НАЧАЛО РАБОТЫ С XML ПРОТОКОЛОМ

вер. 3.27

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

вер. 2.0

МОСКВА
8-495-783-5959

РОССИЯ
8-800-200-0059

ФАКС
8-495-926-4619

WEB
WWW.QIWI.RU

СОДЕРЖАНИЕ

1. ГЛОССАРИЙ 3
2. ВВЕДЕНИЕ 4
3. СТРУКТУРА ПРОТОКОЛА 5
   - 3.1. Формат XML-запроса 6
   - 3.2. Формат XML-ответа 7
4. МЕТОДЫ АВТОРИЗАЦИИ 10
   - 4.1. Авторизация в запросе 10
   - 4.2. Авторизация по цифровой подписи 11
     - 4.2.1. Подготовительные действия 11
     - 4.2.2. Отправка запросов с цифровой подписью 12
5. РОЛИ ПЕРСОН 15
6. ПРИМЕРЫ ФОРМИРОВАНИЯ ЗАПРОСОВ 16
   - 6.1. Запрос без параметров 16
   - 6.2. Запрос с обязательными параметрами 17
7. РЕЖИМЫ ОБРАБОТКИ ДЕЙСТВИЙ 18
   - 7.1. Работа в синхронном режиме 18
   - 7.2. Работа в асинхронном режиме 18
8. НАСТРОЙКИ ДЛЯ ТЕРМИНАЛОВ 21
   - 8.1. Получение интерфейса терминала 21
     - 8.1.1. Структура интерфейса 22
     - 8.1.2. Параметры отображения страниц интерфейса провайдера 23
     - 8.1.3. Дополнительные параметры интерфейса провайдера 25
   - 8.2. Расчет терминальной комиссии 26
9. ПРОВЕДЕНИЕ ПЛАТЕЖЕЙ 28
   - 9.1. Статусы платежей 28
   - 9.2. Сценарии проведения платежей 29
     - 9.2.1. Оффлайн проведение платежей 29
     - 9.2.2. Онлайн проведение платежей 32
   - 9.3. Тестирование запросов платежа 35
10. ЧАСТО ЗАДАВАЕМЫЕ ВОПРОСЫ 36

СПИСОК РИСУНКОВ 38

СПИСОК ТАБЛИЦ 38

СПИСОК ПРИМЕРОВ 38

1. ГЛОССАРИЙ

Табл. 1. Глоссарий

Термин	Определение
Агент	Лицо, осуществляющее предпринимательскую деятельность, заявившее о присоединении к Правилам и подписавшее Договор о приеме Платежей, при условии, что данное лицо принимает условия Правил в целом, в соответствии со ст.428 Гражданского кодекса РФ
Персона	Сотрудник агента: менеджер, кассир, бухгалтер и т.п. – человек, который непосредственно производит разрешенные ему действия (в соответствии с назначенной ему ролью) в *Системе* от имени агента
Право	Разрешение на выполнение XML-запроса. Права группируются в роли
Роль	Набор прав на выполнение действий с системой. Каждому пользователю системы назначена роль
Поставщик услуг (Провайдер)	Коммерческая организация или индивидуальный предприниматель, предоставляющие потребителям услуги, продающие товары от собственного имени, а также благотворительная организация, созданная для осуществления благотворительной деятельности
Система	Автоматизированная система процессинга ЗАО «ОСМП», предназначенная для обеспечения дистанционной оплаты потребителем услуг поставщика
Терминал	Программно-технические комплексы различных типов, в том числе POS-терминалы (специализированные прикассовые устройства типа Point Of Sale), стационарное оборудование, мобильные устройства карманного типа, переносные терминалы и кассовые аппараты, автоматы самообслуживания, а также расчетный веб-сервер для авторизации транзакций в сети Интернет (платежный шлюз)
Протокол	Набор правил обмена данными между ПО терминала и процессингом ОСМП для приема платежей, отправки интерфейса, формирования отчетности и др.

2. ВВЕДЕНИЕ

Данный документ содержит описание структуры протокола взаимодействия с терминалами и запросов, методов авторизации в Системе, получения конфигурации терминалов и проведения платежей, а также часто задаваемые вопросы по отправке запросов и обработке ответов.

Предполагается, что технические процедуры подключения к Системе уже выполнены (см. документ **Новому участнику XML**).

3. СТРУКТУРА ПРОТОКОЛА

Взаимодействие клиентского ПО (ПО терминала) и процессинга происходит по следующему алгоритму:

1. Клиентское ПО формирует запрос в формате XML, включая в него необходимые действия, и передает его по протоколу HTTP на сервер.
2. Сервер осуществляет авторизацию и разбор XML.
3. В случае успешной авторизации сервер выполняет действия (или добавляет действия в очередь, в случае асинхронного режима обработки).
4. Сервер возвращает клиенту ответ в формате XML, в том числе результат выполнения запроса (код ошибки).

Отправляемый терминалом запрос состоит из двух частей:

Пример 1. Пример запроса

```
POST /XMLgate/XML.jsp HTTP/1.0
Connection: keep-alive
Content-Type: text/XML
Content-Length: 249
Host: xml1.qiwi.com
Accept: text/html, */*
Accept-Encoding: identity
User-Agent: Dealer v0

<?XML version="1.0" encoding="utf-8"?>
<request>
…
</request>
```

- HTTP-заголовок.

  Передача данных между клиентским и серверным ПО построена на основе протокола **HTTP** (RFC 2616: HTTP/1.1). Для передачи данных используется метод **RAW POST** – поток данных отправляется в теле запроса от клиента на сервер. Данные передаются в формате XML *(Extensible Markup Language (XML) 1.0).*

  ПРИМЕЧАНИЕ

  Так как документ посвящен началу работы с XML-протоколом, здесь не рассматриваются вопросы составления http-заголовка запроса (подробнее см. Протокол взаимодействия терминального ПО и процессинга).

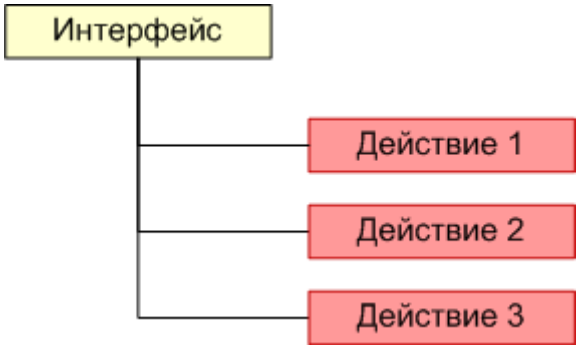
- XML-запрос.

  Реализация тела XML-протокола основана на интерфейсах и действиях.

  – **Интерфейсы (**`interface`**)** – группы возможных действий (аналогичны классам объектов в ООП). Каждый интерфейс объединяет несколько действий (см. Рис. 1);

  – **Действия (**`action`**)** – действия, выполняемые в *Системе* (аналогичны методам объектов в ООП).

Рис. 1. Структура интерфейса



3.1. Формат XML-запроса

В общем виде запрос выглядит следующим образом:

```
<?XML version="1.0" encoding="windows-1251"?>
<request>
  <auth login="login" sign="password-sign" signAlg="MD5"/>
  <client terminal="123" software="Dealer v0" serial="123456" timezone="GMT+23"
   language="en"/>
  <interface1>
    <action1>
     Параметры действия
    </action1>
    <action2>
     Параметры действия
    </action2>
  </interface1>
  <interface2>
    <action3>
     Параметры действия
    </action3>
  </interface2>
</request>
```

где:

- `request` – корневой тег запроса:
  - `auth` – тег с авторизационными данными персоны, отправляющей запрос:

ПРИМЕЧАНИЕ

Данный тег необходимо включать в состав запроса только в том случае, если используется авторизация по данным в теле запроса (см. раздел 4).

  - `login` – логин пользователя;
  - `sign` – подпись, сформированная по алгоритму MD5 на основании пароля;

- `signAlg` – алгоритм авторизации. Для авторизации по данным в теле запроса используется алгоритм MD5;

ПРИМЕЧАНИЕ

Персона, используемая для авторизации, должна иметь разрешение на выполнение данного XML-запроса (см. раздел Роли персон).

- `client` – тег содержит информацию о терминале, с которого отправлен запрос:
  - `terminal` – идентификатор терминала;
  - `software` – версия ПО, установленного на терминале;
  - `serial` – серийный номер терминала;
  - `timezone` – используется для получения результатов во временной зоне, отличной от установленной на текущем терминале;
- `interface1`, `interface2` – теги интерфейсов. Содержат теги действий:
  - `action1`, `action2`, `action3` – теги действий. Содержат параметры (атрибуты и вложенные теги) действий.

3.2. Формат XML-ответа

В общем виде ответ сервера выглядит следующим образом:

```
<?XML version="1.0" encoding="windows-1251"?>
<response result="0">
  <interface1>
    <action1 result="0" result-description="описание ошибки">
     Параметры действия
    </action1>
    <action2 result="0" result-description="описание ошибки">
     Параметры действия
    </action2>
  </interface1>
  <interface2>
    <action3 result="0" result-description="описание ошибки">
     Параметры действия
    </action3>
  </interface2>
</response>
```

где:

- `response` – корневой тег ответа:
  - `result` – атрибут отражает успешность/неуспешность выполнения запроса в целом и содержит код ошибки.

