Devino Documentation Documentation

Выпуск latest

июн. 27, 2022

SMS

В данном разделе представлена документация по различными протоколам для взаимодействия с платформой Devino Telecom. Документация содержит описание API сервисов для отправки и приема SMS-сообщений, отправки Email и выполнения запросов IMSI Каждое руководство включает в себя описание схемы работы с API, краткий обзор всех параметров, а также подробные описания по каждому из параметров, снабженные реальными примерами.

Документация предназначена для разработчиков, которые хотят добавить возможность взаимодействия с сервисом отправки SMS и Email сообщений на страницы своих сайтов или в свои приложения.

Для начала использования API:

1. Зарегистрируйтесь в Личном кабинете;
2. Заключите договор;
3. Ознакомьтесь с интересующей документацией.

Глава 1

HTTP API

1.1 Обзор API

Предоставляемый API сервис отправки SMS-сообщений позволяет осуществить:

- Аутентификацию
- Получение баланса авторизованного пользователя
- Отправку SMS-сообщения на один номер без учета часового пояса получателя
- Отправку SMS-сообщения на один номер с учетом часового пояса получателя
- Отправку SMS-сообщения на несколько номеров без учета часового пояса получателя
- Отправку SMS-сообщения на несколько номеров с учётом часового пояса получателя
- Получение статуса отправленного SMS-сообщения
- Получение SMS-сообщений за период
- Получение статистики по SMS-рассылкам
- Отправку Viber-сообщения на один номер без учета часового пояса получателя
- Отправку Viber-сообщения на несколько номеров без учета часового пояса получателя
- Отправка Viber-сообщения на один номер с переотправкой по SMS
- Отправка Viber-сообщения на несколько номеров с переотправкой по SMS
- Получение статуса отправленного Viber-сообщения

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

API Сервиса отправки SMS-сообщений организован в соответствии с принципами REST, что позволяет обмениваться HTTPS-запросами с URL-encoded кодировкой. HTTPS - это обычный HTTP, работающий через шифрованные транспортные механизмы SSL и TLS. Это позволяет обеспечить защиту от атак, основанных на прослушивании сетевого соединения (снифферских атак и атак типа man-in-the-middle), при условии использования шифрующих средств и подтвержденной надежности сертификата сервера.

Запрос к API состоит из следующих элементов:

- основной URL запроса: https://integrationapi.net/rest/
- ресурс (например: /Sms/SendByTimeZone)
- параметры GET или POST-запроса (в кодировке UTF-8)

Скачать пример решений

1.2 Аутентификация

Сервис создает идентификатор сессии в системе после прохождения аутентификации данных, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/user/sessionid?login=<Login>&password=<Password>
```

Параметры запроса представляют собой последовательность пар вида **{имя параметра}={значение параметра}**, разделенных символом амперсанда (&). Content-Type для параметров запроса: application/x-www-form-urlencoded Это формат для кодирования пар «ключ-значение» с возможностью дублирования ключей. Каждая пара ключ-значение отделяется символом «&», ключ отделен от значения символом «=». При этом пробелы должны заменяться на знак «+», а затем, используя URL-кодирование, могут быть заменены на буквенно-цифровые символы. Например:

```
login: Jonathan  password: a+b==13%!
```

Должно быть закодировано как:

```
login=Jonathan&password=a%2Bb%3D%3D13%25!
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/user/sessionid?login=test&password=11111
```

Табл. 1. Параметры GET-запроса для аутентификации

Параметр	Тип данных	Описание
Login	String	Логин,полученный при регистрации
Password	String	Пароль, соответствующий логину

В случае успешной аутентификации предоставленных данных от сервиса отправки SMS-сообщений будет получен ответ со следующими параметрами:

```
HTTP status code: 200 OK (статус «OperationComplete»);
Cache-Control: private (указание на то, что ответ разрешается сохранять только в закрытом кэше, т.
↪е. только для текущего Пользователя);
Connection: Keep-Alive (наименование заголовка соединения, которое не надо обновлять в кэше);
Content-Type: application/json; charset=utf-8 (фактически значение вернется в виде строки в␣
↪кавычках (не в виде JSON) и кодировке utf-8);
‘‘Идентификатор сессии (GUID)’’
```

Ниже приведен пример ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
"Z5CYSZEKDL1DPICU37WEHQVOYKPOT1GSLHX1"
```

В случае возникновения исключительной ситуации в ходе обработки запроса или ошибки аутентификации Сервис возвращает Код ошибки (см. Табл. 18) в виде JSON следующего формата:

```
{
    Code: <Код ошибки>,
    Desc: <’’Текст ошибки’’>
}
```

Например, при ошибке авторизации:

```
{
    Code: 4,
    Desc: "Invalid user login or password"
}
```

Полученный идентификатор сессии действителен в течение 120 минут.

1.3 Получение баланса авторизованного пользователя

Протокол HTTP не имеет состояний. Это означает, что веб-сервер обрабатывает каждый HTTP-запрос со стороны внешнего приложения или сайта независимо и не сохраняет значения переменных, использованных в предшествующих запросах. Поэтому при выполнении запроса на получение баланса пользователя также необходимо передавать данные, полученные при авторизации этого пользователя. Сервис возвращает значение баланса авторизованного пользователя в соответствии со значениями параметров, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/User/Balance?SessionID=<Идентификатор сессии>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/User/Balance?SessionID=Z5CYSZEKDL1DPICU37WEHQVOYKPOT1GSLHX1
```

Табл. 2. Параметры GET-запроса баланса

Параметр	Тип данных	Описание
SessionID	String	Идентификатор сессии, полученный при аутентификации

Сервис проверяет валидность полученного SessionID (проверяет актуальность и наличие в системе). В случае успеха сервис авторизует пользователя и в ответе передает баланс пользователя со следующими параметрами:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Баланс пользователя>
```

Ниже приведен пример ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
```

В случае возникновения исключительной ситуации при обработке запроса или ошибки аутентификации сервис возвращает код ошибки (см. Табл. 18) в виде JSON следующего формата:

```
{
    Code: <Код ошибки>,
    Desc: <''Текст ошибки''>
}
```

Например, при ошибке аутентификации идентификатора сессии:

```
{
    Code: 4,
    Desc: "SessionID expired"
}
```

1.4 Отправка SMS-сообщений

1.4.1 Отправка SMS-сообщения на один номер без учета часового пояса получателя

Сервис инициирует отправку SMS-сообщения в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/Sms/Send?SessionID=<Идентификатор сессии>&SourceAddress=<Адрес␣
↪отправителя>&    DestinationAddress=<Номер получателя>&Data=<Текст сообщения>&Validity=<Время␣
↪жизни сообщения>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/Sms/Send?SessionId=C619DF83829F4C3094CB54F4D62878786B5B&
↪DestinationAddress=79161002030&SourceAddress=DEVINO&Data=test&Validity=0
```

Табл. 3. Параметры запроса на отправку SMS-сообщения

Параметр	Тип данных	Описание
Обязательные параметры		
SessionID	String	Идентификатор сессии, полученный при аутентификации (36 символов)
DestinationAddress	String	Номер получателя сообщения, в международном формате: код страны и код сети плюс номер телефона. Пример: 79031234567; +79031234567; 89031234567
Data	String	Текст сообщения, не более 2000 символов
SourceAddress	String	Адрес отправителя, не более 11 латинских символов или 15 цифр
Необязательные параметры		
SendDate	DateTime	Дата и время отправки (пример 2011-01-28T16:00:00). Если в запросе передается этот параметр, то сообщение будет отправлено только при наступлении полученных даты и времени без учета текущего часового пояса получателя. Сообщение отправится при наступлении переданного времени в часовом поясе: GMT+00:00. Если не требуется отложенная отправка, то передавать данный параметр не нужно.
Validity	Int	Время жизни сообщения (в минутах)

Перед отправкой SMS-сервис выполняет проверку запроса:

- наличие обязательных параметров;
- валидность сессии Пользователя (аутентификацию и определение, не истекло ли его время жизни SessionID);
- баланс пользователя на отправку SMS (достаточность средств на балансе определяется тарифом текущего пользователя на отправку SMS для мобильного оператора указанного в запросе номера);
- валидность указанного в запросе номера;
- валидность адреса отправителя;
- длина сообщения.

Если все проверки пройдены успешно, сервис отправляет сообщение в SMS-центр и возвращает идентификатор отправленного сообщения с параметрами:

Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["579700854169272358"]
```

В случае превышения длины отправляемого сообщения (70 символов на кириллице или 160 символов на латинице) сервис возвращает ответ в виде последовательности идентификаторов сообщений. Например:

```
["579700854169272358","579700854169272359"]
```

В случае непрохождения других проверок сервис возвращает код ошибки (см. Табл. 20) в виде JSON следующего формата:

```
{
        Code: <Код ошибки>,
        Desc: <'Текст ошибки'>
}
```

Например:

```
{
        Code: 6,
        Desc: "Invalid source address"
}
```

1.4.2 Отправка SMS-сообщения на один номер с учетом часового пояса получателя:

Сервис инициирует отправку SMS-сообщения в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/Sms/SendByTimeZone?SessionID=<Идентификатор сессии>&SourceAddress=
↪<Адрес отправителя>&DestinationAddress=<Номер получателя>&Data=<Текст сообщения>&Validity=<Время␣
↪жизни сообщения>&SendDate=<Дата отправки сообщения>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/Sms/SendByTimeZone?SessionId=Z5CYSZEKDL1DPICU37WEHQVOYKPOT1GSLHX1&
↪SourceAddress=TESTSMS&DestinationAddress=79001234567&Data=testdata&sendDate=2011-01-28T16:00:00&
↪validity=10
```

Табл. 4. Параметры POST-запроса на отправку SMS-сообщения с учетом часового пояса

Параметр	Тип данных	Описание
Обязательные параметры		
SessionID	String	Идентификатор сессии, полученный при аутентификации (36 символов)
DestinationAddress	String	Номер получателя сообщения, в международном формате: код страны и код сети плюс номер телефона. Пример: 79031234567; +79031234567; 89031234567.
Data	String	Текст сообщения (не более 2000 символов)
SourceAddress	String	Адрес отправителя (не более 11 латинских символов или 15 цифр)
SendDate	DateTime	Дата и время отправки (пример 2011-01-28T16:00:00). Если в запросе передается этот параметр, то сообщение будет отправлено только при наступлении полученных даты и времени с учетом текущего часового пояса получателя.
Необязательные параметры		
Validity	Int	Время жизни сообщения (в минутах)

Перед отправкой SMS-сервис выполняет проверку запроса:

- наличие обязательных параметров;
- валидность сессии Пользователя (аутентификацию и определение, не истекло ли его время жизни SessionID);
- баланс пользователя на отправку SMS (достаточность средств на балансе определяется тарифом текущего пользователя на отправку SMS для мобильного оператора указанного в запросе номера);
- валидность указанного в запросе номера;
- валидность адреса отправителя;
- длина сообщения.

Если все проверки пройдены успешно, сервис отправляет сообщение в SMS-центр и возвращает идентификатор отправленного сообщения с параметрами: Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["579700854169272359"]
```

В случае превышения длины отправляемого сообщения (70 символов на кириллице или 160 символов на латинице) сервис возвращает ответ в виде последовательности идентификаторов сообщений. Например:

```
["579700854169272358","579700854169272359"]
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["579700854169272358","579700854169272359"]
```

В случае непрохождения других проверок сервис возвращает код ошибки (см. Табл. 12) в виде JSON следующего формата:

```
{
        Code: <Код ошибки>,
        Desc: <'Текст ошибки'>
}
```

Например:

```
{
        Code: 6,
        Desc: "Invalid source address"
}
```

1.4.3 Отправка SMS-сообщения на несколько номеров без учета часового пояса получателя:

Сервис инициирует отправку SMS-сообщения на несколько номеров в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/Sms/SendBulk?SessionID=<Идентификатор сессии>&SourceAddress=<Адрес␣
↪отправителя>&DestinationAddresses=<Номер(а) получателя>&Data=<Текст сообщения>&Validity=<Время␣
↪жизни сообщения>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/Sms/SendBulk?SessionID=Z5CYSZEKDL1DPICU37WEHQVOYKPOT1GSLHX1&
↪SourceAddress=TESTSMS&DestinationAddresses=79001234567&Data=testdata&Validity=10&
↪DestinationAddresses=79160000000&data=testdata&sendDate=2011-01-28T16:00:00&validity=10
```

Табл. 5. Параметры POST-запроса на отправку SMS-сообщения на несколько номеров

Параметр	Тип данных	Описание
Обязательные параметры		
SessionID	String	Идентификатор сессии, полученный при аутентификации (36 символов)
DestinationAddresses	String	Номер получателя сообщения, в международном формате: код страны и код сети плюс номер телефона. Пример: 79031234567; +79031234567; 89031234567.
Data	String	Текст сообщения (не более 2000 символов)
SourceAddress	String	Адрес отправителя (не более 11 латинских символов или 15 цифр)
Необязательные параметры		
Validity	Int	Время жизни сообщения (в минутах)
SendDate	DateTime	Дата и время отправки (пример 2010-0601T19:14:00). Если не требуется отложенная отправка, то передавать данный параметр не нужно.

Перед отправкой SMS-сервис выполняет проверку запроса:

- наличие обязательных параметров;
- валидность сессии пользователя (аутентификацию и определение, не истекло ли его время жизни SessionID);
- баланс пользователя на отправку SMS (достаточность средств на балансе определяется тарифом текущего пользователя на отправку SMS для мобильного оператора указанного в запросе номера);
- валидность указанного в запросе номера;
- валидность адреса отправителя;
- длина сообщения.

Если все проверки пройдены успешно, сервис отправляет сообщение в SMS-центр и возвращает идентификатор отправленного сообщения с параметрами:

Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["579700854169272358"]
```

В случае превышения длины отправляемого сообщения (70 символов на кириллице или 160 символов на латинице) сервис возвращает ответ в виде последовательности идентификаторов сообщений. Для нескольких сообщений идентификаторы сегментов будут расположены последовательно – сначала последовательно все сегменты одного сообщения, затем – все сегменты другого. Например:

```
["579700854169272358","579700854169272359","579700854169272360","579700854169272361"]
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["579700854169272358","579700854169272359","579700854169272360","579700854169272361"]
```

В случае непрохождения других проверок сервис возвращает код ошибки (см. Табл. 12) в виде JSON следующего формата:

```
{
        Code: <Код ошибки>,
        Desc: <'Текст ошибки'>
}
```

Например:

```
{
        Code: 6,
        Desc: "Invalid source address"
}
```

1.4.4 Отправка SMS-сообщения на несколько номеров с учетом часового пояса получателя:

Сервис инициирует отправку SMS-сообщения в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/Sms/SendByTimeZoneToAddresses?SessionID=<Идентификатор сессии>&
↪SourceAddress=<Адрес отправителя>&DestinationAddresses=<Номер(а) получателя>&Data=<Текст␣
↪сообщения>&Validity=<Время жизни сообщения>&SendDate=<Дата отправки сообщения>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/Sms/SendByTimeZoneToAddresses?
↪SessionID=Z5CYSZEKDL1DPICU37WEHQVOYKPOT1GSLHX1&SourceAddress=TESTSMS&
↪DestinationAddresses=79001234567&Data=testdata&Validity=10&DestinationAddresses=79160000000&
↪data=testdata&SendDate=2011-01-28T16:00:00&Validity=10
```

Табл. 6. Параметры POST-запроса на отправку SMS-сообщения с учетом часового пояса

Параметр	Тип данных	Описание
Обязательные параметры		
SessionID	String	Идентификатор сессии, полученный при аутентификации (36 символов)
DestinationAddresses	String	Номер получателя сообщения, в международном формате: код страны и код сети плюс номер телефона. Пример: 79031234567; +79031234567; 89031234567.
Data	String	Текст сообщения (не более 2000 символов)
SourceAddress	String	Адрес отправителя (не более 11 латинских символов или 15 цифр)
SendDate	DateTime	Дата и время отправки (пример 2011-01-28T16:00:00). Если в запросе передается этот параметр, то сообщение будет отправлено только при наступлении полученных даты и времени с учетом текущего часового пояса получателя. Если не требуется отложенная отправка, то передавать данный параметр не нужно.
Необязательные параметры		
Validity	Int	Время жизни сообщения (в минутах)

Перед отправкой SMS-сервис выполняет проверку запроса:

- наличие обязательных параметров;
- валидность сессии Пользователя (аутентификацию и определение, не истекло ли его время жизни SessionID);
- баланс пользователя на отправку SMS (достаточность средств на балансе определяется тарифом текущего пользователя на отправку SMS для мобильного оператора указанного в запросе номера);
- валидность указанного в запросе номеров;
- валидность адреса отправителя;
- длина сообщения.

Если все проверки пройдены успешно, сервис отправляет сообщение в SMS-центр и возвращает идентификатор отправленного сообщения с параметрами: Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
```

(continues on next page)

(продолжение с предыдущей страницы)

```
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["579700854169272359"]
```

В случае превышения длины отправляемого сообщения (70 символов на кириллице или 160 символов на латинице) сервис возвращает ответ в виде последовательности идентификаторов сообщений. Для нескольких сообщений идентификаторы сегментов будут расположены последовательно – сначала последовательно все сегменты одного сообщения, затем – все сегменты другого. Например:

```
["579700854169272358","579700854169272359","579700854169272360","579700854169272361"]
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["579700854169272358","579700854169272359","579700854169272360","579700854169272361"]
```

В случае непрохождения других проверок сервис возвращает код ошибки (см. Табл. 18) в виде JSON следующего формата:

```
{
        Code: <Код ошибки>,
        Desc: <'Текст ошибки'>
}
```

Например:

```
{
        Code: 6,
        Desc: "Invalid source address"
}
```

1.5 Получение статуса отправленного SMS-сообщения

Предупреждение: Внимание! В случае, если сообщение было отправлено более 48 часов назад, то статус сообщения будет «255». (см. Табл. 18. Статусы SMS)

Сервис возвращает статус отправленного sms-сообщения в соответствии со значениями параметров, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/Sms/State?sessionId=<Идентификатор сессии>&messageId=
↪<Идентификатор сообщения>
```

Ниже приведен пример запроса для односегментного сообщения (длина которого не превышает 70 символов на кириллице или 160 символов на латинице):

```
https://integrationapi.net/rest/Sms/State?sessionId=Z5CYSZEKDL1DPICU37WEHQVOYKPOT1GSLHX1&
↪messageId=579700854169272358
```

Для многосегментных сообщений (длина сообщения превышает 70 символов на кириллице и 160 на латинице) запрос должен формироваться для каждого сегмента сообщения, например:

```
https://integrationapi.net/rest/Sms/State?sessionID=1AED345F65DD4C27BD37A17970C427FAE991&
↪messageID=SAR-W+84333
```

Табл. 7. Параметры GET-запроса статуса отправленного сообщения (сегмента сообщения)

Параметр	Тип данных	Описание
SessionID	String	Идентификатор сессии, полученный при аутентификации (36 символов)
messageId	String	Идентификатор сообщения (сегмента сообщения). Для одного запроса будет выполнен возврат статуса только одного сообщения (сегмента сообщения).

После получения запроса сервис проверяет валидность идентификатора сессии и наличие отправленного сообщения (сегмента сообщения) с присланным идентификатором. Если все проверки пройдены успешно, то сервис вернет статус отправленного sms-сообщения в json-формате со следующими параметрами:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
{"State":<Код статуса сообщения>,
"CreationDateUtc":<Дата создания>,
"SubmittedDateUtc":<Дата отправки сообщения>,
"ReportedDateUtc":<Дата доставки сообщения>,
"TimeStampUtc":"<Дата и время получения отчета>",
"StateDescription":"<Описание статуса>",
"Price":<Стоимость>}
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
{"State":255,"CreationDateUtc":null,"SubmittedDateUtc":null,"ReportedDateU tc":null,"TimeStampUtc":
↪"\/Date(-
62135596800000)\/","StateDescription":"Неизвестный","Price":null}
```

Если какая-нлибо проверка неуспешна, сервис возвращает код ошибки (см. Табл. 12) в виде JSON следующего формата:

```
{
        Code: <Код ошибки>,
        Desc: <'Текст ошибки'>
}
```

Например:

```
{
        Code: 1,
        Desc: "MessageID can not be null or empty Parameter name: messageId"
}
```

Табл. 8. Параметры ответа на запрос статуса сообщения

Наименование поля	Описание
State	Статус сообщения (см. Табл. 13)
TimeStampUtc	Дата и время получения отчета (Гринвич GMT00:00)
StateDescription	Описание статуса
CreationDateUtc	Дата создания
SubmittedDateUtc	Дата отправки
ReportedDateUtc	Дата доставки
Price	Цена за сообщение

1.6 Получение SMS-сообщений за период

Сервис возвращает входящие sms-сообщения за период в соответствии со значениями параметров, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/Sms/In?sessionId=<Идентификатор сессии>&minDateUTC=<Дата и время␣
↪начала периода>&maxDateUTC=<Дата и время окончания периода>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/Sms/In?sessionId=Z5CYSZEKDL1DPICU37WEHQVOYKPOT1GSLHX1&
↪minDateUTC=2011-01-01T00:00:00&maxDateUTC=2011-01-11T00:00:00
```

Табл. 9. Параметры GET-запроса на получение сообщений за период

Параметр	Тип данных	Описание
SessionID	String	Идентификатор сессии, полученный при аутентификации (36 символов)
maxDateUTC	DateTime	Дата и время окончания периода, за который происходит выборка входящих сообщений (например, 2010-06-02T19:14:00).
minDateUTC	DateTime	Дата и время начала периода, за который происходит выборка входящих сообщений (например, 2010-06-01T19:14:00).

Перед получением входящих сообщений, сервис проверяет запрос на:

- Наличие обязательных параметров;
- Валидность SessionID;
- Значение minDateUTC - разница между текущей датой и minDateUTC не может быть больше одного года.
- Значение maxDateUTC и minDateUTC - maxDateUTC должно быть больше чем minDateUTC

Если все проверки пройдены успешно, то сервис вернет перечень сообщений и их параметров за период в json-файла следующего формата:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
[{"Data":<Текст сообщения>,
"SourceAddress":<Адрес отправителя>,
"DestinationAddress":<Номер получателя>,
"ID":<Идентификатор сообщения>,
"CreatedDateUtc":<Дата создания>}]
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
[{"Data":"test1",
"SourceAddress":"79260000000",
"DestinationAddress":"79160000000",
"ID":539187174,
"CreatedDateUtc":"\/Date(1294045911213)\/"},
{"Data":"test2",
"SourceAddress":"79260000001",
"DestinationAddress":"79160000000",
"ID":539187214,
"CreatedDateUtc":"\/Date(1294045911353)\/"}]
```

Если какая-либо проверка неуспешна, сервис возвращает код ошибки (см. Табл. 12) в виде JSON следующего формата:

```
{
        Code: <Код ошибки>,
        Desc: <'Текст ошибки'>
}
```

Например:

```
{
Code: 9,
Desc: "The parameters dictionary contains a null entry for parameter
'maxDateUtc' of non-nullable type 'DateTime' for method
'System.Web.Mvc.ActionResult In(System.String, DateTime, DateTime)' in
'RestService.Controllers.SmsController'. An optional parameter must be a reference type, a␣
↪nullable type, or be declared as an optional parameter.  Parameter name: parameters"
}
```

1.7 Получение статистики по SMS-рассылкам

Сервис возвращает статистику по SMS-рассылкам за период в соответствии со значениями параметров, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/Sms/Statistics?sessionId=<Идентификатор сессии>&startDateTime=
↪<Дата и время начала периода >&endDateTime=<Дата и время конца периода>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/Sms/Statistics?sessionId=FBHKZT9TBBTUWYUR1PYUTYRAGRLUUGOR8A8Z&
↪startDateTime=2012-01-18%2000:00:00&endDateTime=2012-01-18T23:59:00
```

Табл. 10. Параметры GET-запроса на формирование статистики за период

Параметр	Тип данных	Описание
Обязательные параметры		
SessionID	String	Идентификатор сессии (36 символов)
startDateTime	DateTime	Дата и время начала периода, за который необходимо получить статистику, например 2012-01-18T00:00:00.
endDateTime	DateTime	Дата и время конца периода, за который необходимо получить статистику, например 2012-01-18T23:59:00.

После получения запроса сервис проверяет валидность присланного идентификатора сессии, корректность дат начала/окончания формирования статистики, диапазон дат (не более 3 месяцев). Если все проверки пройдены успешно, сервис возвращает статистику по sms-сообщениям в json-формате со следующими параметрами:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
{"Sent":<Отправлено>,
"Delivered":<Доставлено>,
"Errors":<С ошибками>,
"InProcess":<В процессе>,
"Expired":<С истекшим сроком доставки>,
"Rejected":<Отмененные>,
"Total":<Всего>,
"TotalWithErrors":<Всего с ошибками>,
"DeliveryRatio":<Успешно доставлено>}
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
{"Sent":9,
"Delivered":0,
"Errors":0,
"InProcess":7780,
"Expired":0,
"Rejected":56876,
"Total":64665,
"TotalWithErrors":64665,
"DeliveryRatio":0}
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает код ошибки (см. Табл. 12) в виде JSON следующего формата:

```
{
        Code: <Код ошибки>,
        Desc: <'Текст ошибки'>
}
```

Например:

```
{
Code: 2,
Desc: "Нельзя указывать диапазон дат более 90 дней."
}
```

1.8 Отправка Viber-сообщений

Предупреждение: Внимание! Для корректной работы переотправки необходимо запросить имя отправителя для SMS, идентичное имени отправителя Viber.

1.8.1 Отправка Viber-сообщения на один номер без учета часового пояса получателя

Сервис инициирует отправку Viber-сообщения в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/Viber/Send?SessionID=<Идентификатор сессии>&SourceAddress=<Адрес␣
↪отправителя>&DestinationAddress=<Номер получателя>&Data=<Текст сообщения>&Validity=<Время жизни␣
↪сообщения>&Optional=<Доп.Параметр>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/Viber/Send?SessionId=C619DF83829F4C3094CB54F4D62878786B5B&
↪SourceAddress=DTSMS&DestinationAddress=79001234567&Data=testdata&Validity=86400&Optional=123456
```

Табл. 11. Параметры запроса на отправку Viber-сообщения

Параметр	Тип данных	Описание
Обязательные параметры		
SessionID	String	Идентификатор сессии, полученный при аутентификации (36 символов)
DestinationAddress	String	Номер получателя сообщения, в международном формате: код страны + код сети + номер телефона. Пример: 79031234567; +79031234567
Data	String	Текст сообщения, сообщение не должно быть длиннее 1000 символов. Строки разделяются через символ новой строки %0A.
SourceAddress	String	Адрес отправителя сообщения. До 11 латинских или цифровых символов.
Необязательные параметры		
Optional	String	Дополнительный параметр
Validity	Int	Время жизни сообщения (мин, от 1 до 1440)

Перед отправкой Viber-сообщения сервис проверяет запрос на:

- Наличие обязательных параметров;
- Достаточно ли баланса пользователя на отправку Viber-сообщения;
- Валидность указанного в запросе номера;
- Валидность адреса отправителя;

- Длину сообщения.

Если все проверки пройдены успешно, то сервис отправит сообщение и вернет идентификатор отправленного сообщения со следующими параметрами:

Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
 ["GW0261BBD6B3"]
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает Код ошибки (см.Табл. 20) в виде JSON следующего формата:

```
{
Code: <Код ошибки>
Desc: <'Текст ошибки'>
}
```

Например:

```
{
Code: 1
Desc: "error-address-format"
}
```

1.8.2 Отправка Viber-сообщения на несколько номеров без учета часового пояса получателя

Сервис инициирует отправку Viber-сообщения на несколько номеров в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/Viber/SendBulk?SessionID=<Идентификатор сессии>&SourceAddress=
↪<Адрес отправителя>&DestinationAddresses=<Номер(а) получателя>&Data=<Текст сообщения>&Validity=
↪<Время жизни сообщения>&Optionals=<Доп. параметр(ы)>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/Viber/SendBulk?SessionId=C619DF83829F4C3094CB54F4D62878786B5B&
↪SourceAddress=TESTSMS&DestinationAddresses=79001234567&DestinationAddresses=79059999999&
↪Data=testdata&Validity=86400&Optionals=123456&Optionals=789012
```

Табл. 12. Параметры POST-запроса на отправку Viber-сообщения на несколько номеров

Параметр	Тип данных	Описание
Обязательные параметры		
SessionID	String	Идентификатор сессии, полученный при аутентификации (36 символов)
DestinationAddresses	String	Номер получателя сообщения, в международном формате: код страны + код сети + номер телефона. Пример: 79031234567; +79031234567
Data	String	Текст сообщения, сообщение не должно быть длиннее 1000 символов. Строки разделяются через символ новой строки %0A.
SourceAddress	String	Адрес отправителя сообщения. До 11 латинских или цифровых символов.
Необязательные параметры		
Optionals	String	Дополнительный параметр(или параметры в случае нескольких получателей)
Validity	Int	Время жизни сообщения (мин, от 1 до 1440)

Перед отправкой Viber Сервис проверяет запрос на:

- Наличие обязательных параметров;
- Достаточно ли баланса пользователя на отправку Viber;
- Валидность указанных в запросе номеров (если хоть один номер не проходит валидацию, то сообщения не отправляются);
- Валидность адреса отправителя;
- Длину сообщения.

Если все проверки пройдены успешно, то сервис отправит сообщения и вернет идентификаторы отправленных сообщений со следующими параметрами:

Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["GW0261BBD6B3"]
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает Код ошибки (см. Табл. 20) в виде JSON следующего формата:

```
{
Code: <Код ошибки>
Desc: <'Текст ошибки'>
}
```

Например:

```
{
Code: 1
Desc: "error-address-format"
}
```

1.8.3 Отправка Viber-сообщения на один номер с переотправкой по SMS

Сервис инициирует отправку Viber-сообщения в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/Viber/SendWithResend?SessionID=<Идентификатор сессии>&
↪SourceAddress=<Адрес отправителя>&DestinationAddress=<Номер получателя>&Data=<Текст сообщения>&
↪Validity=<Время жизни сообщения>&Optional=<Доп. Параметр>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/Viber/SendWithResend?
↪SessionId=Z5CYSZEKDL1DPICU37WEHQVOYKPOT1GSLHX1&SourceAddress=DTSMS&
↪DestinationAddress=79001234567&Data=testdata&Validity=86400&Optional=123456
```

Табл. 13. Параметры POST-запроса на отправку Viber-сообщения с переотправкой по SMS

Параметр	Тип данных	Описание
Обязательные параметры		
SessionID	String	Идентификатор сессии, полученный при аутентификации (36 символов)
DestinationAddress	String	Номер получателя сообщения, в международном формате: код страны + код сети + номер телефона. Пример: 79031234567; +79031234567
Data	String	Текст сообщения, сообщение не должно быть длиннее 1000 символов. Строки разделяются через символ новой строки %0A.
SourceAddress	String	Адрес отправителя сообщения. До 11 латинских или цифровых символов. Для корректной работы переотправки адрес отправителя SMS должен быть идентичен используемому адресу отправителя Viber
Необязательные параметры		
Optional	String	Дополнительный параметр
Validity	Int	Время жизни сообщения (мин, от 1 до 1440)

Перед отправкой Viber Сервис проверяет запрос на:

- Наличие обязательных параметров;
- Достаточно ли Баланса Пользователя на отправку Viber;
- Валидность указанных в запросе номеров (если хоть один номер не проходит валидацию, то сообщения не отправляются);
- Валидность адреса отправителя;
- Длину сообщения.

Если все проверки пройдены успешно, то сервис отправит сообщение и вернет идентификатор отправленного сообщения со следующими параметрами:

Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["GW0261BBD6B3"]
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает Код ошибки (см. Табл. 20) в виде JSON следующего формата:

```
{
Code: <Код ошибки>
Desc: <'Текст ошибки'>
}
```

Например:

```
{
Code: 1
Desc: "error-address-format"
}
```

1.8.4 Отправка Viber-сообщения на несколько номеров с переотправкой по SMS

Сервис инициирует отправку Viber-сообщения на несколько номеров в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/Viber/SendWithResendBulk?SessionID=<Идентификатор сессии>&
↪SourceAddress=<Адрес отправителя>&DestinationAddresses=<Номер(а) получателя>&Data=<Текст␣
↪сообщения>&Validity=<Время жизни сообщения>&Optionals=<Доп. параметр(ы)>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/Viber/SendWithResendBulk?
↪SessionID=Z5CYSZEKDL1DPICU37WEHQVOYKPOT1GSLHX1&SourceAddress=TESTSMS&
↪DestinationAddresses=79001234567&DestinationAddresses=79059999999&Data=testdata&Validity=86400&
↪Optionals=123456&Optionals=789012
```

Табл. 14. Параметры POST-запроса на отправку Viber-сообщения с переотправкой по SMS

Параметр	Тип данных	Описание
Обязательные параметры		
SessionID	String	Идентификатор сессии, полученный при аутентификации (36 символов)
DestinationAddress	String	Номер получателя сообщения, в международном формате: код страны + код сети + номер телефона. Пример: 79031234567; +79031234567
Data	String	Текст сообщения, сообщение не должно быть длиннее 1000 символов. Строки разделяются через символ новой строки %0A.
SourceAddress	String	Адрес отправителя сообщения. До 11 латинских или цифровых символов. Для корректной работы переотправки адрес отправителя SMS должен быть идентичен используемому адресу отправителя Viber
Необязательные параметры		
Optionals	String	Дополнительный параметр(или параметры в случае нескольких получателей)
Validity	Int	Время жизни сообщения (мин, от 1 до 1440)

Перед отправкой Viber Сервис проверяет запрос на:

- Наличие обязательных параметров;
- Валидность сессии Пользователя (аутентификацию и определение, не истекло ли его время жизни SessionID);
- Достаточно ли Баланса Пользователя на отправку Viber;
- Валидность указанных в запросе номеров (если хоть один номер не проходит валидацию, то сообщения не отправляются);
- Валидность адреса отправителя;
- Длину сообщения.

Если все проверки пройдены успешно, то сервис отправит сообщения и вернет идентификаторы отправленных сообщений со следующими параметрами:

Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["GW0261BBD6B3"]
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает Код ошибки (см. Табл. 20) в виде JSON следующего формата:

```
{
Code: <Код ошибки>
Desc: <'Текст ошибки'>
}
```

Например:

```
{
Code: 1
Desc: "error-address-format"
}
```

1.9 Получение статуса отправленного Viber-сообщения

Сервис возвращает статус отправленного viber-сообщения в соответствии со значениями параметров, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/Viber/State?sessionId=<Идентификатор сессии>&messageId=
↪<Идентификатор сообщения>
```

Табл. 15. Параметры GET-запроса статуса отправленного сообщения

Параметр	Тип данных	Описание
sessionId	String	Идентификатор сессии, полученный при аутентификации (36 символов)
messageId	String	Идентификатор сообщения

После получения запроса сервис проверяет валидность идентификатора сессии и наличие отправленного сообщения с присланным идентификатором. Если все проверки пройдены успешно, то сервис вернет статус отправленного viber-сообщения в json-формате со следующими параметрами:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
{
"State":<Код статуса сообщения>,
"CreationDateUtc":<Дата создания>,
"SubmittedDateUtc":<Дата отправки сообщения>,
"ReportedDateUtc":<Дата доставки сообщения>,
"TimeStampUtc":"<Дата и время получения отчета>",
"StateDescription":"<Описание статуса>",
"Price":<Стоимость>,
"ResentSms":[
    {
        "State":<Код статуса переотправленного смс-сообщения>,
        "CreationDateUtc":<Дата создания переотправленного смс-сообщения>,
        "SubmittedDateUtc":<Дата отправки переотправленного смс-сообщения>,
        "ReportedDateUtc":<Дата доставки переотправленного смс-сообщения>,
        "TimeStampUtc":"<Дата и время получения отчета по переотправленному смс-сообщению>",
        "StateDescription":"<Описание статуса переотправленного смс-сообщения>",
        "Price":<Стоимость переотправленного смс-сообщения>,
        "Id":<Идентификатор переотправленного смс-сообщения>
    }
]}
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает Код ошибки (см. Табл. 20) в виде JSON следующего формата:

```
{
Code: <Код ошибки>
Desc: <’’Текст ошибки’’>
}
```

Например:

```
{
Code: 1
Desc: "MessageID can not be null or empty Parameter name: messageId"
}
```

Табл. 16. Параметры ответа на запрос статуса сообщения

Наименование поля	Описание
State	Статус сообщения
TimeStampUtc	Дата и время получения отчета (Гринвич GMT00:00)
StateDescription	Описание статуса
CreationDateUtc	Дата создания
SubmittedDateUtc	Дата отправки
ReportedDateUtc	Дата доставки
Price	Цена за сообщение
ResentSms	Данные о sms-сообщениях, которые были отправлены в рамках переотправки текущего viber-сообщения

1.10 Коды ошибок. Статусы SMS и Viber

Табл. 17. Коды ошибок

REST error code	HTTP status code	Описание
	200	Operation complete
1	400	Argument cannot be null or empty
2	400	Invalid argument
3	400	Invalid session id
4	401	Unauthorized access
5	403	Not enough credits
6	400	Invalid operation
7	403	Forbidden
8	500	Gateway error
9	500	Internal server error

Табл. 18. Статусы SMS

State	Описание
-1	Отправлено (передано в мобильную сеть)
-2	В очереди
47	Удалено
-98	Остановлено
0	Доставлено абоненту
10	Неверно введен адрес отправителя
11	Неверно введен адрес получателя
41	Недопустимый адрес получателя
42	Отклонено смс-центром
46	Просрочено (истек срок жизни сообщения)
48	Отклонено Платформой
69	Отклонено
99	Неизвестный
255	статус: *сообщение еще не успело попасть в БД, *сообщение старше 48 часов.

Табл. 19. Статусы Viber

State	Описание
0	Отправляется
1	Отправлено
2	Доставлено (не прочитано)
3	Доставлено (прочитано)
4	Не доставлено
5	Ошибка
6	Неизвестно
7	Просрочено
8	Переход по ссылке

Табл. 20. Коды возврата обработки сообщения в рамках запроса (Viber-сообщения)

Код	Описание
error-address-format	неправильный формат номера абонента
error-address-not-specified	номер абонента не указан
error-address-unknown	отправка на номерную емкость, к которой относится номер абонента не разрешена клиенту в конфигурации платформы провайдера
error-content-not-specified	содержимое сообщения не указано
error-subject-format	неправильный формат подписи
error-subject-not-specified	подпись не указана
error-subject-unknown	указанная подпись не разрешена клиенту в конфигурации платформы провайдера
error-system	системная ошибка
error-validity-period-seconds-format	неправильно указано значение времени ожидания доставки
ok	исходящее сообщение успешно принято на отправку

Глава 2

HTTP API Без SessionID

2.1 Обзор API

Предоставляемый API сервис отправки SMS-сообщений позволяет осуществить:

- Получение баланса авторизованного пользователя
- Отправку SMS-сообщения на один номер без учета часового пояса получателя
- Отправку SMS-сообщения на один номер с учетом часового пояса получателя
- Отправку SMS-сообщения на несколько номеров без учета часового пояса получателя
- Отправку SMS-сообщения на несколько номеров с учётом часового пояса получателя
- Получение статуса отправленного SMS-сообщения
- Получение SMS-сообщений за период
- Получение статистики по SMS-рассылкам
- Отправку Viber-сообщения на один номер без учета часового пояса получателя
- Отправку Viber-сообщения на несколько номеров без учета часового пояса получателя
- Отправка Viber-сообщения на один номер с переотправкой по SMS
- Отправка Viber-сообщения на несколько номеров с переотправкой по SMS
- Получение статуса отправленного Viber-сообщения

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

API Сервиса отправки SMS сообщений организовано в соответствии с принципами REST, что позволяет обмениваться HTTPS URL–encoded запросами. HTTPS - это обычный HTTP, работающий через

шифрованные транспортные механизмы SSL и TLS. Это позволяет обеспечить защиту от атак, основанных на прослушивании сетевого соединения: снифферских атак и атак типа man-in-the-middle при условии, что будут использоваться шифрующие средства и сертификат сервера проверен и ему доверяют.

Запрос к API состоит из следующих элементов:

- Основного URL запроса: https://integrationapi.net/rest/v2
- Ресурса, например: /Sms/SendByTimeZone
- Параметров GET или POST-запроса (в кодировке UTF-8)

2.2 Получение баланса авторизованного пользователя

Сервис возвращает значение баланса авторизованного пользователя в соответствии со значениями параметров, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/v2/User/Balance?Login=<Логин>&Password=<Пароль>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/v2/User/Balance?Login=test_login&Password=test123
```

Табл. 1. Параметры GET-запроса баланса

Параметр	Тип данных	Описание
Login	String	Логин,полученный при регистрации
Password	String	Пароль, соответствующий логину

Сервис проверяет валидность Логина/Пароля и в случае успеха авторизует пользователя и в ответе присылает баланс пользователя со следующими параметрами:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Баланс пользователя>
```

Ниже приведен пример ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
20015.3
```

В случае возникновения исключительной ситуации во время обработки запроса или ошибки аутентификации, сервис возвращает код ошибки (см. Табл. 16) в виде JSON следующего формата:

```
{
  Code: <Код ошибки>,
  Desc: <''Текст ошибки''>
}
```

Например, при ошибке авторизации:

```
{
  Code: 4,
  Desc: "Invalid user login or password"
}
```

2.3 Отправка SMS-сообщений

2.3.1 Отправка SMS-сообщения на один номер без учета часового пояса получателя

Сервис инициирует отправку SMS-сообщения в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Sms/Send?Login=<Логин>&Password=<Пароль>&SourceAddress=<Адрес␣
↪отправителя>&DestinationAddress=<Номер получателя>&Data=<Текст сообщения>&Validity=<Время жизни␣
↪сообщения>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/v2/Sms/Send?Login=test_login&Password=test_password&
↪DestinationAddress=79161002030&SourceAddress=DEVINO&Data=test&Validity=0
```

Табл. 2. Параметры запроса на отправку SMS-сообщения

Параметр	Тип данных	Описание
Обязательные параметры		
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
DestinationAddress	String	Номер получателя сообщения, в международном формате: код страны код сети + номер телефона. Пример: 79031234567; +79031234567; 89031234567
Data	String	Текст сообщения, сообщение не должно быть длиннее 2000 символов
SourceAddress	String	Адрес отправителя сообщения. До 11 латинских символов или до 15 цифровых.
Необязательные параметры		
SendDateTime	DateTime	Дата и время отправки (пример 2011-01-28T16:00:00). Если в запросе передается этот параметр, то сообщение будет отправлено только при наступлении полученных даты и времени без учета текущего часового пояса получателя. Сообщение отправится при наступлении переданного времени в часовом поясе: GMT+00:00. Если не требуется отложенная отправка, то передавать данный параметр не нужно.
Validity	Int	Время жизни сообщения (в минутах)

Перед отправкой SMS сервис проверяет запрос на:

- Наличие обязательных параметров;
- Валидность Логина/пароля;
- Достаточно ли Баланса Пользователя на отправку SMS. (Достаточность определяется на основании тарифа пользователя на отправку SMS для мобильного оператора указанного в запросе номера);

- Валидность указанного в запросе номера;
- Валидность адреса отправителя;
- Длину сообщения.

Если все проверки пройдены успешно, то сервис отправит сообщение в SMS-центр и вернет идентификатор отправленного сообщения со следующими параметрами: Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["GW0261BBD6B3"]
```

В случаях, когда длина отправляемого сообщения превышает 70 символов на кириллице или 160 символов на латинице, ответ от сервиса будет в виде последовательности идентификаторов сообщений, например:

```
["SAR-GW01+79160000000-5f3b1972-2-1","SAR-GW01+79160000000-5f3b1972-2-2"]
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает код ошибки (см. Табл. 16) в виде JSON следующего формата:

```
{
  Code: <Код ошибки>,
  Desc: <''Текст ошибки''>
}
```

Например:

```
{
  Code: 6,
  Desc: "Invalid source address"
}
```

2.3.2 Отправка SMS-сообщения на один номер с учетом часового пояса получателя:

Сервис инициирует отправку SMS-сообщения в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Sms/SendByTimeZone?Login=<Логин>&Password=<Пароль>&
↪SourceAddress=<Адрес отправителя>&DestinationAddress=<Номер получателя>&Data=<Текст сообщения>&
↪Validity=<Время жизни сообщения>&SendDate=<Дата отправки сообщения>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/v2/Sms/SendByTimeZone?Login=test_login&Password=test123&
↪SourceAddress=TESTSMS&DestinationAddress=79001234567&Data=testdata&Validity=10&sendDate=2011-01-
↪28T16:00:00
```

Табл. 3. Параметры POST-запроса на отправку SMS-сообщения с учетом часового пояса

Параметр	Тип данных	Описание
Обязательные параметры		
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
DestinationAddress	String	Номер получателя сообщения, в международном формате: код страны и код сети плюс номер телефона. Пример: 79031234567; +79031234567; 89031234567.
Data	String	Текст сообщения, сообщение не должно быть длиннее 2000 символов
SourceAddress	String	Адрес отправителя сообщения. До 11 латинских символов или до 15 цифровых.
SendDate	DateTime	Дата и время отправки (пример 2011-01-28T16:00:00). Если в запросе передается этот параметр, то сообщение будет отправлено только при наступлении полученных даты и времени с учетом текущего часового пояса получателя. Если не требуется отложенная отправка, то передавать данный параметр не нужно.
Необязательные параметры		
Validity	Int	Время жизни сообщения (в минутах)

Перед отправкой SMS сервис проверяет запрос на:

- Наличие обязательных параметров;
- Валидность Логина/пароля;
- Достаточно ли баланса пользователя на отправку SMS. (Достаточность определяется на основании тарифа пользователя на отправку SMS для мобильного оператора указанного в запросе номера);
- Валидность указанного в запросе номера;
- Валидность адреса отправителя;
- Длину сообщения.

