2.1.1.4 Оплата

|  |  |  |
| --- | --- | --- |
| type | Тип оплаты:  1 – Наличными  2 – Картой Мир  3 – Картой Visa  4 – Картой MasterCard  5 – Расширенная оплата 1  6 – Расширенная оплата 2  7 – Расширенная оплата 3  8 – Расширенная оплата 4  9 – Расширенная оплата 5  10 – Расширенная оплата 6  11 – Расширенная оплата 7  12 – Расширенная оплата 8  13 – Расширенная оплата 9  14 – Предвариательная оплата(Аванс)  15 – Последующая оплата(Кредит)  16 – Иная форма оплаты | Число от 1 до 16 |
| amount | Сумма оплаты | Десятичное число с точностью до 2 символов после точки |

Ответ: api может возвращать следующие статус-коды

* 201 Created – чек создан и добавлен в очередь на обработку, пустое тело ответа
* 401 Unauthorized – клиентский сертификат не прошел проверку
* 409 Conflict– чек с данным идентификатором уже был создан в системе, пустое тело ответа
* 400 Bad Request – переданные данные содержат ошибки, тело ответа п.2.1.2

### 2.1.2 Тело ответа с ошибками валидации

|  |  |  |
| --- | --- | --- |
| errors | Массив ошибок валидации запроса | Массив строк |

### Пример запроса:

{

  "id": "12345678990",

  "inn": "123456789012",

  "group": "Main",

  "content": {

    "type": 1,

    "positions": [

      {

        "quantity": 1.000,

        "price": 123.45,

        "tax": 6,

        "text": "Булка"

      },

      {

        "quantity": 2.000,

        "price": 4.45,

        "tax": 4,

        "text": "Спички"

      }

    ],

    "checkClose": {

      "payments": [

        {

          "type": 1,

          "amount": 123.45

        },

        {

          "type": 2,

          "amount": 8.90000

        }

      ],

      "taxationSystem": 1

    },

    "customerContact": "foo@example.com"

  }

}

### Пример ответа с ошибкой:

{

  "errors": [

    "Не указан идентификатор документа 'Id'",

    "Не указан ИНН организации 'INN'",

    "Отсутствует содержимое документа 'Content'"

  ]

}

## 2.2 Состояние чека

Запрос: **GET** **/api/v2/documents/****{inn}/status****/{document\_id}**

**{inn}** – ИНН организации, для которой пробивается чек

**{document\_id}** –идентификатор документа, который был указан при его создании

В данном запросе подпись SHA256-RSA не используется.

Ответ: api может возвращать следующие статус-коды

* 202 Accepted – чек создан и добавлен в очередь на обработку, но еще не обработан, пустое тело ответа
* 400 Not Found – организация не найдена, чек с указанным идентификатором не найден
* 401 Unauthorized – клиентский сертификат не прошел проверку
* 200 OK – чек обработан, тело ответа п.2.2.1

### 2.2.1 Тело ответа

|  |  |  |
| --- | --- | --- |
| id | Идентификатор документа | Строка от 1 до 32 символов |
| deviceSN | Заводской номер устройства пробившего чек | Строка до 20 символов |
| deviceRN | Регистрационный номер устройства пробившего чек | Строка до 20 символов |
| fsNumber | Номер фискального накопителя | Строка 16 символов |
| ofdName | Наименование ОФД | Строка до 256 символов |
| odfWebsite | Web-сайт ОФД | Строка до 58? символов |
| odfINN | ИНН ОФД | Строка 12 символов |
| fnsWebsite | Web-сайт ФНС | Строка до 256 символов |
| companyINN | ИНН пользователя | Строка 12 символов |
| companyName | Наименование пользователя | Строка до 256 символов |
| documentNumber | Номер ФД | Число |
| shiftNumber | Номер смены | Число |
| documentIndex | Номер чека за смену | Число |
| processedAt | Время регистрации фискального документа в ФН | Время в виде строки в формате ISO8601 |
| content | Содержимое документа | Структура п.2.1.1.1 |
| change | Сдача | Десятичное число с точностью до 2 символов после точки |
| fp | Фискальный признак | Строка 10 символов |

### Пример ответа:

{

  "id": "12345678990",

  "deviceSN": "0000000000001358",

  "deviceRN": "0000000400054952",

  "fsNumber": "9999078900001341",

  "ofdName": "ООО \"Ярус\"(\"ОФД-Я\")",

  "ofdWebsite": "www.ofd-ya.ru",

  "ofdinn": "7728699517",

  "fnsWebsite": "www.nalog.ru",

  "companyINN": "123456789012",

  "companyName": "ЗАО ТОРГОВЫЙ ОБЪЕКТ №1",

  "documentNumber": 117,

  "shiftNumber": 20,

  "documentIndex": 5,

  "processedAt": "2017-02-14T14:16:00",

  "content": {

    "type": 1,

    "positions": [

      {

        "quantity": 1.000,

        "price": 123.45,

        "tax": 6,

        "text": "Булка"

      },

      {

        "quantity": 2.000,

        "price": 4.45,

        "tax": 4,

        "text": "Спички"

      }

    ],

    "checkClose": {

      "payments": [

        {

          "type": 1,

          "amount": 123.45

        },

        {

          "type": 2,

          "amount": 8.90000

        }

      ],

      "taxationSystem": 1

    },

    "customerContact": "+79123456789"

  },

  "change": 974.01,

  "fp": "2364009522"

}

# 3. Swagger

Описание api доступно в виде OpenAPI Specification(OAS).

По url **/swagger** доступен swagger-ui

По url **/swagger/v2/swagger.json** – спецификация.