ПРИМЕЧАНИЕ

Если ошибок обработки не выявлено, атрибут result возвращает значение "0".

Если запрос не прошел первичную проверку, ответ от сервера содержит ненулевой код ошибки, а в теге `response` отображается текстовое описание ошибки. При этом будут отсутствовать теги интерфейсов и действий.

Наиболее часто встречающиеся ошибки первичной проверки:

– `<response result="202">` – неверно составлен xml запрос.

Исправьте причину и повторно отправьте запрос.

Возможные причины возникновения ошибки:

- ❖ отсутствие логина и/или пароля в теге авторизации;
- ❖ отсутствие открывающего и/или закрывающего тега;
- ❖ отсутствуют обязательные параметры;

– `<response result="150">` – указанный пароль не соответствует введенному логину.

Уточните правильность вводимых пароля и логина в теге авторизации.

– `<response result="295">` – неверно указан адрес сервера, на который отправлен запрос для обработки.

Проверьте корректность введенного адреса сервера-обработчика (список серверов см. здесь).

– `<response result="13">` – процессинг загружен.

Повторите отправку платежа с теми же реквизитами через минуту.

- `interface1` – тег интерфейса. Содержит теги действий:

– `action1` – тег действия. Содержит параметры действия:

- ⌖ `result` – атрибут возвращает код результата выполнения действия (код ошибки). Полный перечень кодов ошибок см. в Протоколе взаимодействия терминального ПО и процессинга.

ПРИМЕЧАНИЕ

Если ошибок при выполнении действия не возникло, атрибут result возвращает значение "0", а в ответе будет отсутствовать атрибут result-description.

ПРИМЕЧАНИЕ

Коды ошибок подразделяются на фатальные и нефатальные:

- Фатальная ошибка – отрицательный ответ на клиентский запрос. Все ошибки первичной обработки являются фатальными.
- Нефатальная ошибка – промежуточный статус обработки клиентского запроса, дальнейшие действия зависят от конкретной ошибки.

Наиболее часто встречающиеся ошибки:

❖ `<action1 result="5"…>` – номер не принадлежит оператору.

Провайдер вернул ошибку при проверке реквизитов платежа.

Возможные причины возникновения ошибки:

➢ номер счета/телефона отсутствует в базе провайдера;

- неверно введен номер;

❖ `<action1 result="133"…>` – нет прав на прием платежей.

Роль персоны, использованная для авторизации, не имеет прав на выполнение данного XML запроса.

❖ `<action1 result="202"…>` – ошибка данных запроса.

Проверьте корректность параметров данного действия.

❖ `<action1 result="212"…>` – не задана сумма платежа.

Укажите сумму платежа или проверьте, не превышен ли лимит платежа.

❖ `<action1 result="245"…>` – в тегах авторизации неверно указан тип терминала.

Тег `client` должен содержать атрибут `software="Dealer v0"`.

❖ `<action1 result="295"…>` – ошибка в названии действия или интерфейса.

Проверьте наименование интерфейса / действия;

⌖ `result-description` – атрибут содержит краткое описание произошедшей ошибки.

4. МЕТОДЫ АВТОРИЗАЦИИ

Протокол поддерживает два метода авторизации:

- по данным в теле запроса (менее приоритетный метод, но более простой и, как показывает практика, наиболее часто используемый);
- по цифровой подписи пакетов данных (более приоритетный метод, но сложнее в реализации).

4.1. Авторизация в запросе

Для **авторизации по данным в теле запроса** необходимо иметь:

- логин персоны (задается при создании персоны);
- зашифрованный по алгоритму MD5 пароль персоны;
- идентификатор терминала (формируется автоматически для каждого нового терминала);
- серийный номер (если он был указан в настройках терминала).

Пример 2. Пример авторизации по данным в теле запроса

```
POST /XMLgate/XML.jsp HTTP/1.0
Connection: keep-alive
Content-Type: text/XML
Content-Length: 249
Host: xml1.qiwi.com
Accept: text/html, */*
Accept-Encoding: identity
User-Agent: Dealer v0

<?XML version="1.0" encoding="utf-8"?>
<request>
  <client serial="серийный номер" software="Dealer v0" terminal="номер терминала"/>
  <auth login="логин персоны" signAlg="MD5" sign="md5 хеш от пароля персоны"/>
  <agents>
    <getBalance/>
  </agents>
</request>
```

где:

- `client` – тег, содержащий информацию о терминале, с которого отправляются XML-запросы. Атрибуты тега:
  - `serial` – серийный номер терминала, указанный в настройках терминала;
  - `software` – название типа терминала и версии установленного на нем ПО (для XML-агентов всегда `"Dealer v0"`);
  - `terminal` – идентификатор терминала;
- `auth` – тег, содержащий авторизационные данные персоны. Атрибуты:
  - `login` – логин персоны, зарегистрированной в *Системе* (задается при создании персоны);
  - `signAlg` – алгоритм шифрования авторизационных данных (всегда MD5);
  - `sign` – подпись, сформированная по алгоритму шифрования MD5 на основании неодноразового пароля.

4.2. Авторизация по цифровой подписи

Для авторизации по цифровой подписи (ЦП) требуется выполнить следующие шаги:

1. Подготовка данных (выполняется один раз).
1. Формирование подписи XML-запроса (выполняется для каждого XML-запроса).

4.2.1. Подготовительные действия

Данные действия выполняются однократно для каждой персоны, которую предполагается использовать при авторизации:

1. Формирование пары RSA ключей с помощью клиентского ПО (например, с помощью утилит PuTTYgen или OpenSSL).

Пример 3. Генерация RSA ключей.

В данном примере для генерации пары RSA ключей используется утилита OpenSSL (подробную информацию по установке и использованию данного приложения можно получить на сайте www.openssl.org).

1. Создание закрытого ключа. Выполните команду:

```
openssl genrsa -out private.key 1024
```

Команда выдает закрытый ключ (1024 бит) в файл private.key.

2. Создание открытого ключа. Выполните команду:

```
openssl rsa -in private.key -pubout -out public.key
```

Команда выдает открытый ключ в файл public.key.

2. Закрытый ключ сохраняется в файле в локальном хранилище (например, в файловом хранилище или на eToken).
3. Открытый ключ в кодировке **Base64** сохраняется на сервере запросом `setPublicKey` интерфейса `persons`. При этом в параметре запроса `store-type` необходимо указать значение `4`.

ВНИМАНИЕ

Для доступа к серверу используется персона с одноразовым паролем и авторизация в теле запроса.

Пример 4. Сохранение открытого ключа.

В данном примере для шифрования ключа используется утилита OpenSSL (подробную информацию по установке и использованию данного приложения можно получить на сайте www.openssl.org).

Например, сформируем ключ от имени персоны с логином "person_login" и одноразовым паролем "person_password":

1. Шифруем одноразовый пароль по алгоритму MD5. Для этого поместите пароль в файл pass.txt и выполните команду:

```
openssl dgst -md5 pass.txt -out passmd5.txt
```

Команда выдает зашифрованный пароль в файл passmd5.txt. Получим строку `e8cfdbdfc718b22489329048555fb320`.

2. Шифруем открытый ключ по алгоритму base64. Поместите ключ в файл public.key и выполните команду:

```
openssl base64 -in public.key -out publicode.key
```

Команда выдает зашифрованный пароль в файл publicode.key. Получаем строку вида:

```
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDH7Sxozbq5Gkrsr5qIpQEnOXXw9W9JS+hqaQm4K6dou
Y+IDv49iqqTV96U5D7V4Alg7TACpGCs/2XSE6pyGpc50VEYk3jGDFHQcI3lS6kEQLoZ/gDeHVreCRgRJ2
ssoAS1B09rS9vpYWNQKgxFvc0RwIf4aAvW5mKoofJ84o1Z4QIDAQAB
```

3. Добавляем полученные значения в запрос:

```
<?xml version="1.0" encoding="windows-1251"?>
<request>
  <auth login="person_login" sign="e8cfdbdfc718b22489329048555fb320"
    signAlg="MD5"/>
  <persons>
    <setPublicKey>
      <store-type>4</store-type>
      <pubkey>MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDH7Sxozbq5Gkrsr5qIp
      QEnOXXw9W9JS+hqaQm4K6douY+IDv49iqqTV96U5D7V4Alg7TACpGCs/2X
      SE6pyGpc50VEYk3jGDFHQcI3lS6kEQLoZ/gDeHVreCRgRJ2ssoAS1B09rS9
      vpYWNQKgxFvc0RwIf4aAvW5mKoofJ84o1Z4QIDAQAB</pubkey>
    </setPublicKey>
  </persons>
</request>
```

Ответ приходит в следующем виде:

```
<?xml version="1.0" encoding="windows-1251"?>
<response result="0">
  <persons>
    <setPublicKey result="0"/>
  </persons>
</response>
```

где result=0 говорит о том, что запрос выполнен успешно, и вы можете использовать цифровую подпись для авторизации персоны на сервере.