Если все проверки пройдены успешно, то сервис отправит сообщение в SMS-центр и вернет идентификатор отправленного сообщения со следующими параметрами: Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["GW0261BBD6B3"]
```

В случаях, когда длина отправляемого сообщения превышает 70 символов на кириллице или 160 символов на латинице, ответ от сервиса будет в виде последовательности идентификаторов сообщений:

```
["SAR-GW01+79160000000-5f3b1972-2-1","SAR-GW01+79160000000-5f3b1972-2-2"]
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["SAR-GW01+79160000000-5f3b1972-2-1","SAR-GW01+79160000000-5f3b1972-2-2"]
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает код ошибки (см. Табл. 16) в виде JSON следующего формата:

```
{
  Code: <Код ошибки>,
  Desc: <''Текст ошибки''>
}
```

Например:

```
{
  Code: 6,
  Desc: "Invalid source address"
}
```

2.3.3 Отправка SMS-сообщения на несколько номеров без учета часового пояса получателя

Сервис инициирует отправку SMS-сообщения на несколько номеров в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Sms/SendBulk?Login=<Логин>&Password=<Пароль>&SourceAddress=
↪<Адрес отправителя>&DestinationAddresses=<Номер(а) получателя(ей)>&Data=<Текст сообщения>&
↪Validity=<Время жизни сообщения>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/v2/Sms/SendBulk?Login=test_login&Password=test123&
↪SourceAddress=TESTSMS&DestinationAddresses=79001234567&DestinationAddresses= 79059999999&
↪Data=testdata&Validity=10
```

Табл. 4. Параметры POST-запроса на отправку SMS-сообщения на несколько номеров

Параметр	Тип данных	Описание
Обязательные параметры		
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
DestinationAddress	String	Номер получателя сообщения, в международном формате: код страны и код сети плюс номер телефона. Максимальное количество получателей сообщения не должно превышать 2999. Пример: 79031234567; +79031234567; 89031234567.
Data	String	Текст сообщения, сообщение не должно быть длиннее 2000 символов
SourceAddress	String	Адрес отправителя сообщения. До 11 латинских символов или до 15 цифровых.
Необязательные параметры		
Validity	Int	Время жизни сообщения (в минутах)
SendDate	DateTime	Дата и время отправки (пример 2010-0601T19:14:00). Если не требуется отложенная отправка, то передавать данный параметр не нужно.

Перед отправкой SMS Сервис проверяет запрос на:

- Наличие обязательных параметров;
- Валидность Логина/пароля;
- Достаточно ли Баланса Пользователя на отправку SMS. (Достаточность определяется на основании тарифа пользователя на отправку SMS для мобильного оператора указанного в запросе номера);
- Валидность указанных в запросе номеров (если хоть один номер не проходит валидацию, то сообщения не отправляются);
- Валидность адреса отправителя;
- Длину сообщения.

Если все проверки пройдены успешно, то сервис отправит сообщения в SMS-центр и вернет идентификаторы отправленных сообщений со следующими параметрами: Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["GW0261BBD6B3"]
```

В случаях, когда длина отправляемого сообщения превышает 70 символов на кириллице или 160 символов на латинице, ответ от сервиса будет в виде последовательно расположенных идентификаторов сегментов сообщения. Для нескольких сообщений идентификаторы сегментов будут расположены последовательно – сначала последовательно все сегменты одного сообщения, затем – все сегменты другого, например:

- [«SAR-GW01+79160000000-5f3b1972-2-1»,»SAR-GW01+79160000000-5f3b1972-2-2»,
- [«SAR-GW01+79053500000-5d3b1972-2-1»,»SAR-GW01+79053500000-5d3b1972-2-2]

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["SAR-GW01+79160000000-5f3b1972-2-1","SAR-GW01+79160000000-5f3b1972-2-2",
["SAR-GW01+79053500000-5f3d1972-2-1","SAR-GW01+79053500000-5f3d1972-2-2]
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает код ошибки (см. Табл. 16) в виде JSON следующего формата:

```
{
  Code: <Код ошибки>,
  Desc: <''Текст ошибки''>
}
```

Например:

```
{
  Code: 6,
  Desc: "Invalid source address"
}
```

2.3.4 Отправка SMS-сообщения на несколько номеров с учетом часового пояса получателя:

Сервис инициирует отправку SMS-сообщения в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Sms/SendByTimeZoneToAddresses?Login=<Логин>&Password=<Пароль>&
↪SourceAddress=<Адрес отправителя>&DestinationAddresses=<Номер(а) получателя(ей)>&Data=<Текст␣
↪сообщения>&Validity=<Время жизни сообщения>&SendDate=<Дата отправки сообщения>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/v2/Sms/SendByTimeZoneToAddresses?Login=test_login&Password=test123&
↪SourceAddress=TESTSMS&&DestinationAddresses=79001234567&DestinationAddresses=79059999999&
↪Data=testdata&Validity=10&sendDate=2011-01-28T16:00:00
```

Табл. 5. Параметры POST-запроса на отправку SMS-сообщения с учетом часового пояса

Параметр	Тип данных	Описание
Обязательные параметры		
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
DestinationAddresses	String	Номер получателя сообщения, в международном формате: код страны и код сети плюс номер телефона. Пример: 79031234567; +79031234567; 89031234567.
Data	String	Текст сообщения (не более 2000 символов)
SourceAddress	String	Адрес отправителя (не более 11 латинских символов или 15 цифр)
SendDate	DateTime	Дата и время отправки (пример 2010-0601T19:14:00) в UTC. Если в запросе передается этот параметр, то сообщение будет отправлено только при наступлении полученных даты и времени с учетом текущего часового пояса получателя.
Необязательные параметры		
Validity	Int	Время жизни сообщения (в минутах)

Перед отправкой SMS-сервис выполняет проверку запроса:

- наличие обязательных параметров;
- валидность логина/пароля;
- баланс пользователя на отправку SMS (достаточность средств на балансе определяется тарифом текущего пользователя на отправку SMS для мобильного оператора указанного в запросе номера);
- валидность указанного в запросе номеров;
- валидность адреса отправителя;

- длина сообщения.

Если все проверки пройдены успешно, сервис отправляет сообщения в SMS-центр и возвращает идентификаторы отправленных сообщений с параметрами: Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["579700854169272359"]
```

В случаях, когда длина отправляемого сообщения превышает 70 символов на кириллице или 160 символов на латинице, ответ от сервиса будет в виде последовательно расположенных идентификаторов сегментов сообщения. Для нескольких сообщений идентификаторы сегментов будут расположены последовательно – сначала последовательно все сегменты одного сообщения, затем – все сегменты другого, например:

```
     [‘‘SAR-GW01+79160000000-5f3b1972-2-1’’,’’SAR-GW01+79160000000-5f3b1972-2-2’’,
‘‘SAR-GW01+79053500000-5d3b1972-2-1’’,’’SAR-GW01+79053500000-5d3b1972-2-2"]
```

Например:

```
     HTTP/1.1 200 OK
     Cache-Control: private
     Connection: Keep-Alive
     Content-Type: application/json; charset=utf-8
     ["SAR-GW01+79160000000-5f3b1972-2-1","SAR-GW01+79160000000-5f3b1972-2-2",
"SAR-GW01+79053500000-5f3d1972-2-1","SAR-GW01+79053500000-5f3d1972-2-2"]
```

В случае непрохождения других проверок сервис возвращает код ошибки (см. Табл. 16) в виде JSON следующего формата:

```
{
        Code: <Код ошибки>,
        Desc: <’’Текст ошибки’’>
}
```

Например:

```
{
        Code: 6,
        Desc: "Invalid source address"
}
```

2.4 Получение статуса отправленного SMS-сообщения

Предупреждение: Внимание! В случае, если сообщение было отправлено более 48 часов назад, то статус сообщения будет «255». (см. Табл. 18. Статусы SMS)

Сервис возвращает статус отправленного sms-сообщения в соответствии со значениями параметров, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Sms/State?
Login=<Логин>&
Password=<Пароль>&
messageId=<Идентификатор сообщения>
```

Ниже приведен пример запроса для односегментного сообщения (длина которого не превышает 70 символов на кириллице или 160 символов на латинице):

```
https://integrationapi.net/rest/v2/Sms/State?Login=test_login&Password=test123&
↪messageId=GW0261BA732
```

Для сообщений, длина которых превышает 70 символов на кириллице и 160 на латинице, запрос должен формироваться для каждого сегмента сообщений, например:

```
https://integrationapi.net/rest/v2/Sms/State?Login=test_login&Password=test123&messageID=SAR-
↪W+84333
```

Табл. 6. Параметры GET-запроса статуса отправленного сообщения (сегмента сообщения)

Параметр	Тип данных	Описание
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
messageId	String	Идентификатор сообщения (сегмента сообщения). Для одного запроса будет выполнен возврат статуса только одного сообщения (сегмента сообщения).

После получения запроса сервис проверит валидность логина/пароля и наличие отправленного сообщения (сегмента сообщения) с присланным идентификатором. Если все проверки пройдены успешно, то сервис вернет статус отправленного sms-сообщения в json формате со следующими параметрами:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
{"State":{Код статуса сообщения},
"CreationDateUtc":{Дата создания},
"SubmittedDateUtc":{Дата отправки сообщения},
"ReportedDateUtc":{Дата доставки сообщения},
"TimeStampUtc":"{Дата и время получения отчета}",
"StateDescription":"{Описание статуса}",
"Price":{Стоимость}
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
{"State":255,"CreationDateUtc":null,"SubmittedDateUtc":null,"ReportedDateU tc":null,"TimeStampUtc":
↪"\/Date(-
62135596800000)\/","StateDescription":"Неизвестный","Price":null}
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает код ошибки (см. Табл. 16) в виде JSON следующего формата:

```
{
  Code: <Код ошибки>,
  Desc: <’’Текст ошибки’’>
}
```

Например:

```
{
  Code: 1,
  Desc: "MessageID can not be null or empty Parameter name: messageId"
}
```

Табл. 7. Параметры ответа на запрос статуса сообщения

Наименование поля	Описание
State	Статус сообщения (см. Табл. 17)
TimeStampUtc	Дата и время получения отчета (Гринвич GMT00:00)
StateDescription	Описание статуса
CreationDateUtc	Дата создания
SubmittedDateUtc	Дата отправки
ReportedDateUtc	Дата доставки
Price	Цена за сообщение

2.5 Получение SMS-сообщений за период

Сервис возвращает входящие sms-сообщения за период в соответствии со значениями параметров, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Sms/In?
Login=<Логин>&
Password=<Пароль>&
minDateUTC=<Дата и время начала периода>&
maxDateUTC=<Дата и время окончания периода>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/v2/Sms/In?Login=test_login&Password=test123&minDateUTC=2011-01-
↪01T00:00:00&maxDateUTC=2011-01-11T00:00:00
```

Табл. 8. Параметры GET-запроса на получение сообщений за период

Параметр	Тип данных	Описание
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
maxDateUTC	DateTime	Дата и время окончания периода, за который происходит выборка входящих сообщений (например, 2010-06-02T19:14:00).
minDateUTC	DateTime	Дата и время начала периода, за который происходит выборка входящих сообщений (например, 2010-06-01T19:14:00).

Перед получением входящих сообщений, сервис проверяет запрос на:

- Наличие обязательных параметров;
- Валидность Логина/пароля;
- Значение minDateUTC - разница между текущей датой и minDateUTC не может быть больше одного года.
- Значение maxDateUTC и minDateUTC - maxDateUTC должно быть больше чем minDateUTC

Если все проверки пройдены успешно, то сервис вернет перечень сообщений и их параметров за период в json-файла следующего формата

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
[{"Data":{Текст сообщения},
"SourceAddress":{Адрес отправителя},
"DestinationAddress":{Номер получателя},
"ID":{Идентификатор сообщения},
"CreatedDateUtc":{Дата создания}]
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
[{"Data":"test1",
"SourceAddress":"79260000000",
"DestinationAddress":"79160000000",
"ID":539187174,
"CreatedDateUtc":"\/Date(1294045911213)\/"},
{"Data":"test2",
"SourceAddress":"79260000001",
"DestinationAddress":"79160000000",
"ID":539187214,
"CreatedDateUtc":"\/Date(1294045911353)\/"}]
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает код ошибки (см. Табл. 16) в виде JSON следующего формата:

```
{
  Code: <Код ошибки>,
  Desc: <''Текст ошибки''>
}
```

Например:

```
{
  Code: 9,
  Desc: "The parameters dictionary contains a null entry for parameter
  'maxDateUtc' of non-nullable type 'DateTime' for method
  'System.Web.Mvc.ActionResult In(System.String, DateTime, DateTime)' in
  'RestService.Controllers.SmsController'. An optional parameter must be a reference type, a␣
↪nullable type, or be declared as an optional parameter. Parameter name: parameters"
}
```

2.6 Получение статистики по SMS-рассылкам

Сервис возвращает статистику по SMS-рассылкам за период в соответствии со значениями параметров, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Sms/Statistics?
Login=<Логин>&
Password=<Пароль>&
startDateTime=<Дата и время начала периода>&
endDateTime=<Дата и время конца периода>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/v2/Sms/Statistics?Login=test_login&Password=test123&
↪startDateTime=2017-07-18T23:59:00&endDateTime=2017-08-25T23:59:00
```

Табл. 9. Параметры GET-запроса на формирование статистики за период

Параметр	Тип данных	Описание
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
startDateTime	DateTime	Дата и время конца периода, за который необходимо получить статистику, например 2017-07-18T23:59:00.
endDateTime	DateTime	Дата и время конца периода, за который необходимо получить статистику, например 2017-08-25T23:59:00.

После получения запроса сервис проверит валидность логина/пароля и дат начала/окончания формирования статистики (включая ограничение на то, что охватываемый диапазон должен не превышать 3 месяцев). Если все проверки пройдены успешно, то сервис вернет статистику по sms-сообщениям в json формате со следующими параметрами:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
{"Sent":{Отправлено},
"Delivered":{Доставлено},
"Errors":{С ошибками},
"InProcess":{В процессе},
"Expired":{С истекшим сроком доставки},
"Rejected":{Отмененные},
```

(continues on next page)

(продолжение с предыдущей страницы)

```
"Total":{Всего},
"TotalWithErrors":{Всего с ошибками},
"DeliveryRatio":{Успешно доставлено}
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
{"Sent":9,
"Delivered":0,
"Errors":0,
"InProcess":7780,
"Expired":0,
"Rejected":56876,
"Total":64665,
"TotalWithErrors":64665,
"DeliveryRatio":0}
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает код ошибки (см. Табл. 16) в виде JSON следующего формата:

```
{
  Code: <Код ошибки>,
  Desc: <'Текст ошибки'>
}
```

Например:

```
{
  Code: 2,
  Desc: "Нельзя указывать диапазон дат более 90 дней."
}
```

2.7 Отправка Viber-сообщений

Предупреждение: Внимание! Для корректной работы переотправки необходимо запросить имя отправителя для SMS, идентичное имени отправителя Viber.

2.7.1 Отправка Viber-сообщения на один номер без учета часового пояса получателя

Сервис инициирует отправку Viber-сообщения в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Viber/Send?Login=<Логин>&Password=<Пароль>&SourceAddress=<Адрес␣
↪отправителя>&DestinationAddress=<Номер получателя>&Data=<Текст сообщения>&Validity=<Время жизни␣
↪сообщения>&Optional=<Доп. Параметр>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/v2/Viber/Send?Login=Test&Password=Test&SourceAddress=DTSMS&
↪DestinationAddress=79001234567&Data=testdata&Validity=86400&Optional=123456
```

Табл. 10. Параметры запроса на отправку Viber-сообщения

Параметр	Тип данных	Описание
Обязательные параметры		
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
DestinationAddress	String	Номер получателя сообщения, в международном формате: код страны + код сети + номер телефона. Пример: 79031234567; +79031234567
Data	String	Текст сообщения, сообщение не должно быть длиннее 1000 символов. Строки разделяются через символ новой строки %0A.
SourceAddress	String	Адрес отправителя сообщения. До 11 латинских или цифровых символов.
Необязательные параметры		
Optional	String	Дополнительный параметр
Validity	Int	Время жизни сообщения (мин, от 1 до 1440)

Перед отправкой Viber-сообщения Сервис проверяет запрос на:

- Наличие обязательных параметров;
- Валидность Логина/пароля;
- Достаточно ли Баланса Пользователя на отправку Viber-сообщения;
- Валидность указанного в запросе номера;
- Валидность адреса отправителя;
- Длину сообщения.

Если все проверки пройдены успешно, то Сервис отправит сообщение и вернет идентификатор отправленного сообщения со следующими параметрами:

Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
 ["GW0261BBD6B3"]
```

Если какая-нибудь проверка не проходит успешно, то Сервис возвращает Код ошибки (см.Табл. 19) в виде JSON следующего формата:

```
{
Code: <Код ошибки>
Desc: <’’Текст ошибки’’>
}
```

Например:

```
{
Code: 1
Desc: "error-address-format"
}
```

2.7.2 Отправка Viber-сообщения на несколько номеров без учета часового пояса получателя

Сервис инициирует отправку Viber-сообщения на несколько номеров в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Viber/SendBulk?Login=<Логин>&Password=<Пароль>&SourceAddress=
↪<Адрес отправителя>&DestinationAddresses=<Номер(а) получателя>&Data=<Текст сообщения>&Validity=
↪<Время жизни сообщения>&Optional=<Доп. параметр(ы)>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/v2/Viber/SendBulk?Login=Test&Password=Test&SourceAddress=TESTSMS&
↪DestinationAddresses=79001234567&DestinationAddresses=79059999999&Data=testdata&Validity=86400&
↪Optionals=123456&Optionals=789012
```

Табл. 11. Параметры POST-запроса на отправку Viber-сообщения на несколько номеров

Параметр	Тип данных	Описание
Обязательные параметры		
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
DestinationAddresses	String	Номер получателя сообщения, в международном формате: код страны + код сети + номер телефона. Пример: 79031234567; +79031234567
Data	String	Текст сообщения, сообщение не должно быть длиннее 1000 символов. Строки разделяются через символ новой строки %0A.
SourceAddress	String	Адрес отправителя сообщения. До 11 латинских или цифровых символов.
Необязательные параметры		
Optionals	String	Дополнительный параметр(или параметры в случае нескольких получателей)
Validity	Int	Время жизни сообщения (мин, от 1 до 1440)

Перед отправкой Viber Сервис проверяет запрос на:

- Наличие обязательных параметров;
- Валидность Логина/Пароля;
- Достаточно ли Баланса Пользователя на отправку Viber;

- Валидность указанных в запросе номеров (если хоть один номер не проходит валидацию, то сообщения не отправляются);
- Валидность адреса отправителя;
- Длину сообщения.

Если все проверки пройдены успешно, то Сервис отправит сообщение и вернет идентификатор отправленного сообщения со следующими параметрами:

Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["GW0261BBD6B3"]
```

Если какая-нибудь проверка не проходит успешно, то Сервис возвращает Код ошибки (см. Табл. 19) в виде JSON следующего формата:

```
{
Code: <Код ошибки>
Desc: <'Текст ошибки'>
}
```

Например:

```
{
Code: 1
Desc: "error-address-format"
}
```

2.7.3 Отправка Viber-сообщения на один номер с переотправкой по SMS

Сервис инициирует отправку Viber-сообщения в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Viber/SendWithResend?Login=<Логин>&Password=<Пароль>&
↪SourceAddress=<Адрес отправителя>&DestinationAddress=<Номер получателя>&Data=<Текст сообщения>&
↪Validity=<Время жизни сообщения>&Optional=<Доп. Параметр>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/v2/Viber/SendWithResend?Login=Test&Password=Test&&
↪SourceAddress=DTSMS&DestinationAddress=79001234567&Data=testdata&Validity=86400&Optional=123456
```

Табл. 12. Параметры запроса на отправку Viber-сообщения

Параметр	Тип данных	Описание
Обязательные параметры		
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
DestinationAddress	String	Номер получателя сообщения, в международном формате: код страны + код сети + номер телефона. Пример: 79031234567; +79031234567
Data	String	Текст сообщения, сообщение не должно быть длиннее 1000 символов. Строки разделяются через символ новой строки %0A.
SourceAddress	String	Адрес отправителя сообщения. До 11 латинских или цифровых символов. Для корректной работы переотправки адрес отправителя SMS должен быть идентичен используемому адресу отправителя Viber
Необязательные параметры		
Optional	String	Дополнительный параметр
Validity	Int	Время жизни сообщения (мин, от 1 до 1440)

Перед отправкой Viber-сообщения Сервис проверяет запрос на:

- Наличие обязательных параметров;
- Валидность Логина/пароля;
- Достаточно ли Баланса Пользователя на отправку Viber-сообщения;
- Валидность указанного в запросе номера;
- Валидность адреса отправителя;
- Длину сообщения.

Если все проверки пройдены успешно, то Сервис отправит сообщение и вернет идентификатор отправленного сообщения со следующими параметрами:

Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
 ["GW0261BBD6B3"]
```

Если какая-нибудь проверка не проходит успешно, то Сервис возвращает Код ошибки (см.Табл. 19) в виде JSON следующего формата:

```
{
Code: <Код ошибки>
Desc: <'Текст ошибки'>
}
```

Например:

```
{
Code: 1
Desc: "error-address-format"
}
```

2.7.4 Отправка Viber-сообщения на несколько номеров с переотправкой по SMS

Сервис инициирует отправку Viber-сообщения на несколько номеров в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Viber/SendWithResendBulk?Login=<Логин>&Password=<Пароль>&
↪SourceAddress=<Адрес отправителя>&DestinationAddresses=<Номер(а) получателя>&Data=<Текст␣
↪сообщения>&Validity=<Время жизни сообщения>&Optionals=<Доп. параметр(ы)>
```

Ниже приведен пример запроса:

```
https://integrationapi.net/rest/v2/Viber/SendWithResendBulk?Login=Test&Password=Test&
↪SourceAddress=TESTSMS&DestinationAddresses=79001234567&DestinationAddresses=79059999999&
↪Data=testdata&Validity=86400&Optionals=123456&Optionals=789012
```

Табл. 13. Параметры POST-запроса на отправку Viber-сообщения на несколько номеров

Параметр	Тип данных	Описание
Обязательные параметры		
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
DestinationAddresses	String	Номер получателя сообщения, в международном формате: код страны + код сети + номер телефона. Пример: 79031234567; +79031234567
Data	String	Текст сообщения, сообщение не должно быть длиннее 1000 символов. Строки разделяются через символ новой строки %0A.
SourceAddress	String	Адрес отправителя сообщения. До 11 латинских или цифровых символов. Для корректной работы переотправки адрес отправителя SMS должен быть идентичен используемому адресу отправителя Viber
Необязательные параметры		
Optionals	String	Дополнительный параметр(или параметры в случае нескольких получателей)
Validity	Int	Время жизни сообщения (мин, от 1 до 1440)

Перед отправкой Viber Сервис проверяет запрос на:

- Наличие обязательных параметров;
- Валидность Логина/Пароля;
- Достаточно ли Баланса Пользователя на отправку Viber;
- Валидность указанных в запросе номеров (если хоть один номер не проходит валидацию, то сообщения не отправляются);
- Валидность адреса отправителя;
- Длину сообщения.

Если все проверки пройдены успешно, то Сервис отправит сообщение и вернет идентификатор отправленного сообщения со следующими параметрами:

Формат ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
<Идентификатор сообщения>
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["GW0261BBD6B3"]
```

Если какая-нибудь проверка не проходит успешно, то Сервис возвращает Код ошибки (см. Табл. 19) в виде JSON следующего формата:

```
{
Code: <Код ошибки>
Desc: <'Текст ошибки'>
}
```

Например:

```
{
Code: 1
Desc: "error-address-format"
}
```

2.8 Получение статуса отправленного Viber-сообщения

Сервис возвращает статус отправленного viber-сообщения в соответствии со значениями параметров, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Viber/State?Login=<Логин>&Password=<Пароль>&messageId=
↪<Идентификатор сообщения>
```

Табл. 14. Параметры GET-запроса статуса отправленного сообщения

Параметр	Тип данных	Описание
Login	String	Логин, полученный при регистрации
Password	String	Пароль, соответствующий логину
messageId	String	Идентификатор сообщения

После получения запроса сервис проверяет валидность связки логина и пароля и наличие отправленного сообщения с присланным идентификатором. Если все проверки пройдены успешно, то сервис вернет статус отправленного viber-сообщения в json-формате со следующими параметрами:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
{
"State":<Код статуса сообщения>,
"CreationDateUtc":<Дата создания>,
"SubmittedDateUtc":<Дата отправки сообщения>,
"ReportedDateUtc":<Дата доставки сообщения>,
"TimeStampUtc":"<Дата и время получения отчета>",
"StateDescription":"<Описание статуса>",
"Price":<Стоимость>,
"ResentSms":[
    {
        "State":<Код статуса переотправленного смс-сообщения>,
        "CreationDateUtc":<Дата создания переотправленного смс-сообщения>,
        "SubmittedDateUtc":<Дата отправки переотправленного смс-сообщения>,
        "ReportedDateUtc":<Дата доставки переотправленного смс-сообщения>,
        "TimeStampUtc":"<Дата и время получения отчета по переотправленному смс-сообщению>",
        "StateDescription":"<Описание статуса переотправленного смс-сообщения>",
        "Price":<Стоимость переотправленного смс-сообщения>,
        "Id":<Идентификатор переотправленного смс-сообщения>
    }
]}
```

Если какая-нибудь проверка не проходит успешно, то сервис возвращает Код ошибки (см. Табл. 19) в виде JSON следующего формата:

```
{
Code: <Код ошибки>
Desc: <’’Текст ошибки’’>
}
```

Например:

```
{
Code: 1
Desc: "MessageID can not be null or empty Parameter name: messageId"
}
```

Табл. 15. Параметры ответа на запрос статуса сообщения

Наименование поля	Описание
State	Статус сообщения
TimeStampUtc	Дата и время получения отчета (Гринвич GMT00:00)
StateDescription	Описание статуса
CreationDateUtc	Дата создания
SubmittedDateUtc	Дата отправки
ReportedDateUtc	Дата доставки
Price	Цена за сообщение
ResentSms	Данные о sms-сообщениях, которые были отправлены в рамках переотправки текущего viber-сообщения

2.9 Коды ошибок. Статусы SMS и Viber

Табл. 16. Коды ошибок

REST error code	HTTP status code	Описание
•	200	Operation complete
1	400	Argument cannot be null or empty
2	400	Invalid argument
4	401	Unauthorized access
5	403	Not enough credits
6	400	Invalid operation
7	403	Forbidden
8	500	Gateway error
9	500	Internal server error

Табл. 17. Статусы SMS

State	Описание
-1	Отправлено (передано в мобильную сеть)
-2	В очереди
47	Удалено
-98	Остановлено
0	Доставлено абоненту
10	Неверно введен адрес отправителя
11	Неверно введен адрес получателя
41	Недопустимый адрес получателя
42	Отклонено смс центром
46	Просрочено (истек срок жизни сообщения)
48	Отклонено Платформой
69	Отклонено
99	Неизвестный
255	По запросу возвращается этот статус, если сообщения еще не успело попасть в БД, либо сообщение старше 48 часов.

Табл. 18. Статусы viber-сообщений

State	Описание
0	Отправляется
1	Отправлено
2	Доставлено (не прочитано)
3	Доставлено (прочитано)
4	Не доставлено
5	Ошибка
6	Неизвестно
7	Просрочено
8	Переход по ссылке

Табл. 19. Коды возврата обработки сообщения в рамках запроса (Viber-сообщения)

Код	Описание
error-address-format	неправильный формат номера абонента
error-address-not-specified	номер абонента не указан
error-address-unknown	отправка на номерную емкость, к которой относится номер абонента не разрешена клиенту в конфигурации платформы провайдера
error-content-not-specified	содержимое сообщения не указано
error-subject-format	неправильный формат подписи
error-subject-not-specified	подпись не указана
error-subject-unknown	указанная подпись не разрешена клиенту в конфигурации платформы провайдера
error-system	системная ошибка
error-validity-period-seconds-format	неправильно указано значение времени ожидания доставки
ok	исходящее сообщение успешно принято на отправку

Глава 3

Протокол SMPP

SMPP — это протокол, описывающий взаимодействие клиента с SMS-сервером платформы, для передачи SMS и USSD сообщений.

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

3.1 Точка доступа

Взаимодействие клиента с SMS-сервером платформы осуществляется по адресам:

- smpp01.integrationapi.net (194.226.179.12) – основной адрес для рекламных рассылок.
- smpp02.integrationapi.net (194.226.179.13) – резервный адрес для рекламных рассылок.
- smpp03.integrationapi.net (194.226.179.10) – основной адрес для транзакционных рассылок.
- smpp04.integrationapi.net (194.226.179.11) – резервный адрес для транзакционных рассылок.

Проверка доступного баланса осуществляется через личный кабинет клиента.

3.2 Техническая часть

Настройка SMPP-клиента Для работы с платформой по протоколу SMPP необходимо произвести следующие настройки SMPP-клиента:

```
Host: 194.226.179.12
Port: 2775
```

Обязательные параметры:

```
System_ID: Логин*
Password: Пароль*
Interface_Version: 0x34
System_Type: NULL (Пустое значение)
Src_Addr_TON: 0x05
Src_Addr_NPI: 0x01
Dest_Addr_TON: 0x01
Dest_Addr_NPI: 0x01
```

Наименование поля описание:

```
System_ID Логин, присвоенный клиенту
Password Пароль, присвоенный клиенту
Interface_Version версия SMPP
System_Type Тип системы SMSC
Src_Addr_TON Тип адреса источника
Src_Addr_NPI Нумерация адреса источника
Dest_Addr_TON Тип адреса назначения
Dest_Addr_NPI Нумерация адреса назначения
```

3.3 Описание взаимодействия

- Протокол работы сервиса SMPP 3.4 (TCP/IP/SMPP).
- Сервис работает в асинхронном режиме. Размер окна определяется клиентом, количество сессий до 4-х.
- Каждая сессия может работать в любом из режимов: tx (transmitter - только отправка), rx (receiver - только прием отчетов), trx (transceiver - прием/передача в одной сессии).
- После обрыва сессии по SMPP и TCP/IP необходимо выдержать таймаут не менее 60 секунд, только после этого можно устанавливать TCP/IP сессию и отправлять PDU Bind_transceiver. В случае неудачной попытки соединения таймаут увеличивается до 90 секунд.
- При получении сообщения отправлять на каждый принятый Deliver_sm подтверждение Deliver_sm_resp (Data_sm_resp) всегда со статусом 0 (ok). Generic_nack не допускаются! В случае отправки будет произведен сброс сессии путем отправки пакета unbind
- Отправлять запрос Enquire_link следует только в случае отсутствия приёма любых SMPP PDU со стороны SMSC в течение 60 секунд. При отсутствии ответа от SMSC (Enquire_link_resp) следует закрывать SMPP сессию командой Unbind.
- Сообщения, состоящие более чем из одной части, должны разбиваться на стороне клиента (кроме отправки через Massage Payload) одним из методов отправки составных сообщений:

    1.Использование опционных параметров (SAR метод)

    2.Использование UDH

- Отправка через Message Payload - разбивка происходит на стороне СМС-центра, тарификация сообщений будет проводиться по сформированным сегментам. В данном случае отчет о доставке будет отправлен только на одну (первую) часть.
- UDH-заголовок должен занимать 6 байт.
- В одной части составного сообщения можно передавать 66-67 символов в кириллице и 150-153 в латинице (поле message length должно быть 132-134 байта).

- Если сервис использует более одного sys_id (несколько аккаунтов), то для корректной склейки все части разбитого сообщения должны отсылаться через один и тот же sys-id (через один и тот же аккаунт).
- При получении ошибки Invalid Destination Address сообщение необходимо удалить из своей очереди и больше не перепосылать.
- При получении ошибки Throttling error сообщение нужно вернуть в очередь, но необходимо выдержать таймаут на данном соединении = 1 сек.
- При получении ошибки Message Queue Full необходимо ставить сообщение, на которое вернулась данная ошибка, в конец очереди и сделать еще 3-5 попыток доставки этого сообщения, каждый раз возвращая это сообщение в конец очереди при получении той же ошибки. Рекомендуется применять прогрессивный метод обработки этой ошибки – при первом получении делать паузу перед отправкой в 5 сек, при второй – 15, третьей – 45 и т.д.
- Параметр validity period должен быть не менее 60 секунд. Возможны ограничения доставки сообщений с указанием меньшего периода.
- Платформа поддерживает запросы query_sm, скорость отправки запросов и окно устанавливаются клиентом

Глава 4

SMTP (EMail2SMS) for SMS

4.1 Общие положения

Отправка SMS-сообщений через Email2SMS-шлюз платформы осуществляется посредством создания и последующей передачи клиентом сообщений электронной почты (e-mail) на сервисный адрес платформы:

```
email2sms@integrationapi.net
```

4.2 Техническая часть

Тело (текст) сообщения электронной почты должно быть сформировано в соответствии приведенным ниже шаблоном:

```
==|| НАЧАЛО ТЕКСТА ||==

    UserLogin=<Логин>

    Password=<Пароль>

    SourceAddress=<Адрес отправителя>

    PhoneNumber=<Получатель>

    <Текст сообщения>

==|| КОНЕЦ ТЕКСТА ||==
```

Предупреждение: Внимание! Для сообщения электронной почты необходимо указать формат данных: plain text only и кодировку: windows-1251, в противном случае DEVINO не гарантирует корректность передаваемой информации.

Наименование поля	Описание
UserLogin	Логин, присвоенный клиенту
Password	Пароль, соответствующий логину
SourceAddress	Адрес отправителя, присвоенный Клиенту
PhoneNumber	Мобильный телефонный номер получателя SMS-сообщения в международном формате: код страны + код сети + номер телефона. Также возможно указание нескольких номеров получателей (до 20), номера разделяются запятой.
<Текст сообщения>	Текст SMS-сообщения

Пример сообщения с одним получателем:

```
UserLogin=7284_smuser777
Password=qwerty12
SourceAddress=My Bank
PhoneNumber=+79161234567
Summa scheta 1038.35 Summa k oplate:1038.26
==|| KOHEЦ TEKCTA ||==
```

Если необходимо отправить сообщение нескольким отправителям, то необходимо ставить «,» после номера и следующий начинать опять с «+».

Пример:

```
PhoneNumber=+79151234567,+79161234567,+79031234567
```

4.3 AddonToEmail2SMS

Дополнительная опция к Email2sms.Формат тела письма (в одну строку без переноса):

```
StartWithPoint;login;password;mobilenumber;sourceaddress;text of message;EndWithPoint
```

Пример:

```
StartWithPoint;Ваш логин;Ваш пароль;79001234567;TEL;Привет, мир!;EndWithPoint
```

Формат заголовка письма (если текст на латинице):

```
Content-Type: text/plain; charset=us-ascii
Content-Transfer-Encoding: 7bit
```

Формат заголовка письма (если текст на кириллице):

```
Content-Type: text/plain; charset=windows-1251
Content-Transfer-Encoding: quoted-printable
```

4.4 AddonToEmail2SMS_Base64

Отправка сообщений через сервис платформы в кодировке Base64 Тема письма (регистр символов не имеет значения) должна быть *MODEB64* Формат тела письма:

```
login;password;mobilenumber;sourceaddress;text of message
```

Пример:

```
Ваш логин;Ваш пароль;79001234567;TEL;Привет, мир!
```

Обращаем Ваше внимание:

1. Перед отправкой письма убедитесь, что в заголовке Content-Transfer-Encoding установлено base64
2. Номера телефонов можно указывать через запятую (до 1000 в одном письме),

пример:

```
Ваш логин;Ваш пароль;79001234567,79001234567;TEL;Привет мир!
```

Глава 5

Модуль интеграции с сервисом SMS через 1с

Для использования данного вида интеграции вам необходимо зарегистрироваться на серверной платформе кампании, либо через менеджера компании. При регистрации вы получите логин и пароль, а так же стандартное имя отправителя **DTSMS**, которое необходимо будет поменять, запросив другое в ЛК.

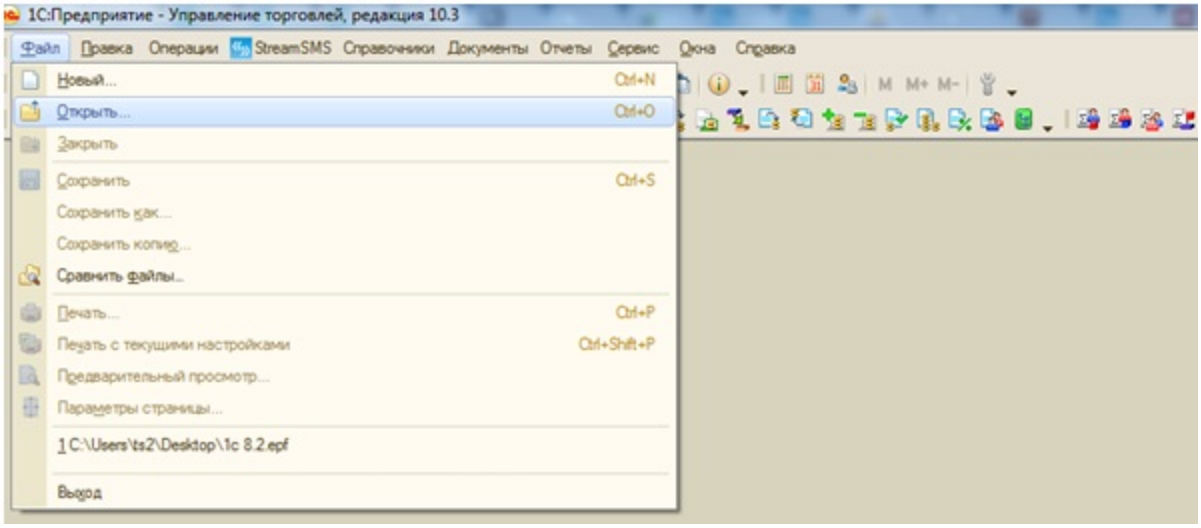
Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

Скачать модуль

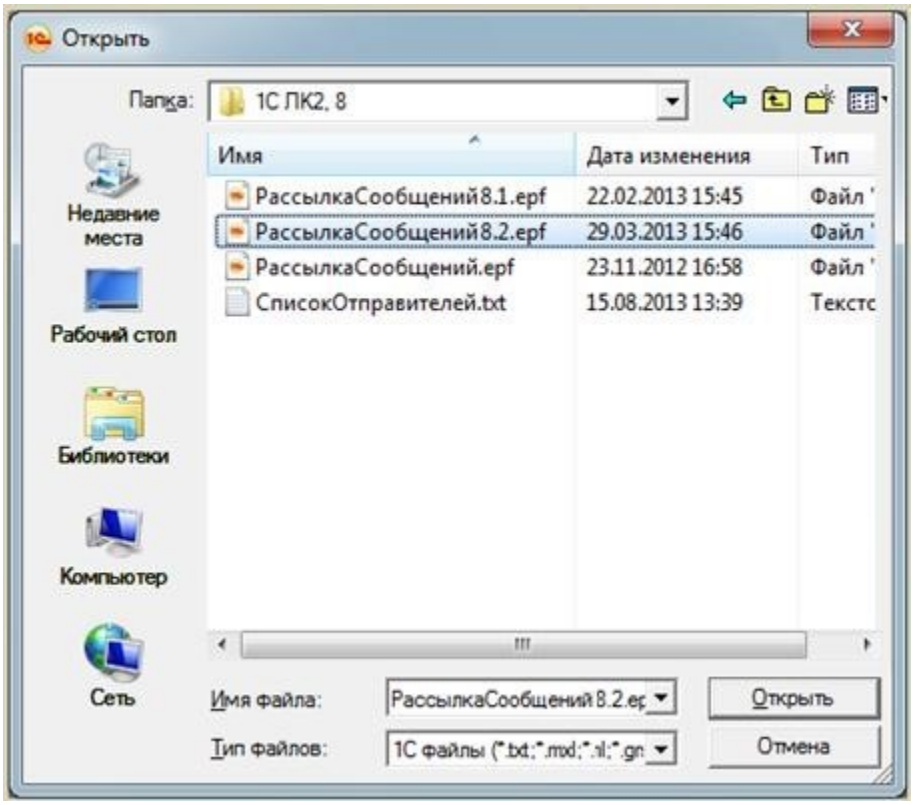
Модуль 1с работает только с версиями 8.1, 8.2, 8.3

5.1 Установка интеграционного модуля в 1С

Чтобы пользоваться интеграционным модулем (далее ИМ) для 1С, необходимо открыть его через 1С. Для этого во вкладке «Файл», необходимо выбрать пункт выпадающего меню «открыть».



Затем, выбрать путь до ИМ на вашем компьютере

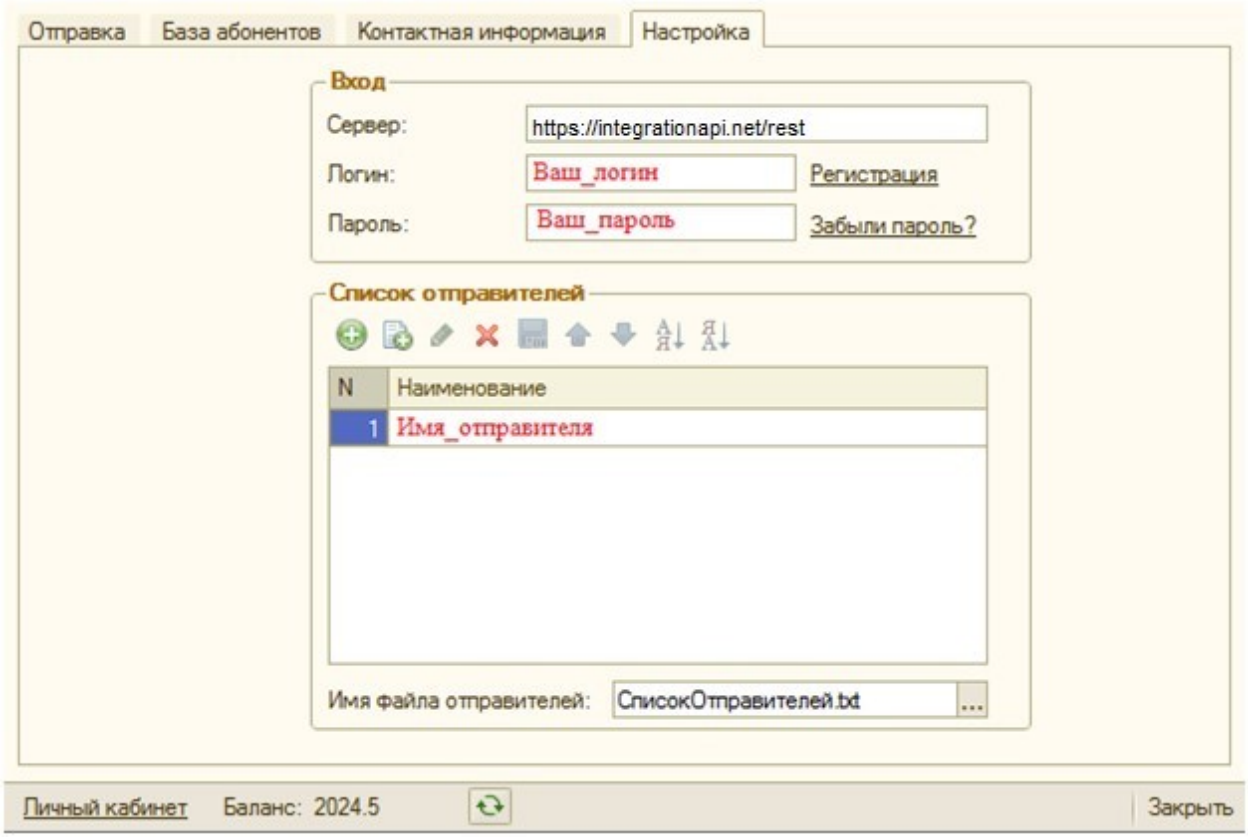


После чего, в 1С, откроется ИМ.

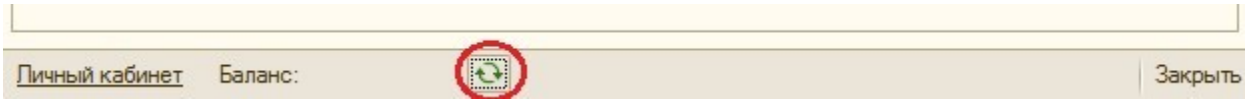
5.2 Авторизация в интеграционном модуле

Для того чтобы отправлять сообщения через 1С, необходимо во вкладке «Настройка» ввести ваши личные данные: *логин и пароль*, а так же имя отправителя (отличное от testsms), зарегистрированное в личном кабинете. Адрес сервера вписан в обработку изначально:

```
https://integrationapi.net/rest
```



Для проверки правильности ввода данных, нажмите кнопку обновления баланса, если логин с паролем введены правильно, то Вы увидите свой баланс.

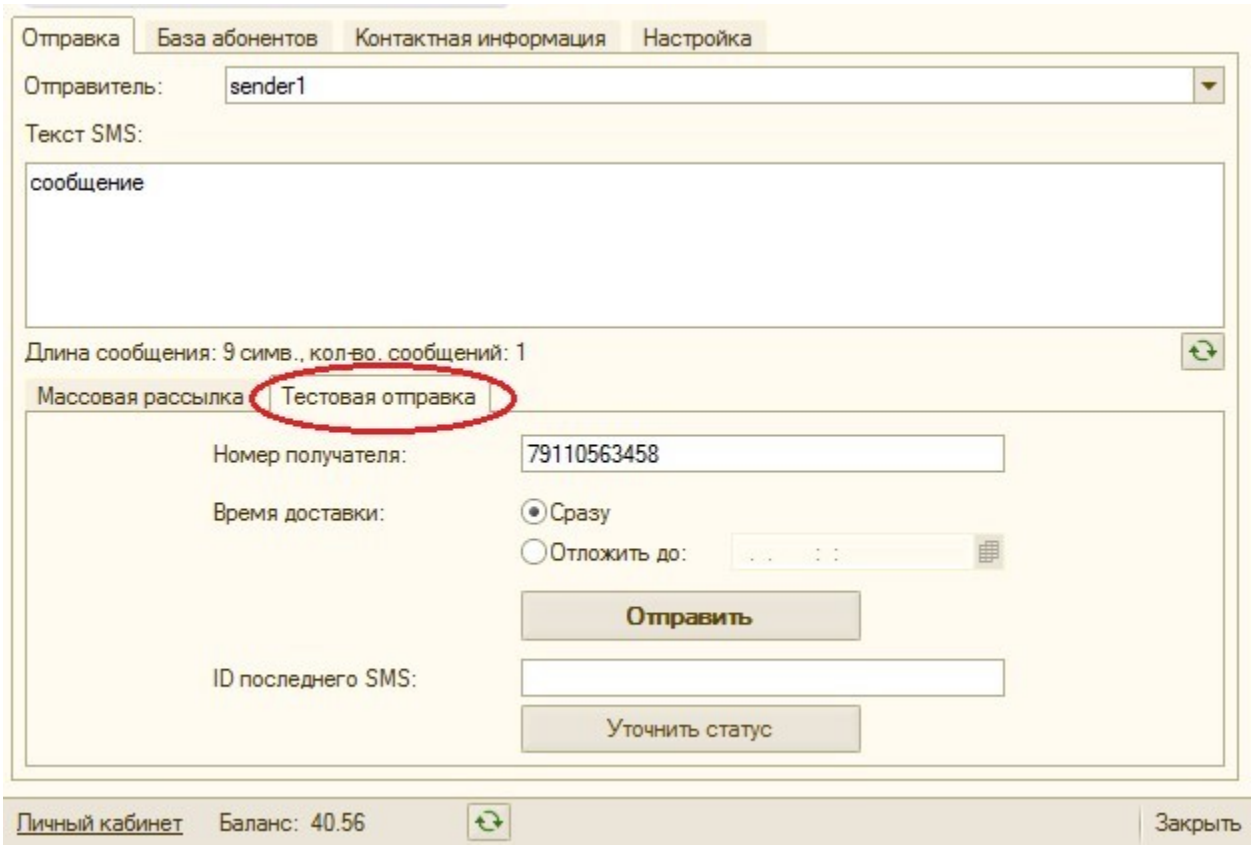


Если данные, для подключения введены неверно, то появится ошибка: Result: **Error_Invalid_Login.**

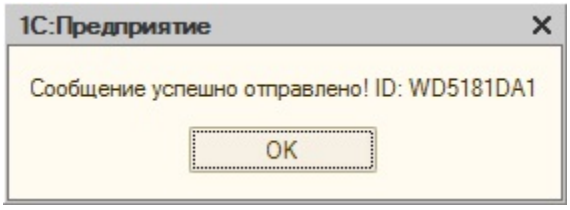
5.3 Единичная отправка сообщений

5.3.1 Мгновенная отправка единичного сообщения

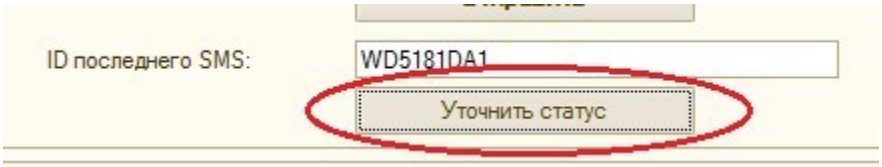
Для отправки одного сообщения следует выбрать вкладку «Тестовая отправка». В поле «Отправитель» выбрать имя отправителя, из списка, который вы вводили во вкладке «Настройки» (все имена отправителей должны быть зарегистрированы в личном кабинете). В поле текст SMS, введите текст отправляемого сообщения. Номер получателя – номер, на который вы хотите отправить SMS (в формате 79214563763).



После отправки сообщения (если все параметры введены правильно), появится диалоговое окно, с ID, отправленного сообщения, который запишется в поле «ID последнего SMS».



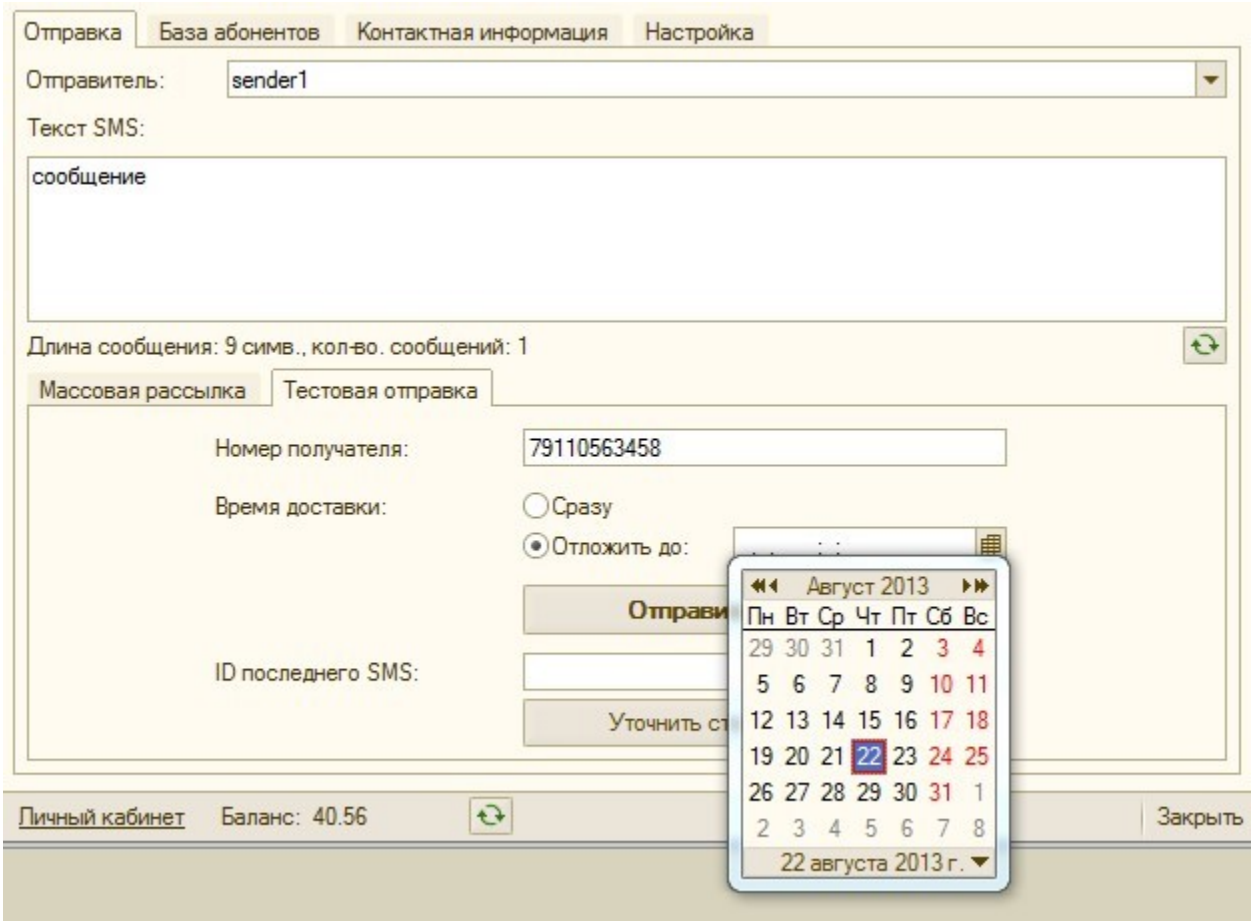
Чтобы узнать статус отправленного сообщения, следует нажать кнопку «Уточнить статус».



В появившемся окне, будет информация о времени отправки сообщения и его статусе (сообщения, ID, которого записано в поле «ID последнего SMS»).

5.3.2 Отправка отложенного по времени единичного сообщения

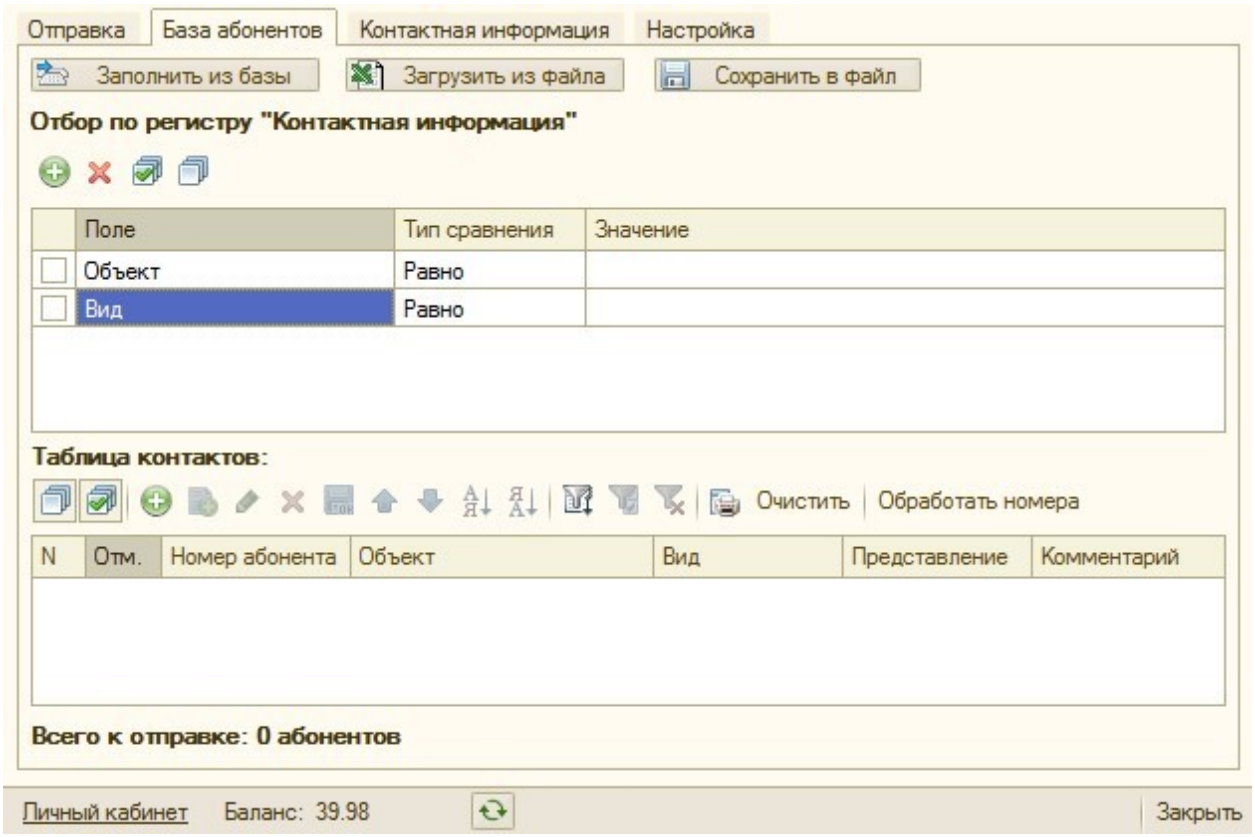
Чтобы задать конкретные дату и время отправки единичного сообщение, необходимо перед отправкой выбрать «Отложить до» и дату, время отправки сообщения.



5.4 Пакетная отправка сообщений

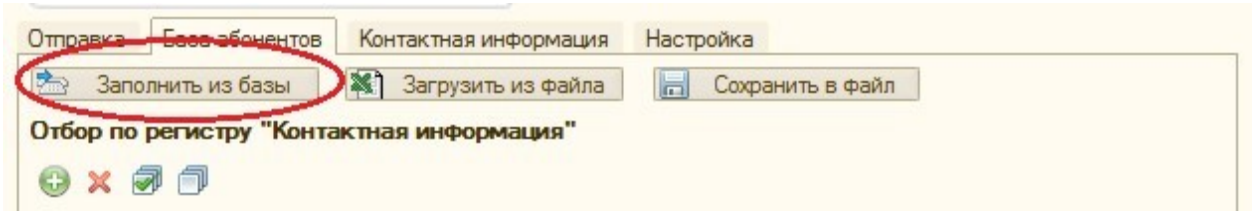
С помощью данного модуля для 1С, возможно осуществлять рассылку множеству получателей.

Создание списка получателей сообщений. Для массовой рассылки сообщений, необходимо во вкладке «База абонентов» добавить контакты для рассылки.

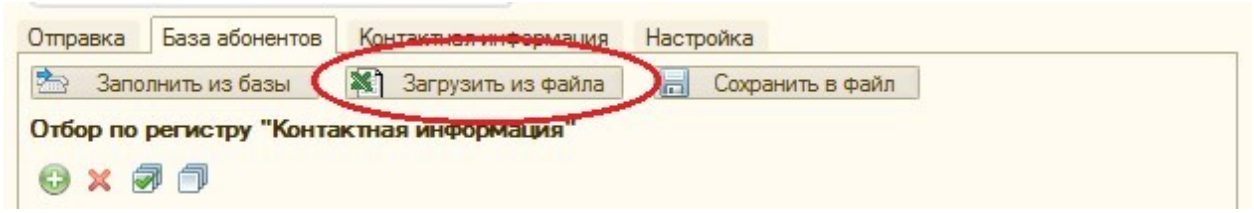


Это можно сделать разными способами:

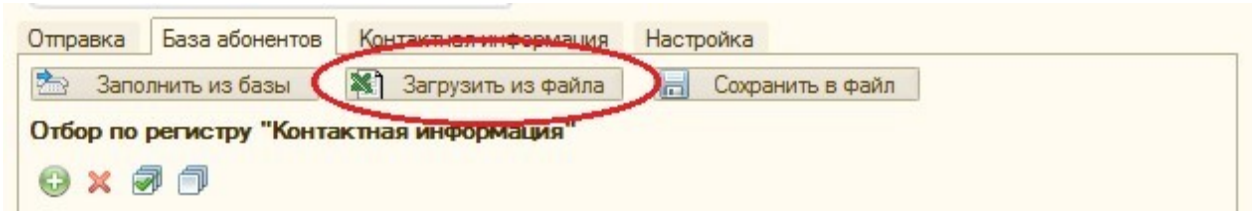
- Добавлять единичные контакты



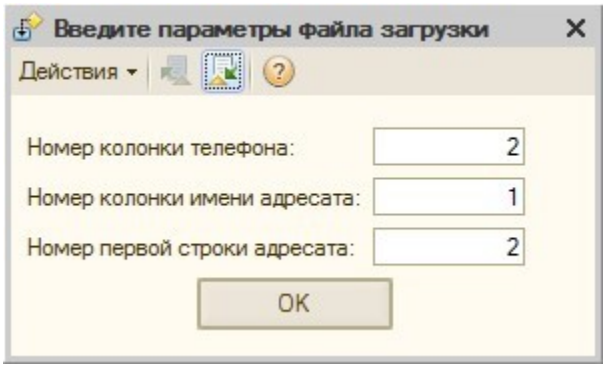
- Заполнять номерами из базы (добавить контакты из справочника контрагентов)



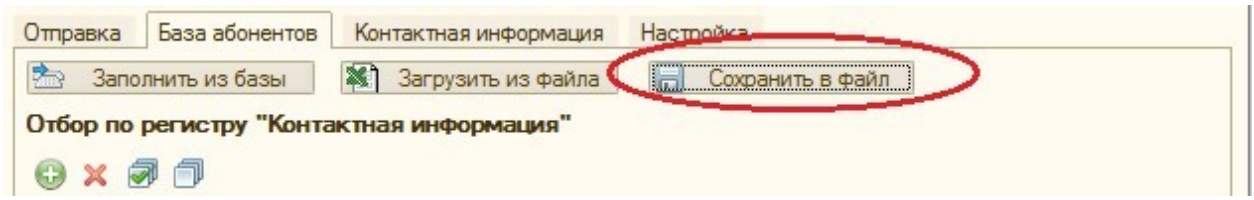
- Загружать из xls (xlsx) файла



При выборе данного типа загрузки, в открывшимся окне, будет предложено выбрать параметры для загрузки базы абонентов из файла. Здесь нужно выставить соответствие между колонками в xls-файле и наименованиями (телефон, имя адресата) в модуле.

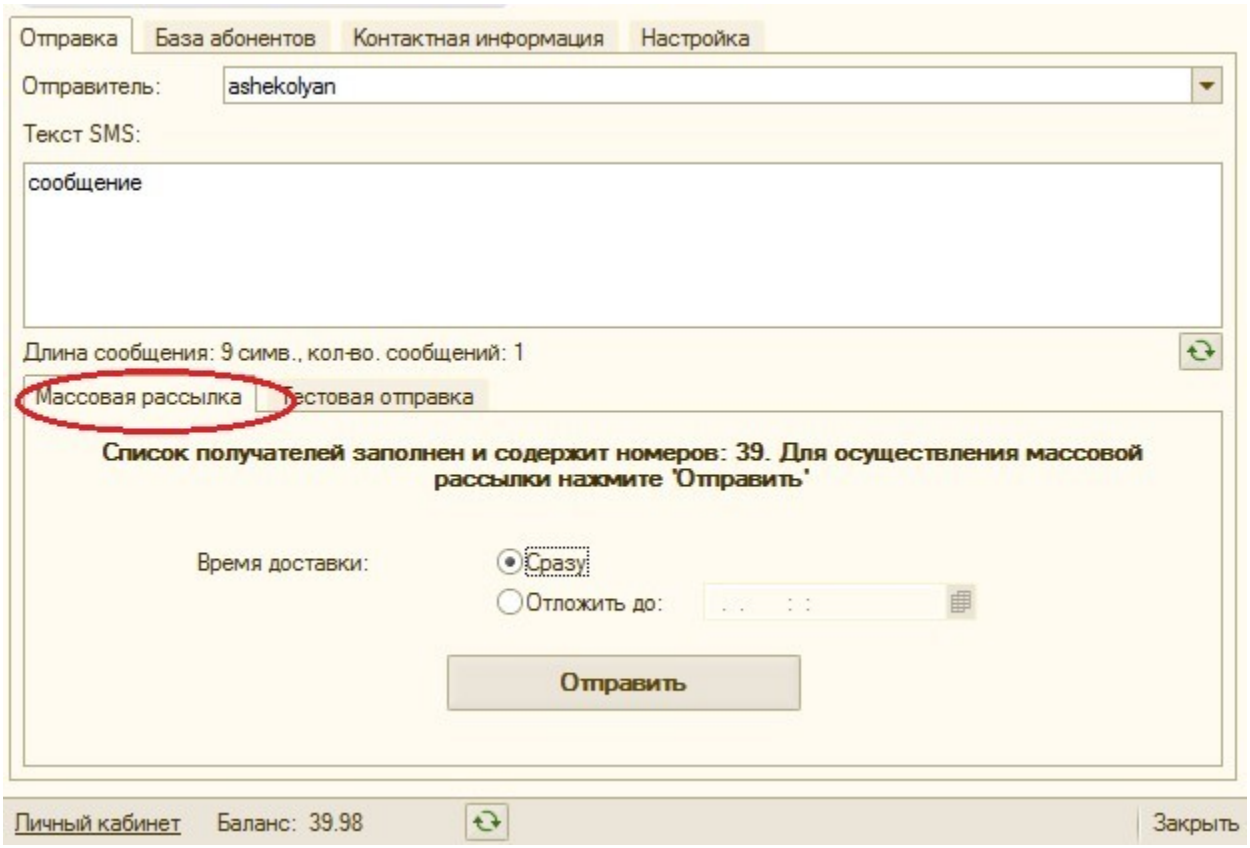


Так же, если во время работы с уже загруженной базой она будет изменена, есть возможность сохранить базу в формате xls.

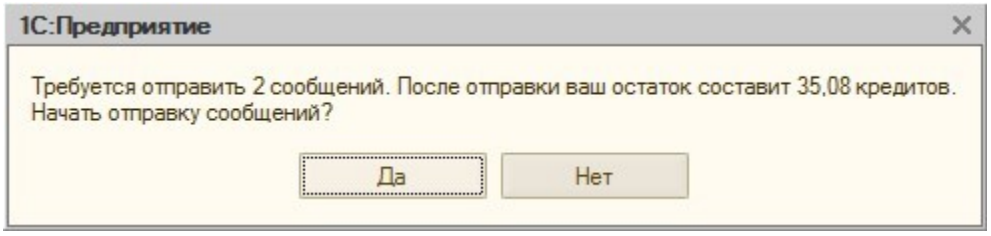


5.4.1 Мгновенная отправка сообщений

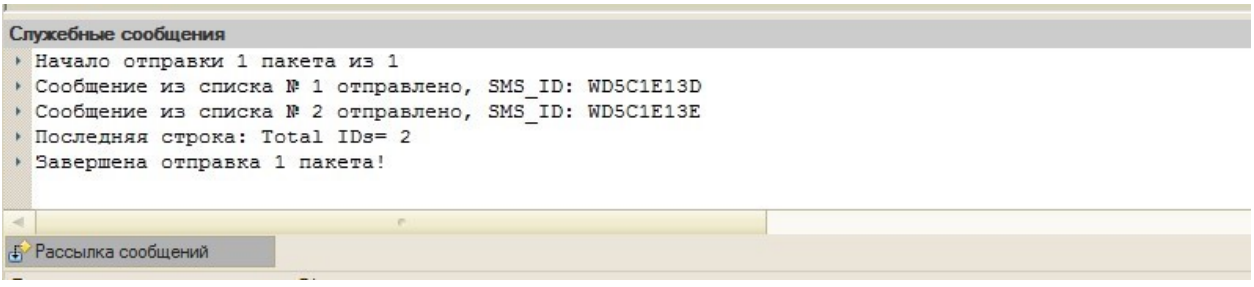
После того, как списки получателей сообщений сформированы, во вкладке «Отправка», следует выбрать вкладку «Массовая рассылка». В поле «Отправитель» выбрать имя отправителя, из списка, который вы вводили во вкладке «Настройки» (все имена отправителей должны быть зарегистрированы в личном кабинете). В поле текст SMS, вводите текст отправляемого сообщения.



После нажатия кнопки «Отправить», модуль подсчитает стоимость рассылки, и предупредит о том, какой баланс у Вас будет после осуществления рассылки SMS.

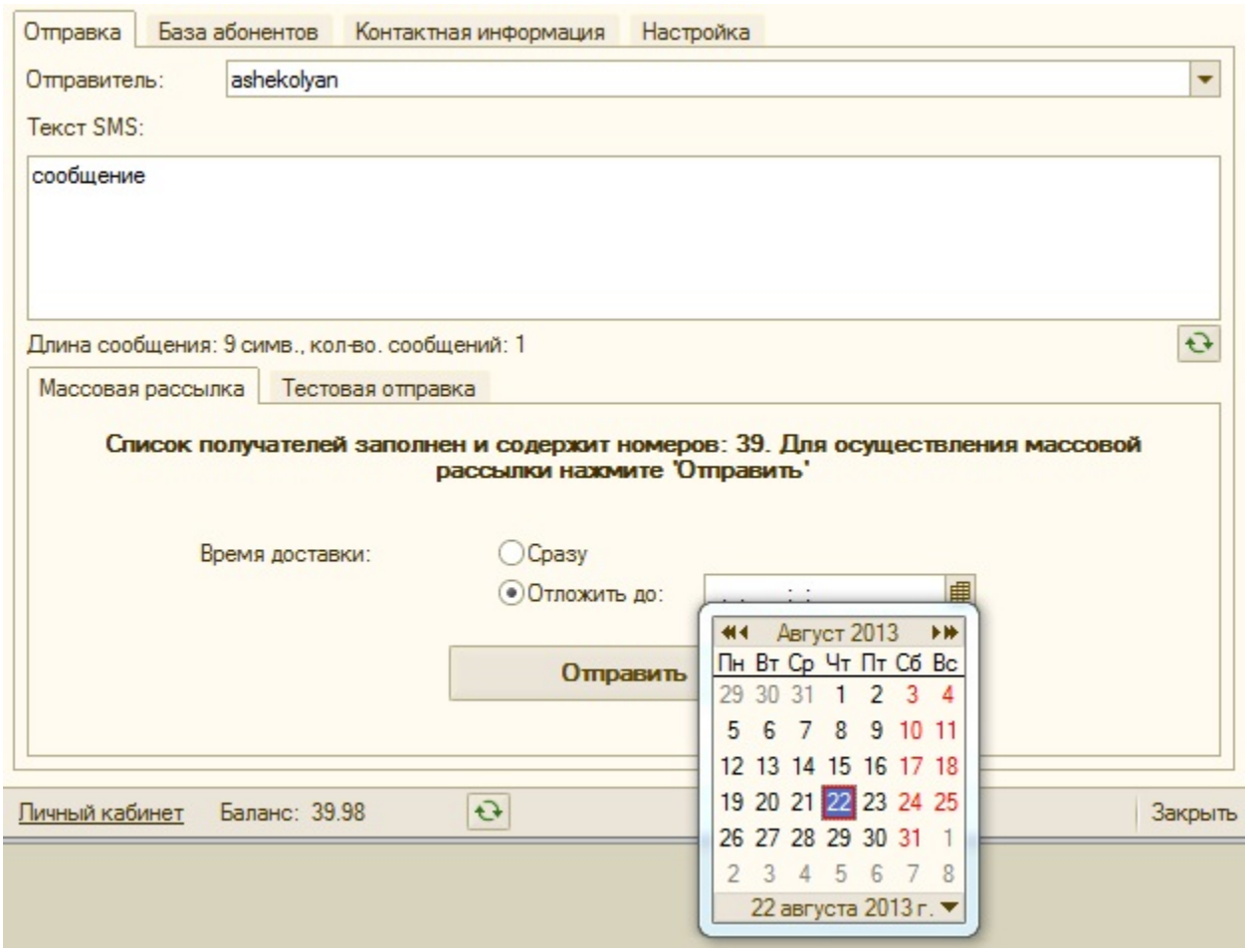


После удачной отправки сообщений, в служебных сообщениях появятся ID, отправленных сообщений, и их количество. (В случае, если некоторые данные были заполнены неверно вернется один из статусов операций, который можно посмотреть ниже)



5.4.2 Пакетная отправка отложенных по времени сообщений

Для того, чтобы создать пакетную отложенную отправку сообщений, следует, перед оправкой, выбрать дату и время, начиная с которого будет осуществляться рассылка.



5.5 Статусы операций и сообщений

Статусы операций

Статусы	Значение
OK_Operation_Completed	Операция выполнена
Error_Not_Enough_Credits	Ошибка: недостаточно кредитов
Error_Message_Rejected	Ошибка: сообщение отклонено
Error_Invalid_Source_Address	Ошибка: некорректный адрес отправителя сообщения
Error_Invalid_Destination_Address	Ошибка: некорректный номер получателя сообщения
Error_SMS_User_Disabled	Ошибка: СМС - пользователь заблокирован
Error_Invalid_MessageID	Ошибка: некорректный идентификатор сообщения
Error_Invalid_Header	Ошибка: некорректно переданы параметры
Error_Invalid_Login	Ошибка: неправильный логин
Error_Invalid_Password	Ошибка: неправильный пароль
Error_Unauthorised_IP_Address	Ошибка: не авторизованный IP-адрес
Error_SMS_User_Not_Activated	Ошибка: СМС - пользователь не активирован
Error_Message_Queue_Full	Ошибка: очередь сообщений полна
Error_Gateway_Offline	Ошибка: сервер недоступен
Error_Gateway_Busy	Ошибка: сервер занят другим запросом
Error_Database_Offline	Ошибка: сервер базы данных недоступен

Статусы сообщений

Статусы	Значение
Enqueued	Ожидает отправки
Delivered_To_Gateway	Отправлено
Sent	Отправлено
Delivered_To_Recipient	Доставлено
Error_Invalid_Destination_Address	Ошибка: некорректный номер получателя сообщения
Error_Invalid_Source_Address	Ошибка: некорректный адрес отправителя сообщения
Error_Rejected	Ошибка: сообщение отклонено
Error_Expired	Ошибка: истек срок жизни сообщения
Все остальное	Статус не распознан

Статусы передаются на русском языке, в случае служебных запросов могут быть на английском.

Глава 6

XML API

6.1 Обзор

Документация включает описание API для отправки SMS-сообщений через платформу Devino Telecom с примерами запросов. API включает в себя возможность как транзакционных, так и массовых рассылок, получение подробной статистики по рассылкам. Документ предназначен для разработчиков, которые хотят добавить возможность взаимодействия с Сервисом отправки SMS-сообщений на страницы своих сайтов или в свои приложения. Общение с сервисом осуществляется при помощи отправки XML запросов в кодировке UTF-8 на заданный адрес сервиса по протоколу HTTPS методом POST, проверка типа контента не осуществляется. Каждый запрос может состоять из отправляемых сообщений и (или) запросов для получения статусов и (или) запросов для получения входящих сообщений. Авторизация пользователя происходит путем передачи учетных сведений в теге package, обязательна при выполнении любых запросов.

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

Сервис доступен по адресу:

```
https://xmlapi.devinotele.com
```

6.2 Отправка сообщений

https://xmlapi.devinotele.com/Send.ashx

Отправка сообщений осуществляется в соответствии с очередностью по получению сообщений и временем отправки.

Пример запроса:

```
<?xml version="1.0" encoding="utf-8" ?>
<package login="login" password="123456">
  <message>
    <default sender="SMSINFO"/>
    <msg id="1234" recipient="+79021234567" sender="SMSINFO" date_beg="2008-12-27T15:55" date_end=
↪"2008-12-28T15:55" type="0">text</msg>
    <msg recipient="+79021234568">text</msg>
  </message>
</package>
```

Параметр	Тип данных	Обязательный	Описание
Default		Нет	Тег, в котором определяются общие атрибуты, указываемые для всех отправляемых сообщений. Если какой-либо атрибут указан в сообщении, то атрибут данного тега игнорируется
msg		да	Тег сообщения, в качестве параметра указывается текст отправляемого сообщения
id	integer	Нет	Пользовательский числовой идентификатор сообщения
recipient	varchar(21)	Да	Номер телефона. Пример: 79031234567, +79031234567
sender	varchar(11)	Да	Адрес отправителя
date_beg	datetime, ISO8601	Нет	Дата отправки сообщения, указывается для отложенной отправки сообщений.
date_end	datetime, ISO8601	Нет	Дата, после которой сообщение теряет актуальность и если оно еще не было отправлено абоненту, отправляться не будет.
url	varchar(100)	Да *	Ссылка для создания wap-push сообщения
type	integer	Да	Тип сообщения: 0-обычное сообщение 1-flash сообщение 2-wap-push сообщение

*Поле обязательное для отправки wap-push сообщений

На полученный запрос сервис возвращает в виде параметра статус сообщения, а в атрибутах идентификаторы, присвоенные сообщениям.

Пример ответа:

```
<?xml version="1.0" encoding="utf-8" ?>
<package>
  <message>
    <msg id="1234" sms_id="0" sms_count="1">201</msg>
    <msg sms_id="1234568" sms_count="1">1</msg>
  </message>
</package>
```

Параметр	Тип данных	Описание
msg		Тег сообщения, в качестве параметра возвращается код статуса
id	integer	Пользовательский числовой идентификатор сообщения, необязательный атрибут, возвращается при указании данного атрибута в запросе
sms_id	integer	Числовой идентификатор сообщения, присвоенный шлюзом
sms_count	integer	Реальное количество SMS к отправке

6.3 Запрос статусов сообщений

https://xmlapi.devinotele.com/Statistics.ashx

Статусы сообщений содержат информацию о текущем состоянии сообщения, регулярно обновляются и могут быть запрошены пользователем в любое время. **Статусы можно запрашивать за последние 3 календарных дня. Максимальное количество ID в запросе 1000.**

Пример запроса:

```
<?xml version="1.0" encoding="utf-8" ?>
<package login="login" password="123456">
  <status>
    <msg id="1234"/>
    <msg sms_id="1234568"/>
  </status>
</package>
```

Параметр	Тип данных	Обязательный	Описание
msg		Нет	Тег сообщения, для которого происходит запрос статуса
id	integer	Нет *	Пользовательский числовой идентификатор сообщения, возвращается при указании данного атрибута в запросе
sms_id	integer	Да *	Числовой идентификатор сообщения, присвоенный шлюзом

*Можно указать любой из этих параметров

В ответ на запрос возвращается XML документ содержащий статусы для запрошенных сообщений, либо соответствующие коды ошибок.

Пример ответа:

```
<?xml version="1.0" encoding="utf-8" ?>
<package>
  <status>
    <msg id="1234" sms_id="0" sms_count="1" date_completed="2009-03-14T15:27:03">102</msg>
    <msg sms_id="1234568" sms_count="1">1</msg>
  </status>
</package>
```

Параметр	Тип данных	Описание
msg		Тег сообщения, для которого происходит запрос статуса, в качестве параметра возвращается код статуса
id	integer	Пользовательский числовой идентификатор сообщения, возвращается при указании данного атрибута в запросе
sms_id	integer	Числовой идентификатор сообщения, присвоенный шлюзом
sms_count	integer	Реальное количество SMS к отправке
date_completed	datetime, ISO8601	Дата присвоения финального статуса

6.4 Коды статусов документа

При отправке XML документа могут возвращаться следующие коды ошибок:

Код	HTTP Status	Расшифровка
ERR_UNKNOWN	200	Неизвестная ошибка
ERR_FORMAT	201	Неправильный формат документа
ERR_AUTHORIZATION	202	Ошибка авторизации

Пример ответа:

```
<?xml version="1.0" encoding="utf-8" ?>
<package>
  <error>201</error>
</package>
```

Коды статусов сообщений

Данные коды используются при возврате статусов сообщений.

Статусы сообщений:

Код	HTTP Status	Расшифровка
SCHEDULED	100	Запланировано
ENROUTE	101	В очереди
DELIVERED	102	Доставлено
EXPIRED	103	Просроочено
DELETED	104	Просрочено
UNDELIVERABLE	105	Не доставлено
ACCEPTED	106	Принято
UNKNOWN	107	Ошибка
REJECTED	108	Отклонено
DISCARDED	109	Отклонено

Статусы ошибок:

Код	HTTP Status	Расшифровка
ERR_UNKNOWN	200	Неизвестная ошибка
ERR_ID	201	Неправильный ID сообщения
ERR_SENDER	202	Неправильный адрес отправителя
ERR_RECIPIENT	203	Неправильный номер получателя
ERR_LENGTH	204	Слишком длинное или пустое сообщение
ERR_USER_DISABLE	205	Пользователь отключен
ERR_BILLING	206	Ошибка биллинга
ERR_OVERLIMIT	207	Превышение лимита выделенных сообщений

Глава 7

SOAP

7.1 Обзор API

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

Предоставляемый API Сервиса отправки SMS сообщений позволяет осуществить:

- аутентификацию;
- получение баланса пользователя текущей сессии;
- получение входящих сообщений пользователя текущей сессии;
- отправка SMS с учетом часового пояса получателя;
- отправка SMS абонентам и возвращение системных идентификаторов SMS;
- отправка SMS абонентам и возвращение системных идентификаторов SMS с учетом часового пояса получателя;
- получение статуса отправленного SMS и время обновления статуса;
- получение статистики по SMS-рассылкам за заданный промежуток времени;
- отправка Viber адресатам и возвращение системных идентификаторов сообщений.
- отправка Viber-сообщений адресатам и возвращение системных идентификаторов сообщений с переотправкой по SMS;
- получение статуса отправленного viber-сообщения.

API сервиса отправки SMS организовано в соответствии с принципами SOAP. Протокол используется для обмена произвольными сообщениями в формате XML. SOAP может использоваться с любым протоколом прикладного уровня: SMTP, FTP, HTTP, HTTPS и др.

WSDL-документ для SOAP доступен по адресу:

```
https://ws.devinotele.com/SmsService.asmx?WSDL
```

Точка подключения:

```
https://ws.devinotele.com/SmsService.asmx
```

Предупреждение: Все запросы необходимо выполнять в кодировке UTF-8. Количество запросов 10 запросов/1 сек.

Исключительные ситуации. В случае возникновения исключительной ситуации во время обработки запроса или ошибки аутентификации, сервис возвращает код ошибки в виде:

```
<soap:Code>
  <soap:Value>soap:Receiver</soap:Value>
</soap:Code>
<soap:Reason>
  <soap:Text xml:lang="en">
      Server was unable to process request. ---
      &gt; Invalid user login or password
  </soap:Text>
</soap:Reason>
```

7.2 Аутентификация

Перед началом работы с методами сервиса необходимо получить идентификатор сессий. Он получается путем вызова GetSessionID и передачи логина/пароля. Если логин и пароль валидены, то возвращается идентификатор сессии, время жизни которого - 2 часа. Он является первым параметром и используется во всех запросах к этому сервису. Полученный идентификатор сессии действителен в течение 120 минут.

Протокол HTTP не имеет состояний. Это означает, что веб-сервер обрабатывает каждый HTTP-запрос со стороны внешнего приложения или сайта независимо, а сервер не сохраняет данные о значениях переменных, использованных в предшествующих запросах. Поэтому данные, полученные при авторизации. Пользователя, должны быть переданы и при осуществлении запроса получения баланса авторизованного пользователя.

Сервис создает идентификатор сессии в системе после прохождения аутентификации данных, передаваемых сервису, в POST-запросе следующего формата. Пример запроса:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetSessionID xmlns="http://ws.devinosms.com">
     <login>string</login>
     <password>string</password>
    </GetSessionID>
  </soap12:Body>
</soap12:Envelope>
Content-Type для параметров запроса:
Content-Type: application/soap+xml; charset=utf-8
```

Табл. 1. Описание параметров GetSessionID

Параметр	Тип данных	Обязательность	Описание
Login	String	Да	Логин, полученный при регистрации
Password	String	Да	Пароль, соответствующий логину

Пример ответа. В случае успешного прохождения аутентификации присланных данных сервис отправки SMS пришлет ответ со следующими параметрами:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetSessionIDResponse xmlns="http://ws.devinosms.com">
      <GetSessionIDResult>string</GetSessionIDResult>
    </GetSessionIDResponse>
  </soap12:Body>
</soap12:Envelope>
```

7.3 Получение баланса пользователя

Сервис возвращает значение баланса авторизованного пользователя по SessionID. Овердрафт при этом учитывается. **Пример запроса:**

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetBalance xmlns="http://ws.devinosms.com">
       <sessionID>string</sessionID>
    </GetBalance>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 2. Описание параметров GetBalance

Параметр	Тип данных	Обязательность	Описание
SessionID	String	Да	Идентификатор сессии, полученный при аутентификации

Сервис проверяет валидность полученного SessionID (проверяет актуальность и наличие в Системе) и, в случае успеха, авторизует Пользователя и в ответе присылает баланс пользователя следующего вида. **Пример ответа:**

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
```

(continues on next page)

(продолжение с предыдущей страницы)

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetBalanceResponse xmlns="http://ws.devinosms.com">
      <GetBalanceResult>decimal</GetBalanceResult>
    </GetBalanceResponse>
  </soap12:Body>
</soap12:Envelope>
```

7.4 Отправка SMS

7.4.1 Отправка SMS с учетом часового пояса получателя

Для того чтобы сообщение получателю было доставлено в срок, задается отложенная отправка SendMessageByTimeZone. Часовой пояс вычисляется на основе номера получателя и, в зависимости от него, сообщение отправляется через заданный временной интервал, чтобы осуществилась доставка по местному времени получателя.

Пример запроса:

```
POST /smsservice.asmx HTTP/1.1
Host: ws.devinotele.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
 <soap12:Body>
    <SendMessageByTimeZone xmlns="http://ws.devinosms.com">
      <sessionID>string</sessionID>
      <sourceAddress>string</sourceAddress>
      <destinationAddress>string</destinationAddress>
      <data>string</data>
      <sendDate>dateTime</sendDate>
      <validity>int</validity>
    </SendMessageByTimeZone>
 </soap12:Body>
</soap12:Envelope>
```

Табл. 3. Описание параметров SendMessageByTimeZone

Параметр	Тип данных	Обязательность	Описание
SessionID	String	Да	Идентификатор сессии, полученный при аутентификации (36 символов).
DestinationAddress	String	Да	Номер получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567, +79031234567, 89031234567
Data	String	Да	Текст сообщения, сообщение не должно быть длиннее 2000 символов
SourceAddress	String	Да	Адрес отправителя сообщения. До 11 латинских символов или до 15 цифровых. Как получить адресотправителя см. в начале документа.
SendDate	DateTime	Да	Дата и время отправки (пример 2010-0601T19:14:00). Сообщение будет отправлено только при наступлении полученных даты и времени с учетом текущего часового пояса получателя.
Validity	Int	Нет	Время жизни сообщения (мин), по умолчанию 2880 мин.

Перед отправкой SMS Сервис проверяет запрос на:

- наличие обязательных параметров;
- валидность сессии Пользователя (аутентификацию и определение, не истекло ли его время жизни SessionID);
- достаточно ли Баланса Пользователя на отправку SMS (достаточность определяется на основании тарифа Пользователя на отправку SMS для мобильного оператора указанного в запросе номера);
- валидность указанного в запросе номера;
- валидность адреса отправителя;
- длину сообщения.

Если все проверки пройдены успешно, то сервис отправит сообщение в SMS-центр и вернет идентификатор отправленного сообщения с параметрами как в примере ответа. Размер 1 сообщения составляет: 70 русских символов или 160 символов латиницей. Сервис может возвратить более 1 идентификатора, если текст сообщения выходит за пределы 1 sms.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
 <soap12:Body>
    <SendMessageByTimeZoneResponse xmlns="http://ws.devinosms.com">
      <SendMessageByTimeZoneResult>
        <string>string</string>
        <string>string</string>
      </SendMessageByTimeZoneResult>
    </SendMessageByTimeZoneResponse>
  </soap12:Body>
</soap12:Envelope>
```

7.4.2 Отправка SMS адресатам и возвращение системных идентификаторов сообщений

Данный метод поддерживает массовую отправку сообщений (до 1000 сообщений) в одном запросе.

Пример запроса:

```
POST /smsservice.asmx HTTP/1.1
Host: ws.devinotele.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <SendMessage xmlns="http://ws.devinosms.com">
      <sessionID>string</sessionID>
      <message>
          <Data>string</Data>
          <DelayUntilUtc>dateTime</DelayUntilUtc>
          <DestinationAddresses>
            <string>string</string>
            <string>string</string>
          </DestinationAddresses>
          <SourceAddress>string</SourceAddress>
          <ReceiptRequested>boolean</ReceiptRequested>
          <Validity>int</Validity>
      </message>
    </SendMessage>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 4. Описание параметров SendMessage

Параметр	Тип данных	Обязательность	Описание
SessionID	String	Да	Идентификатор сессии, полученный при аутентификации (36 символов).
Data	String	Да	Текст сообщения, сообщение не должно быть длиннее 2000 символов
DelayUntilUtc	DateTime	Нет	Время отправки. Если не заполнено, то отправляется немедленно.
DestinationAddresses	String[]	Да	Номер получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567, +79031234567, 89031234567
SourceAddress	String	Да	Адрес отправителя сообщения. До 11 латинских символов или до 15 цифровых.
ReceiptRequested	Boolean	Нет	Запрос о доставке
Validity	Int	Нет	Время жизни сообщения (мин), по умолчанию 2880 мин.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
```

(continues on next page)

(продолжение с предыдущей страницы)

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
   <SendMessageResponse xmlns="http://ws.devinosms.com">
      <SendMessageResult>
         <string>string</string>
         <string>string</string>
       </SendMessageResult>
   </SendMessageResponse>
  </soap12:Body>
</soap12:Envelope>
```

7.4.3 Отправка SMS адресатам и возвращение системных идентификаторов сообщений с учетом часового пояса получателей

Для того чтобы сообщение получателям было доставлено в срок, задается отложенная отправка SendMessageByTimeZoneToAddresses. Часовой пояс вычисляется на основе номера получателя и, в зависимости от него, сообщение отправляется через заданный временной интервал, чтобы осуществилась доставка по местному времени получателя. Данный метод поддерживает массовую отправку сообщений (до 1000 сообщений) в одном запросе.

Пример запроса:

```
POST / HTTP/1.1
Host: localhost
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.
↪org/2001/XMLSchema"     xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <SendMessageByTimeZoneToAddresses xmlns="http://ws.devinosms.com">
      <sessionID>string</sessionID>
      <sourceAddress>string</sourceAddress>
      <destinationAddresses>
        <string>string</string>
        <string>string</string>
      </destinationAddresses>
      <data>string</data>
      <sendDate>dateTime</sendDate>
      <validity>int</validity>
    </SendMessageByTimeZoneToAddresses>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 5. Описание параметров SendMessageByTimeZoneToAddresses

Параметр	Тип данных	Обязательность	Описание
SessionID	String	Да	Идентификатор сессии, полученный при аутентификации (36 символов).
DestinationAddresses	String[]	Да	Номер получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567, +79031234567, 89031234567
Data	String	Да	Текст сообщения, сообщение не должно быть длиннее 2000 символов
SourceAddress	String	Да	Адрес отправителя сообщения. До 11 латинских символов или до 15 цифровых.
SendDate	DateTime	Да	Дата и время отправки (пример 2010-0601T19:14:00). Сообщение будет отправлено только при наступлении полученных даты и времени с учетом текущего часового пояса получателя.
Validity	Int	Нет	Время жизни сообщения (мин), по умолчанию 2880 мин.

Перед отправкой SMS Сервис проверяет запрос на: * наличие обязательных параметров; * валидность сессии пользователя (аутентификацию и определение, не истекло ли его время жизни SessionID); * достаточно ли баланса пользователя на отправку SMS (достаточность определяется на основании тарифа Пользователя на отправку SMS для мобильного оператора указанного в запросе номера); * валидность указанных в запросе номеров; * валидность адреса отправителя; * длину сообщения.

Если все проверки пройдены успешно, то сервис отправит сообщение в SMS-центр и вернет идентификаторы отправленных сообщений с параметрами как в примере ответа. Размер 1 сообщения составляет: 70 символов кириллицей или 160 символов латиницей. Сервис может вернуть более 1 идентификатора, если текст сообщения выходит за пределы 1 sms.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.
↪org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <SendMessageByTimeZoneToAddressesResponse xmlns="http://ws.devinosms.com">
      <SendMessageByTimeZoneToAddressesResult>
        <string>string</string>
        <string>string</string>
      </SendMessageByTimeZoneToAddressesResult>
    </SendMessageByTimeZoneToAddressesResponse>
  </soap12:Body>
</soap12:Envelope>
```

7.5 Получение статуса отправленного SMS

Сервис возвращает статус отправленного sms в соответствии со значениями параметров sessionID и messageID.

Пример запроса:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetMessageState xmlns="http://ws.devinosms.com">
      <sessionID>string</sessionID>
      <messageID>string</messageID>
    </GetMessageState>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 6. Описание параметров GetMessageState

Параметр	Тип данных	Обязательность	Описание
sessionID	String	Да	Идентификатор сессии, полученный при аутентификации (36 символов).
messageID	String	Да	Идентификатор сообщения (сегмента сообщения). Для одного запроса будет выполнен возврат статуса только одного сообщения (сегмента сообщения).

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetMessageStateResponse xmlns="http://ws.devinosms.com">
      <GetMessageStateResult>
        <State>int</State>
        <CreationDateUtc>dateTime</CreationDateUtc>
        <SubmittedDateUtc>dateTime</SubmittedDateUtc>
        <ReportedDateUtc>dateTime</ReportedDateUtc>
        <StateDescription>string</StateDescription>
        <Price>decimal</Price>
      </GetMessageStateResult>
    </GetMessageStateResponse>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 7. Описание возвращаемых параметров

Название	Тип	Описание
State	int	Статус. Типы статусов сообщений приведены в примечании.
CreationDateUtc	dateTime	Дата и время создания (пример 2010-0601T19:14:00) в UTC.
SubmittedDateUtc	dateTime	Время получения в Devino (в UTC).
ReportedDateUtc	dateTime	Время получения отчета (в UTC).
StateDescription	string	Описание статуса (например Description(«Недопустимый адрес получателя»)).
Price	decimal	Цена

7.6 Получение статистики по SMS-рассылкам за заданный промежуток времени

Сервис возвращает статистику по SMS-рассылкам за период, в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата.

Пример запроса:

```
POST /smsservice.asmx HTTP/1.1
Host: ws.devinotele.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetStatistics xmlns="http://ws.devinosms.com">
     <sessionId>string</sessionId>
     <startDateTime>dateTime</startDateTime>
     <endDateTime>dateTime</endDateTime>
    </GetStatistics>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 8. Описание параметров GetStatistics

Параметр	Тип данных	Обязательность	Описание
sessionId	String	Да	Идентификатор сессии (36 символов).
startDateTime	DateTime	Да	Дата и время начала периода, за который необходимо получить статистику, например 2012-01-18T00:00:00. Время в UTC.
endDateTime	DateTime	Да	Дата и время конца периода, за который необходимо получить статистику, например 2012-01-18T23:59:00. Время в UTC.

После получения запроса сервис проверит валидность присланного идентификатора сессии и даты начала/окончания формирования статистики (включая ограничение на то, что охватываемый диапазон должен не превышать 3 месяцев). Если все проверки пройдены успешно, то сервис вернет статистику по sms со следующими параметрами:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetStatisticsResponse xmlns="http://ws.devinosms.com">
      <GetStatisticsResult>
        <Sent>int</Sent>
        <Delivered>int</Delivered>
        <Errors>int</Errors>
        <InProcess>int</InProcess>
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        <Expired>int</Expired>
        <Rejected>int</Rejected>
      </GetStatisticsResult>
    </GetStatisticsResponse>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 9. Описание возвращаемых параметров

Название	Тип	Описание
Sent	int	Количество отправленных сообщений
Delivered	int	Количество доставленных сообщений.
Errors	int	Количество ошибок
InProcess	int	Количество сообщений «в процессе отправки»
Expired	int	Количество просроченных сообщений.
Rejected	int	Количество отклоненных сообщений

7.7 Получение входящих сообщений

Система позволяет заводить входящие номера и на них получать sms. Входящий номер заводится через личный кабинет. Сервис возвращает входящие сообщения пользователя в интервале maxDate, minDate(который передан в этом запросе). Пример запроса:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetIncomingMessages xmlns="http://ws.devinosms.com">
      <sessionID>string</sessionID>
      <maxDateUTC>dateTime</maxDateUTC>
      <minDateUTC>dateTime</minDateUTC>
    </GetIncomingMessages>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 10. Описание параметров GetIncomingMessages

Параметр	Тип данных	Обязательность	Описание
sessionId	String	Да	Идентификатор сессии, полученный при аутентификации
maxDateUTC	DateTime	Да	Значение интервала _по. Пример: 2014-11-01T11:30
minDateUTC	DateTime	Да	Значение интервала с_. Пример: 2014-11-01T11:30 например 2012-01-18T23:59:00. Время в UTC.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
```

(continues on next page)

(продолжение с предыдущей страницы)

```
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
<soap12:Body>
 <GetIncomingMessagesResponse xmlns="http://ws.devinosms.com">
    <GetIncomingMessagesResult>
      <IncomingMessage>
        <Data>string</Data>
        <SourceAddress>string</SourceAddress>
        <DestinationAddress>string</DestinationAddress>
        <CreatedDateUtc>dateTime</CreatedDateUtc>
      </IncomingMessage>
      <IncomingMessage>
        <Data>string</Data>
        <SourceAddress>string</SourceAddress>
        <DestinationAddress>string</DestinationAddress>
        <CreatedDateUtc>dateTime</CreatedDateUtc>
      </IncomingMessage>
    </GetIncomingMessagesResult>
  </GetIncomingMessagesResponse>
 </soap12:Body>
</soap12:Envelope>
```

Табл. 11 Описание параметров GetIncomingMessages

Название	Тип	Описание
Data	String	Текст сообщения
SourceAddress	String	Адрес отправителя
DestinationAddress	String	Адрес получателя
CreatedDateUtc	DateTime	Дата создания
Expired	int	Количество просроченных сообщений.
Rejected	int	Количество отклоненных сообщений

7.8 Отправка Viber-сообщений

7.8.1 Отправка Viber адресатам и возвращение системных идентификаторов сообщений

Данный метод поддерживает массовую отправку сообщений (до 1000 сообщений) в одном запросе.

Пример запроса:

```
POST /ViberService.asmx HTTP/1.1
Host: ws.devinotele.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <SendMessage xmlns="http://ws.devinosms.com">
```

(continues on next page)

(продолжение с предыдущей страницы)

```
      <sessionID>string</sessionID>
      <message>
          <Data>string</Data>
          <DestinationAddresses>
            <string>string</string>
            <string>string</string>
          </DestinationAddresses>
          <SourceAddress>string</SourceAddress>
          <Validity>int</Validity>
          <Optional>string</Optional>
      </message>
    </SendMessage>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 12. Описание параметров SendMessage

Параметр	Тип данных	Обязательность	Описание
sessionId	String	Да	Идентификатор сессии (36 символов).
Data	String	Да	Текст сообщения, сообщение не должно быть длиннее 1000 символов
DestinationAddresses	String []	Да	Номер получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567, +79031234567, 89031234567
SourceAddress	String	Да	Адрес отправителя сообщения. До 11 латинских или цифровых символов.
Validity	Int	Да	Время жизни сообщения (мин, от 1 до 1440)
Optional	String	Нет	Дополнительный параметр

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
   <SendMessageResponse xmlns="http://ws.devinosms.com">
      <SendMessageResult>
         <string>string</string>
         <string>string</string>
       </SendMessageResult>
   </SendMessageResponse>
  </soap12:Body>
</soap12:Envelope>
```

7.8.2 Отправка Viber адресатам и возвращение системных идентификаторов сообщений с переотправкой по sms

Данный метод поддерживает массовую отправку сообщений (до 1000 сообщений) в одном запросе.

Пример запроса:

```
POST /ViberService.asmx HTTP/1.1
Host: ws.devinotele.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <SendMessageWithResend xmlns="http://ws.devinosms.com">
      <sessionID>string</sessionID>
      <message>
          <Data>string</Data>
          <DestinationAddresses>
            <string>string</string>
            <string>string</string>
          </DestinationAddresses>
          <SourceAddress>string</SourceAddress>
          <Validity>int</Validity>
          <Optional>string</Optional>
      </message>
    </SendMessageWithResend>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 13. Описание параметров SendMessageWithResend

Параметр	Тип данных	Обязательность	Описание
sessionId	String	Да	Идентификатор сессии (36 символов).
Data	String	Да	Текст сообщения, сообщение не должно быть длиннее 1000 символов
DestinationAddresses	String []	Да	Номер получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567, +79031234567, 89031234567
SourceAddress	String	Да	Адрес отправителя сообщения. До 11 латинских или цифровых символов.
Validity	Int	Да	Время жизни сообщения (мин, от 1 до 1440)
Optional	String	Нет	Дополнительный параметр

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
   <SendMessageWithResendResponse xmlns="http://ws.devinosms.com">
      <SendMessageResult>
         <string>string</string>
         <string>string</string>
```

(continues on next page)

(продолжение с предыдущей страницы)

```
      </SendMessageResult>
    </SendMessageWithResendResponse>
  </soap12:Body>
</soap12:Envelope>
```

7.9 Получение статуса отправленного Viber-сообщения

Сервис возвращает статус отправленного viber-сообщения в соответствии со значениями параметров sessionID и messageID.

Пример запроса:

```
POST /ViberService.asmx HTTP/1.1
Host: 127.0.0.1
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.
↪org/2001/XMLSchema"   xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetMessageState xmlns="http://ws.devinosms.com">
      <sessionID>string</sessionID>
      <messageID>long</messageID>
    </GetMessageState>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 14. Описание параметров GetMessageState

Параметр	Тип данных	Обязательность	Описание
sessionID	String	Да	Идентификатор сессии, полученный при аутентификации (36 символов).
messageID	String	Да	Идентификатор viber-сообщения. Для одного запроса будет выполнен возврат статуса только одного сообщения.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.
↪org/2001/XMLSchema"   xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
  <soap12:Body>
    <GetMessageStateResponse xmlns="http://ws.devinosms.com">
      <GetMessageStateResult>
        <State>Enqueued or Sent or Delivered or Read or Undelivered or Failed or Unknown or Expired
↪</State>
        <ResentSms>
```

(continues on next page)

(продолжение с предыдущей страницы)

```
          <ViberSmsMessageStateInfo>
            <Id>long</Id>
          </ViberSmsMessageStateInfo>
          <ViberSmsMessageStateInfo>
            <Id>long</Id>
          </ViberSmsMessageStateInfo>
        </ResentSms>
      </GetMessageStateResult>
    </GetMessageStateResponse>
  </soap12:Body>
</soap12:Envelope>
```

Табл. 15. Описание возвращаемых параметров

Название	Тип	Описание
State	int	Статус. Типы статусов сообщений приведены в примечании.
CreationDateUtc	dateTime	Дата и время создания (пример 2010-0601T19:14:00) в UTC.
SubmittedDateUtc	dateTime	Время получения в Devino (в UTC).
ReportedDateUtc	dateTime	Время получения отчета (в UTC).
StateDescription	string	Описание статуса. Например Description(«Недопустимый адрес получателя»).
Price	decimal	Цена
ResentSms	ViberSmsMessageStateInfo[]	Коллекция статусов сообщений, которые были переотправлены по текущему viber-сообщению

7.10 Коды ошибок и статусы сообщений

Табл. 16. Статусы сообщений

БД Devino	Наименование	Описание	Подробное описание
-200	Ошибка	Errors=-200	Статус для фильтра «Ошибка» в детализации
-100	Протарифицировано	Tarificated = -100	Статус для фильтра «Протарифицировано» в детализации
-3	Ошибка	ErrorSendingDateTimeInterpretation = -3	Ошибка интерпретации даты и времени отправки
-1	Отправлено	Sent = -1	Сообщение отправлено
-2	Отправляется	LocalQueued = -2	Сообщение отправляется
-40	Ожидание	Queued = -40	Сообщение в статусе «ожидание»
-30	Остановлено	Sending_To_Gateway = -30	Отправлено в шлюз
-20	Отправлено/ получателю	Sending_To_Recipient = -20	Сообщение отправлено получателю
0	Доставлено	Delivered_To_Recipient = 0	Сообщение доставлено
0x0000000B	Ошибка	Error_Invalid_Destination_Address =0x0000000B	Неверно введён адрес получателя
0x0000000A	Ошибка	Error_Invalid_Source_Address =0x0000000A	Неверно введён адрес отправителя
41	Ошибка	Error_Incompatible_Destination = 41	Недопустимый адрес получателя
42	Ошибка	Error_Rejected = 42	Отклонено
46	Ошибка	Error_Expired = 46	Просрочен
47	Ошибка	Deleted = 47	Просрочено
48	Ошибка	Devino_Rejected = 48	Ошибка
0x000000FF	Неизвестный	Unknown = 0x000000FF	Внутренняя ошибка
0x00000008	Ошибка	System_Error = 0x00000008	Внутренняя ошибка

Глава 8

SOAP без SessionID

8.1 Обзор API

Предоставляемый API Сервиса отправки SMS сообщений позволяет осуществить:

- получение баланса пользователя текущей сессии;
- получение входящих сообщений пользователя текущей сессии;
- отправка SMS с учетом часового пояса получателя;
- отправка SMS абонентам и возвращение системных идентификаторов SMS;
- получение статуса отправленного SMS и время обновления статуса;
- получение статистики по SMS-рассылкам за заданный промежуток времени

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

API Сервиса отправки SMS организовано в соответствии с принципами SOAP. Протокол используется для обмена произвольными сообщениями в формате XML. SOAP может использоваться с любым протоколом прикладного уровня: *SMTP, FTP, HTTP, HTTPS и др.*

WSDL-документ для SOAP доступен по адресу:

```
https://ws.devinotele.com/SmsServicev2.asmx?WSDL
```

Точка подключения:

```
https://ws.devinotele.com/SmsServicev2.asmx
```

Предупреждение: Все запросы необходимо выполнять в кодировке UTF-8. Количество запросов 10 запросов/1 сек

Исключительные ситуации. В случае возникновения исключительной ситуации во время обработки запроса или ошибки аутентификации, сервис возвращает код ошибки в виде:

```
<soap:Code>
    <soap:Value>soap:Receiver</soap:Value>
</soap:Code>
<soap:Reason>
    <soap:Text xml:lang="en">
        Server was unable to process request. ---
        &gt; Invalid user login or password
    </soap:Text>
</soap:Reason>
```

8.2 Получение баланса пользователя

Сервис возвращает значение баланса авторизованного пользователя по Login/Password. Овердрафт при этом учитывается. Пример запроса:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
   <soap12:Body>
        <GetBalance xmlns="http://ws.devinosms.com">
            <login>string</login>
            <password>string</password>
        </GetBalance>
    </soap12:Body>
</soap12:Envelope>
```

Таблица 1 - Описание параметров GetBalance

Параметр	Тип данных	Обязательность	Описание
Login	String	Да	Логин, полученный при регистрации
Password	String	Да	Пароль, соответствующий логину

Сервис проверяет валидность Login/Password и, в случае успеха, авторизует Пользователя и в ответе присылает баланс пользователя следующего вида. Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <GetBalanceResponse xmlns="http://ws.devinosms.com">
            <GetBalanceResult>decimal</GetBalanceResult>
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        </GetBalanceResponse>
    </soap12:Body>
</soap12:Envelope>
```

8.3 Отправка SMS

8.3.1 Отправка SMS с учетом часового пояса получателя

Для того чтобы сообщение получателю было доставлено в срок, задается отложенная отправка SendMessageByTimeZone. Часовой пояс вычисляется на основе номера получателя и, в зависимости от него, сообщение отправляется через заданный временной интервал, чтобы осуществилась доставка по местному времени получателя. Пример запроса:

```
POST /smsservicev2.asmx HTTP/1.1
Host: ws.devinotele.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <SendMessageByTimeZone xmlns="http://ws.devinosms.com">
            <login>string</login>
            <password>string</password>
            <sourceAddress>string</sourceAddress>
            <destinationAddress>string</destinationAddress>
            <data>string</data>
            <sendDate>dateTime</sendDate>
            <validity>int</validity>
        </SendMessageByTimeZone>
    </soap12:Body>
</soap12:Envelope>
```

Таблица 2 - Описание параметров SendMessageByTimeZone

Параметр	Тип данных	Обязательность	Описание
Login	String	Да	Логин, полученный при регистрации
Password	String	Да	Пароль, соответствующий логину
DestinationAddress	String	Да	Номер получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567, +79031234567, 89031234567
Data	String	Да	Текст сообщения, сообщение не должно быть длиннее 2000 символов
SourceAddress	String	Да	Адрес отправителя сообщения. До 11 латинских символов или до 15 цифровых. Как получить адресотправителя см. в начале документа.
SendDate	DateTime	Да	Дата и время отправки (пример 2010-0601T19:14:00). Сообщение будет отправлено только при наступлении полученных даты и времени с учетомтекущего часового пояса получателя.
Validity	Int	Нет	Время жизни сообщения (мин), по умолчанию 2880 мин.

Перед отправкой SMS Сервис проверяет запрос на:

- наличие обязательных параметров;
- валидность Login/Password;
- достаточно ли Баланса Пользователя на отправку SMS (достаточность определяется на основании тарифа Пользователя на отправку SMS для мобильного оператора указанного в запросе номера);
- валидность указанного в запросе номера;
- валидность адреса отправителя;
- длину сообщения.

Если все проверки пройдены успешно, то Сервис отправит сообщение в SMS-центр и вернет идентификатор отправленного сообщения с параметрами как в примере ответа. Размер 1 сообщения составляет: 70 русских символов или 160 символов латиницей. Сервис может возвратить более 1 идентификатора, если текст сообщения выходит за пределы 1 sms. Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <SendMessageByTimeZoneResponse xmlns="http://ws.devinosms.com">
            <SendMessageByTimeZoneResult>
                <string>string</string>
                <string>string</string>
            </SendMessageByTimeZoneResult>
        </SendMessageByTimeZoneResponse>
   </soap12:Body>
</soap12:Envelope>
```

8.3.2 Отправка SMS адресатам и возвращение системных идентификаторов сообщений

Данный метод поддерживает массовую отправку сообщений (до 1000 сообщений) в одном запросе.

Пример запроса:

```
POST /smsservicev2.asmx HTTP/1.1
Host: ws.devinotele.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <SendMessage xmlns="http://ws.devinosms.com">
            <login>string</login>
            <password>string</password>
            <message>
```

(continues on next page)

(продолжение с предыдущей страницы)

```
                <Data>string</Data>
                <DelayUntilUtc>dateTime</DelayUntilUtc>
                <DestinationAddresses>
                    <string>string</string>
                    <string>string</string>
                </DestinationAddresses>
                <SourceAddress>string</SourceAddress>
                <ReceiptRequested>boolean</ReceiptRequested>
                <Validity>int</Validity>
            </message>
        </SendMessage>
    </soap12:Body>
</soap12:Envelope>
```

Таблица 3 - Описание параметров SendMessage

Параметр	Тип данных	Обязательность	Описание
Login	String	Да	Логин, полученный при регистрации
Password	String	Да	Пароль, соответствующий логину
Data	String	Да	Текст сообщения, сообщение не должно быть длиннее 2000 символов
DelayUntilUtc	DateTime	Нет	Время отправки. Если не заполнено, то отправляется немедленно.
DestinationAddress	String[]	Да	Номер получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567, +79031234567, 89031234567
SourceAddress	String	Да	Адрес отправителя сообщения. До 11 латинских имволов или до 15 цифровых.
ReceiptRequested	Boolean	Нет	Запрос о доставке
Validity	Int	Нет	Время жизни сообщения (мин), по умолчанию 2880 мин.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <SendMessageResponse xmlns="http://ws.devinosms.com">
            <SendMessageResult>
                <string>string</string>
                <string>string</string>
            </SendMessageResult>
        </SendMessageResponse>
    </soap12:Body>
</soap12:Envelope>
```

8.3.3 Отправка SMS адресатам и возвращение системных идентификаторов сообщений с учетом часового пояса получателя

Для того чтобы сообщение получателю было доставлено в срок, задается отложенная отправка SendMessageByTimeZoneToAddresses. Часовой пояс вычисляется на основе номера получателя и, в зависимости от него, сообщение отправляется через заданный временной интервал, чтобы осуществилась доставка по местному времени получателя. Пример запроса:

Пример запроса:

```
POST / HTTP/1.1
Host: localhost
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.
↪org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <SendMessageByTimeZoneToAddresses xmlns="http://ws.devinosms.com">
            <login>string</login>
            <password>string</password>
            <sourceAddress>string</sourceAddress>
            <DestinationAddresses>
                    <string>string</string>
                    <string>string</string>
            </DestinationAddresses>
            <data>string</data>
            <Validity>int</Validity>
        </SendMessageByTimeZoneToAddresses>
     </soap12:Body>
</soap12:Envelope>
```

Таблица 4 - Описание параметров SendMessageByTimeZoneToAddresses

Параметр	Тип данных	Обязательность	Описание
Login	String	Да	Логин, полученный при регистрации
Password	String	Да	Пароль, соответствующий логину
Data	String	Да	Текст сообщения, сообщение не должно быть длиннее 2000 символов
SendDate	DateTime	Да	Дата и время отправки (пример 2010-0601T19:14:00). Сообщение будет отправлено только при наступлении полученных даты и времени с учетом текущего часового пояса получателя.
DestinationAddresses	String[]	Да	Номер получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567, +79031234567, 89031234567
SourceAddress	String	Да	Адрес отправителя сообщения. До 11 латинских имволов или до 15 цифровых.
Validity	Int	Нет	Время жизни сообщения (мин), по умолчанию 2880 мин.

Перед отправкой SMS Сервис проверяет запрос на:

- наличие обязательных параметров;
- валидность Login/Password;

- достаточно ли Баланса Пользователя на отправку SMS (достаточность определяется на основании тарифа Пользователя на отправку SMS для мобильного оператора указанного в запросе номера);
- валидность указанных в запросе номеров;
- валидность адреса отправителя;
- длину сообщения.

Если все проверки пройдены успешно, то Сервис отправит сообщение в SMS-центр и вернет идентификаторы отправленных сообщений с параметрами как в примере ответа. Размер 1 сообщения составляет: 70 русских символов или 160 символов латиницей. Сервис может возвратить более 1 идентификатора, если текст сообщения выходит за пределы 1 sms. Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.
↪org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <SendMessageByTimeZoneToAddressesResponse xmlns="http://ws.devinosms.com">
            <SendMessageByTimeZoneToAddressesResult>>
                <string>string</string>
                <string>string</string>
            <SendMessageByTimeZoneToAddressesResult>
        </SendMessageByTimeZoneToAddressesResponse>
    </soap12:Body>
</soap12:Envelope>
```

8.4 Получение статуса отправленного SMS

Сервис возвращает статус отправленного sms в соответствии со значениями параметров по Login/Password и messageID. Пример запроса:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
 <soap12:Body>
     <GetMessageState xmlns="http://ws.devinosms.com">
         <login>string</login>
         <password>string</password>
         <messageID>string</messageID>
     </GetMessageState>
 </soap12:Body>
</soap12:Envelope>
```

Таблица 5 - Описание параметров GetMessageState

Параметр	Тип данных	Обязательность	Описание
Login	String	Да	Логин, полученный при регистрации
Password	String	Да	Пароль, соответствующий логину
messageId	String	Да	Идентификатор сообщения (сегментасообщения). Для одного запроса будет выполнен возврат статуса только одного сообщения (сегмента сообщения).

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <GetMessageStateResponse xmlns="http://ws.devinosms.com">
            <GetMessageStateResult>
                <State>int</State>
                <CreationDateUtc>dateTime</CreationDateUtc>
                <SubmittedDateUtc>dateTime</SubmittedDateUtc>
                <ReportedDateUtc>dateTime</ReportedDateUtc>
                <StateDescription>string</StateDescription>
                <Price>decimal</Price>
            </GetMessageStateResult>
        </GetMessageStateResponse>
   </soap12:Body>
</soap12:Envelope>
```

Таблица 6 - Описание возвращаемых параметров

Название	Тип	Описание
State	int	Статус. Типы статусов сообщений приведены в примечании.
CreationDateUtc	dateTime	Дата и время создания (пример 2010-0601T19:14:00) в UTC.
SubmittedDateUtc	dateTime	Время получения в Devino (в UTC).
ReportedDateUtc	dateTime	Время получения отчета (в UTC).
StateDescription	string	Описание статуса (напримерDescription(«Недопустимый адрес получателя»)).
Price	decimal	Цена

8.5 Получение статистики по SMS-рассылкам за заданный промежуток времени

Сервис возвращает статистику по SMS-рассылкам за период, в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата. Пример запроса:

```
POST /smsservicev2.asmx HTTP/1.1
Host: ws.devinotele.com
```

(continues on next page)

(продолжение с предыдущей страницы)

```
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <GetStatistics xmlns="http://ws.devinosms.com">
            <login>string</login>
            <password>string</password>
            <startDateTime>dateTime</startDateTime>
            <endDateTime>dateTime</endDateTime>
        </GetStatistics>
    </soap12:Body>
</soap12:Envelope>
```

Таблица 7 - Описание параметров GetStatistics

Параметр	Тип данных	Обязательность	Описание
Login	String	Да	Логин, полученный при регистрации
Password	String	Да	Пароль, соответствующий логину
startDateTime	DateTime	Да	Дата и время начала периода, закоторый необходимо получитьстатистику, например 2012-01-18T00:00:00. Время в UTC.
endDateTime	DateTime	Да	Дата и время конца периода, закоторый необходимо получить статистику, например 2012-01-18T23:59:00. Время в UTC.