ПРИМЕЧАНИЕ

После получения ответа от сервера об успешно выполненном запросе необходимо подождать не менее 20 минут до начала использования цифровой подписи при отправке запросов (для синхронизации данных).

4.2.2. Отправка запросов с цифровой подписью

Данные действия выполняются для каждого XML-запроса:

1. Так как при авторизации по ЦП тег `auth` в XML-запросе не используется, то необходимо подготовить XML-запрос следующего вида:

```
<?xml version="1.0" encoding="windows-1251"?>
<request>
  <client terminal="идентификатор терминала" software="Dealer v0"/>
  <interface>
    <action/>
  </interface>
</request>
```

где:

- `client` – тег, содержащий информацию о терминале, с которого отправляются XML-запросы:
  - `terminal` – идентификатор терминала;
  - `software` – название типа терминала и версии установленного на нем ПО (всегда `"Dealer v0"`);

2. Вычисление хэш-функции от текста XML-запроса с использованием выбранного алгоритма MD5 / SHA1, подписывание его предварительно созданным (см. раздел 4.2.1) закрытым ключом и кодирование по схеме **Base64** (см. Пример 5).

Пример 5. Формирование цифровой подписи XML-запроса.

В данном примере для шифрования ключа используется утилита OpenSSL (подробную информацию по установке и использованию данного приложения можно получить на сайте www.openssl.org).

1. Формируем XML-запрос:

```
<?xml version="1.0" encoding="windows-1251"?>
<request>
  <client terminal="1234567" software="Dealer v0"/>
    <agents>
      <getBalance/>
    </agents>
</request>
```

2. Поместите текст запроса в файле request.txt и выполните команду (предполагается, что закрытый ключ хранится в файле private.key):

```
openssl dgst -sha1 -out request.sign -sign private.key request.txt
```

где для шифрования используется алгоритм SHA1. В результате выполнения команды получаем ЦП пакета в файле request.sign.

3. Полученную ЦП пакета шифруем по алгоритму Base64 командой:

```
openssl base64 -in request.sign -out request.enc
```

В результате зашифрованная ЦП пакета будет записана в файл request.enc вида:

```
XTC5UVZ3mjXA+uZq5rb76iNK7x41A2S8expSPsYVsDlNlbSheG0ltsbuXZJCcPTUr7RCIFZmDbKv
IGD57f8ipPUrtjObMnd02hSQVpO6J0/+V2UdE/aLBKg7e3pyTIBE9gUOW/W8ySEqV3Mg7gd700Pp
WmHv7Uu7jpKcs5a8Jmc=.
```

3. Добавление в заголовок HTTP запроса следующих параметров, необходимых для выполнения авторизации и проверки целостности данных пакета (см. также Пример 6):

- `"X-Digital-Sign"` – ЦП пакета, закодированная по схеме **Base64**;
- `"X-Digital-Sign-Alg"` – алгоритм вычисления ЦП. Укажите одно из следующих значений, соответствующее алгоритму, выбранному в п.2:
  - `MD5withRSA` – если для формирования ЦП использовался алгоритм MD5;
  - `SHA1withRSA` – если использовался алгоритм SHA1;
- `"X-Digital-Sign-Login"` – логин персоны (может быть закодирован по схеме **Base64**).

ВНИМАНИЕ

Так как открытый ключ привязан к персоне, то перед отправкой XML-запроса необходимо проверить, что соответствующая персона имеет разрешение на выполнение данного запроса (см. раздел Роли персон).

Пример 6. Формирование заголовка запроса с цифровой подписью.

Используется XML-запрос:

```
<?xml version="1.0" encoding="windows-1251"?>
<request>
  <client terminal="1234567" software="Dealer v0"/>
    <agents>
      <getBalance/>
    </agents>
</request>
```

Формирование ЦП XML-запроса описано выше (см. Пример 5). Получаем строку вида:

```
XTC5UVZ3mjXA+uZq5rb76iNK7x41A2S8expSPsYVsDlNlbSheG0ltsbuXZJCcPTUr7RCIFZmDbKv
IGD57f8ipPUrtjObMnd02hSQVpO6J0/+V2UdE/aLBKg7e3pyTIBE9gUOW/W8ySEqV3Mg7gd700Pp
WmHv7Uu7jpKcs5a8Jmc=
```

1. Добавляем следующие поля в HTTP заголовок запроса:

```
X-Digital-Sign:
XTC5UVZ3mjXA+uZq5rb76iNK7x41A2S8expSPsYVsDlNlbSheG0ltsbuXZJCcPTUr7RC
IFZmDbKvIGD57f8ipPUrtjObMnd02hSQVpO6J0/+V2UdE/aLBKg7e3pyTIBE9gUOW/W8ySEqV3Mg7
gd700PpWmHv7Uu7jpKcs5a8Jmc=
X-Digital-Sign-Alg: SHA1withRSA
X-Digital-Sign-Login: person_login
```

так как для шифрования пакета использовался алгоритм SHA1 и закрытый ключ персоны с логином person_login.

2. В результате получаем следующий запрос (приведены не все HTTP-заголовки):

```
X-Digital-Sign: XTC5UVZ3mjXA+uZq5rb76iNK7x41A2S8expSPsYVsDlNlbSheG0ltsbuXZJCc
PTUr7RCIFZmDbKvIGD57f8ipPUrtjObMnd02hSQVpO6J0/+V2UdE/aLBKg7e3pyTIBE9gUOW/W8yS
EqV3Mg7gd700PpWmHv7Uu7jpKcs5a8Jmc=
X-Digital-Sign-Alg: SHA1withRSA
X-Digital-Sign-Login: person_login

<?xml version="1.0" encoding="utf-8"?>
<request>
  <client terminal="1234567" software="Dealer v0" />
  <agents>
    <getBalance/>
  </agents>
</request>
```

5. РОЛИ ПЕРСОН

Право – разрешение на выполнение XML-запроса. Права группируются в роли. **Роль** – набор разрешенных на выполнение действий – прав. Каждому пользователю системы назначена роль:

1. Если персоне назначена роль, то все XML-запросы, разрешенные этой роли, могут быть выполнены данной персоной.
2. Если персоне не назначена роль при создании, в системе ей назначается роль «бесправная» с самым ограниченным набором прав. Персона с такой ролью также может выполнять некоторые XML-запросы.

Пример 7. Права и роли персоны

Роль **Бесправная** может только просматривать открытую информацию и отправлять запросы, для выполнения которых не требуется наличие особых прав (например, запрос `getPayments`).

Роль **Мониторинг** имеет права на просмотр баланса субагентов, мониторинг терминалов, просмотр статистики и некоторые другие операции.

Роль **Продавец** имеет права на просмотр баланса субагентов и прием платежей.

В Протоколе взаимодействия терминального ПО и процессинга для каждого XML-запроса указан набор ролей, имеющих права на его выполнение. Кроме того, запрос может быть разрешен для выполнения любым зарегистрированным пользователям системы.

Просмотреть роль, назначенную вашей персоне, вы можете с помощью следующих запросов:

1. Отправить запрос `getPersonInfo` для получения идентификатора роли;
2. С помощью запроса `getRoles` вы можете найти название роли по полученному идентификатору.

6. ПРИМЕРЫ ФОРМИРОВАНИЯ ЗАПРОСОВ

6.1. Запрос без параметров

Для получения регистрационных данных терминала (запрос `getTerminalInfo`) необходимы только авторизационные данные, дополнительных данных не требуется.

Для формирования запроса выполните следующие действия:

1. Составьте запрос по формату (см. Протокол взаимодействия терминального ПО и процессинга):

```
<?XML version="1.0" encoding="windows-1251"?>
<request>
  <auth …/>
  <client terminal="123" software="Dealer v0" serial="123456"/>
  <terminals>
    <getTerminalInfo/>
  </terminals>
</request>
```

2. Заполните авторизационные данные в запросе (см. Пример 2):

```
  <auth login="login" sign="password-sign" signAlg="MD5"/>
  <client terminal="123" software="Dealer v0" serial="123456"/>
```

3. Отправьте сформированный запрос на один из следующих адресов:
   - http://xml1.qiwi.com/xmlgate/xml.jsp
   - http://xml2.qiwi.com/xmlgate/xml.jsp
   - https://xml1.qiwi.com/xmlgate/xml.jsp
   - https://xml2.qiwi.com/xmlgate/xml.jsp

4. В ответе Вы получите:

```
<?XML version="1.0" encoding="windows-1251"?>
<response result="0">
  <terminals>
    <getTerminalInfo result="0">
      <terminal address="Window" agent="1234567" agent-name=" "
configuration="111821814" id="123" info=" " phone="8-901-111-11" receipt-
address="Варшавское ш., 125" region-id="26" region-name="г Москва" type="4"/>
    </getTerminalInfo>
  </terminals>
</response>
```

6.2. Запрос с обязательными параметрами

Для выгрузки движений по терминальным счетам (запрос `getAccountingTerminal`) необходимо указывать обязательный параметр – дату выгрузки:

ПРИМЕЧАНИЕ

Если Вы не укажете в запросе обязательный параметр, в ответе вернется ненулевой код ошибки или будут отсутствовать данные.

1. Составьте запрос по формату (см. Протокол взаимодействия терминального ПО и процессинга):

```
<?XML version="1.0" encoding="windows-1251"?>
<request>
  <auth … />
  <client … />
  <accountingReports>
    <getAccountingTerminal>
      <day>2012-04-12</day>
    </getAccountingTerminal>
  </accountingReports>
</request>
```

2. Заполните авторизационные данные в запросе (см. Пример 2):

```
  <auth login="login" sign="password-sign" signAlg="MD5"/>
  <client terminal="123" software="Dealer v0" serial="123456"/>
```

3. Отправьте сформированный запрос на один из следующих адресов:
   - http://xml1.qiwi.com/xmlgate/xml.jsp
   - http://xml2.qiwi.com/xmlgate/xml.jsp
   - https://xml1.qiwi.com/xmlgate/xml.jsp
   - https://xml2.qiwi.com/xmlgate/xml.jsp

4. В ответе Вы получите:

```
<?XML version="1.0" encoding="windows-1251"?>
<response result="0">
  <accountingReports>
    <getAccountingTerminal result="0">
      <row amnt="2402.76" cnt="16" from_acc="10" pay_id="5743" prv_id="1"
to_acc="23" trm_id="11111111"/>
      <row amnt="197.24" cnt="16" from_acc="10" pay_id="5743" prv_id="1"
to_acc="25" trm_id="11111111"/>
      <row amnt="2600" cnt="16" from_acc="22" pay_id="5743" prv_id="1" to_acc="10"
       trm_id="11111111"/>
      <row amnt="2600" cnt="16" from_acc="30" pay_id="5" prv_id="1" to_acc="29"
       trm_id="11111111"/>
      ...
    </getAccountingTerminal>
  </accountingReports>
</response>
```

7. РЕЖИМЫ ОБРАБОТКИ ДЕЙСТВИЙ

Протокол поддерживает следующие режимы обработки действий:

- В *Синхронном режиме* (`mode="sync"`) ответ на действие возвращается сразу после выполнения запроса. Данный режим рекомендуется использовать в следующих случаях:
  - выбранное действие не поддерживает асинхронный режим обработки;
  - система формирует и возвращает ответ быстро, дополнительной нагрузки на БД нет. Примерами таких действий являются получение справочников, интерфейса терминала, проведение платежей и некоторые др.
- В *асинхронном режиме* (`mode="async"`) запрос попадает в очередь, а в ответ пользователю возвращается его идентификатор в очереди. По имеющемуся идентификатору запроса в очереди пользователь далее должен отправить еще один запрос и получить сформированный системой ответ.

  Использовать данный режим рекомендуется, если выполнение выбранного вами запроса требует значительного времени обработки со стороны системы. Примерами таких запросов являются поиск платежа (`getPayments`) и все запросы интерфейса `accountingReports`.

7.1. Работа в синхронном режиме

Использование синхронного режима:

1. Сформируйте запрос (примеры составления XML-запросов см. в разделе 6):

```
<?XML version="1.0" encoding="windows-1251"?>
<request>
  <auth login="login" sign="password-sign" signAlg="MD5"/>
  <client terminal="123" software="Dealer v0" serial="123456"/>
  <system>
    <getActions/>
  </system>
</request>
```

2. Отправьте сформированный запрос на сервер.
3. Сервер вернет ответ с запрошенными данными или код ошибки обработки запроса:

```
<?XML version="1.0" encoding="windows-1251"?>
<response result="0">
  <system>
    <getActions result="0">
      <row description="Баланс агента" execution="Возможно асинхронно"
interface="agents" name="getBalance"/>
      <row description="Описание агента" execution="Возможно асинхронно"
interface="agents" name="getAgentInfo"/>
      ...
    </getActions>
  </system>
</response>
```

7.2. Работа в асинхронном режиме

Использование асинхронного режима:

1. Сформируйте запрос (примеры составления XML-запросов см. в разделе 6):

```
<?XML version="1.0" encoding="windows-1251"?>
<request>
```

```
  <auth login="login" sign="password-sign" signAlg="MD5"/>
  <client terminal="123" software="Dealer v0" serial="123456"/>
  <system>
    <getActions/>
  </system>
</request>
```

2. В тег с названием запроса добавьте атрибут `mode="async"`:

```
<?XML version="1.0" encoding="windows-1251"?>
<request>
  <auth login="login" sign="password-sign" signAlg="MD5"/>
  <client terminal="123" software="Dealer v0" serial="123456"/>
  <system>
    <getActions mode=”async”/>
  </system>
</request>
```

3. Отправьте сформированный запрос на сервер.
4. Ответ от сервера содержит следующие данные об обработке отправленного Вами запроса:

```
<?XML version="1.0" encoding="windows-1251"?>
<response result="0">
  <system>
    <getActions quid="123456" result="0" status="1"/>
  </system>
</response>
```

- `quid` – идентификатор запроса в очереди;
- `result` – код ошибки обработки запроса:
  - `0` – запрос составлен корректно и принят в обработку;
  - `отличен от 0` – запрос не принят в обработку (возможно, неверный формат запроса, пропущен обязательный параметр, провайдер в данный момент не доступен и др.).
- `status` – статус выполнения запроса, отправленного в асинхронном режиме. Возможные статусы обработки запроса, находящегося в очереди:
  - `1` – ожидание обработки.

    Запрос поставлен в очередь на обработку, ему присвоен идентификатор в очереди. Для получения ответа необходимо продолжить опрашивать сервер;
  - `2` – обрабатывается.

    Запрос находится в обработке на сервере. Для получения ответа необходимо продолжить опрашивать сервер;
  - `3` – успешно обработан;
  - `4` – ошибка обработки.

    Проверьте сформированный асинхронный запрос на наличие ошибок, а затем повторите его отправку;
  - `5` – таймаут.

    Попробуйте сократить временной отрезок, за который ищите данные, или наложите дополнительные ограничения в запросе;
  - `6` – результат запроса был удален из БД

    После присвоения запросу идентификатора в очереди запрос и сформированный ответ на него будет храниться в течение 3-5 дней на сервере. После указанного срока сформированный системой ответ будет удален из базы данных.

5. Для получения ответа на запрос (или статуса обработки) по его идентификатору в очереди необходимо в отправленный запрос добавить полученный от сервера идентификатор запроса (атрибут `quid`):

```
<?XML version=”1.0” encoding=”windows-1251”?>
<request>
  <auth login="login" sign="password-sign" signAlg="MD5"/>
  <client terminal="123" software="Dealer v0" serial="123456"/>
  <system>
    <getActions mode="async" quid="123456"/>
  </system>
</request>
```

6. В ответ сервер присылает статус обработки асинхронного запроса (Пример 8) и, если запрос успешно обработан, запрошенные данные (Пример 9).

Пример 8. Ответ сервера на запрос пользователя о статусе асинхронного запроса

```
<?XML version="1.0" encoding="windows-1251"?>
<response result="0">
  <system>
    <getActions quid="123456" result="0" status="1"/>
  </system>
</response>
```

Пример 9. Ответ сервера на запрос пользователя о статусе асинхронного запроса с запрошенными данными

```
<?XML version="1.0" encoding="windows-1251"?>
<response result="0">
  <providers>
    <getProviders count="2256" quid="50" result="0" status="3">
      <row fiscal-name="Телевидение" long-name="Полное название провайдера"
max-amount="14999" min-amount="1" prv-id="1" prv-inn="1111111111"prv-
suppotr-phone="8(495) 111-11-11, 8 (495) 111-11-12" receipt-name="Провайдер"
regexp="^\d+$" short-name="Короткое название провайдера"\>
      …
    </getProviders>
  </providers>
</response>
```

8. НАСТРОЙКИ ДЛЯ ТЕРМИНАЛОВ

В Системе ОСМП используются следующие настройки для терминалов:

- **Интерфейс** – иерархическая структура групп и содержащихся в них провайдеров. Для начала работы необходимо получить список провайдеров и настроить интерфейс оплаты услуг для этих провайдеров. Подробнее о настройке интерфейса читайте в разделе 8.1.
- **Терминальная комиссия** – размер дополнительной комиссии, которую агент может взимать при платежах с терминала. Существует два вида дополнительной комиссии: фиксированная и дифференцированная. Подробнее о механизмах расчета комиссии см. в разделе 8.2.

ВНИМАНИЕ

Конфигурация – характеристика текущего состояния настроек терминала, включающих интерфейс, комиссии, номерные емкости и др.

Конфигурация характеризуется параметром «идентификатор конфигурации» (тег `configuration` в ответе на запрос `getTerminalInfo`). При изменении какого-либо параметра в настройках терминала данный идентификатор изменяется.

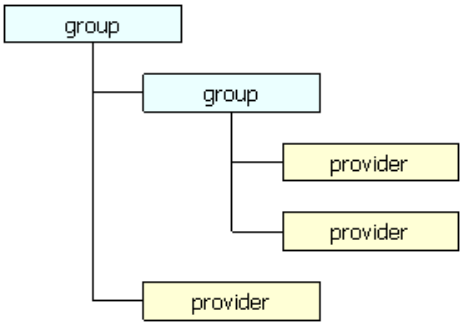
Если терминал получает информацию о том, что идентификатор конфигурации изменился, должно выполняться обновление изменившихся данных.