После получения запроса сервис проверит валидность присланного по Login/Password и даты начала/окончания формирования статистики (включая ограничение на то, что охватываемый диапазон должен не превышать 3 месяцев). Если все проверки пройдены успешно, то сервис вернет статистику по sms со следующими параметрами:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <GetStatisticsResponse xmlns="http://ws.devinosms.com">
            <GetStatisticsResult>
                <Sent>int</Sent>
                <Delivered>int</Delivered>
                <Errors>int</Errors>
                <InProcess>int</InProcess>
                <Expired>int</Expired>
                <Rejected>int</Rejected>
            </GetStatisticsResult>
        </GetStatisticsResponse>
    </soap12:Body>
</soap12:Envelope>
```

Таблица 8 - Описание возвращаемых параметров

Название	Тип	Описание
Sent	int	Количество отправленных сообщений
Delivered	int	Количество доставленных сообщений.
Errors	int	Количество ошибок
InProcess	int	Количество сообщений «в процессе отправки»
Expired	int	Количество просроченных сообщений.
Rejected	int	Количество отклоненных сообщений

8.6 Получение входящих сообщений

Система позволяет заводить входящие номера и на них получать sms. Входящий номер заводится через личный кабинет. Сервис возвращает входящие сообщения пользователя в интервале maxDate minDate(который передан в этом запросе). Пример запроса:

```
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <GetIncomingMessages xmlns="http://ws.devinosms.com">
            <login>string</login>
            <password>string</password>
            <maxDateUTC>dateTime</maxDateUTC>
            <minDateUTC>dateTime</minDateUTC>
        </GetIncomingMessages>
    </soap12:Body>
</soap12:Envelope>
```

Таблица 9 - Описание параметров GetIncomingMessages

Параметр	Тип данных	Обязательность	Описание
Login	String	Да	Логин, полученный при регистрации
Password	String	Да	Пароль, соответствующий логину
maxDateUTC	DateTime	Да	Значение интервала _по. Пример: 2014-11-01T11:30
minDateUTC	DateTime	Да	Значение интервала **с** _. Пример: 2014-11-01T11:30 например 2012-01-18T23:59:00. Время в UTC.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
 <soap12:Body>
    <GetIncomingMessagesResponse xmlns="http://ws.devinosms.com">
        <GetIncomingMessagesResult>
            <IncomingMessage>
                <Data>string</Data>
```

(continues on next page)

(продолжение с предыдущей страницы)

```
                <SourceAddress>string</SourceAddress>
                <DestinationAddress>string</DestinationAddress>
                <CreatedDateUtc>dateTime</CreatedDateUtc>
            </IncomingMessage>
            <IncomingMessage>
                <Data>string</Data>
                <SourceAddress>string</SourceAddress>
                <DestinationAddress>string</DestinationAddress>
                <CreatedDateUtc>dateTime</CreatedDateUtc>
            </IncomingMessage>
        </GetIncomingMessagesResult>
    </GetIncomingMessagesResponse>
 </soap12:Body>
</soap12:Envelope>
```

Таблица 10 - Описание параметров GetIncomingMessages

Название	Тип	Описание
Data	String	Текст сообщения
SourceAddress	String	Адрес отправителя
DestinationAddress	String	Адрес получателя
CreatedDateUtc	DateTime	Дата создания

8.7 Отправка Viber-сообщений

8.7.1 Отправка Viber адресатам и возвращение системных идентификаторов сообщений

Данный метод поддерживает массовую отправку сообщений (до 1000 сообщений) в одном запросе.

Пример запроса:

```
POST /ViberServiceV2.asmx HTTP/1.1
Host: ws.devinotele.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <SendMessage xmlns="http://ws.devinosms.com">
            <login>string</login>
            <password>string</password>
            <message>
                <Data>string</Data>
                <DestinationAddresses>
                    <string>string</string>
                    <string>string</string>
                </DestinationAddresses>
                <SourceAddress>string</SourceAddress>
                <Validity>int</Validity>
                <Optional>string</Optional>
```

(continues on next page)

(продолжение с предыдущей страницы)

```
            </message>
        </SendMessage>
    </soap12:Body>
</soap12:Envelope>
```

Таблица 11 - Описание параметров SendMessage

Параметр	Тип данных	Обязательность	Описание
Login	String	Да	Логин, полученный при регистрации
Password	String	Да	Пароль, соответствующий логину
Data	String	Да	Текст сообщения, сообщение не должно быть длиннее 1000 символов
DestinationAddress	String[]	Да	Номер получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567, +79031234567, 89031234567
SourceAddress	String	Да	Адрес отправителя сообщения. До 11 латинских имволов или до 15 цифровых.
Validity	Int	Да	Время жизни сообщения (мин, от 1 до 1440)
Optional	String	Нет	Дополнительный параметр

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <SendMessageResponse xmlns="http://ws.devinosms.com">
            <SendMessageResult>
                <string>string</string>
                <string>string</string>
            </SendMessageResult>
        </SendMessageResponse>
    </soap12:Body>
</soap12:Envelope>
```

8.7.2 Отправка Viber адресатам и возвращение системных идентификаторов сообщений с переотправкой по sms

Данный метод поддерживает массовую отправку сообщений (до 1000 сообщений) в одном запросе.

Пример запроса:

```
POST /ViberServiceV2.asmx HTTP/1.1
Host: ws.devinotele.com
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

(continues on next page)

(продолжение с предыдущей страницы)

```
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <SendMessageWithResend xmlns="http://ws.devinosms.com">
            <login>string</login>
            <password>string</password>
            <message>
                <Data>string</Data>
                <DestinationAddresses>
                    <string>string</string>
                    <string>string</string>
                </DestinationAddresses>
                <SourceAddress>string</SourceAddress>
                <Validity>int</Validity>
                <Optional>string</Optional>
            </message>
        </SendMessageWithResend>
     </soap12:Body>
</soap12:Envelope>
```

Таблица 12 - Описание параметров SendMessageWithResend

Параметр	Тип данных	Обязательность	Описание
Login	String	Да	Логин, полученный при регистрации
Password	String	Да	Пароль, соответствующий логину
Data	String	Да	Текст сообщения, сообщение не должно быть длиннее 1000 символов
DestinationAddress	String []	Да	Номер получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567, +79031234567, 89031234567
SourceAddress	String	Да	Адрес отправителя сообщения. До 11 латинских имволов или до 15 цифровых.
Validity	Int	Да	Время жизни сообщения (мин, от 1 до 1440)
Optional	String	Нет	Дополнительный параметр

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <SendMessageWithResendResponse xmlns="http://ws.devinosms.com">
            <SendMessageResult>
                <string>string</string>
                <string>string</string>
            </SendMessageResult>
        </SendMessageWithResendResponse>
    </soap12:Body>
</soap12:Envelope>
```

8.8 Получение статуса отправленного Viber-сообщения

Сервис возвращает статус отправленного viber-сообщения в соответствии со значениями параметров по Login/Password и messageID.

Пример запроса:

```
POST /ViberServiceV2.asmx HTTP/1.1
Host: 127.0.0.1
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.
↪org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <GetMessageState xmlns="http://ws.devinosms.com">
            <login>string</login>
            <password>string</password>
            <messageID>long</messageID>
        </GetMessageState>
     </soap12:Body>
</soap12:Envelope>
```

Таблица 13 - Описание параметров GetMessageState

Параметр	Тип данных	Обязательность	Описание
Login	String	Да	Логин, полученный при регистрации
Password	String	Да	Пароль, соответствующий логину
messageId	String	Да	**Идентификатор сообщения. Для о** только одного сообщения.

Пример ответа:

```
HTTP/1.1 200 OK
Content-Type: application/soap+xml; charset=utf-8
Content-Length: length
<?xml version="1.0" encoding="utf-8"?>
<soap12:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.
↪org/2001/XMLSchema" xmlns:soap12="http://www.w3.org/2003/05/soap-envelope">
    <soap12:Body>
        <GetMessageStateResponse xmlns="http://ws.devinosms.com">
            <GetMessageStateResult>
                <State>Enqueued or Sent or Delivered or Read or Undelivered or Failed or Unknown␣
↪or Expired</State>
                <ResentSms>
                  <ViberSmsMessageStateInfo>
                    <Id>long</Id>
                  </ViberSmsMessageStateInfo>
                  <ViberSmsMessageStateInfo>
                    <Id>long</Id>
                  </ViberSmsMessageStateInfo>
```

(continues on next page)

(продолжение с предыдущей страницы)

```
            </ResentSms>
          </GetMessageStateResult>
        </GetMessageStateResponse>
    </soap12:Body>
</soap12:Envelope>
```

Таблица 14 - Описание возвращаемых параметров

Название	Тип	Описание
State	int	Статус. Типы статусов viber-сообщений приведены в примечании.
CreationDateUtc	DateTime	Дата и время создания (пример 2010-0601T19:14:00) в UTC.
SubmittedDateUtc	DateTime	Время получения в Devino (в UTC).
ReportedDateUtc	DateTime	Время получения отчета (в UTC).
StateDescription	string	Описание статуса (например Description (“Недопустимый адрес получателя”)).
Price	decimal	Цена
ResentSms	ViberSmsMessageStateInfo[]	Коллекция статусов sms-сообщений, которые были отправлены в результате отправки текущего viber-сообщения.

8.9 Приложение. Коды ошибок и статусы сообщений

БД Devino	Наименование	Описание	Подробное описание
-200	Ошибка	Errors=-200	Статус для фильтра «Ошибка» вдетализации
-100	Протарифицировано	Tarificated = -100	Статус для фильтра «Протирифицировано» в детализации
-3	Ошибка	ErrorSendingDateTimeInterpretation = -3	Ошибка интерпретации даты и времени отправки
-1	Отправлено	Sent = -1	Сообщение отправлено
-2	Отправляется	LocalQueued = -2	Сообщение отправляется
-40	Ожидание	Queued = -40	Сообщение в статусе «ожидание»
-30	Остановлено	Sending_To_Gateway = -30	Отправлено в шлюз
-20	Отправлено/получателю	Sending_To_Recipient = -20	Сообщение отправлено получателю
0	Доставлено	Delivered_To_Recipient = 0	Сообщение доставлено
0x0000000B	Ошибка	Error_Invalid_Destination_Address =0x0000000B	Неверно введён адрес получателя
0x0000000A	Ошибка	Error_Invalid_Source_Address =0x0000000A	Неверно введён адрес отправителя
41	Ошибка	Error_Incompatible_Destination = 41	Недопустимый адрес получателя
42	Ошибка	Error_Rejected = 42	Отклонено
46	Ошибка	Error_Expired = 46	Просрочен
47	Ошибка	Deleted = 47	Просрочено
48	Ошибка	Devino_Rejected = 48	Ошибка
0x000000FF	Неизвестный	Unknown = 0x000000FF	Внутренняя ошибка
0x00000008	Ошибка	System_Error = 0x00000008	Внутренняя ошибка

Глава 9

FTP/FTPS-протокол

9.1 Интеграция с использованием шаблона

9.1.1 Общие положения

Интеграционный сервис предоставляет возможность просто и быстро создавать sms-рассылки путем формирования и копирования файлов в формате csv. Интеграция осуществляется путем выкладывания Клиентом файла в определенную папку на FTP- сервер Платформы. Пароль на папку должен сообщаться Клиенту безопасным способом. Сервис платформы 1 раз в 10 секунд проверяет папку на наличие архивного файла. При обнаружении файла происходит его обработка, после чего он удаляется с FTP-сервера. Если в папке присутствует много архивов, то сервис начинает обработку самого раннего файла. Файлы обрабатываются по очереди.

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

9.1.2 Точка доступа

FTP – сервер Платформы располагается по адресу: **ftp1.integrationapi.net**

9.1.3 Техническая часть

9.1.4 Формат приема сообщений по FTP

Формат названия архива: [логин_Клиента]_[YYYYMMDDhhmmss].zip
Пример: guest_20120207231932.zip

Файлы для отправки должны быть упакованы в архив .zip без пароля и находиться в папке [sms]. Папка [sms] должна содержать два файла формата csv (данные разделяются символом «;»). Кодировка всех файлов по умолчанию - UTF-8. При необходимости выкладывать файлы в другой кодировке, обратитесь в техподдержку. Структура архива:

- **sms**
  - contacts.csv
  - template.csv

Первый файл должен называться contacts.csv

Наименование столбца	Обязательность	Описание
phone	Да	Номера получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567
senddate	Нет	Дата и время отправки сообщения в формате YYYYMMDDThh:mm:ss (локальное время Клиента). Если сообщение нужно отправить сразу после получения файла, то поле должно содержать пустое значение. Значение этого поля является более приоритетным по сравнению со значением из файла template.csv. Если поле не заполнено, то при отправке должно браться значение из файла template.csv. Может учитывать часовой пояс абонента, если заполнено поле localtime в файле template.csv. Пример: 2010-06-01T19:14:00
attr1	Нет	Текстовое поле для подстановки в шаблон.
attrN	Нет	N-е текстовое поле для подстановки в шаблон.

Второй файл должен называться template.csv (в файле должна быть только одна строка с данными).

Наименование столбца	Обязательность	Описание
message	Да	Шаблон сообщения с полями для подстановки. Название поля подстановки должно быть заключено в символ #. Данные подставляются из соответствующего столбца в файле contacts.csv. Пример: Уважаемый #attr1# #attr2# #attr3#! Если соответствующее значение в файле contacts.csv не найдено, то поле подстановки заменяется на пустое значение
sourceaddress	Да	Адрес отправителя сообщения. До 11 латинских символов или до 15 цифровых. Предварительно должен быть запрошен через ЛК и подтвержден техподдержкой. Пример: TESTSMS (регистр имеет значение).
validity	Нет	Время жизни сообщения, устанавливается в минутах. Пример: 180 (по умолчанию подставляется 1440 = 24 часа)
senddate	Нет	Дата и время отправки сообщения в формате YYYYMMDDThh:mm:ss (локальное время Клиента). Если сообщение нужно отправить сразу после получения файла, то поле должно содержать пустое значение. Поле может переопределяться значением для конкретного абонента из файла contacts.csv. Пример: 2010-06-01T19:14:00
localtime	Нет	Учитывать часовой пояс абонента. 1 - учитывать,0 – нет. Применяется только если указана дата отправки в файле template.csv, либо непосредственно у контакта в файле contacts.csv. Часовой пояс определяется автоматически по номеру абонента. Пример: 1

Предупреждение: Не заполненных столбцов ни в одном из файлов не должно быть. Если нет необходимости заполнять данными какие-то столбцы, то и сами столбцы добавлять в файл не нужно!

9.1.5 Спецификация запросов

Для работы с FTP-сервером платформы существует два варианта:

- Работа через FTP-клиент (например, Windows Explorer)
- Работа через командную строку Windows или через Telnet

Работа через FTP-клиент (Например, Windows Explorer). FTP-клиентом, встроенным в ОС Windows, является Windows Explorer. В нем работа с FTP-архивами практически не отличается от работы с файлами на компьютере.

- Сначала необходимо открыть окно Windows Explorer и установить соединение с FTP сервером. Для этого в строке адреса нужно ввести ftp.integrationapi.net
- После установки соединения Windows Explorer запросит пароль, соответствующий выданному логину.

Работа через командную строку Windows или через Telnet. Для работы с FTP необходимо ввести в командной строке: C:ftp ftp.integrationapi.net После подключения к данному серверу необходимо пройти следующие обязательные этапы:

- Идентификация (ввод имени-идентификатора и пароля).
- Выбор каталога.

- Определение режима обмена (поблочный, поточный, ASCII или двоичный).
- Выполнение команд обмена (get, mget, dir, mdel, mput или put).
- Завершение процедуры (quit или close).

На первом этапе необходимо ввести свои учетные данные. Управление доступом осуществляется с помощью команд: * USER - имя пользователя * PASS - пароль * CWD - имя новой рабочей директории * CDUP - перейти на один уровень директории вверх * QUIT – выход

Также необходимо определиться с параметрами передачи данных: PORT ip1,ip2,ip3,ip4,p1,p2 - IP адрес клиента (ip1,ip2,ip3,ip4) и порт (p1,p2) (расчет порта p1*256+p2=номер порта). Пример:

```
Entering Passive Mode (194,87,5,52,9,79)
194.87.5.52 - IP адрес
2383 - номер порта, расчет порта 9*256+79=2383
PASV - сервер должен определить нестандартный порт данных, начать его слушать и вернуть ip-
адрес и номер порта в формате PORT.
TYPE { { A | E } [ N | T | C ] } | I | L размер-байта (по умолчанию - A N) - специфицирует тип
информации.
```

Для копирования файла из удаленного сервера используется команда GET, для копирования группы файлов - MGET. Аналогом команды GET в какой-то степени является команда DIR (ls), только она переносит содержимое каталога, что для некоторых операционных систем эквивалентно. При использовании модификации mget проявляйте осторожность - вы можете заблокировать телекоммуникационный канал длительным копированием. Для записи файла в удаленный сервер применяется команда PUT. При операциях обмена обычно используется текущий каталог локальной ЭВМ. Статистику по рассылкам Клиент может посмотреть в своем Личном Кабинете.

9.2 Интеграция без шаблона

9.2.1 Общие положения

Интеграционный сервис предоставляет возможность просто и быстро создавать sms-рассылки путем формирования и копирования файлов в формате csv или txt. Интеграция осуществляется путем выкладывания Клиентом файла в определенную папку на FTP- сервер Платформы. Пароль на папку должен сообщаться Клиенту безопасным способом. Сервис Платформы 1 раз в 10 секунд проверяет папку на наличие архивного файла. При обнаружении файла происходит его обработка, после чего он удаляется с FTP-сервера. Если в папке присутствует много архивов, то сервис начинает обработку самого раннего файла. Файлы обрабатываются по очереди.

9.2.2 Точка доступа

FTP – сервер Платформы располагается по адресу: **ftp1.integrationapi.net**

9.2.3 Техническая часть

ФОРМАТ ПРИЕМА СООБЩЕНИЙ ПО FTP

Формат названия архива: [логин_Клиента]_[YYYYMMDDhhmmss].zip Пример: guest_20120207231932.zip Файл для отправки должен быть упакован в архив .zip. Архив должен содержать один файл формата txt или csv (данные разделяются символом «;»). Кодировка всех файлов по умолчанию - UTF-8. Структура архива: * sms o name.csv (либо name.txt) В txt (или csv)

– файле строки, начиная со второй, содержат данные для отправки сообщений (одна строка – одно сообщение).

Состав полей в файле с данными для отправки сообщений

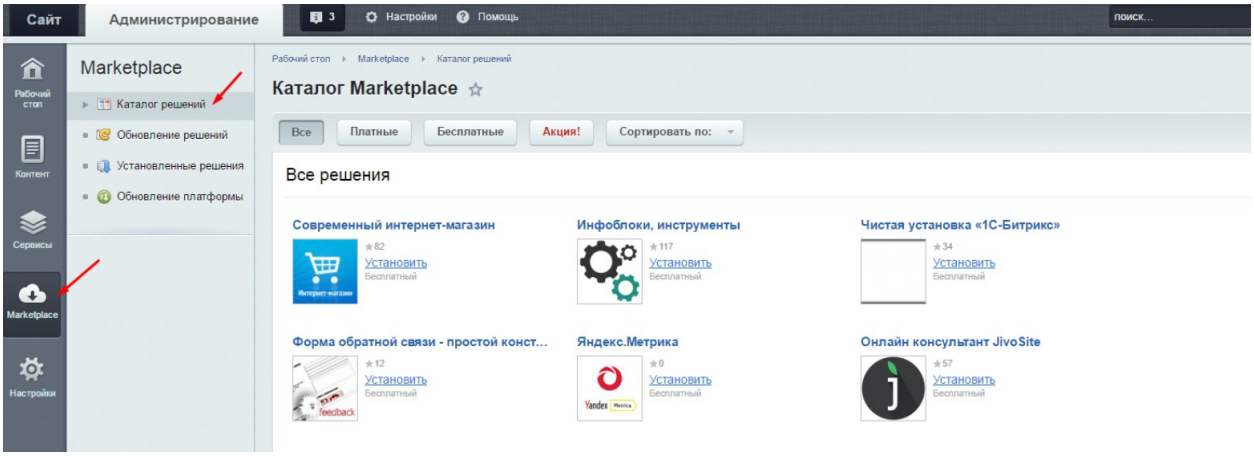
Наименование столбца	Обязательность	Описание
TELNR_LONG	Да	Номера получателя сообщения в международном формате: код страны + код сети + номер телефона. Пример: 79031234567
TEXT	Да	Текст сообщения

- Адрес отправителя задается по умолчанию, необходимо заранее сообщить его в техподдержку support@devinotele.com
- Время жизни сообщения, по умолчанию, составляет 24 часа. При необходимости изменить время жизни сообщения, также необходимо обратиться в техподдержку.
- Отправка по часовым поясам и отложенная отправка в данном виде интеграции не предусмотрена.
- Статистику по рассылкам Клиент может посмотреть в своем Личном Кабинете.

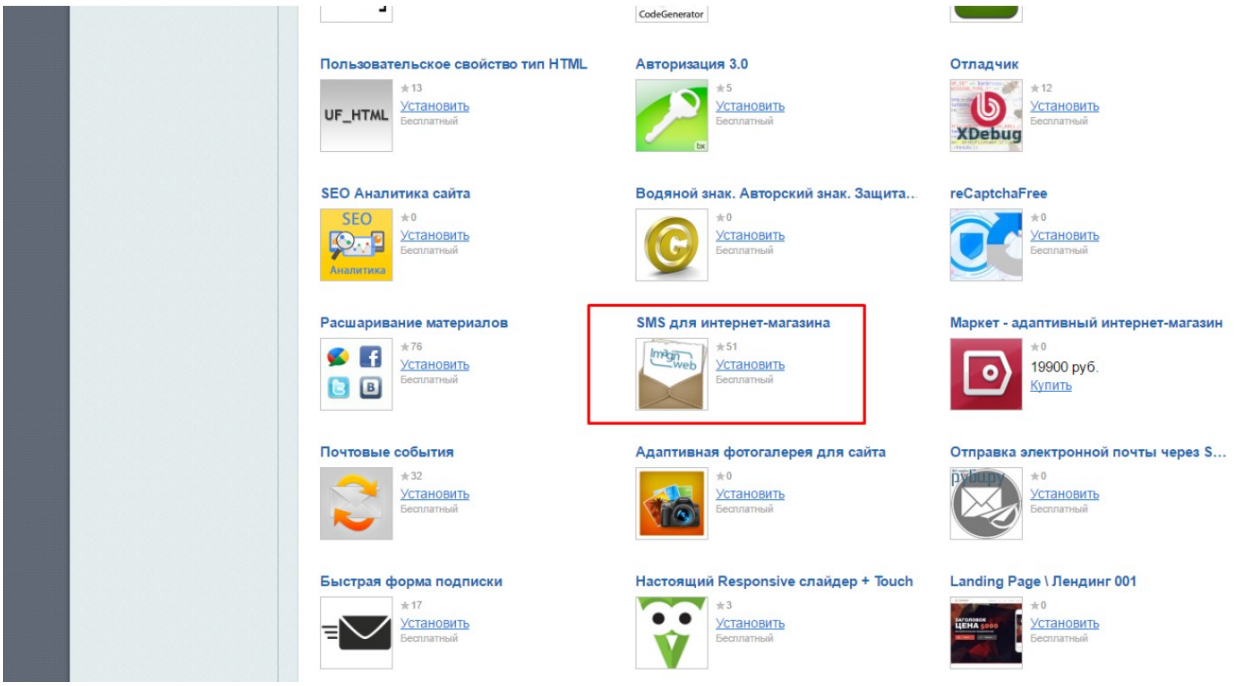
Глава 10

Модуль SMS рассылки с 1С Битрикс

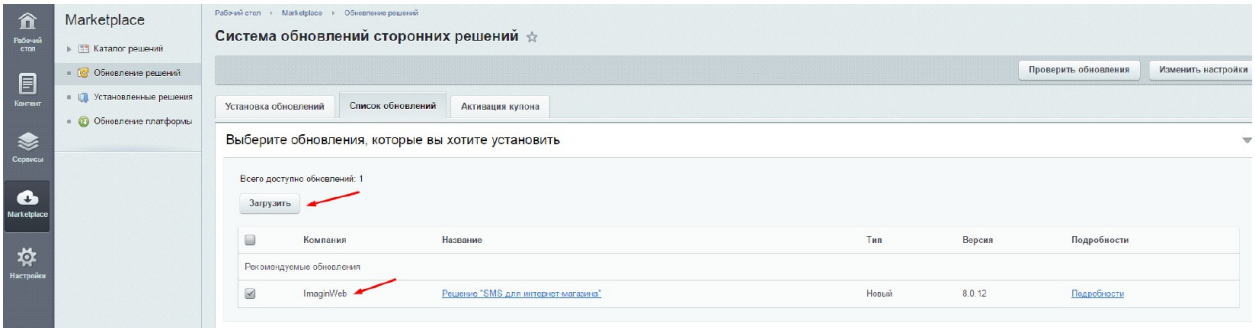
Для установки модуля необходимо перейти в Каталог решений на Marketplace как показано на скриншоте, либо перейти по ссылке: https://marketplace.1c-bitrix.ru/solutions/imaginweb.sms/

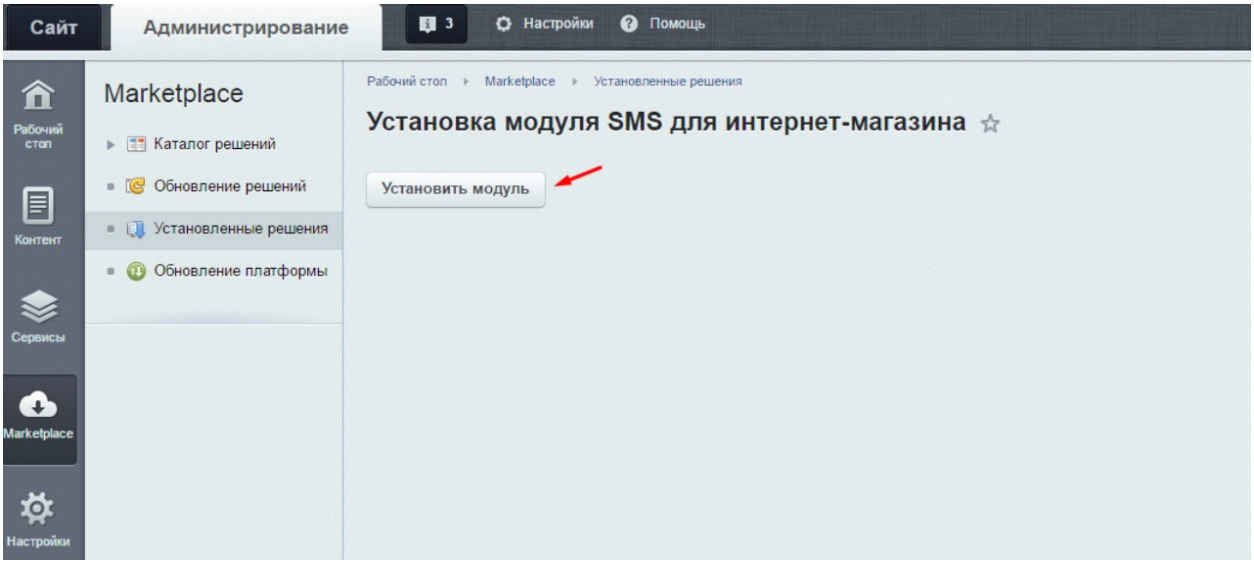


В перечне готовых модулей необходимо найти модуль SMS для интернет-магазина от компании ImaginWeb и установить его.

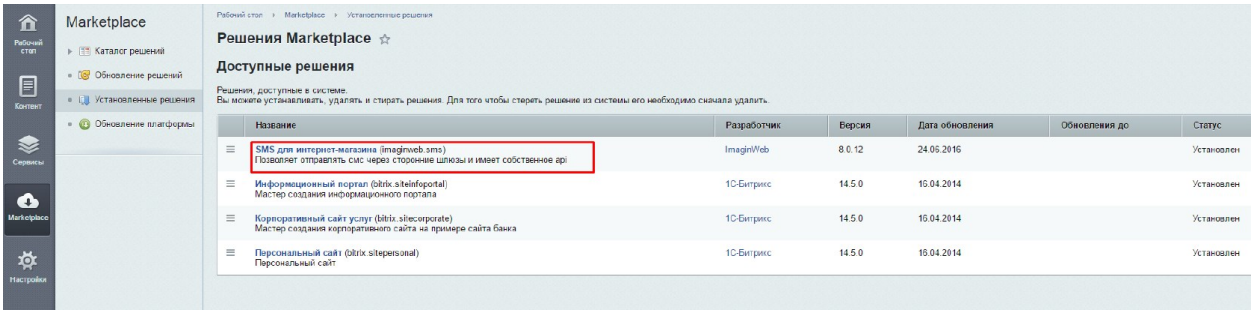


Производим установку модуля для 1С-Битрикс как показано на скриншотах ниже.

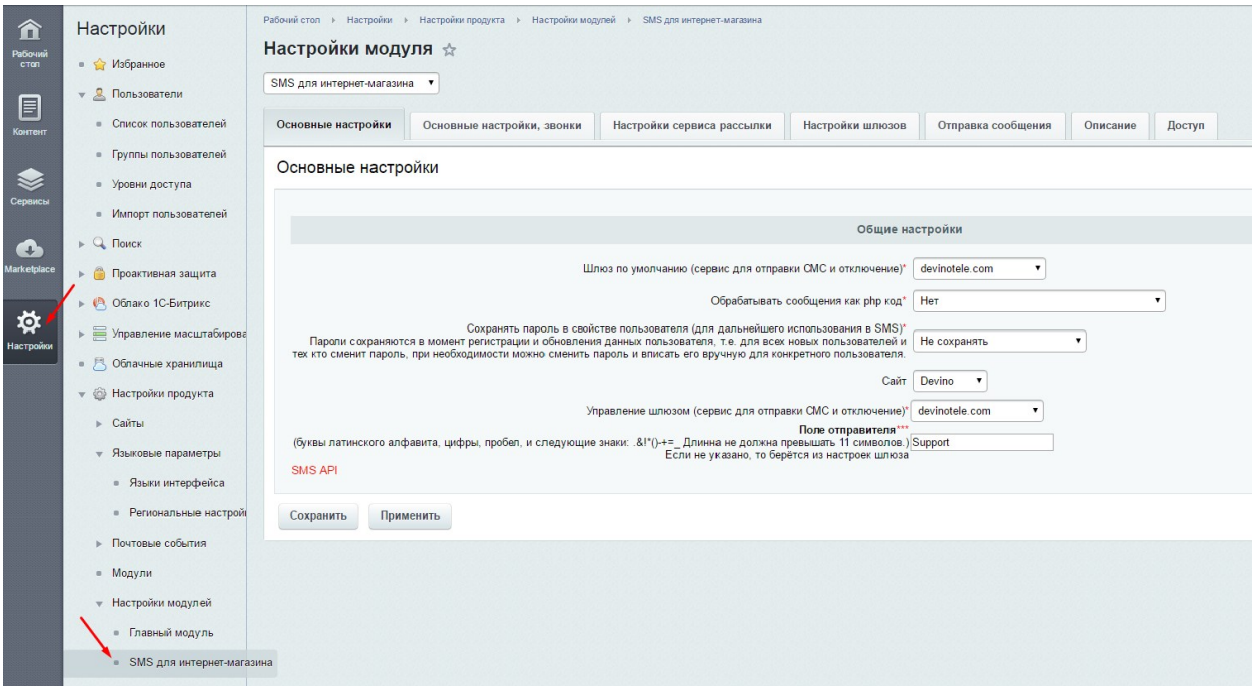




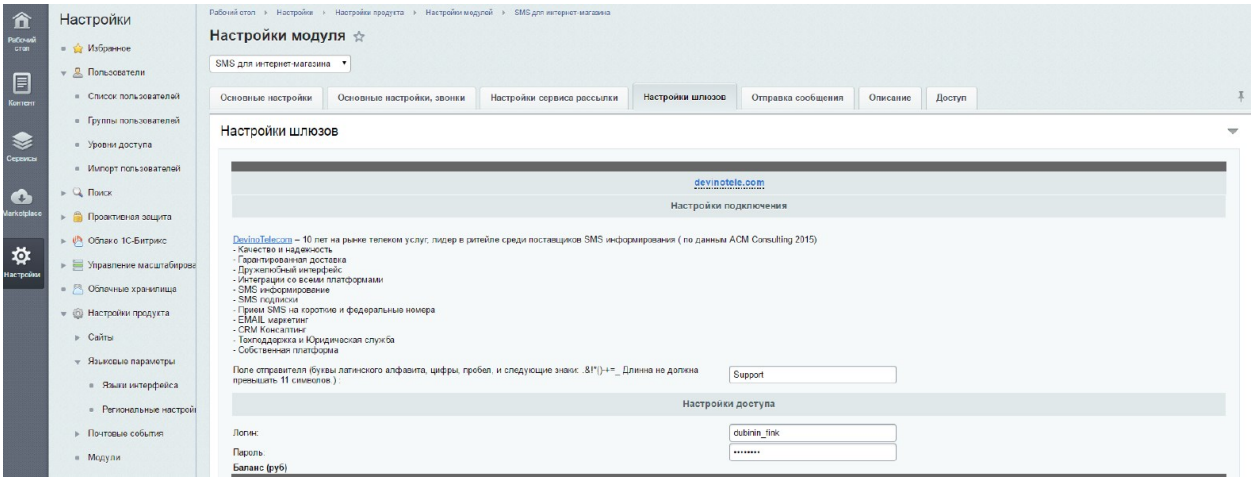
Теперь, когда модуль установлен и появился в списке Установленных решений, переходим к настройке модуля.



Заходим в Настройки и выбираем модуль SMS для интернет-магазина



В Основных настройках выбираем шлюз Devinotele.com, вводим в Поле отправителя Адрес отправителя, от которого будут отправляться смс. Далее переходим во вкладку Настройка шлюзов и вводим настройки для шлюза Devinotele.com. В Поле отправителя вводим Адрес отправителя, от которого будут отправляться смс, в полях Логин/Пароль вводим свой логин и пароль от вашего ЛК в devino telecom.



Сохраняем все изменения. После сохранения в поле Баланс должен отобразиться размер баланса на вашем логине. Для проверки работоспособности шлюза выберете свойство заказа содержащее телефон. Добавьте текст сообщения, можно использовать по шаблону #ORDER_NUMBER# - номер заказа, #ORDER_SUMM# - сумма заказа, #PRICE_DELIVERY# - стоимость доставки, #PRICE# - полная стоимость. Сделайте заказ и убедитесь в работоспособности сервиса.

Глава 11

Работа с входящими SMS сообщениями (HTTP для приёма)

11.1 Назначение документа

Документ предназначен для использования в качества руководства для разработчиков и сотрудников технической поддержки. Руководство содержит описание функционала работы с входящими SMS-сообщениями. Функционал позволяет осуществлять:

- Получение входящих SMS-сообщений;
- Отображение входящих SMS-сообщений за определенный период в «Личном кабинете»;
- Отправку ответного SMS, включая:
- Отправку общего ответного SMS-сообщения;
- Отправку индивидуального ответного SMS-сообщения;
- Отображение отправленных ответных SMS в «Личном кабинете»;
- Отображение перечня входящих номеров в «Личном кабинете»;
- Создание Клиентом группы для входящих сообщений в «Личном кабинете»;
- Экспорт входящих сообщений в файл формата *.csv.

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

11.2 Настройка приёма SMS-сообщения

11.2.1 Общие положения

«Прием входящих SMS» - это услуга, позволяющая получить обратную связь от Абонентов Клиента через SMS сообщения на закрепленный за Клиентом номер. На сообщение, присланное Абонентом, можно отправить Ответное сообщение. Клиент может осуществлять «Прием входящих SMS» следующими способами:

- По протоколу http через API «Сервиса доставки SMS» платформы;
- Передачей входящего сообщения по протоколу http на внешний обработчик Клиента;
- По протоколу SMPP.

Для начала использования услуги получения sms-сообщений от Абонентов Клиенту необходимо: 1. Зарегистрироваться в Личном кабинете; 2. Заключить договор; 3. Подать заявку на подключение входящего номера, указав при этом необходимые параметры:

- Логин;
- Номер;
- Текст ответного сообщения (опционально, если необходимо отправлять Ответные сообщения);
- URL для приема входящих SMS-сообщений (опционально, если необходимо отправлять входящие SMS по протоколу http внешнему обработчику);
- Примечание (опционально, примечание к заявке на подключение нового входящего номера).

11.2.2 По протоколу HTTP

Возможно получение входящих sms-сообщений за период по протоколу HTTP с использованием предоставляемого API платформы. Для этого необходимо пройти процедуру предварительной аутентификации и получить идентификатор сессии. После этого платформа вернет данные входящих SMS в соответствии со значениями параметров, которые необходимо передать ей в GET-запросе следующего формата:

```
application/x-www-form-urlencoded
https://integrationapi.net/rest/Sms/In?
sessionId=<Идентификатор сессии>&
minDateUTC=<Дата и время начала периода>&
maxDateUTC=<Дата и время окончания периода>
```

Ниже приведен пример запроса:

```
application/x-www-form-urlencoded
https://integrationapi.net/rest/Sms/In?
sessionId=Z5CYSZEKDL1DPICU37WEHQVOYKP0T1GSLHX1&
minDateUTC=2011-01-01T00:00:00&
maxDateUTC=2011-01-11T00:00:00
```

Параметр	Тип данных	Описание
Обязательные		
sessionId	String	Идентификатор сессии (36 символов). Обязательный.
maxDateUTC	DateTime	Дата и время окончания периода, за который происходит выборка входящих сообщений (например, 2010-06-02T19:14:00).
Необязательные		
minDateUTC	DateTime	Дата и время начала периода, за который происходит выборка входящих сообщений (например, 2010-06-01T19:14:00).

После получения запроса платформа проверит валидность идентификатора сессии, даты-времени начала и окончания периода выборки. Если все проверки пройдены успешно, то платформа вернет перечень сообщений и их параметров за период в json-файла следующего формата:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
[{"Data":<Текст сообщения>,
"SourceAddress":<Адрес отправителя>,
"DestinationAddress":<Номер получателя>,
"ID":<Идентификатор сообщения>,
"CreatedDateUtc":<Дата создания>}]
```

Например:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
[{"Data":"test1",
"SourceAddress":"79260000000",
"DestinationAddress":"79160000000",
"ID":539187174,
"CreatedDateUtc":"\/Date(1294045911213)\/"},
{"Data":"test2",
"SourceAddress":"79260000001",
"DestinationAddress":"79160000000",
"ID":539187214,
"CreatedDateUtc":"\/Date(1294045911353)\/"}]
```

Если какая-нибудь проверка не проходит успешно, то платформа возвращает Код ошибки в виде JSON следующего формата:

```
{
    Code: <Код ошибки>
    Desc: <''Текст ошибки''>
}
```

Например:

```
{
    Code: 9
    Desc: "The parameters dictionary contains a null entry for parameter
    'maxDateUtc' of non-nullable type 'DateTime' for method
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    'System.Web.Mvc.ActionResult In(System.String, DateTime, DateTime)' in
    'RestService.Controllers.SmsController'. An optional parameter must be a
    reference type, a nullable type, or be declared as an optional parameter.
    Parameter name: parameters"
}
```

11.2.3 Через внешний обработчик

Возможен прием входящих SMS сообщений на короткие номера через внешний обработчик. Для этого Клиент при создании запроса на подключение входящего номера должен указать адрес обработчика (параметр «URL для приема входящих SMS-сообщений»). Если внешний обработчик не отвечает в течении 30 секунд - будет выполнена повторная попытка отправки. Если обработчик возвращает ошибку - платформа повторяет попытку отправить запрос через 300 секунд.

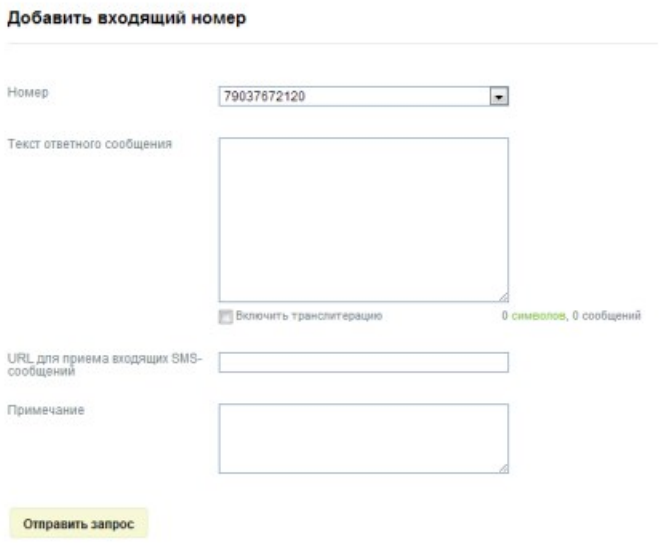


Рис. 1. Создание запроса на добавление входящего номера в «Личном кабинете»

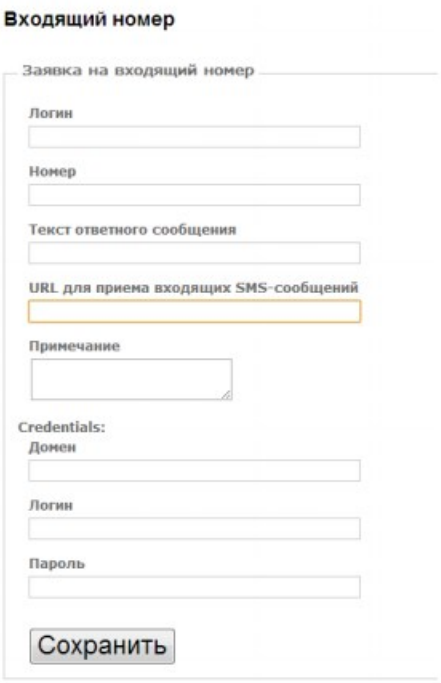


Рис. 2. Создание запроса на добавление входящего номера в подсистеме администрирования

11.2.4 По протоколу SMPP

Возможен прием входящих SMS сообщений по протоколу SMPP на заранее согласованный с менеджером номер посредством приема пакетов Deliver_SM в рамках открытой SMPP сессии.

11.3 Настройка отправки ответного SMS-сообщения

11.3.1 Общие положения

«Ответное SMS-сообщение» - это услуга, позволяющая отправлять ответное сообщение Абоненту, приславшему сообщение на короткий или федеральный номер.

11.3.2 Общее ответное SMS

Для подключение услуги «Общее ответное SMS» Клиенту необходимо создать запрос. В запросе должен быть указан текст общего ответного SMS-сообщения. Если текст указан, то после подтверждения заявки Менеджером и активации услуги все Абоненты, приславшие SMS сообщение на входящий номер Клиента, получат ответное SMS-сообщение с текстом, указанным в запросе.

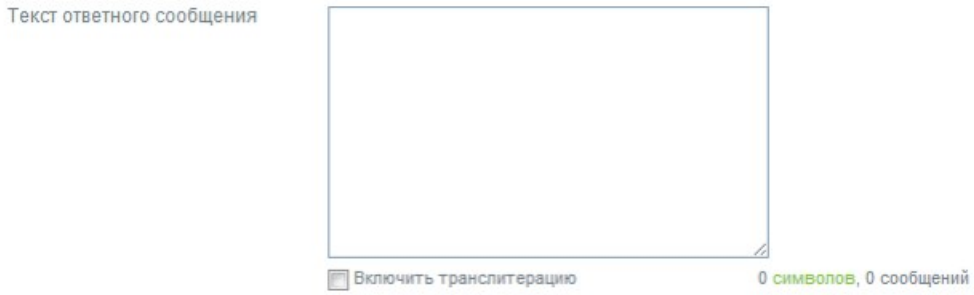


Рис. 3. Поле для ввода текста общего ответного SMS-сообщения при создании заявки на добавление нового «Входящего номера» через «Личный кабинет»

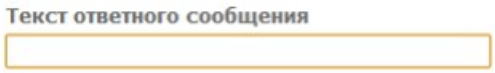


Рис. 4. Поле для ввода текста общего ответного SMS-сообщения при создании заявки на добавление нового «Входящего номера» через подсистему администрирования

11.3.3 Индивидуальное ответное SMS

Если входящие SMS-сообщения передаются на внешний обработчик, становится возможна отправка индивидуального ответного SMS-сообщения. Для этого, когда платформа посылает запрос внешнему обработчику со следующими параметрами:

```
application/x-www-form-urlencoded
<Адрес внешнего обработчика (ExternalHandler)>?
sourcePhone=<Номер Абонента, которому требуется ответить>&
number=<Номер с которого будет прислан ответ>&
prefix=<Префикс>&
text=<Текст>&
messageId=<Идентификатор сообщения>
```

Например:

```
application/x-www-form-urlencoded
<Адрес внешнего обработчика (ExternalHandler)>?
sourcePhone=79089876534&
number=2435&
prefix= loto&
text= Играйте с нами в лото!&
messageId=235515
```

Внешнему обработчику необходимо отправить ответ платформе (в течение 1 минуты) со следующими параметрами:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: text/plain; charset=utf-8
{sms=<Текст ответного сообщения>}
```

Если платформа получит ответ, то абонент с номером «sourcePhone» (из запроса платформы) получит ответное SMS-сообщение с номера отправителя «number» (из запроса платформы) и текстом «sms» (из ответа внешнего обработчика).