8.1. Получение интерфейса терминала

Интерфейс терминала – это совокупность средств и методов взаимодействия пользователя с *Системой ОСМП* посредством терминалов.

Интерфейс терминала представляет собой иерархическую структуру групп (категорий услуг) и содержащихся в них провайдеров (Рис. 2), а также параметры отображения страниц интерфейса провайдера (порядок и элементы управления для каждой страницы см. на Рис. 3).

Рис. 2. Структура XML-тегов интерфейса



Загрузка интерфейса на терминал производится с помощью следующих действий интерфейса `providers`:

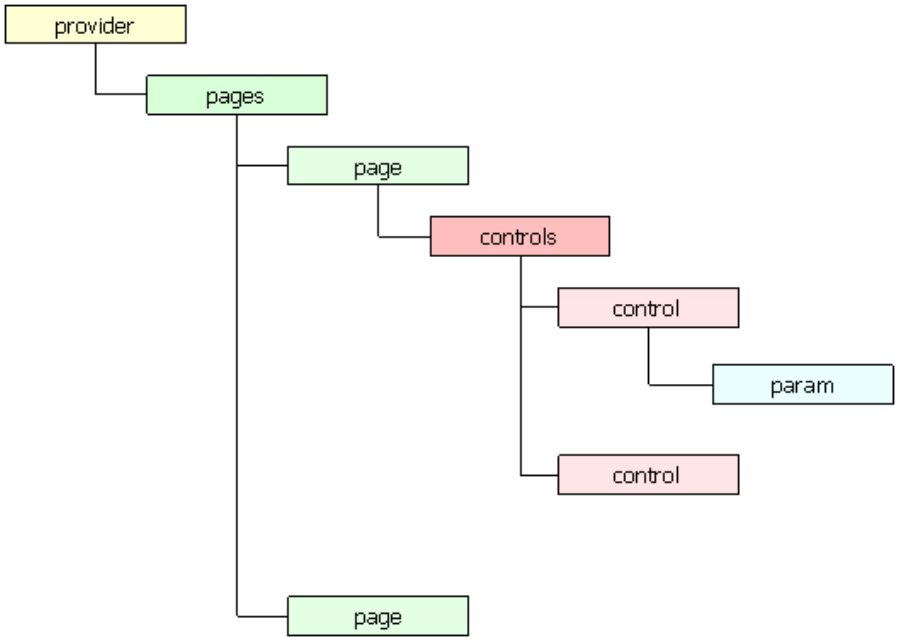
- `getUIGroups` – получение структуры групп и провайдеров для формирования пользовательского интерфейса терминала (описание структуры см. в Протоколе взаимодействия терминального ПО и процессинга).

ВНИМАНИЕ

Запрос возвращает только разрешенных для отображения на текущем терминале провайдеров.

Каждой группе и провайдеру присваивается порядковый номер, определяющий его положение в экранном интерфейсе (Рис. 4).

Рис. 3. Структура XML-тегов страниц интерфейса провайдера



- `getUIProviders` – получение подробной информации об отображении страниц провайдеров, включая порядковый номер провайдера, а также элементы управления: маски полей ввода и др.

  Для каждого провайдера сформирована последовательность страниц интерфейса, определен порядок отображения страниц, а также элементы управления страницами (описание формата элементов страниц см. в Протоколе взаимодействия терминального ПО и процессинга).

8.1.1. Структура интерфейса

Рассмотрим пример получения и построения интерфейса:

Пример 10. Интерфейс терминала

```
<?xml version="1.0" encoding="windows-1251"?>
<response result="0">
  <providers>
    <getUIGroups result="0">
      <group id="20" logo="logos/!anonymous/icon_mobile.gif" logo_crc="CFE03E84"
logo_size="5054" name="Сотовая связь" orderId="1" tag="visible,ranges">
        <logos>
         <logo crc="CFE03E84" path="logos/!anonymous/icon_mobile.gif" size="5054"
type="standard"/>
        </logos>
```

```
        <provider id="2" orderId="13" showInTop="5"/>
        ...
      </group>
      <group id="103" logo="logos/!anonymous/grp103Icon_internet_telephons.gif"
logo_crc="fdf52353" logo_size="10744" name="Интернет и телефония" orderId="16"
tag="visible">
        <logos>
          <logo crc="fdf52353" path="logos/!anonymous/internet_telephons.gif"
size="10744" type="standard"/>
        </logos>
        <provider id="217" orderId="2" showInTop="2" tag="visible"/>
        <provider id="507" orderId="16" tag="visible"/>
        <provider id="561" orderId="17" showInTop="3" tag="visible"/>
        ...
      </group>
        ...
    </getUIGroups>
  </providers>
</response>
```

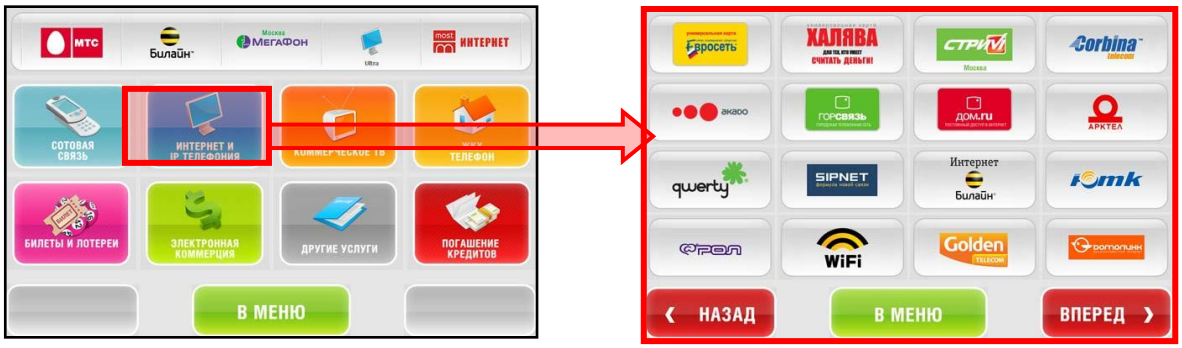
В соответствии со списком групп, полученным в запросе `getUIGroups`, страница категорий услуг (упорядоченных по параметру `orderId`) выглядит следующим образом (Рис. 4):

1. Группа **Сотовая связь** (ID=20).
2. Группа **Интернет и телефония** (ID=103) и т.д.

Внутри группы **Интернет и телефония** провайдеры также упорядочены по параметру `orderId`:

1. Евросеть (ID=217)
2. Халява (ID=507)
3. Стрим (ID=561) и т.д.

Рис. 4. Внешний вид групп и провайдеров в интерфейсе АСО (пример)



8.1.2. Параметры отображения страниц интерфейса провайдера

Для каждого провайдера (см. Пример 11) определяются страницы сценария оплаты услуги, порядок их отображения, а также элементы управления для страниц интерфейса:

- `page` – содержит идентификатор страницы и порядковый номер ее отображения в интерфейсе провайдера,
  - `control` – элементы управления, например:

- цифровая клавиатура (`type="keyboard"`),
- поле ввода номера телефона/счета/дополнительного номера (`type="text_input"`);
- атрибуты элемента, например, для поля ввода номера:
  - маска ввода номера;
  - регулярное выражение для проверки номера;
  - заголовок поля.

Например, для провайдера МТС реализованы следующие страницы (Пример 11, Рис. 5):

Пример 11. Интерфейс провайдера МТС

```
<?xml version="1.0" encoding="windows-1251"?>
<response result="0">
  <providers>
    <getUIProviders result="0">
      <provider id="1" fiscalName="Сотовая св." grpId="20" … receiptName="Сотовая
связь" sName="MTC " supportPhone="8-800-250-0890" keywords="mts|MTC">
        <logos>
          <logo type="standard" path="logos/!anonymous/prv11.gif" crc="b9410bc4"
size="2505"/>
        </logos>
        <constParams/>
        <pages>
          <page pageId="104" orderId="1">
            <controls>
              <control type="text_input" orderId="1" strip="true" nobr="false"
mask="(<!^\\d+${3}>)<!^\\d+${3}>-<!^\\d+${2}>-<!^\\d+${2}>" name="account"
footer="Номер телефона вводится без '8'" regexp="^\\d{10}$" header="Введите номер
телефона"/>
              <control type="keyboard" orderId="2" layout="DG"/>
            </controls>
          </page>
        </pages>
      ...
    </getUIProviders>
  </providers>
</response>
```

Рис. 5. Внешний вид страницы провайдера МТС в интерфейсе

В данном случае интерфейс содержит только одну страницу с единственным полем ввода `<control type="text_input" …/>`, в котором указаны:

- заголовок поля `header="Введите номер телефона"`,

- комментарий `footer="Номер телефона вводится без '8'"`,
- маска ввода `mask="(<!^\\d+${3}>)<!^\\d+${3}>-<!^\\d+${2}>-<!^\\d+${2}>"`
- регулярное выражение для проверки номера `regexp="^\\d{10}$"`

Также на странице отображается цифровая клавиатура `<control type="keyboard" orderId="2" layout="DG"/>`.