Глава 12

Viber HTTP API

12.1 Общие сведения

API предоставляет интерфейс для автоматизации процесса отправки мгновенных сообщений Viber мессенджера через платформу Devino Telecom. С помощью API можно производить пакетные рассылки, отправлять транзакционные сообщения, а также осуществлять переотправку SMS в случае недоставки Viber сообщений одной командой.

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

Работа с API осуществляется с использованием метода HTTP-POST. Тела запроса клиента и ответа сервера представлены в формате JSON. Данные должны быть в кодировке UTF-8.

Данным методом можно отправлять не более 100 сообщений.

API поддерживает базовую авторизацию через заголовок Authorization (https://en.wikipedia.org/wiki/Basic_access_authentication). В заголовке запроса необходимо передать логин и пароль из Личного Кабинета в формате login:password в base64 кодировке.

Например:

```
Authorization: Basic dGVzdGVyOjExMTExMQ==
```

Точка подключения:

```
https://viber.devinotele.com:444
```

Используемые типы данных:

Тип данных	Описание
Integer	Положительные числа больше нуля
DateTime	Значение, обозначающее дату и время, представляются в пакетах в формате «yyyy-MMdd hh:mm:ss», например, 1999-05-31 13:20:00
Address	Адрес абонента. Номер мобильного телефона абонента в международном формате (в формате E.164). Например, 79031112233, для России или 491791112233 для Германии
String	Строка символов в формате UTF-8

Ограничения и допустимые значения описаны далее в разделе описания пакетов протокола

12.2 Отправка сообщения

```
https://viber.devinotele.com:444/send
```

Метод служит для отправки сообщения на указанные номера абонента. Параметры сообщения и адреса абонентов передаются в теле запроса в формате JSON.

Типы сообщений, допустимые к отправке:

Тип сообщения	Возможность переотправки	contentType
Текст+кнопка	Да	button
Текст+картинка+кнопка	Да	button
Текст	Да	text
Картинка	Нет	image

12.2.1 Отправка текстового сообщений

Запрос на отправку текстового сообщения собирается с указанием contentType: «text». Возможна переотправка по СМС.

Пример запроса:

https://viber.devinotele.com:444/send

```
{
    "resendSms" : "true",
    "messages" :
        [ {
            "subject" : "Subject",
            "priority" : "high",
            "validityPeriodSec" : 3600,
            "comment" : "comment",
            "type" : "viber",
            "contentType" : "text",
            "content" :
            {
                 "text" : "Message text"
            },
            "address" : "79250000000",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        "smsText":"1sms Message text",
        "smsSrcAddress":"1TEST",
        "smsValidityPeriodSec":5000
    } ]
}
```

Описание полей тела запроса отправки сообщения:

Параметр	Тип данных	Описание	Допустимые значения	Обязательное поле
resendSms	boolean	Признак переотправки сообщения, по умолчанию (если параметр не передаётся) - переотправка выключена	**true** – переотправка включена **false**– переотправка выключена	Нет
subject	String	Подпись для сообщения, которая отображается в мессенджере абонента	Все подписи предварительно должны регистрироваться на платформе провайдера Длина имени не более 11 символов.	да
priority	String	Приоритет сообщения. Используется для управления оперативностью доставки сообщения абоненту. Для транзакционных сообщений приоритет должен быть высоким, для рекламы низким.	low – низкий приоритет. normal – нормальный приоритет high – высокий приоритет. realtime – высочайший приоритет	Да
validityPeriodSec	Integer	Время ожидания доставки Viber сообщения в секундах	30 – 86400.	Да
comment	String	Произвольный текстовый комментарий.		Нет
type	String	Тип отправляемого сообщения. Определяет канал, которые используется для доставки сообщения на мобильный телефон абонента	viber – Viber messenger	Да
contentType	String	Тип содержимого сообщения.	text – текстовое сообщение image – изображение button – гиперссылка в виде кнопки	Да
content	Составной тип	Содержимое сообщения. Зависит от значения contentType	Определяется значением contentType	Да
address	Address	Номер телефона абонента, на который отправляется сообщение	Положительные целые числа. Номер мобильного телефона абонента в международном формате (в формате E.164)	Да
smsText	String	Текст СМС сообщения		Да
smsSrcAddress	String	Адрес отправителя СМС сообщения	Адрес отправителя должен быть согласован на СМС в личном кабинете, длина имени не более 11 латинский символов или цифр.	Да
smsValidityPeriodSec	Integer	Время ожидания доставки СМС сообщения в секундах	60 – 86400. Если параметр не указан, то время жизни сообщения будет выставлено по-умолчанию СМС-центром оператора.	Да

Пример ответа:

```
{
    "status" : "ok"
    "messages" :
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        [ {
            "providerId" : 3158611117333282816,
            "code" : "ok"
        } ]
}
```

Описание полей ответа на запрос отправки сообщения:

Параметр	Тип данных	Описание	Допустимые значения	Обязательное поле
status	String	Статус ответа провайдера на запрос send	Список возможных кодов и их значений указан в таблице кодов возврата	Да
providerId	Integer	Поле возвращается только в случае когда код ответа провайдера для сообщения равен “ok”. На стороне клиента providerId должно сохраняться для последующего запроса статуса сообщения.	Положительные целые числа	Нет
code	String	Код ответа провайдера для конкретного сообщения	Список возможных кодов и их значений указан в таблице кодов возврата	Да

12.2.2 Отправка текста с кнопкой

Запрос для отправки абоненту текста с кнопкой в качестве сообщения отличается от запроса для отправки простого текстового сообщения кодом contentType, в котором в данном случае нужно указать значение button и заполнить дополнительные атрибуты text, caption, action и imageUrl (при необходимости добавить изображение) составного поля content. Данный тип сообщений поддерживается только в Viber. Возможна переотправка СМС.

Пример запроса отправки кнопки:

https://viber.devinotele.com:444/send

```
{
    "resendSms" : "true",
    "messages" :
    [ {
        "subject" : "Subject",
        "priority" : "high",
        "validityPeriodSec" : 3600,
        "comment" : "comment",
        "type" : "viber",
        "contentType" : "button",
        "content" : {
            "text" : "text",
            "caption" : "caption",
            "action" : "http://company.com/resource",
            "imageUrl" : "http://company.com/image.jpg"
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        },
        "address" : "79250000000",
        "smsText":"1sms Message text",
        "smsSrcAddress":"1TEST",
        "smsValidityPeriodSec":5000
    } ]
}
```

Описание полей содержимого для отправки кнопки:

Параметр	Тип данных	Описание	Обязательное поле
text	String	Текст сообщения. Не менее 2 и не более 1000 символов.	Да
caption	String	Наименование кнопки. Не более 30 символов.	Да
action	String	URL страницы, на которую будет отправлен пользователь при нажатии на кнопку	Да
imageUrl	String	URL изображения, которое размещено на серверах Клиента	Нет

12.2.3 Отправка изображения

Запрос для отправки абоненту изображения отличается от запроса для отправки текстового сообщения кодом contentType, в котором в данном случае нужно указать значение image и заполнить дополнительный атрибут imageUrl для составного параметра content. Переотправка не предполагается, т.к. отсутствует поле text. В случае указания resendSms = true для отправки image сервис возвращает ошибку валидации

Пример запроса отправки изображения:

https://viber.devinotele.com:444/send

```
{
    "resendSms" : "false",
    "messages" :
    [ {
        "subject" : "Subject",
        "priority" : "high",
        "validityPeriodSec" : 3600,
        "comment" : "comment",
        "type" : "viber",
        "contentType" : "image",
        "content" : {
            "imageUrl" : "http://company.com/image.jpg"
        },
        "address" : "79250000000"
    } ]
}
```

Описание полей содержимого отправки изображения:

Параметр	Тип данных	Описание	Обязательное поле
image	String	URL изображения	Да

12.2.4 Отправка нескольких сообщений

При осуществлении массовой рассылки однотипных сообщений, чтобы не дублировать данные, можно использовать секцию запроса messageCommonData, данные из которой будут использованы для всех сообщений в запросе, но могут быть переопределены ими.

Пример отправки нескольких сообщений:

https://viber.devinotele.com:444/send

```
{
    "resendSms" : "false",
    "commonData" : {
        "subject" : "Subject",
        "priority" : "high",
        "validityPeriodSec" : 3600,
        "comment" : "comment",
        "type" : "viber",
        "contentType" : "button",
        "content" : {
            "text" : "text",
            "caption" : "caption",
            "action" : "http://company.com/resource",
            "imageUrl" : "http://company.com/image.jpg"
        }
    },
    "messages" :
        [ {
            "address" : "79250000001"
        },
        {
            "priority" : "low",
            "content" : {
                "text" : "Message text"
            },
            "address" : "79250000002"
    } ]
}
```

В данном примере второе сообщение будет отправлено с текстом «Message text» и с более низким приоритетом.

12.3 Проверка статуса доставки сообщения

```
https://viber.devinotele.com:444/status
```

Данный метод предназначен для проверки статусов по ранее полученным providerId на запросы «/send» В одном запросе можно передавать не более 100 ID сообщений. Статусы по сообщениям можно запрашивать в течении 5 дней с даты отпарвки.

Пример запроса:

https://viber.devinotele.com:444/status

```
{
   "messages" :
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        [3158611117333282816, 3158611117333282817, 3158611117333282818]
}
```

Пример ответа на запрос статуса доставки:

```
{

   "status": "ok",
   "messages": [
       {
           "providerId": 3158611117333282816,
           "code": "ok",
           "smsStates": [
               {
                   "id": 583465579822710784,
                   "state": "delivered"
               },
               {
                   "id": 583465579822710785,
                   "state": "delivered"
               }
           ]
       },
       {
           "providerId": 3158611117333282817,
           "code": "ok",
           "status": "read",
           "statusAt": "2016-08-10 15:28:50"
       },
       {
           "providerId": 3158611117333282818,
           "code": "ok",
           "smsStates": [
               {
                   "id": 583465579822710798,
                   "status": "delivered"
               }
           ]
       }
   ]
}
```

Описание полей ответа на запрос статуса доставки

Параметр	Тип данных	Описание	Допустимые значения	Обязательное поле
status	String	Результат обработки запроса	Возможные коды ошибок и их описание определены в таблице кодов возврата	Да
code	String	Результат обработки запроса для конкретного сообщения с провайдеским идентификатором	Возможные коды ошибок и их описание определены в таблице кодов возврата	Да
smsStatus	Массив (Составное поле)	Текущий статус доставки СМС сообщения. Указывается, только если была переотправка сообщения.		Нет
smsStatus.status	String	Код статуса доставки СМС сообщения	**enqueued** – сообщение находится в очереди на отправку. **sent** – сообщение отправлено абоненту **delivered** – сообщение доставлено абоненту. **undelivered** – сообщение отправлено, но не доставлено абоненту.	Нет
smsStatus.smsid	Long	ID СМС сообщения с СМС-Центра провайдера. Если сообщение многосегментное, то будет возвращен ID для каждого сегмента сообщения и его статус.		Да
Status	String	Код статуса доставки Viber сообщения.	**enqueued** – сообщение находится в очереди на отправку. **sent** – сообщение отправлено абоненту **delivered** – сообщение доставлено абоненту. **read** – сообщение просмотрено абонентом. **visited** абонент перешел по ссылке в сообщении. **undelivered** – сообщение отправлено, но не доставлено абоненту. **failed** – сообщение не было отправлено в результат сбоя. **cancelled** –отправка сообщения отменена. **vp_expired** – сообщение просрочено, финальный статус не получен в рамках заданного validity period	Да
statusDate	DateTime	Дата и время получения статуса по UTC		Да
errorCode	String	Причина, по которой сообщение не было доставлено абоненту (status=undelivered)	**user-blocked** – абонент заблокирован **not-viber-user** – абонент не является пользователем Viber.	Нет

12.4 Прием статусов с помощью callback-запросов

Данный метод позволяет не обращаться к API Devino каждый раз, когда требуется получить статус доставки сообщения, а обрабатывать входящие события от платформы Devino на своем внутреннем ресурсе.

При получении статуса сообщения от Viber платформа Devino отправляет HTTP-POST запрос (JSON, UTF-8) на URL сервера. В случае, если сервер возвращает ошибку или не предоставляет ответ, то платформа будет совершать повторные запросы в течение 24 часов. Ответ сообщающий о приеме должен быть 200 OK с пустым телом запроса.

Предупреждение: Внимание! Для подключения URL для приема статусов Viber-сообщений обратитесь к вашему менеджеру или напишите письмо в техническую поддержку support@devinotele.com

Пример запроса

```
    [{
"id": 3158611117333282816,
"receivedAt": "1527861323068",
"status": "UNDELIVERED",
"errorCode": "USER_BLOCKED"
    },
    ...
    ]
```

Описание полей запроса со статусами доставки

Параметр	Тип данных	Описание	Допустимые значения	Обязательное поле
id	Long	Уникальный идентификатор сообщения на платформе		Да
receivedDate	LocalDateTime	Дата и время получения статуса		Да
Status	String	Код статуса доставки Viber сообщения.	**VP_EXPIRED** – сообщение просрочено, финальный статус не получен в рамках заданного validity period **DELIVERED** – сообщение доставлено абоненту. **READ** – сообщение просмотрено абонентом. **VISITED** абонент перешел по ссылке в сообщении. **UNDELIVERED** – сообщение отправлено, но не доставлено абоненту. **FAILED** – сообщение не было отправлено в результат сбоя.	Да
errorCode	String	Причина, по которой сообщение не было доставлено абоненту (status=undelivered)	**USER_BLOCKED** – абонент заблокирован **NOT_VIBER_USER** – абонент не является пользователем Viber. **ERROR_VP_EXPIRED** - сообщение просрочено, финальный статус не получен в рамках заданного validity period **BAD_DATA BAD_PARAMETERS ERROR_INSTANT_MESSAGE_TYPE_FORMAT BLOCKED_MESSAGE_TYPE UNKNOWN_ERROR** - неизвестная ошибка **NO_SUITABLE_DEVICE**	Нет

12.5 Прием входящих сообщений

Прием входящих сообщений может использоваться для сбора обратной связи от Абонентов после рекламной/сервисной рассылки с помощью Viber.

Платформа Devino передает HTTP-POST запрос с данными в формате JSON по URL сервера, содержащий пачку новых входящих Viber-сообщений по факту обработки платформой.

Предупреждение: Внимание! Для подключения URL для приема входящих Viber-сообщений обратитесь к вашему менеджеру или напишите письмо в техническую поддержку support@devinotele.com

В случае, если сервер возвращает ошибку или не предоставляет ответ, то платформа будет совершать повторные запросы в течение 1 часа. Ответ сообщающий о приеме должен быть 200 OK с пустым телом запроса.

Пример запроса отправляемого на URL:

```
[
{
```

(continues on next page)

(продолжение с предыдущей страницы)

```
  "id": 2,
  "parentId": 1,
  "receivedAt": "2007-11-29 00:00:00",
  "subject": "test",
  "address": "7916123456789",
  "contentType": "text",
  "content": "balance"
},
...
]
```

Описание полей запроса с входящими сообщениями

Параметр	Тип данных	Описание	Обязательное поле
id	Long	Уникальный идентификатор входящего сообщения на платформе	Да
parentId	Long	Уникальный идентификатор исходящего сообщения на платформе, ответ на которое был отправлен получателем	Да
receivedAt	DateTime	Время получения входящего сообщения поставщиком	Да
subject	String	Адрес отправителя, с которого было отправлено исходящее сообщение	Да
address	String	Номер телефона, с которого отправлено входящее сообщение	Да
contentType	String	Всегда значение «text» - возможен прием только текстовых сообщений	Да
content	String	Текст входящего сообщения	Да

12.6 Таблица кодов возврата

Коды возврата обработки запроса (status)

Код	Описание
ok	Запрос был успешно обработан
error-syntax	ошибка синтаксиса
error-auth	ошибка аутентификации
error-system	системная ошибка
error-account-locked	аккаунт клиента заблокирован
error-instant-message-typeformat	неправильный формат типа исходящего сообщения
error-instant-message-content-type-format	неправильный формат типа содержимого сообщения
error-instant-message-content-image-id-format	неправильный формат идентификатора изображения для содержимого сообщения

Коды возврата обработки сообщения в рамках запроса (code)

Код	Описание
ok	исходящее сообщение успешно принято на отправку
error-system	системная ошибка
not-permitted	запрещено
error-subject-format	неправильный формат подписи
error-subject-unknown	указанная подпись не разрешена клиенту в конфигурации платформы провайдера
error-subject-not-specified	подпись не указана
error-address-format	неправильный формат номера абонента
error-address-unknown	отправка на номерную емкость, к которой относится номер абонента не разрешена клиенту в конфигурации платформы провайдера
error-address-not-specified	номер абонента не указан
error-priority-format	неправильный формат значения приоритета
error-comment-format	неправильный формат значения комментария
error-instant-message-type-format	неправильный формат типа сообщения
error-instant-message-type-not-specified	неправильный формат типа содержимого сообщения
error-content-type-format	неправильный формат содержимого сообщения
error-content-not-specified	содержимое сообщения не указано
error-validity-period-seconds-format	неправильно указано значение времени ожидания доставки
error-instant-message-provider-id-format	неправильный формат провайдерского идентификатора
error-instant-message-provider-id-duplicate	провайдерский идентификатор исходящего сообщения неуникален в рамках запроса проверки статуса
error-instant-message-provider-id-unknown	исходящее сообщение с данным провайдерским идентификатором не найдено на платформе провайдера
error-resend-sms-error	указаны поля для переотправки смс но переотправка не включена
error-resend-sms-validity-period-error	неверное время жизни для смс
error-route-not-found	нет маршрута

Глава 13

SMTP API

13.1 Обзор

Платформа Devino Telecom позволяет клиентам отправлять транзакционные (одиночные) email-сообщения с помощью стандартного протокола SMTP. Данный вид интеграции позволит легко подключить CRM, CMS или другую систему к платформе Devino Telecom для отправки email-сообщений вашим клиентам. При отправке сообщений через протокол SMTP платформа DevinoTelecom будет автоматически фильтровать отписавшиеся и hard bounce адреса.

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

Предупреждение: Внимание! В тексте обязательно должны присутствовать ссылка на отписку и ссылка на веб-версию.

13.2 Подключение

Для того, чтобы начать отправлять транзакционные email-сообщения по протоколу SMTP, вам необходимо:

- Зарегистрироваться в личном кабинете Devino Telecom.
- Получить SMTP-логин и SMTP-пароль у вашего персонального менеджера, либо через службу технической поддержки
- Сообщить IP-адрес, с которого вы будете подключаться к платформе Devino Telecom, вашему персональному менеджеру, либо службе технической поддержки.

- Прописать в качестве SMTP-сервера для вашего приложения адрес integrationapi.net, порт 587 (TLS), либо 465 (SSL)
- Указать SMTP-логин и SMTP-пароль, полученные на шаге 2.

13.3 Отправка: требования и ограничения

При отправке email-сообщений необходимо соблюдать следующие правила:

- Использовать подтвержденный адрес отправителя (Вы можете запросить адрес отправителя в личном кабинете на странице создания email-рассылки)
- Указывать корректный адрес получателя
- Не отправлять письма, размер которых превышает 500 КБ.
- Не загружать в письмо файлы. Если необходимо отправить файл, то вы можете указать на него ссылкой.
- Не указывать в поле «TO» несколько адресов получателя, так как отправка будет сделана только на первый адрес.
- Сообщения отправляются в кодировке UTF-8.
- Указывать ссылку на отписку [Unsubscribe] и ссылку на веб-версию [WebVersion].

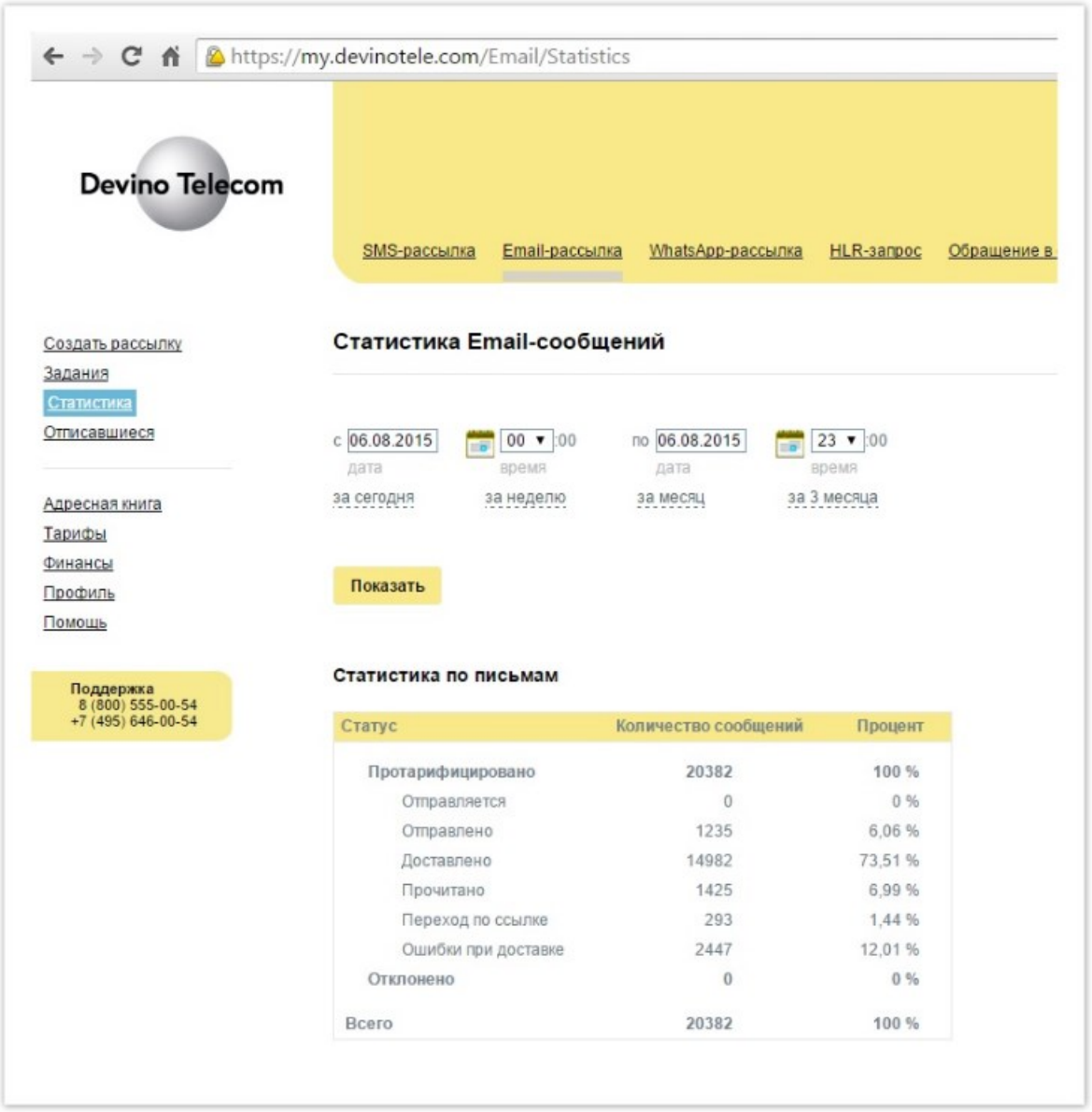
13.4 Дополнительные функции: ссылка на отписку

Вы можете добавить в письмо ссылку на отписку. Для этого необходимо в теле сообщения передавать тег *[Unsubscribe]*

Пример: Если вы хотите отписаться от рассылки нажмите *здесь*

13.5 Получение статистики

Статистика по письмам отправленным через SMTP-протокол собирается аналогично статистике по рассылкам отправленным из личного кабинета. Таким образом вы сможете видеть полноценную статистику по прочитанным письмам, кол-ву переходов по ссылке, кол-ву ошибок при доставке и т.д. Статистику по транзакционным email-сообщениям вы можете получить в личном кабинете на странице «Статистика».



13.6 Обработка ошибок

В случае возникновения ошибки при валидации email-сообщения, платформа Devino Telecom возвращает стандартный код ошибки SMTP 554 Transaction failed и текстовое описание.

Возможные описания ошибок:

Текст ошибки	Причина
Must authenticate before sending mail	Не указан, или указан некорректный логин/пароль
Internal server error	Ошибка сервера
Not allowed attachement type <расширение файла>	Загружен запрещенный файл
Message exceeds fixed size limit	Превышен допустимый размер письма
Invalid recipient address: <адрес получателя>	Некорректный адрес получателя
Disallowed source address: <адрес отправителя>	Неподтвержденный адрес отправителя

13.7 Настройка почтового клиента Outlook

Необходимо создать учетную запись типа «IMAP/SMTP», для этого:

- Откройте Outlook
- Файл –> Добавить учетную запись
- Ручная настройка или дополнительные типы серверов –> Далее
- Протокол POP или IMAP –> Далее

Заполните поля, используя данные учетной записи любого почтового сервиса:

- Имя - любое имя
- Адрес электронной почты
- Тип учетной записи - IMAP
- Сервер входящей почты - например, imap.gmail.com
- Сервер исходящей почты (SMTP): integrationapi.net

Пользователь/пароль - данные для входа в учетную запись –> Другие настройки –> Сервер исходящей почты –> Smtp серверу требуется проверка подлинности –> Вход с помощью –> ввести smtp-логин и smtp-пароль –> Дополнительно –> Smtp-сервер - указать порт 587 –> Сохранить.

13.8 Отправка письма из .NET

```
using System;
using System.Diagnostics;
using System.Net;
using System.Net.Mail;
namespace Devino.Email.SmtpClient
{
    class Program
    {
        static void Main(string[] args)
        {
            using (var smtpClient = new SmtpClient())
            {
                var sourceEmail = "noreplay@devinotele.com";
                var subject = "Test from smtp";
                var messageText = "Привет! <a href=\"http://www.devinotele.com\">Кликни меня</a>";
```

(continues on next page)

(продолжение с предыдущей страницы)

```
                var email = "test@devinotele.com";

                smtpClient.Host = "integrationapi.net";
                smtpClient.Port = 587;
                smtpClient.EnableSsl = true;
                smtpClient.Credentials = new NetworkCredential("1website", "test");

                var message = new MailMessage(sourceEmail, email) { Sender = new␣
↪MailAddress(sourceEmail), Subject = subject, Body = messageText };
                try
                {
                    smtpClient.Send(message);
                }
                catch (Exception ex)
                {
                    Trace.TraceError(ex.Message);
                }
            }
        }
    }
}
```

Глава 14

EMAIL HTTP API

14.1 Обзор

API предоставляет удобный интерфейс для автоматизации процесса отправки email-рассылок через платформу Devino Telecom. С помощью API можно отправлять массовые и транзакционные email-рассылки, управлять рассылками, получать полноценную статистику.

Работа с API осуществляется в соответствии с принципами REST, посредством HTTP-запросов с использованием методов GET, POST, PUT, PATCH и DELETE.

Для использования API необходима авторизация. Авторизация происходит по логину паролю от Личного Кабинета платформы Devino Telecom.

Предупреждение: Внимание! В тексте обязательно должны присутствовать ссылка на отписку и ссылка на веб-версию.

14.1.1 Авторизация

API поддерживает базовую авторизацию через заголовок Authorization (https://en.wikipedia.org/wiki/Basic_access_authentication). В заголовке запроса необходимо передать логин и пароль из Личного Кабинета в формате login:password в base64 кодировке.

```
*Authorization: Basic dGVzdGVyOjExMTExMQ==*
```

14.1.2 Формат запроса

Запрос к API задается в следующем формате:

```
{тип_метода} https://integrationapi.net/email/v{версия}/{ресурс}?{параметры}
```

где:

```
{тип_метода} - HTTP метод GET, POST, PUT, PATCH и DELETE.
{версия} - Версия API. Текущая версия - 2.
{ресурс} - URL ресурса, над которым выполняется действие. Список всех ресурсов смотрите в␣
↪соответствующем разделе.
{параметры} - обязательные и необязательные параметры запроса, которые не входят в состав URL␣
↪ресурса.
Обязательный пункт: Accept */*
```

Сервис позволяет передавать параметры и получать ответы в следующих форматах: JSON и XML.

14.1.3 Формат ответа

Ответ API состоит из двух частей:

- Код с описанием - эта часть присутствует во всех ответах.
- Результат - специфичный для каждого запроса. Может отсутствовать.

```
{
    "Result": {},
    "Code": "not_found",
    "Description": "user not found"
}
```

Код можно использовать для проверки статуса запроса, а описание предназначено для диагностики возможных проблем. Описание может быть изменено в новой версии API без предупреждения о нарушении обратной совместимости. Набор кодов также может быть расширен.

Список кодов ответов:

Код	HTTP status	Расшифровка
ok	200, 201	Запрос выполнен успешно
validation_error	400 - 404	Ресурс не изменён
internal_error	500	Внутренняя ошибка сервиса, можно повторить запрос позже

14.1.4 Запрос диапазонов

Некоторые запросы предполагают возвращение только части данных. Для таких запросов необходимо передавать специальный заголовок:

```
Range: items=1-100
```

Оба предела диапазона включаются. При отсутствии заголовка такие запросы возвращают ошибку validation_error с HTTP кодом 416 RequestedRangeNotSatisfiable.

14.1.5 Локализация

В поле Description может возвращаться локализованная строка с текстом ошибки. Для этого необходимо передать заголовок Accept-Language с нужным языком. В текущей версии поддерживаются русский и английский языки. По умолчанию, если заголовок не передан или язык не найден среди доступных возвращаются ответы на английском.

```
Accept-Language: ru-RU
```

14.2 Управление адресами отправителя

14.2.1 Получение адресов отправителя

GET /UserSettings/SourceAddresses

Метод возвращает адреса отправителя авторизованного пользователя - подтверждённые и запрошенные.

Возвращаемый результат - список записей.

Параметр	Тип данных	Описание
SourceAddress	string	Адрес отправителя
State	SourceAddressState	Статус адреса отправителя 0 - Запрошен (Request) 1 - Подтверждён (Approve) 2 - Отклонён (Reject) 3 - Удалён (Deleted)
IsDefault	bool	Флаг, указывающий является ли адрес адресом по умолчанию

Пример ответа

```
{
    "Result":[
        {
            "SourceAddress": "blabla@gmail.com",
            "State": 1,
            "IsDefault": true
        },
        {
            "SourceAddress": "eeee@mailforspam.com",
            "State": 1,
            "IsDefault": false
        }
    ],
    "Code": "ok",
    "Description": "ok"
}
```

14.2.2 Добавление адреса отправителя

POST /UserSettings/SourceAddresses

Метод отправляет запрос на добавление нового адреса отправителя. Адрес должен быть валидным email адресом. Домен адреса должен быть подтверждён в Личном кабинете.

Если запрос был успешно отправлен, возвращается код «ok» и http код 201. Метод возвращает только стандартный ответ, без поля Result.

Параметры запроса

Параметр	Тип данных	Описание
SenderAddress	string	Адрес отправителя

Пример запроса

```
{
    "SourceAddress":"test@gmail.com"
}
```

Пример ответа

```
{
    "Code": "ok",
    "Description": "ok"
}
```

14.2.3 Удаление адреса отправителя

DELETE UserSettings/SourceAddresses/{SourceAddress}

Параметры запроса

Параметр	Тип данных	Описание
SourceAddress	string	Адрес отправителя

Пример ответа

```
{
    "Code": "ok",
    "Description": "ok"
}
```

14.3 Управление рассылками

14.3.1 Получение списка рассылок

GET /Tasks

Возвращает список рассылок.

Параметры запроса

Параметр	Тип данных	Описание
CreatorLogin	string	Логин создателя рассылки, задаёт фильтр (будут возвращены только те рассылки, что были созданы от имени указанного логина создателя рассылки).
Range	ItemsRange	Диапозон

Метод требует аутентификации с помощью BasicAuthentication Header. Список рассылок возвращается именно для того, кто авторизовался через BasicAuthentication, если только авторизованный не обладает правами админа и параметром Login не задан другой логин.

В случае, если задан CreatorLogin, в ответ попадут только те рассылки, что были созданы сублогином, заданным в CreatorLogin.

Пример ответа


```
{
    "Result": [
        {
            "SourceName": "test",
            "Price": 0.23,
            "SendDuplicates": false,
            "Cancellable": true,
            "Deletable": false,
            "NextStartDateTime": "/Date(1473417269843-0000)/",
            "State": "Waiting",
            "TotalContacts": 10000,
            "CompletedContacts": 10000,
            "ErrorCount": 0,
            "IsExecuting": false,
            "ServiceType": "Email",
            "IsSmooth": false,
            "IsPersonalized": false,
            "ID": 130872,
            "Name": "test",
            "OwnerLogin": "test",
            "Type": "Distribution",
            "Groups": [],
            "IncludedContacts": [],
            "ExcludedContacts": [],
            "ManualContacts": [],
            "StopList": [],
            "Text": "<p>test</p>",
            "Subject": "test",
            "MessageValidity": 0,
            "MessageType": "Email",
            "TaskMessageType": "11",
            "DoTransliterate": false,
            "SourceAddress": "pavel.voropaev@seedway.ru",
            "StartDateTime": "/Date(1395809939517-0000)/",
            "Period": "None",
            "GlobalState": "Paused",
            "GlobalStateInfo":{
                "State": "Paused"
            },
            "PercentageCompleted": 100,
            "MessageValidityAsTimeSpan": "1.00:00:00"
        }
    ],
    "Code": "ok",
    "Description": "ok"
}
```


14.3.2 Получение рассылки

GET /Tasks/{TaskId}

Метод возвращает данные о рассылке.