ПРИМЕЧАНИЕ

Страницы подтверждения номера и внесения денег одинаковы для всех провайдеров и не содержатся в описании интерфейса.

8.1.3. Дополнительные параметры интерфейса провайдера

В параметрах страниц интерфейса провайдера могут присутствовать дополнительные параметры (экстра-поля) для корректной обработки платежа провайдером. Эти параметры необходимо передавать в платежных запросах в специальном теге `<extras …>`.

Список необходимых экстра-полей содержится в ответе на запрос интерфейса провайдеров `getUIProviders`. Из ответа на запрос необходимо извлечь следующие атрибуты (параметры экстра-полей) конфигурации провайдера внутри тега `<provider id="…">` (см. также Пример 12):

- данные с префиксом `_extra_` из вложенного тега `constParams`, например:
  - `ev_bank="osmp"` (`<param name="_extra_ev_bank" value="osmp"/>`);
  - `PT="1"` (`<param name="_extra_PT" value="1"/>`);
- данные во вложенном теге `control` с атрибутом `disp_type="extra"` или поля ввода, кроме стандартного поля номера телефона/счета `account`. Например:
  - `ev_fname`="имя плательщика, введенное в сценарии оплаты" из элемента интерфейса в теге `<control type="text_input" orderId="2" name="ev_fname" mask="<!^.+${1,20}>" strip="false" nobr="false" regexp="^.{1,20}$" disp_type="extra" header="Введите имя"/>`.

ПРИМЕЧАНИЕ

Если минимальное количество цифр в регулярном выражении для проверки дополнительного поля ввода (атрибут `regexp`) – 0 (например, `regexp="^\\d{0,10}$`), то данное поле ввода передавать в запросах не обязательно.

- данные из вложенного тега `param` внутри тега `control`. Например:
  - `account_type="2"` (`<param name="_extra_account_type" value="2"/>`);
  - `oper_type="3"` (`<param name="_extra_oper_type" value="3"/>`);

ВНИМАНИЕ

Если имя дополнительного параметра начинается с префикса `_extra_`, необходимо удалить префикс перед помещением параметра в список экстра-полей платежного запроса.

Пример 12. Интерфейс провайдера с дополнительными полями (выделены цветом)

```
<?xml version="1.0" encoding="windows-1251"?>
<response result="0">
  <providers>
    <getUIProviders result="0">
      ...
<provider id="9998" grpId="113" … buttonName="Skype" … maxSum="15000"
keywords="Skype|skaip|skypecom|скайп">
  <logos>
    <logo type="standard" path="logos/5543810459565143056" crc="F4A8B0F6"
size="3687" />
  </logos>
  <constParams>
    <param name="_extra_sum" value="14999" />
  </constParams>
  <pages>
    <page pageId="53201" orderId="1">
     <controls>
      <control type="text_input" orderId="1" mask="<!^[a-zA-Z0-9,._-]+${6,32}>"
strip="false" header="Введите ваш логин Skype" footer="Внимание! Установлено
ограничение по количеству платежей: не более 4 платежей в день или 6 в неделю на
один логин в Skype." name="account-number" disp_desc="Логин Skype"
disp_type="receipt" nobr="false" regexp="^[a-zA-Z][a-zA-Z0-9-_,.]{5,31}$" />
      <control type="keyboard" orderId="2" layout="AL" />
     </controls>
    </page>
    <page pageId="53202" orderId="2" pageType="input_page">
     <controls>
       <control type="keyboard" orderId="2" layout="DG" />
       <control type="text_input" orderId="1" mask="(<!^\\d+${3}>)<!^d+${3}>-<!^
d+${2}>-<!^d+${2}>" strip="true" header="Введите номер телефона"
footer="Ознакомиться с текстом публичной оферты ОБ ОКАЗАНИИ УСЛУГ БЕЗНАЛИЧНЫХ
РАСЧЕТОВ С ИСПОЛЬЗОВАНИЕМ ПРЕДОПЛАЧЕННОЙ КАРТЫ «ЕДИНАЯ РАСЧЕТНАЯ КАРТА QIWI» КИВИ
БАНК (ЗАО) Вы можете в разделе Информация.\n Если Вы ввели чужой номер телефона, то
претензии по зачислению рассмотрены не будут." name="account" nobr="false"
regexp="^d{10}$" />
     </controls>
    </page>
  </pages>
</provider>
      ...
</getUIProviders>
</providers>
</response>
```

8.2. Расчет терминальной комиссии

В *Системе* для каждого провайдера может быть установлена *фиксированная ставка комиссии* (в процентах от суммы платежа).

ПРИМЕЧАНИЕ

Для получения XML файла с информацией о ставках комиссии выполняется запрос `getCommissions`.

Кроме того, существует возможность установить различные правила взимания комиссии в зависимости от времени суток, суммы платежа и др. параметров. Для этого необходимо создать *Профиль комиссии* – набор правил вычисления суммы комиссии.

ПРИМЕЧАНИЕ

Для получения XML файла с информацией о профилях комиссии выполняется запрос `getCommissionProfiles`.

Для расчета комиссии используются следующие правила (см. Пример 13):

- Если для провайдера задан и профиль, и фиксированная ставка комиссии, то профиль имеет более высокий приоритет.
- В случае, если несколько правил профиля удовлетворяют условиям, комиссия будет рассчитана в соответствии с правилом, имеющим меньший порядковый номер.
- Если не может быть применен ни один из профилей, будет использоваться фиксированная ставка комиссии.

Пример 13. Пример расчета комиссии

Пример расчета комиссии при следующих параметрах:

1. Фиксированная ставка комиссии для провайдера: 2,5%.

2. Профиль комиссии:

Правило	Условия	Комиссия %	Абсолютное значение комиссии	Не меньше
1	Сумма платежа < 500 руб	3% от суммы платежа	10 руб	20 руб
2	Сумма < 500 руб Время с 06:00 до 16:00		7 руб	

В случае если несколько правил профиля удовлетворяют условиям, комиссия будет рассчитана в соответствии с правилом, имеющим меньший порядковый номер – в данном случае правило №1.

Результат: при платеже на сумму 400 руб. комиссия составит 400 руб.*3% +10 = 22 руб

Если не может быть применен ни один из профилей, будет использоваться фиксированная ставка комиссии (2,5% от суммы платежа).

9. ПРОВЕДЕНИЕ ПЛАТЕЖЕЙ

Для проведения платежей в протоколе используются следующие действия интерфейса **providers**:

Табл. 2. Запросы на проведение платежей

Действие	Описание	Режимы обработки
addOfflinePayment	Запрос на добавление платежа (в режиме оффлайн)	Строго синхронно
checkPaymentRequisites	Авторизация платежа без записи информации в базу	Возможно асинхронно
authorizePayment	Запрос на авторизацию платежа	Строго синхронно
confirmPayment	Запрос позволяет подтвердить платеж, авторизованный в результате выполнения запроса **authorizePayment**	Строго синхронно

Подробное описание запросов см. в Протоколе взаимодействия терминального ПО и процессинга.

9.1. Статусы платежей

При выполнении запросов на авторизацию и проведение платежей в теге `payment` ответа возвращаются следующие данные (см. Пример 14):

- статус платежа (атрибут `status`),
- код ошибки проведения платежа (атрибут `result`),
- логический признак фатальности ошибки (атрибут `fatal`).

Пример 14. Ответ на запрос авторизации платежа.

```
<?xml version="1.0" encoding="windows-1251"?>
<response result="0">
  <providers>
    <authorizePayment result="0">
      <payment … fatal="false" result="0" status="3" …>
        …
      </payment>
    </authorizePayment>
  </providers>
</response>
```

Платеж может находиться в одном из следующих статусов:

- **3** – «авторизован»;
- **1** – «проводится»;
- **2** – «проведен» (финальный статус);
- **0** – «ошибка проведения» (финальный статус).

Полный перечень кодов ошибок см. в Протоколе взаимодействия терминального ПО и процессинга.

Наличие признака фатальной ошибки (`fatal="true"`) означает отрицательный ответ на клиентский запрос (статус "0"), тогда как нефатальная ошибка (`fatal="false"`) означает промежуточный статус обработки клиентского запроса (дальнейшие действия зависят от конкретной ошибки).