Параметры запроса:

Параметр	Тип данных	Обязательность	Описание
TaskId	int	Да	Идентификатор рассылки (предаётся в url)

Возвращаемый результат:

Параметр	Тип данных	Описание
TaskId	int	Идентификатор рассылки
Login	string	Логин пользователя
Name	string	Название
Sender	EmailAddress	Отправитель - адрес и имя
Subject	string	Тема
Text	string	Текст
StartDateTime	DateTime	Начало отправки в UTC формате
EndDateTime	DateTime	Окончание отправки в UTC формате (для плавных рассылок)
Type	TaskType	Тип рассылки
UserCampaignId	string	Пользовательский идентификатор рассылки
Contacts	ContactDto[]	Список контактов
ContactGroups	ContactGroupDto[]	Список групп контактов
State	TaskState	Статус рассылки
Price	decimal	Цена за сообщение
CreatorLogin	string	Логин создателя рассылки (сублогин из ролевой модели)
SendDuplicates	bool	Отправлять дубликаты или нет (по умолчанию - нет)
SmsResendTask	SmsResendTaskDto	Данные для переотправки через смс
Counters	EmailTaskCounters	Количество контактов (общее, дубликаты, отписавшиеся, исключённые)

ContactDto

Параметр	Тип данных	Описание
Id	long	Идентификатор контакта
Included	bool	Включать или исключать контакт из рассылки (true или false)

ContactGroupDto

Параметр	Тип данных	Описание
Id	long	Идентификатор группы контакта
Included	bool	Включать или исключать группу из рассылки (true или false)

EmailAddress

Параметр	Тип данных	Описание
Name	string	Имя
Address	string	Адрес

TaskType

Текст	Число	Описание
Distribution	1	Одноразовая рассылка
Birthday	2	Рассылка по дням рождения

SmsResendTaskDto

Параметр	Тип данных	Описание	Обязательный
ContactGroupId	int	Группа контактов для переотправки	Да
Text	strin	Текст SMS	Да
SenderAddress	strin	Адрес отправителя	Да
StartDelay	TimeSpan	Задержка после отправки email рассылки	Да

EmailTaskCounters

Параметр	Тип данных	Описание
TaskId	int	Идентификатор рассылки
TotalContacts	int	Количество получателей
Dublicates	int	Количество отфильтрованных дубликатов
Unsubscribed	int	Количество отфильтрованных отписавшихся
Excluded	int	Количество отфильтрованных исключённых контактов
OverPackage	int	Контакты сверх пакета (на них отправки не будет)
SpamScore	int	Оценка спамности письма

Пример ответа:

```
{
    "Result":{
        "Login": "TEST",
        "Name": "q",
        "Sender":{
            "Address": "xxx@gmail.com",
            "Name": "yyy"
        },
        "Subject": "%Имя%",
        "Text": "blablabla",
        "StartDateTime": "/Date(1440501564737-0000)/",
        "UserCampaignId": "",
        "State": "Finished",
        "Price": 10,
        "Counters":{
            "TotalContacts": 2,
            "Duplicates": 0,
            "Unsubscribed": 0,
            "Excluded": 0,
            "OverPackage": 0,
            "SpamScore": 2.2,
            "TaskId": 10500700
        },
        "Type": "Distribution",
        "Contacts":[
            {
                "Id": 7907323000,
                "Included": true
            },
            {
                "Id": 8603950002,
                "Included": true
            }
        ],
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        "ContactGroups":[],
        "CreatorLogin": "TEST",
        "SendDuplicates": false,
        "TaskId": 10592701
    },
    "Code": "ok",
    "Description": "ok"
}
```

14.3.3 Создание рассылки

POST /Tasks

Метод создаёт рассылку. Если рассылка была успешно создана, возвращается код «ok» и http код 201.

В качестве Result возвращается идентификатор рассылки и набор счётчиков. При их расчёте учитываются только уникальные группы и контакты (из нескольких групп с одинаковыми идентификаторами учитывается только одна).

Максимальное количество получателей в одной рассылке - 2 миллиона.

Порядок вычисления счётчиков:

- дубли
- исключённые группы и контакты
- отписавшиеся

Валидируются:

- текст - на отсутствие стоп-слов и на наличие
- текст - на наличие макросов [Unsubscribe] и [WebVersion]
- тема - на отсутствие стоп-слов
- размер текста и темы (не более 10 МБ)
- отправитель - имя на отсутствие стоп-слов и подтверждён ли адрес
- группы контактов - на существование
- тип рассылки - допустимы только 1 (Distribution) и 2 (Birthday).
- логин - на существование (не актуально для внешнего API)
- шаблон - на существование

Параметры запроса:

Параметр	Тип данных	Описание	Обязательный
Name	string	Название	Да
Sender	EmailAddress	Отправитель - адрес и имя	Да
Subject	string	Тема	Да
Text	string	Текст	Да
StartDateTime	DateTime	Начало отправки в UTC формате	Нет
EndDateTime	DateTime	Окончание отправки в UTC формате (для плавных рассылок)	Нет
Type	TaskType	Тип рассылки	Да
UserCampaignId	string	Пользовательский идентификатор рассылки	Нет
Contacts	ContactDto[]	Список контактов	Нет
ContactGroups	ContactGroupDto[]	Список групп контактов	Нет
TemplateId	string	Идентификатор шаблона	Нет
SmsResendTask	SmsResendTaskDto	Данные для переотправки смс в случае непрочтения письма	Нет
SendDuplicates	bool	Отправлять дубликаты или нет (по умолчанию - нет)	Нет

ContactDto

Параметр	Тип данных	Описание	Обязательный
Id	long	Идентификатор контакта	Да
Included	bool	Включать или исключать контакт из рассылки (true или false)	Да

ContactGroupDto:

Параметр	Тип данных	Описание	Обязательный
Id	long	Идентификатор контакта	Да
included	bool	Включать или исключать группу из рассылки (true или false)	Да

SmsResendTaskDto

Параметр	Тип данных	Описание	Обязательный
Text	strin	Текст SMS	Да
SenderAddress	strin	Адрес отправителя	Да
StartDelay	TimeSpan	Задержка после отправки email рассылки	Да

Возвращаемый результат:

Параметр	Тип данных	Описание
TaskId	int	Идентификатор рассылки
TotalContacts	int	Количество получателей
Dublicates	int	Количество отфильтрованных дубликатов
Unsubscribed	int	Количество отфильтрованных отписавшихся
Excluded	int	Количество отфильтрованных исключённых контактов
OverPackage	int	Контакты сверх пакета (на них отправки не будет)
SpamScore	int	Оценка спамности письма

Пример запроса:

```
{
    "Name":"test",
    "Sender":{
        "Address":"xxx@gmail.com",
        "Name":"yyy"
    },
    "Subject":"test subj",
    "Body": {
    "Html": "test [Unsubscribe][WebVersion]",
    "PlainText": "test"},
    "StartDateTime":"08/31/2015 13:30:38",
    "UserCampaignId":"",
    "ContactGroups":[
        {
            "Id":252,
            "Included":true
        },
        {
            "Id":234,
            "Included":true
        }
    ]
}
```

Пример ответа:

```
{
    "Result":{
        "TaskId": 133875,
        "TotalContacts": 1,
        "Dublicates": 0,
        "Unsubscribed": 0,
        "Excluded": 0
    },
    "Code": "ok",
    "Description": "new task added"
}
```

14.3.4 Редактирование рассылки

PUT /Tasks/{TaskId}

Метод редактирования рассылки. Если рассылка была успешно отредактирована, возвращается код «ok» и http код 200. Параметры запроса и ответ идентичны Tasks POST. Редактировать можно только

рассылки в статусе «New». При этом все поля являются обязательными и заменяются.

14.3.5 Изменение статуса рассылки

PUT /Tasks/{TaskId}/State

Обновление статуса рассылки для остановки, возобновления, отмены, удаления. Возвращается только стандартный ответ.

Параметры запроса:

Параметр	Тип данных	Описание	Обязательный
TaskId	int	Идентификатор рассылки (предаётся в url)	Да
State	TaskState	Текстовый или числовой статус рассылки	Да

TaskState:

Текст	Число	Описание	Можно ли использовать этот статус для PUT
New	0	Статус только что добавленной рассылки	Да
Created	1	Создание рассылки завершено, рассылка готова к выполнению	Да
Started	2	Рассылка отправляется (также используется для возобновления после остановки)	Да
Stopped	3	Рассылка остановлена (с возможностью возобновления)	Да
Canceled	4	Рассылка отменена (без возможности возобновления)	Да
Finished	5	Оправка рассылки завершена успешно	Да
Deleted	6	Рассылка удалена	Да
Failed	7	При отправке рассылки произошла ошибка	Да

Пример запроса:

```
{
    "State":1
}
```

Пример ответа:

```
{
    "Code": "ok",
    "Description": "ok"
}
```

14.4 Шаблоны

14.4.1 Получение шаблона

GET Templates/{TemplateId}

Метод получения шаблона. В качестве результата возвращается шаблон.

Параметры запроса

Параметр	Тип данных	Описание
TemplateId	int	Идентификатор рассылки (предаётся в url)

Возвращаемый результат

Параметр	Тип данных	Описание
TemplateId	int	Идентификатор шаблона
Name	string	Название
Sender	EmailAddress	Отправитель - адрес и имя
Subject	string	Тема
Body	Body	Объект, который содержит HTML и PlainText письма
UserTemplateId	string	Внешний идентификатор
MergeFields	array[str]	Доступные ключи для персонализации

Пример ответа

```
{
    "Result":{
        "MergeFields":[],
        "Name": "test",
        "Sender":{},
        "Body": {
            "Html": "test [Unsubscribe] [WebVersion]",
            "PlainText": "test message"
        },
        "TemplateId": 1
    },
    "Code": "ok",
    "Description": "ok"
}
```

14.4.2 Получение списка шаблонов

GET Templates/

Метод получения списка шаблонов. В качестве результата возвращается шаблон.

Возвращаемый результат

Параметр	Тип данных	Описание
TemplateId	int	Идентификатор шаблона
Name	string	Название
Sender	EmailAddress	Отправитель - адрес и имя
Subject	string	Тема
Body	Body	Объект, который содержит HTML и PlainText письма
UserTemplateId	string	Внешний идентификатор

Пример ответа

```
{
    "Result":[
        {
            "Name": "test",
            "Sender":{},
            "Body": {
                "Html": "test [Unsubscribe] [WebVersion]",
                "PlainText": "test message"
            },
            "TemplateId": 1
        },
        {
            "Name": "test",
            "Sender":{},
            "Body": {
                "Html": "test [Unsubscribe] [WebVersion]",
                "PlainText": "test message"
            },
            "TemplateId": 2
        }
    ],
    "Code": "ok",
    "Description": "ok"
}
```

14.4.3 Создание шаблона

POST /Templates

Метод добавляет шаблон. Если шаблон успешно добавлен, возвращается код «ok» и http код 201. В качестве Result возвращается идентификатор шаблона (int).

Валидируются: * наличие непустого названия * текст - на отсутствие стоп-слов и на наличие макросов [Unsubscribe] и [WebVersion] * тема - на отсутствие стоп-слов * размер текста и темы (не более 10 МБ) * отправитель - имя на отсутствие стоп-слов и подтверждён ли адрес

Параметры запроса:

Параметр	Тип данных	Описание	Обязательный
Name	string	Название шаблона	Да
Sender	EmailAddress	Отправитель - адрес и имя	Нет
Subject	string	Тема	Нет
Body	Body	Объект, содержащий HTML и PlainText шаблона	Да
UserTemplateId	string	Внешний идентификатор	Нет

Пример запроса:

```
{
    "Name":"test",
    "Sender":{
        "Name":"good sender"
    },
    "Body": {
        "Html": "test [Unsubscribe] [WebVersion]",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
        "PlainText": "test message"
    }
}
```

Пример ответа

```
{
    "Result": 123,
    "Code": "ok",
    "Description": "ok"
}
```

14.4.4 Обновление шаблона

PUT Templates/{TemplateId}

Метод обновления шаблона. Если шаблон был успешно обновлён, возвращается код «ok» и http код 200 и обновлённый шаблон.

Параметры запроса

Параметр	Тип данных	Описание	Обязательный
TemplateId	int	Идентификатор шаблона, полученный из метода Templates POST	Да
Name	string	Название шаблона	Да
Sender	EmailAddress	Отправитель - адрес и имя	Нет
Subject	string	Тема	Нет
Body	Body	Объект, содержащий HTML и PlainText шаблона	Да
UserTemplateId	string	Внешний идентификатор	Нет

Возвращаемый результат

Параметр	Тип данных	Описание
TemplateId	int	Идентификатор шаблона
Name	string	Название
Sender	EmailAddress	Отправитель - адрес и имя
Subject	string	Тема
Body	Body	Объект, содержащий HTML и PlainText шаблона
UserTemplateId	string	Внешний идентификатор

Пример запроса

```
{
    "Name":"test",
    "Sender":{
        "Name":"good sender"
    },
    "Body": {
        "Html": "test [Unsubscribe] [WebVersion]",
        "PlainText": "test message"
    }
}
```

Пример ответа


```
{
    "Result":{
        "Name":"test",
        "Sender":{"Name":"good sender"},
        "Body": {
            "Html": "test [Unsubscribe] [WebVersion]",
            "PlainText": "test message"
        },
        "TemplateId": 1
    },
    "Code": "ok",
    "Description": "ok"
}
```


14.4.5 Удаление шаблонов

DELETE Templates/{TemplateId}

Удаление шаблона. Возвращается только стандартный ответ.

Параметры запроса

Параметр	Тип данных	Описание	Обязательный
TemplateId	int	Идентификатор шаблона, полученный из метода Templates POST	Да

Пример ответа


```
{
    "Code": "ok",
    "Description": "ok"
}
```


14.5 Статистика

14.5.1 Получение статистики

GET /Statistics?Login={Login}&TaskId={TaskId}&StartDateTime={StartDateTime}&EndDateTime={E

Получение статистики по сообщениям в виде набора счётчиков (сколько было отправлено, сколько было доставлено, сколько не было отправлено и т.д.).

Параметры запроса

Параметр	Тип данных	Описание	Обязательный
TaskId	int	Идентификатор рассылки, в рамках которой были созданы сообщения, для которых необходимо вернуть статистику.	Да
StartDateTime	DateTime	Дата в формате UTC, задающая начало временного диапазона, которому должны принадлежать сообщения, для которых необходимо вернуть статистику.	Да
EndDateTime	DateTime	Дата в формате UTC, задающая конец временного диапазона, которому должны принадлежать сообщения, для которых необходимо вернуть статистику.	Да

Сервис расчитан на получение в параметрах либо TaskId, - тогда возвращается статистика по сообщениям, отправленным в рамках рассылки с указанным идентификатором TaskId, - либо StartDateTime и EndDateTime, - тогда возвращается статистика по сообщениям, отправленным за временной диапазон, заданный с помощью StartDateTime и EndDateTime (даты должны быть приведены к UTC зоне).

Пример ответа


```
{
    "Result": {
        "NotSent": 30,
        "Sent": 0,
        "Delivered": 0,
        "Read": 0,
        "Clicked": 0,
        "Bounced": 0,
        "Rejected": 0,
        "Total": 30
    },
    "Code": "ok",
    "Description": "ok"
}
```


14.5.2 Получение детализации

GET /Statistics/Messages?Login={Login}&TaskId={TaskId}&StartDateTime={StartDateTime}&EndDate

Получение детализации по сообщениям.

Параметр	Тип данных	Описание	Обязательный
TaskId	int	Идентификатор рассылки, в рамках которой были созданы сообщения, для которых необходимо вернуть статистику.	Да
StartDateTime	DateTime	Дата в формате UTC, задающая начало временного диапазона, которому должны принадлежать сообщения, для которых необходимо вернуть статистику.	Да
EndDateTime	DateTime	Дата в формате UTC, задающая конец временного диапазона, которому должны принадлежать сообщения, для которых необходимо вернуть статистику.	Да
State	string	Выполняет роль фильтра, требует вернуть статистику по тем сообщениям, что находятся в указанном состоянии.	Нет
Range	ItemsRange	Диапозон	Да

Сервис расчитан на получение параметров либо TaskId, - тогда возвращается статистика по сообщениям, отправленным в рамках рассылки с указанным идентификатором TaskId, - либо StartDateTime и EndDateTime, - тогда возвращается статистика по сообщениям, отправленным за временной диапазон, заданный с помощью StartDateTime и EndDateTime (даты должны быть приведены к UTC зоне), так же в заголовках необходимо передавать диапазон в формате Range: items=1-100.

Параметр State является опциональным и может применяться в обоих из ранее описанных сценариев, - тогда возвращается ранее описанная статистика по сообщениями, находящимя в указанном состоянии.

Пример ответа:

```
{
    "Result": [
        {
            "State": "NotSent",
            "Price": 0,
            "Id": 141471292110003601,
            "DestinationEmail": "user@devinotele.com",
            "LastUpdateUtc": "/Date(1485937304700-0000)/",
            "CreatedDateUtc": "/Date(1485937304000-0000)/"
        }
    ],
    "Code": "ok",
    "Description": "ok"
}
```

14.6 Отправка транзакционного сообщения

POST v2/messages

Метод отправляет транзакционное сообщение нескольким получателям с возможностью использования макросов. Если сообщение успешно добавлено в очередь, возвращается код «ok» и http код 201. В качестве Result возвращается идентификатор сообщения (string).

Валидируются:

- текст - на отсутствие стоп-слов (нецензурная лексика)
- тема - на отсутствие стоп-слов

- размер текста и темы (не более 10 МБ)
- отправитель - имя на отсутствие стоп-слов и подтверждён ли адрес
- получатель - имя на отсутствие стоп-слов и валидность e-mail адреса, также проверяется по списку отписавшихся
- шаблон - на существование

Параметры запроса:

Параметр	Тип данных	Описание	Обязательный
Sender	Массив String	Отправитель - адрес и имя	Да
Recipients	Список	Список получателей (см. табл. 2)	Да
Subject	String	Тема письма	Да
Body	Массив String	Тело сообщения HTML и PlainText	Да
TemplateId	String	Идентификатор шаблона	Нет
UserCampaignId	String	Идентификатор рассылки в системе пользователя	Нет

Recipient:

Параметр	Тип данных	Описание	Обязательный
MergeFields	Массив String	Пользовательские макросы вида ключ – значение. В названии макроса запрещены спец. символы	Нет
RecipientId	String	Пользовательский идентификатор получателя, не более 32 символов	Нет
Address	String	Адрес получателя	Да
Name	String	Имя получателя	Нет

Пример запроса:

```
{
    "Sender":{
        "Address": "sourceaddress@example.com",
        "Name": "Test"
    },
    "Recipients": [
        {
            "MergeFields": {
            "ExtField":"5 дней",
            "Name": "Иван"
        },
            "RecipientId": "",
            "Address": "ivan@example.com",
            "Name": "Ivan"
        }
    ],
    "Subject": "Ув. [Name]!",
    "Body": {
        "Html": "Ув. [Name]! Осталось [ExtField]<br><a href=\"[Unsubscribe]\">Отписаться</a>",
        "PlainText": "Ув. {ExtField}! Ждем вас завтра! [Unsubscribe]"
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    },
    "UserCampaignId": "1234"
}
```

Пример ответа:

```
{
    "Result": [
        {
            "Index":0,
            "Address":"ivan@example.com",
            "MessageId":"Mdz0i7z1Dyp",
            "Code":"ok"
        }
    ],
    "Code":"ok",
    "Description":"ok"
}
```

Сценарии:

- Перед началом отправки необходимо подтвердить адрес отправителя
- В текст письма может быть включен макрос [Unsubscribe] - на его место будет подставлена ссылка на страницу отписки.

14.7 Отправка транзакционного сообщения (old)

POST /Messages

Предупреждение: С выходом Email API v2 данный метод не поддерживается

Метод отправляет транзакционное сообщение. Если сообщение успешно добавлено в очередь, возвращается код «ok» и http код 201. В качестве Result возвращается идентификатор сообщения (string).

Валидируются:

- текст - на отсутствие стоп-слов (нецензурная лексика)
- тема - на отсутствие стоп-слов
- размер текста и темы (не более 10 МБ)
- отправитель - имя на отсутствие стоп-слов и подтверждён ли адрес
- получатель - имя на отсутствие стоп-слов и валидность e-mail адреса, также проверяется по списку отписавшихся
- шаблон - на существование

Параметры запроса:

Параметр	Тип данных	Описание	Обязательный
Sender	EmailAddress	Отправитель - адрес и имя	Да
Recipient	EmailAddress	Получатель - адрес и имя	Да
Subject	string	Тема	Да
Text	string	Текст	Да
UserMessageId	string	Идентификатор сообщения в системе пользователя	Нет
UserCampaignId	string	Идентификатор рассылки в системе пользователя	Нет
TemplateId	string	Идентификатор шаблона (внешний или внутренний)	Нет

Пример запроса:

```
{
    "Sender":{
        "Address":"test@test.com",
        "Name":"name"
    },
    "Recipient":{
        "Address":"test@supertest.com",
        "Name":"name"
    },
    "Subject":"test subj",
    "Text":"test"
}
```

Пример ответа:

```
{
    "Result": "kaAtrHbZ72",
    "Code": "ok",
    "Description": "message queued to send"
}
```

Сценарии:

- Перед началом отправки необходимо подтвердить адрес отправителя(«Sender»: {«Address»})
- В текст письма может быть включен макрос [Unsubscribe] - на его место будет подставлена ссылка на страницу отписки.

14.8 Получение статусов транзакционных сообщений

GET /Messages/{MessageId},{MessageId}

Метод используется для получения статусов транзакционных сообщений. Допускается передача сразу нескольких идентификаторов сообщений через запятую. Можно передавать не более 150 идентификаторов. При этом возвращаются статусы только уникальных сообщений и только сообщений доступных пользователю.

Параметры запроса

Параметр	Тип данных	Описание	Обязательный
MessageId	string	Идентификатор сообщения (предаётся в url, можно указать несколько через запятую)	Да

Возвращаемый результат (массив для нескольких сообщений)

Параметр	Тип данных	Описание
MessageId	string	Идентификатор сообщения
Email	string	Email получателя
State	string	Статус сообщения
RecipientId	string	Пользовательский идентификатор получателя

State

Значение	Описание
NotSent	Отправляется
Sent	Отправлено
Delivered	Доставлено
Read	Прочитано
Clicked	Переход по ссылке
Bounced	Не удалось доставить
Rejected	Отклонено (сообщение не было отправлено)

Пример ответа

```
{
    "Result":[
        {
            "MessageId": "y49EiXaPY1",
            "Email": "ftw@gmail.com",
            "State": "Sent"
        },
        {
            "MessageId": "y49cjxHxxI",
            "Email": "blabla@gmail.com",
            "State": "NotSent"
        }
    ],
    "Code": "ok",
    "Description": "ok"
}
```

14.9 Получение callback

Данный метод позволяет не обращаться к API Devino каждый раз, когда требуется получить статус доставки сообщения, а обрабатывать входящие события от платформы Devino на своем внутреннем ресурсе. Формат Callback: json или XML.

Предупреждение: Внимание! Для подключения URL для приема статусов Email-сообщений обратитесь к вашему менеджеру или напишите письмо в техническую поддержку support@devinotele.com

Запросы производятся по следующим событиям:

- Отправлено (Sent)
- Доставлено (Delivered)
- Прочитано (Opened)
- Переход по ссылке (Clicked)
- Не удалось доставить (Bounced)
- Отписался от рассылки (Unsubscribed)
- Подписался на рассылки (Subscribed)
- Жалоба (Complained)

14.9.1 Параметры запроса:

Параметр	Тип данных	Описание	Доступен в событиях
messageId	string	Идентификатор сообщения	Во всех
taskId	int	Идентификатор рассылки	Во всех
userMessageId	string	Клиентский идентификатор сообщения	Во всех, кроме delivered bounced, complained
userCampaignId	string	Клиентский идентификатор кампании	Во всех, кроме delivered bounced, complained
email	string	Email получателя	Во всех
event	Event	Событие	Во всех
url	string	Url, по которому перешел получатель	clicked
dateTime	string	Дата и время события	Во всех
clientInfo	ClientInfo	Информация о получателе	opened, clicked, unsubscribed
isHardbounce	bool	Указывает на то, является ли передаваемый статус hard bounce	По умолчанию NULL. Заполняется только в случае если поле event равен Bounced
reason	string	Содержит причину, по которой сообщение не было принято почтовым сервером.	По умолчанию NULL. Заполняется только в случае если поле event равен Bounced

В случае, если параметр недоступен в событии, то значение параметра будет null

14.9.2 Event

Значение	Описание
SENT	Отправлено
DELIVERED	Доставлено
OPENED	Прочитано
CLICKED	Переход по ссылке
BOUNCED	Не удалось доставить
UNSUBSCRIBED	Получатель отписался
SUBSCRIBED	Получатель подписался
COMPLAINED	Получатель пожаловался

14.9.3 ClientInfo

Значение	Описание
platform	Тип платформы. Например, DESKTOP
operatingSystem	Операционная система
browser	Браузер
userAgent	userAgent
ipAddress	IP адрес
geolocation	Геолокация

14.9.4 Geolocation

Значение	Описание
country	Страна
region	Регион
city	Город

В данный момент геолокация не определяется, поэтому в ближайшее время параметры country, region и city будут пустыми.

14.9.5 Reason

Значение	Описание
already_unsubscribed	Получатель отписан от рассылок
bad-configuration	Некорректная конфигурация
bad-connection	Ошибка в соединении
bad-domain	Домен не принимает почту или не существует
bad-mailbox	Адрес не существует, доставка не удалась
content-related	Заблокировано из-за содержания
inactive-mailbox	Адрес когда-то существовал, но сейчас отключен
invalid-sender	Неправльный адрес отправителя
message-expired	Истек срок давности доставки
no-answer-from-host	Сервер получателя не отвечает
other	Неизвестная ошибка
overpackage-related	Превышение пакета
policy-related	Не соответствует настройкам правил сервера получателя
protocol-errors	Отклонено из-за ошибок SMTP
quota-issues	Адрес существует, но почта не принимается, т.к. исчерпана дисковая квота
relaying-issues	Ошибка ретрансляции
routing-errors	Ошибка маршрутизации
spam-related	Сообщение отвергнуто сервером получателя как спам
virus-related	Отклонено как инфицированное

Пример запроса Sent:


```
{
    "messageEventDtos":
    [
        {
            "messageId": "MsA29aT4zJe",
            "taskId": 0,
            "userMessageId": "userMessageId-1234",
            "userCampaignId": "UserCampaignId-1234",
            "email": "address@example.com",
            "event": "SENT",
            "url": null,
            "dateTime": "2020-01-14T08:27:57.7377849",
            "clientInfo":
            {
                "platform": null,
                "operatingSystem": null,
                "browser": null,
                "userAgent": null,
                "ipAddress": null,
                "geolocation":
                {
                    "country": null,
                    "region": null,
                    "city": null
                }
            },
            "isHardbounce": null,
            "reason": null
        }
```


(continues on next page)

(продолжение с предыдущей страницы)

```
    ]
}
```

Пример запроса Delivered:

```
{
    "messageEventDtos":
    [
        {
            "messageId": "MsA29aT4zJe",
            "taskId": 0,
            "userMessageId": "userMessageId-1234",
            "userCampaignId": "UserCampaignId-1234",
            "email": "address@example.com",
            "event": "DELIVERED",
            "url": null,
            "dateTime": "2020-01-14T08:27:57.7377849",
            "clientInfo":
            {
                "platform": null,
                "operatingSystem": null,
                "browser": null,
                "userAgent": null,
                "ipAddress": null,
                "geolocation":
                {
                    "country": null,
                    "region": null,
                    "city": null
                }
            },
            "isHardbounce": null,
            "reason": null
        }
    ]
}
```

Пример запроса Opened:

```
{
    "messageEventDtos":
    [
        {
            "messageId": "MsA29aT4zJe",
            "taskId": 0,
            "userMessageId": "userMessageId-1234",
            "userCampaignId": "UserCampaignId-1234",
            "email": "address@example.com",
            "event": "OPENED",
            "url": null,
            "dateTime": "2020-01-14T08:27:57.7377849",
            "clientInfo":
            {
                "platform": "DESKTOP",
                "operatingSystem": "Windows",
                "browser": "Outlook",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
                "userAgent": "Mozilla/4.0(compatible;MSIE7.0;WindowsNT10.0;Win64;x64;Trident/7.0;.
↪NET4.0C;.NET4.0E;.NETCLR2.0.50727;.NETCLR3.0.30729;.NETCLR3.5.30729;ASU2JS;MicrosoftOutlook16.0.
↪9029;ms-office;MSOffice16)",
                "ipAddress": "192.168.0.1",
                "geolocation":
                {
                    "country": null,
                    "region": null,
                    "city": null
                }
            },
            "isHardbounce": null,
            "reason": null
        }
    ]
}
```

Пример запроса Clicked:

```
{
    "messageEventDtos":
    [
        {
            "messageId": "MsA29aT4zJe",
            "taskId": 0,
            "userMessageId": "userMessageId-1234",
            "userCampaignId": "UserCampaignId-1234",
            "email": "address@example.com",
            "event": "CLICKED",
            "url": "http://example.com",
            "dateTime": "2020-01-14T08:27:57.7377849",
            "clientInfo":
            {
                "platform": "DESKTOP",
                "operatingSystem": "Windows",
                "browser": "Chrome",
                "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,␣
↪like Gecko) Chrome/65.0.3325.181 Safari/537.36",
                "ipAddress": "192.168.0.1",
                "geolocation":
                {
                    "country": null,
                    "region": null,
                    "city": null
                }
            },
            "isHardbounce": null,
            "reason": null
        }
    ]
}
```

Пример запроса Unsubscribed:

```
{
    "messageEventDtos":
```

(continues on next page)

(продолжение с предыдущей страницы)


```
    [
        {
            "messageId": "MsA29aT4zJe",
            "taskId": 0,
            "userMessageId": "userMessageId-1234",
            "userCampaignId": "UserCampaignId-1234",
            "email": "address@example.com",
            "event": "UNSUBSCRIBED",
            "url": null,
            "dateTime": "2020-01-14T08:27:57.7377849",
            "clientInfo":
            {
                "platform": "DESKTOP",
                "operatingSystem": "Windows",
                "browser": "Chrome",
                "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,␣
↪like Gecko) Chrome/65.0.3325.181 Safari/537.36",
                "ipAddress": "192.168.0.1",
                "geolocation":
                {
                    "country": null,
                    "region": null,
                    "city": null
                }
            },
            "isHardbounce": null,
            "reason": null
        }
    ]
}
```


Пример запроса Subscribed:


```
{
    "messageEventDtos":
    [
        {
            "messageId": "MsA29aT4zJe",
            "taskId": 0,
            "userMessageId": "userMessageId-1234",
            "userCampaignId": "UserCampaignId-1234",
            "email": "address@example.com",
            "event": "SUBSCRIBED",
            "url": null,
            "dateTime": "2020-01-14T08:27:57.7377849",
            "clientInfo":
            {
                "platform": "DESKTOP",
                "operatingSystem": "Windows",
                "browser": "Chrome",
                "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,␣
↪like Gecko) Chrome/65.0.3325.181 Safari/537.36",
                "ipAddress": "192.168.0.1",
                "geolocation":
                {
                    "country": null,
```


(continues on next page)

(продолжение с предыдущей страницы)

```
                    "region": null,
                    "city": null
                }
            },
            "isHardbounce": null,
            "reason": null
        }
    ]
}
```

Пример запроса Bounced:

```
{
    "messageEventDtos":
    [
        {
            "messageId": "MsA29aT4zJe",
            "taskId": 0,
            "userMessageId": "userMessageId-1234",
            "userCampaignId": "UserCampaignId-1234",
            "email": "address@example.com",
            "event": "BOUNCED",
            "url": null,
            "dateTime": "2020-01-14T08:27:57.7377849",
            "clientInfo":
            {
                "platform": null,
                "operatingSystem": null,
                "browser": null,
                "userAgent": null,
                "ipAddress": null,
                "geolocation":
                {
                    "country": null,
                    "region": null,
                    "city": null
                }
            },
            "isHardbounce": true,
            "reason": "bad-mailbox"
        }
    ]
}
```

Пример запроса Complained:

```
{
    "messageEventDtos":
    [
        {
            "messageId": "MsA29aT4zJe",
            "taskId": 0,
            "userMessageId": "userMessageId-1234",
            "userCampaignId": "UserCampaignId-1234",
            "email": "address@example.com",
            "event": "COMPLAINED",
```

(continues on next page)

(продолжение с предыдущей страницы)


```
            "url": null,
            "dateTime": "2020-01-14T08:27:57.7377849",
            "clientInfo":
            {
                "platform": "DESKTOP",
                "operatingSystem": "Windows",
                "browser": "Chrome",
                "userAgent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,␣
↪like Gecko) Chrome/65.0.3325.181 Safari/537.36",
                "ipAddress": "192.168.0.1",
                "geolocation":
                {
                    "country": null,
                    "region": null,
                    "city": null
                }
            },
            "isHardbounce": null,
            "reason": null
        }
    ]
}
```



```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<messageEventDtos>
    <messageEventDto>
        <messageId>MsAqJyjEjwP</messageId>
        <taskId>0</taskId>
        <userMessageId>UserMessageId-1234</userMessageId>
        <userCampaignId>UserCampaignId-1234</userCampaignId>
        <email>address@example.com</email>
        <event>SENT</event>
        <dateTime>2020-01-14T08:34:02.9306227</dateTime>
        <clientInfo>
            <geolocation/>
        </clientInfo>
    </messageEventDto>
</messageEventDtos>
```



```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<messageEventDtos>
    <messageEventDto>
        <messageId>MsA247Nalun</messageId>
        <taskId>0</taskId>
        <userMessageId>userMessageId-1234</userMessageId>
        <userCampaignId>UserCampaignId-1234</userCampaignId>
        <email>address@example.com</email>
        <event>BOUNCED</event>
        <dateTime>2020-01-14T08:26:27.0083642</dateTime>
        <clientInfo>
            <geolocation/>
        </clientInfo>
        <isHardbounce>true</isHardbounce>
        <reason>bad-mailbox</reason>
    </messageEventDto>
</messageEventDtos>
```

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<messageEventDtos>
    <messageEventDto>
        <messageId>MsAqREbsSUB</messageId>
        <taskId>0</taskId>
        <userMessageId>UserMessageId-1234</userMessageId>
        <userCampaignId>UserCampaignId-1234</userCampaignId>
        <email>address@example.com</email>
        <event>BOUNCED</event>
        <dateTime>2020-01-14T08:33:12.1570284</dateTime>
        <clientInfo>
            <geolocation/>
        </clientInfo>
        <isHardbounce>false</isHardbounce>
        <reason>already_unsubscribed</reason>
    </messageEventDto>
</messageEventDtos>
```

Глава 15

VK API

15.1 Общие сведения

Сервис предоставляет REST API для отправки VK-сообщений через платформу. Devino Telecom с возможностью переотправки по Viber и SMS в случае недоставки, а также для получения отчетов о статусах доставки сообщений.

Работа с API осуществляется с использованием протокола HTTP.

Тела запроса клиента и ответа сервера представлены в формате JSON. Данные должны быть в кодировке UTF-8.

API поддерживает базовую авторизацию через заголовок Authorization (https://en.wikipedia.org/wiki/Basic_access_authentication).

В заголовке запроса необходимо передать логин и пароль из Личного Кабинета в формате login:password в base64 кодировке, например:

```
Authorization: Basic dGVzdGVyOjExMTExMQ==
```

15.1.1 Подключение услуги

Предупреждение: Для подключения услуги необходимо предоставить следующую информацию Вашему персональному менеджеру:

- Название и адрес группы ВКонтакте
- Описание Вашей компании и вида ее деятельности
- Ссылку на Ваш сайт или мобильное приложение.
- Шаблоны которые будут использоваться для отправки сообщений.

15.1.2 Используемые типы данных

Тип данных	Описание
Integer	Положительные числа больше нуля
DateTime	Значение, обозначающее дату и время, представляются в пакетах в формате «yyyy-MMdd hh:mm:ss», например, 1999-05-31 13:20:00
Address	Адрес абонента. Номер мобильного телефона абонента в международном формате (в формате E.164). Например, 79031112233, для России или 491791112233 для Германии
String	Строка символов в формате UTF-8
Object	Объект (включает в себя несколько полей других типов)

Ограничения и допустимые значения описаны далее в разделе описания пакетов протокола.

Предупреждение: На стороне VK имеются ограничения на отпарвку:

- не более 50 уведомлений в секунду для одного service (для одной группы в VK)
- не более 5 уведомлений в сутки для одного пользователя от одного service (одной группы в VK)

15.2 Методы работы с API

15.2.1 Отправка сообщения

(/send/vk)

Платформа Devino инициирует отправку VK-сообщения абоненту в соответствии с данными,переданными в теле POST-запроса:

```
https://im.devinotele.com/send/vk
```

В одном запросе может быть информация об отправке только одного сообщения.

Пример тела запроса отправки сообщения

```
{
 "vk":{
    "subject":"TEST",
    "priority":"high",
    "routes":[
       "vk"
    ],
    "validityPeriod":180,
    "phone":"79999999999",
    "templateId":"123456",
    "templateData":{
       "param1":"value1",
       "param2":"value2"
    }
 },
```

(continues on next page)

(продолжение с предыдущей страницы)

```
  "viber":{
     "subject":"TEST",
     "priority":"high",
     "validityPeriodSec":30,
     "type":"viber",
     "comment":"comment",
     "contentType":"button",
     "text":"text",
     "caption":"caption",
     "action":"http://company.com/resource",
     "imageUrl":"http://company.com/image.jpg",
     "dstAddress":"79999999999"
  },
  "sms":{
     "srcAddress":"TESTSMS",
     "text":"тест сообщения",
     "validityPeriod":60,
     "dstAddress":"79999999999"
  }
}
```

15.2.2 Описание полей тела запроса отправки сообщения

Поле	Тип данных	Допустимые занчения	Описание	Обязательное поле
Описание полей объекта VK				
subject	String	Строка от 1 до 11 символов	Адрес отправителя	Да
priority	String	Варианты: 1) «low» 2) «medium» 3) «high» 4) «realtime»	Приоритет сообщения	Да
routes	массив String	Варианты: 1) «vk» 2) «ok» 3) «ok»,»vk» 4) «vk»,»ok»	Массив маршрутов VK в порядке использования, пример «routes»:[«ok»,»vk»]	Да
validityPeriod	Long	Целое число от 15 до 86400	Время жизни сообщения в секундах	Да
deliveryPolicy	String	Варианты: 1) «any» 2) «mobile_device_required»	По умолчанию any. Если указано mobile_device_required, то доставка производится только в случае наличия у пользователя мобильного приложения и его использования в течение последних 7 дней. Доставка при этом производится во все имеющиеся устройства, а не только мобильные. в секундах.	Да
phone	String	Номер телефона в соответствии со стандартом E.164, возможен + в начале	Номер телефона получателя сообщения	Да
templateId	Long	Целое число	Идентификатор шаблона	Да
templateData	Object		Значения параметров шаблона - например если шаблон «Уважаемый #abonent# с #startTime# по #endTime# сервис будет недоступен», то пример templateData может быть такой: «templateData»: { «abonent»: «Иванов А.Б.», «startTime»: «10.01.2017 15.15», «endTime»: «10.01.2017 15.30» } Шаблон должен быть согласован VK	Да
Описание полей объекта Viber				
subject	String		Имя отправителя Viber-сообщения	Да
priority	String	Варианты: 1) «low» 2) «medium» 3) «high» 4) «realtime»	Приоритет сообщения	Да
validityPeriod	Long	Целое число от 15 до 86400	Время жизни сообщения в секундах	Да
comment	String	Произвольный текстовый комментарий.		Нет
type	String	Тип отправляемого сообщения. Определяет канал, которые используется для доставки сообщения на мобильный телефон абонента	viber	Да
contentType	String	Тип содержимого сообщения.	text – текстовое сообщение image – изображение button – гиперссылка в виде кнопки	Да
dstAddress	String	Номер телефона в соответствии со стандартом E.164, возможен + в начале	Номер телефона получателя сообщения	Да

Пример ответа на запрос отправки сообщения

```
{
  "code": "ok",
  "description": "",
  "result": {
    "code": "ok",
    "messageId": 3222269333010907000
  }
}
```

15.2.3 Описание полей тела ответа на запрос отправки сообщения

Поле	Тип данных	Допустимые занчения	Описание	Обязательное поле
code	String	Возможные значения перечислены в таблице кодов ответа на запрос отправки сообщения	Код ответа на запрос отправки сообщения	Да
description	String	Возможные значения перечислены в таблице кодов ответа на запрос отправки сообщения	Описание ошибки обработки запроса отправки сообщения (если была)	Да
result	Object		Информация о коде валидации и ID сообщения	Да, если code=»ok»
Описание полей объекта result				
code	String	Возможные значения перечислены в таблице кодов валидации сообщения	Код валидации сообщения	Да
messageId	Long		Уникальный идентификатор сообщения	Да, если code=»ok»

15.2.4 Коды ответа на запрос отправки сообщения

code	description
ok	
validation_error	login_not_specified
validation_error	messages_not_specified
validation_error	invalid_json
queue_full	login_send_queue_overflow
system_error	Описание внутренней ошибки сервера

15.2.5 Коды валидации сообщения

code	Описание
ok	Сообщение добавлено в очередь на отправку
subject_not_specified	Не указан адрес отправителя
subject_invalid	Недопустимый адрес отправителя
priority_not_specified	Не указан приоритет сообщения
priority_invalid	Недопустимый приоритет сообщения
routes_not_specified	Не указаны маршруты доставки
routes_invalid	Недопустимый набор маршрутов доставки
vp_invalid	Недопустимый validityPeriod
phone_not_specified	Не указан номер телефона
phone_invalid	Недопустимый номер телефона
text_not_specified	Не указан текст сообщения
text_invalid	Недопустимый текст сообщения
sms_text_not_specified	Не указан текст SMS-сообщения
sms_subject_not_specified	Не указан номер отправителя SMS-сообщения
sms_validity_period_not_specified	Не указано время жизни SMS-сообщения
invalid_sms_validity_period	Недопустимое время жизни SMS-сообщения
not-enough-credits	Недостаточно средств на лицевом счете

15.3 Получение статуса сообщения

(/status/vk)

Платформа Devino возвращает статус доставки ранее отправленного VK-сообщения, messageId которого был ранее передан в теле GET-запроса:

```
https:/im.devinotele.com/status/vk?message=<ID Вашего сообщения>
```

Описание параметров запроса статусов

Поле	Тип данных	Допустимые занчения	Описание	Обязательное поле
message	Long		Идентификатор сообщения	Да

Пример ответа на запрос статусов

```
{
  "code": "ok",
  "description": "",
  "result": {
    "providerId": 3287014702114144256,
    "code": "ok",
    "status": "failed",
    "statusAt": "2018-07-03 16:31:40",
    "smsStates": [
      {
        "id": 711869146186383364,
        "status": "delivered"
      }
    ],
```

(continues on next page)

(продолжение с предыдущей страницы)


```
    "viberStatus": {
      "id": 3287014702114144256,
      "status": "undelivered",
      "statusAt": "2018-07-03 16:31:41",
      "code": "not-viber-user"
    }
  }
}
```

15.3.1 Описание полей тела ответа на запрос статусов

Поле	Тип данных	Допустимые занчения	Описание	Обязое поле
code	String	Возможные значения перечислены в таблице кодов ответа на запрос статусов	Код ответа на запрос отправки сообщения	Да
description	String	Возможные значения перечислены в таблице кодов ответа на запрос статусов	Описание ошибки обработки запроса запроса статусов (если была)	Да
result	Object		Каждому объекту из массива messages запроса соответствует объект в массиве result ответа	Да, если code=»ok
Описание полей объекта result				
providerId	Long		Идентификатор сообщения	Да
code	String	Возможные значения перечислены в таблице кодов валидациисообщения идентификаторов сообщений	Код валидации идентификатора	Да
status	String	**enqueued** – сообщение добавлено в очередь на отправки, **sent** – сообщение отправлено, **delivered** – сообщение доставлено, **undelivered** – сообщение отправлено, но не доставлено, **failed** – сообщение не доставлено в результате сбоя, **vp_expired** – сообщение не доставлено в течение validityPeriod, **read** – сообщение просмотрено абонентом.	Статус доставки сообщения VK	Да
statusDate	DateTime	Возможные значения перечислены в таблице	Время обновления статуса доставки сообщения VK	Да
error	String	Набор всех возможных ошибок заранее не предопределен	Информация о статусе сообщения	Нет
viberStatus	Object		Информация о статусе сообщения	Да, если code=»ok»
smsStatus	Object		Статусы доставки SMS-сообщения	Нет
Описание полей объекта viberStatus				

15.3.2 Коды ответа на запрос статусов

code	description
ok	
validation_error	message_not_specified
system_error	Описание внутренней ошибки сервера

15.3.3 Коды валидации идентификаторов сообщений

code	description
ok	Известный идентификатор сообщения
unknown_message_id	Неизвестный идентификатор сообщения

Коды возврата обработки viber сообщения в рамках запроса (code)

Код	Описание
ok	исходящее сообщение успешно принято на отправку
error-system	системная ошибка
error-instant-message-client-id-not-unique	клиентский идентификатор сообщения не уникален в рамках всего взаимодействия между клиентом и провайдером.
error-subject-format	неправильный формат подписи
error-subject-unknown	указанная подпись не разрешена клиенту в конфигурации платформы провайдера
error-subject-not-specified	подпись не указана
error-address-format	неправильный формат номера абонента
error-address-unknown	отправка на номерную емкость, к которой относится номер абонента не разрешена клиенту в конфигурации платформы провайдера
error-address-not-specified	номер абонента не указан
error-priority-format	неправильный формат значения приоритета
error-comment-format	неправильный формат значения комментария
error-instant-message-type-format	неправильный формат типа сообщения
error-instant-message-type-not-specified	неправильный формат типа содержимого сообщения
error-content-type-format	неправильный формат содержимого сообщения
error-content-not-specified	содержимое сообщения не указано
error-validity-period-seconds-format	неправильно указано значение времени ожидания доставки
error-instant-message-provider-id-format	неправильный формат провайдерского идентификатора
error-instant-message-provider-id-duplicate	провайдерский идентификатор исходящего сообщения неуникален в рамках запроса проверки статуса
error-instant-message-provider-id-unknown	исходящее сообщение с данным провайдерским идентификатором не найдено на платформе провайдера
error-resend-sms-error	указаны поля для переотправки смс но переотправка не включена
error-resend-sms-validity-period-error	неверное время жизни для смс

15.4 Получение статуса сообщения с помощью Callback-запросов

Для получения статуса сообщения могут использоваться callback-запросы. В таком случае Платформа Devino будет отправлять POST-запрос на выбранный Вами URL каждый раз, когда у отправленного Вами сообщения будет меняться статус. Запрос считается доставленным, если в ответ на него был получен статус HTTP(200). В противном случае будут совершаться повторные попытки доставки в течение 24 часов и по истечению этого срока статус сообщения можно будет получить только с помощью GET-запроса, описанного выше.

Предупреждение: Обратите внимание, что информация о переотправке по SMS в callback-запросе не предоставляется.

Предупреждение: Для получения callback-запросов от сервиса необходимо передать Вашему персональному менеджеру или в техническую поддержку (support@devinotele.com) информацию об URL, на который будут отправляться запросы.

Пример тела callback-запроса

```
[
  {
    "messageId":1343343,
    "status": "DELIVERED",
    "receivedAt": "2017-05-31 14:51:12",
    "error":"Доставлено"
  }
]
```

15.4.1 Описание полей запроса

Поле	Тип данных	Описание	Обязательное поле
id	Long	Уникальный идентификатор сообщения в Платформе Devino	Да
status	String	Статус доставки сообщения VK	Да
time	DateTime	Время получения статуса (по Москве, UTC+3)	Да
error	String	Ошибка доставки сообщения VK (если есть)	Да

Глава 16

HLR HTTP API

16.1 Отправка HLR-собщения на несколько номеров (POST)

Сервис инициирует отправку HLR-сообщения в соответствии со значениями параметров, передаваемых сервису в POST-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Hlr/SendBulk?Login=<Логин>&Password=<Пароль>&
↪DestinationAddresses=<Номерполучателя>&Validity=<Время жизни сообщения>
```

или вместо пары логинпароль можно передавать SessionID

```
https://integrationapi.net/rest/Hlr/SendBulk?SessionID=<Идентификатор сессии>&DestinationAddresses=
↪<Номер получателя>&Validity=<Время жизни сообщения>
```

Пример:

```
https://integrationapi.net/rest/Hlr/SendBulk?SessionID=4E1C44388AB54B38B097C17D5F949ECA4005&
↪DestinationAddresses=+70000000001&DestinationAddresses=80000000002&Validity=5
```

Параметры POST-запроса на отправку HLR-сообщения на несколько номеров

Параметр	Тип данных	Описание	Обязательный
Login	String	Логин,полученный при регистрации	Да
Password	String	Пароль, соответствующий логину	Да
DestinationAddresses	String	Номера получателей сообщения, в международном формате: код страны и код сети плюс номер телефона. Максимум 200 номеров Пример формата: 79031234567;+79031234567;89031234567	Да
Validity	Int	Время жизни сообщения (в минутах)	Да

Ниже приведен пример ответа:

```
HTTP/1.1 200 OK
Cache-Control: private
Connection: Keep-Alive
Content-Type: application/json; charset=utf-8
["703112828852109312", "703112828852109313"]
```

16.2 Запрос статуса по HLR-сообщениям (GET)

Сервис возвращает статус отправленного HLR-сообщения в соответствии со значениями параметров, передаваемых сервису в GET-запросе следующего формата:

```
https://integrationapi.net/rest/v2/Hlr/State?Login=<Логин>&Password=<Пароль>&messageIds=
↪<Идентификаторы сообщений>
```

или вместо пары логинпароль можно передавать SessionID

```
https://integrationapi.net/rest/Hlr/State?SessionID=<Идентификатор сессии>&messageIds=
↪<Идентификаторы сообщений>
```

Пример:

```
https://integrationapi.net/rest/Hlr/State?SessionID=4E1C44388AB54B38B097C17D5F949ECA4005&
↪messageIds=703112828852109312&messageIds=703112828852109313
```

Поля ответа на статус сообщения:

Параметр	Тип данных	Описание
DestinationAddress	String	Номер получателя сообщения, в международном формате: код страны и код сети плюс номер телефона.
StateCode	Int	Код статуса сообщения
StateDesc	String	Краткое описание ошибки
Country	String	Страна абонента
MobileOperator	String	Мобильный оператор абонента

Ниже приведен пример ответа:

```
[{
     "messageId" : "703112828852109314"
     "DestinationAddress": "+79001234567",
     "StateCode" : 1,
     "StateDesc" : "Отправляется",
     "Country" : "Россия",
     "MobileOperator" : "ОАО Вымпел-Коммуникации(Москва)",
}]
```

Статусы сообщений

StateCode	Описание
0	Отправляется
1	Абонент доступен
2	Абонент временно недоступен
3	Абонент не существует
4	Ошибка запроса
99	Неизвестно

Глава 17

Devino AddressBook Api

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

17.1 Описание

Сервис представляет собой удобный интерфейс для управления списками контактов адресной книги и получения списка отписавшихся от рассылок.

17.1.1 Список методов

Набор методов для работы с группами контактов - ContactGroups.

Ресурс	Метод	Описание
/ContactGroups	GET	Получение списка всех групп
/ContactGroups/{ContactGroupId}	GET	Получение группы с подробной информацией
/ContactGroups	POST	Добавление группы
/ContactGroups/{ContactGroupId}	PUT	Редактирование группы
/ContactGroups/{ContactGroupId}	DELETE	Удаление группы

Набор методов для импорта контактов - ContactsImport.

Ресурс	Метод	Описание
/ContactsImport	POST	Импорт контактов
/ContactsImport/{ImportId}/Progress	GET	Запрос прогресса добавления контактов в базу
/ContactsImport/{ImportId}/Duplicates	DELETE	Удаление дубликатов из завершённого импорта

Набор методов для работы с контактами - Contacts.

Ресурс	Метод	Описание
/Contacts?Query={Key}	GET	Получение диапазона контактов по телефону или почте
/Contacts/{ContactId}	GET	Получение контакта с подробной информацией
/Contacts	POST	Создание контакта
/Contacts/{ContactId}	PATCH	Редактирование контакта
/Contacts/{ContactId}	DELETE	Удаление контакта

Набор методов для работы с отписавшимися - Unsubscribed.

Ресурс	Метод	Описание
/Unsubscribed?TaskId={TaskId}	GET	Получение диапазона отписавшихся по рассылке
/Unsubscribed	GET	Получение диапазона отписавшихся

Запрос к API состоит из следующих элементов:

- Основного URL запроса: https://integrationapi.net/addressbook/v2
- Ресурса, например: /Contacts
- Параметров запроса в кодировке UTF-8.

Сервис позволяет передавать параметры в следующих форматах: XML, JSON, JSV, CSV.

17.1.2 Авторизация

Для доступа к сервису необходимо пройти авторизацию. Сервис поддерживает базовую авторизацию через заголовок Authorization (https://en.wikipedia.org/wiki/Basic_access_authentication):

```
Authorization: Basic dGVzdGVyOjExMTExMQ==
```

В заголовке необходимо передать логин и пароль из ЛК (https://my.devinotele.com) в формате login:password в base64 кодировке.

Ответ API состоит из 2х частей:

1. Код с описанием - эта часть присутствует во всех ответах.
2. Result, специфичный для каждого запроса. Может отсутствовать.

```
{
    "Result":{
        ...
    },
    "Code": "validation_error",
    "Description": "user not found"
}
```

Набор кодов ограничен, а набор описаний зависит от конкретного метода. Код можно использовать для проверки статуса запроса, а описание предназначено для диагностики возможных проблем. Описание может быть изменено в новой версии API без предупреждения о нарушении обратной совместимости. Набор кодов также может быть расширен.

17.1.3 Локализация

В поле Description может возвращаться локализованная строка с текстом ошибки. Для этого необходимо передать заголовок Accept-Language с нужным языком. В текущей версии поддерживаются русский и английский языки.

По умолчанию, если заголовок не передан или язык не найден среди доступных возвращаются ответы на английском.

```
Accept-Language: ru-RU
```

17.1.4 Запрос диапазонов

Некоторые запросы предполагают возвращение только части данных. Для таких запросов необходимо передавать специальный заголовок:

```
Range: items=1-100
```

Оба предела диапазона включаются. Запросы, для которых нужен данный заголовок:

- /Unsubscribed
- /Contacts?Query={Key}

При отсутствии заголовка данные запросы возвращают ошибку validation_error с http кодом 416 RequestedRangeNotSatisfiable.

17.1.5 Список кодов ответов

Код	Http StatusCode	Расшифровка
validation_error	400 - 404, 416	Ошибки валидации, авторизации и доступа
ok	200, 201, 206	Запрос выполнен успешно
internal_error	500	Внутренняя ошибка сервиса, можно повторить запрос позже

17.2 Работа с группами контактов

17.2.1 ContactGroups GET (all)

```
https://integrationapi.net/addressbook/v2/ContactGroups
```

Метод возвращает список всех групп контактов пользователя. Возвращаемый результат - список объектов типа ContactGroupDto.

Возвращаемый результат - список записей ContactGroupDto

Параметр	Тип данных	Описание
ContactGroupId	int	Идентификатор группы
Name	string	Имя группы
Description	string	Описание группы
CreatedDate	DateTime	Дата создания
ContactsCount	int	Количество контактов в группе

Пример ответа:


```
{
    "Result":[
        {
            "ContactGroupId": 252,
            "Name": "snuk",
            "Description": "",
            "CreatedDate": "/Date(1426504354337-0000)/",
            "ContactsCount": 3
        },
        {
            "ContactGroupId": 331,
            "Name": "zzzzzzz04.02.2016 16:49:35",
            "Description": "AB api intgration test",
            "CreatedDate": "/Date(1454582978323-0000)/",
            "ContactsCount": 0
        }
    ],
    "Code": "ok",
    "Description": "ok"
}
```


17.2.2 ContactGroups GET

```
https://integrationapi.net/addressbook/v2/ContactGroups/{ContactGroupId}
```

Метод возвращает группу по идентификатору. В качестве Result возвращается объект ContactGroupDto, описание см. выше.

Параметры запроса

Параметр	Тип данных	Описание	Обязательнй
ContactGroupId	int	Идентификатор группы (предаётся в url)	Да

Пример ответа:


```
{
    "Result":{
        "ContactGroupId": 332,
        "Name": "new group",
        "Description": "best new group",
        "CreatedDate": "/Date(1454587881407-0000)/",
        "ContactsCount": 0
    },
    "Code": "ok",
    "Description": "ok"
}
```


17.2.3 ContactGroups POST

```
https://integrationapi.net/addressbook/v2/ContactGroups
```

Метод добавляет новую группу контактов. Если группа была успешно добавлена, возвращается код «ok» и http код 201. Метод возвращает идентификатор группы ContactGroupId в качестве Result.

Параметры запроса

Параметр	Тип данных	Описание	Обязательнй
Name	string	Имя группы	Да
Description	string	Описание группы	Нет

Пример запроса:

```
{
    "Name":"new group",
    "Description":"best group"
}
```

Пример ответа:

```
{
"Result": 332,
"Code": "ok",
"Description": "ok"
}
```

17.2.4 ContactGroups PUT

```
https://integrationapi.net/addressbook/v2/ContactGroups/{ContactGroupId}
```

Метод обновляет имя и описание группы, затирая старые значения, возвращается только стандартный ответ, без поля Result.

Параметры запроса

Параметр	Тип данных	Описание	Обязательнй
ContactGroupId	int	Идентификатор группы (передаётся в url)	Да
Name	string	Имя группы	Да
Description	string	Описание группы	Нет

Пример запроса:

```
{"Name":"new group","Description":"best new group"}
```

Пример ответа:

```
{
    "Code": "ok",
    "Description": "ok"
}
```

17.2.5 ContactGroups DELETE

```
https://integrationapi.net/addressbook/v2/ContactGroups/{ContactGroupId}
```

Метод удаляет группу, возвращается только стандартный ответ, без поля Result.

Параметры запроса:

Параметр	Тип данных	Описание	Обязательнй
ContactGroupId	int	Идентификатор группы (передаётся в url)	Да

Пример ответа:

```
{
    "Code": "ok",
    "Description": "ok"
}
```

17.3 Работа с контактами

17.3.1 ContactsImport POST

```
https://integrationapi.net/addressbook/v2/ContactsImport/
```

Метод валидирует пачку контактов и добавляет во внутреннюю очередь для добавления в базу. Если контакты были успешно добавлены в очередь, возвращается код «ok» и http код 201. Метод возвращает счётчики провалидированных контактов в качестве Result.

Валидируются:

- наличие группы, в которую импортируются контакты
- максимальное количество контактов - не более 10 000

Контакты валидируются на:

- наличие хотя бы одного поля - номер телефона или email адрес
- валидность номера телефона, если он передан (непустая строка, состоящая из цифр, длиной от 8 до 15 символов, в начале может быть «+»)
- валидность email адреса, если он передан
- длина полей FirstName, MiddleName и LastName не должна превышать 100 символов, для ExtraField1 и ExtraField2 - ограничение 700 символов
- пол, если передано значение отличное от 1 и 2, будет проставлено 3

Параметры запроса

Параметр	Тип данных	Описание	Обязательнй
ContactGroupId	int	Идентификатор группы, в которую импортируются контакты	Да
ImportId	GUID	Идентификатор импорта	Нет
ConvertDirectPhones	bool	Преобразовывать ли прямые номера в федеральные	Нет
FillGender	bool	Заполнять ли автоматически пол контакта на основе его ФИО	Нет
Contacts	ContactDto[]	Список импортируемых контактов	Да

ContactDto

Параметр	Тип данных	Описание	Обязательнй
PhoneNumber	string	Номер телефона	см. описание
Email	string	Email адрес	см. описание
FirstName	string	Имя	Нет
MiddleName	string	Отчество	Нет
LastName	string	Фамилия	Нет
Gender	int	Пол (1 - м, 2 - ж, 3 - неизвестно)	Нет
DateOfBirth	DateTime	Дата рождения	Нет
ExtraField1	string	Дополнительное поле №1	Нет
ExtraField2	string	Дополнительное поле №2	Нет
MergeFields	Dictionary	Пользовательский словарь «ключ-значение»	Нет

Возвращаемый результат

Параметр	Тип данных	Описание
ImportId	GUID	Идентификатор импорта - генерируется автоматически, если не был передан
ValidContacts	int	Количество валидных контактов
RejectedContacts	RejectedContactDto[]	Список невалидных контактов
InvalidPhoneNumbers	string[]	Список невалидных номеров телефонов
InvalidEmails	string[]	Список невалидных email адресов

RejectedContactDto

Параметр	Тип данных	Описание
Contact	ContactDto	Контакт
ErrorCode	ContactValidationCode	Тип ошибки валидации
ErrorDescription	string	Описание ошибки валидации
DuplicatesCount	int	Количество дублирований в начальном запросе

ContactValidationCode

Текст	Число	Описание
Ok	0	Значение по умолчанию
Duplicate	1	Дубликат
NoPhoneNoEmail	2	Не передан телефон или email адрес
InvalidPhone	3	Невалидный номер телефона
InvalidEmail	4	Невалидный email адрес
InvalidField	5	Невалидное поле - ФИО или ExtraField1, ExtraField2

Пример запроса

```
{
    "ContactGroupId" : 9731,
    "ImportId" : "50210B41-1C4E-4E48-9837-FB382A9BAE01",
    "ConvertDirectPhones" : true,
    "FillGender" : false,
    "Contacts" :[
        {
            "PhoneNumber": "",
            "LastName": "Ivanov",
            "FirstName": "Ivan",
            "Email": "ivanov@ivanov.com",
            "DateOfBirth": "/Date(1454533200000-0000)/"
        },
        {
            "PhoneNumber": "+79001234567",
        }
    ]
}
```

Пример ответа

```
{
    "Result":{
        "ImportId" : "50210B41-1C4E-4E48-9837-FB382A9BAE01",
        "ValidContacts": 2,
        "RejectedContacts":[],
        "InvalidPhoneNumbers":[],
        "InvalidEmails":[]
    },
    "Code": "ok",
    "Description": "ok"
}
```

17.3.2 ContactsImport Progress GET

```
https://integrationapi.net/addressbook/v2/ContactsImport/{ImportId}/Progress
```

Метод возвращает прогресс добавления контактов в базу по идентификатору импорта, который передаётся в url.

Возвращаемый результат

Параметр	Тип данных	Описание
Uploaded	int	Общее количество загруженных контактов через ContactsImport POST
Processed	int	Количество контактов, добавленных в базу

Пример ответа

```
{
    "Result":{
        "Uploaded" : 1123456,
        "Processed": 1100000
    },
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "Code": "ok",
    "Description": "ok"
}
```

17.3.3 ContactsImport Duplicates DELETE

```
https://integrationapi.net/addressbook/v2/ContactsImport/{ImportId}/Duplicates
```

Метод удаляет дубликаты по идентификатору импорта, который передаётся в url.

Параметры запроса

Параметр	Тип данных	Описание	Обязательнй
ContactGroupId	int	Идентификатор группы, в которую импортированы контакты	Да
ImportId	GUID	Идентификатор импорта	Да
Field	DuplicatesCheckField	Ключевое поле для проверки - email или номер телефона	Нет
Scope	DuplicatesCheckScope	Область проверки на дубли	Нет
Groups	int[]	Список групп для проверки	Нет

Перечисление DuplicatesCheckField

Текст	Число	Описание
No	0	Не проверяем на дубли
PhoneNumber	1	Проверяем по номеру телефона
Email	2	Проверяем по email адресу

Перечисление DuplicatesCheckScope

Текст	Число	Описание
No	0	Не проверяем на дубли
Import	1	Проверяем в рамках импорта
Groups	2	Проверяем в переданных группах

DuplicatesCheckScope является битовой маской, можно передавать сочетания значений, т.е. можно передать 3 и проверка будет выполнена в рамках импорта и в переданных группах.

Возвращаемый результат

Параметр	Тип данных	Описание
InCurrentGroup	int	Количество дублей в текущей группе
InOtherGroups	int	Количество дублей в других группах
InCurrentImport	string[]	Список дублей в текущем импорте

Пример запроса

```
{
    "ContactGroupId" : 9731,
    "ImportId" : "50210B41-1C4E-4E48-9837-FB382A9BAE01",
    "Field" : "Email",
    "Scope" : 3,
    "Groups" : [123, 345, 9731]
}
```

Пример ответа

```
{
    "Result":{
        "InCurrentGroup" : 34,
        "InOtherGroups": 55,
        "InCurrentImport": ["79251234567"]
    },
    "Code": "ok",
    "Description": "ok"
}
```

17.3.4 Contacts GET (query)

```
https://integrationapi.net/addressbook/v2/Contacts?Query={Key}
```

Метод возвращает контакты по ключу, в качестве ключа может выступать email или номер телефона. Возвращаемый результат - список объектов типа ContactDto.

Также необходимо задать диапазон возвращаемых записей.

Параметры запроса:

Параметр	Тип данных	Описание	Обязательнй
Query	string	Ключ для поиска контактов (передаётся в url)	Да

Возвращаемый результат - список записей ContactDto

Параметр	Тип данных	Описание
ContactId	long	Идентификатор контакта
PhoneNumber	string	Номер телефона
Email	string	Email адрес
FirstName	string	Имя
MiddleName	string	Отчество
LastName	string	Фамилия
Gender	int	Пол (1 - м, 2 - ж, 3 - неизвестно)
DateOfBirth	DateTime	Дата рождения
ExtraField1	string	Дополнительное поле №1
ExtraField2	string	Дополнительное поле №2
ContactGroupId	int	Идентификатор группы, в которой находится контакт
MergeFields	Dictionary	Пользовательский словарь в виде «ключ-значение»

Пример ответа:

```
{
    "Result":[{
        "ContactId": 1,
        "PhoneNumber": "",
        "LastName": "Snuk",
        "MiddleName": "Snuk",
        "FirstName": "Snuk",
        "Email": "xx@gmail.com",
        "Gender": 3,
        "DateOfBirth": "/Date(1454533200000-0000)/",
        "ExtraField1": "ddddddddddddddddd",
        "ExtraField2": "ccccccccccccccc",
        "ContactGroupId": 252
    },
    {
        "ContactId": 100005,
        "PhoneNumber": "",
        "LastName": "sdfsdfdsf",
        "MiddleName": "sfddsf",
        "FirstName": "sdfdsfds",
        "Email": "yy@list.ru",
        "Gender": 3,
        "ContactGroupId": 252
    }],
    "Code": "ok",
    "Description": "ok"
}
```


17.3.5 Contacts GET

```
https://integrationapi.net/addressbook/v2/Contacts/{ContactId}
```

Метод возвращает контакт по идентификатору, в качестве Result возвращается объект ContactDto, описание см. выше.

Параметры запроса:

Параметр	Тип данных	Описание	Обязательнй
ContactId	int	Идентификатор контакта (передаётся в url)	Да

Пример ответа:


```
{
    "Result":{
        "ContactId": 1,
        "PhoneNumber": "",
        "LastName": "Snuk",
        "MiddleName": "Snuk",
        "FirstName": "Snuk",
        "Email": "xx@gmail.com",
        "Gender": 3,
        "DateOfBirth": "/Date(1454533200000-0000)/",
        "ExtraField1": "ddddddddddddddddd",
        "ExtraField2": "ccccccccccccccc",
```


(continues on next page)

(продолжение с предыдущей страницы)

```
        "ContactGroupId": 252
    },
    "Code": "ok",
    "Description": "ok"
}
```

17.3.6 Contacts POST

```
https://integrationapi.net/addressbook/v2/Contacts
```

Метод создаёт контакт. Если контакт был успешно создан, возвращается код «ok» и http код 201. В качестве Result возвращается идентификатор контакта.

Валидируются:

- наличие хотя бы одного поля - номер телефона или email адрес
- валидность номера телефона, если он передан
- валидность email адреса, если он передан
- длина полей FirstName, MiddleName и LastName не должна превышать 100 символов, для ExtraField1 и ExtraField2 - ограничение 700 символов
- пол, если передано значение отличное от 1 и 2, будет проставлено 3
- наличие группы, в которую добавляется контакт

Параметры запроса:

Параметр	Тип данных	Описание	Обязательнй
PhoneNumber	string	Номер телефона	см. описание
Email	string	Email адрес	см. описание
FirstName	string	Имя	Нет
MiddleName	string	Отчество	Нет
LastName	string	Фамилия	Нет
Gender	int	Пол (1 - м, 2 - ж, 3 - неизвестно)	Нет
DateOfBirth	DateTime	Дата рождения	Нет
ExtraField1	string	Дополнительное поле №1	Нет
ExtraField2	string	Дополнительное поле №2	Нет
MergeFields	Dictionary	Пользовательский словарь «ключ-значение»	Нет
ContactGroupId	int	Идентификатор группы, в которой находится контакт	Да

Пример запроса:

```
{
    "PhoneNumber": "",
    "LastName": "Snuk",
    "MiddleName": "Snuk",
    "FirstName": "Snuk",
```

(continues on next page)

(продолжение с предыдущей страницы)

```
    "Email": "zzz@gmail.com",
    "Gender": 3,
    "DateOfBirth": "/Date(1454533200000-0000)/",
    "ExtraField1": "ddddddddddddddddd",
    "ExtraField2": "ccccccccccccccc",
    "ContactGroupId": 252
}
```

Пример ответа:

```
{
    "Result": 100013,
    "Code": "ok",
    "Description": "ok"
}
```

17.3.7 Contacts PATCH

```
https://integrationapi.net/addressbook/v2/Contacts/{ContactId}
```

Метод обновляет контакт. (PATCH по идеологии аналогичен PUT, с той лишь разницей, что PUT полностью заменяет ресурс, а PATCH меняет только те параметры, которые переданы.) Валидация идентична методу Contacts POST, исключается только проверка наличия группы, так как её менять нельзя. Возвращается только стандартный ответ, без поля Result.

Параметры запроса:

Параметр	Тип данных	Описание	Обязательнй
ContactId	long	Идентификатор контакта (предаётся в url)	Да
PhoneNumber	string	Номер телефона	см. описание
Email	string	Email адрес	см. описание
FirstName	string	Имя	Нет
MiddleName	string	Отчество	Нет
LastName	string	Фамилия	Нет
Gender	int	Пол (1 - м, 2 - ж, 3 - неизвестно)	Нет
DateOfBirth	DateTime	Дата рождения	Нет
ExtraField1	string	Дополнительное поле №1	Нет
ExtraField2	string	Дополнительное поле №2	Нет
MergeFields	Dictionary	Пользовательский словарь «ключ-значение»	Нет

Пример запроса:

```
{
    "PhoneNumber": "",
    "LastName": "Snuk",
    "MiddleName": "Snuk",
    "FirstName": "Snuk",
    "Email": "zz@gmail.com",
    "Gender": 3,
    "DateOfBirth": "/Date(1454533200000-0000)/",
    "ExtraField1": "ddddddddddddddddd",
    "ExtraField2": "ccccccccccccccc"
}
```

Пример ответа:

```
{
    "Code": "ok",
    "Description": "ok"
}
```

17.3.8 Contacts DELETE

```
https://integrationapi.net/addressbook/v2/Contacts/{ContactId}
```

Метод удаляет контакт, возвращается только стандартный ответ, без поля Result.

Параметры запроса:

Параметр	Тип данных	Описание	Обязательнй
ContactId	int	Идентификатор контакта (передаётся в url)	Да

Пример ответа:

```
{
    "Code": "ok",
    "Description": "ok"
}
```

17.4 Работа с отписавшимися

17.4.1 Unsubscribed GET

```
https://integrationapi.net/addressbook/v2/Unsubscribed?TaskId={TaskId}
```

Метод возвращает список отписавшихся от email рассылок. Можно получить список отписавшихся от определённой рассылки, для этого предусмотрен параметр taskId. Возвращаемый результат - список объектов типа UnsubscribedDto. Также необходимо задать диапазон возвращаемых записей.

Параметры запроса:

Параметр	Тип данных	Описание	Обязательнй
TaskId	int	Идентификатор рассылки	Нет

Возвращаемый результат - список записей UnsubscribedDto

Параметр	Тип данных	Описание
Email	string	Email адрес
DateTime	DateTime	Дата и время добавления
Reason	string	Причина отписки
TaskId	int	Идентификатор рассылки

Пример ответа:

```
{
    "Result":[{
        "Email": "generated_1@generated.com",
        "DateTime": "/Date(1439219917910-0000)/",
        "Reason": "Другая причина",
        "TaskId": 133696
    },
    {
        "Email": "generated_11@generated.com",
        "DateTime": "/Date(1439219917910-0000)/",
        "Reason": "Скучные рассылки у вас",
        "TaskId": 133696
    }],
    "Code": "ok",
    "Description": "ok"
}
```

Глава 18

Сервис двухфакторной аутентификации

18.1 Описание

Сервис предоставляет возможность абоненту подтверждать свой номер телефона посредством ввода специального короткого кода, который был отправлен ему на мобильный телефон через платформу Devino.

Предупреждение: Внимание! Для использования данного сервиса вам нужно обратиться в техническую поддержку support@devinotele.com для произведения необходимых настроек.

Запрос к API состоит из следующих элементов:

- Основного URL запроса: https://phoneverification.devinotele.com/
- Ресурса, например: /GenerateCode
- Параметров запроса в кодировке UTF-8. Сервис реализован с использованием технологии ServiceStack , что позволяет передавать параметры в следующих форматах: XML, JSON, JSV, CSV.
- ApiKey, переданного в заголовке запроса X-ApiKey, который можно получить у менеджера

Ответ API состоит из двух частей:

Код с описанием - эта часть присутствует во всех ответах. В качестве описания возвращается описание ошибки.

```
{
        "Code": "0",
        "Description": "Code Sent"
}
```

Все коды отличные от 0, считаются ошибками, информация о которых, содержится в поле Description.

18.2 Список методов

Ресурс	Метод	Описание
/GenerateCode	POST	Запрос на отправку короткого кода
/CheckCode	POST	Запрос на подтверждение кода

18.3 Отправить код абоненту

/GenerateCode POST

https://phoneverification.devinotele.com/GenerateCode

Метод генерирует и отправляет код на указанный в запросе номер телефона на основе настроек пользователя. Опционально можно передать параметр IMSICode. В этом случае сервис проверяет текущий IMSICode абонента, и сравнивает с переданным. Отправка кода осуществляется только в случае, если переданный IMSI код, совпадает с актуальным IMSI кодом абонента. Эта функция подключается отдельно.

Параметры запроса:

Параметр	Тип данных	Обязательность	Описание
DestinationNumber	String	Да	Номер получателя кода
IMSICode	String	Нет	IMSI код абонента

Пример запроса:

```
{
        "DestinationNumber":"79169492211",
        "IMSICode": "D92C2B0A1D5C64BBA8DD8BCE43C4BA11"
}
```

Пример ответа:

```
{
        "Code": "0",
        "Description": "Code Sent"
}
```

Список кодов ответов:

Код	Description	Расшифровка
0	Code Sent	Успешный статус работы метода
1	Invalid ApiKey	Передан неизвестный ApiKey
2	Invalid Phone	Передан невалидный номер получателя кода
3	Limit requsts by time is over	Превышен лимит запросов в минуту
4	Internal Server Error	Внутренняя ошибка сервиса
5	Error sending message	Ошибка отправки СМС с кодом
6	Invalid IMSI Code	Некорректный IMSI код
7	Code Exist	Для данного номера уже есть неиспользованный код, у которого не кончилось время жизни

18.4 Проверить код от абонента

/CheckCode POST

https://phoneverification.devinotele.com/CheckCode

Метод проверяет, совпадает ли последний отправленный клиенту код, с переданным в запросе, а также, не кончилось ли время жизни кода. Параметры запроса:

Параметр	Тип данных	Обязательность	Описание
DestinationNumber	String	Да	Номер получателя кода
Code	String	Да	Код, полученный абонентом на телефон

Пример запроса:

```
{
        "DestinationNumber":"79169492283",
        "Code": "17565"
}
```

Пример ответа:

```
{
        "Code": 0,
        "Description": "Valid Code"
}
```

Список кодов ответов:

Код	Description	Расшифровка
0	Valid Code	Успешный статус работы метода
1	Invalid ApiKey	Передан неизвестный ApiKey
2	Invalid Phone	Передан невалидный номер получателя кода
3	Limit requsts by time is over	Превышен лимит запросов в минуту
4	Internal Server Error	Внутренняя ошибка сервиса
5	Code not Found	Переданный код не найден
6	Code Expired	Код верен, но кончилось его время жизни

Глава 19

Сервис аутентификации IMSI

19.1 Общее описание сервиса

Компания Devino Telecom предоставляет финансовым учреждениям услуги по управлению финансами с помощью мобильных технологий. Используя сервисы, предоставляемые Devino Telecom, у клиентов Банка существует возможность управлять своими финансами с мобильного телефона.

В 2009 году в рамках обеспечения безопасности передачи разовых паролей, используемых для входа в интернет-банк, компанией была предложена и реализована технология проверки подлинности SIM-карт посредством проверки уникального кода IMSI. Благодаря внедренной технологии, при передаче разовых паролей удалось полностью исключить мошенничество с подменой SIM-карт злоумышленниками.

Предупреждение: Внимание! Для использования данного вида интеграции необходимо обратиться к своему менеджеру, либо в техническую поддержку support@devinotele.com для настройки доступа.

19.2 Алгоритм аутентификации IMSI на стороне Банка

Логика хранения и проверки реализована в банковских информационных системах.

Devino Telecom предоставляет Банку SOAP Web-сервис, позволяющий запрашивать IMSI SIM-карты абонента. При этом Web-сервис установлен на технологической площадке Devino Telecom, а программное обеспечение, установленное на технических мощностях Банка, осуществляет удаленные вызовы сервиса.

Реализация данного алгоритма взаимодействия дает Банку следующие возможности:

- Запрашивать IMSI SIM-карты абонента по номеру мобильного телефона (адресу) абонента. После получения первого значения IMSI, Банк сохраняет его значение в своей базе и считает эталон-

ным значением IMSI для текущего клиента. Перед отправкой сообщений, требующих проверки IMSI, Банк обращается в свою БД для сравнения значения IMSI, полученного перед отправкой, и эталонного значения и, в зависимости от результата сравнения, реализует бизнес-логику.

- Запрашивать данные о коммутаторе, на котором зарегистрирован абонент по номеру мобильного телефона (адресу) абонента. Данный запрос позволяет определить реальное местоположение клиента и сравнить его с заявленным местоположением, например, на период отпуска.
- Запрашивать статус сервиса Devino Telecom. Используется для мониторинга доступности сервиса со стороны Банка.

Все виды операций в рамках взаимодействия между Банком и Devino Telecom инициируются Банком.

19.3 Алгоритм аутентификации IMSI на стороне Devino Telecom

Логика хранения и проверки реализована на стороне платформы Devino Telecom.

По SMPP-подключению Devino Telecom получает от Банка сообщение для абонента, в этот момент мы отправляем сообщение и асинхронно запрашиваем эталонное значение IMSI у оператора. Полученное значение IMSI добавляется в таблицу базы данных и считается эталонным.

Впоследствии при поступлении сообщения для этого абонента мы синхронно проверяем значение IMSI и, в случае совпадения с эталонным, отправляем сообщение абоненту. Если значение не совпадает, то абоненту отправляется сообщение, например, «Ваша SIM-карта не прошла проверку, обратитесь в Банк».

В личном кабинете Банка есть интерфейс, который позволяет просматривать базу абонентов, запускать процесс получения эталонного IMSI, детально просматривать запросы IMSI и их результат. Там же есть возможность выставить признак, проверять/не проверять IMSI для данного абонента.

В случаях, когда эталонное значение мы не можем получить, например, если нет технической возможности получать IMSI абонентов определенного оператора, а также в случае недоступности абонента и сбоя на оборудовании оператора, то в зависимости от полученных ошибок, решается продолжать ли попытки (асинхронные) получить IMSI или отправлять сообщение без проверки.

Банк может реинициализировать абонента (запросить новое эталонное значение IMSI) посредством интерфейса личного кабинета или командой, переданной в e-mail на специальный адрес.

19.4 Сценарий использования «Проверка IMSI»

При проверке есть **3 возможные** ситуации: * Новый IMSI записывается базу (т.е. пока сравнивать значение IMSI не с чем, самый первый запрос). * Проверка IMSI успешна (полученное значение IMSI совпадает с тем, что записано в базе). * Проверка IMSI не успешна (полученное значение IMSI не совпадает с тем, что записано в базе).

Для п.1 и п.2 отправляется SMS по операции.

Для п.3 SMS по операции блокируется, отправляется SMS с текстом Банка, например, «Уважаемый клиент, операция не проведена, обратитесь в банк по телефону... »

Описание:

- Новый IMSI записывается в базу (т.е. пока сравнивать значение IMSI не с чем, самый первый запрос)

  >>В БД для номера телефона ничего нет. Просто записывается новое значение и считается, что это значение правильное.

- Проверка IMSI успешна (полученное значение IMSI совпадает с тем, что записано в базе).

  >>Перед отправкой SMS Банк проверяет IMSI и значение в БД совпадает с тем, что было получено. Проверка пройдена – SMS по операции отправляется.

- Проверка IMSI не успешна (полученное значение IMSI не совпадает с тем, что записано в базе). Клиент звонит в Банк, т.к. получил SMS о том, что операция невозможна в связи с непройденной проверкой IMSI.

  >>Перед отправкой SMS Банк проверяет IMSI и значение в БД не совпадает с тем, что было получено. Новое значение полученного IMSI не сохранено в БД и теперь Банк должен решить, верить ли этому значению или нет. Если Банк верит (после верификации клиента), то надо нажать «Перепроверить», после этого будет сформирован еще один запрос на IMSI и полученное значение будет сохранено в БД как эталонное.

  *Если Банк не верит (верификация клиента не прошла) – никаких действий не требуется. Если Банк знает, что он будет верить всегда и проверять не надо – то он нажимает «Отключить проверку, всегда отправлять».

  *Если Банк знает, что он никогда не будет верить и проверять не надо – это IMSI-телефон мошенника, то необходимо нажать «Отключить проверку, всегда отклонять».