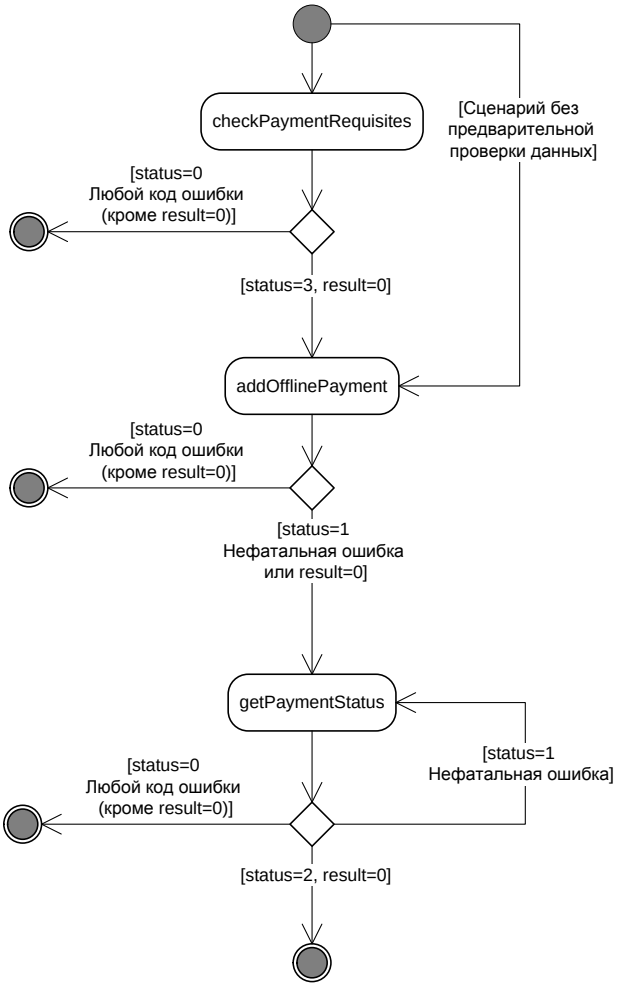
9.2. Сценарии проведения платежей

В *Системе* поддерживаются два метода проведения платежей:

1. оффлайн,
2. онлайн.

9.2.1. Оффлайн проведение платежей

Рис. 6. Сценарий оффлайн проведения платежа



1. Сценарий может выполняться как с предварительной проверкой реквизитов платежа, так и без нее. В первом случае с терминала отправьте запрос **providers.checkPaymentRequisites**, во втором – см. п. 5 ниже.

ВНИМАНИЕ

Предварительная проверка реквизитов требуется, если в интерфейсе провайдера присутствует страница с атрибутом `useOnline="true"`, например: `<page pageId="36930" orderId="3" nextPage="-1" useOnline="true">`.

В запросе **checkPaymentRequisites** укажите параметры платежа:

- идентификатор платежа на терминале;
- данные о принятых от клиента денежных средствах;
- данные о зачисляемых на счет клиента у провайдера денежных средствах (в том числе идентификатор услуги в ОСМП);
- информация о предчеке;
- значения дополнительных (экстра) полей, необходимые для корректной обработки запроса.

2. Сервер проверяет реквизиты платежа, однако не записывает данные в базу и не присваивает платежу номер транзакции.
3. Сервер отправляет реквизиты платежа поставщику услуг.
4. **Ответ** от сервера может включать информацию о результатах проверки реквизитов, а также некоторые дополнительные поля, которые необходимо отображать пользователю:
   - данные об успешности выполнения запроса;
   - данные о платеже, реквизиты которого были отправлены на проверку:
     - московские дата и время приема платежа в процессинг;
     - идентификатор платежа на терминале;
     - код ошибки обработки платежа;
     - статус успешной проверки (`status="3"`);

ВНИМАНИЕ

В случае получения финального статуса `status="0"` необходимо рассмотреть код ошибки для выяснения причины ошибки проверки реквизитов.

     - идентификатор транзакции в процессинге;
   - данные в дополнительных (экстра) полях.
5. В случае успешной авторизации с терминала (возвращен статус `status="3"`) отправьте запрос **providers.addOfflinePayment**. В запросе укажите параметры платежа:
   - идентификатор платежа на терминале;
   - данные о принятых от клиента денежных средствах;
   - данные о зачисляемых на счет клиента у провайдера денежных средствах (в том числе идентификатор услуги в ОСМП);
   - информация о чеке;
   - значения дополнительных (экстра) полей, необходимые для корректной обработки запроса.
6. **Ответ** от сервера содержит информацию о результате записи платежа в базу, а также уникальный идентификатор транзакции в *Системе* (в случае успешной записи):

- данные об успешности выполнения запроса;
- данные о проводимом платеже:
  - московские дата и время приема платежа в процессинг;
  - идентификатор платежа на терминале;
  - код ошибки обработки платежа;
  - статус проведения платежа;

ВНИМАНИЕ

В случае получения финального статуса `status="0"` необходимо рассмотреть код ошибки для выяснения причины ошибки проверки реквизитов.

  - идентификатор транзакции в процессинге;
- данные в дополнительных (экстра) полях.

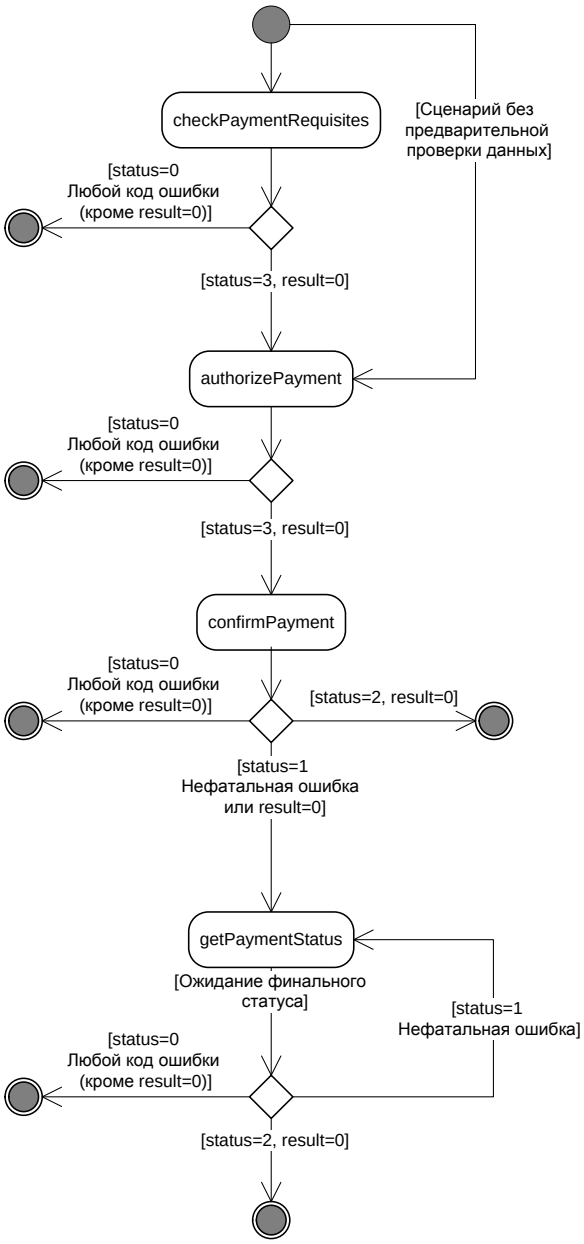
7. Сервер отправляет реквизиты платежа поставщику услуг.
8. В случае получения нефинального статуса `status="1"` необходимо повторно запрашивать статус платежа для получения информации о том, проведен платеж успешно или нет:
   8.1. С терминала отправьте запрос **providers.getPaymentStatus**. Необходимо указать идентификатор платежа на терминале (`<payment id="7473517739148">`).
   8.2. **Ответ** от сервера включает информацию о статусе платежа.

ВНИМАНИЕ

Запрос статуса платежа (providers.getPaymentStatus) необходимо отправлять на сервер до тех пор, пока не будет получен финальный статус платежа. До получения финального статуса платежа вывод о том, проведен платеж или нет, сделать нельзя.

9.2.2. Онлайн проведение платежей

Рис. 7. Сценарий онлайн проведения платежа



1. Сценарий может выполняться как с предварительной проверкой реквизитов платежа, так и без нее. В первом случае с терминала отправьте запрос **providers.checkPaymentRequisites**, во втором – см. п.5 ниже.

ВНИМАНИЕ

Предварительная проверка реквизитов требуется, если в интерфейсе провайдера присутствует страница с атрибутом `useOnline="true"`, например: `<page pageId="36930" orderId="3" nextPage="-1" useOnline="true">`.

В запросе **checkPaymentRequisites** укажите следующие параметры:

- идентификатор платежа на терминале;
- данные о принятых от клиента денежных средствах;
- данные о зачисляемых на счет клиента у провайдера денежных средствах (в том числе идентификатор услуги в ОСМП);
- информация о предчеке;
- значения дополнительных (экстра) полей, необходимые для корректной обработки запроса.

2. Сервер проверяет реквизиты платежа, однако не записывает данные в базу и не присваивает платежу номер транзакции.
3. Сервер отправляет реквизиты платежа поставщику услуг.
4. **Ответ** от сервера содержит информацию о результатах проверки реквизитов, а также некоторые дополнительные поля, которые необходимо отображать пользователю:
   - данные об успешности выполнения запроса;
   - данные о платеже, реквизиты которого были отправлены на проверку:
     - московские дата и время приема платежа в процессинг;
     - идентификатор платежа на терминале;
     - код ошибки обработки платежа;
     - статус успешной проверки (`status="3"`);

ВНИМАНИЕ

В случае получения финального статуса `status="0"` необходимо рассмотреть код ошибки для выяснения причины ошибки проверки реквизитов.

     - идентификатор транзакции в процессинге;
   - данные в дополнительных (экстра) полях.
5. С терминала отправьте запрос авторизации платежа **Providers.authorizePayment**.

   В запросе **authorizePayment** укажите следующие параметры:
   - идентификатор платежа на терминале;
   - данные о принятых от клиента денежных средствах;
   - данные о зачисляемых на счет клиента у провайдера денежных средствах (в том числе идентификатор услуги в ОСМП);
   - информация о чеке;
   - значения дополнительных (экстра) полей, необходимые для корректной обработки запроса.
6. Выполняется авторизация платежа у поставщика услуг.
7. **Ответ** от сервера содержит информацию о результате авторизации платежа у поставщика услуг, уникальный идентификатор транзакции в *Системе* ОСМП, а также некоторые дополнительные поля, которые необходимо отображать пользователю:
   - данные об успешности выполнения запроса;
   - данные о платеже, реквизиты которого были отправлены на проверку:
     - московские дата и время приема платежа в процессинг;
     - идентификатор платежа на терминале;

- код ошибки обработки платежа;
- статус успешной проверки (`status="3"`);

ВНИМАНИЕ

В случае получения финального статуса `status="0"` необходимо рассмотреть код ошибки для выяснения причины ошибки авторизации.

- идентификатор транзакции в процессинге;

– данные в дополнительных (экстра) полях.

8. В случае успешной авторизации с терминала (возвращен статус `status="3"`) отправьте запрос на проведение платежа **Providers.confirmPayment**.

   В запросе указывается идентификатор платежа на терминале из предыдущего запроса **authorizePayment**, см. выше п. 5.

9. Сервер отправляет платеж поставщику услуг и получает от него ответ.

10. **Ответ** от сервера включает информацию о результате отправки платежа поставщику услуг, а также уникальный идентификатор транзакции в *Системе* ОСМП:

    – данные об успешности выполнения запроса;

    – данные о платеже, реквизиты которого были отправлены на проверку:

    - московские дата и время приема платежа в процессинг;
    - идентификатор платежа на терминале;
    - код ошибки обработки платежа;
    - статус платежа (`status="1"`);

ВНИМАНИЕ

В случае получения финального статуса `status="0"` необходимо рассмотреть код ошибки для выяснения причины ошибки проведения платежа.

    - идентификатор транзакции в процессинге.

11. В случае получения нефинального статуса `status="1"` необходимо повторно запрашивать статус платежа для получения информации о том, проведен платеж успешно или нет:

    11.1. С терминала отправьте запрос **providers.getPaymentStatus**. Необходимо указать идентификатор платежа на терминале (`<payment id="7473517739148">`).

    11.2. **Ответ** от сервера включает информацию о текущем статусе платежа.

ВНИМАНИЕ

Запрос статуса платежа (providers.getPaymentStatus) необходимо отправлять на сервер до тех пор, пока не будет получен финальный статус платежа. До получения финального статуса платежа вывод о том, проведен платеж или нет, сделать нельзя.

9.3. Тестирование запросов платежа

Для тестирования правильности обработки кодов ошибок или задержки при получении ответа на запрос платежа в ПО XML-агента, нужно отправить запрос авторизации или проведения платежа (см. Табл. 2) в пользу провайдера «Тестирование XML-протокола» (идентификатор услуги 11111).

Вы управляете ответом на запрос, указывая код ошибки или время задержки в дополнительных (экстра) полях:

- `error` – код ошибки, который будет получен в ответе на запрос.

СОВЕТ

Для случайного выбора кода ошибки из списка возможных ошибок укажите в экстра-поле `error` значение *random*.

- `delay` – время в секундах, через которое должен быть получен ответ на запрос.

СОВЕТ

Для случайного выбора времени задержки укажите в экстра-поле `delay` значение *random*.

ВНИМАНИЕ

Если данные экстра-поля не содержатся в запросе, то, по умолчанию, устанавливаются `delay=0` и `error=0` (Ok).

Пример 15. XML-запрос для тестирования обработки ошибок провайдера

```
<providers>
  <authorizePayment>
    <payment id="12300">
      <from amount="5.00" currency="100"/>
      <to service="11111" account="9151234567" amount="5.00" currency="100"/>
      <extras delay="random" error="272" />
      <receipt id="1234567" date="2012-01-17T15:52:00"/>
    </payment>
  </authorizePayment>
</providers>
```

10. ЧАСТО ЗАДАВАЕМЫЕ ВОПРОСЫ

Как посмотреть список провайдеров, их маски ввода и т.д.?

Для получения списка доступных провайдеров и их интерфейсов (см. раздел 8.1.2) необходимо использовать запрос `getUIProviders`.

Формат ответа на запрос и примеры обработки ответов для разных провайдеров можно посмотреть в документе Протокол взаимодействия терминального ПО и процессинга, который можно загрузить с сайта www.qiwi.ru в разделе **Бизнесу→Агентам→Скачать ПО→Другое**.

На какие адреса необходимо отправлять запросы по протоколу?

Запросы по протоколу можно отправлять на любой из следующих адресов:

- http://xml1.qiwi.com/xmlgate/xml.jsp
- http://xml2.qiwi.com/xmlgate/xml.jsp
- https://xml1.qiwi.com/xmlgate/xml.jsp
- https://xml2.qiwi.com/xmlgate/xml.jsp

Существуют ли тестовые сервера или тестовые провайдеры?

Тестовых серверов не существует.

Для тестирования проведения платежей можно использовать провайдера «Тестирование XML-протокола» (идентификатор услуги 11111). Подробнее см. раздел 9.3.

Какие способы авторизации существуют? В каком способе авторизации необходим одноразовый пароль персоны, а в каком – постоянный?

Протокол поддерживает два способа авторизации:

- По данным в теле запроса (менее приоритетный способ авторизации, но более простой и, как показывает практика, наиболее часто используемый). Авторизация осуществляется по хэшу MD5 от постоянного (неодноразового) пароля персоны. Подробнее см. раздел 4.1.
- По цифровой подписи пакетов данных (более приоритетный способ, но сложнее в реализации). При первичной регистрации цифровой подписи используется одноразовый пароль. Подробнее см. раздел 4.2.

Как можно узнать минимальную и максимальную суммы платежа по провайдеру?

Минимальную/максимальную сумму платежа по провайдеру можно получить с помощью запроса `getUIProviders` (см. атрибуты `minSum`, `maxSum` корневого тега `providers`).

ВНИМАНИЕ

Узнать минимальную/максимальную сумму платежа можно только по провайдерам, подключенным у агента.

Если минимальная/максимальная сумма в настройках провайдера не указана, то по умолчанию она должна попадать в диапазон 0,01-15000,00 рублей.

В ответе на запрос возвращается ошибка 300

Рекомендуемые действия:

- Проверить правильность составления XML-запроса.
- Проверить доступность провайдера для агента с помощью интерфейса **сервис→управление видимостью провайдеров** на сайте agent.qiwi.com.
- Узнать, нет ли проблем с проведением платежей по данному провайдеру. Оповещения о задержке платежей публикуются на главной странице сайта agent.qiwi.com в разделе **Сообщения службы технической поддержки**.
- Проверить наличие необходимых экстра-полей в запросе платежа (подробнее см. раздел 8.1.3).

Сколько платежей можно отправлять в одном пакете?

В одном пакете можно отправить от 1 до 50 платежей.

ВНИМАНИЕ

Данное правило действует только для запроса addOfflinePayment.

Какова размерность уникального ID платежа на терминале?

Максимальная разрядность идентификатора платежа - 18 цифр. Максимальное значение идентификатора 9 223 372 036 854 775 807.

Что делать в случае ошибки с кодом 75 при проведении платежей МТС?

Данный код ошибки означает, что терминал не зарегистрирован в МТС. Необходимо обратиться к курирующему менеджеру.

СПИСОК РИСУНКОВ

Рис. 1. Структура интерфейса 6
Рис. 2. Структура XML-тегов интерфейса 21
Рис. 3. Структура XML-тегов страниц интерфейса провайдера 22
Рис. 4. Внешний вид групп и провайдеров в интерфейсе АСО (пример) 23
Рис. 5. Внешний вид страницы провайдера МТС в интерфейсе 24
Рис. 6. Сценарий оффлайн проведения платежа 29
Рис. 7. Сценарий онлайн проведения платежа 32

СПИСОК ТАБЛИЦ

Табл. 1. Глоссарий 3
Табл. 2. Запросы на проведение платежей 28

СПИСОК ПРИМЕРОВ

Пример 1. Пример запроса 5
Пример 2. Пример авторизации по данным в теле запроса 10
Пример 3. Генерация RSA ключей. 11
Пример 4. Сохранение открытого ключа. 11
Пример 5. Формирование цифровой подписи XML-запроса. 13
Пример 6. Формирование заголовка запроса с цифровой подписью. 14
Пример 7. Права и роли персоны 15
Пример 8. Ответ сервера на запрос пользователя о статусе асинхронного запроса 20
Пример 9. Ответ сервера на запрос пользователя о статусе асинхронного запроса с запрошенными данными 20
Пример 10. Интерфейс терминала 22
Пример 11. Интерфейс провайдера МТС 24
Пример 12. Интерфейс провайдера с дополнительными полями (выделены цветом) 26
Пример 13. Пример расчета комиссии 27
Пример 14. Ответ на запрос авторизации платежа. 28
Пример 15. XML-запрос для тестирования обработки ошибок провайдера 35